GC27-6999-0
File No. S360/S370-09

Systems

Introduction to Programming the IBM 3270

IBM

First Edition (February 1973)

Changes made to this first edition will be reported in subsequent revisions or technical newsletters.

Before using this publication in connection with the operation of IBM systems, refer to the latest IBM System/360 or System/370 SRL Newsletter, Order No. GN20-0360 for editions that are applicable and current.

Copies of this and other IBM publications can be obtained through your IBM branch office.

A form for reader's comments appears at the back of this publication. Address any additional comments concerning the contents of this publication to IBM Corporation, Programming Publications, Department 636, Neighborhood Road, Kingston, New York 12401.

Who This Book Is For

This book is for programmers and other people who need to know what's involved in programming the IBM 3270 Information Display System.

For those programmers who plan and code the messages seen on 3270 displays, this book may be the only book required.

For those programmers who also write the access method macro instructions or other I/O instructions, this book is to be used in conjunction with the appropriate access method or IBM Program Product publications.

How This Book Is Organized

This book is divided into these sections:

1: SCREEN DESIGN
Introduces important 3270 concepts. Shows an example of what a 3270 display message might look like, what coding elements are required to write this message in your program, and how terminal operator input might be handled.

2: SCREEN MANAGEMENT
Suggests macro definitions and programming routines that might be written to encode and decode messages to and from the display.

3: DEVICE MANAGEMENT
Suggests including I/O operations (reading, writing, error recovery) in a module separate from message formatting. Contains flowcharts to aid in writing error recovery routines for use with BTAM.

Other Books You May Need

As a general introduction to the 3270:

An Introduction to the IBM 3270 Information Display System, GA27-2739

To understand how the terminal operator will see the 3270:

Operator's Guide for IBM 3270 Information Display System, GA27-2742
IBM 3270 Information Display System Problem Determination Guide, GA27-2750

As a reference on how the 3270 works:

IBM 3270 Information Display System Component Description, GA27-2749

Suggested programming tools:

A green booklet: IBM 3270 Information Display System Reference Summary, GX20-1878
Panel layout sheets: IBM 3270 Information Display System Layout Sheet, GX27-2951

If you are using BTAM:

IBM 2260 BTAM and 2260 GAM to IBM 3270 BTAM Conversion Guide, GC27-6975
IBM System/360 Disk Operating System Basic Telecommunications Access Method, GC30-5001
DOS Programming Supplement for the 3270 Information Display System, GC27-6977 (applicable to DOS Release 26 only)
DOS Version 4 BTAM, GC27-6978
IBM System/360 Operating System Basic Telecommunications Access Method, GC30-2004
OS/VS BTAM, GC27-6980

If you are using TCAM:

Planning for TCAM with the IBM 3270 Information Display System, GC30-2021
OS TCAM Programmer's Guide and Reference Manual, GC30-2024
OS TCAM User's Guide, GC30-2025

If you are using an IBM Program Product, see the appropriate Program Product publications.

CONTENTS

ILLUSTRATIONS

FIELD CONCEPT

People dealing with information see it as a collection of individual elements. For example, what we know about John Smith's employment may be a collection of individual elements: his name, serial number, location, and date of hire. The size of the element is the amount of data required to convey useful information. You do not think of "J" and "O" and "H" and "N" as useful individually, but collectively, as the name JOHN. You do not think of JOHNSMITH963981BOSTON070262 as being useful collectively, but see the elements individually: name: JOHN SMITH, serial number: 963981, location: BOSTON, date of hire: 07/02/62.

Each data element has its own characteristics. In this example, the serial number is 6 numeric digits and varies from employee to employee. The word "NAME" is 4 characters, is alphabetic, is all uppercase, and does not change. When people record these elements of data on paper they take on such additional characteristics as position (where on the sheet of paper the item is written), color (what ink or media is used), size of the letters, and writing style.

In the past, when information was handled by a data processing device it was generally handled as an artificial entity called a record. The contents and characteristics of a record were primarily determined by device requirements and little or no attention was given to the individual information elements. Data processing users had to adjust their thought pattern to conform to the machine requirements.

The IBM 3270 Information Display System recognizes that people deal with individual units of information. The system has been designed to conform to human needs and requirements and it enables you to deal with data by individual elements or "fields," each with its own individual characteristics.

You may describe data to the 3270 on a field basis and specify the characteristics or "attributes" of each individual field. The 3270 then provides program and data control based on your individual field definitions.

How Fields are Defined

Each data field is established by writing a field attribute control code, or attribute character, as the first position of the field. A field is defined as the attribute character, plus all the data following it up to the next attribute character. The placement of attribute characters defines the field lengths, and the content of the attribute characters defines the other field characteristics. In the following examples, the symbol ☐ designates an attribute character.

All the characters in a field, except the attribute character itself, assume identical characteristics based on the specifications within the attribute character. In Figure 1 the characteristics of the field NAME: are controlled by the attribute [1] and the field NAME: is terminated by the attribute [2]. The placement of attributes controls the length of the fields.

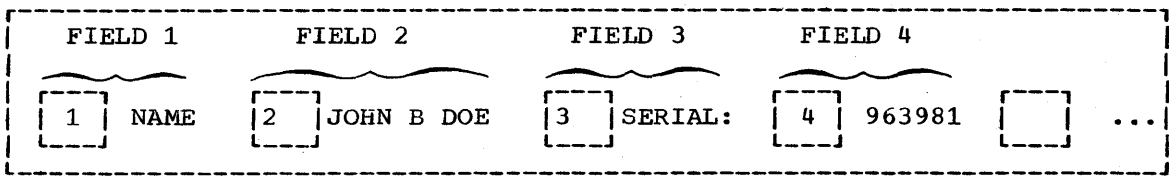


Figure 1. Example of 4 fields and attribute characters.

Field attributes may be modified or removed by a 3270 program. Removal of the attribute character [2] causes NAME: JOHN DOE to beconsidered by the 3270 as a single field. Changing the content of the attribute [3] alters the characteristics of SERIAL even though SERIAL itself has not been altered and it still remains associated with that attribute.

What Attributes May be Assigned to a Field

Besides length, which is controlled by the position of attributes, you may specify additional characteristics with the attribute character:

Protection: A field is either protected or unprotected. When it is protected, it means that the operator cannot enter or modify data in any location within that field.

In an unprotected field, the operator can enter characters or can delete or modify characters that are already there, with the keyboard. Headings, labels, titles, and formats are commonly specified as protected. Any field in which the 3270 operator should enter or modify data must be specified as unprotected.

In Figure 1, NAME: would most likely be specified as protected. JOHN B DOE would be specified as protected if it was written by the computer and is to remain unchanged. If JOHN B DOE is to be entered or modified by the operator, the attribute [2] must specify unprotected.

Character Content: A field is either alphameric or numeric. An operator can enter alphameric, numeric, or special characters in an alphameric field.

The numeric attribute is more complex; it depends upon whether the Numeric Lock feature is present and which keyboard is attached to the display. Figure 2 shows what characters may be entered with various combinations of keyboards and field types.

Visibility and Detectability: A field is either displayable or nondisplayable. When it is displayable, and it contains characters, those characters are displayed. When it is nondisplayable, any characters within that field will not be displayed. The nondisplayable attribute is useful for entering classified or security information at a display unit that is in public view. Nondisplayable data is accepted by the 3270 but it is not visible on the screen.

All characters within a displayable field can be displayed at regular brightness or at a high intensity so that they stand out among regular display fields. High intensity may be used to call attention to error conditions or to highlight protected or format fields. Normal intensity may be used for all input fields, so the terminal operator can tell at a glance which fields require operator action. You should not specify unprotected fields as high intensity since such fields may become selector pen detectable (if this feature is installed) if the operator enters a question or blank as the first input character.

Keyboard Type	Keyboard Numeric Lock	Shift Key Pressed	Field Type	Protected	Resulting Characters		
					In Buffer	Displayed On Screen	Read Into Storage
Typewriter	No	No	Alpha or Numeric	No	Lowercase	Uppercase	Lowercase
Typewriter	No	Yes	Alpha or Numeric	No	Uppercase	Uppercase	Uppercase
Typewriter	Yes	No	Alpha	No	Lowercase	Uppercase	Lowercase
Typewriter	Yes	Yes	Alpha	No	Uppercase	Uppercase	Uppercase
Typewriter	Yes	No	Numeric	No	Can only enter 0-9, period, and minus sign; any other characters lock keyboard.		
Typewriter	Yes	Yes	Numeric	No	Can only press dup key; any other action locks keyboard.		
Data Entry	No	--	Alpha	No	Alpha keys produce uppercase alpha characters. Numeric shift key produces numeric characters. Alpha shift key has no effect.		
Data Entry	No	--	Numeric	No	Numeric shift key has no effect. Alpha shift key overrides numeric specification and allows alpha character entry.		
Data Entry	Yes	--	Alpha	No	Alpha keys produce uppercase alpha characters. Numeric shift allows numeric character entry. Alpha shift key has no effect.		
Data Entry	Yes	--	Numeric	No	Can only enter 0-9, period, dup, and minus sign. Numeric shift key allows numeric character entry, alpha shift key allows alpha character entry.		

Figure 2. Results of keyboard and field combinations

Fields are specified as either detectable or nondetectable. When a field is detectable, it can be used for selector pen operations. A nondetectable field location can not be detected by the Selector Pen. You are urged to designate all detectable fields as protected to prevent the the operator's changing the content of the sensitive field.

Transmission: The most common operation of the 3270 (Read Modified) sends to the computer only those fields which have been entered, deleted, or changed by the operator. The 3270 keeps track of such modifications and uses that information to select data to send to the computer. If you wish to pass a field into the computer regardless of modification, you may assign the "modified" or "modified data tag (MDT) on" attribute.

You can decide which combination of attributes you want within the limitations specified in the IBM 3270 Component Description. Certain attribute combinations produce additional characteristics. For example, the numeric (limiting keyboard use) and protected (eliminating keyboard use) attributes seem contradictory but when specified together automatically skip the cursor past the field.

You should also be aware that the computer is not limited by attributes. The computer may, for example, place alphabetic information in a field defined as numeric, or protected, or both. The operator does not have such liberty.

If you do not specify any combination of attributes, a field is assumed to have the following attributes:

- alphameric
- unprotected
- displayable (at regular brightness)
- nondetectable by the Selector Pen
- not modified

You will find that these attributes are the most commonly used.

The attribute character for each field utilizes a single nondisplayed and protected character position on the screen and serves as a visual separation between successive fields.

Example of Field Definition

A typical sign-on procedure illustrates how you might define fields. Figure 3 illustrates a simple procedure in which the computer requests the operator to provide his name, location, and serial number.

FIELD 1: "SIGN-ON PROCEDURE"

This field is a heading which is a permanent part of the screen's format and which the operator should not be able to alter. It is unnecessary for the words "SIGN-ON PROCEDURE" to be returned to the computer when the ENTER key is pressed. This field should be protected, alphameric, displayed at normal intensity, not detectable by the Selector Pen, and not modified. All default attributes can be assumed, except that you must specify this field as protected.

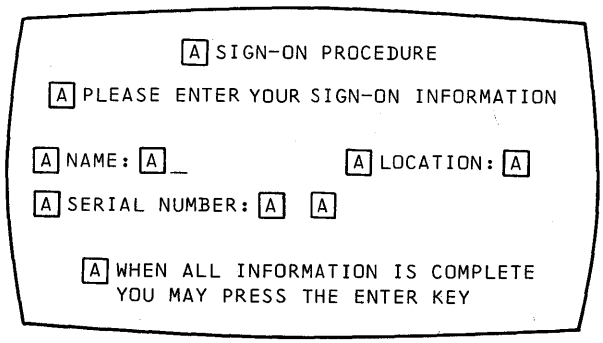


Figure 3. Example of attribute specification

FIELD 2: "PLEASE ENTER ... INFORMATION"

You should specify this field as protected. Remember that the characteristics of a field are determined by the attribute character at the beginning of the field. Field 1 and field 2 have identical attributes and are adjacent to each other. You may choose to define them separately and use two attribute characters or you may choose to omit the attribute character at the beginning of field 2. In the latter case the two headings combine to become a single field of greater length.

FIELD 3: "NAME:"

This field should be protected, alphameric, not modified, and not detectable by the Selector Pen. The heading could be displayed at high intensity. Specify the protected and high intensity attributes (the two deviations from the default attributes).

FIELD 4: the area following "NAME:"

The null area following NAME: is an input area for the operator and must therefore be unprotected. The 3270 marks this field as modified if anything is entered into it, so you should not specify the modified attribute. The default attributes (alphameric, unprotected, displayable at normal intensity, not detectable by the Selector Pen, and not modified) apply. Use a default attribute at the beginning of this field.

The maximum number of characters the operator can enter is determined by the length of this field. The length is equivalent to the number of nulls, or available positions on the screen, between the attribute character for field 4 and the attribute character for field 5.

FIELD 5: "LOCATION:"

The attribute character for this field is the same as that specified for field 3; protected and high intensity should be specified. This attribute prevents the operator from keying a name longer than the maximum length desired. If the name is shorter than the maximum field size, the operator presses the TAB key when the name is complete. The TAB automatically skips the cursor past protected fields, such as this one, and stops at the first character position in which data can be entered (the next unprotected field). In this example, the cursor would be positioned for entry of location. If the operator attempts to key too many characters (a name greater than 17 characters in the example) the cursor is positioned under this attribute for the 18th character. The next keystroke attempts to destroy this attribute but fails to do so since attribute characters are protected. The keyboard will be inhibited, the clicker will shut off, and the "input inhibited" indicator will turn on. The operator's attention is assured since this condition requires pressing the RESET key to continue.

If the attribute character for this field were omitted, the word "LOCATION:" would become part of field 4 and would be normal intensity and unprotected. This is undesirable since the operator could continue entering name information beyond the desired maximum length and could modify the heading information by entering data in the screen locations occupied by "LOCATION:."

FIELD 6: the area following "LOCATION:"

This field is for operator input and therefore must be unprotected. The rest of the default attribute values apply and so a default attribute may be used. You need specify only that a field is to begin following "LOCATION:." This field ends with the attribute character at the beginning of field 7, which determines the length of the field.

FIELD 7: "SERIAL NUMBER:"

This field, like "NAME:" and "LOCATION:" should be specified as protected and high intensity. This also limits the location field length to 5 characters. Note that if field 6, the input field for location, were defined as always a five-character code, field 7, "SERIAL NUMBER:", could be defined auto-skip to save the operator from having to press TAB after filling in the location code.

FIELD 8: the area following "SERIAL NUMBER:"

The null area following "SERIAL NUMBER:" is an input area for the operator and must be unprotected. It should also be specified as numeric so that if the operator tries to enter alphabetic data in the field (and the keyboard has the Numeric Lock feature), the keyboard inhibits entry of the incorrect character, the keyboard clicker shuts off, and the "input inhibited" indicator appears at the right of the screen to notify the operator of the error. The improper character does not appear on the screen and the correct digit may be entered after the operator presses the RESET key.

The serial number in the example always contains a fixed number of digits and is the last field entered. The maximum length of the field is determined by the location of the attribute for the next field. But the next field in the example is too far away ("WHEN ALL ... KEY").

By placing an additional attribute character following input field 8, the operator cannot enter a serial number that is too long. If the positions allocated to the serial number are filled, the next keystroke locks the keyboard, as in the name and location fields.

This additional length check is used here because this is the last field to be entered. If you had another field to enter after SERIAL NUMBER, it might be more advantageous to omit this length check, as explained in field 9.

FIELD 9: the area between the additional attribute described in Field 8 and "WHEN ALL ... KEY"

By definition, the additional attribute character you used to delimit the serial number field begins a new field. The protected attribute alone is sufficient for this field and this attribute limits length for the serial number field. Normally, however, protected (output) fields that follow fixed-length input fields should be defined as protected and numeric. The protected and numeric attribute defines a field as auto-skip. Auto-skip automatically positions the cursor at the location following the attribute character for the next unprotected field, which is the next place you want to key data. This technique saves keystrokes for the operator. When the operator keys the last character of the preceding fixed-length field, the cursor normally enters the next field, which may be protected. But since the next field is auto-skip, the cursor skips this intervening protected field, and automatically positions itself for entry of the next field, without an extra keystroke.

FIELD 10: "WHEN ALL ... KEY"

This field is a heading and a permanent part of the screen image. It need not be high intensity and thus it may be defined as protected only. Field 10 does not automatically terminate when the last screen position is reached. The field definition continues from the bottom right screen position to the upper left screen position until the next attribute character is reached. This is called wraparound. Keep this in mind, particularly if you define the last field on a screen as unprotected!

Since fields 9, 10, and 1 are adjacent to each other (by wraparound) and all have the same attributes, they may be combined into a single field by the omission of attributes before "WHEN" and "SIGN-ON." The result is a single protected field beginning after the input area for serial number, wrapping around the screen, and terminating either at "PLEASE", or at "NAME" if fields 1 and 2 have been previously combined.

Combining fields in the above manner may be convenient but may cause confusion and error if you change the screen layout later. It is a better practice to specify separate fields in all cases.

The panel is completely formatted when the fields are positioned, the attribute characters are all defined, and the cursor is placed. You must now begin the transition from the visual image, or human-oriented panel, to the detailed data necessary for the 3270 to implement your panel design.

PANEL DESIGN

This is a Panel:

You can think of a panel as a single 3270 display screen image created by your program. (The terms "screen" or "screen image" or "display image" could also have been used.)

If the terminal operator filled in the information requested in the panel in Figure 4, he might receive another panel such as the one shown in Figure 5.

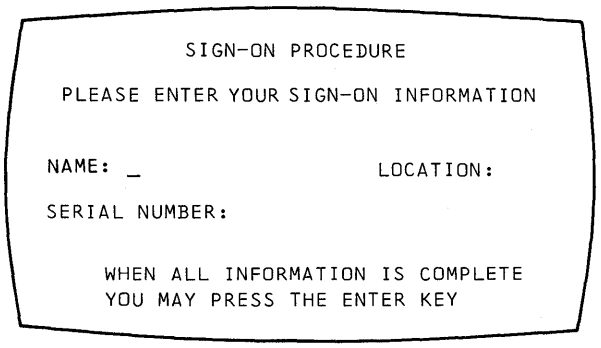


Figure 4. An example of a panel

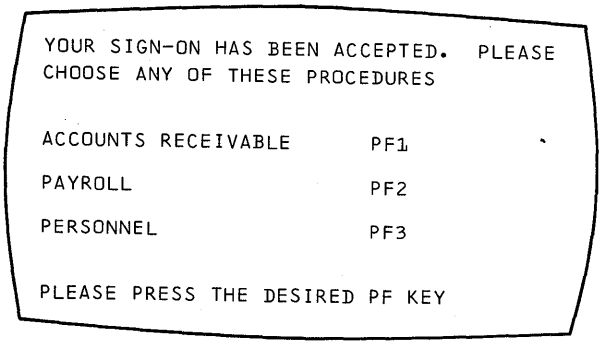


Figure 5. Another example of a panel

An Example of a Sequence of 3270 Panels

Assume you are given the assignment of designing the panels for an accounts receivable application. You are to create the panels that will allow a terminal operator to post a customer payment against his unpaid invoices. The terminal operator will be sitting at a 3270 work station, removing checks and invoice copies from envelopes. If the invoice copies are returned with the check, the terminal operator will for each invoice enter the customer number, payment, and invoice number. If the invoice copies are not returned, the terminal operator will have to find the customer number based on the customer name and then decide which open invoices to apply the payment against. It will be helpful if the operator has some way to add various open invoices to find a combination that totals the payment.

The 1920-character panels that follow show one possible solution.

The first panel in the application is shown in Figure 6. If the invoice copies come with the check, the terminal operator can enter the customer number, amount, and invoice number, and press the ENTER key. This posts the payment against the specified invoice. The terminal operator can then post the next payment and so forth; so long as the customer number and invoice number is known, only panel 1 is displayed.

If, however, no invoice is returned and the customer number is not known, the customer name can be entered. The name need not be the complete name of the company; it can be the first name of the company. In our example, the check says only "CAPITOL" so that is what the operator enters. When the name has been entered, the terminal operator presses the ENTER key. The customer number is missing, so Panel 2 is displayed.

Panel 2, shown in Figure 7, shows all customers and customer numbers phonetically similar to the name entered in response to Panel 1. Item numbers in Panel 2 allow the terminal operator to select one by using a corresponding Program Function (PF) key (see "Program Attention Keys" in this section).

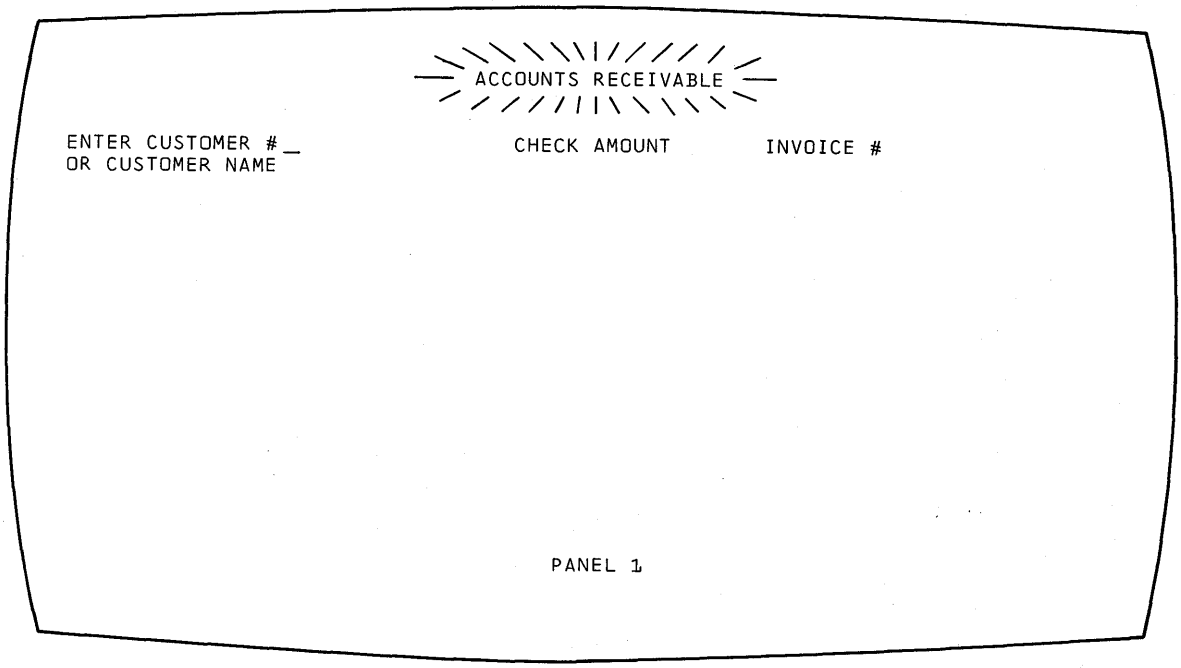
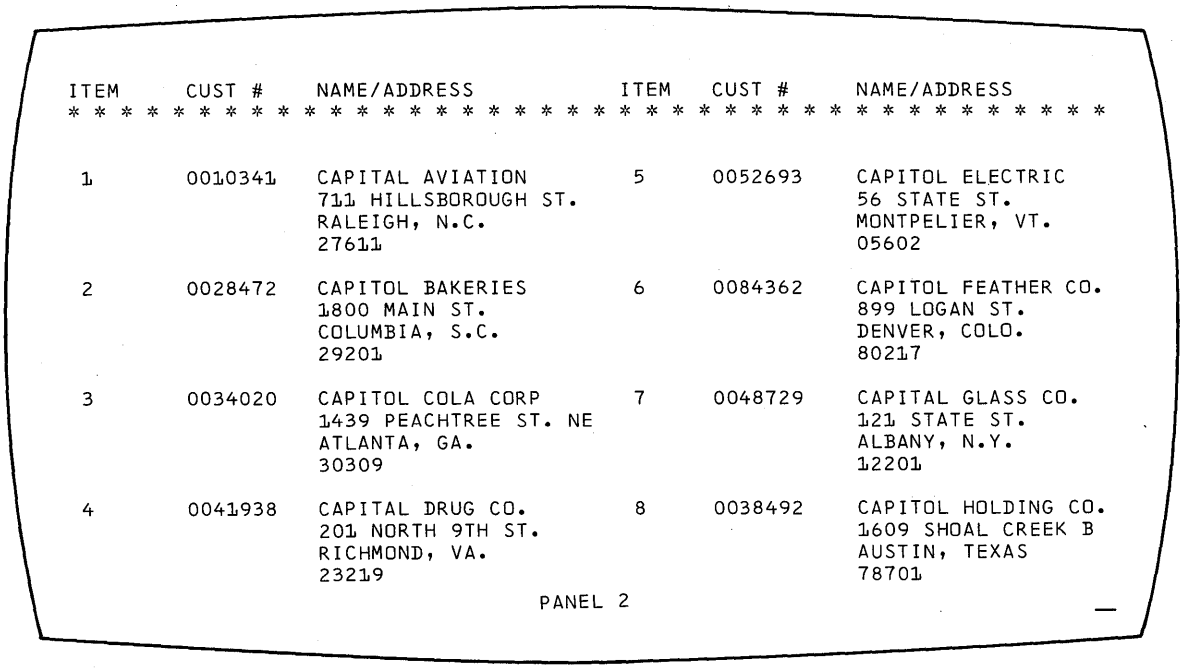


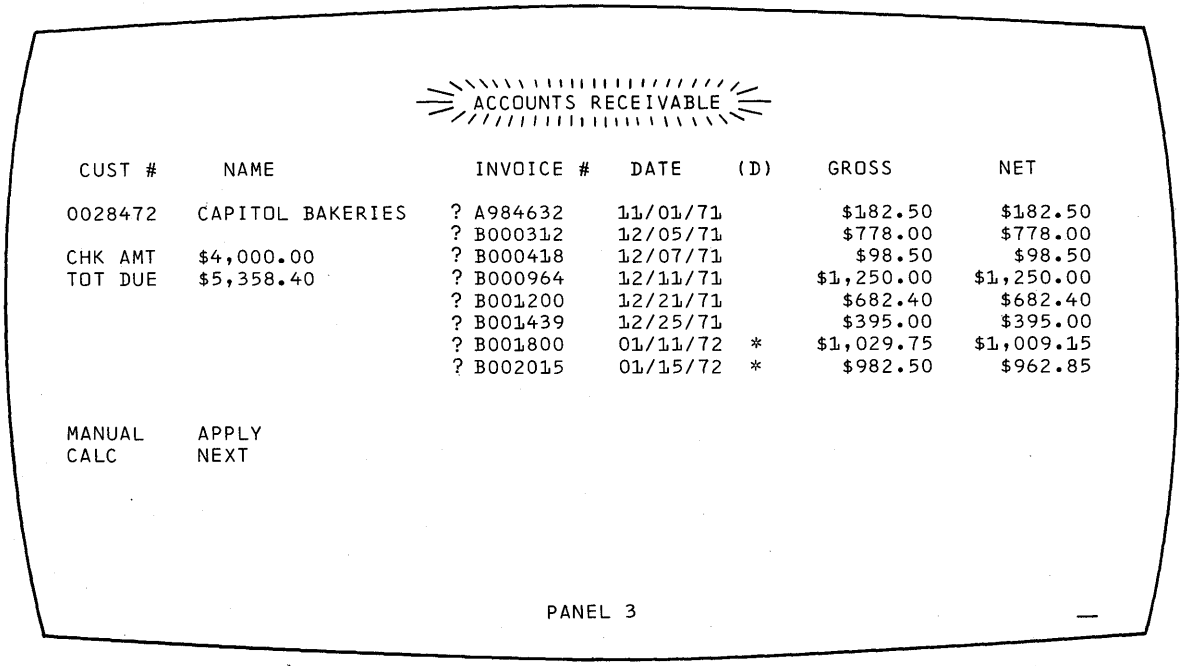
Figure 6. Panel 1 of an accounts receivable application



ITEM	CUST #	NAME/ADDRESS	ITEM	CUST #	NAME/ADDRESS
1	0010341	CAPITAL AVIATION 711 HILLSBOROUGH ST. RALEIGH, N.C. 27611	5	0052693	CAPITOL ELECTRIC 56 STATE ST. MONTPELIER, VT. 05602
2	0028472	CAPITOL BAKERIES 1800 MAIN ST. COLUMBIA, S.C. 29201	6	0084362	CAPITOL FEATHER CO. 899 LOGAN ST. DENVER, COLO. 80217
3	0034020	CAPITOL COLA CORP 1439 PEACHTREE ST. NE ATLANTA, GA. 30309	7	0048729	CAPITAL GLASS CO. 121 STATE ST. ALBANY, N.Y. 12201
4	0041938	CAPITAL DRUG CO. 201 NORTH 9TH ST. RICHMOND, VA. 23219	8	0038492	CAPITOL HOLDING CO. 1609 SHOAL CREEK B AUSTIN, TEXAS 78701

PANEL 2

Figure 7. Panel 2, showing the results of a search on a customer name



ACCOUNTS RECEIVABLE

CUST #	NAME		INVOICE #	DATE	(D)	GROSS	NET
0028472	CAPITOL BAKERIES	?	A984632	11/01/71		$182.50	$182.50
		?	B000312	12/05/71		$778.00	$778.00
CHK AMT	$4,000.00	?	B000418	12/07/71		$98.50	$98.50
TOT DUE	$5,358.40	?	B000964	12/11/71		$1,250.00	$1,250.00
		?	B001200	12/21/71		$682.40	$682.40
		?	B001439	12/25/71		$395.00	$395.00
		?	B001800	01/11/72	*	$1,029.75	$1,009.15
		?	B002015	01/15/72	*	$982.50	$962.85

MANUAL APPLY
CALC NEXT

PANEL 3

Figure 8. Panel 3, showing the customer's open invoices

As a result of terminal operator response to Panel 2, Panel 3 (shown in Figure 8) displays all open invoices for the identified customer. The terminal operator can now use the Selector Pen to specify the open invoices to which the payment applies. He does this by touching the Selector Pen to the question mark adjacent to each desired invoice number; selection is verified immediately by the question mark changing to a > character. To post the payment against the selected invoice numbers, the operator can select APPLY. If, however, the operator can not easily

tell the invoices to which the payment is applied, he can select CALC instead of APPLY.

Selecting CALC displays Panel 4 (Figure 9); this is the same as Panel 3 except that ACCOUNTS RECEIVABLE which was high intensity in Panel 3 is now normal intensity in Panel 4. A new line with CALCULATOR in high intensity indicates the screen mode and explains the PF keys' functions. The terminal operator can now use the lower right hand quadrant of the screen as a "scratch pad" to figure out a combination of open invoices that will total the payment check. This use of one part of the screen for a separate function is sometimes called a "split-screen capability."

The calculator could be programmed a number of different ways. It could, as our example illustrates, show all invoice numbers selected (shown with > in Figure 9) prior to selecting CALC in one column in the CALCULATOR quadrant and in another column show any balance remaining from the check amount after subtracting the selected invoice numbers. In Figure 9, Panel 4 is shown as it would appear if the terminal operator had first selected four invoice numbers and then selected CALC. In this example, the selected invoices equal the check amount so .00 is shown as the balance after subtracting the selected invoices.

Panel 4 shows that the CALCULATOR could also allow the operator to key in amounts and add or subtract them from the check amount (pressing PF1 in our example adds keyed-in amounts; PF2 subtracts one keyed-in amount from another). To start over at any point, the operator can press PF3 to clear the calculator quadrant. In our example, the selected invoices equal the check amount, so they can now be posted. But first the terminal operator must leave the CALCULATOR routine by pressing PF4 (RETURN). This displays Panel 5, shown in Figure 10.

Panel 5 is the same as Panel 4 except that, with the operator having signaled completion of the CALCULATOR, that word now appears in normal intensity and ACCOUNTS RECEIVABLE once again appears in high intensity. The terminal operator can now, using the Selector Pen, select the invoices against which to apply the payment and then select APPLY to post the payment.

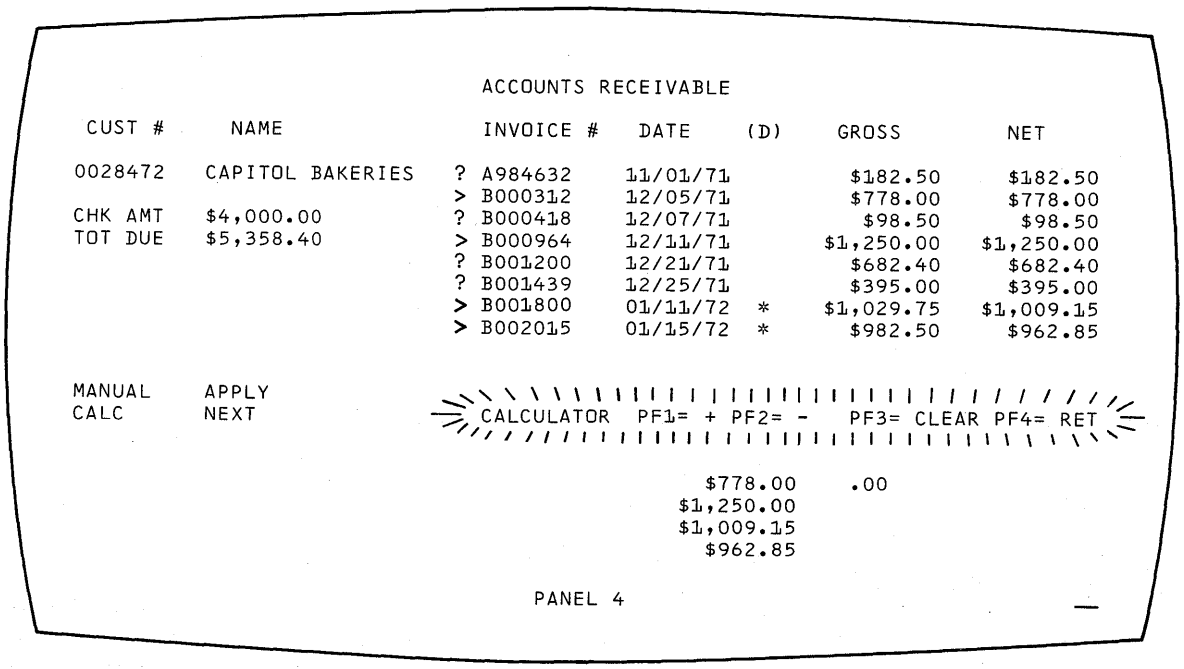


Figure 9. Panel 4, showing use of the calculator

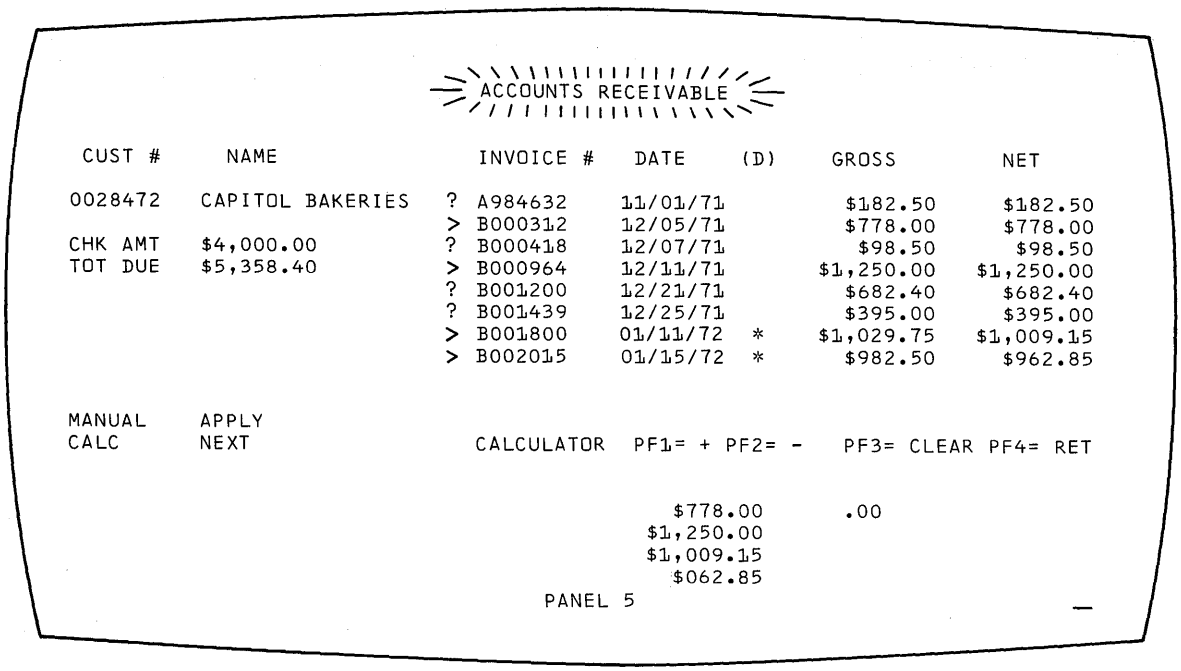


Figure 10. Panel 5, showing selection of invoices after using the calculator

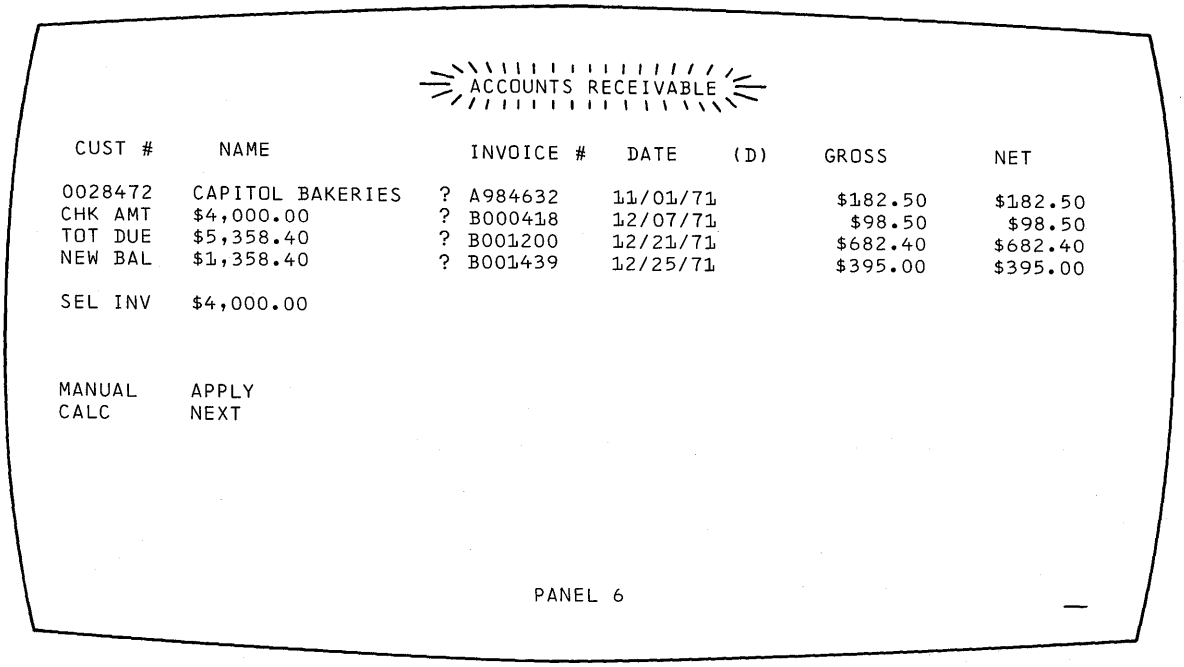


Figure 11. Panel 6, showing new balance after posting

Panel 6, in Figure 11, shows the ACCOUNTS RECEIVABLE file for the customer after posting the payment, with the new balance and the total amount applied. To continue to the next customer, the operator selects NEXT and returns to Panel 1.

Not all of the 3270's possibilities are shown in the these six panels and not all users will have the Selector Pen; this example was designed only to show what panels are and how the 3270 can be used.

Note that, in the above example, the terminal operator does not see as many panels as the programmer must create; not all panels necessarily appear to the operator in any given application. What the programmer regards as separate panels may, to the terminal operator, appear to be one changing panel.

In the above example, a number of additional panels or variations to the panels shown would be required. For example, if the terminal operator presses an invalid PF key, a variation of the panel would be required to send a message to the operator over the panel presently at his display. In programming panels that are variations of one main panel, it may be useful to assign panel designations (for example, Panel 4A, 4B, and so forth) for variations of Panel 4.

Planning a Sequence of Panels

After an application program has been defined, the information that will be passed between the program and the terminal operator must be defined. This information can be thought of as output panels and input response to panels. Usually, you will be able to approximate the sequence of panels. The exact sequence of output panels often depends on the input response to panels. The following discussion shows one way to define a sequence of panels.

Defining the Purpose of Each Panel

Assuming you have a good understanding of the type of application program (such as data entry, order entry, or inquiry), and the kind of information that must be exchanged and processed (such as customer name, invoices, and check amounts), you can consider which panels come first. Suppose the first panel required is a sign-on panel, as shown in Figure 12.

After sign-on, the next panel might allow the terminal operator to choose one of several different applications or procedures that he would use. But what if the name or word entered was not an authorized sign-on? Another panel might tell the terminal operator about this and ask him to re-enter a sign-on name, as shown in Figure 13.

This technique, sometimes called "block diagramming," may help in laying out a sequence of panels.

Using the Panel Layout Sheet

After block diagramming the panels in the application or procedure, you are ready to decide on the exact contents of each panel: the fields that will be in the panel, what attributes each field will have, and what words will be displayed in the panel. This can be done on graph paper. The IBM 3270 Information Display System Panel Layout Sheet is useful for layout.

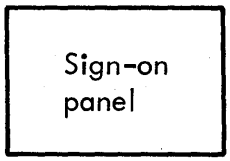


Figure 12. Sign-on panel block diagram

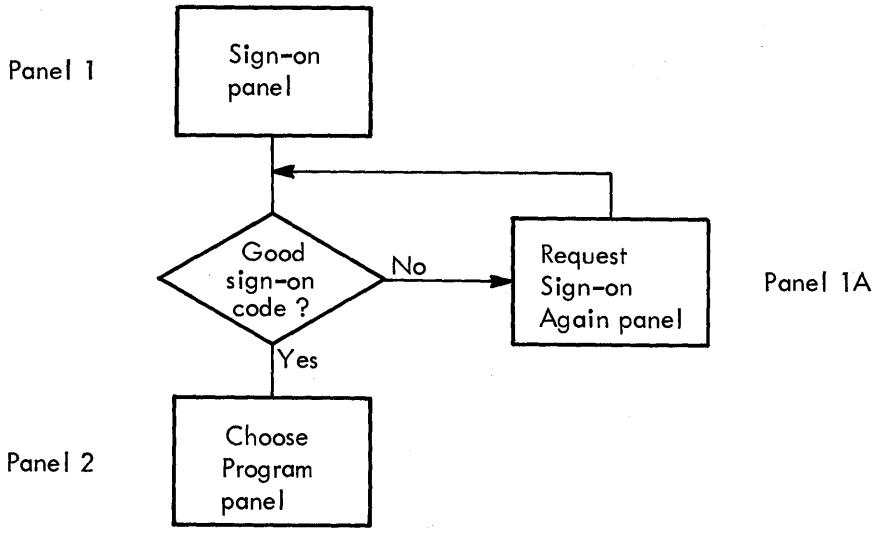


Figure 13. Block diagrams

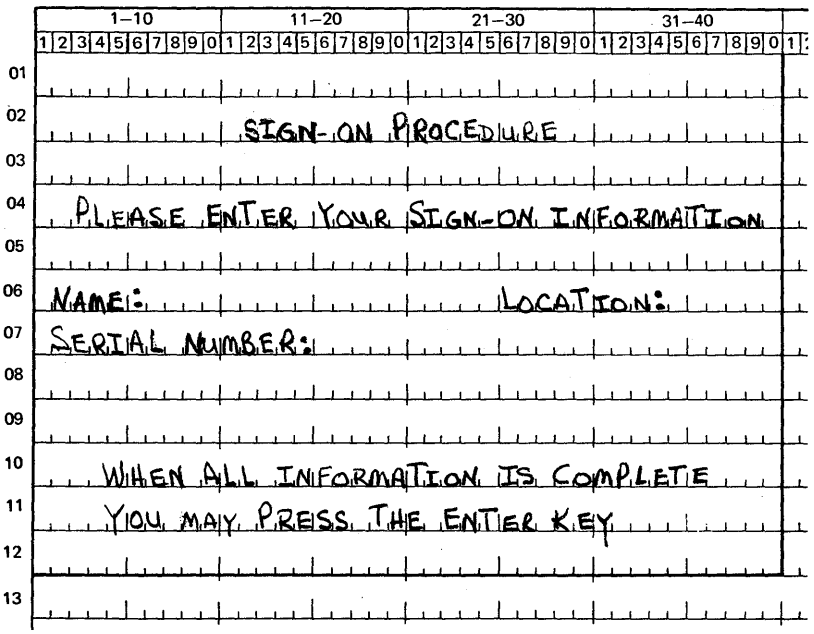


Figure 14. Sign-on panel as written out on layout sheet

One of these sheets can be used for each panel, for either 480-character or 1920-character displays. After laying out a sequence of panels, you have a collection of panel layout sheets. Using the information on these sheets and the block diagram showing the relationship between panels, the program can be written to send the panels to a terminal and handle an operator's response to them.

An Example of Laying Out a Panel

To lay out a panel, consider the sign-on panel shown in Figure 12. You might jot down on a piece of paper the information required for the panel, or you might write it directly on the panel layout sheet. Figure 14 shows what the panel part of the layout sheet might look like after you put the text you wanted for your sign-on panel on the layout sheet. It is assumed you are using the 480-character display.

Now that you have written out what you want the terminal operator to see, you can define as fields the separate items of displayed text and spaces you are allowing for operator input. Remember that a field is always preceded by an attribute character. The attribute character occupies a space on the panel even though it appears as a blank space to the operator. Before deciding the attributes of a field, insert some character such as A on the layout sheet to indicate the space for the attribute character. As you get used to creating panels, you may want to enter the A at the same time you are laying out the text. You should also show the cursor location on the panel layout sheet to indicate to the operator where to start his response. The cursor position can be indicated by an underscore (_) under the space where you want it to appear. After adding the indications for attribute characters and the cursor position, the sign-on panel appears as shown in Figure 15.

You could have designed the panel as one long field (or even no field at all), but if you did, you would not be taking advantage of the 3270's capabilities. By designating various items on the panel as fields, each field can have different attributes, as discussed in "What Attributes May be Assigned to a Field."

For example, you might want the fields NAME:, LOCATION:, and SERIAL NUMBER: to have high intensity attribute to focus the operator's attention on them, since these fields indicate where the operator enters information. You might want to protect the fields other than the operator input fields so the operator could not erase them; the operator input fields following NAME:, LOCATION:, and SERIAL NUMBER: should be unprotected so the operator can type in information. The operator input field following SERIAL NUMBER: can be numeric to allow some work station editing; the operator would not be allowed to accidentally enter an alphabetic character. Field length can be defined by beginning a new field where you want the previous field to end (in some cases, this new field serves only to give a length attribute to a previous field).

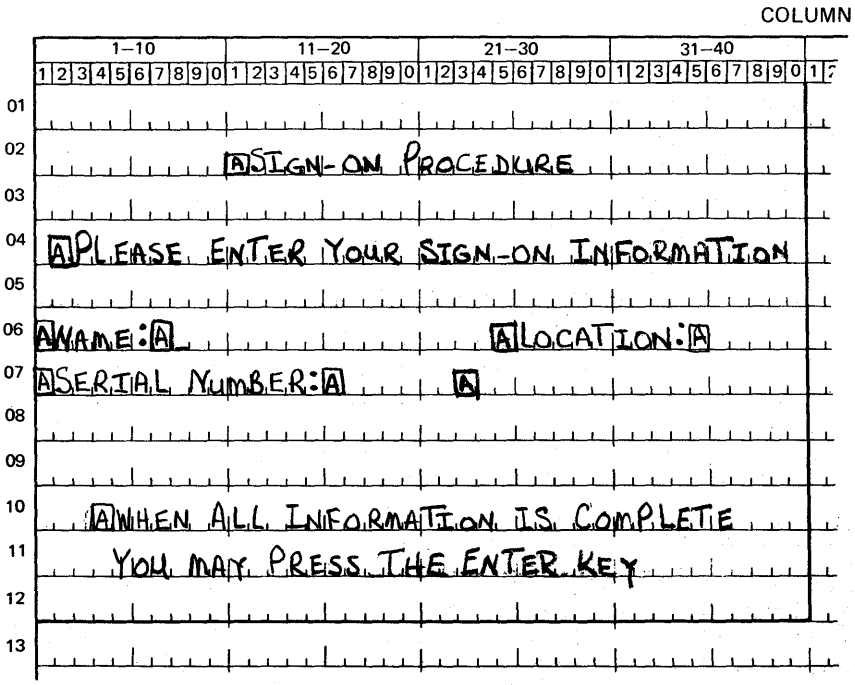


Figure 15. Panel layout including attribute and cursor positions

ITEM	DISPLAY/PRINTER ROW	DISPLAY/PRINTER COL	BUFFER ADDRESS DEC	BUFFER ADDRESS HEX	ORDERS	ATTRIBUTE PROT	ATTRIBUTE NO.	ATTRIBUTE HI INT	ATTRIBUTE SEL PEN DET	ATTRIBUTE NON DISP/PRT	ATTRIBUTE MDT ON
1	2	11				✓					
2	4	2				✓					
3	6	1				✓		✓			
4	6	7									
5	6	25				✓		✓			
6	6	35									
7	7	1				✓		✓			
8	7	16					✓				
9	7	23				✓					
10	10	4				✓					

Figure 16. Laying out field attributes

Having decided on these attributes, you can use the columns on the right side of the layout sheet to record the locations and attributes of the fields you have created. Your recording in these columns might appear as in Figure 16.

The use of these columns depends on whether the panel designer also codes the panels or only designs them. The information now on the layout sheet can be used to write a line of code that, when sent to the display, displays your panel with its specified field characteristics. The next section, "Data Stream Coding and Decoding," shows how the panel in this example is coded.

DATA STREAM CODING AND DECODING

Elements of a Data Stream

There are three categories of data which may be sent to or received from a 3270:

- I/O Control. Certain information is necessary to control the orderly transfer of information between the 3270 and the computer. For example, with remote display units a set of rules is necessary to maintain order on the common carrier communication lines. The 3270 uses Binary Synchronous Communication (BSC), which requires particular control characters to regulate transmission. The local 3270 does not use BSC. I/O Control is the subject of later chapters and will not be considered at this time.

- Device Control. Once you have the ability to send to and receive from a 3270 with the aid of I/O Control, you must communicate additional information to the device or its control unit so it can use panels. This information may be commands, control characters, and orders. Commands control such things as whether you write to or read from a display and whether the screen is erased before new data is written. (Commands are discussed in more detail in this section under "The Relationship of one Data Stream to Another." For these examples, assume that you begin with a clear screen: all writes to the 3270 are Erase/Write Commands and all positions are set to nulls. Control characters are used with certain commands to perform such functions as sounding the audible alarm, formatting the printer, and restoring or enabling the keyboard. Control characters are discussed later in this section. Orders are instructions written to the 3270 to tell the display unit how to format your panel. They

control the creation and placement of fields and data. You may reduce the size of your data streams by careful order selection.

- Text. This is the data you have in your computer that you want the terminal operator to have, such as field headings or inquiry responses, and it is the data that the operator has that must be provided to the computer, such as serial number, part number, or quantity desired.

Orders

Orders (1) position, define, and format data being written to the device; (2) erase selected unprotected data stored in the device; and (3) reposition the cursor.

Three orders cause the next character to be interpreted as an attribute and provide enough instruction to format every panel:

- Start Field (SF) order - specifies that the next character is an attribute character.
- Set Buffer Address (SBA) order - specifies an address for data and successive orders.
- Insert Cursor (IC) order - moves the cursor to the current buffer address.

These orders are included with the text data sent to the display unit and are interpreted by a control unit to which the display unit is attached. The control unit formats the panel text before it is actually displayed at the display station.

Adding Orders to the Panel Layout Sheet

The right side of the Panel Layout Sheet is used for writing the panel orders. The headings at the top of the columns indicate what the columns should contain.

The first six columns as shown in Figure 17 identify items in the text, their addresses, and the orders required to format them. The column headings are explained below:

- ITEM - refers to any part of the panel that requires an order to the control unit to format it. There are 11 items in the sign-on panel:

  1. SIGN-ON PROCEDURE

  2. PLEASE ENTER YOUR SIGN-ON INFORMATION

ITEM	DISPLAY PRINTER		BUFFER ADDRESS		ORDERS
	ROW	COL	DEC	HEX	

Figure 17. Text items on panel layout sheet

3. NAME:

4. Input field

5. LOCATION:

6. Input field

7. SERIAL NUMBER:

8. Input field

9. Field to limit size of serial number input

10. WHEN ALL INFORMATION IS COMPLETE

11. YOU MAY PRESS THE ENTER KEY

It is only by coincidence that the number of items in this example equals the number of fields. Since each field requires an SF order, there are always at least as many items as fields. There are more items than fields when, for example, the SBA order is used to space over unused positions within a single large field, as in Item 11.

- ROW, COL - This column contains the starting location (row, column) address of each item.

- DEC, HEX - This column is for a different addressing format which you do not need if you use the row, column addressing format. Therefore, you may use this column for any notes to yourself or leave it blank.

- ORDERS - This column contains the orders you are writing such as SBA, SF, or IC.

As shown in Figure 18, the next six columns under the word ATTRIBUTE provide the field attributes that can be defined with each attribute character. The programmer checks the appropriate columns of the attributes he is changing from the default values:

- PROT - protected

- NO. - numeric

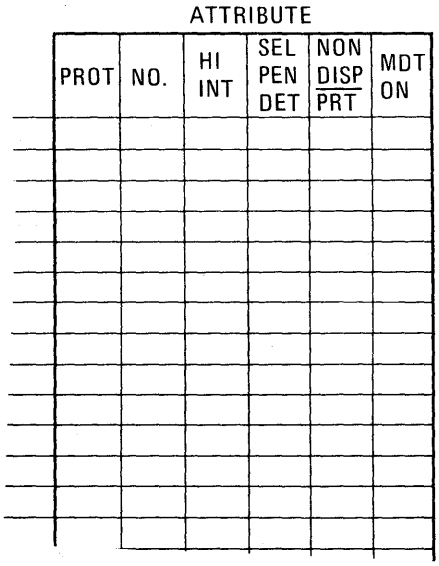


Figure 18. Field attributes

- HI INT - high intensity
- SEL PEN DET - Selector Pen detectable
- NONDISP/PRT - not displayed (nor printed at printer)
- MDT ON - modified data tag on

At the bottom of the six columns, the attribute values are shown (Figure 19) that are automatically provided unless you specify a change. You must however specify a hexadecimal order value for the default attributes, as discussed under "Coding the Panel" in this section. The default values are:

- UNPR - unprotected
- A/N - alphameric (alphabetic and numeric)
- NORM - displayed at regular brightness
- NON - not detectable by the selector pen
- NORM - displayed (at regular brightness)
- OFF - not modified

You are now ready to add the required orders to the panel layout form. This may require that you rewrite the right half of the form if it was originally prepared without regard to orders or if insufficient space was allowed.

Figure 20 shows a completed layout sheet containing all the orders to be sent with the sign-on panel. The hexadecimal order values are discussed under "Coding the Panel" in this section and shown in Figure 22. Each item on the panel has been assigned an item number to help you correlate the text with its associated orders.

Item 1. SIGN-ON PROCEDURE. To write this title, you must tell the control unit:

1. Where you want the title displayed on the panel. The SBA order sets the buffer address (SBA) to location R2, C11.
2. That this location is the start of a field. The SF order tells the control unit that the location contains an attribute character and not a text character. You also indicate which attributes the attribute character is defining. In this case, the field is protected. The rest of the attributes for the field are default attributes and therefore do not have to be changed.

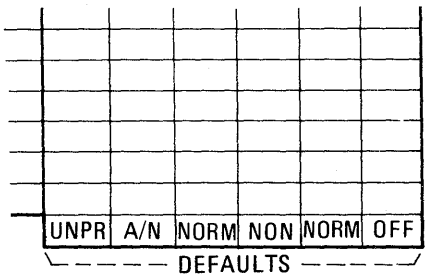


Figure 19. Attribute default values

ITEM	DISPLAY PRINTER ROW	DISPLAY PRINTER COL	BUFFER ADDRESS DEC	BUFFER ADDRESS HEX	OR-DERS		ATTRIBUTE PROT	NO.	HI INT	SEL PEN DET	NON DISP PRT	MDT ON
1	02	11			SBA							
					SF	Att	✓					
2	04	02			SBA							
					SF	Att	✓					
3	06	01			SBA							
					SF	Att	✓		✓			
4	06	07			SF	Att						
					IC							
5	06	25			SBA							
					SF	Att	✓		✓			
6	06	35			SF	Att						
7	07	01			SBA							
					SF	Att	✓		✓			
8	07	16			SF	Att		✓				
9	07	23			SBA							
					SF	Att	✓					
10	10	03			SBA							
					SF	Att	✓					
11	11	05			SBA							

Figure 20. Completed order and attribute information

Item 2. PLEASE ENTER YOUR SIGN-ON INFORMATION: To write this information, the control unit must know only where the text is located. Therefore, you must write an SBA instruction followed by the address R4, C2. This is also the beginning of a protected field, so you should include an SF order and a protected attribute.

Item 3. NAME: Like Item 2, you must identify where this text is displayed. Therefore, you must write an SBA order followed by the buffer address R6, C1, where the text begins. R6, C1 is also the beginning of a protected, high-intensity field and you should include an SF and an attribute as shown.

Item 4: Input Field for operator's name. Since this item immediately follows Item 3, the control unit already knows the correct address. Therefore, there is no reason to issue an SBA order. Item 4 is the start of a new field, however, so you must issue an SF order to instruct the display to expect an attribute character next. The attribute character defines the input field as unprotected (U), alphameric (A), normal intensity, non-selector pen detectable, and no MDT on. Since these are the default attributes, you do not have to check anything in the attribute definition columns.

The cursor should follow the attribute character to indicate where the operator should begin to enter information. The Insert Cursor (IC) order displays the cursor at this current buffer address. After the display has stored the attribute character in location R6, C7, the new current address is R6, C8, and this is where the cursor appears on the panel.

Item 5: LOCATION: The control unit must have two orders for this item which (1) give the starting buffer address (SBA) of the field as R6, C25, and (2) indicate that it is the start of a new field (SF), that it is protected, and high intensity.

Item 6: Input field for operator's location code. This item immediately follows the text of the last item so there is no need to set the buffer address. Write only the SF order to indicate the start of a new unprotected field, and use default attributes.

Item 7: SERIAL NUMBER: This field requires an SBA to location R7, C1, and an SF to begin a new field. The attribute is specified the same as Item 5.

Item 8: Input field for serial number. The attribute character for this input field immediately follows the last character of the previous field so an SBA is not required. The attribute is numeric only.

Item 9: An extra field created to limit the size of the serial number input field. This follows the input field and is protected only. An SBA is required for location R7, C23, for proper placement of the attribute.

Item 10: "WHEN ALL ... COMPLETE". The control unit must have two orders for this item: an SBA order that gives the starting address of R10, C3, and an SF order to indicate that it is the start of a new field. The attribute character defines a protected field, and the rest of the field attributes are defaulted.

Item 11: "YOU MAY ... KEY" All the words from "WHEN ALL" through "KEY" could have been treated as a single item, but 8 blank spaces would have to be sent between "COMPLETE" and "YOU" to position "YOU" properly at R11, C5. Use only the 3 characters required for an SBA order and its associated address, breaking the field into 2 items, to position "YOU" at R11, C5.

Coding the Panel

To write a panel in Assembler Language so that it can be part of the application program, you must transfer the panel's text and orders to an Assembler coding sheet or to any other form you find suitable.

On the coding sheet (and in your program) a panel is represented by a series of Assembler DC statements, each with a name to which your program can refer. In the example, SIGNPANL is the name of the sign-on panel. When the application program wants to send the sign-on panel to a display unit, it issues a WRITE command and designates SIGNPANL as the panel for display.

The display orders must be written in the DC statements in the hexadecimal codes listed in Figure 21. Thus, SF is represented by 1D, SBA by 11, and IC by 13.

Each part of each order must be written in hexadecimal including the attribute character that follows the SF order and the buffer address that follows the SBA order. The IBM 3270 Reference Card contains the hexadecimal codes for all the attribute character combinations and the hexadecimal code for every buffer location in both EBCDIC and ASCII.

Begin coding with the first item on the panel layout sheet: the title, SIGN-ON PROCEDURE. Start with the orders for the panel text, which must always precede the text itself so that the control unit knows what to do with the text.

The first order for the title is the SBA order. Figure 21 shows that the SBA hexadecimal code is 11 so you write this code in a DC statement as:

```
DC  X'11'
```

BUFFER CONTROL ORDERS AND ORDER CODES

Order Sequence / Order	Byte 1 (Order Code) EBCDIC Hex	Byte 1 (Order Code) ASCII Hex	Byte 2	Byte 3	Byte 4
Start Field (SF)	1D	1D	Attribute		
Set Buffer Address (SBA)	11	11	Address	Address	
Insert Cursor (IC)	13	13			
Program Tab (PT)	05	09			
Repeat To Address (RA)	3C	14	Address	Address	Char.
Erase Unprotected To Address (EUA)	12	12	Address	Address	
Keyboard Only					
Duplicate (DUP)	1C	1C			
Field Mark (FM)	1E	1E			

Figure 21. Buffer control orders and order codes

ITEM	DISPLAY PRINTER ROW	DISPLAY PRINTER COL	BUFFER ADDRESS DEC	BUFFER ADDRESS HEX	ORDERS		ATTRIBUTE PROT	NO.	HI INT	SEL PEN DET	NON DISP PRT	MDT ON
1	02	11		40F2	SBA							
					SF	AH	✓					
2	04	02		C1F9	SBA							
					SF	AH	✓					
3	06	01		C3C8	SBA							
					SF	AH	✓		✓			
4	06	07			SF	AH						
					IC							
5	06	25		C360	SBA							
					SF	AH	✓		✓			
6	06	35			SF	AH						
7	07	01		C3F0	SBA							
					SF	AH	✓		✓			
8	07	16			SF	AH		✓				
9	07	23		C4C6	SBA							
					SF	AH	✓					
10	10	03		C56A	SBA							
					SF	AH	✓					
11	11	05		C6D4	SBA							

Figure 22. Sign-on procedure panel orders and attributes

Now look up the R2, C11 address that must follow the SBA order. The EBCDIC address is 40F2 and it follows the SBA code in the DC statement:

```
DC  X'1140F2'
```

You should also record this statement in the Buffer Address HEX column to the left of the SBA on the layout form for possible future reference. You may, if you prefer, look up all the addresses and record them in a similar manner before you begin to write your DC statements. See Figure 22 for an example.

The next order for the title is the SF order, which is followed by the attribute character. Attribute characters are shown in Figure 23. The SF code, 1D, and the attribute code, 60, are read from the table and added to the DC statement, which is then closed with a single quotation mark:

```
DC  X'1140F21D60'
```

ATTRIBUTE CHARACTER BIT DEFINITIONS

ATTRIBUTE						Bits	Hex
Prot	A/N	MDT ON	High Intens	Sel Pen Det	Non Disp PRT	23 4567	
U						00 0000	40
U		Y				00 0001	C1
U				Y		00 0100	C4
U		Y		Y		00 0101	C5
U			H	Y		00 1000	C8
U		Y	H	Y		00 1001	C9
U			–	–	Y	00 1100	4C
U		Y	–	–	Y	00 1101	4D
U	N					01 0000	50
U	N	Y				01 0001	D1
U	N			Y		01 0100	D4
U	N	Y		Y		01 0101	D5
U	N		H	Y		01 1000	D8
U	N	Y	H	Y		01 1001	D9
U	N		–	–	Y	01 1100	5C
U	N	Y	–	–	Y	01 1101	5D
P						10 0000	60
P		Y				10 0001	61
P				Y		10 0100	E4
P		Y		Y		10 0101	E5
P			H	Y		10 1000	E8
P		Y	H	Y		10 1001	E9
P			–	–	Y	10 1100	6C
P		Y	–	–	Y	10 1101	6D
P	S					11 0000	F0
P	S	Y				11 0001	F1
P	S			Y		11 0100	F4
P	S	Y		Y		11 0101	F5
P	S		H	Y		11 1000	F8
P	S	Y	H	Y		11 1001	F9
P	S		–	–	Y	11 1100	7C
P	S	Y	–	–	Y	11 1101	7D

S = Skip Y = Yes
U = Unprotected H = High
P = Protected N = Numeric

Figure 23. Attribute character combinations in hexadecimal

Following the DC statement containing the orders for the title is the DC statement containing the text for the title:

```
DC   X'1140F21D60'

DC   C'SIGN-ON PROCEDURE'
```

To code an input field that contains no text, such as the input field for NAME:, write just one DC statement that contains the orders for that field:

```
DC   X'1D4013'
```

1D is the hexadecimal code for the SF order, 40 is the hexadecimal code for an attribute character that defines an unprotected field (and all other default attributes), and 13 is the hexadecimal code for the IC order.

A DC statement can be written as two or more statements. The DC statement above, for example, could be written as:

```
DC   X'1D40'

DC   X'13'
```

Each item from the panel layout sheet is coded in this fashion. Figure 24 shows the complete code required to display the sign-on panel. Except for one control character, it consists entirely of the panel text, preceded by the display orders for that text.

Write Control Character (WCC)

The control unit to which the display unit is attached uses the orders to format the panel. One control character for the control unit must be included as the first character of every panel you write: the Write Control Character (WCC). The WCC is a hexadecimal code that provides control information for the control unit and defines printer information for printing panels. The other information in the WCC specifies:

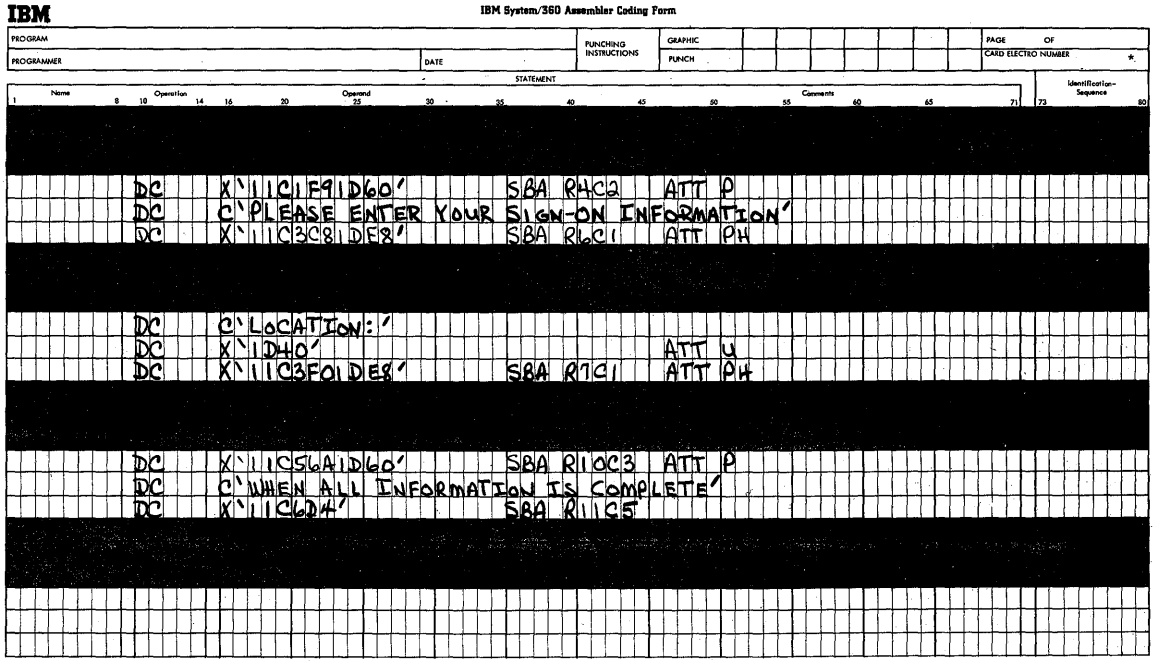


Figure 24. Assembly language statements for sign-on panel

- Whether to sound the Audible Alarm. The Audible Alarm is an optional display unit feature that sounds a tone at the display unit upon program request. You can request this function by selecting the appropriate WCC hexadecimal code. If this feature is not installed on a display unit, the request is ignored.

- Whether to restore the keyboard at the end of your panel operation. If this option is requested, the keyboard, which locks when the operator completes a panel operation, is automatically unlocked when the program has finished processing the operator's input. Keyboard restoration means the operator does not have to press the RESET key.

  You might not want to unlock the keyboard after each panel is displayed. For example, if you plan to write out another panel before you want to accept input, locking the keyboard prevents the operator from entering data before it is needed. Also, after writing an incorrect panel you may want to force the operator to press the RESET key to make sure you have gained his attention.

- Whether to reset the Modified Data Tag (MDT). If this option is specified, the attribute characters of all modified fields are reset. This function resets all input fields to their original (unmodified) status when an operation is completed so they are ready for the next operation.

Each panel written to a display unit or printer must begin with the WCC to identify whether these functions are requested.

The hexadecimal code for each possible WCC combination is shown in Figure 25.

WCCs for the Display

Start Printer	Sound Audible Alarm	Restore Keyboard	Reset MDTs	Code This Hex Value
No	Yes	Yes	Yes	C7
No	Yes	Yes	No	C6
No	Yes	No	Yes	C5
No	Yes	No	No	C4
No	No	Yes	Yes	C3
No	No	Yes	No	C2
No	No	No	Yes	C1
No	No	No	No	C0

WCCs for the Printer

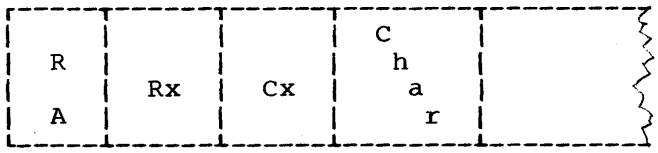
Start Printer	Sound Audible Alarm	Restore Keyboard	Reset MDTs	Code This Hex Value If You Want			
				NL and EM Codes Honored	40-Char. Line	64-Char. Line	80-Char. Line
Yes	Yes	Yes	Yes	CF	5F	6F	7F
Yes	Yes	Yes	No	CE	5E	6E	7E
Yes	Yes	No	Yes	CD	5D	6D	7D
Yes	Yes	No	No	CC	5C	6C	7C
Yes	No	Yes	Yes	CB	5B	6B	7B
Yes	No	Yes	No	CA	5A	6A	7A
Yes	No	No	Yes	C9	D9	E9	F9
Yes	No	No	No	C8	D8	E8	F8

Figure 25. WCC hexadecimal codes

The sign-on panel data is now complete and can be sent to the display unit.

Repeat to Address Order

The Repeat to Address (RA) order stores a specified alphameric or null character in up to 480 buffer locations, starting at the current buffer address and ending at (but not including) the specified stop address. The current buffer address becomes the stop address. You specify the stop address immediately following the RA order, just as you specify an address after an SBA order. After the stop address you specify the character that you want repeated. Symbolically this appears as:



RA is 3C in hexadecimal. RA can repeat null characters and can erase selected parts of the screen. You may also use it to repeat any other character. To put a row of asterisks under the last title in the sign-on panel, after the DC statement for "YOU MAY PRESS THE ENTER KEY" you specify an SBA for R12, C1. The RA order should repeat the asterisk character to location R1, C1 (the address after the last *). This is noted on the layout form as shown in Figure 26.

The order in the example is coded as:

```
DC  X'3C4040'

DC  C'*'
```

If you want to delete a field already on the screen, you can repeat the "null" character to delete it.

ITEM	DISPLAY PRINTER ROW	DISPLAY PRINTER COL	BUFFER ADDRESS DEC	BUFFER ADDRESS HEX	OR-DERS		ATTRIBUTE PROT	NO.	HI INT	SEL PEN DET	NON DISP PRT	MDT ON
1	02	11		40F2	SBA							
					SF	Att	✓					
2	04	02		C1F9	SBA							
					SF	Att	✓					
3	06	01		C3C8	SBA							
					SF	Att	✓		✓			
4	06	07			SF	Att						
					IC							
5	06	25		C360	SBA							
					SF	Att	✓		✓			
6	06	35			SF	Att						
7	07	01		C3F0	SBA							
					SF	Att	✓		✓			
8	07	16			SF	Att		✓				
9	07	23		C4C6	SBA							
					SF	Att	✓					
10	10	03		C5A	SBA							
					SF	Att	✓					
11	11	05		C6D4	SBA							
12	01	01			RA	*						

Figure 26. Example of RA order

The Operator's Response

When the sign-on panel is displayed, the operator responds by entering name, location, and serial number as shown in Figure 27. As the operator keys this information, the entered data characters are stored in the display unit's buffer and are displayed as part of the panel. Data that is entered in a nondisplayable field is stored in the buffer but does not appear on the panel.

When the operator finishes entering the requested sign-on data, he indicates the end of this operation by pressing the ENTER key, which sends the following information to your program:

- An attention code to identify that the ENTER key was pressed
- The address of the cursor's location
- The starting addresses of every modified field, followed by the data in the modified fields.

Figure 28 shows this sequence of input data, which is explained below.

Attention Identifier (AID)

The Attention Identifier (AID) is a hexadecimal code. By identifying this code, your program can determine in which of 20 possible ways the operator contacted the program and determine what request is being made. For example, pressing the ENTER key requests "Please enter this data"; pressing the CLEAR key requests, "Please erase my input so I can start over."

The AID code is always the first code received from the display unit by your program. The hexadecimal codes for all AID codes are shown in Figure 29.

For a READ Modified, the AID code is followed by the cursor address, which is the hexadecimal code for the row and column location of the cursor when the operator contacted your program.

```
                SIGN-ON PROCEDURE

   PLEASE ENTER YOUR SIGN-ON INFORMATION

NAME: JOHN SMITH          LOCATION: BOSTN

SERIAL NUMBER: 963981

      WHEN ALL INFORMATION IS COMPLETE
      YOU MAY PRESS THE ENTER KEY
```

Figure 27. Sign-on panel with operator's input

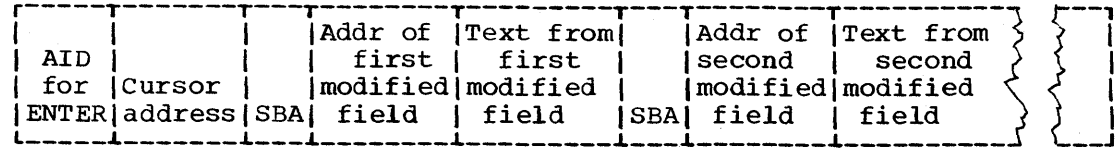


Figure 28. Input data sequence

ATTENTION IDENTIFICATION (AID) CONFIGURATION

AID VALUES FOR TEXT READ

Character	EBCDIC Hex	Operator Action
–	60	No action by display operator
Y	E8	No action (printer)
'	7D	ENTER key depressed
1	F1	PF key 1 depressed
2	F2	PF key 2 depressed
3	F3	PF key 3 depressed
4	F4	PF key 4 depressed
5	F5	PF key 5 depressed
6	F6	PF key 6 depressed
7	F7	PF key 7 depressed
8	F8	PF key 8 depressed
9	F9	PF key 9 depressed
:	7A	PF key 10 depressed
#	7B	PF key 11 depressed
@	7C	PF key 12 depressed
=	7E	Immediately detectable field selected
0	F0	TEST REQUEST key depressed
W	E6	Data transferred from card reader

AID VALUES FOR SHORT READ

—	6D	CLEAR key depressed (screen cleared)
%	6C	PA1 key depressed
>	6E	PA2 (cancel) key depressed
,	6B	PA3 key depressed

Figure 29. Attention identifiers (AID) in hexadecimal codes

Input Data

All the modified fields from the panel follow the AID code and the cursor address. A modified field is any field whose attribute character has the MDT on. A modified field can be one that was modified by the operator or one that was defined by you in your program with the MDT on in its attribute character.

When any character location in an input field is modified by the operator, the MDT in the attribute character for that field is automat-

ically turned on. An input field is not necessarily a modified field. If the operator made no entry in the SERIAL field, for example, only his name, location, and the date would be sent as modified fields to your program.

The display unit sends all the data in a modified field except nulls. When an operator finishes an operation, the display unit reads through the buffer for every attribute character whose code indicates its MDT is on. Each time one is found the display unit provides an SBA code and the starting address (the attribute character's address plus one) of the modified field. The SBA code identifies to your program that an address follows. It is the same X'11' code that you coded in your panel to identify the starting locations of the panel's text.

SBA Codes

SBA codes identify the incoming data by cross-referencing it to the correct input field.

For the sign-on panel, your program knows that row 6, column 8 (X'C34F') is the start of the name input field. When it receives the first SBA code (X'11'), it checks the address that follows to see if it is (X'C34F'). If it is, your program knows the text that follows it (until the next SBA code) is the operator's name and can process the input accordingly.

The first part of the input from the sign-on panel is as follows:

The hexadecimal codes are:

7D = the AID code for the ENTER key (Refer to Figure 29)

C4C6 = the cursor address R7, C23. The cursor is at the next character location after the entered serial number.

11 = the SBA (Set Buffer Address) order code which tells the program that the next 2 characters are addresses. (Refer to Figure 21)

C34F = the location (R6, C8) where the following text is located on the panel.

JOHN SMITH ... = the first modified field containing the operator's name.

PROGRAM ATTENTION KEYS

Program Access (PA) Keys

Each 3270 keyboard has at least one program access (PA) key that the operator can use to request program attention without sending any input data.

The AID codes for the PA keys are shown under a separate heading in Figure 29, because they are not followed by input data even though there may be modified fields on the panel when a PA key is pressed. All four short read codes consist of just the AID code.

Your program should use these keys for operator requests for immediate action such as trouble alerts or requests for termination. For example, the assignment of PA keys might be:

PA1 -- Terminate current application
PA2 -- Return to starting (master) panel
PA3 -- Explain system message

Program Function (PF) Keys

Program function (PF) keys are an optional keyboard feature. Your program defines the function that each key requests when it is pressed by the operator.

There is a separate AID code for each PF key so that your program can quickly identify which key was pressed and consequently which function was requested. When a PF key is pressed, all modified fields on the panel and their addresses are sent with the AID code and cursor address, the same as the ENTER key. For this reason, a PF key can be a valuable time-saving device for the operator.

For example, the assignment of PF keys might be:

PF1 Return to previous panel

PF2 Clear (without using data) and repeat current panel

PF3 Set up next panel

PF4 Page forward

PF5 Page backward

PF6 Return to page #1

PF7 Print this panel

PF8 Total input figures

PF9 Send input to program X

PF10 Send input to program Y

PF11 Send input to program Z

PF12 Operator requests assistance or explanation for this panel

SELECTOR PEN INPUT AND OUTPUT

Positioning data for Selector Pen (optional feature) use and setting the attribute characters is the same as for any other type of data but the Selector Pen has additional data-stream requirements.

Selector Pen Field Format

A field for selector pen operations must be defined as shown in Figure 30.

The attribute character, the designator character, and displayed alphameric characters must be on the same line. If the field is longer than one line, only those characters on the same line as the attribute character can be detected by the Selector Pen.

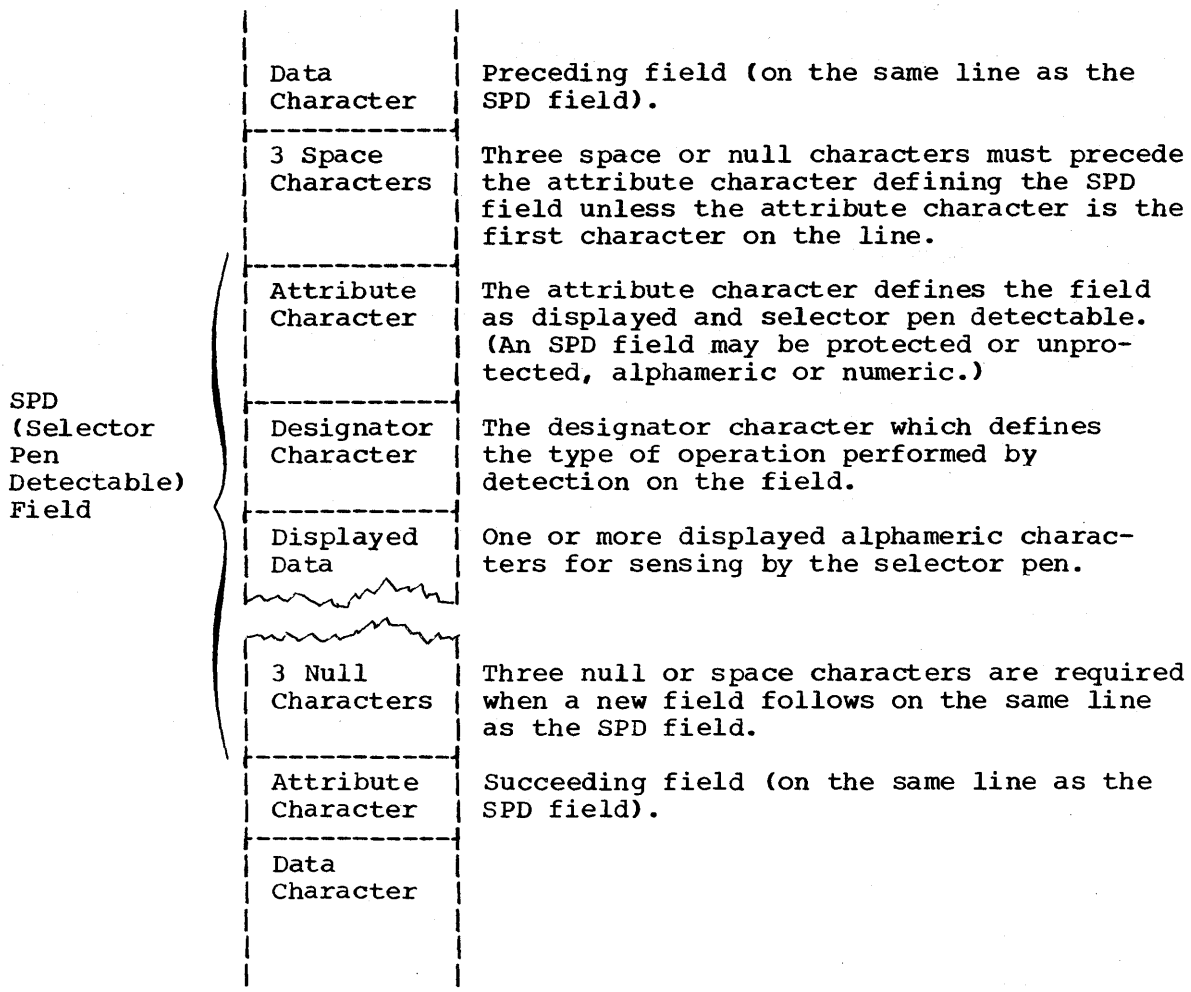


Figure 30. Definition of field for selector pen operation

Designator Characters

Designator characters define two types of Selector Pen fields: selection and attention. Each type of field performs a different selector pen operation.

The selection field is defined by a question mark (?) designator character. When the Selector Pen detects a selection field, the MDT bit in the attribute character for that field is set in the display buffer. Also, the designator character is automatically changed on the screen to a greater-than (>) sign to provide a visible indication to the operator that the detection was successful. If a mistake was made and the operator again detects on that same field, the > reverts to a ? and the MDT is reset. An attention field is defined by a space or null designator character. Probing an attention field is similar to using an ENTER key. The input information is released to be read by the computer when it is ready to do so.

Figure 31 shows a sample selector pen panel that illustrates some of the special input and output data stream considerations.

For output, an Erase/Write creates the panel. In the WCC you enable input and optionally reset the MDT's. Next you specify an SBA sequence to get you to R1, C7, followed by an SF with a protected attribute.

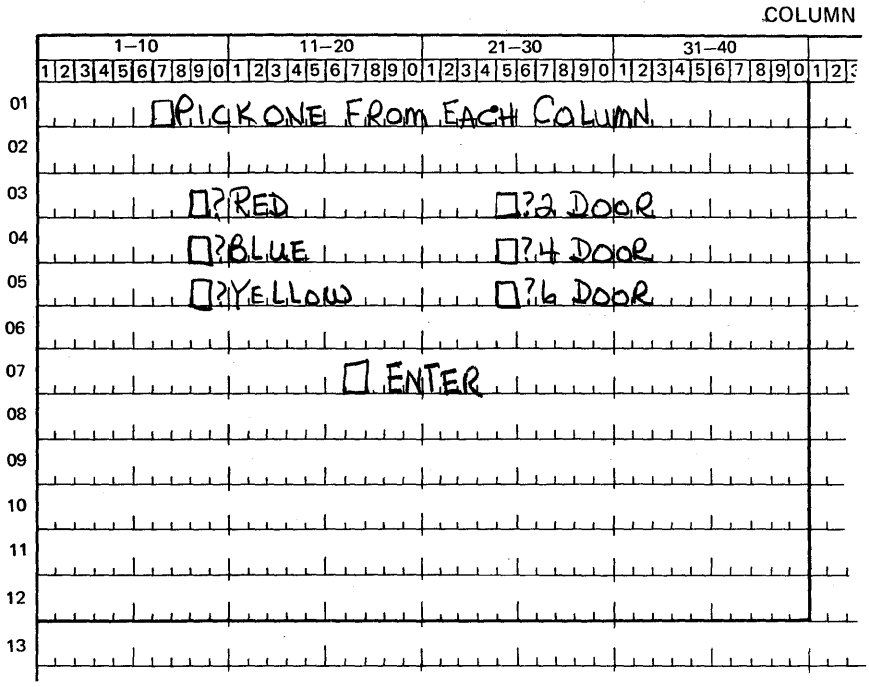


Figure 31. Sample panel for Selector Pen detection

This should be followed by the heading "PICK ... COLUMN" and another SBA to R3, C9. Then specify an SF order, followed by a protected (detectable fields may be protected) and detectable attribute. Next you need the designator "?" followed by "RED":

C	O	L	U	M	N	SBA	R3	C9	SF	P+D	?	R	E	D

An SBA after "RED" to R3, C25 provides more than the 3 required null characters and positions the SF, attribute, and designator for "2 DOOR". This type of sequence is repeated for the remaining fields to location R7, C28. The designator here must be a null or a blank so that probing the "ENTER" field releases the selection to the computer.

As the operator uses the Selector Pen, the program correlates the address of each Selector-Pen-detectable field with the data associated with it. To minimize line loading, channel loading, and buffer size requirements, only the addresses of Selector-Pen-detected fields are sent to the application program; the field data is not included.

To combine Selector-Pen-detectable input with keyboard input, use the keyboard to release the data to the computer by pressing the ENTER key or a PF key. Use of the Selector Pen to release the data, such as by probing "ENTER" in our example, transmits only the addresses of the fields in which the MDT was set.

In the example, if you pick RED and 4 DOOR the symbolic input would appear as follows:

Pen AID	Cursor ADDR	SBA	R3	C10	SBA	R4	C26	

Shortening transmissions by eliminating unnecessary data requires some caution. If you design a penel requiring both pen selection and keyboard entry, do not put an attention designator (blank or null) on the panel. An attention designator after keyboard entry transmits only the address of the keyboard input field and causes the loss of its contents. Not having an attention designator on the panel assures you that an ENTER or PF key will be used and the modified field contents will be transmitted (and the words "RED" and "4 DOOR" in the example).

THE RELATIONSHIP OF ONE DATA STREAM TO ANOTHER

The examples used so far have assumed that you started with a blank screen and that you built the entire panel into your data stream with ERASE or WRITE commands. This approach may lead to tedious work and lengthy data streams, which you can avoid if the panel you wish to display differs only slightly from the one that is presently displayed.

MODIFYING EXISTING PANELS

Suppose the displayed panel is the sign-on panel in the previous sections. If the operator keys an invalid serial number, you may wish to notify him of his error and request reentry of the serial number field only. You could create a new error message panel, write it to the display, require that the operator acknowledge its receipt, create a special serial number entry panel, write it, and finally read the corrected serial number. A better way might be to use the existing sign-on panel.

After the operator has keyed the data and it has been read into the computer, the screen appears as shown in Figure 32.

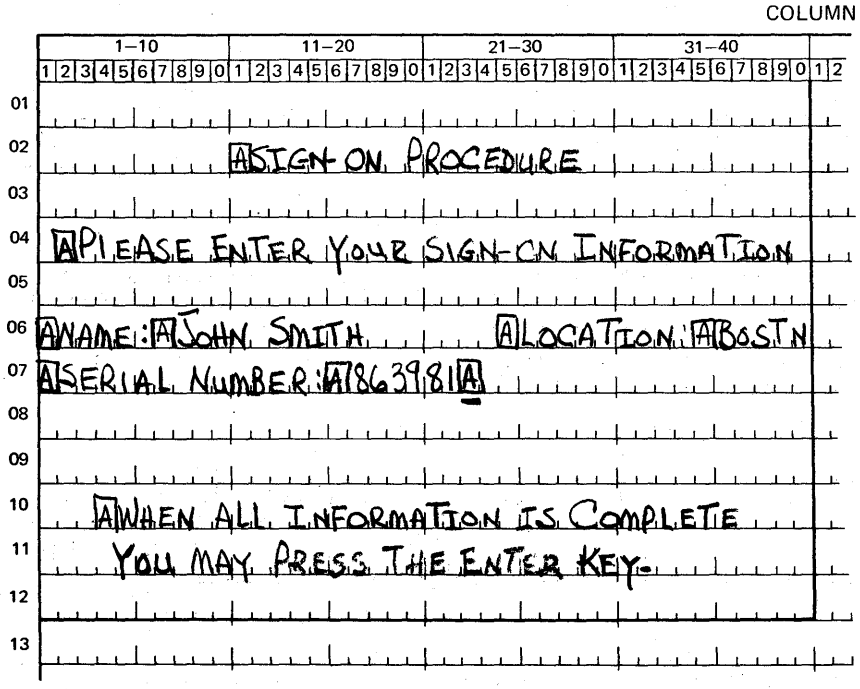


Figure 32. Modifying an existing panel -- basic panel

You would like the screen to look like Figure 33.

Most of the information you want displayed is already there. An Erase/Write would clear the screen and require writing a data stream containing all the information for the new panel.

You could use a Write command which modifies existing data in the 3270's buffer.

To change the panel in Figure 32 to look like Figure 33, you would:

1. Position the cursor at R7, C17;
2. Replace the message beginning at R10, C5 with the error message;
3. Change the attribute at R10, C4 to high intensity for the error message.

To do this the right side of your panel layout for the error panel might (in abbreviated form) look like Figure 34.

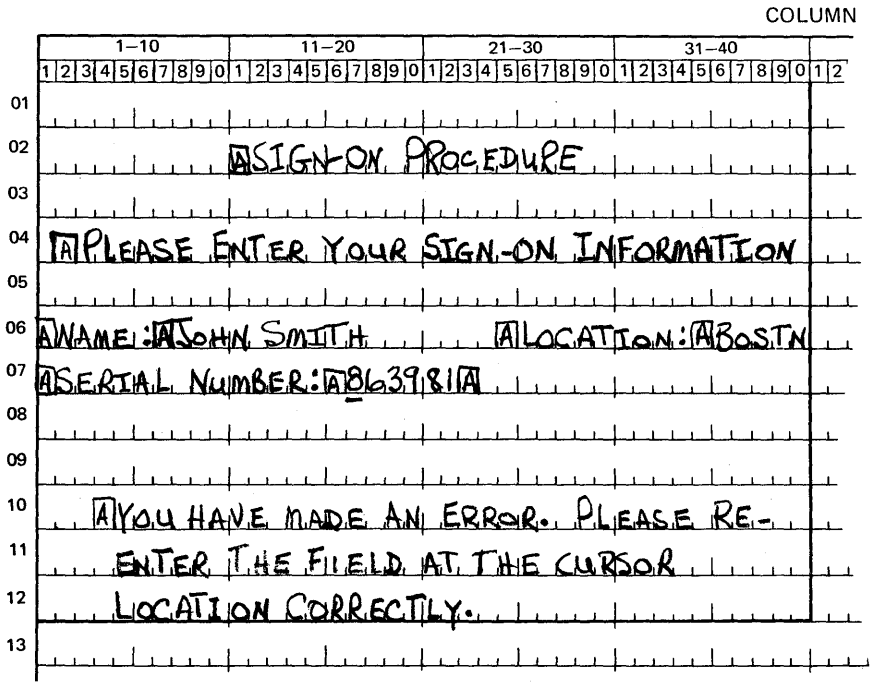


Figure 33. Existing panel with error message

ITEM	DISPLAY PRINTER ROW	DISPLAY PRINTER COL	BUFFER ADDRESS DEC	BUFFER ADDRESS HEX	ORDERS		ATTRIBUTE PROT	ATTRIBUTE NO.	ATTRIBUTE HI INT	ATTRIBUTE SEL PEN DET	ATTRIBUTE NON DISP PRT	ATTRIBUTE MDT ON
1	07	17			SBA							
					IC							
2	10	04			SBA							
					SF	Att	✓		✓			
		"LINE	1		OF	ERROR		MESSAGE"				
3	11	05			SBA							
		"LINE	2		OF	ERROR		MESSAGE"				
4	12	05			SBA							
		"LINE	3		OF	ERROR MESSAGE"						

Figure 34. Panel layout changes for error message (keyed to text)

ITEM 1. Repositions the cursor to R7, C17.

ITEM 2. Changes the attribute to R10, C4 to protected and high intensity. If the designer of the sign-on panel had combined the original field at this location with the previous field, the field "SIGN-ON PROCEDURE," and the following field by omitting the attributes at R10, C4, R2, C11, and R4, C2, (as you saw under the discussion of attributes) the result would be undesirable. The attribute placed at R10, C4 would begin a new field. This would not affect the preceding field but, by wraparound, would cause "SIGN-ON PROCEDURE" and "PLEASE ... INFORMATION" to be high intensity even though they were neither intended to be so, nor were they rewritten. For this reason you should adhere closely to the "Field Concept" and not combine fields unless necessary for efficiency, and be extra careful if you must combine fields to avoid undesired results.

ITEM 3. Repositions the data flow to correctly place the second line of the error message. 3 characters are used instead of 6 null characters.

ITEM 4. Repositions the data flow for the third line of the error message.

Since there are two different types of Write commands, you must tell the I/O portion of your program which type to use for the data stream. You may want to indicate the type you want in a comment in the data stream. It is suggested that you establish some convention for indicating command selection by discussing it at your installation with the people responsible for the I/O portion of the program.

Write Control Character (WCC)

When the operator presses the ENTER key after filling in the sign-on panel the keyboard automatically locks, as it always does after an operator-initiated input operation. One of the functions of the Write Control Character, which was also discussed under "Coding the Panel," is to enable the keyboard. You should now decide if you want the WCC at the beginning of the error panel data stream to enable the keyboard for the operator. While it is normal to enable the keyboard at this point, you may not want to do it here. It might be better for the operator to press the RESET key, calling further attention to the error panel.

In Figure 32, assume that the operator now keys "9" and presses the ENTER key. The "9" corrects the original entry error and the serial number field now reads "963981". What goes into the computer? The prior discussion of input data streams shows the basic format, but which fields can you expect? You know that the serial number input field will be received in its entirety, since keying the "9" caused the 3270 to turn on the MDT for this field, and any field which has been modified is transmitted in its entirety (except nulls).

The input field MDTs for NAME, LOCATION, and SERIAL NUMBER were all turned on by the data entered into those fields in the sign-on panel. While an Erase/Write resets all MDTs, a Write does not; therefore, if you do not reset them, all 3 input fields are returned to the computer. Since not all of them have changed, all 3 should not return to the computer. You may specify in the WCC that all MDTs in the device are reset "off" or "not modified"; you should do so here.

You may also want to sound the Audible Alarm, if you have one, with the error panel. A WCC to reset the keyboard, reset all MDTs, and sound the alarm is defined as DC X'C7' (see Figure 25). You can now use the Write command to change the sign-on panel into the error message panel.

CAUTION: As you have seen, the Write command allows you to modify an existing screen image while retaining all or a portion of the information already displayed. With the Write command, you can treat the 3270 as a typewriter-type terminal and write your panel line by line or field by field. Using multiple Write commands to create a panel, while technically possible, may create problems.

The operator may start keying data into the panel before you have finished writing it all to the screen. You can prevent this by not enabling the keyboard (see WCC above) until the last Write in the series.

Using successive Write commands to accomplish what one Write command can do is an inefficient use of the communication line on remote 3270s, and unnecessary I/O overhead on local 3270s. In addition, in both local and remote use, successive Write commands without an intervening READ may result in a "blinking" effect while you build up the panel. "Blinking" may be annoying to the operator.

Wherever possible, use a single Write command to avoid the inconveniences noted above.

Erase Unprotected to Address

The error panel shown in Figure 33, left the erroneous serial number displayed. All the operator had to do was key over the incorrect digits. This may sometimes be confusing. You might instead want to erase only the serial number input field as shown in Figure 35.

Begin again with the desired WCC. Place the cursor at R7, C17 with an SBA to R7, C17, followed by an IC order. To erase what was entered in the serial number input field, use the Erase Unprotected to Address order, or EUA (watch the sequence of these letters so you do not confuse them with EAU, which is discussed next). EUA inserts nulls (erases all unprotected positions) from the current buffer address to, but not including, the specified stop address. The current buffer address becomes the stop address. The format of the order is similar to an SBA; the code for the order itself (X'12' for EUA) is immediately followed by a row and column address.

At the first position to be erased (a result of prior SBA and IC) you should include an EUA order. For a terminating address, you may use R7, C23 (the first position after the last to be erased). There is a better stop address, however. Since EUA only erases unprotected fields, and since the field beginning at R7, C23 is protected, it can be included in the range covered by the EUA. If R10, C4 is used as the stop address, nothing additional is erased, but you can then write the next attribute without using an SBA, saving three characters of transmission (see Figure 36). The current buffer address is the stop address. Any data or an SF order that follow go into the buffer at this address.

EUA erases all unprotected fields within its range and can erase multiple fields. Suppose you wanted all three input fields erased on the error panel, as shown in Figure 37.

First place the cursor at R7, C17, then "back up" with an SBA to R6, C8 (the name input field) before issuing the EUA to R10, C4 (see Figure 38).

You could have started at R6, C8 with an SBA to R6, C8, followed by the EUA to R10, C4. Sometime later in the data stream you would have to "back up", probably with an SBA to insert the cursor.

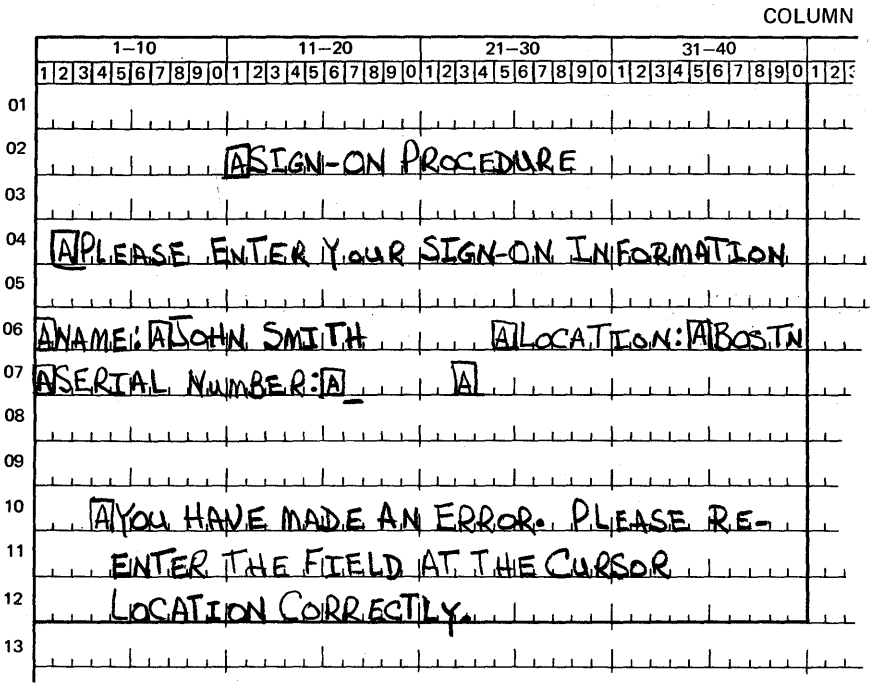


Figure 35. Error message panel with serial number field erased

ITEM	DISPLAY/PRINTER ROW	DISPLAY/PRINTER COL	BUFFER ADDRESS DEC	BUFFER ADDRESS HEX	ORDERS		ATTRIBUTE PROT	ATTRIBUTE NO.	ATTRIBUTE HI INT	ATTRIBUTE SEL PEN DET	ATTRIBUTE NON DISP/PRT	ATTRIBUTE MDT ON
1	07	17			SBA							
					IC							
	10	04			EUA							
2					SF	AH	✓		✓			
	"LINE 1	OF	ERROR		MESSAGE"							
					•							
					•							
					•							

Figure 36. Example of EUA use

Erase All Unprotected Command

In the preceding example, you wanted to erase all unprotected data, reposition the cursor, and add some new titles to the sign-on panel to make it an error panel. The Erase All Unprotected command:

1. Clears all unprotected fields to nulls
2. Resets MDTs in all unprotected fields
3. Unlocks the keyboard
4. Resets the AID (see "Program Attention Keys")
5. Repositions the cursor to the first character of the first unprotected field

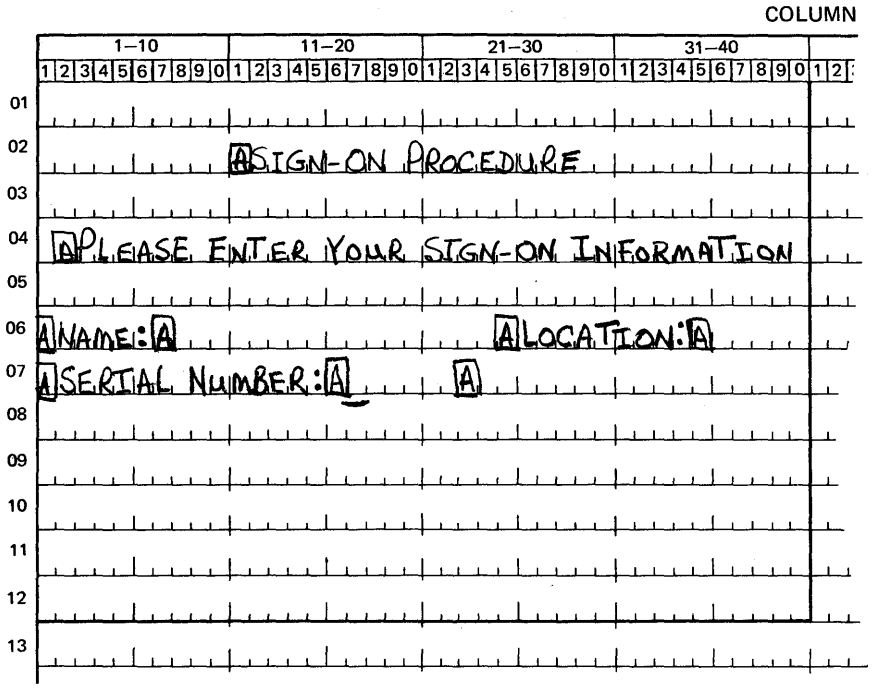


Figure 37. Sign-on panel with three erased fields

ITEM	DISPLAY PRINTER ROW	DISPLAY PRINTER COL	BUFFER ADDRESS DEC	BUFFER ADDRESS HEX	ORDERS		ATTRIBUTE PROT	ATTRIBUTE NO.	ATTRIBUTE HI INT	ATTRIBUTE SEL PEN DET	ATTRIBUTE NON DISP PRT	ATTRIBUTE MDT ON
01	07	17			SBA							
					IC							
	06	08			SBA							
	10	04			EUA							
					SF	AH	✓		✓			
					•							
					•							
					•							

Figure 38. Erasing multiple fields with EUA

This command appears to do what you want (it even does what the WCC would have done), but it does not write any data to the screen. You could tell the I/O control section that you want an Erase All Unprotected command issued before the Write command. Then you would just write the new titles in their proper positions. You have then issued two commands to create one panel. What, then, is EAU for? It logically resets the panel for repetitive input using the same panel. Do not use EAU to change panels.

Data Entry Example: You can use the EAU command to change a sign-on panel slightly and make it a data entry panel. Then the operator just keys in NAME, LOCATION, and SERIAL NUMBER for the first employee. If an error is made, an error panel is shown. If there is no error, you may want to clear the input, reset the MDTs, unlock the keyboard, and reposition the cursor.

The data entry panel might appear as shown in Figure 39.

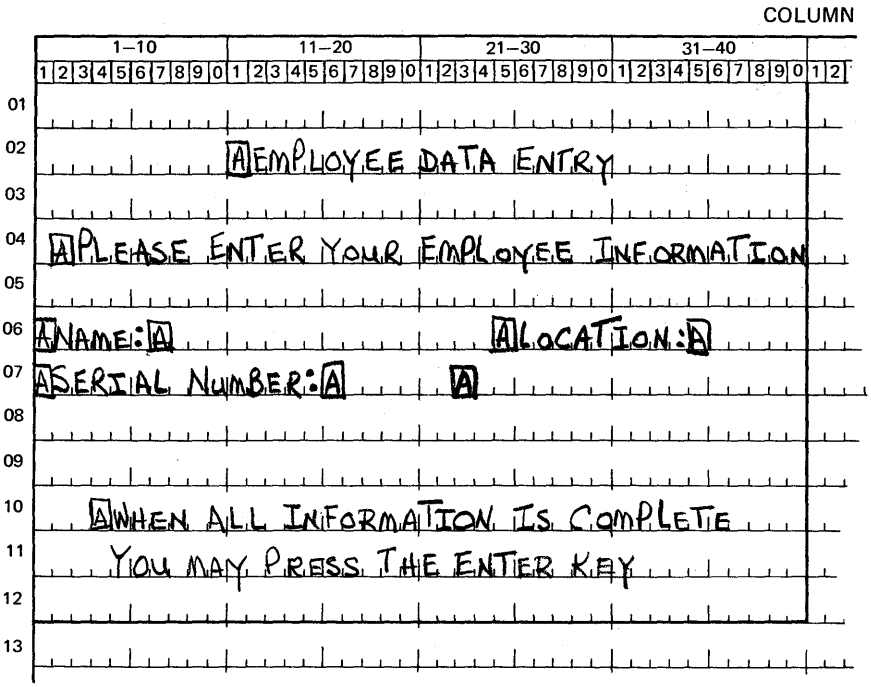


Figure 39. Example of data entry panel

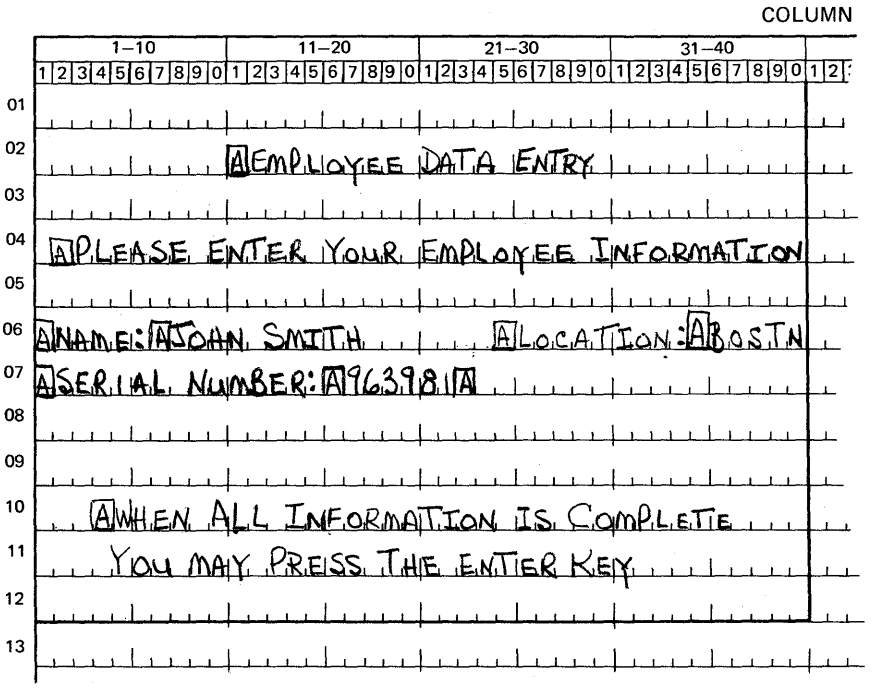


Figure 40. Data entry panel with entered data

The operator keys JOHN SMITH, presses TAB, keys BOSTN, presses TAB, keys 963981, and presses ENTER (Figure 40).

You simply send the 3270 an EAU command to unlock the keyboard. The operator then sees the same panel as in Figure 39. The operator may now key data for the next employee. You have used your knowledge of what is displayed already to arrive at the next panel or recreate the present panel.

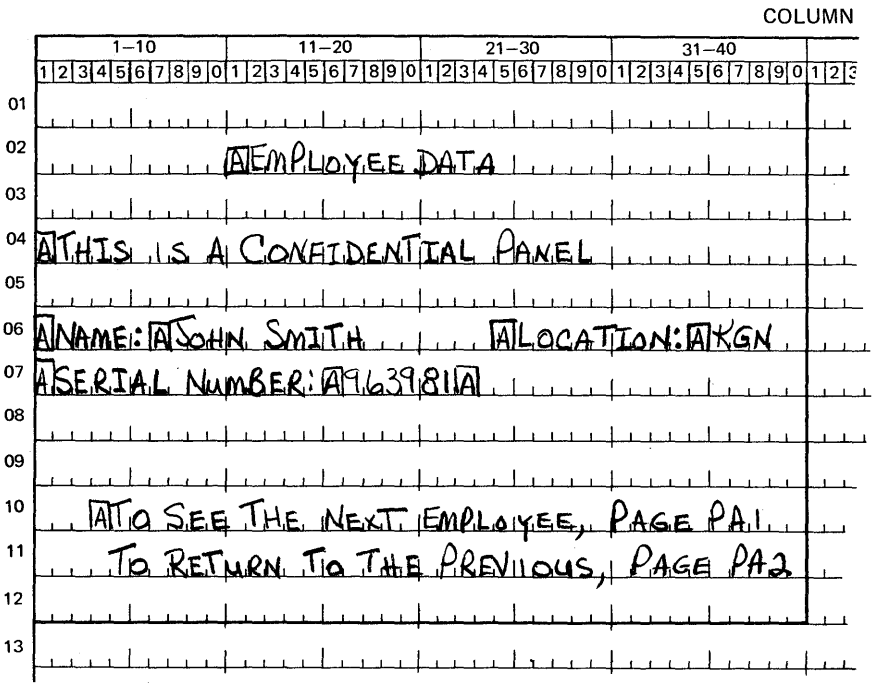


Figure 41. Employee data panel

Repetitive Output

In the data entry example you used one panel repetitively for input of employee information. You can reverse the requirement and design an employee data screen. For this example assume the application is inquiry with "browsing" capability. Assume also that the operator has previously used another panel to request the information for employee number 963981. The display might appear as shown in Figure 41.

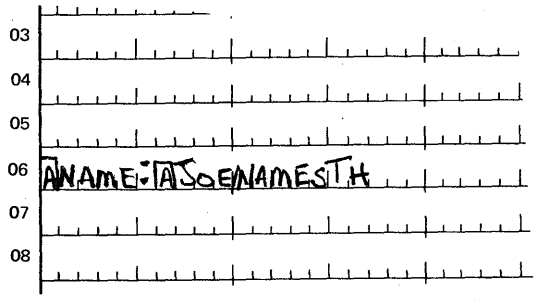
At the bottom of the panel the operator is instructed to use the PA1 key to see the next employee page, probably number 963981. The PA2 key is assigned to page backwards. Remember, PA keys are assigned by the program. Program access keys cause a short transmission; they do not even transmit the contents of changed fields. For an inquiry and browsing application, there should be no input. The PA key assures there is no input even if the operator changes one of the unprotected fields, so its use is preferred to the ENTER or PF keys.

Program Tab

The input fields in the previous examples are output fields in this example. You could designate them as protected, but you could not use another 3270 function called "Program Tab". The Program Tab (PT) order advances the current buffer address to the address of the first character location of the next unprotected field. When the PT order immediately follows an alphameric or null character (not another order) in the WRITE data stream (other than the character specified by the Repeat to Address order, which is discussed later), it also inserts nulls in all the character positions from the current buffer address to the end of the current field. The PT order can be used to page through the employee data file.

When the operator is ready to view the information for the next employee, he presses the PA1 key. Since you want to modify only the present panel, not erase it or blank the unprotected fields, you request a WRITE command with a WCC to unlock the keyboard. Since you are not sure

of the present buffer address you might begin with an SBA order to R6, C8 followed by the next employee name from the disk file - JOE AMES. Since this name contains fewer characters than JOHN SMITH, the screen would look like this if you did not clear the remainder of the field:



You must also place the location code at location R6, C36. You could use blanks after the name and an SBA sequence, or EUA with its associated address. Use PT instead. Insert a PT order after the "S" in "AMES". The single PT order clears the remainder of the unprotected name field to nulls and positions for the location code. PT should also follow the location code to position for the serial number. The data stream might look like this:

W	S										P					P						
C	B	R6	C36	J	O	E		A	M	E	S	T	K	N	G	S	T	T	9	3	9	8	2
C	A																						

The screen would appear as shown in Figure 42.

As you have seen, you can write each new panel out in its entirety with an Erase/Write command. You should understand the relationship between the past data streams and the one you are building.

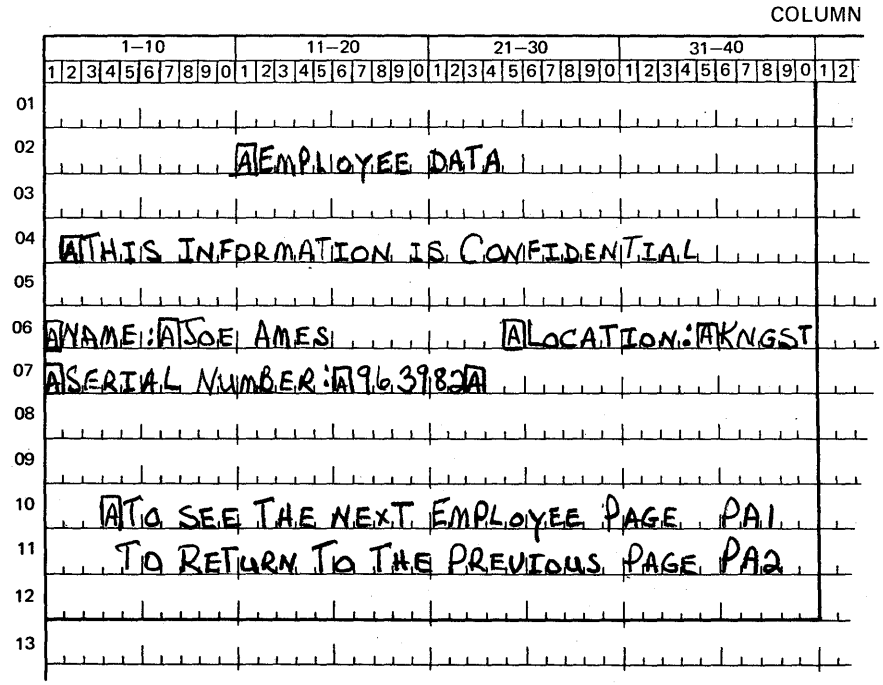


Figure 42. Panel defined with program tab

A screen management program module is a set of subroutines physically separate from application programs and from the device management (line control) program module of an online 3270 system. Screen management is a logical intermediary between line control and application programs as shown in Figure 43. Support functions in a screen management program may reduce the amount of detail work required by the application programs, and effectively use the features of the 3270 system.

Even though screen management can be thought of as a logical extension of device management, it should be programmed as a separate physical entity with user-defined standard interfaces to the application programs and the line control module. This structure allows screen management to be modified with little or no impact on application programs or the line control module.

Screen management may include:

- Decoding input data streams
- Dynamic building of output data streams
- Generating multiple I/O requests to the Line Control Module based upon a single request from an application program (i.e., WRITE then READ)
- Automatic paging; the application program passes multiple pages to screen management, which asks the line control module to write a particular page to a display, depending on the display operator's request
- Automatic copying (providing a hard copy of a display image at the operator's request)

The COPY command supports data movement between any type of device attached to the same 3271 control unit: display to display, display to printer, printer to display, and printer to printer. To prevent copying information from an unauthorized device, the 3271 provides a program-controlled copy-lock for devices attached to it. If the first position of a device buffer contains an attribute character with the protected option and the second buffer position contains a null character, the 3271 control unit rejects any attempt to copy from that device.

DECODING AND GENERATING DATA STREAMS

The data streams sent between application programs and the 3270 contain unique orders that request particular operations by the 3270 displays and printers. Generalized subroutines can be written to assist the application programmer's interface with the 3270 system and a standard interface can be built to simplify maintenance of online programs.

This section discusses several approaches to the development of a screen management module whose functions can be used by the application programmer to prepare output data streams and to decode input data streams. The approaches demonstrate how some 3270 device-dependent considerations can be removed from the application programmer's responsibility. The different techniques for 3270 input or output data stream manipulation can be used in various combinations to suit the needs of the installation.

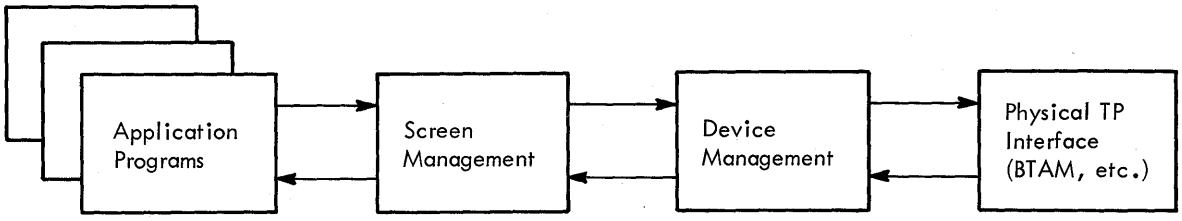


Figure 43. Relationship of screen management to device management and application programs

This discussion assumes that the user-written device management routines (line control) discussed in Section 3 make the local and remote 3270 transparent to the application program. Therefore, discussion of data streams in this section ignores all header data in the input stream to and including the AID character, and all header data in the output stream up to but not including the WCC.

DECODING READ MODIFIED INPUT DATA STREAM

A READ Modified command for a display station with a formatted screen (a screen with at least one attribute character defined) produces a data stream consisting of the data from each field whose modified data tag has been turned on (either by program control or by data entered in the field). Each transmitted data field is preceded by the 3270 buffer address where that data is located on the display. The order of the fields transmitted from the screen is from left to right for each line, starting at the top of the screen and ending at the bottom of the screen. All null characters in a transmitted field are stripped out by the control unit during transmission.

The data stream, ignoring the header information up to and including the AID character, appears as:

SBA	A1	A2	DATA	SBA	A1	A2	. . .

If the data entered in a field is variable-length or if a field can be skipped by the terminal operator, the data from a particular field on a given panel can appear in a different location within the data stream for each set of operator input. A READ Modified command produces a variable-length data stream of fixed-length fields and variable-length fields concatenated together.

Each two-character screen address in the data stream is immediately preceded by a Set Buffer Address (SBA) order. The detection of each SBA order in the data stream identifies the next two characters in the stream as a 3270 screen address and also indicates the end of the preceding data field. The System/360 and System/370 translate and test instruction (TRT) can be used to scan the data stream and to stop at each main storage address containing an SBA order. If the detected main storage address of the current SBA order is known, the following calculations can be performed for a given data stream:

SBA(1), ADD(1A), ADD(1B), DATA FIELD(1),

SBA(2), ADD(2A), ADD(2B), DATA FIELD(2),

SBA(3),

The numbers in parentheses are used as subscripts to provide unique identification:

1. The length of data field(1) = (Address of SBA(2) - Address of SBA (1)) -3.

2. The two-character 3270 screen address of data field(1) can be found at the address of SBA(1) +1.

3. The length of data field(2) = (Address of SBA(3) - Address of SBA (2)) -3.

4. The two-character screen address of data field(2) can be found at the address of SBA(2) +1.

The two-character 3270 screen address as it appears in the input stream does not provide a direct decimal or binary numeric value that can be used to calculate the relative position in the 3270 buffer from which the data was read. However, you can use the following routine to convert the 3270 address as it appears in the input data stream to a binary value which directly indicates the position (relative to zero) of the data in the 3270 buffer.

Assume that R3 contains the address of SBA(1) and that R4 and R5 are work registers. R5 will contain the result at the end of the routine.

```
ADDCNVRT   EQU   *
           SR    R4, R4          CLEAR WORK REG
           SR    R5, R5          CLEAR WORK REG
           IC    R4, 1(R3)       GET FIRST ADDRESS CHAR (ADD (1A))
           N     R4, = F'63'     TURN OFF ALL BITS EXCEPT LAST SIX
           IC    R5, 2(R3)       GET SECOND ADDRESS CHAR (ADD (1B))
           N     R5, = F'63'     TURN OFF ALL BITS EXCEPT LAST SIX
           SLL   R4, 6           SHIFT FIRST ADDRESS SIX BITS TO THE LEFT
           AR    R5, R4          ADD THE RESULTS TOGETHER
```

By using the above technique, several approaches may be developed to a general purpose subroutine that decodes the variable field length data stream for the application program, and returns the data in a more easily processed format.

Nonselector Pen Data Streams

DISPLAY BUFFER IMAGE TECHNIQUE: You can use the display buffer image technique to return to the application program a main storage buffer area the same size as the display buffer (480 or 1920). The data read from the display is placed in the same relative position in the main storage buffer as it occupied in the display buffer, with all other positions in the returned buffer cleared to blanks.

For this technique, use the TRT instruction and the 3270 address conversion routine. You must know the relative locations in the display buffer where data can be entered by the operator, so that the decoded buffer can be processed when returned by the mapping subroutine. The completed layout sheet for the panel in which the operator enters data will give you the required addresses relative to the respective buffers.

Using the image technique, all data received from the 3270 is left-justified in its respective fields. This has no effect on fixed-length fields, variable-length alphameric fields (which are normally left-justified), or omitted input fields. However, you must be aware of variable-length numeric fields where the operator can omit leading zeros.

Although the image technique requires little main storage for the mapping subroutine, main storage can be wasted if the routine returns a 1920-character buffer with little data. To help overcome this problem, the decoding routine can pass back to the application program, a field at the beginning of the buffer. The field indicates the total length of the buffer, which allows the decoding routine to use a buffer area just large enough to accomodate the relative address of the last data field read.

MAPPING FROM A TABLE OF REQUIREMENTS: This mapping technique requires a table assembly for each unique input panel that the mapping subroutine decodes for the application program. The table provides information to the subroutine so that the input data stream in one main storage buffer can be decoded a field at a time and moved to a specified relative offset in another main storage buffer (the target buffer) according to the directions assembled in the table. The preassembled table could be used to specify the following information to the mapping subroutines:

1. The 3270 buffer address preceding each field, which could be read from a particular panel. This is the buffer address as it appears in the data stream which corresponds to the first data position in a field, not to the buffer location of the attribute character which defines the field. Any data fields in the 3270 input stream which do not have a matching buffer address in the table would be ignored by the typical mapping routine using the table approach.

2. An offset relative to zero which provides the starting position of each field in the target buffer. This information allows the application programmer to order the fields in the target buffer in a sequence which may or may not agree with the field sequence in the transmitted data stream.

3. A value which indicates the maximum length of each field in the target buffer. This information allows the mapping routine to truncate data stream fields which are too long for the target fields. The maximum field length value is also required if the mapping routine supports right-justification of fields during mapping.

4. A flag byte consisting of bit switches which could indicate:

   - Whether left justification with low-order blank padding is requested
   - Whether right justification with high-order zero fill is requested
   - Whether the field should be translated to ensure uppercase characters only
   - Any additional functions the installation wishes to implement in the mapping routine

Figure 44 shows some typical logical contents of the table. The order of the elements within each table entry is optional.

Assume that you map the following input data stream in hexadecimal using the sample table in Figure 44:

'1140D4F1F2F31140E8818283848511C1C6E385A7A3'

The following target buffer, also in hexadecimal, would be returned to the application program:

'C1C2C3C4C54040404040F0F0F1F2F3E385A7A34040'

```
TABLE     DS 0H
ENTRY1    DC X'40D4'   ACTUAL 3270 ADDRESS FOR POS 20
          DC H'10'     RELATIVE OFFSET IN TARGET BUFFER
          DC HL1'5'    MAX FIELD LENGTH OF TARGET FIELD
          DC X'80'     RIGHT JUSTIFY, NO TRANSLATE FLAG
ENTRY2    DC X'40E8'   ACTUAL 3270 ADDRESS FOR POS 40
          DC H'0'      RELATIVE OFFSET IN TARGET BUFFER
          DC HL1'10'   MAX FIELD LENGTH OF TARGET FIELD
          DC X'40'     LEFT JUSTIFY, TRANSLATE FLAG
ENTRY3    DC X'C1C6'   ACTUAL 3270 ADDRESS FOR POS 70
          DC H'15'     RELATIVE OFFSET IN TARGET BUFFER
          DC HL1'6'    MAX FIELD LENGTH OF TARGET FIELD
          DC X'00'     LEFT JUSTIFY, NO TRANSLATE FLAG
ENDOLIST  DC X'FF'     END OF LIST INDICATOR
```

Note: 3270 buffer addresses in the table are shown relative to buffer location zero; relative offsets in the target buffer are shown relative to zero.

Figure 44. Table of requirements

This approach to mapping makes the application program's input processing routine device-independent.

Instead of the mapping table, you could write a macro instruction to prepare the table, which would convert written requests into the proper machine language constants.

A typical format for a macro instruction to build the sample table shown in Figure 44 might be:

```
MAP   NAME=TABLE,MODEL=2
MAP   ADD=(1,21),OFFSET=11,MAXL=5,JUST=RIGHT
MAP   ADD=(1,41),OFFSET=1,MAXL=10,JUST=LEFT,TRAN=YES
MAP   ADD=(1,71),OFFSET=16,MAXL=6,JUST=LEFT
```

Note: The ADD parameter specifies the 3270 buffer in row and column notation relative to one. For example, buffer position zero equals row 1, column 1. The offset values are expressed relative to one. The macro instruction can have default options; for example, if JUST=RIGHT is not specified, JUST=LEFT can be assumed.

The following example shows the logic flow for a table-driven input mapping technique:

1. Find the 3270 buffer address of a data field to be processed in the input data stream using the TRT instruction.

2. Determine the length of the data field in the data stream using the techniques discussed in this section.

3. Search the table of requirements, using the 3270 buffer address found in step 1 as a search argument to find a matching entry.

4. Add the offset value from the entry found in the table to the starting address of the main storage map buffer, to produce the main storage address of the start of the receiving field.

5. If the length of the data field determined in step 2 is greater than the maximum field length value in the entry found in the table, go to step 10.

6. Check the flag byte in the entry found in the table. If left justification is requested, go to step 10. Otherwise proceed to step 7 for right justification.

7. Move zoned decimal zeroes to the receiving field, using the field starting address determined in step 4. Use the maximum field length value in the entry found in the table as the length for the move.

8. Develop a new main storage address for the start of the receiving field to accomodate the request for right justification. The right-justified starting address for the receiving field = (field starting address determined in step 4 + maximum field length value in the entry found in the table) - length of the data field in the data stream found in step 2.

9. Move the data field from the data stream to the main storage address developed in step 8, using the length of the data in the data stream determined in step 2. Return to the start of this routine to find the next data field in the data stream.

10. Move blanks to the receiving field using the starting address of the field as determined in step 4. Use the maximum field length value in the entry found in the table as the length for the move.

11. Move the data field from the data stream to the receiving field using the field address determined in step 4. Use the length of the data in the data stream (determined in step 2) as the length for the move.

12. Check the flag byte in the entry found in the table to determine if uppercase translation is requested. If it is not requested, return to the start of this routine to find the next data field in the data stream.

13. Translate the data in the receiving field to uppercase, then return to the start of this routine to find the next data field in the data stream. The translation can be done in two ways:

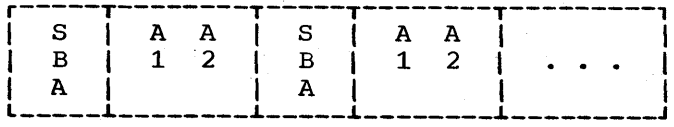
    - Use the TRANSLATE instruction with the translation table built to convert lowercase alphabetic characters to uppercase.

    - Use the OR instruction to place blanks in the field. This will change the DUP and FM characters. The FM appears as a ; on the screen, but appears in the data stream as X'1E'. It will be converted to a true ; that is, X'5E'. The DUP appears as a * on the screen, but appears in the data stream as X'1C'. It will be converted to a true * (X'5C').

Immediate Selector Pen Data Stream

When a READ Modified command is executed for a display station as a result of an immediate detection by the Selector Pen, the resulting data stream consists of address strings that identify which fields on the screen have the modified data tag set. No field data is transmitted in the data stream.

The data stream, ignoring the header information up to and including the AID character, appears as:



If the operator keys into a field and an immediate Selector Pen field is selected, the keyed data is not transmitted. However, if keyed data is entered by the operator, then delayed Selector Pen fields are selected and the ENTER key or a PF key is pressed, the address and data for all fields, whether selected or keyed, are included in the data stream.

You can use a subroutine to free the application program from determining which fields were selected on a panel. A table can be built which consists of the 3270 buffer addresses, giving the location of each selectable field on a panel. The mapping routine can then compare the addresses in the table, and return to the application program a list of indicators which identifies the selected fields.

The list of indicators can be returned to the application program. A string of one-position fields can be used, and each position indicate with a unique character that a field was selected. The first position in the returned list can be marked if a field in the data stream has the same address as the first element in the address table; the second position in the returned list can be marked if a field in the data stream has the same address as the second element in the address table. The application program can then determine which relative positions in the list have been marked, to determine which fields have been selected by the operator.

Because the input from a display using Selector Pen detection is a series of fixed-length addresses, the mapping routine can analyze the input stream and decode it.

For example, using the Selector Pen panel illustration in Figure 45, assume that the operator has selected the delayed-detectable fields located at row 5, column 10 and row 3, column 26 and the immediate-detectable field located at row 7, column 18. The input data stream transmitted in hexadecimal from the display would be:

11C1E911C2E911C4C1

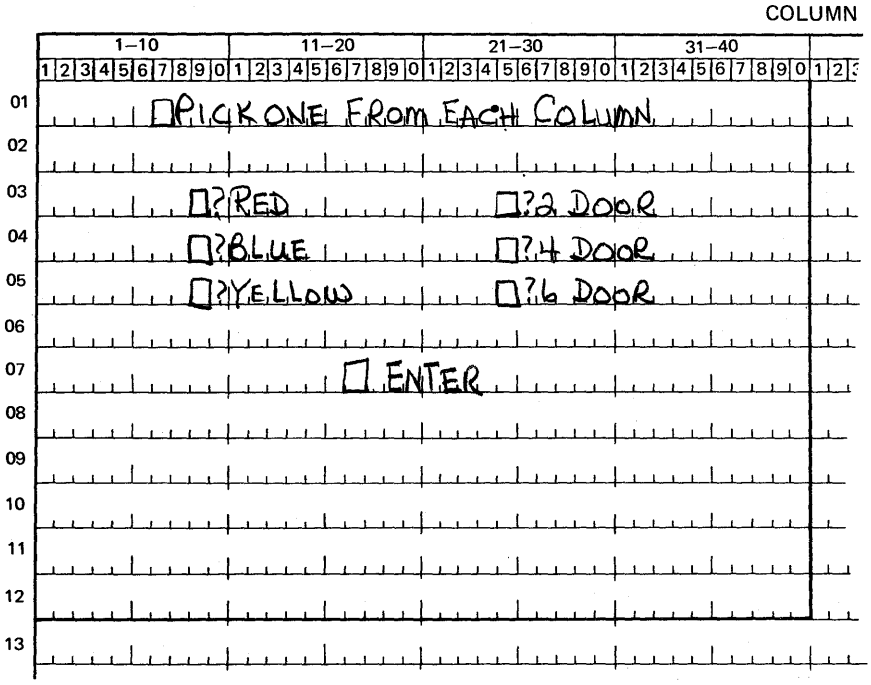


Figure 45. Example of Selector Pen panel

```
SELTABLE  EQU *         FOR MODEL 1 DISPLAY
          DC  X'C1D9'   ROW 3 COL 10
          DC  X'C1E9'   ROW 3 COL 26
          DC  X'C2C1'   ROW 4 COL 10
          DC  X'C2D1'   ROW 4 COL 26
          DC  X'C2E9'   ROW 5 COL 10
          DC  X'C2F9'   ROW 5 COL 26
          DC  X'C4C1'   ROW 7 COL 18
          DC  X'FF'     TABLE STOP INDICATOR
```

Note: The 3270 addresses used in the above table correspond to the buffer position of the Selector Pen designator character in a field, not to the location of the attribute character which defines the field.

Figure 46. Sample mapping table

Using the sample table in Figure 46, the mapping routine returns a list in hexadecimal to the application program:

406F40406F406F

This list indicates that the second, fifth, and seventh fields were selected. Note that the addresses of the selected fields appear in the data stream in the same sequence as the fields appear in the display buffer. When designing a selector pen panel by columns, the address of the field selected from the first column may not occur before the address of the field selected from the second column in the input data stream.

You can write a macro instruction similar to the one used to build the table in Figure 44 to build the selector pen table:

```
MAP     NAME=SELTABLE,MODEL=1
MAP     ADD=(3,10)
MAP     ADD=(3,26)
MAP     ADD=(4,10)
. . .
```

Mixed READ Modified Input Data Streams

When some keyed input and some delayed Selector Pen detection can occur in a panel during the same input operation from a display, you can use the table-driven mapping technique for non-Selector Pen panels. Specify the table elements so that all delayed Selector Pen fields have a maximum length of one character. The mapping routine places the first character from the appropriate data stream field in the target field. The first character in a delayed Selector Pen field that has been selected is always a (>); that is, X'6E'. The application program can examine the target buffer for that character in the proper target field to determine if the field has been selected.

BUILDING OUTPUT DATA STREAMS

The 3270 requires specific bit patterns for order sequences, control characters, and buffer addressing. The data streams can be prepared in several different ways. A data stream to build a static panel (a panel which will always be displayed in exactly the same manner) can be assembled in an application program as a set of data constants. A semi-dynamic panel, which may occasionally be modified or added to, can have the static portion assembled in the application program and have the program dynamically modify or add to the data stream. A data stream for

a dynamic panel (a panel with a high degree of change) must be created or assembled as a unit at execution. This section discusses how to reduce the considerations of device-dependency required to support static, semi-dynamic, and dynamic output data streams.

Static Data Streams

You can write macro instructions to simplify the preparation of static data streams for the 3270. One approach is to write a set of macro instructions in which each macro instruction prepares a single order sequence. Another approach is to write one macro instruction which can prepare all types of order sequences, but prepares only one sequence for each execution of the macro instruction in a program.

A sample macro instruction of the first type might be:

```
$MOD MODEL = 1 or 2
```

This macro instruction sets a global value so that the specified model number is used until another $MOD macro instruction is encountered. The model number is required to correctly calculate 3270 buffer addresses. The buffer address 'C2D5' represents column 4, row 30 for a Model 1 display, and column 2, row 70 for a Model 2 display.

$SBA (1,10) generates the SBA order sequence X'1140C9'

$SF (PROT,NUM,SKIP,MDT,HI,DET,NONDISP) generates an SF order (X'1D') followed by the appropriate attribute character defined by the options selected in parentheses. Notice that if PROT is not specified, unprotected is assumed.

$RA (1,10,'*') generates the RA order sequence X'3C40C95C'.

$EUA (1,10) generates an EUA order sequence X'1240C9'.

$WCC (RESET,RESTORE,ALARM,PRINT,40CHAR,64CHAR,80CHAR,NLEM) generates the proper WCC, depending on the options selected in parentheses.

$CCC (PRINT,40CHAR,64CHAR,80CHAR,ALARM,ATT,UNPROT,PROT,ALL) generates the proper CCC, depending on the options selected in parentheses.

$IC generates X'13'.

After you have defined the macro instruction, the data stream required to build the sign-on panel shown in Figure 12 could be created as follows:

```
SIGNON  $MOD    MODEL=1
        $WCC    (RESET,RESTORE)
        $SBA    (2,11)
        $SF     (PROT)
        DC      C'SIGN-ON PROCEDURE'
        $SBA    (4,2)
        $SF     (PROT)
        DC      C'PLEASE ENTER YOUR SIGN-ON INFORMATION'
        $SBA    (6,1)
        $SF     (PROT,HI)
        DC      C'NAME:'
        $SF
        $IC
```

```
        $SBA    (6,25)
        $SF     (PROT,HI)
        DC      C'LOCATION:'
        $SF
        $SBA    (7,1)
        $SF     (PROT,HI)
        DC      C'SERIAL NUMBER:'
        $SF     (NUM)
        $SBA    (7,23)
        $SF     (PROT)
        $SBA    (10,4)
        $SF     (PROT)
        DC      C'WHEN ALL ... ENTER KEY'
```

You could also write a single 3270 data stream macro instruction, the format of which might be:

```
┌────────┬──────┬──────────┬──────────────────┬──────────────┬──────────────────┐
│        │      │          │ [,(attributes)]  │              │ [        {1}]    │
│[symbol]│ $MAC │ op-type  │ [,(row,column)]  │ [,character] │ [,MODEL={2}]     │
└────────┴──────┴──────────┴──────────────────┴──────────────┴──────────────────┘
```

symbol
: specifies a symbol that refers to the data stream

op-type
: specifies the type of screen control operation to generate. Valid values are: SF, SBA, IC, RA, EUA, WCC, and CCC.

(row,column)
: specifies the row (1 to 24) and column (1 to 80) where the operation starts or ends (depending on the op-type). This parameter is required for op-types SBA, RA, and EUA.

(attributes)
: indicates attributes or control bits for SF, WCC, and CCC:
  Some valid values for SF are: PROT, SKIP, NUM, MDT, HI, DET, NONDISP.

  Some valid values for WCC are: RESET, RESTORE, ALARM, PRINT, 40CHAR, 64CHAR, 80CHAR, NLEM.

  Some valid values for CCC are: PRINT, 40CHAR, 64CHAR, 80CHAR, ALARM, ATT, UNPROT, PROT, ALL.

character
: specifies the character used in the RA function.

MODEL=
: indicates the model of 3270. This model number is used to calculate the buffer address. This parameter is specified only once in the first macro instruction of a data stream series or whenever the data stream to be generated is for a different model than the preceding series.

After you have defined the macro instruction, the data stream required to create the sign-on panel shown in Figure 12 could be as follows:

```
SIGNON  $MAC    WCC,(RESET,RESTORE),MODEL=1
        $MAC    SBA,(2,11)
        $MAC    SF,(PROT)
        DC      C'SIGN-ON PROCEDURE'
        $MAC    SBA,(4,2)
        $MAC    SF,(PROT)
```

```
DC      C'PLEASE ENTER YOUR SIGN-ON INFORMATION'
$MAC    SBA,(6,1)
$MAC    SF,(PROT,HI)
DC      C'NAME'
$MAC    SF
$MAC    IC
$MAC    SBA,(6,25)
$MAC    SF,(PROT,HI)
DC      C'LOCATION:'
$MAC    SF
$MAC    SBA, (7,1)
$MAC    SF,(PROT,HI)
DC      C'SERIAL NUMBER:'
$MAC    SF,(NUM)
$MAC    SBA,(7,23)
$MAC    SF(PROT)
$MAC    SBA,(10,4)
$MAC    SF,(PROT)
DC      C'WHEN ALL ... ENTER KEY'
```

These two types of macro instructions can either generate a total static data stream or generate static sections of data streams which can be dynamically assembled at execution by the application program.

Semi-Dynamic Output Streams

A semi-dynamic panel requires some dynamic modification. Perhaps an error message must be written to a particular part of the panel and the cursor must be moved to the input field in which an error was detected during editing. The application program can concatenate preassembled static data stream segments into the program, such as field error messages. The same macro instructions that build static data streams can build partial static streams. As the input from a panel is edited, the standard error message for each field can be assembled in the output buffer, thus allowing multiple brief messages to be sent to the display in one operation.

You may have to change one or two attribute characters from high intensity to low intensity and erase the unprotected fields on a display. For example, an error message segment may have changed a field to high intensity to call the operator's attention to the field; the operator has recognized the error and re-entered the correct information. The display must now be made ready for the next input on the panel. Concatenate the order stream segments to change the attribute characters and use the ERASE unprotected to Address (EUA) order to restore the panel; do not transmit all the data and orders to completely refresh the panel.

Dynamic Output Streams

It may become physically impossible to hold in main storage all possible output data and order stream combinations that could occur during the execution of an application. You can incorporate a subroutine into screen management to accept parameters from an application program to decode the parameters and to create the data stream. You can also write a macro instruction for the application program which builds a parameter list inline from entries you specify in the macro instruction, and then branches to the screen management routine to build the required orders and data in the buffer area.

The macro instruction could appear as follows:

```
$BUILD,ADD=ADDFIELD ATT=(R3),DATA=(R4),LEN=(R5)
```

The ADDFIELD contains the 3270 buffer address in either row-column format, binary offset, or 3270 address form. R3 contains the address of the attribute character, R4 contains the address of the data to be entered in the field, and R5 contains the length of the data. The attribute character parameter is optional.

The subroutine could convert row and column buffer addresses relative to one to decimal offsets relative to zero with the following formula:

```
Model 1 Buffer:  ((R-1)X40)+(C-1)
Model 2 Buffer:  ((R-1)X80)+(C-1)
```

If the row and column buffer addresses relative to one are in two single-byte areas in binary, the conversion to binary offsets relative to zero can be coded as follows:

```
         SR      R3,R3
         IC      R3,COLUMN
         BCTR    R3,0
         SR      R4,R4
         IC      R4,ROW
         BCTR    R4,0
         MH      R4,=H'40' USE VALUE OF 80 FOR MODEL 2
         AR      R4,R3     RESULT IN R3
```

The following subroutine converts a binary halfword that represents the offset relative to zero of a position in a 3270 buffer, to an equivalent two-character 3270 address. R3 is a work register, and R4 points to the binary halfword to be converted. The converted result is found at ANSWER.

```
         LH      R3,0(R4)
         STC     R3,ANSWER+1
         SRL     R3,6
         STC     R3,ANSWER
         NI      ANSWER+1,X'3F'
         TR      ANSWER(2),TAB
         .
         .
         .
         .
ANSWER   DC      X'0000'
TAB      DC      X'40C1C2C3C4C5C6C7C8C94A4B'
         DC      X'4C4D4E4F50D1D2D3D4D5D6D7'
         DC      X'D8D95A5B5C5D5E5F6061E2E3'
         DC      X'E4E5E6E7E8E96A6B6C6D6E6F'
         DC      X'F0F1F2F3F4F5F6F7F8F97A'
         DC      X'7B7C7D7E7F'
```

Automatic Copy Function

Many applications require complete and unaltered hard copy (printout) of the terminal's current screen contents for the display station operator. The printer on which the display contents are printed may support one or more display stations depending on the 3270 configuration. The programming required to support the copy function depends on 1) whether the operator of a remote 3275 display station wants to obtain a hard copy on a slave printer attached to the 3275; 2) whether the operator of a 3277 attached to a remote 3271 wants to obtain a hard copy on a printer attached to the same 3271 control unit; or 3) whether the operator of a 3277 attached to a local 3272 wants to obtain a hard copy on a printer attached to the same 3272 control unit.

You should associate pressing a Program Attention key with a terminal operator request for hard copy on an assigned terminal printer. The

screen management program can be notified of the operator's request and perform the appropriate action.

When a data transfer to the computer occurs from pressing a Program Attention key, a remote 3277 or 3275 transmits STX, control unit address, device address, AID, and ETX; a local 3277 only transfers the AID (Attention Identifier) character. The AID character identifies which key transferred the data. No screen data is transmitted, thus the program is notified of a specific request.

Once the request is identified by inspecting the AID character, the program must identify what type of unit has made the copy request. This can be done by examining the characteristics of the specific device in a terminal characteristics table that you can create. Depending on the type of device, the following procedures can be used to accomplish the copy:

1. To copy from a remote 3275 to the slave printer attached to the 3275, the program should send the following data stream to the 3275: STX, ESC, WRITE command, WCC, ETX. The WCC (Write Control Character) specifies keyboard restore, start printer, and print 40 or 80 characters per line depending on the 3275 model number. Because the slave printer attached to the 3275 uses the same buffer as the display, all that is necessary to print the buffer (which contains the screen data), is the start print bit in a WCC sent in a valid WRITE command sequence.

2. To copy from a 3277 attached to a remote 3271 to a printer attached to the same 3271, the program should send the following data stream to the printer: STX, ESC, COPY command, CCC, from-device address, ETX. The CCC (copy control character) specifies start printer, the option to copy all data, and either 40 or 80 characters per line depending on the 3277 Model . A model 2 display can not be copied to a Model 1 printer, but all other copy combinations are valid. The device address following the CCC is a single-character address which identifies the device to be copied from, and which is identical to the device address used to specifically poll the display requesting the copy function.

   The COPY command allows the buffer contents of a device attached to a 3271 to be copied to the buffer of another device attached to the same 3271, without moving the data to be copied to and from the computer. Once the prior data stream has been sent to the printer, the program should send the following data stream to the display station that requested the copy: STX, ESC, WRITE command, WCC, ETX. The WCC specifies keyboard restore. The operator has a positive response that the request has been honored, and the keyboard is restored to allow the operator to continue without manual intervention.

3. To copy from a local 3277 to a local terminal printer, the program should execute a READ Buffer command to the display that made the copy request. The READ Buffer command is executed, and the display station transmits AID, a two-byte cursor address, and the screen data to the computer. The program should then remove the AID character and the cursor address from the received data, and immediately preceding the remaining data insert a WCC that specifies start printer, and 40 or 80 characters per line depending on the model of the printer. The altered data stream beginning with WCC should then be sent to the printer to copy the data. The program should then send a WCC with the restore keyboard option to the display that requested the copy function.

Printer Busy Considerations: If the program determines that the receiving printer is busy, and the requested copy function can not be immediately completed, one of the following actions should be taken:

3271: Notify the terminal operator of the situation and ask the operator to wait or cancel the request.

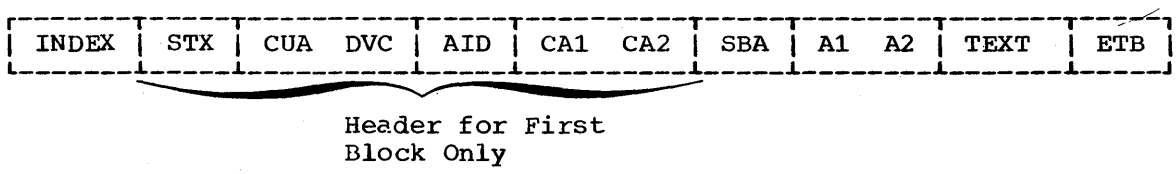
3271 or 3272: Perform a READ Buffer to bring the screen data into the computer where it can be queued until the printer is available, without delaying the operator.

Generally, a line control program polls for input from devices, passes data to terminals, transmits data from application programs to terminals, and translates data. An access method such as BTAM or TCAM communicates between the terminals and the computer. Additional factors to consider for a 3270 line control program are:

- Different devices on the lines have different characteristics.
- Application programs require information contained in the data stream.
- Screen management should receive the same data from a device management program, regardless of device type, to maintain a standard interface.

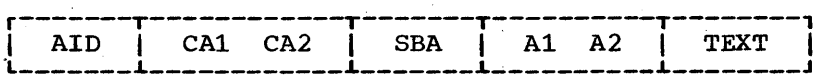
The following examples show data stream concatenation and standardization.

Example 1: The normal READ Modified message from a remote 3270 on a leased or switched line appears as follows:



The line control program can concatenate the blocks (which are generally 256 bytes or less) in a particular data stream, and strip the index, STX, ETB, ETX, and EOT characters. Control unit address and device address can be converted to a specific terminal name or ID with a table. The attention identification may be used to take a standard action (such as printing the buffer contents) defined for the terminal key that caused the interruption. A subroutine may be used to convert the cursor address into screen position (by number, such as 440, or row-column, such as 10, 15). The line control program can then pass the combined text (preceded with the SBA and address characters) to a screen management routine.

Example 2: A message from a local 3270 appears as follows:



The line control program should know where the data came from so the application program can send data to the source. You can check the relative line number in the DECB for the device address. The attention identification and cursor address information may be used as described in Example 1, and the text then sent to the screen management routine.

Example 3: A 3275 with the Dial feature sends a message that appears as follows:

As in Example 1, the line control program can concatenate the blocks, strip the STX and ETB or ETX characters, use the AID and cursor address

data to provide meaningful information, and supply a complete data stream to the screen management routine.

A line control program should include error recovery procedures to prevent unnecessary system or program failure. The program should be able to recognize, record, analyze, and correct error conditions and isolate a defective terminal, line, or control unit after a specific number of retries. Human intervention should be avoided by including error recovery procedures in the creation of the line control program.

TECHNIQUES FOR MANAGING DEVICES

THE ADVANTAGES OF A TERMINAL CONTROL PROGRAM

A terminal control program may be part of your application program or it may serve a number of applications. The terminal control program issues the BTAM (or other access method) macro instructions that initiate input and output. Usually it handles the error recovery you've specified. By separating this program from the processing application or applications, you allow future expansion of individual modules in your teleprocessing programming system without having to change all of them. A terminal control program can:

- Free the application program form the details of I/O, including error recovery. The terminal control program can be invoked by an instruction such as GET or PUT.
- Provide some buffering for the calling program. The terminal control program might collect all 256-byte blocks to be read from a 3270 terminal in its input area, then return with the address of the entire message.
- Simplify input for the processing program. For example, the AID byte in the data stream from entry of data might not matter to the processing program; the terminal control program can strip the AID byte or bypass it.
- Insert certain data stream characters. The terminal control program might contain some or all of the mapping functions suggested in the section "Screen Management" or only the I/O macro instructions and error recovery, and interface directly with a mapping module.

THE ADVANTAGES OF A MASTER TERMINAL PROGRAM

A master terminal program allows changes (in configuration, for example) in a teleprocessing application while the system is in operation. It provides a central control that allows the teleprocessing application or system to react flexibly to variables such as time of day, user or system priority, and system operator or remote supervisor messages. A master terminal program can usually be invoked (perhaps by the terminal control module) from a message by the console operator or from a local or remote 3277 designated as a master terminal.

The master terminal can communicate exclusively with the master terminal program or serve as a work terminal and be used as a master terminal when required. Access to the master terminal program may be available to any operator at a terminal designated as a master terminal, to a supervisor using his identification card at any of a number of terminals equipped with a card reader, to any terminal operator who entered a password authorizing use of the master terminal program, or to an operator at the system console.

Here are some uses for a master terminal program:

- A common use would be to change the configuration of a teleprocessing network; add or remove one or more terminals to a line. A supervisor at a master terminal in a Denver office could send a message to the central office in Kansas City to remove a temporarily inactive terminal from a line; the master terminal program would then (perhaps using the BTAM CHGNTRY macro instruction) set the skip bit for that device in the appropriate terminal list. Time would not be wasted polling that terminal.

- One or more application programs that depend on input and output from a system console or master terminal could use the master terminal program as a common interface to the master terminal or system console operator.

- On receipt of a master terminal message that the teleprocessing system will be switched from one operating system to another, the master terminal program could arrange an orderly collection of outstanding messages prior to system shutdown, then start up the teleprocessing system again after the new operating system is running.

- If it is desirable to switch disk files at a particular time for a given data entry application, a supervisor at a master terminal could request the switch and the master terminal program could send a request message to the system operator.

- A master terminal program could broadcast messages to all or designated terminals in a system. For example, operators could be notified of temporary system shutdown. A bank might use such a broadcast message to send branches or affiliates the serial numbers of stolen $100 bills.

- A master terminal program could maintain the time of day and assure that no terminals were polled in time zones that were not yet at work.

TECHNIQUES FOR KEEPING TRACK OF DEVICE STATUS

There are several reasons why you may want to maintain tables in your program with entries for each control unit or terminal. These tables can be used to store logical or symbolic names for use in messages (you may want to refer to a particular terminal in a message as MIAMI rather than by its device address number), to record the activity of each device, or to store other information such as dial digits used when calling a nonswitched device through a switched network backup facility (for example, using the IBM 3872 Modem).

The tables in Figure 47 may be used to:

- Associate each control unit or terminal with a name based on its geographic location or work station number.

- Keep track of the number of transactions (inquiries or entries, for example) from a terminal either for billing purposes or to see how much the terminal is being used.

- Keep track of various kinds of errors.

- Keep a priority assignment number when the network is heavily used.

- Keep the phone number to be dialed when using switched network backup to a control unit.

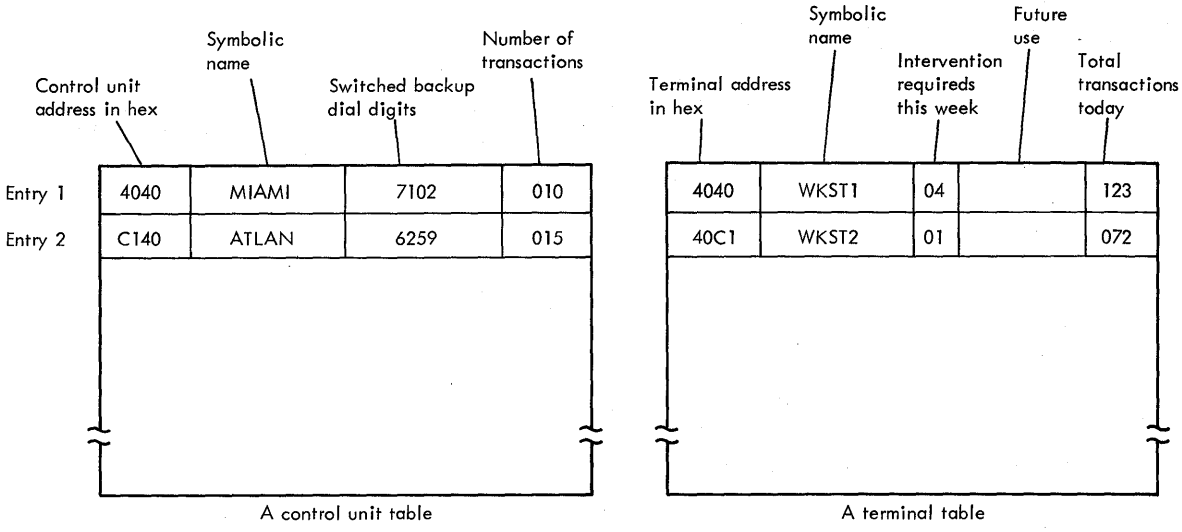


Figure 47. Table of control unit and terminal information

One way to associate a table with line activity is to build your own extension to the DECB. Figure 48 suggests some of the uses for such a DECB extension.

TRANPTR is an area that stores a pointer to a translation table used after an input operation. Another byte, PREVOP, stores information on what I/O operation took place prior to the one the DECB is presently associated with; this could be useful in determining error recovery actions. Another use would be to store the return address associated with an I/O operation when one READ or WRITE macro instruction and DECB is branched to from more than one place in a program. Here is one DOS coding technique for saving a return address with a DECB extension:

```
        BAL   R11,WRTTT
         .
         .
         .
WRTTT   WRITE (R6),TT,DTFBT1,AREA1,200,,3,MF=E
         B    TWAIT
         .
         .
         .
TWAIT   EQU    *
        ST     R11,RETURN
        TWAIT (R6),TERMTST,ECBLIST=LIST
        L      R11,RETURN
        BR     R11
```

RETURN, shown in Figure 48, is a three-byte area in the DECB extension.

RELIABILITY AND ERROR RECOVERY

Remote Leased Line Event Completion Analysis

On completion of a 3270 I/O operation, the terminal control program should analyze the circumstances of the completion and decide what action to take. This section applies to any terminal control program that uses BTAM.

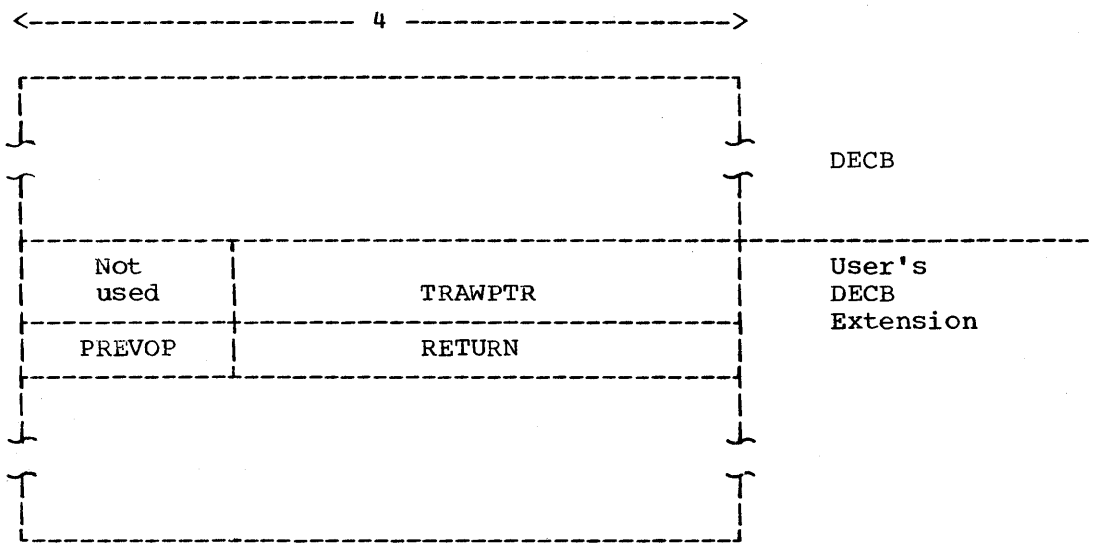


Figure 48. Example of a user-built DECB extension

The 3270 remote leased line completion analysis is organized in six parts. Four of the parts are the flowcharts in Figures 49 through 52, which are a logical sequence for analyzing completion information after a read or write operation. The flowcharts refer to the READ action descriptions or WRITE action descriptions which follow the READ or WRITE flowcharts.

The action descriptions are in the following format:

- The BTAM operations to which the action applies.
- An explanation of the causes of the completion condition.
- The advised actions and an explanation, where appropriate.

Certain completion conditions indicate that a 3271 or 3275 sense/status message has been received. These messages are generated by the remote 3270 in a variety of circumstances to inform the computer of changes in the status of 3270 devices. Examples of such changes are the completion of a mechanical print operation or the receipt by the 3271 or 3275 of an invalid command. For further information on the sense/status message, refer to the description of remote operations in the IBM 3270 Information Display System, Component Description. Where completion conditions exist, the action description contains the advised procedure for processing the receipt of the message as input. However, the description of sense/status analysis should be consulted to interpret the information in the message and the actions that follow. The 3270 sense/status message must be processed to maintain the availability of the remote 3270 devices.

Read Action Description One

BTAM Operations: Follows completion of a Read Initial (TI) or Read Continue (TT).

Explanation: A text block has been received without hardware or line error. The input message may take one of several formats. The format generally appears as follows:

AUTOPOLL INDEX BYTE	STX	CU ADDR	DVC ADDR	AID	CURSOR ADDRESS	FIELD DATA	ETB / ETX

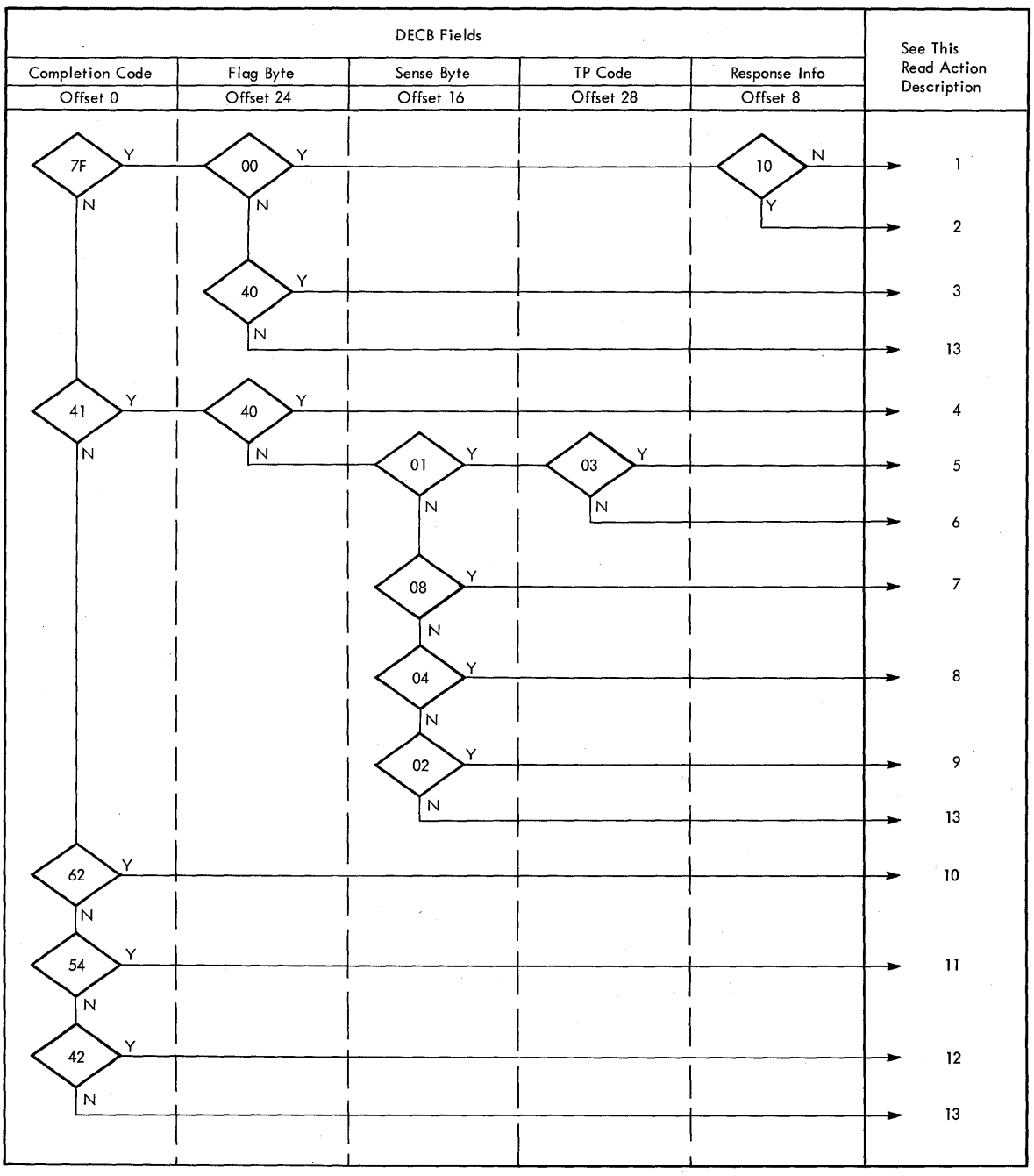


Figure 49. DOS BTAM remote leased line READ completion analysis

If an operator has initiated the message other than with the TEST REQUEST key, the first block from a device has the above format. The maximum block length is 256 characters from Auto Poll index byte through ETX or ETB. The block could be less than 256 even if there are subsequent blocks, because the 3270 does not break an SBA sequence. The following variations to the above format are also possible:

- If Auto Poll is not used or if the message is not from the first device to respond to a general poll, STX is the first character in the input area.

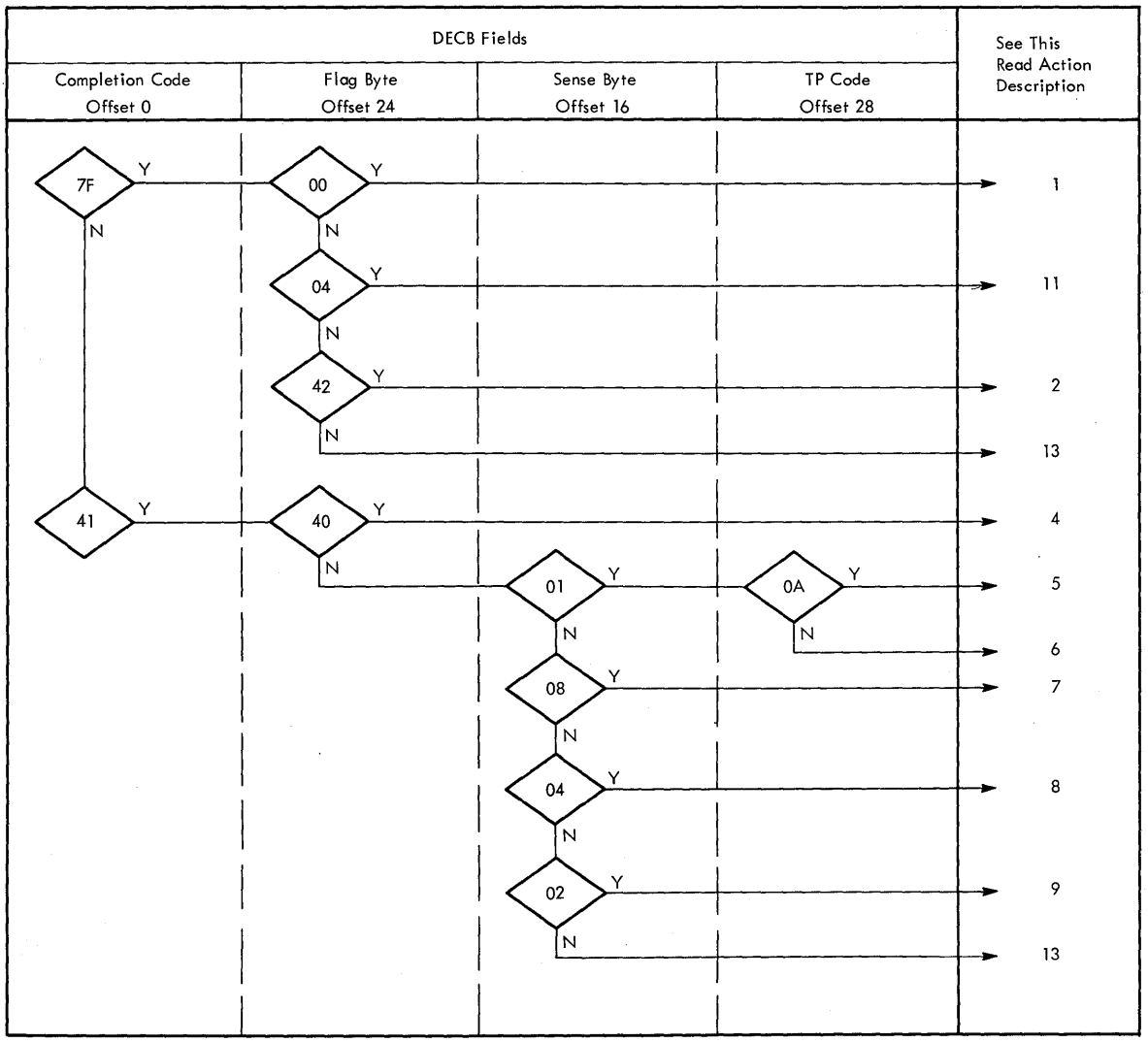


Figure 50. OS BTAM remote leased line READ completion analysis

- If the CLEAR key or a Program Attention key was pressed, the ETX is the only character following the AID byte.

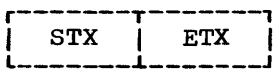
- If there are no modified fields, the ETX is the only character following the cursor address.

- If the input block is not the last from the device, an ETB terminates the block. ETX terminates the last or only block.

STX	FIELD DATA	{ETB / ETX}

If the input block is not the first from the device, it has the above format. STX is the first character in the input area and there is no 3270 header information.

The above format could be received if the operator pressed the TEST REQUEST key and the Binary Synchronous test facility is not included in the BTAM support.



The above null message format may be received as the last block under unusual circumstances.

BTAM passes the EOT character to the user in the input area as a normal completion. Bits are set in the DECB to indicate that an EOT has been received.

Action:

- Issue READ Continue (TT) until EOT is received if multiple messages from a control unit on general poll are acceptable.
- Issue READ Interrupt (TRV) if multiple screens from any additional devices pending on the cluster are not desired.

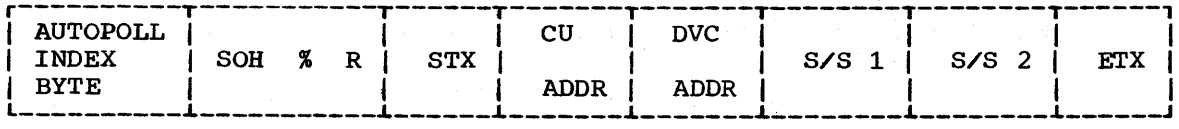
General polling is being performed and all blocks associated with the first message have been read. All blocks of a screen should be read and concatenated before processing. To do this, move the STX plus one location of succeeding blocks to the ETB location of preceding blocks. Receipt of a null block can be processed in the same way. If a TEST REQUEST message is received in this manner, the Binary Synchronous test facility should be included. If this is not possible, the message may be processed as a CLEAR key depression.

If OS BTAM is used and an EOT is received, the response depends on the application. See Read Action Description Three.

Read Action Description Two

BTAM Operation: Follows completion of a Read Initial (Ti) or Read Continue (TT).

Explanation: A text block containing a sense/status message has been received from the remote control unit.



The Auto Poll index is not present if the Auto Poll feature is not used or if the message is not the first received in response to a general poll.

Action: Same as Read Action Description One.

In order to maintain the availability of the remote 3270 devices, the sense/status message must be analyzed and acted upon. For guidance in processing this message, see the section "Sense/Status Analysis".

Read Action Description Three

BTAM Operation: Follows completion of a Read Continue (TT).

Explanation: An EOT was received in response to the previous Read Continue (TT) for a text block ending in ETX.

Action: The action taken depends on the line control program.

- A held line system holds the communications line open after receipt of a message in anticipation of a response to the device. For this type of system the line is left inactive until a response is created. At this time a Write Initial (TI) is issued to select the device and send the response.

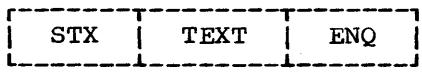
- A non-held line system does not hold the line open after receipt of a message. In this type of system the line control program might check and issue a WRITE Initial if output is available. If there is no output for the line, polling might be initiated with the READ Initial (TI) macro instruction.

In most systems where message throughput is a primary objective, non-held lines are preferred.

Read Action Description Four

BTAM Operation: Follows completion of a Read Initial (TI) or Read Continue (TT).

Explanation: A text block has been received that terminates with an ENQ character.



This message format indicates that the 3270 control unit has detected an internal parity check or a cursor check during transmission. A character with invalid parity is transmitted as a SUB character (EBCDIC '3F' or ASCII '1A') and the ending ETX or ETB is replaced with the ENQ character. In addition, a data check sense/status condition is recorded at the 3270 control unit.

Action: Issue a READ Initial (TI) using the specific polling characters for the sending device to retrieve the sense/status message and reset the status condition at the control unit. Use Action Description Two after receipt of the sense/status message.

Read Action Description Five

BTAM Operation: Follows completion of a Read Initial (TI).

Explanation: A timeout has occurred. No response to the previous polling sequence has been received in the time allowed by the transmission control unit. Possible causes are:

- The 3270 control unit is unable to respond, perhaps due to lack of power, a malfunction, or the keylock has not been unlocked.

- Conditions on the communications line prevent transmission.
- A modem is not functioning.
- The transmitted polling sequence is not valid for any control unit on the communications line.

Action:

- To retry the polling operation, issue a Read Initial (TI). BTAM will have retried the operation; reissuing the macro instruction begins a new sequence. After retrying the polling operation, if the condition persists:
  - Take the control unit or terminal out of service and off the polling list. This can be done either under program control or in response to operator intervention through a master terminal (see "Advantages of a Master Terminal Program" in this section). Whether the action is automatic or in response to a command entered by an operator, you should issue the BTAM CHGNTRY macro instruction.

    The following is an example of using the CHGNTRY macro instruction to remove a control unit from a polling list:

```
CHGNTRY   (R2),AUTOWLST,(R3),5,SKIP
```

    (R2) is a register with the address of the polling list, (R3) is a register with the relative position of the entry to be changed, and 5 is the number of characters in a 3270 polling list entry. The example specifies the list as an Auto Poll wrap list but should agree with the type specified in the DFTRMLST macro instruction used to create the list. A CHGNTRY macro instruction with the ACTIVATE parameter can reinstate the control unit when the difficulty has been corrected.

    The fact that the control unit is not available should be recorded for use of the terminal control program, and the operator should be notified to take manual recovery action if the system has not previously informed him.

    Read Initial (TI) should be reissued after forcing the DECB polling entry address (OFFSET 21) to another control unit. If there is no other control unit on the line, or all on the line are out of service, the line should be recorded as out of service and no further operations initiated until it is placed back in service, perhaps by a master terminal.

Read Action Description Six

BTAM Operation: Follows completion of a Read Initial (TI) or Read Continue (TT).

Explanation: A time-out has occurred. No further transmission has been received after a text block acknowledgement (ACK0 or ACK1), or text flow has stopped without a proper ending sequence (ETC, ETX, ENQ). The possible causes include those in Read Action Description Five, except an invalid polling sequence does not apply.

Action:

- Issue a Read Repeat (TP) to acknowledge no transmission received and to receive the response, if you want more retries than BTAM error recovery provides.

- If the problem is not corrected issue a WRITE Reset (TR) to reset the line with an EOT. Remove the control unit from the polling list as in Read Action Description Five.

Read Action Description Seven

BTAM Operation: Follows completion of a Read Initial (TI) or Read Continue (TT).

Explanation: The transmission control unit has detected an erroneous parity or BCC check on the received data.

Action: See Read Action Description Six.

Read Action Description Eight

BTAM Operation: Follows completion of a Read Initial (TI) or Read Continue (TT).

Explanation: An overrun condition has occurred. The I/O channel has not maintained the speed of the incoming data.

Action: See Read Action Description Six.

Read Action Description Nine

BTAM Operation: Follows completion of Read Initial (TI) or Read Continue (TT).

Explanation: A lost data condition has occurred. This is usually due to receipt of a data stream that exceeds the length specified for the Read operation.

Action: See Read Action Description Six.

Read Action Description Ten

BTAM Operation: Follows a Read Continue (TT).

Explanation: The positive acknowledgement of the preceding text block was not properly received by the remote control unit, which responded with an ENQ character.

Action: Issue a READ Continue (TT) to retry the acknowledgement. If the condition persists, take Read Action Description Five.

Read Action Description Eleven

BTAM Operation: Follows a Read Initial (TI).

Explanation: A negative response was received from the last active terminal in an open polling list (DFTRMLST AUTOLST), or a RESETPL macro instruction terminated polling.

Action: The appropriate action depends on the line control program:

- If output is available for the line, issue a WRITE Initial (TI) macro instruction to send the message.
- Resume polling at the beginning of the list.
- Suspend polling long enough to reduce the impact of processing negative polling responses.

Read Action Description Twelve

BTAM Operation: Follows a Read Initial (TI) or Read Continue (TT).

Explanation: A TEST REQUEST message has been received but the TWAIT macro instruction has not been issued.

Action: Issue a TWAIT macro instruction of the form:

```
TWAIT   (R1),TERMTST,ECBLIST=(R2)
```

where (R1) specifies a register which will contain the address of the DECB posted complete when the TWAIT is satisfied, and (R2) is loaded with the address of the DECB with the X'42' completion.

Do not alter the completion code prior to issuing the TWAIT macro instruction.

Read Action Description Thirteen

BTAM Operation: Follows Read Initial (TI) or Read Continue (TT).

Explanation: This is an unrecognized completion and should not occur; it is probably a software problem.

Action:

- Take a SNAP dump or PDUMP of the system and analyze it.
- Notify the operator of the condition.
- Issue a READ Initial (TI) to reset the line and resume polling.

Write Action Description One

BTAM Operation: Follows Write Initial (TI), Write Continue (TT), Write Conversational (TV), or Write Initial Conversational (TIV).

Explanation: Text transfer has completed normally.

Action:

- If the previous operation was Write Initial (TI), issue WRITE Continue (TT) if blocked output is being sent and more blocks remain. Note that the 3270 does not accept conventionally blocked output.

  Issue WRITE Reset (TR) to send an EOT which resets the line. If it is desirable to resume polling on the line, issue READ Initial (TI), which resets the line and begins polling.

- If the previous operation was Write Initial Conversational (TIV), Write Continue Conversational (TTV) or Write Conversational (TV), issue READ Continue (TT) to read all blocks and the final EOT.

Write Action Description Two

BTAM Operation: Follows a Write Initial Conversational (TIV), Write Continue Conversational (TTV), Write Conversational (TV), Write Initial (TI), or Write Continue (TT).

Explanation: An EOT has been received in response to a text transmission. This response indicates that the device could not perform the operation specified by the command code in the text. Examples are a busy or unavailable device, or a check condition. The exact cause has been recorded at the control unit.

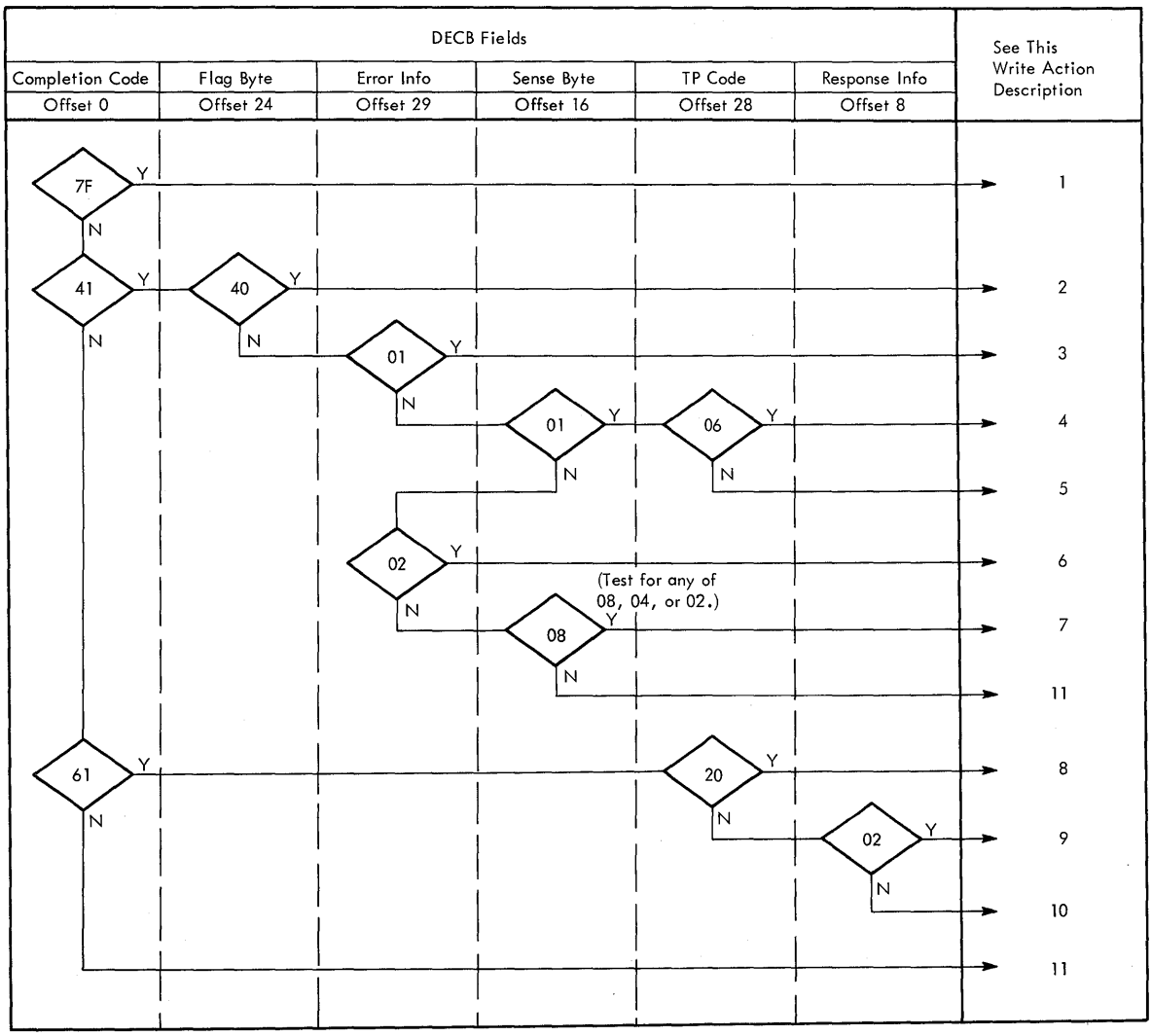


Figure 51. DOS BTAM remote leased line WRITE completion analysis

Action: Issue a READ Initial (TI) command specifying as the polling entry a list containing the specific polling characters of the control unit and device that returned the EOT. You could code a specific polling list with the DFTRMLST macro instruction for each terminal in the system or code one DFTRMLST macro instruction and modify the entry prior to issuing the READ Initial (TI) macro instruction. See the description of the DFTRMLST format in the appropriate BTAM SRL.

Write Action Description Three

BTAM Operation: Follows Write Initial (TI), Write Continue (TT), Write Initial Conversational (TIV), Write Continue Conversational (TTV), or Write Conversational (TV).

Explanation: A NAK has been received in response to a text transmission. This indicates that the 3270 control unit has detected an ENQ character in the transmission or that the 3271 (not 3275) has detected an invalid BCC. A 3275 sends an EOT and indicates an invalid BCC in the sense/status message. The cause could be a transmission error or invalid text. BTAM will have retried the operation.

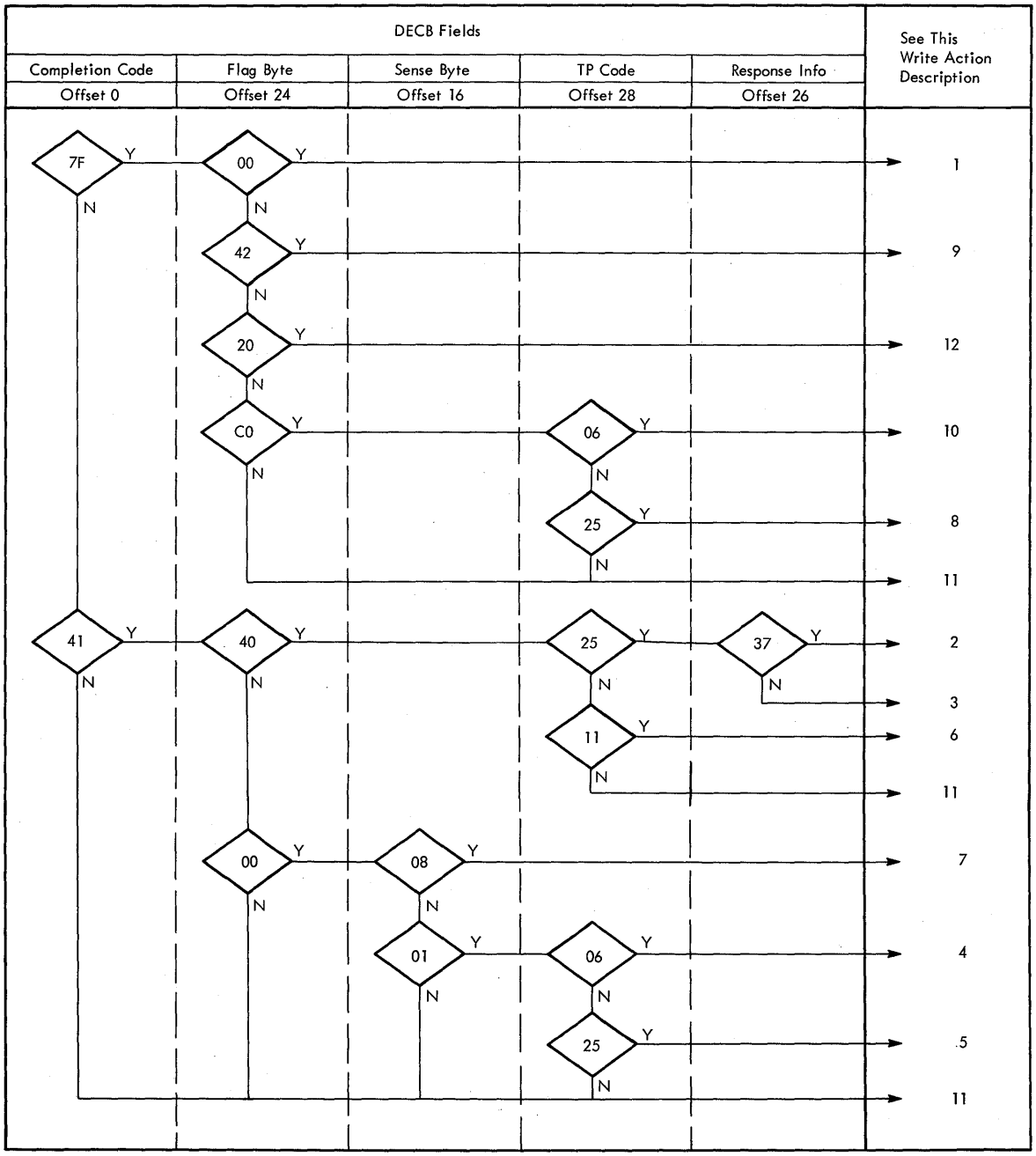


Figure 52. OS BTAM remote leased line WRITE completion analysis

Action:

- If the preceding operation was Write Initial (TI) or Write Continue (TT), issue a WRITE Initial (TI) to retry the operation.
- If the preceding operation was Write Initial Conversational (TIV) or Write Conversational (TV), issue a WRITE Initial Conversational (TIV) to retry the operation.
- If the condition permits, take a SNAP dump or PDUMP of the text block and notify the operator of the condition so that the fault can be isolated.

Write Action Description Four

BTAM Operation: Follows a Write Initial (TI) or Write Initial Conversational (TIV).

Explanation: No response to the previous selection sequence has been received. Possible causes include:

- There is no device on this line for the selection sequence that was sent.
- A hardware transmission error prevented recognition of the selection sequence or the acknowledgement.
- The 3270 is unavailable due to lack of power or a malfunction.

BTAM will have retried the operation.

Action:

- If the previous operation was Write Initial (TI), issue a WRITE Initial (TI) to retry the operation.
- If the previous operation was Write Initial Conversational (TIV), issue a WRITE Initial Conversational (TIV) to retry the operation. If the condition persists, take the device out of service. Record the fact that the terminal is out of service for use by the terminal control program and notify the operator. See "Advantages of a Master Terminal Program" in this section.

Write Action Description Five

BTAM Operation: Follows Write Initial (TI), Write Continue (TT), Write Initial Conversational (TIV), Write Continue Conversational (TTV), or Write Conversational (TV).

Explanation: No response to the preceding text transmission has been received. Possible causes include:

- The preceding text was received by the 3270 without valid framing characters (STX/ETX).
- The 3270 has been unavailable.
- A transmission error has prevented receipt of the response.

Action: Reissue the preceding macro instruction to retry the operation. If the condition permits, take the terminal out of service and proceed as in Write Action Description Four.

Write Action Description Six

BTAM Operation: Follows Write Initial Conversational (TIV), Write Continue Conversational (TTV), or Write Conversational (TV).

Explanation: Text ending in ENQ has been received. This indicates that the 3270 has detected an internal parity check or a cursor check. A SUB character (EBCDIC '3F' or ASCII '1A') has been substituted for the error character and the ENQ character is transmitted in place of the ETX/ETB and BCC. A status condition has been stored at the 3270's control unit.

Action: Issue a READ Initial (TI) with a polling list containing the specific polling sequence for the device transmitting the ENQ character. This retrieves the 3270 sense/status message (see "Sense/Status Analysis" in this section) and resets the status condition.

Write Action Description Seven

BTAM Operation: Follows Write Initial Conversational (TIV).

Explanation: Text was received in error:

X'08' - data check
X'04' - overrun
X'02' - lost data

See Read Action Descriptions Seven through Nine.

Action: Issue a READ Repeat (TP) to send a NAK, which transmits the lost text again.

Write Action Description Eight

BTAM Operation: Follows Write Initial (TI) or Write Continue (TT).

Explanation: A WACK was received in response to a text write. This is a normal response to a text data stream that contains a copy control character or write control character specifying 'start printer.' It implies that the print operation has begun and that the printer is now busy.

Action: Issue a WRITE Initial (TI) to send to another device on the line, or issue a READ Initial (TI) to begin polling on the line.

Write Action Description Nine

BTAM Operation: Follows Write Initial (TI) or Write Initial Conversational (TIV).

Explanation: An RVI has been received in response to addressing. This response indicates that the 3270 has pending status, other than Device end or device busy, which must be retrieved prior to writing to the 3270.

Action:

- Issue a WRITE Reset (TR) to reset the line.
- Then issue a READ Initial (TI) using a polling list with the specific polling sequence for the device.

Write Action Description Ten

BTAM Operation: Follows a Write Initial (TI).

Explanation: A WACK has been received in response to addressing. This indicates that the addressed device is busy.

Action:

- Issue a WRITE Reset (TR) to terminate the operation.
- Then check for output to another device on the line and issue a WRITE Initial (TI) or initiate polling on the line with a READ Initial (TI).

Write Action Description Eleven

BTAM Operation: Follows Write Initial (TI), Write Continue (TT), Write Initial Converstional (TIV), Write Continue Conversational (TTV), or Write Conversational (TV).

Explanation: This is an unrecognized completion and should not occur. The probable cause is a software problem.

Action:

- Take a SNAP dump of the system for analysis.
- Notify the operator of the condition.
- Issue a WRITE Reset (TR) to reset the line, then issue a READ Initial (TI) to resume polling.

Write Action Description Twelve

BTAM Operation: Follows a Write Initial (TI).

Explanation: An incorrect alternating acknowledgement was received in response to the text transmission. BTAM has validated the incorrect acknowledgement is incorrect by sending an ENQ to request retransmission of the ACK.

Action:

- Issue a WRITE Reset (TR) to reset the line. Then retry the WRITE Initial (TI).
- If the problem persists, notify the operator of the condition. Record the control unit out of service and proceed as in Read Action Description Five.

Remote Dial Event Completion Analysis

This section should help you design or code the portion of a terminal control program which, upon completion of a 3270 I/O operation, analyzes the completion and decides the proper action.

The description of the 3270 remote Dial event completion analysis is organized in six parts. Four of the parts are flowcharts contained in Figures 53 through 56. These flowcharts are a logical sequence in which completion information can be analyzed after a Read or Write operation. The flowcharts refer to the action descriptions that immediately follow the flowcharts.

The action descriptions are in the following formats:

- An explanation of the causes of the completion condition.
- The advised actions, and comments, where appropriate.

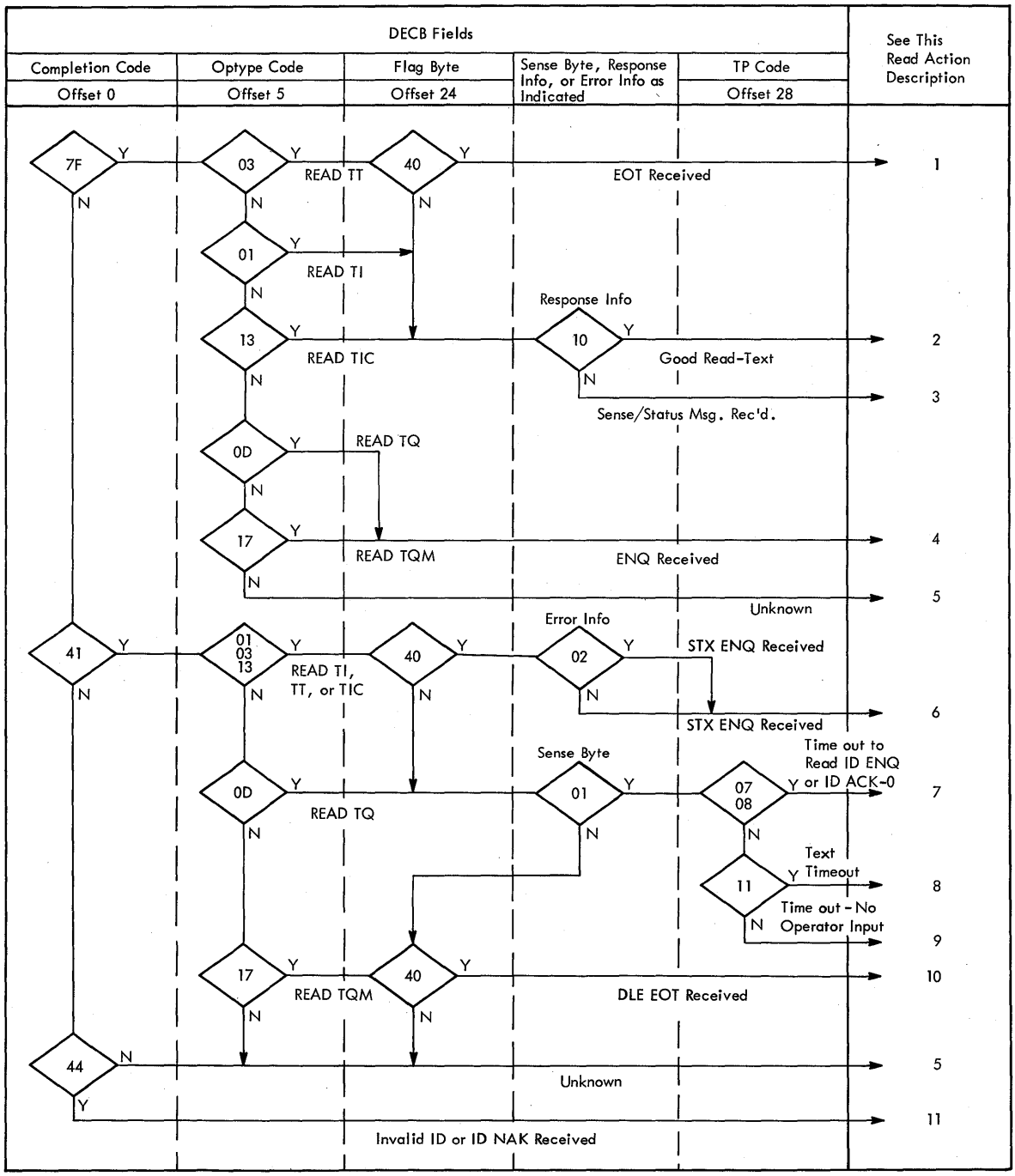


Figure 53. DOS BTAM remote Dial READ completion analysis

Read Action Description One

Explanation: The 3275 has responded EOT to a READ continue, indicating it has no more text to send and is releasing control of the line.

Action: Normal completion of a read sequence from the 3275 allows you to:

1. Issue a WRITE Disconnect if no more input or output is desired from or to the 3275.

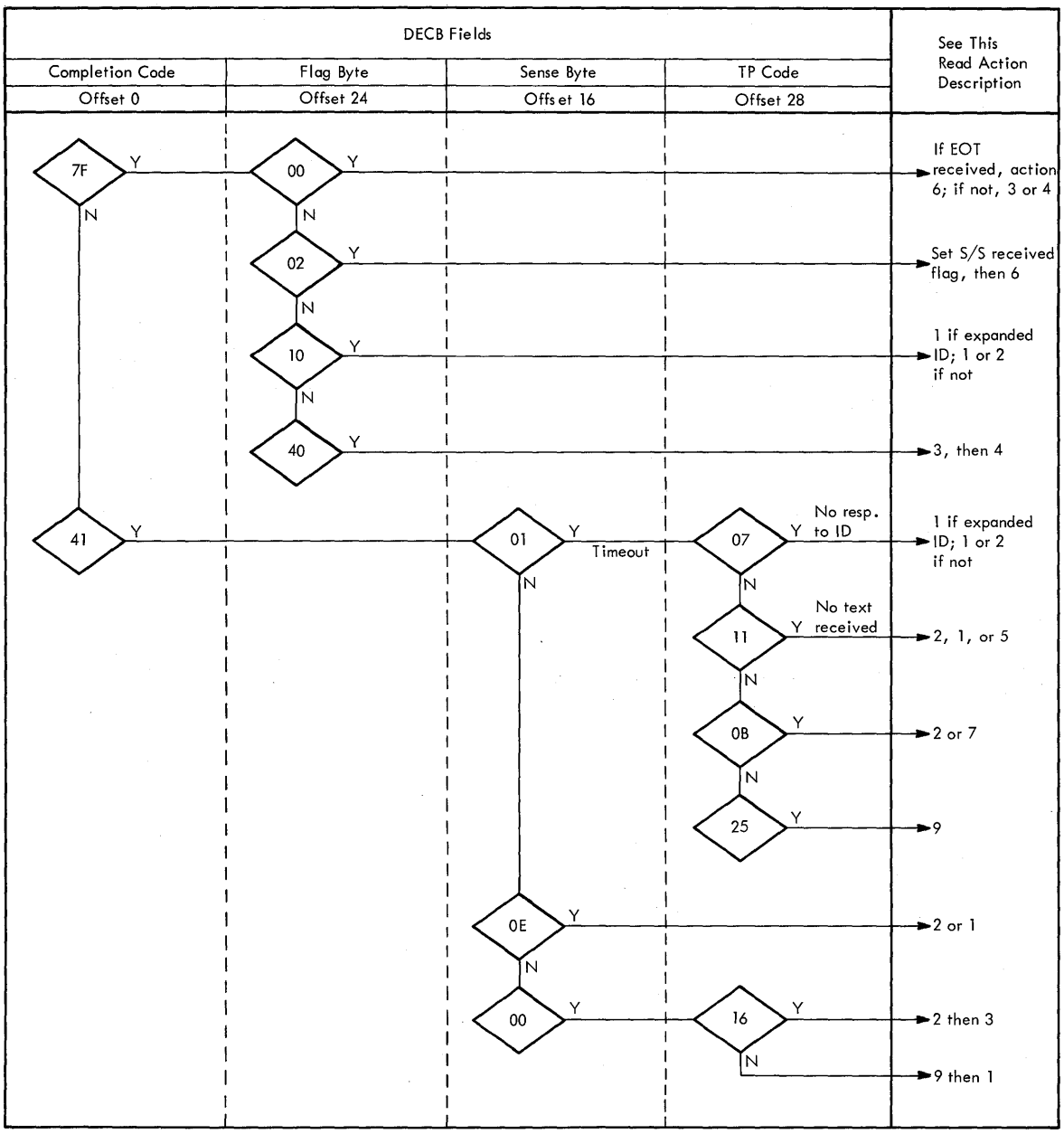


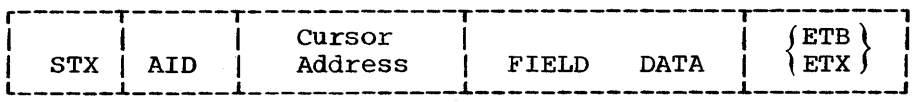
Figure 54. OS BTAM remote Dial READ completion analysis

2. Issue a READ Inquiry or READ Inquiry Monitor to await another bid from the 3275. The READ Inquiry Monitor does not cause a 3275 time-out. The 3275 does not send more data quickly, but it does allow the phone connection to be maintained even if the 3275 operator does not disconnect. You might set a timer interval with the Monitor version and end the read with RESETPL after the interval, followed by a WRITE Disconnect, or a WRITE Inquiry followed by WRITE Continue to prompt the operator.

3. Issue a WRITE Inquiry to bid for the line, if you want to send a message to the terminal.

Read Action Description Two

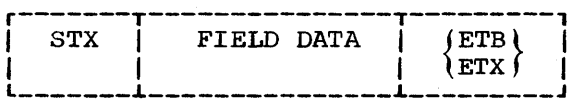
Explanation: A text block has been successfully received from the 3275 from a READ Initial, READ Continue, or READ Connect macro instruction. The input may have the following format:



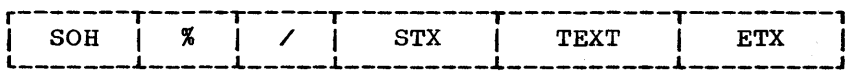
If the operator initiated the message other than with the TEST REQUEST key, the first block has the above general format. The maximum block length is 256 characters from STX to ETX or ETB. The block could be fewer than 256 characters even if there are subsequent blocks because the 3275 does not break an SBA sequence (3 bytes represent a buffer address).

The following variations in the above format are possible:

- If the CLEAR key or a Program Attention key was pressed, the ETX is the only character following the AID byte.
- If there are no modified fields, the ETX is the only character following the cursor address.
- If the block is not the last from the device, an ETB terminates the block. ETX or ETB terminates the last block.



If the input block is not the first block of the transmission series, it has the above format (no AID or cursor address). ETX or ETB terminates the last block.



The above format could be received if the operator pressed the TEST REQUEST key and the Binary Synchronous Test Facility is not included in BTAM.

The above format could be received as the last block under certain unusual circumstances, and can be ignored.

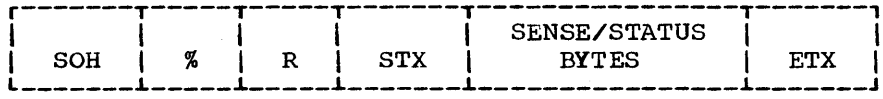
Action: Issue READ Continue macro instructions until EOT is received from the 3275, ending its control of the line. The data blocks can then be concatenated and passed to the application modules, as discussed in "Remote Leased Line Event Completion Analysis" in this section.

Read Action Description Three

Explanation: A 3275 sense/status message has been received (see "Sense/Status Analysis" in this section), indicating either a "busy" condition, a return to "ready" from "not-ready" status, or a hardware error condition at the terminal.

Action: Issue a READ Continue macro instruction to receive EOT from the 3275 and allow it to relinquish control of the line. Analyze the two sense/status bytes as outlined in "Sense/Status Analysis" in this section.

The message has the format:



Read Action Description Four

Explanation: The 3275 has responded ENQ to a READ Inquiry or READ Inquiry Monitor macro instruction, indicating the operator wishes to send a message.

Action: Issue a READ Continue to receive the first text block.

Read Action Description Five

Explanation: A completion condition unknown to the 3275 Dial support has occurred. This is probably caused by a programming error such as issuing an invalid macro instruction sequence.

Action: See Write Action Description Ten.

Read Action Description Six

Explanation: The 3275 has detected an internal buffer parity error (STX-ENQ received) or a malfunction other than a parity error (STX-EOT received), and has aborted its current transmission. This indicates that the 3275 has an error sense/status message pending, which you must retrieve and analyze before continuing.

Action: To reset the line, issue a WRITE EOT for DOS or a WRITE RESET for OS. Then issue a READ Continue macro instruction to read the sense/status message from the 3275. See Read Action Description Three for the format of a sense/status message.

Read Action Description Seven

Explanation: While establishing a connection with the READ Initial or READ Connect macro instruction, the 3275 failed to transmit its ID-ENQ sequence (CPU answering operation) or ID-ACK0 sequence (CPU called the terminal) within the TCU's timeout interval. This is probably a terminal hardware error.

Action: Issue a WRITE Disconnect to disconnect and disable the line, followed by another READ Initial or READ Connect, as appropriate.

Read Action Description Eight

Explanation: The 3275 has failed to transmit text within the TCU's time-out period in response to a READ Initial, READ Connect, or READ Continue macro instruction. This can occur due to incorrect operator procedures on READ Initial or READ Connect. You should retry the operation. On READ Continue, it is probably a hardware error and is not worth retrying.

Action: Issue a WRITE Disconnect to disable and disconnect the line, followed by another READ Initial or READ Connect, as appropriate. Alternately, for READ Initial or READ Connect, you might issue a READ

Inquiry or READ Inquiry Monitor to await input, or issue a WRITE Inquiry to bid for the line and send a prompting message.

Read Action Description Nine

Explanation: The 3275 operator has failed to bid for the line within the TCU's time-out interval, following the Read Inquiry. This is a normal occurrence, depending on your usage of Read Inquiry and WRITE Reset instead of Read Inquiry Monitor and Write Reset Monitor, which cannot time out.

Action: You may issue WRITE EOT, WRITE Reset, or READ Inquiry to initiate another time-out interval, or use the Monitor form of either macro instruction to wait indefinitely for operator action. The Monitor form of the macro instructions may cause extended toll costs if the operator is not at the display station.

Read Action Description Ten

Explanation: The 3275 operator has requested immediate termination of the connection by pressing the DISCONNECT key on the 3275.

Action: Issue a WRITE Disconnect to disconnect and disable the line, followed by another Initial or connect-type macro instruction to establish another connection with the same or another operator.

Read Action Description Eleven

Explanation: The 3275 has responded with an invalid ID sequence or with ID-NAK to your Read Initial or Read Connect, indicating either a wrong terminal or a hardware error. In either case, further communication is not desired.

Action: Issue a WRITE Disconnect to disconnect and disable the line, followed by an Initial operation to establish communication with the same or a different terminal.

Write Action Description One

Explanation: The 3275 has responded positively to the last text transmission or bid for the line. The exact response meaning is as follows:

WRITE Initial - text received properly
WRITE Continue - text received properly
WRITE Connect - proper terminal ID has been sent by the 3275; terminal is ready for data.
WRITE Inquiry - positive response to the bid for the line; 3275 is ready to receive data.

Action: Issue a WRITE Continue to transmit text, or a WRITE EOT, WRITE EOT Monitor, or a WRITE Reset/WRITE Reset Monitor to terminate this transmission.

Write Action Description Two

Explanation: The 3275 has responded ENQ to a WRITE EOT or WRITE EOT Monitor, indicating that it desires to transmit a message.

Action: Issue a READ Continue macro instruction to the message.

Write Action Description Three

Explanation: The 3275 has a sense/status error message pending, which you must retrieve and analyze before continuing.

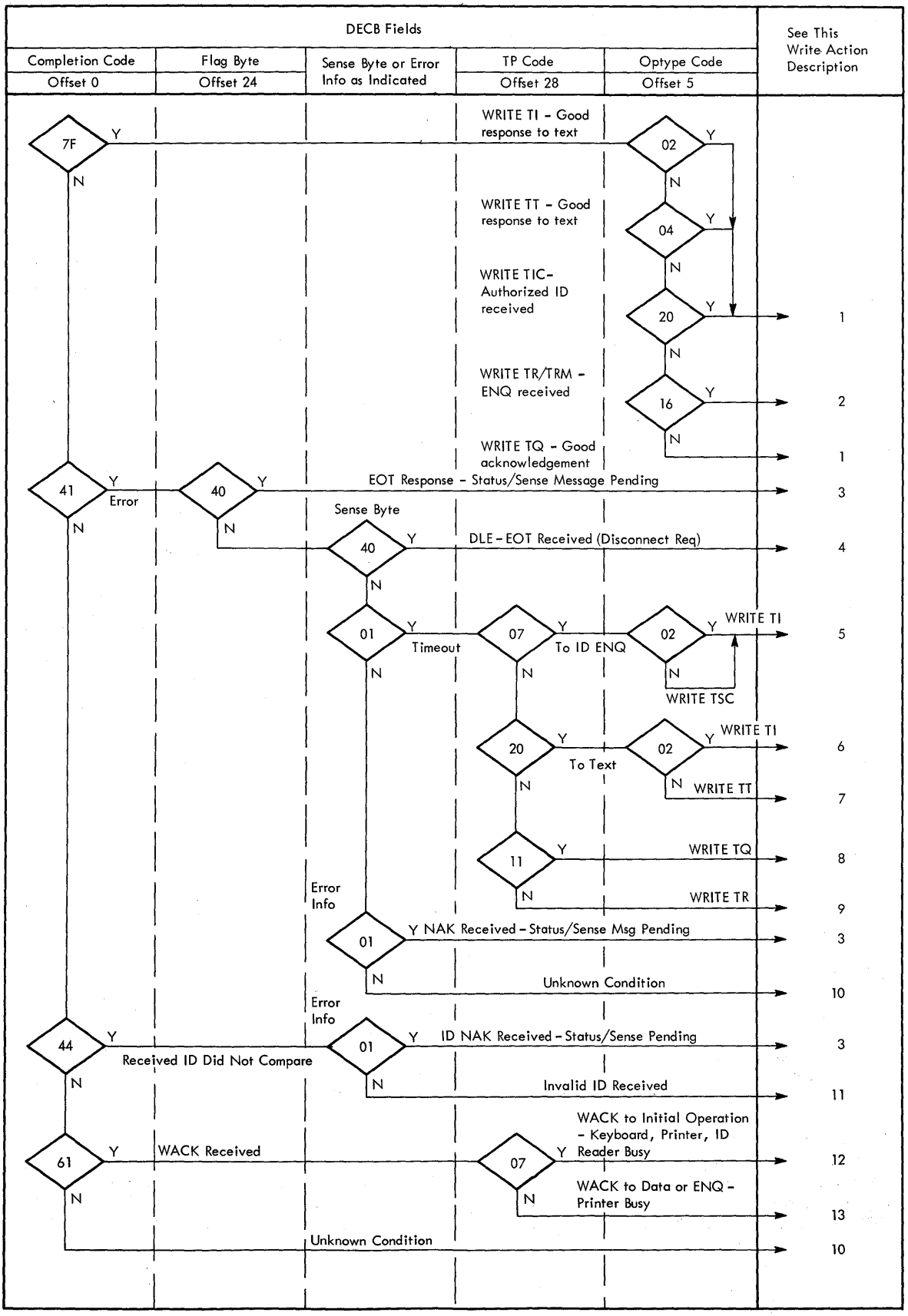


Figure 55. DOS BTAM remote Dial WRITE completion analysis

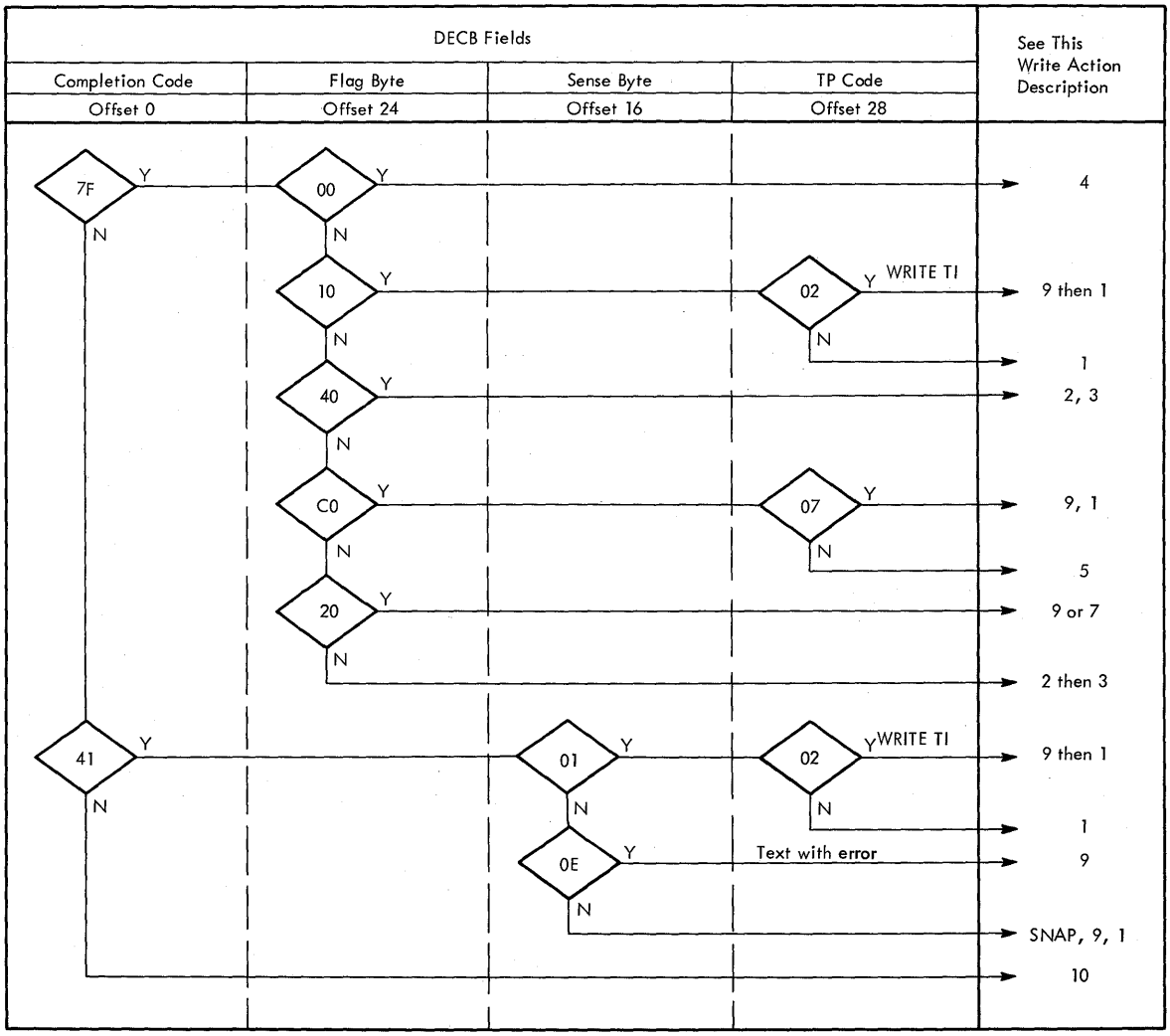


Figure 56. OS BTAM remote Dial WRITE completion analysis

The notification is by one of the following:

1. EOT response to text - WRITE Initial or WRITE Continue
2. NAK response to ENQ - WRITE Inquiry
3. ID-NAK response to Initial Connection - WRITE Initial or WRITE Connect.

Action: Issue a WRITE EOT or WRITE Reset macro instruction to reset the line, then issue a READ Continue macro instruction to read the sense/status message from the 3275. You should issue another READ Continue after receipt of the sense/status message to allow the 3275 to transmit EOT and clear its status condition. The message has the form:

SOH	%	R	STX	SS1	SS2	ETX

You should now analyze the two SS bytes, as described in the section "Sense/Status Analysis" and take the appropriate corrective action.

Write Action Description Four

Explanation: The remote operator has requested immediate disconnection through use of the DISCONNECT switch on the display, and has probably already hung up the phone.

Action: Depending on data set options at the transmission control unit:

1. If the data set has Auto Disconnect and the device was dialed by the computer, issue a CONTROL Disable or a WRITE Break macro instruction on the same DECB.

   Example: [symbol] CONTROL DECBNAME,TD,MF=E

2. If the data set does not have the Auto Disconnect Feature, or if the data set has Auto Disconnect and the device was not dialed by the computer issue a WRITE Disconnect macro instruction against the same DECB. Then issue a WRITE Initial or WRITE Connect to call another terminal, or a READ Initial or READ Connect to await another call.

Write Action Description Five

Explanation: While establishing a connection with the WRITE Connect or WRITE Initial macro instructions, the 3275 failed to respond to your program's transmitted ID sequence within the TCU 3-second time-out interval. This is probably a terminal hardware error; this particular terminal should probably not be called again until checked.

Action:

1. Issue a WRITE Disconnect macro instruction on the same DECB to disconnect and disable the line.

2. Issue another WRITE Initial or WRITE Connect macro instruction to the same terminal (if immediate retry is desired), or to another terminal.

Write Action Description Six

Explanation: After your WRITE Initial has successfully established a connection with the 3275 (including receipt of the ID sequence), the 3275 has failed to respond to the text block transmitted as part of the Write Initial.

Action: Same as Write Action Description Five.

Write Action Description Seven

Explanation: A text block was sent to the 3275 using a Write Continue, but the 3275 failed to acknowledge the block within the TCU's 3-second time-out interval.

Action:

A. Issue a WRITE Inquiry macro instruction to ask the 3275 to retransmit its last response. The response may have been garbled or a momentary line-loss may have occurred. When issuing a WRITE TQ in DOS, you should provide an area for the response to be read into. OS provides this area. There are two ways to provide BTAM with pointers to this area:

   1. The sNLIST technique. You provide a parameter list pointer with the ENTRY operand of the WRITE TQ. The parameter list contains the fullword address of the response input area, and a fullword constant containing the length of the input area (2 is sufficient).

      For Example:

```
[symbol] WRITE DECBNAME,TQ,,,,PARMLIST,MF=E
PARMLIST DC A (RESPAREA) ADDR OF RESPONSE INPUT AREA
         DC F'2'         LENGTH OF RESPONSE INPUT AREA
RESPAREA DS CL2          RESPONSE INPUT AREA
```

   2. The DECB extension technique. You provide the address and length of the response input area in the WRITE TQ macro instruction itself, which then stores the information in the DECB extension.

      Example:

```
[symbol] WRITE DECBNAME,TQ,,(,RESPAREA),(,2)MF=E
```

      The method you use must agree with the specification of the BTMOD macro instruction DECBEXT parameter. If BTMOD specifies DECBEXT=YES, you must use the DECB extension technique, and vice-versa. The BTMOD parameter choice may be dictated by other terminal types in the system. Conversational reads and writes to a 2770 require the DECB extension, and you would then have to use the same technique.

B. If the WRITE TQ completes successfully, you may send another block of data with WRITE Continue, or end with WRITE EOT. If WRITE TQ is not successful, issue a WRITE Disconnect followed by WRITE Initial or WRITE Connect, as desired.

Write Action Description Eight

Explanation: The 3275 has failed to respond to the WRITE Inquiry macro instruction; BTAM has retried the number of times specified in the DTFBT. Further attempts are probably useless.

Action: Same as Write Action Description Five.

Write Action Description Nine

Explanation: You issued a WRITE EOT or WRITE Reset macro instruction to relinquish control of the line, and the 3275 has not bid for control within 7 seconds for OS, or the number of seconds represented by 3 x the number of retries specified in DTFBT in DOS.

Action: You should keep count of the number of consecutive occurrences of this type of time-out, and if a reasonable number is exceeded, disconnect the line and reinitialize for another call.

You could also issue a READ Inquiry Monitor, which will not time out, following this first completion of the WRITE EOT or WRITE Reset.

Alternately, you could issue a WRITE EOT Monitor or WRITE Reset Monitor, which also do not time out, instead of the WRITE EOT.

If the Monitor macro instructions (which do not time out) are used, be aware of the possibility an operator may leave the terminal without requesting disconnection, which could result in substantial line toll costs. You should perhaps set a timer interval when issuing a Monitor operation, and stop the operation with a RESETPL macro instruction if the operation is not complete within the time interval.

Write Action Description Ten

Explanation: A completion condition unknown to 3275 Dial support has occurred, which is probably caused by a programming error such as issuing an invalid macro instruction sequence.

Action: Obtain as complete a picture of current system status as possible, including, in order of importance: the DECB (with any user extensions you may have appended); DTFBT for the line group; any terminal-status tables you may have in your program; I/O buffers; and possibly application program areas. These should be dumped to an external device for printing and analysis. You should then issue a WRITE Disconnect to disable the line, followed by an Initial or Connect macro instruction to prepare for further operation.

Write Action Description Eleven

Explanation: An ID sequence other than any specified in the DFTRMLST macro instruction has been received from the 3275 during a Write Initial or Write Connect operation.

Action: Issue a WRITE Disconnect to hang up and disable the line, followed by another WRITE Initial or WRITE Connect to another device, as appropriate.

Write Action Description Twelve

Explanation: The 3275 has responded with WACK to your attempt to establish communications with a Write Initial or Write Connect. This indicates a "busy" condition involving the keyboard, operator ID card reader, or printer.

Action: Issue a WRITE Disconnect to disconnect and disable the line, followed by another WRITE Initial or WRITE Connect to establish connection with the screen or another terminal.

Write Action Description Thirteen

Explanation: The 3275 has responded WACK to your transmission of text. This occurs only if you had the Start-Print bit on in the Write Control Character. WACK indicates that the text has been successfully received and that printing has started. No further text transfers are possible until the printing is finished.

Action: Issue a WRITE EOT Monitor or WRITE Reset Monitor to relinquish control of the line and monitor for completion. When this WRITE TRM completes, issue a READ Continue to receive the sense/status message from the 3275 indicating device end (sense/status bytes = X'C240'). On receipt of this message and its following EOT, you may initiate any new transmissions you desire, or monitor the line for terminal activity.

LOCAL EVENT COMPLETION ANALYSIS

The information in this section should help you design or code the portion of a terminal control program which, upon completion of 3270 I/O operation, analyzes the circumstances of the completion and decides the proper action.

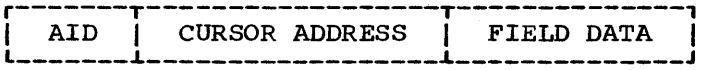
The description of the 3270 local event completion analysis is organized in six parts. Four of the parts are flowcharts contained in Figures 57 through 60. These flowcharts are a logical sequence in which completion information can be analyzed after a Read or Write operation. The flowcharts refer to the action descriptions that immediately follow the flowcharts.

The action descriptions are in the following format:

- An explanation of the causes of the completion condition
- The advised actions and comments, where appropriate

Read Action Description One

Explanation: The Read operation has completed successfully. The input message may take one of several formats.



If the operator has initiated the transaction other than with the TEST REQUEST key, the message has the above general format. The following variations are also possible:

- If the CLEAR key or Program Attention key was pressed, the AID byte is the only character received.
- If there are no modified fields and the operation was a READ MODIFIED, only the AID byte and cursor address are received.

Action: The action taken depends on the line control program. You may:

- Check for the availability of output and if present, write to a terminal with a WRITE Initial (TI), WRITE Erase (TS), or issue a WRITE Unprotected Erase (TUS) to the same format again. WRITE Unprotected Erase does not transmit text.
- If no output is available, issue a READ Initial (TI) so that the next operator action is recognized.

Read Action Description Two

Explanation: The previously initiated READ Initial (TI) was terminated without message receipt, by a RESETPL macro instruction.

Action: The appropriate action depends on the line control program. You may:

- If there is output for the control unit, issue a WRITE Initial (TI) or WRITE Erase (TS) to send the message.
- Resume the Read Initial (TI).
- Issue a CLOSE macro instruction to terminate operations on the control unit.

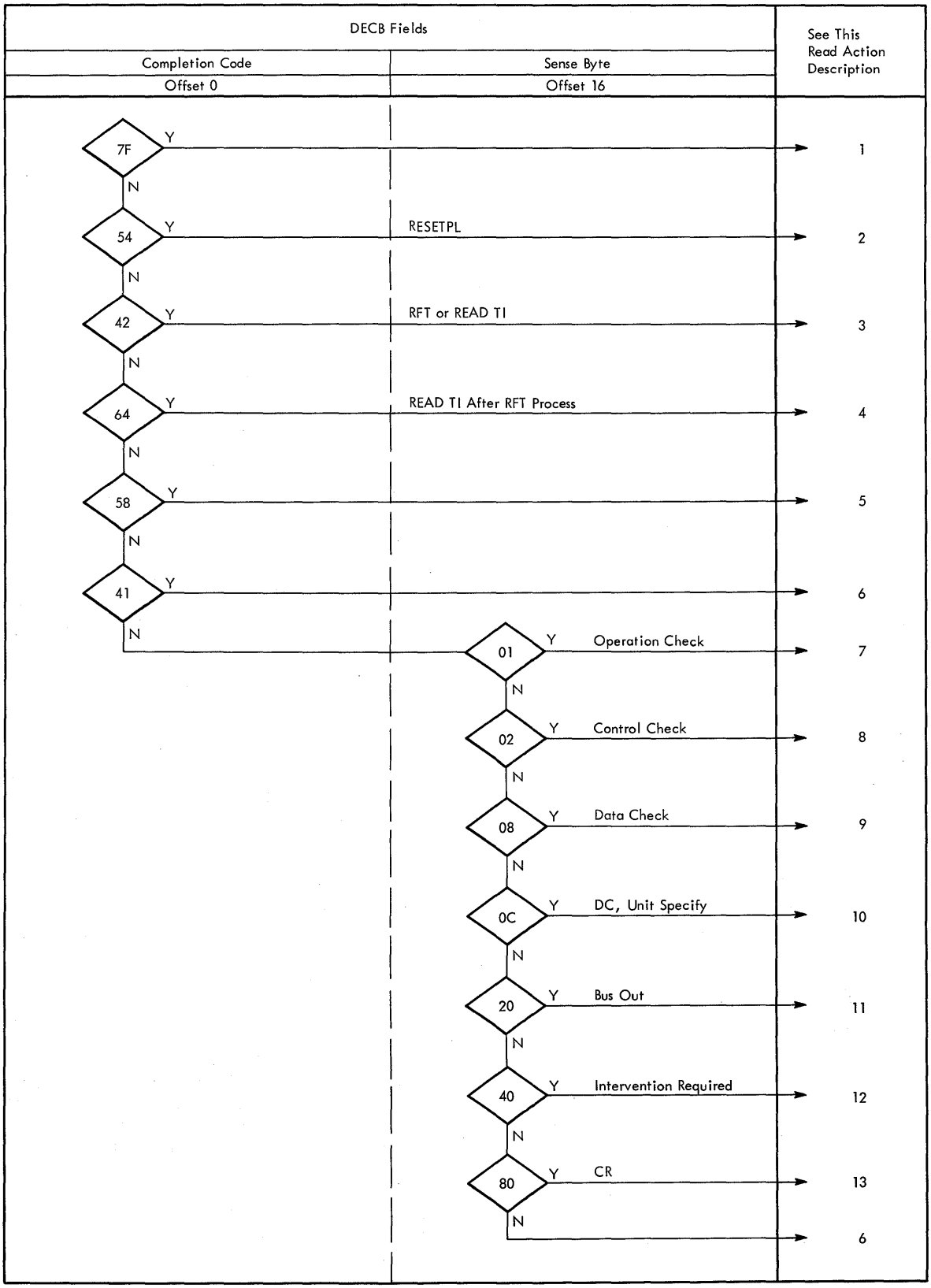


Figure 57. DOS BTAM local READ completion analysis

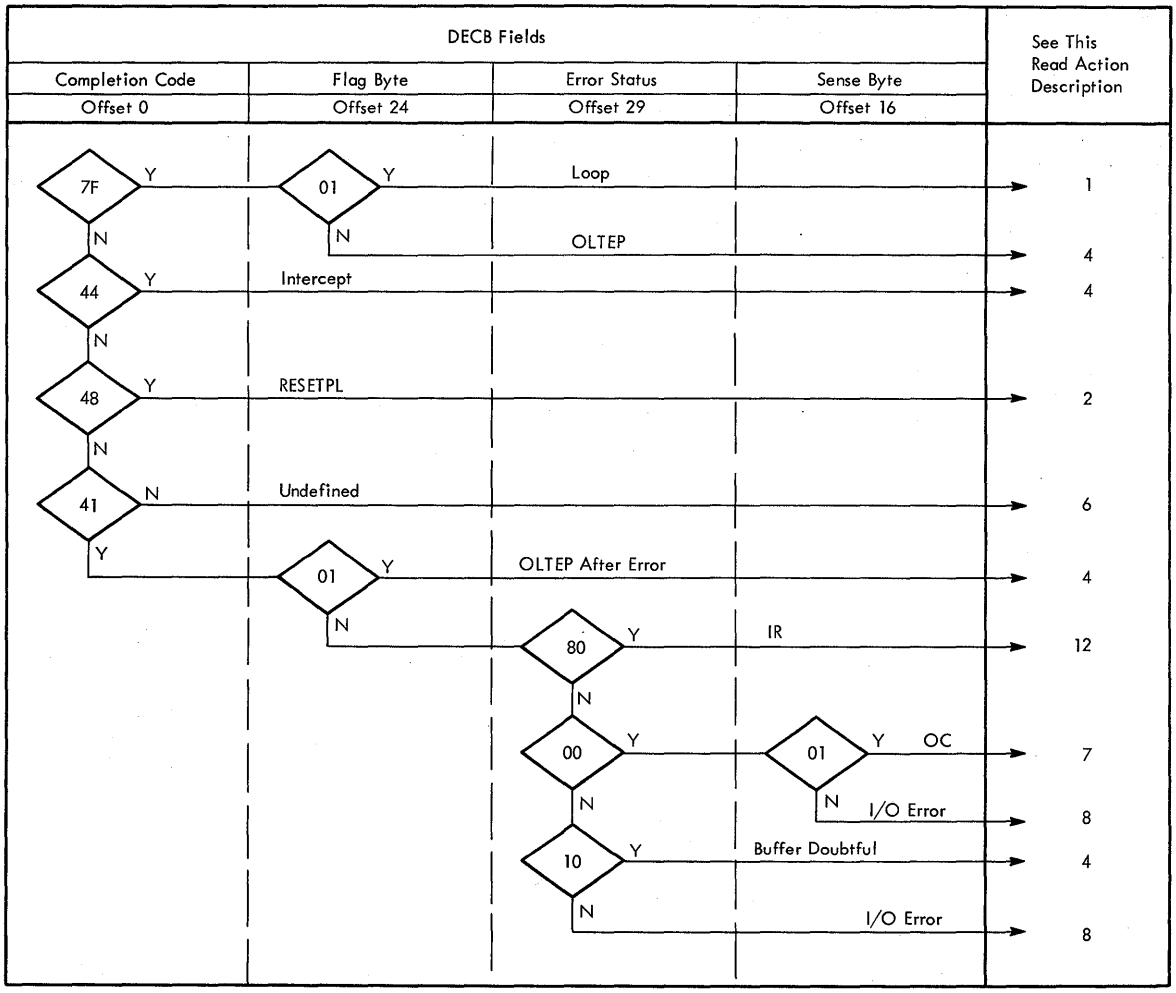


Figure 58. OS BTAM local READ completion analysis

Read Action Description Three

Explanation: A REQUEST-FOR-TEST message has been received in response to a READ Initial (TI), and a TWAIT macro instruction has not been issued.

Action: Issue a TWAIT macro in the form:

```
TWAIT  (R1),TERMTST,(R2)
```

(R1) is a register that contains the address of the DECB, which is posted as complete on the satisfaction of the TWAIT; (R2) is loaded with the address of the DECB, which is posted with the X'42' completion code. The completion code should not be altered prior to the issuance of the TWAIT.

If the online terminal test facility is not available, the REQUEST-FOR-TEST message might be processed the same as a CLEAR key depression.

Read Action Description Four

Explanation: The contents of the 3270 buffer are unreliable due to previous processing of a REQUEST-FOR-TEST message.

Action: The entire buffer must be reinitialized with a WRITE Erase (TS) macro instruction. This could require maintaining an image of the device buffer during processing because the buffer may be the cumulative result of multiple I/O transactions.

Read Action Description Five

Explanation: BTAM has detected a cancel condition.

Action: Take a PDUMP or snap dump of the system. Issue an operator awareness message and terminate the system after recording checkpoint/restart information, if required.

Read Action Description Six

Explanation: This is an unrecognized error condition that should not occur. The probable cause is a software error.

Action:

- Take a PDUMP or snap dump of the system for analysis.
- Notify the operator of the condition.
- Issue a READ Initial (TI) to resume input.

Read Action Description Seven

Explanation: The 3272 has detected an operation check. Possible causes are:

- The data stream transmitted as the result of a READ Modified from Position (TMP) or READ Buffer from Position (TBP) contains an illegal buffer address.
- The data streams contain a Write Control Character that specified start print.

Action:

- Take a snap dump or PDUMP of the offending data streams for analysis.
- Issue a 'transaction cancelled' message to the terminal.
- Inform the system operator of the occurrance.

Read Action Description Eight

Explanation: The 3272 has detected a control check condition. The addressed device failed to perform an operation or respond to the 3272 within a period of time determined by the control unit.

Action:

- Retry the failing operation the specified number of times.
- Notify the operator of the occurrence.
- Mark the terminal as out of service.
- Remove the terminal from the LCB list by issuing a CHGNTRY macro instruction of the form:

  (for OS)

  CHGNTRY (R1),ATTLST, (R2),,SKIP

(for DOS)

CHGNTRY (R1),ATTLST,(R2),SKIP

R1 is loaded with the address of the DTF/DCB and R2 is loaded with the relative line number of the terminal.

- After repairs have been made, the terminal can be placed back in service in response to an operator command through the use of a CHGNTRY macro instruction with the activate parameter.

Read Action Description Nine

Explanation: The 3272 has detected a data check.

Action: Same as Read Action Description Eight.

Read Action Description Ten

Explanation: A 3284/3286 printer or 3277 display has detected a data check condition.

Action:

- The entire device buffer must be reconstructed with an ERASE Write command (TS).
- If desired, the failing operation may then be retried. This may require maintaining an image of the current buffer content which may consist of several I/O operations.
- It may be preferable to issue a WRITE Erase (TS) command indicating that the transaction has aborted.
- Proceed as in Read Action Description Eight.

Read Action Description Eleven

Explanation: The 3272 has detected a bus out check (incorrect parity on a command or data received from the channel).

Action: Same as Read Action Description Eight.

Read Action Description Twelve

Explanation: The addressed device is unavailable (powered off or not operational).

Action:

- Notify the system operator of the condition.
- Take the terminal out of service. See Read Action Description Eight for an example of the CHGNTRY macro instruction.
- Reissue the failing macro instruction if it was READ Initial (TI).

Read Action Description Thirteen

Explanation: The 3272 has detected an invalid command.

Action: Same as Read Action Description Seven.

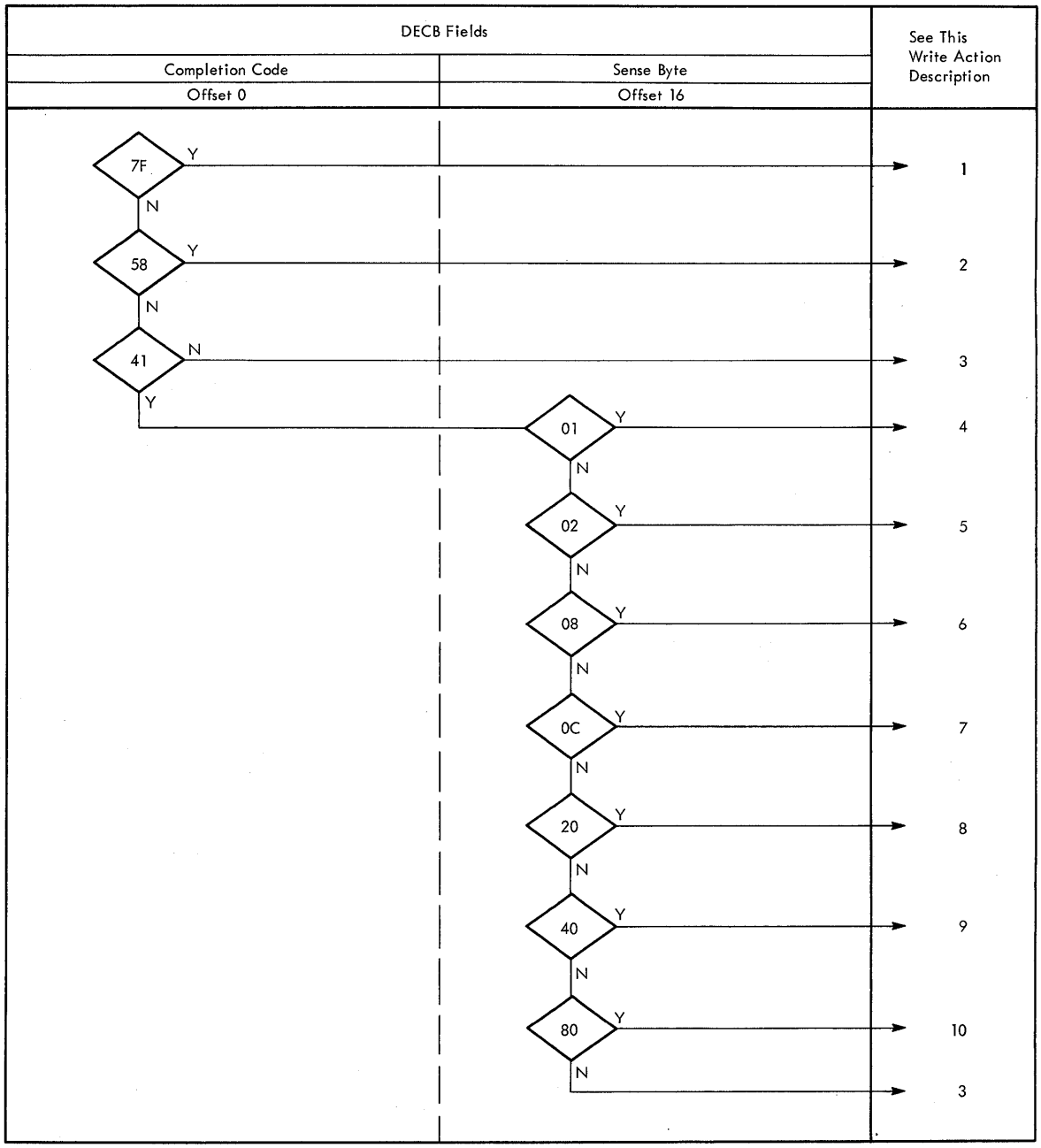


Figure 59. DOS BTAM local WRITE completion analysis

Write Action Description One

Explanation: The Write operation has been posted complete without error. For a DOS printer this only signals that the print operation has begun. Print errors which occur after channel end are posted through a return code '30' on the subsequent Read or Write operation.

Action: Check for the availability of additional output for other terminals on this control unit and send a WRITE Erase (TS) or WRITE Initial (TI). Issue a READ Initial (TI).

Note: Channel end posting for printers releases the DECB prior to the relatively long mechanical print cycle. However, channel end posting

does not immediately initiate the next output. To do this, define the printers at system generation as 3277s. An additional DECB must be defined for each printer. After the printer's DECBs are opened, they should be skipped for input through a CHGNTRY SKIP (see Read Action Description Eight). This approach provides device end posting for printers.

Write Action Description Two

Explanation: BTAM has detected a cancel condition, which is probably a user software problem.

Action: Same as Read Action Description Five.

Write Action Description Three

Explanation: This is an undefined error that should not occur, probably caused by software.

Action: Same as Read Action Description Six.

Write Action Description Four

Explanation: The 3272 has detected an operation check. Possible causes are:

- The data stream contains an illegal buffer address.
- The data stream ends prior to the completion of an order sequence.

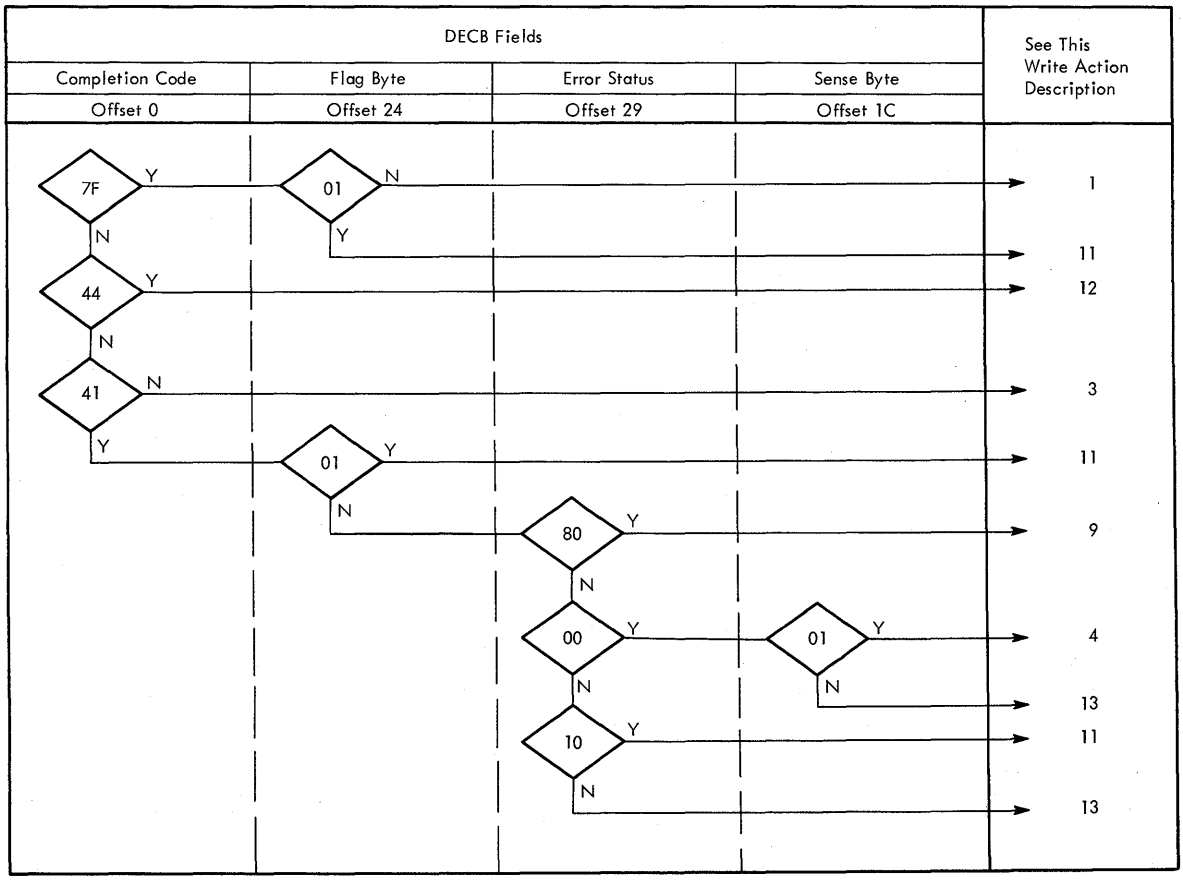


Figure 60. OS BTAM local WRITE completion analysis

Action: Same as Read Action Description Seven.

Write Action Description Five

Explanation: The 3272 has detected a control check. The device failed to perform an operation or respond to the 3272 in the amount of time allowed by the control unit.

Action: Same as Read Action Description Eight.

Write Action Description Six

Explanation: The 3272 has detected a data check. This is a hardware error.

Action: Same as Read Action Description Eight.

Write Action Description Seven

Explanation: A 3284/3286 printer or 3277 display has detected a data check. This is a hardware error.

Action: Same as Read Action Description Ten.

Write Action Description Eight

Explanation: The 3272 has detected a bus out check (incorrect parity on a command or data received from the channel).

Action: Same as Read Action Description Eight.

Write Action Description Nine

Explanation: The addressed device is unavailable (powered off or not operational).

Action: Same as Read Action Description Twelve.

Write Action Description Ten

Explanation: The 3272 has detected an invalid command.

Action: Same as Read Action Description Seven.

Write Action Description Eleven

Explanation: The device buffer is unreliable due to diagnostic testing after completion of the previous output operation.

Action: Same as Read Action Description Four.

Write Action Description Twelve

Explanation: The input/output request was rejected due to an error which occurred following the previous operation or request-for-test processing.

Action: See Read Action Description Four.

Write Action Description Thirteen

Explanation: A permanent I/O error has occurred.

Action: Same as Read Action Description Eight.

SENSE/STATUS ANALYSIS

Unlike previous terminal systems which told you only that an error had occurred, the 3270 Information Display System has a self-diagnosis system to inform the central site of error or completion conditions. In remote configurations this sense/status information is communicated in a special message format as illustrated in the Read Completion Analysis sections (Read Action Description Three for remote dial, Read Action Description One for remote leased multipoint).

Proper analysis and use of the sense/status bytes may improve system availability. In many cases, a specific retry operation can correct an error condition and allow normal system operation to proceed. Also, conditions requiring manual intervention, such as a powered-down display or printer, or a printer out of paper, can be quickly identified. Proper personnel can be notified to correct the situation. Serious hardware malfunctions may be identified, logged out, and communicated to proper maintenance personnel as an aid in diagnosing and correcting the problem.

The following charts and figures outline the possible sense/status byte combinations, cause of occurrence, meaning, and suggested corrective actions. Figure 61 shows the combinations of conditions.

The following information is included in Figure 62:

1. Condition
2. Hexadecimal representation for S/S message.
3. Whether condition applies to display, printer, or either.
4. Whether condition applies to 3271, 3275, or either.
5. Whether condition transmitted during specific poll only, or either specific or general poll.
6. Whether condition detected during:

   SA (selection addressing)
   SP (specific poll)
   GP (general poll)
   CMD (a 3270 command)
   ASYNC (device idle or printout)

7. Hardware explanation of condition.
8. Suggested user action.

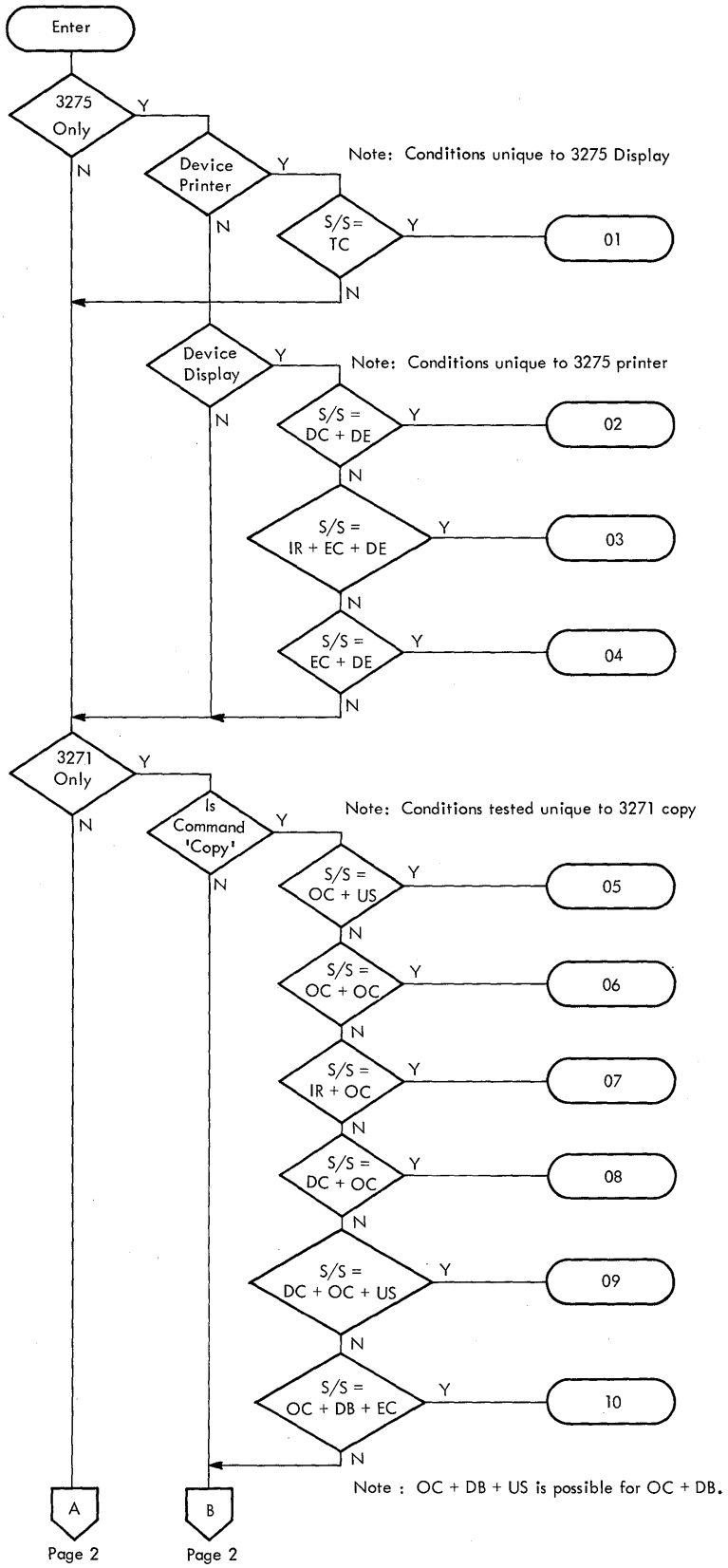


Figure 61 (part 1 of 2). Sense/status conditions, keyed to Figure 62

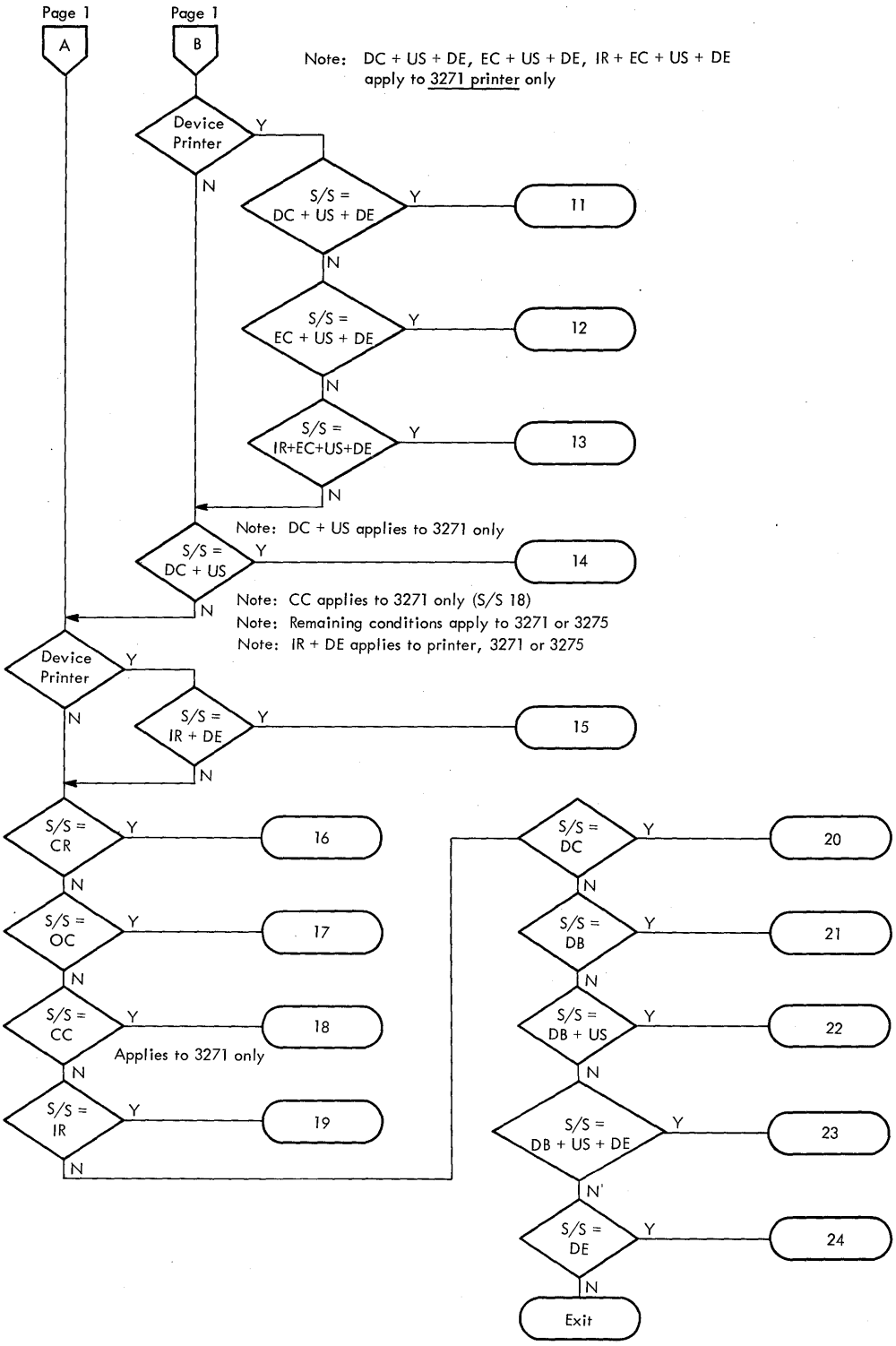


Figure 61 (part 2 of 2). Sense/status conditions, keyed to Figure 62

TABLE ID	CONDITION	S/S MSG IN HEX	DEVICE TYPE	3271 OR 3275	POLL TYPE	DETECTED DURING	DEVICE RESPONSE	COMMANDS	MEANING	IN FIG. 63 DO THIS
01	TC	C140	Display	3275	Spec.	CMD	EOT	Read/Write	A BCC error is detected at the 3275.	1
02	DC + DE	C2C4	Printer	3275	Gen.	Async.	N/A	N/A	A data check error (parity or cursor) occurred in the 3275 buffer during printout.	2
03	IR + EC + DE	C2D8	Printer	3275	Spec. or Gen.	Async.	N/A	N/A	An error occurred during a printout (character generator or disabled print mechanism).	3
04	EC + DE	C2C8	Printer	3275	Spec. or Gen.	Async.	N/A	N/A	Same as 03	4
05	OC + US	C4C1	Display or Printer	3271	Spec.	CMD	EOT	Copy	Attempt to issue a copy command with the "from" address not authorized for copy function (the device has a buffer).	5
06	CC + OC	40C3	Display or Printer	3271	Spec.	CMD	N/A	Copy	The "from" address in the copy command is malfunctioning; timeouts are occurring between the 3271 control unit and the "from" device buffer.	6
07	IR + OC	40D1	Display or Printer	3271	Spec.	CMD	EOT	Copy	The "from" device in the copy command is not available and must be readied.	7
08	DC + OC	40C6	Display or Printer	3271	Spec.	CMD	EOT	Copy	The 3271 or the "from" device detect a data check (parity or cursor check) during execution of the copy command.	8
09	DC + OC + US	C4C5	Display or Printer	3271	Spec.	CMD	EOT	Copy	Same as 08.	9
10	OC + DB	C8C1	Display or Printer	3271	Spec.	CMD	EOT	Copy	The "from" device specified in the copy command is "busy" (executing an operation, a printout, reading data, or performing a keyboard operation). The copy command is aborted.	10
11	DC + US + DE	C6C4	Printer	3271	Spec. or Gen.	Async.	N/A	N/A	Figure 62. DC + US indicates a data check in the printer device buffer during printout. DE indicates end of printout.	11
12	EC + US + DE	C6C8	Printer	3271	Spec. or Gen.	Async.	EOT	Spec. and Gen. Poll	EC + US indicates a printout error (disables print mechanism or character generator error), but the error was recovered from and device end (DE) has occurred.	12
13	IR + EC + US + DE	C6D8	Printer	3271	Spec. or Gen.	Async.	EOT	Spec. and Gen. Poll	DE indicates end of printout. IR + EC + US indicate an unrecoverable mechanical error.	13
14	DC + US	C4C4	Display or Printer	3271	Spec. or Gen.	SA, SP, GP, CMD, Async.	RV1, EOT	Spec. and Gen. Poll	A parity or cursor check on data the addressed device is sending to the control unit.	14

Figure 62 (part 1 of 2). Sense/status byte information and its meaning

TABLE ID	CONDITION	S/S MSG IN HEX	DEVICE TYPE	3271 OR 3275	POLL TYPE	DETECTED DURING	DEVICE RESPONSE	COMMANDS	MEANING	IN FIG. 63 DO THIS
15	IR + DE	C250	Printer	3271 or 3275	Spec. or Gen.	Async.	RV1, EOT	Spec. and Gen. Poll	The addressed printer is out of paper, ATS power is turned off, or cover is open, and the selection addressing sequence has detected a change in status that posts device end.	15
16	CR	4060	Display or Printer	3271 or 3275	Spec.	CMD	EOT	Read, Write	An invalid or illegal 3270 command was received.	16
17	OC	40C1	Display or Printer	3271 or 3275	Spec.	CMD	EOT	Read, Write	1. An invalid command sequence (ESC not in second data position). 2. An illegal buffer address was received. 3. An incomplete order sequence was received. 4. An overrun condition occurred.	17
18	CC	40C2	Display or Printer	3271	Spec. or Gen.	SA, SP, GP, CMD	RV1, EOT	Selection or poll, write commands, specific poll	The addressed device fails to complete an operation or respond to a 3271 in a certain time (timeout check).	18
19	IR	4050	Display or Printer	3271 or 3275	Spec.	SA, SP, CMD	RV1, EOT	Selection of device or poll, read commands, write, copy, erase/ write	Addressed device is not available or not ready.	19
20	DC	40C4	Display or Printer	3271 or 3275	Spec. or Gen.	SA, SP, GP, CMD, Async.	RV1, EOT	Selection or poll, read commands, write commands, specific and general poll	The 3271 detects a parity or cursor check on its buffer during command instruction.	20
21	DB	C840	Display or Printer	3271 or 3275	Spec.	SP	EOT	Specific Poll	The addressed device is busy.	21
22	DB + US	4C40	Display or Printer	3271 or 3275	Spec.	CMD	RV1	Read, copy, or write command	The addressed device is busy.	22
23	DB + US + DE	4E40	Display or Printer	3271 or 3275	Spec.	CMD	EOT	Read or write command	The addressed device becomes "not busy" before a specific poll is issued to retrieve the DB + US status.	23
24	DE	C240	Display or Printer	3271 or 3275	Spec. or Gen.	SP, GP	EOT	Specific poll, general poll	Poll finds device ready or available although the 3271 or 3275 had previously recorded device unavailable. Note: When a device is powered on, DE is generated. When a device is busy and an attempt is made to poll or address this device, a DE sense/ status message is generated.	24

Figure 62 (part 2 of 2). Sense/status byte information and its meaning

1. Try to reconstruct the entire buffer if the write data stream includes data or if any command in the chain prior to this command does not have an SBA immediately following the WCC. If the condition continues, the error is assumed unrecoverable. As an alternative, retry the current operation three times before classifying the device as unrecoverable (in need of maintenance).

2. Reconstruct the entire buffer and retry up to three times before classifying the device as unrecoverable (in need of maintenance). As an alternative, retry a write operation three times before classifying the device as unrecoverable.

3. The action depends on whether or not printout is to be retried (no indication exists of what was printed). If printout is required, manual intervention is required to ready the device. Then either rewrite the buffer (WCC and no data stream) or reconstruct the buffer and retry.

4. Same as Number 3, IR+EC+DE: if retry is required, try manual intervention to ready the device and retry. See Number 3 for complete details.

5. The implication is a programming or application error, as the "from" device must not have a locked buffer.

6. A. Execute a new selection addressing sequence and retry the failing command twice. If this fails, assume the device needs maintenance, and:

   B. After maintenance, if applicable, restructure the device buffer beginning with Erase/Write.

7. After the "from" device is available, transmit the COPY command again.

8. Transmit a selection addressing sequence again and transmit the COPY command at least twice before classifying the device as unrecoverable and in need of maintenance. After maintenance, if possible, reconstruct the "from" device buffer.

9. Same as Number 8.

10. Periodically issue a specific/general poll on the "from" device (with general poll, the CU). When the "from" device is not busy a device end (DE) sense/status message is transmitted, allowing the COPY command to be retried.

11. Attempt to reconstruct the printer buffer and to retry the failing WRITE command chain at least three times before classifying the device as unrecoverable. If this is not possible, the failing command chain should be retried at least three times.

12. This is an application development. Since recovery is implied, this may be treated as completion, or the operation may be retried.

13. The exact action depends on the application, as it must be determined whether or not another printout is required. The device must be readied and the failing operation retried. See Number 3.

Figure 63 (part 1 of 2). Sense/status recovery actions

14.	If possible, reconstruct the device buffer and try the failing command chain again up to three times before classifying the device as unrecoverable (in need of maintenance). Or you may retry the failing command up to three times before classifying the device as unrecoverable.
15.	Request operator intervention to correct the condition. Any subsequent retry need not write the data stream, as the device buffer remains intact. If this cannot be determined, reconstruct the device buffer and retry the failing chain of commands.
16.	This indicates a programming or application error.
17.	The operation should be retried. Continued failure implies an application programming problem which could be detected by analyzing the failing write data stream.
18.	Reconstruct the entire buffer and retry the failing command chain up to three times before classifying the device as unrecoverable. Or you may retry the failing command up to three times before classifying the device as unrecoverable.
19.	If the addressed device is a printer, it is out of paper, its cover is open, or a disable condition exists that requires operator intervention. If the device is a display, it is not available. After the device has been made available, retry a WRITE command with data stream = zero, as the device buffer is still intact.
20.	Reconstruct the device buffer and retry the failing command or chain up to three times before classifying the device as unrecoverable.
21.	Periodically issue a specific poll to check DE status when the device becomes "not busy."
22.	Periodically issue specific or general polls to check DE sense/status when the device becomes "not busy." Then retry the failing command or chain up to three times, or reconstruct the device buffer, before classifying the device as unrecoverable.
23.	Reconstruct the device buffer and retry the failing chain or command up to three times before classifying the device as unrecoverable.
24.	This depends on normal processing when an operation completes. With this sense/status message, you can record the end of a printout.

Figure 63 (part 2 of 2). Sense/status recovery actions

Introduction to
Programming the
IBM 3270

READER'S COMMENT FORM

GC27-6999-0

Your views about this publication may help improve its usefulness; this form will be sent to the author's department for appropriate action. Using this form to request system assistance or additional publications will delay response, however. *For more direct handling of such requests, please contact your IBM representative or the IBM Branch Office serving your locality.*

How did you use this publication?

- ☐ As an introduction
- ☐ As a reference manual
- ☐ As a text (student)
- ☐ As a text (instructor)
- ☐ For another purpose (explain)_____

_____

Please comment on the general usefulness of the book; suggest additions, deletions, and clarifications; list specific errors and omissions (give page numbers):

What is your occupation?_____

Number of latest Technical Newsletter (if any) concerning this publication:_____

Please include your name and address in the space below if you wish a reply.

Thank you for your cooperation. No postage stamp necessary if mailed in the U.S.A. (Elsewhere, an IBM office

Cut or Fold Along Line

GC27-6999-0

Your comments, please . . .

This manual is part of a library that serves as a reference source for systems analysts, programmers, and operators of IBM systems. Your comments on the other side of this form will be carefully reviewed by the persons responsible for writing and publishing this material. All comments and suggestions become the property of IBM.

Fold Fold

First Class
Permit 40
Armonk
New York

Business Reply Mail
No postage stamp necessary if mailed in the U.S.A.

Postage will be paid by:

International Business Machines Corporation
Department 636
Neighborhood Road
Kingston, New York 12401

Fold Fold

IBM®

International Business Machines Corporation
Data Processing Division
1133 Westchester Avenue, White Plains, New York 10604
(U.S.A. only)

IBM World Trade Corporation
821 United Nations Plaza, New York, New York 10017
(International)

Cut or Fold Along Line

Intro. to Programming the 3270 (File No. S360/S370-09) Printed in U.S.A. GC27-6999-0

GC27-6999-0

Intro. to Programming the 3270 (File No. S360/S370-09) Printed in U.S.A. GC27-6999-0

International Business Machines Corporation
Data Processing Division
1133 Westchester Avenue, White Plains, New York 10604
(U.S.A. only)

IBM World Trade Corporation
821 United Nations Plaza, New York, New York 10017
(International)