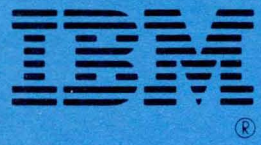


Document Content Architecture:

Final-Form-Text Reference

SC23-0757-0

Document Content Architecture:

Final-Form-Text Reference

First Edition (June 1983)

Changes are periodically made to the information herein; any such changes will be reported in subsequent revisions or Technical Newsletters.

References in this publication to IBM products, programs, or services do not imply that IBM intends to make these available in all countries in which IBM operates. Any reference to an IBM program product in this publication is not intended to state or imply that only IBM's program product may be used. Any functionally equivalent program may be used instead.

Publications are not stocked at the address given below. Requests for IBM publications should be made to your IBM representative or to the IBM branch office serving your locality.

A form for readers' comments is provided at the back of this publication. If the form has been removed, comments may be addressed to IBM Corporation, Department D31, 11400 Burnet Road, Austin, Texas 78758. IBM may use or distribute whatever information you supply in any way it believes appropriate without incurring any obligation to you.

PREFACE

This publication describes the Final-Form-Text Document Content Architecture and is intended for product designers and system programmers with experience in text processing. Final-form text is text that has already been formatted and is ready for presentation. The architecture has the following major purposes:

- To specify the interface for the interchange of final-form-text documents by describing the data, the controls, and the functions to be performed. The physical mechanisms to generate the data and control functions and the algorithms for implementing them are not specified. Rather, the emphasis is on the functions and the logical definition of the interface structures.

- To specify certain aspects of the interchange design and implementation that are necessary to assure that the interface requirements are met. This architecture provides direction for interface implementation selection.

- To provide a uniform interface definition for the interchange of final-form-text documents among a broad range of products.

How to Use this Publication

This manual is divided into three parts: Introduction, Interface Description, and Appendixes.

- Chapter 1 is an introduction to final-form text.

- Chapter 2 contains the interface description and includes the following:

  - Overview of the format used for the controls and definition of terms
  - Syntax and semantics of multibyte controls
  - Syntax and semantics of one-byte controls
  - Control sequencing, defaults, and exception handling.

- The appendixes provide the font style identifiers, a table of width values for proportionally spaced characters, and the document profile used for final-form-text documents.

Prerequisite Publication

Office Information Architectures: Concepts, GC23-0765, is a general introduction to the purposes, requirements, and subject of information interchange and should be read before this manual.

Related Publications

Documents that contain related conceptual information or serve as reference materials are:

- Document Content Architecture: Revisable-Form-Text Reference, SC23-0758
- Document Interchange Architecture: Concepts and Structures, SC23-0759
- Document Interchange Architecture: Document Library Services Reference, SC23-0760
- Document Interchange Architecture: Application Processing Services Reference, SC23-0761
- Document Interchange Architecture: Document Distribution Services Reference, SC23-0762
- Document Interchange Architecture: Transaction Programmer's Guide, SC23-0763
- Document Interchange Architecture: Interchange Document Profile Reference, SC23-0764
- SNA Format and Protocol Reference Manual: Architectural Logic, SC30-3112

CONTENTS

FIGURES

CHAPTER 1. INTRODUCTION

Final-Form-Text Document Content Architecture specifies the structure of the data stream used for interchange of text documents formatted for presentation.

Document Content Architecture

The purpose of Document Content Architectures is to define a uniform interface for interchanging information among products used in an office environment.

Related Architecture

Document Content Architecture: Revisable-Form-Text Reference provides the architecture specification for the information interchange of documents whose text is in revisable form.

PURPOSE

A final-form-text document consists of text and formatting information that controls the presentation of the text. The intent of Final-Form-Text Document Content Architecture is to establish a single interpretation of the text presentation control functions. In a document distribution environment, control over print fidelity by the document originator is a requirement. Therefore, it is critical that the data and controls be interpreted and processed in the same way from one product to another. Without common syntax and precise semantics, the same data stream when sent to different systems could result in printed or displayed documents that differ in the placement, or even the content, of the output text. Therefore, a primary objective of the final-form-text semantic definition is to provide a precise, unambiguous description of the semantics of the final-form-text controls.

OBJECTIVES

The Final-Form-Text Document Content Architecture has the following objectives:

- Provide a suitable base for supporting document distribution applications within and across office product lines
- Enable specification of high quality image output for users
- Provide a means of ensuring presentation (for example, print image) fidelity and information integrity
- Define a simple data stream structure capable of being processed sequentially by synchronous output devices
- Be independent of any specific device characteristics.

CONCEPTS

Final-Form-Text Architecture specifies a simple data structure of one-byte and multibyte controls for presenting the text. This data stream is designed for text document interchange to achieve information compatibility independent of product types and exchange medium.

Interface Structures and Definitions

Final-form text contains the definition of text and control functions describing formatted text. These definitions specify the output from the processes that create, revise, or format text and the input to the receiving processes that present the document. Text, as used here, means an ordered string of characters (graphic characters) that are suitable for representing and presenting information. Text is further ordered into units of composition and presentation referred to as lines. The lines of text, when assembled into an ordered collection, constitute a presentation unit called a page. A single page or a group of pages constitute a document that is the object or unit of transfer for interchange. The term print or presentation used here, includes displaying a document on a volatile medium such as a video display, reproducing a permanent image on paper or photo-sensitive medium, as with impact, ink jet, or photo printers, and recording the document image on some form of recording medium.

Control functions are designated by specific code points (control codes) within the character set used for the text string. The codes assigned to graphic characters are never assigned to control functions. The control codes are imbedded within the text string where a control function is to be activated. Control codes activate functions to position text, to activate a state condition, or to inform an operator about device operation. All the control functions that are supported in final-form text are either EBCDIC one-byte controls or SNA character string (SCS) extended controls.

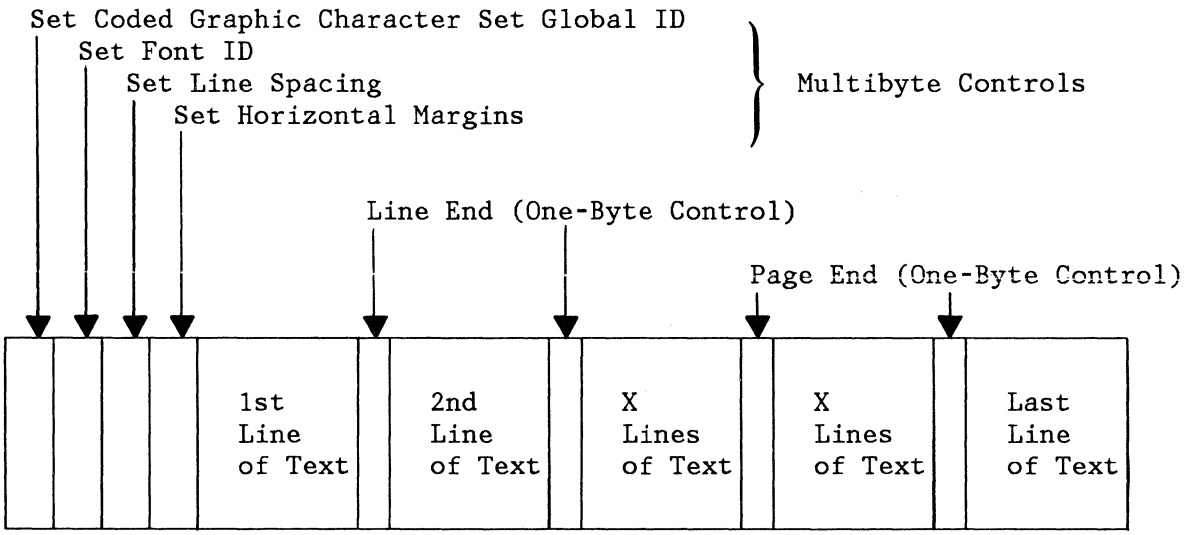
The coded graphic character set identification must always be included with the data stream when the data stream is created with a character set different from the default final-form-text character set value (see "Parameter Values and Default Values" on page 53). An example of a coded character graphic set supported by this architecture is included in Appendix B.

Interchange Requirements

The global requirement for information interchange is to define the architectural interfaces between office products. Information compatibility is required between products that exchange information using recording media, local attachments, and remote communications facilities. Final-Form-Text Architecture specifies a uniform technique for achieving information compatibility for the purpose of interchanging final-form text. The defined controls and processes for interchange must support the graphic representations of the text for multilingual applications.

Data Stream Organization

Figure 1 shows an example of a typical organization of the final-form-text data stream:



Note: There are no requirements for multibyte control functions to appear in the data stream.

Figure 1. Typical Final-Form-Text Data Stream Organization

The EBCDIC one-byte controls generally provide functions, such as line end, backspace, and indent, that have an immediate effect. The extended controls (multibyte) provide functions such as line spacing, horizontal and vertical margins, and tab settings that have a continuing, as well as an immediate, effect. A few of the multibyte controls remain in effect only until the line end, but most remain in effect until the end of the document or until they are redefined. Both one-byte and multibyte controls can occur throughout the data stream.

Document Interchange Architecture (DIA)

The DIA document unit structures used for interchange of final-form-text documents are Document Unit Type 3, Document Unit ID Document Type 2, and Interchange Document Profile Type 3 as defined in the DIA publications.

Segmentation, as defined by DIA for a document unit, may be used when the document unit contains a final-form-text document. Specific uses for segmentation would be a final-form-text document whose length is unknown when the length field is constructed or a document for which the entire document unit length exceeds 32767 bytes.

OPERATING ENVIRONMENT

The interface for the Final-Form-Text Architecture is defined primarily for formatted text document interchange within Systems Network Architecture (SNA) networks. For more information about the protocols used to communicate final-form-text documents, see SNA Format and Protocol Reference Manual: Architectural Logic. This does not prohibit using the final-form-text interface format with other communication protocols and procedures for information portability.

Note that interchange of information may require host application programming support. This architecture does not address such programming, or other implementation, and is concerned with providing an interface definition that supports access to office information in a consistent composition and format.

CHAPTER 2. FINAL-FORM-TEXT INTERFACE FUNCTIONAL DESCRIPTION

This section describes the control semantics and format of the final-form-text interface. Support notes are included to clarify an alternative that may exist and to achieve a consistent result.

In the control descriptions, bits are consecutively numbered from left to right starting with zero.

MULTIBYTE CONTROL FORMAT

The format for all of the multibyte controls is the same as the format sequence that is used for the SCS extended controls. The format is:

CONTROL-SEQUENCE-PREFIX,CLASS,COUNT,TYPE,PARM1,...PARMn

CONTROL-SEQUENCE-PREFIX(CSP) = A one-byte value that is defined in EBCDIC as X'2B'.

CLASS(CLS) = A one-byte binary value that designates a group of SCS extended controls that have a common purpose or attribute. The following classes have been assigned.

- D1 = Codes and character set controls.
- D2 = Sequential character and or line printer positioning controls.
- D4 = Paginating, text positioning, and processing controls.

COUNT(CNT) = A one-byte binary count of the number of bytes in the remainder of this control including this byte.

CONTROL TYPE CODE(TYPE) = a one-byte binary number that designates a specific control function.

PARM1 to PARMn = Parameters containing the control settings.

Notational Conventions

1. REQUIRED OCCURRENCE PARAMETERS are shown without any brackets. A value, within the range specified in the control description, must be specified for each required parameter when controls containing such parameters appear in the data stream.

2. OPTIONAL OCCURRENCE TERMINAL PARAMETERS are shown in small brackets. A terminal (trailing) parameter is the last parameter in a parameter string. When bracketed, a terminal parameter may be specified or omitted. The process is iterative for controls with multiple optional (bracketed) parameters. However, if a terminal parameter is specified, then all optional non-terminal (intervening) parameters in the control must appear, since parameters in multibyte extended controls are positional. When an optional parameter is omitted, the current value (which is the default value, if no prior specification of the parameter has occurred in this instance of the data stream) is used.

3. An ellipsis (...) indicates that a parameter, or string of parameters, can be repeated when specifying the control. Any architected limits on the number of repetitions allowed is defined in the control description itself. If no such limit is defined, the COUNT field (one-byte binary value limitation) is the only restriction that must be satisfied.

Support Note

The length of some multibyte controls can vary depending on the number of parameters specified. If the count excludes an optional parameter or part of an optional parameter, then the value for that parameter and any parameters to the right of it is not changed; that is, the COUNT field is used as specified. If a multibyte control is received in which the count exceeds the maximum value required to include all parameters, a Class 4 exception is raised since the additional values are considered to be unsupported parameters. If the count field excludes a required parameter, a Class 3 exception is raised.

FINAL-FORM-TEXT INTERFACE

This section describes the composition of the final-form-text interface. A summary of the contents of a final-form-text document is discussed and is followed by the detailed description of the control semantics and format for all the multibyte controls. The detailed semantics for all the final-form-text one-byte controls are contained in "One-Byte Controls" on page 40.

Final-Form-Text Document Contents

A final-form-text document is identified external to the data stream by the use of a data set label on recording media, or by the SNA header in an SNA environment, or by a DIA type 3 (Interchange) Document Unit with document type 2 in a DIA interchange environment. The boundaries of a final-form-text document unit are defined external to the data stream. The DIA type 3 document profile associated with the final-form text is described in Appendix C.

A final-form-text data stream contains only text and final-form-text controls. All the one-byte controls supported in final-form text are a subset of EBCDIC. Non-final-form-text controls that are part of a document when a document is entered by an operator are to be processed into final-form-text controls when a document is converted into a final-form-text document.

Final-Form-Text Term Definitions

The following definitions apply to Final-Form-Text Architecture.

Default Values and Initial Conditions

A default is a value that is assigned to a parameter when no explicit assignment of value has been made in the data stream. In Final-Form-Text Architecture, this occurs when a multibyte control with parameters has not been specified in the data stream or when an optional parameter with no prior assigned value is omitted. Final-Form-Text Architecture defines default values for all parameters with the exception of Set Exception Action (SEA; no default exception class or exception action), Set Font ID (SFG; no default font or font width), and Set Horizontal Tab Stops (STAB; no tab stops). These default values, defined in "Parameter Values and Default Values" on page 53, are the final-form-text initial conditions for the presentation process. This default set is equivalent to Initialization Set 1 of the Set Initial Conditions (SIC) control. The Final-Form-Text Architecture requires that the environment be logically re-established (initialized) with this set of default values before processing every final-form-text document. The default set may be overridden by including in the document an SIC control specifying an alternate set of defined (architected) initial values.

PRESENTATION SPACE

A presentation space is a two-dimensional surface upon which a final-form-text document is positioned via the controls embedded in the final-form-text data stream (or defaulted if unspecified). The presentation space is bounded by two parameter values: Set Presentation Page Size (SPPS) width and depth. The presentation position is never allowed to move outside the range of these two values. If not specified, the final-form-text default values for SPPS are 215.9mm x 279.4mm (8 1/2 inches x 11 inches) with the depth as 11 inches and the width as 8 1/2 inches.

A presentation position is defined by a pair of coordinates that specify the line number (vertical position) and the column number (horizontal position) where the baseline and left edge of the next graphic character will be positioned. The basic presentation space contains a matrix of presentation positions with an origin of line 1 and column 1. Line number one is the topmost presentation position that can be specified on the presentation space. This is the topmost presentation position at which text can appear vertically. Column number one is the leftmost presentation position that can be specified on the presentation space. This is the leftmost presentation position at which text can appear horizontally. The smallest unit defined in Final-Form-Text Architecture for either coordinate is 0.0176mm (1/1440th of an inch). In Final-Form-Text Architecture, the presentation space is equivalent to a physical page. The page is then physically manifested on some medium, such as paper or a display screen.

Optional(Exception Alternate Action) O(EAA) Controls

There is no requirement to support optional controls, but if they are supported, they must be executed according to the Final-Form-Text Architecture. If an optional final-form-text control, control parameter, or parameter value is not supported, an exception condition exists. An exception alternate action is an architecturally defined fallback action that is to be performed when an unsupported optional construct (control, parameter, parameter value) is detected and the SEA Action for the corresponding Exception Class is accept; that is, AC = 0. All optional controls, parameters, and parameter values that are unsupported must be recognized as such when they appear in a final-form-text data stream. The SEA Class or Action in effect when the unsupported control, parameter, or parameter value is encountered determines how the unsupported entity is to be handled. See "Parameter Values and Default Values" on page 53 for exception alternate actions for unsupported parameter values and "Exception Handling" on page 56 for a description of exception handling.

If an unsupported control is encountered and the SEA Action parameter in effect for unsupported controls is AC=0 (accept), then the final-form-text exception alternate action is to ignore the control.

If an unsupported parameter is encountered and the SEA Action parameter in effect for unsupported parameters is AC=0 (accept), then the parameter is ignored.

If an unsupported parameter value is encountered and the SEA Action parameter in effect for unsupported parameter values is AC=0 (accept), then the exception alternate action listed in "Parameter Values and Default Values" on page 53 is executed.

The originator of a final-form-text document must observe the requirements for the supported controls and parameters to ensure proper results. Minimum function receiving devices (devices with limitations due to performance, storage capacity, and functional capabilities) are required to detect device exception conditions when they occur, but they may not have facilities to check for all parameter values that could cause severe processing exceptions.

Mandatory (M) Controls without Parameters (One-Byte Controls and RLM)

Mandatory controls must be executed according to the Final-Form-Text Architecture. These controls cannot be defaulted to an exception alternate action.

Mandatory (M) Controls with Parameters (Multibyte Controls)

Mandatory parameter values must be executed according to the Final-Form-Text Architecture for the parameter values. Mandatory parameter values cannot be defaulted to an exception alternate action or value. Not all parameter values of mandatory multibyte controls are mandatory. See the final-form-text parameter value table ("Parameter Values and Default Values" on page 53) for a list of mandatory and optional parameter values. There is no requirement to support non-mandatory (optional) parameter values, but if they are supported they must be executed according to the Final-Form-Text Architecture. If a non-mandatory (optional) parameter value is not supported, an exception condition exists. Final-form-text exception handling for an unsupported parameter value is to process the exception according to the Action parameter of the SEA control in effect for Class 4 exceptions when the unsupported parameter value was encountered. If the SEA Action parameter in effect is AC=0 (accept) then the final-form-text exception alternate action listed in "Parameter Values and Default Values" on page 53 is executed. See "Exception Handling" on page 56 for a description of exception handling.

PSM and Justification Support Requirements

In addition to the mandatory (M) controls specified in "Summary of the Final-Form-Text Controls" on page 11, support of proportional spacing (PSM) and justification functions require additional controls to be supported during the printing or display of a final-form-text document.

PSM requires support of the Unit Backspace (UBS) control, when it appears in a final-form-text data stream.

The justification function requires support of either (or both) the Justify Text Field (JTF) or Set Justify Mode (SJM) control, when they appear in a final-form-text data stream. Support of SJM also requires:

- Support of the Right Margin (RM) of the Set Horizontal Margin (SHM) control
- Recognition of paragraph boundaries as specified in the SJM control definition
- If Unit Backspace is not supported but Set Justify Mode is, then the sequence of New Line, Unit Backspace must still be recognized as a paragraph boundary.

Tolerance on Units of Measure

The unit of measure (1440ths of an inch) is the lowest common denominator of units commonly in use (for example, 10ths, 12ths, 72nds, or 96ths). It is not required that a device be capable of operating at 0.0176mm (1/1440ths of an inch).

If a particular parameter value specified in n/1440ths inches is not supported, the closest supported value should be used. Print fidelity is considered satisfied if the character box is within (±) 0.423mm (1/60th inch) of the specified value (that is, of exact placement within the presentation space) in both the vertical and horizontal direction. If SEA AC = 2, for example, then this tolerance restriction of 0.423mm (1/60th inch) must be met or an exception exists causing termination with an error message or indicator. It is not a requirement to calculate the variance. An exception can simply be raised if the specified parameter value is not supported.

Summary of the Final-Form-Text Controls

A summary of the final-form-text controls is listed in Figure 2 on page 12. There are no controls that must appear in a final-form-text data stream (no mandatory occurrence requirements are placed on the originator). Instead, the Final-Form-Text Architecture requires initialization of all control parameter values to the default values specified in "Parameter Values and Default Values" on page 53 before presenting the document. When the data stream is processed, the parameter default values specified in "Parameter Values and Default Values" on page 53 are used until they are reset by control values in the data stream. These final-form-text default values therefore will be used when:

- A final-form-text control is not specified in the data stream.
- An optional parameter is omitted and no prior specification of that parameter occurred in the data stream (that is, current value = default value).

The multibyte controls are described in detail in "Multibyte Controls" on page 13. The one-byte controls are discussed in "One-Byte Controls" on page 40. Any one-byte controls or multibyte controls that are not listed in Figure 2 on page 12 are not allowed in a final-form-text data stream.

Note: The following one-byte controls have both a proper EBCDIC name and a historical Word Processing name. To avoid possible confusion, all references to Carriage Return (X'0D') will include the name ZICR (Zero Index Carrier Return) in parenthesis.

EBCDIC	Word Processing
Bell(BEL)	Stop(STP)
Carriage Return(CR)	Zero Index Carrier Return(ZICR)
Form Feed(FF)	Page End(PE)
Line Feed(LF)	Index(INX)
New Line(NL)	Carrier Return(CRE)
Required Form Feed(RFF)	Required Page End(RPE)
Required New Line(RNL)	Required Carrier Return(RCR)

Multibyte Controls	Class	Type	Support During Document Printing
Begin Overstrike (BOS)	D4	72	M
End Overstrike (EOS)	D4	76	M
Begin Underscore (BUS)	D4	0A	M
End Underscore (EUS)	D4	0E	M
Justify Text Field (JTF)	D2	03	O(EAA)
Page Presentation Media (PPM)	D2	48	O(EAA)
Release Left Margin(RLM)	D2	0B	M
Set Exception Action (SEA)	D2	85	M
Set FID thru GFID (SFG)	D1	05	M
Set CGCSGID (SCG)	D1	01	M
Set Horizontal Margins (SHM)	D2	11	M
Set Horizontal Tab Stops (STAB)	D2	01	M
Set Indent Level (SIL)	D2	07	M
Set Initial Conditions (SIC)	D2	45	M
Set Justify Mode (SJM)	D2	0D	O(EAA)
Set Line Spacing (SLS)	D2	09	M
Set Presentation Page Size (SPPS)	D2	40	O(EAA)
Set Print Setup (SPSU)	D2	4C	O(EAA)
Set Single Line Distance (SSLD)	D2	15	M
Set Vertical Margins (SVM)	D2	49	M

One-Byte Controls	Code Point	
Backspace (BS)	16	M
Bell (BEL) (STP)	2F	O(EAA)
Carriage Return (CR) (ZICR)	0D	M
Form Feed (FF) (PE)	0C	M
Horizontal Tab (HT)	05	M
Indent Tab (IT)	39	M
Line Feed (LF) (INX)	25	M
New Line (NL) (CRE)	15	M
Null (NUL)	00	M
Numeric Space (NSP)	E1	O(EAA)
Required Form Feed (RFF) (RPE)	3A	M
Required Hyphen (HYP)	60	M
Required New Line (RNL) (RCR)	06	M
Required Space (RSP)	41	M
Space (SP)	40	M
Subscript (SBS)	38	O(EAA)
Substitute (SUB)	3F	M
Superscript (SPS)	09	O(EAA)
Syllable Hyphen (SHY)	CA	M
Unit Backspace (UBS)	1A	O(EAA)
Word Underscore (WUS)	23	M

M = Mandatory
O = Optional(Exception Alternate Action)

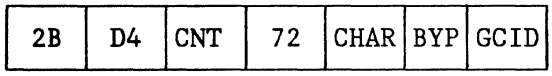
Figure 2. Final-Form-Text Control Summary

MULTIBYTE CONTROLS

This section describes all the multibyte controls that are valid in a final-form-text data stream. The default values initialized before processing every final-form-text document are indicated in each control as the underlined parameter value.

Begin Overstrike (BOS)

CSP,CLASS,COUNT,TYPE,GRAPHIC-CHARACTER,
BYPASS[,CODED-GRAPHIC-CHARACTER-SET-GLOBAL-ID]



The BOS (MANDATORY) control is the starting control of a pair of controls that identifies text that is to be overstruck with a specified graphic character. The ending control of the pair is End Overstrike (EOS). The Graphic-Character and Coded-Graphic-Character-Set-Global-ID parameters identify the specific graphic with which subsequent text is to be automatically overstruck. Once a BOS is activated, it stays activated until it is terminated by an EOS control or the end of the document.

CHAR = Graphic-character; a one-byte hexadecimal value from X'40' to X'FE' representing the overstrike character whose graphic is determined by the CGCSGID (character set and code page) parameter value. The graphic will be presented in the font currently active.

X'00' to X'3F' and X'FF' = Reserved.

BYP = Bypass; a one-byte hexadecimal number indicating text characters that are not to be overstruck by BOS.

X'00' = Bypass HT and IT (X'05' & X'39').
X'01' = Continuous Overstrike.
X'80' = Bypass blanks and tabs (X'40', X'05', & X'39').
X'02' - X'7F' and X'81' - X'FF' = Reserved.

CGCSGID = Coded Graphic Character Set Global ID; a concatenation of 2 two-byte numbers. The first two bytes identify the Graphic Character Set Global ID (GCSGID) expressed as a binary value from 1 to 65534. The second two bytes identify the Code Page Global ID (CPGID) expressed as a binary value from 1 to 65534.

GCSGID = The Graphic Character Set Global IDs are assigned with reserved values in the range of X'00nn' to be assigned to currently designated IBM character sets and values in the range of X'FFnn' to be used for customer defined character sets.
X'0000' = Use the GCSGID value currently active.
X'FFFF' = Reserved.

CPGID = The Code Page Global IDs are assigned with reserved values in the range of X 00nn' to be assigned to currently designated IBM character code assignments and values in the range of X'FFnn' to be used for customer designated character code assignments.

X'0000' = Use the CPGID value currently active.
X'FFFF' = Reserved.

Support Note

The Final-Form-Text Architecture places no requirements on when or precisely how to place the overstrike across the characters, as long as the semantic results are achieved. This may be done with a single or multiple keystroke, with an overstruck font, or it may be done a line or page at a time, or any other technique.

When the Bypass parameter value is set to X'00', white space resulting from a Horizontal Tab (HT) or Indent Tab (IT) control is not overstruck. Blanks (X'40') following the last printable graphic on the line and occurring before the line-end control are overstruck.

When the Bypass parameter value is set to X'01' (continuous), all graphic characters and all control white space except that which follows the last printable graphic on the line are overstruck. Control white space is white space resulting from SP (X'40'), a multibyte control, or a single-byte control. In the sequence of BOS, IT, and then text, leading white space resulting from an Indent Tab (IT) is overstruck. On subsequent lines with IT still in effect, white space that appears between the left margin and the temporary left margin established by the IT is not overstruck.

When the Bypass parameter value is set to X'80', white space resulting from blanks and tabs (SP, HT, & IT) is not overstruck.

Required Space (X'41') and Numeric Space (X'E1') are treated as printable graphics.

Support of the overstrike graphic character (CHAR) and continuous overstrike (BYP = X'01') is required.

If a specified CGCSGID is not supported, and SEA allows continued processing for the Class 1 exception, the default character set of the country of the product presenting the document should be used. If the Bypass parameter value specified

is not supported, and SEA allows continued processing, the exception alternate action for the Class 2 exception is continuous overstrike. The exception alternate action for invalid overstrike characters (undefined or non-graphic code points in the CGCSGID specified) is not defined by the Final-Form-Text Architecture. It is recommended, if SEA allows continued processing, that the BOS be ignored in this situation. (State management must still be considered and maintained.)

The CGCSGID parameter value specified in the BOS control remains in effect for the overstrike graphic until the corresponding EOS or end of document (that is, intervening SCG controls have no effect on the overstrike graphic presented).

The typestyle, font width, and font attributes of the overstrike character change as a result of SFG controls that occur within the BOS-EOS pair.

BOS-EOS pairs cannot be nested. If a BOS control follows another BOS control without an intervening EOS, a Class 1 exception condition exists. If SEA allows continued processing, it is recommended that overstriking continue using the parameter values specified in the subsequent BOS; that is, the second BOS that occurs in the sequence BOS...BOS.

There are no architectural occurrence restrictions between the BOS-EOS and BUS-EUS control pairs, other than the nesting restrictions for each individual pair of controls. This provides the capability to automatically overstrike a string of text with a maximum of two graphics (the specified underscore character and the specified overstrike character).

Semantically, BOS-EOS and BUS-EUS apply independently to the text they encompass. A BOS control requesting bypass of blanks, for example, causes blanks not to be overstruck even if a BUS control appearing within the BOS-EOS string underscored those blank spaces.

Line and page ending controls (NL, RNL, CR, FF, RFF), as well as Line Feed (X'25'), that occur while BOS is active terminate overstriking for that line. Overstriking is continued on the next line.

There are no syntactic restrictions on the occurrence of Carriage Return (X'0D'), Word Underscore (WUS), or backspace(s)-character(s) sequences within an overstruck text string. If they occur, however, the function they specify is logically applied first; that is, Spaces (X'40') that are backspaced over and then overstruck are not bypassed by a BOS control with BYP = X'80' (bypass blanks). This is different from the interaction between BOS and BUS noted above.

Moreover, these controls (CR, WUS, BS, and UBS) remain independent of the BOS semantics; that is, they acquire no additional capability themselves if they occur between a BOS-EOS pair. For example, the final-form-text justification function does not justify text containing backspaces; that is, the width of the backspace is not expandable for justification. This restriction on the use of justification with BS applies whether BOS is active or not.

A Word Underscore control occurring within a BOS-EOS string may be used to allow emphasis of an individual word within a string that is overstruck. There is no architectural requirement, however, that the overstrike not overstrike the WUS. A BOS control does not affect the word boundaries used for delimiting the word

to be underscored by WUS; that is, Spaces (X'40') to be overstruck by BOS are treated as word delimiters by WUS.

A Carriage Return (X'0D') control that (a) occurs while BOS is active or (b) is followed by a line of text containing a BOS does not affect the BOS function specified. The parameter values for BOS are used for overstriking the line, or partial line, following the CR control.

Subscripts and Superscripts are overstruck in the baseline position.

Substitute control (SUB) is logically resolved into the installation-defined graphic prior to overstriking. Therefore, the semantics of BOS are applied to the substituted graphic.

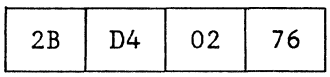
Extra white space (X'40'), inserted for justification (as a result of JTF or SJM), within a BOS-EOS sequence is overstruck unless Space (X'40') codes are explicitly excluded by the Bypass parameter of BOS.

Because the string of text to be overstruck can contain characters to be presented in a proportionally spaced font, special consideration must be given to ensuring that information is not lost. It is recommended, when overstriking proportionally spaced text, that the whole number of overstriking characters required for each overstrike area (after justification) be equal to the sum of the escapement units of the overstrike area divided by the escapement unit of the overstrike character. If the division results are less than one, at least one overstrike character must be presented in the overstrike area. In this single character overstrike case, adjacent graphics may be overstruck.

For optimum quality output, a precise correlation between the text and the overstrike graphic is recommended (that is, position the center of the overstrike character at the center of each text character, for example). Overstruck proportionally spaced fonts is another possible alternative for achieving the optimum in quality output.

End Overstrike (EOS)

CSP,CLASS,COUNT,TYPE



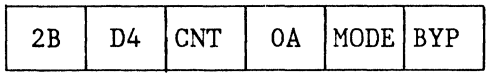
The EOS (MANDATORY) control is the ending control of the paired controls, Begin and End Overstrike. The EOS control terminates the BOS control, thereby delimiting the end of text that is to be overstruck. For information on the semantics of the BOS-EOS pair, please refer to "Begin Overstrike" on page 13.

Support Note

An unpaired EOS is considered a Class 1 exception condition. If SEA allows continued processing, such an EOS control is treated as an NUL (X'00') control.

Begin Underscore (BUS)

CSP,CLASS,COUNT,TYPE,MODE,BYPASS



The BUS (MANDATORY) control is the starting control of a pair of controls that identifies text that is to be underscored. The ending control of the pair is End Underscore (EUS). Once a BUS is activated, it stays activated until it is terminated by an EUS control or the end of the document.

MODE = A one-byte binary number specifying the type of underscoring to be used.

1 = Single underscore.
0 and 2 to 255 = Reserved.

BYP = Bypass; a one-byte hexadecimal number indicating text characters that are not to be underscored by BUS.

X'00' = Bypass HT and IT (X'05' & X'39').
X'01' = Continuous Underscore.
X'80' = Bypass blanks and tabs (X'40', X'05', & X'39').
X'02' - X'7F' and X'81' - X'FF' = Reserved.

Support Note

The Final-Form-Text Architecture places no requirements on when or precisely how to place the underscore under the characters, as long as the semantic results are achieved. This may be done with a single or multiple keystroke, with an underscored font, or it may be done a line or page at a time, or any other technique.

When the Bypass parameter value is set to X'00', white space resulting from a Horizontal Tab (HT) or Indent Tab (IT) control is not underscored. Blanks (SP = X'40') following the last printable graphic on the line and occurring before the line-end control are underscored.

When the Bypass parameter value is set to X'01' (continuous), all graphic characters and all control white space except that which follows the last printable graphic on the line are underscored. Control white space is white space resulting from an SP (X'40'), a multibyte control, or a single-byte control. In the sequence of BUS, IT, and then text, leading white space resulting from an Indent Tab (IT) is underscored. On subsequent lines with IT still in effect, white space that appears between the left margin and temporary left margin established by the IT is not underscored.

When the Bypass parameter value is set to X'80', white space resulting from blanks and tabs (SP, HT, & IT) is not underscored.

Required Space (X'41') and Numeric Space (X'E1') are treated as printable graphics.

Support of single underscore (MODE = 1) and continuous underscore (BYP = X'01') is required.

If the Bypass parameter value specified is not supported, and SEA allows continued processing, the exception alternate action for the Class 2 exception is continuous underscore.

BUS-EUS pairs cannot be nested. If a BUS control follows another BUS control without an intervening EUS, a Class 1 exception condition exists. If SEA allows continued processing, it is recommended that underscoring continue using the parameter values specified in the subsequent BUS; that is, the second BUS that occurs in the sequence BUS...BUS.

There are no architectural occurrence restrictions between the BUS-EUS and BOS-EOS (Begin Overstrike and End Overstrike) control pairs, other than the nesting restrictions for each individual pair of controls. This provides the capability to automatically overstrike a string of text with a maximum of two graphics (the specified underscore character and the specified overstrike character).

Semantically, BUS-EUS and BOS-EOS apply independently to the text they encompass. A BUS control requesting bypass of blanks, for example, causes blanks not to be underscored even if a BOS control appearing within the BUS-EUS string overstruck those blank spaces.

Line and page ending controls (NL, RNL, CR, FF, RFF), as well as Line Feed (X'25'), that occur while BUS is active terminate underscoring for that line. Underscoring is continued on the next line.

There are no syntactic restrictions on the occurrence of Carriage Return (X'0D'), Word Underscore (WUS), or backspace(s)-character(s) sequences within an underscored text string. If they occur, however, the function they specify is logically applied first; that is, Spaces (X'40') that are backspaced over and then overstruck are not bypassed by a BUS control with BYP = X'80' (bypass blanks). This is different from the interaction between BUS and BOS noted above.

Moreover, these controls (CR, WUS, BS, and UBS) remain independent of the BUS semantics; that is, they acquire no additional capability themselves if they occur between a BUS-EUS pair. For example, the final-form-text justification function does not justify text containing Backspaces; that is, the width of the backspace is not expandable for justification. This restriction on the use of justification with BS applies whether BUS is active or not.

A Word Underscore control occurring within a BUS-EUS string may be used to allow emphasis of an individual word within a string that is itself underscored. There is no architectural requirement, however, that the underscore not overstrike the WUS. A BUS control does not affect the word boundaries used for delimiting the word to be underscored by WUS; that is, Spaces (X'40') to be underscored by BUS are treated as word delimiters by WUS.

A Carriage Return (X'0D') control that (a) occurs while BUS is active or (b) is followed by a line of text containing a BUS does not affect the BUS function specified. The parameter values for BUS are used for underscoring the line, or partial line, following the CR control.

Subscripts and Superscripts are underscored in the baseline position.

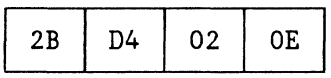
Substitute control (SUB) is logically resolved into the installation-defined graphic prior to underscoring. Therefore, the semantics of BUS are applied to the substituted graphic.

Extra white space (X'40'), inserted for justification (as a result of JTF or SJM), within a BUS-EUS sequence is underscored unless Space (X'40') codes are explicitly excluded by the Bypass parameter of BUS.

Because the string of text to be underscored can contain characters to be presented in a proportionally spaced font, special consideration must be given to ensuring that information is not lost. It is recommended that underscoring begin at the horizontal position of the first character in the proportionally spaced text and end at the horizontal position of the last character plus its width-1, calculated in escapements.

End Underscore (EUS)

CSP,CLASS,COUNT,TYPE



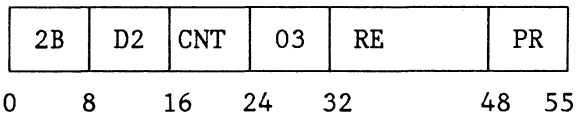
The EUS (MANDATORY) control is the ending control of the paired controls, Begin and End Underscore. The EUS control terminates the BUS control, thereby delimiting the end of text that is to be underscored. For information on the semantics of the BUS-EUS pair, please refer to "Begin Underscore" on page 17.

Support Note

An unpaired EUS is considered a Class 1 exception condition. If SEA allows continued processing, such an EUS control is treated as an NUL (X'00') control.

Justify Text Field (JTF)

CSP,CLASS,COUNT,TYPE,RIGHT-EDGE[,PERCENT-RULE]



The JTF (OPTIONAL-EAA) control specifies justification of a field of text and the horizontal position that the text is justified to. Line-ending decisions are not affected by JTF. The purpose of the control is to allow multicolumn justification by the insertion of variable width spaces between words. The control justifies the following field of text and can be used for multicolumn justification by specifying it once for each column in a line. JTF is also used to justify single-column output when there are line numbers at the right edge. A field of text, as used in the JTF control, is delimited by the JTF and the next subsequent JTF or line-ending code (CR, NL, RNL, FF, RFF). Tab controls (IT, HT) and LF are not field delimiters in JTF.

RE = Right Edge; a two-byte binary number from 0 to 32767 specifying in 1440ths of an inch the horizontal position that text is justified to. The horizontal position is the number of 1440ths of an inch between the desired position and the left edge of the presentation space. The right edge value is the last horizontal position that the text field can occupy (that is, the rightmost edge of the character box of the last character, or space if not fully justified text, in the text field).
Valid RE values are less than or equal to the right edge value for the presentation space.

0 = Terminate justification.
32678 to 65535 = Reserved.

PR = Percent Rule; a one-byte binary value from 0 to 100 specifying the percent of alignment.

0 = Use the current percent rule.
100 = Fully justify text.
67 = 2/3 justify text.
50 = 1/2 justify text.
101 to 255 = Reserved.

Support Note

The text field to be justified is delimited by the JTF and the next subsequent JTF, or line-ending control (CR(ZICR),NL,RNL,FF, or RFF). The placement of characters within the text field is determined as follows:

- The distance between the right edge of the last character box in the field and the right edge position is computed and multiplied by the specified per cent rule to determine the excess white space that must be distributed between words.

- Then starting with the first code point that is not X'40' in the field (first code point after the last IT, LF, or HT that is not X'40' if the field contains an IT, LF, or HT), the excess white space is distributed evenly among all the subsequent spaces (X'40's) in the field.

If the PR value specified is not supported, the exception alternate action for the Class 2 exception (if SEA allows continued processing) is to use the nearest supported value.

A JTF control that delimits a text field for a previous JTF control in the same line causes the horizontal print position to be set to the right edge value specified in the previous JTF plus 1 escapement.

The JTF function does not justify text containing backspaces; that is, the width of the backspace is not expandable for justification.

When a line is justified, any Spaces or Numeric Spaces at the right end of the line are ignored. Required Spaces at the right edge of a line are treated as graphics, causing the rightmost printable character to be unaligned.

Text to the left of an HT or LF is not justified, although the remainder of the line is justified.

Spaces that appear immediately to the right of the justification start point (line beginning, HT or LF) are not expanded.

If the Right Edge parameter is omitted, an exception is declared.

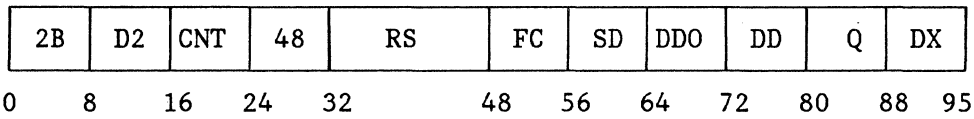
When justification of a line is requested by JTF, it is performed even if the line ends with a paragraph delimiter. This is different from SJM.

The SP control is the only control that is expanded during justification. NSP and RSP are not expanded.

A Class 2 exception condition exists if JTF or SJM control functions are active simultaneously. The exception alternate action, if SEA allows continued processing, is to process the JTF control as specified.

Page Presentation Media (PPM)

```
CSP,CLASS,COUNT,TYPE,RESERVED,RESERVED,FORMS CONTROL
                [[[[[,SOURCE DRAWER],DESTINATION-DRAWER-OFFSET]
                ,DESTINATION-DRAWER],QUALITY],DUPLEX]
```



The PPM (OPTIONAL-EAA) control specifies the presentation media device settings. These can only be changed on a page boundary. PPM specifies the forms setting, the source drawer, the destination drawer offset, the destination drawer, the print quality, and the duplex function.

RS = Reserved; a two-byte field reserved for future use. Must be set to X'0000'.

FC = Forms control; a one-byte binary number from 0 to 2 specifying the forms selection.

0 = Use the current form setting.
1 = Deactivate envelope selection and select paper.
2 = Deactivate paper selection and select envelopes.
3 to 255 = Reserved.

SD = Source Drawer; a one-byte binary number from 0 to 127 specifying the number of the source drawer that is used for paper or envelopes.

0 = Use the source drawer currently selected.
1 = Source Drawer 1.
2 to 127 = Source Drawer 2 to 127.
128 to 255 = Reserved.

Example: If SD = 1 and paper is selected, paper drawer 1 is used.
If SD = 1 and envelopes are selected, envelope drawer 1 is used.

DDO = Destination Drawer Offset; a one-byte binary number from 0 to 254 specifying an offset in the destination drawer. This can be used to alternate the alignment of stacks of print output in the destination drawer.

0 = Use the current offset value.
254 = Switch to alternate offset.
1 to 253 and 255 = Reserved.

DD = Destination Drawer: a one-byte binary number from 0 to 20 specifying the destination drawer to be used.

0 = Use the current destination drawer.
1 = Use destination drawer 1.
2 to 20 = Use destination drawer 2 to 20.
21 to 255 = Reserved.

Q = Quality; a one-byte binary number from 0 to 3 specifying the print quality to be used.

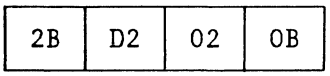
0 = Use the current print quality value.
1 = Use draft quality.
2 = Use standard print quality.
3 = Use intermediate quality.
4 to 255 = Reserved.

DX = Duplex; a one-byte binary number from 0 to 2 that activates printing on both sides of each page.

0 = Use the current duplex setting.
1 = Print on one side of each page.
2 = Print on both sides of each page.
3 to 255 = Reserved.

Release Left Margin (RLM)

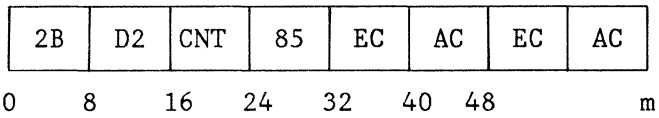
CSP,CLASS,COUNT,TYPE



The RLM (MANDATORY) control specifies that any backspace (BS, UBS) is permitted to move the output pointer to the left of the left margin, but never to the left of the left edge of the presentation space. CR(ZICR), NL, RNL, FF, and RFF always return to the left margin or the tab stop corresponding to the current indent level even if this causes the output pointer to move to the right. An RLM control is effective from the point of appearance to the next CR(ZICR), NL, RNL, FF, or RFF. When RLM follows an NL control (NL-RLM), it defines a paragraph boundary to the justification function.

Set Exception Action (SEA)

CSP,CLASS,COUNT,TYPE,EXCEPTION-CLASS,ACTION
[,..EXCEPTION-CLASS,ACTION]



The SEA (MANDATORY) control specifies an explicit action to be taken when an exception condition occurs. The exceptions can arise from an unsupported control, an unsupported control parameter, or unsupported parameter values. SEA is also used to control recovery procedures for exception conditions.

EC = Exception Class; a one-byte binary number from 0 to 4 that designates a group of exception conditions for which the specified action should be taken if the exception arises.

- 0 = Action should be taken for all exceptions in classes 1, 2, 3, and 4.
- 1 = Detection of a condition that may cause loss of text information.
- 2 = Condition detected that may alter the appearance of the information text.
- 3 = A multibyte control detected that contains an unsupported type or class code.
- 4 = A multibyte control detected that contains an unsupported parameter or parameter value.
- 5 to 255 = Reserved.

AC = Action; a one-byte binary number from 0 to 3 that specifies the action to be taken by a process when the designated exception is detected in the text stream.

- 0 = Accept; indicate or present a message to the receiver that identifies the exception condition and proceed with the process. Additionally, if EC = 3, processing resumes with the next byte in the text stream and if EC = 4 and the control description specifies a method for selecting an alternate parameter value, that value is used, else the current value is used and processing resumes.
- 1 = Ignore; no action is required and no exception indicator or message is required at the receiver.
- 2 = Cancel; the receiving process terminates with an indicator or message presented to the receiver that identifies the exception that caused process termination.

3 = Intervention and response required; present an indicator or message to the receiver that identifies the exception condition and requests an operator start or cancel response. An operator response of cancel causes the process to terminate. An operator response of start causes processing to resume. If EC = 3, processing resumes with the next byte in the text stream. If EC = 4 and the control description specifies a method for selecting an alternate parameter value, that value is used, else the the current value is used and processing resumes.

4 to 255 = Reserved.

Support Note

There are no defined default values for SEA. Instead, support of any of the Class and Action parameter values as defaults in the absence of an explicit SEA control in the data stream is allowed. It is recommended, however, that the default (initialization value) be Class = 0 (All) and Action = 2 (Cancel). If print fidelity is a requirement, then an SEA control with the appropriate parameter values must be present in the final-form-text document.

The Exception Class and Action parameters of SEA must be given in pairs. The SEA control can only be specified at line or page boundaries. One pair of parameters may be specified for each exception class for which an explicit action is to be taken. If an exception, action pair is repeated for the same Exception Class then the last parameter pair for that class is used. For example, if SEA 0 1 3 2 is encountered, the action code for exception class 3 is 2. If SEA 3 2 0 1 is encountered, the action code for exception class 3 (and all other exception classes) is 1. Subsequent SEA controls may be included in the text stream to change one or more of the specified exception actions.

Support for the following two Exception Class and Action parameter pairs is required:

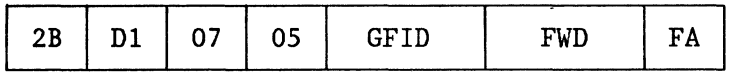
Class 0, Action 0
Class 0, Action 2

This mandatory support requirement provides two extremes of originator control over the final disposition of the document. The first allows processing to continue within the bounds of the architected exception alternate actions if an exception should occur. The second class and action pair provides a means of specifying that the document must be processed exactly as specified, with process termination if this requirement cannot be met. This provides the originator with the capability to totally control the presentation of the content and appearance of the document.

To reset all exception classes, either EC = 0 must be used, or each of the four exception classes must be individually specified.

Set FID thru GFID (SFG)

CSP,CLASS,COUNT,TYPE,GLOBAL-FONT-ID,FONT-WIDTH,
FONT-ATTRIBUTES



The SFG (MANDATORY) control specifies the font identification (for example, Elite or Courier), font width, and font attributes that are used for printing or displaying subsequent text. The Font ID (GFID) and Font Width (FWD) are independent parameters: the FWD parameter specifies the font width; the GFID indicates typestyle only. The font width and attributes are also used for exception recovery processing (evaluation for an exception alternate action) if the specified font (typestyle) is not supported.

GFID = Global Font ID; a two-byte binary number from 1 to 65534.

The GFIDs are assigned with reserved values in the range of X'00nn' to be assigned to currently designated IBM fonts and values in the range of X'FFnn' reserved for user-assigned fonts.

1 to 255 = Assigned to released fonts. See Appendix A for a list of assigned font IDs.
256 to 65279 = IBM GFID registered fonts.
65280 to 65534 = User-assigned fonts.
0 and 65535 = Reserved.

FWD = Font Width; a two-byte binary number from 1 to 1440 that specifies the designated width of the font in 1440ths of an inch.
0 and 1441 to 65535 = Reserved.

FA = Font Attribute; a one-byte binary value that identifies the characteristic attributes of the font.

1 = Font is monospaced.
2 = Font is proportionally spaced using Type 1 PSM character increments. See Appendix B for width table for Type 1 PSM.
0 and 3 to 255 = Reserved.

Support Note

When SFG controls are specified, they must include all three SFG parameters: GFID, FWD, and FA.

There are no defined default values for SFG. To ensure print-fidelity the SFG control must be specified in the final-form-text document.

The Font ID parameter (GFID) indicates typestyle and an implicit width; in Final-Form-Text Architecture, however, the implied width is ignored. The Font Width parameter (FWD) is used for determining font width. The two parameters are treated as independent variables; as a result, sometimes the specified Font Width parameter and the font width implied by the Font ID are inconsistent. In the Final-Form-Text Architecture, the font width specified by the FWD parameter is always honored, if supported. Therefore, there is no requirement to perform a consistency check or to raise an exception if such an inconsistency exists. By disassociating the font width from the Font ID, additional functional capability is provided. An exception is declared if the Font ID parameter is present and the Font Width parameter is not present. If processing continues, the previous font width is used.

If the specified font is not supported, and SEA allows continued processing, a font with the width and attributes specified in FWD and FA should be used if possible. If the font width specified is not supported, the next smaller width supported should be used, or if a smaller width is not supported, the next larger width should be used.

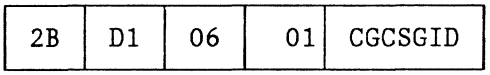
On a device that requires operator action to support the specified font, (for example, mounting the appropriate print wheel) there may be no way to guarantee that the requested font is used. In this case, from an architecture viewpoint, the device supports the font if it recognizes the Font ID and informs the operator of the requirement.

Attempts to go outside the presentation space could cause information loss and therefore cause a Class 1 exception to be raised.

See Appendix A for Font IDs and Appendix B for the width of PSM characters.

Set CGCSGID (SCG)

CSP,CLASS,COUNT,TYPE,CODED-GRAPHIC-CHARACTER-SET-GLOBAL-ID



The SCG (MANDATORY) control specifies the coded graphic character set global identification that is used to map subsequent text into printable graphics.

CGCSGID = Coded Graphic Character Set Global ID; a concatenation of 2 two-byte numbers. The first two bytes identify the Graphic Character Set Global ID (GCSGID) expressed as a binary value. The second two bytes identify the Code Page Global ID (CPGID) expressed as a binary value.

GCSGID = The Graphic Character Set Global IDs are assigned values in the range of 1 to 65534.

CPGID = The Code Page Global IDs are assigned values in the range of 1 to 65534.

Support Note

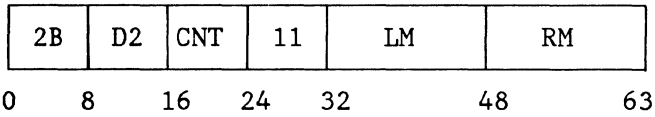
GCSGID and CPGID are used to determine how coded text characters are translated to the graphic characters to be presented.

Appendix B describes 2 CGCSGIDs. GCSGID 110 CPGID 256 and GCSGID 337 CPGID 256 are defined.

The Final-Form-Text Architecture requirement to support the mandatory default values (110-256) is considered satisfied if the product implements and properly processes its national character set on its national EBCDIC code page or on EBCDIC code page 256, 257, or 258 (GCSGID and CPGID support is described in applicable IBM product publications). When presenting graphic characters outside the default character set, if the exact graphic character specified cannot be presented by the receiver, an exception condition must be raised and processed according to the current SEA Action parameter for unsupported parameter values. If a specified SCG is not supported, a warning must be printed or displayed with the document and processing may continue using the default character set of the country of the product printing or displaying the document, if the SEA Action for unsupported parameter values is accept or ignore.

Set Horizontal Margins (SHM)

CSP,CLASS,COUNT,TYPE,LEFT-MARGIN[,RIGHT-MARGIN]



The SHM (MANDATORY) control specifies left and right margins. The SHM parameters are specified in 1440ths of an inch relative to the left edge of the presentation space. A parameter value of 1 for the LM parameter is equivalent to the left edge of the presentation space.

LM = Left Margin value; a two-byte binary number from 0 to 32767 specifying in 1440ths of an inch the leftmost presentation position relative to the left edge of the presentation space.

0 = Use the current left margin value.
2160 = 1 1/2 inch left margin.
32768 to 65535 = Reserved.

RM = Right Margin; a two-byte binary number from 0 to 32767 specifying in 1440ths of an inch the rightmost presentation position relative to the left edge of the presentation space (that is, the rightmost edge of the character box of the last character or space on the line).

0 = Use the current right margin value.
10800 = 7 1/2 inches from the left edge.
32768 to 65535 = Reserved.

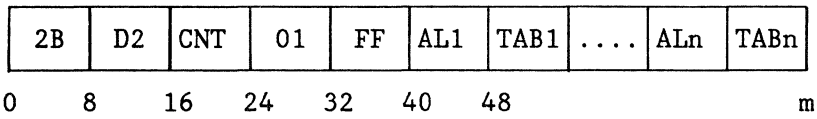
Support Note

Support of RM is not required unless SJM is also supported. In final-form-text documents, RM is only used for justification and is ignored when SJM is not active.

If the RM parameter is specified and SJM is active, a Class 2 exception should be raised when LM > RM.

Set Horizontal Tab Stops (STAB)

```
CSP,CLASS,COUNT,TYPE,FIXED-FLOAT[,ALIGNMENT,TAB-SETTING...
                                [,ALIGNMENT,TAB-SETTING]]
```



The STAB (MANDATORY) control specifies horizontal tab settings and the alignment characteristic associated with each tab setting. An alignment and tab setting parameter is required to define each tab setting. Tab stops specified in STAB always replace the current tab stop settings.

FF = Fixed or Floating; a one-byte binary number of 0 or 1 specifying whether the specified tab stops are defined in absolute units of 1440ths of an inch (fixed tabs) or are defined in character units (floating tabs). Final-Form-Text Architecture requires support of both absolute and floating tab stops.

0 = Tab Stops are defined in character (current font width) units (FLOATING).
1 = Tab Stops are defined in 1440ths of an inch (FIXED).
2 to 255 = Reserved.

ALn = Alignment; a one-byte binary number specifying the type of alignment to be performed by the formatter in positioning text at the designated tab stop. The only valid value in Final-Form-Text Architecture is zero.

0 = Null Align; normal tab formatting.
1-255 = Reserved.

TABn = Tab Setting; a two-byte binary number from 0 to 32767, specifying a horizontal tab (HT) position. When the FF parameter value is = 0, the tab stop locations are computed relative to the font width (as specified by SFG) and the left margin. When the FF parameter value is = 1, tab stop locations are in absolute units of 1440ths of an inch, with absolute unit 1 (1440th) at the left margin. The tab string may be in any order. When tab position values are not set to the right of the current presentation position, the horizontal tab function invokes an exception action. The minimum number of tab stops that must be supported in Final-Form-Text Architecture is fourteen.

TABn = 0 - Ignore this parameter and its associated alignment type.
TABn = 1 - Set a tab at the left margin.
32768 to 65535 = Reserved.

Support Note

There are no defined default tab stop settings.

No tab can be set to the left of the left margin.

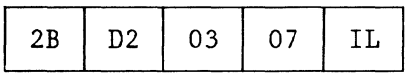
If the value of FF is out of range or if any of the alignment parameters (ALn) is non-zero, an exception is declared.

If the specified number of tab stop settings is not supported, the exception alternate action for the Class 1 exception (if SEA allows continued processing) is to ignore all tab settings with a tab position value that is greater than (that is, to the right of) the highest value in the set supported. When these unsupported tab stop settings are referenced by an HT control (and SEA allows continued processing), they are treated as a space.

If the tab-setting parameter value specified is not supported, the exception alternate action for the Class 1 exception (if SEA allows continued processing) is to use the nearest supported larger parameter value. If no larger parameter value is supported, the next smaller value supported should be used.

Set Indent Level (SIL)

CSP,CLASS,COUNT,TYPE,INDENT-LEVEL



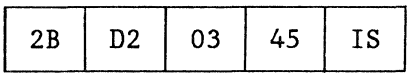
The SIL (MANDATORY) control specifies the indent tab level. This control allows the indent tab level to be set without specifying an Indent Tab, a Required Form Feed, or a Required New Line. The indent level set by SIL is processed just as if the indent level was set by Indent Tabs.

IL = Indent Level; a one-byte binary number from 0 to 255 specifying the indent tab level in number of tabs from the left margin.

0 = Reset indent level to the left margin.
1 to 255 = The number of tabs to the right of the left margin.

Set Initial Conditions (SIC)

CSP,CLASS,COUNT,TYPE,INITIALIZATION SET



The SIC (MANDATORY) control specifies a set of values used to initialize the final-form-text control values.

IS = Initialization Set; a one-byte binary number from 1 to 2 specifying a set of values that is used to initialize the final-form-text control values.

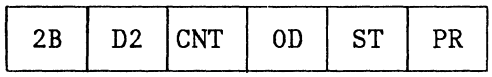
1 = Use the default set of final-form-text values. The final-form-text default values are specified in "Parameter Values and Default Values" on page 53.
2 = Use the SIC non-U.S. values. The SIC non-U.S. values are specified in "Parameter Values and Default Values" on page 53.
0 and 3 to 255 = Reserved.

Support Note

Support of IS = 2 (SIC non-U.S. values) is not required.

Set Justify Mode (SJM)

CSP,CLASS,COUNT,TYPE,STATE[,PERCENT-RULE]



The SJM (OPTIONAL-EAA) control specifies the justification of the following lines of text. Line-ending decisions are not affected by SJM. The purpose of the control is to allow the justification of documents by the insertion of variable width spaces between words. SJM is used to produce right-aligned text. Once an SJM is activated, it stays activated until the next SJM control. The text is justified to the horizontal location that has been set by the RM parameter of a previous SHM control.

The right margin value is the last horizontal position (that is, the rightmost edge of the character box of the last character, or space if not fully justified text, in the line) that the text line can occupy.

ST = State; a one-byte binary value of 0 or 1 specifying the activation and deactivation of justification.

0 = Deactivate justification.
1 = Activate justification.
2 to 255 = Reserved.

PR = Percent Rule; a one-byte binary value from 0 to 100 specifying the percent of alignment.

0 = Use the current percent rule.
100 = Fully justify text.
67 = 2/3 justify text.
50 = 1/2 justify text.
101 to 255 = Reserved.

Support Note

The placement of characters within a line is determined as follows:

- The distance between the right edge of the last character box in the line and the right margin position is computed and multiplied by the specified percent rule to determine the excess white space that must be distributed between words.
- Then starting with the first code point that is not X'40' in the line (first code point after the last IT, LF, or HT that is not X'40' if the line contains an IT, LF, or HT), the excess white space is distributed evenly among all the subsequent spaces (X'40's) in the line.

If the PR value specified is not supported, the exception alternate action for the Class 2 exception (if SEA allows continued processing) is to use the nearest supported value.

White space specified by leading tabs and spaces is not expanded.

The SP control is the only control that is expanded by justification. NSP and RSP are not expanded.

The SJM justification function does not justify text containing backspaces; that is, the width of the backspace is not expandable for justification.

When a line is justified, any Spaces or Numeric Spaces at the right end of the line are ignored. Required Spaces at the right edge of a line are treated as graphics, causing the rightmost printable character to be unaligned.

The recognition of the following paragraph boundaries is required to support SJM:

Two or more New Lines
Required New Line
New Line, Line Feed
New Line, Horizontal Tab
New Line, Indent Tab
New Line, Space
New Line, Numeric Space
New Line, Required Space
New Line, (any backspace)
New Line, Carriage Return(ZICR)
New Line, Release Left Margin
Required Form Feed

In order for two controls to jointly constitute a paragraph boundary, there must not be any intervening one-byte codes above X'40'. Intervening one-byte codes below X'40' and intervening multibyte controls (except Release Left Margin) are ignored when computing paragraph boundaries.

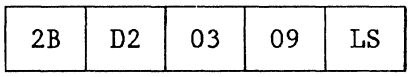
When SJM is active, Final-Form-Text Architecture does not require that the last line of a paragraph be justified. However, if the last line of a paragraph is to be justified, then the algorithm specified is: When SJM (rather than JTF) is in effect, the last line of a paragraph (a line ended by a paragraph delimiter, such as RNL or NL-NL) is not justified unless the distance between the last character on the line and the right margin is less than or equal to six times the font width and no space would be expanded to a width greater than twice the font width.

When it is necessary to justify the last line (not a paragraph boundary) of body text in a page also containing bottom margin text, the sequence of CR(ZICR), LF, LF... should be used.

A Class 2 exception condition exists if JTF and SJM are active simultaneously. The exception alternate action, if SEA allows continued processing, is to process the SJM control as specified.

Set Line Spacing (SLS)

CSP,CLASS,COUNT,TYPE,LINE-SPACING



The SLS (MANDATORY) control specifies multiple line spacing and the number of half lines to be moved for each NL, LF, and RNL.

LS = Line Spacing; a one-byte binary number from 1 to 8 specifying the number of half lines moved for an NL, LF or RNL.

0 = Reserved.
1,3 to 8 = Number of half lines per NL.
2 = 2 half lines per NL (single line spacing).
9 to 255 = Reserved.

Support Note

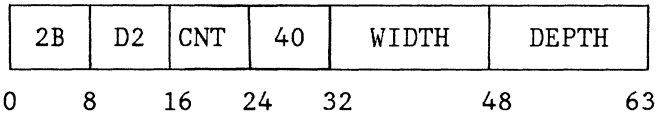
Support for single and double line spacing for printing final-form-text documents is mandatory.

The requirement to respecify line spacing such that it takes effect at the immediate next line is achieved using the following sequence: CR(ZICR), SSLD, SLS, NL.

If the LS value specified is not supported, the next smaller value supported should be used, or if a smaller value is not supported the next larger value supported should be used.

Set Presentation Page Size (SPPS)

CSP,CLASS,COUNT,TYPE,WIDTH[,DEPTH]



The SPPS (OPTIONAL-EAA) control specifies the width and depth of the page presentation media.

WIDTH = A two-byte binary number from 0 to 32767 specifying the horizontal extent of the page presentation media in terms of 1440ths of an inch.

0 = Use current page width value.
12240 = 8 1/2 inches.
32768 to 65535 = Reserved.

DEPTH = A two-byte binary number from 0 to 32767 specifying the vertical extent of the page presentation media in terms of 1440ths of an inch.

0 = Use current page depth value.
15840 = 11 inches.
32768 to 65535 = Reserved.

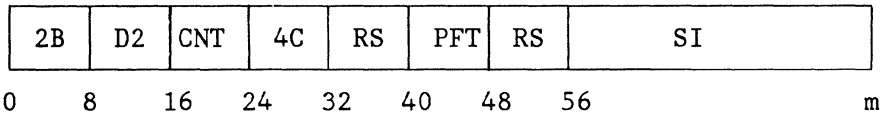
Support Note

In Final-Form-Text Architecture, the presentation page or physical page with WIDTH and DEPTH extents are equivalent to the presentation space. The two-dimensional area bounded by the specified width and depth is the surface upon which graphic characters are positioned, according to the positioning controls embedded in the final-form-text data stream (or default values, if unspecified). The presentation position is never allowed to go outside the bounds of the presentation space (page).

Nonzero values of SPPS can optionally be used to check for page overruns.

Set Print Setup (SPSU)

CSP,CLASS,COUNT,TYPE,RESERVED,PAPER-FEED-TECHNIQUE
[,RESERVED,SETUP-IDENTIFIER]



The SPSU (OPTIONAL-EAA) control specifies the paper feed technique that is to be used and the print setup that is to be used. The SPSU control does not affect the current PPM source drawer setting.

RS = Reserved; a one-byte field reserved for future use. Must be set to X'00'.

PFT = Paper Feed Technique; a one-byte binary number from 0 to 5 specifying the paper feed mechanism to be used.

0 = Use the current value for paper feed technique.
1 = Use cut sheet manual feed.
2 = Use continuous paper tractor feed.
3 = Use cut sheet automatic feed.
4 = Use continuous paper reversible tractor feed.
5 = Use continuous paper friction feed.
6 to 255 = Reserved.

RS = Reserved; a one-byte field reserved for future use. Must be set to X'00'.

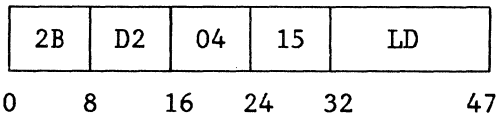
SI = Setup Identifier; a one to sixty character alphanumeric message that may identify the print form, source drawer, ribbon, or all three to be used for subsequent print output.

Support Note

A paper feed technique default is not specified if the cut sheet automatic feed feature is not supported.

Set Single Line Distance (SSLD)

CSP,CLASS,COUNT,TYPE,LINE-DISTANCE



The SSLD (MANDATORY) control defines the depth of one line. This control is used in combination with the SLS control to specify the distance moved for the vertical space controls NL, LF, and RNL.

LD = Line Distance; a two-byte binary number from 1 to 1440 specifying in 1440ths of an inch the depth of one line.

240 = 240/1440ths of an inch (1/6 of an inch).
0 and 1441 to 65535 = Reserved.

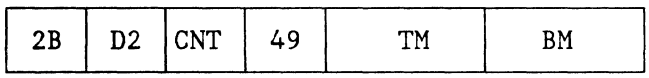
Support Note

If the LD value specified is not supported, the next smaller value supported should be used or, if a smaller value is not supported, the next larger value supported should be used.

The requirement to respecify line spacing such that it takes effect at the immediate next line is achieved using the following sequence: CR(ZICR), SSLD, SLS, NL.

Set Vertical Margins (SVM)

CSP,CLASS,COUNT,TYPE,TOP-MARGIN[,BOTTOM-MARGIN]



The SVM (MANDATORY) control specifies the top margin and the bottom margin. The Top Margin (TM) parameter defines the vertical distance between the top of the page and the baseline of the first print line on the page. The Bottom Margin (BM) parameter defines the vertical distance between the top of the page and the last print line on the page.

TM = Top Margin; a two-byte binary number from 0 to 32767 specifying in 1440ths of an inch the position of the first print line relative to the top of the page.

0 = Use the current top margin value.
1680 = 1 inch top margin if line density equals 6 lines per inch.
32768 to 65535 = Reserved.

BM = Bottom Margin; a two-byte binary number from 0 to 32767 specifying in 1440ths of an inch the position of the last print line relative to the top of the page.

0 = Use the current bottom margin value.
14400 = 10 inches from top of page if line density equals 6 lines per inch.
32768 to 65535 = Reserved.

Support Note

The first vertical position at the top of the page is represented by a value of 1 in the TM parameter.

Support of the Bottom Margin parameter is not required. In the final-form-text data stream, the Bottom Margin parameter value is ignored.

ONE-BYTE CONTROLS

The one-byte controls in Final-Form-Text Architecture are listed in Figure 2 on page 12 (Final-Form-Text summary) along with a specification of when they are mandatory and when they are optional with an exception alternate action. This section describes all the one-byte controls that are valid in a final-form-text data stream. The one-byte controls that are handled differently in the adjust and the non-adjust modes in revisable-form text are executed when they appear in a final-form-text data stream as if they were in non-adjust mode. Unrecognized one-byte code points (invalid, unassigned, and unsupported EBCDIC code point values) may cause information to be misinterpreted and thus cause a Class 1 exception to be raised. The following one-byte code points have been removed from the Final-Form-Text Architecture: IRT(X'33'), NBS(X'36'), SWT(X'2A'), and RPT(X'0A'). However, for the purposes of migration and compatibility, it is not a requirement that an exception condition be raised if an IRT is processed as a RNL, an NBS is processed as a BS, or an SWT or a RPT is processed as an NUL(X'00').

Backspace (BS)

EBCDIC Code X'16'

The BS (MANDATORY) control causes the output pointer to move left a distance that is fixed for any given font.

Support Note

The Backspace control is used for the following:

- Overstriking
- Moving text to the left of a tab or temporary left margin
- Centering (on many machines, using the center key causes backspaces to be inserted)
- Decimal tabbing (using the decimal tab key causes backspaces to be inserted)
- Underscoring (for parts of words or on machines without WUS)
- Representing revisable-form-text alignment functions (such as center align and right align) in final-form text.

When the Backspace control is encountered, the width of a backspace is subtracted from the current horizontal escapement count, so that the next character is placed to the left of where it would otherwise have gone. In fixed pitches, the distance is always one character width. In proportional spacing it is constant for a given font. A backspace cannot move the output pointer across the left margin unless the margin is released. However, attempting to do so does not cause an exception, but simply places the output pointer at the left margin. A backspace can never move the output pointer out of the presentation space, and an exception is declared if this is attempted. Backspace is a Word-Underscore delimiter.

The final-form-text justification function does not justify text containing backspaces; that is, the width of the backspace is not expandable for justification. This applies for both JTF and SJM.

See "Release Left Margin" on page 23.

Bell (BEL)

EBCDIC Code X'2F'

The BEL (OPTIONAL-EAA) control causes some attended products to stop and wait for operator intervention.

Support Note

BEL is also called Stop (STP).

In Final-Form-Text Architecture, BEL is ignored.

Carriage Return (CR) (ZICR)

EBCDIC Code X'0D'

The CR(ZICR) (MANDATORY) control causes the output pointer to move to the first printing position on the current printing line.

Support Note

CR is also called Zero Index Carrier Return (ZICR).

Carriage Return (ZICR) allows two or more lines to be printed on the same printing line.

NL-CR (CRE-ZICR) is a paragraph boundary.

CR(ZICR) is a line-end control.

CR(ZICR) resets the RLM condition.

CR(ZICR) has no effect on justification. Since these lines may overlay each other, it may seem that justification should be suppressed. However, this would make it impossible to justify ASCII documents, which use CR(ZICR)-LF instead of NL. Additionally, the only way to justify the last line of body text in a page when it is not the end of a paragraph and the page also includes bottom margin text is with a CR(ZICR), LF, LF...sequence. Therefore, ZICR does not suppress justification. When CR(ZICR) is used for the overstrike function, justification must be turned off; otherwise results are unpredictable.

Name: Several names are in use to describe this control code.

Carrier Return	- EBCDIC
Carriage Return	- ISO, ASCII
Zero Index Carrier Return	- Historical WP

Form Feed (FF)

EBCDIC Code X'0C'

The FF (MANDATORY) control causes a new page to be started.

Support Note

FF is also called Page End (PE).

FF is a line-end and a page-end control.

The Form Feed control causes a new page to be started when the next graphic character or one-byte control is encountered, using the parameter values in effect at that time. At the time that the new page is started, a Carriage Return (Zero Index Carrier Return) operation is performed using the left margin and indent level then in effect.

1. The Form Feed control causes an immediate eject but does not cause a form feed until the first text character is encountered (that is, the first code point that is not part of a X'2B' control).

2. This sequence allows any multibyte controls (for example margin changes) that come after this Form Feed control and before the next one-byte control or graphic character to affect the format of the new page.

FF (PE) implies CR (ZICR), it does not imply NL, and therefore no new paragraph boundary is created by inserting a FF, which means that justification is not suppressed on lines ending with NL-FF.

Graphic Character

A graphic character is represented by a visible symbol, or one of the following: space, numeric space, or required space.

Horizontal Tab (HT)

EBCDIC Code X'05'

The HT (MANDATORY) control causes the output pointer to move right to the next tab stop.

Support Note

The Horizontal Tab control is used to arrange text in columns, for indenting, and as a substitute for several consecutive spaces.

In Final-Form-Text Architecture, tab stops are set relative to the left margin.

The Horizontal Tab control sets the horizontal component of the output pointer to the first tab stop to the right of the output pointer, so that the next

character is placed there. If there is no tab stop, an exception exists. If SEA allows continued processing (SEA AC = 0), the Horizontal Tab control is executed as if it were a space. If the output pointer is left of the left margin when an HT encounters the condition where no tab stop settings are specified, a space is executed. Exception processing is the same regardless of where the HT is encountered. The Horizontal Tab control is a Word-Underscore delimiter and it prevents justification of the portion of the line before it.

Tab Overrun and Underrun: When the pitch of a piece of text is changed from fixed to proportional (or vice versa), tab controls in it may no longer go to the same tab stops. This occurs because the changed character widths mean that the output pointer is not at exactly the same horizontal position just before the tab control is processed. If the output pointer is slightly to the left of a tab stop in one case and slightly to the right in the other, then the two outputs will be very different. The same problem can occur between two proportional typestyles if the character widths are different.

The general problem cannot be solved, because with the new font there may not be room between two tab stops for the characters that previously fit there. It is possible, by keeping track of the input pitch, to fix the problem whenever the information will fit and to give a message when the information will not fit. However, this is a complicated calculation, and is predictable only if there is reliable information about the input pitch. Therefore, Final-Form-Text Architecture does not attempt to prevent or detect tab overrun or underrun. This means that the pitch of a document should not be changed between fixed and proportional after typing, unless the document is to be manually checked for tab overrun and underrun errors. It also means that if a proportional-spaced document is transmitted to a device with a fixed-pitch printer, it may not print properly, and a warning message should be issued.

Indent Tab (IT)

EBCDIC Code X'39'

The IT (MANDATORY) control performs the function of a Horizontal Tab control and, in addition, causes an automatic tab to be executed at the beginning of every line until cancelled by a RNL or RFF.

Support Note

The Indent Tab control is used when a paragraph or series of lines is to be indented to a tab stop.

If one or more Indent Tab controls have been executed since the last RNL or RFF, the same number of automatic tabs are done at the beginning of every line.

Indent Tabs are additive; that is, multiple Indent Tabs may be used for greater indentation.

Line Feed (LF)

EBCDIC Code X'25'

The LF (MANDATORY) control causes the output pointer to move to the current horizontal printing position on the next printing line.

Support Note

LF is also called Index (INX).

The Line Feed code terminates a line and begins the next line at the current horizontal position.

LF is not a line-end control.

The Line Feed control can be used in conjunction with a Carriage Return (Zero Index Carrier Return) code to perform a new line function and for special page formatting applications.

The Line Feed control is a Word-Underscore delimiter.

On a line containing Line Feed controls, the portion of the line to the left of the last Line Feed control is not justified.

Line Feed followed by a New Line is a paragraph boundary.

As a device control, Line Feed is used to end a line without ending recording on a magnetic card track to permit multiple lines on the same recording track.

New Line (NL)

EBCDIC Code X'15'

The NL (MANDATORY) control causes the output pointer to move to the first printing position (left margin modified by the current indent level) on the next printing line.

Support Note

NL is also called Carrier Return (CRE).

An NL does not deactivate indent tabs, as Required New Line does.

NL is a line-end control.

An NL is equivalent to a Carriage Return (ZICR) followed by a Line Feed (Index).

A NL is a paragraph boundary when followed by one of the following controls:

Backspace	Release Left Margin
Carriage Return(ZICR)	Required Form Feed
Horizontal Tab	Required New Line
Indent Tab	Required Space
Line Feed	Space
New Line	Unit Backspace
Numeric Space	

In Final-Form-Text Architecture, paragraph boundaries are only used to suppress justification of the last line of a paragraph.

Paragraph Boundaries with JTF: Justification requested by the JTF control is not affected by paragraph boundaries. See notes in"Justify Text Field" on page 20.

Name: Several names are in use to describe this control code.

New Line	- EBCDIC
Carrier Return	- Historical WP
Line Feed	- ASCII(in some cases)

As a device control, New Line terminates a magnetic card track.

Null (NUL)

EBCDIC Code X'00'

The NUL (MANDATORY) control causes nothing to be printed and no action to be taken.

Support Note

Null is used to fill space where something has been deleted or to pad records to a certain length.

In Final-Form-Text Architecture, NUL is ignored.

Numeric Space (NSP)

EBCDIC Code X'E1'

NSP (OPTIONAL-EAA) is a nonprinting graphic character that is a word delimiter and has the same width as a digit.

Support Note

Numeric Space is useful in proportional spacing for aligning columns of numbers and whenever it is important that the size of a space be exactly the size of a digit.

The Numeric Space function is the same as the function of a graphic character, except that NSP is a Word-Underscore delimiter.

The width of an NSP is not expanded during justification.

Required New Line (RNL)

EBCDIC Code X'06'

The RNL (MANDATORY) control causes the output pointer to move to the first printing position on the next printing line.

Support Note

RNL is also called Required Carrier Return (RCR).

RNL is a paragraph boundary control.

RNL is a line-end control.

RNL resets the indent level to 0 (deactivates any indent tabs).

As a device control, Required New Line terminates a magnetic card track boundary.

Required Hyphen (HYP)

EBCDIC Code X'60'

The HYP (MANDATORY) control always prints as a hyphen.

Support Note

HYP is used in revisable-form text to create a hyphen that is always printed. Syllable Hyphen is not suitable, because it is printed only if it falls at the end of a line.

Required Hyphen functions exactly like and is treated as a graphic character.

Required Form Feed (RFF)

EBCDIC Code X'3A'

The RFF (MANDATORY) control causes a mandatory form feed, terminating the line and paragraph, and resetting the indent level.

Support Note

RFF is also called Required Page End (RPE).

RFF is identical to Form Feed except that RFF resets the indent level to 0 and is a paragraph boundary.

RFF is a line-end and a page-end control.

RFF (RPE) implies CR (ZICR), RFF is like FF in this respect.

Required Space (RSP)

EBCDIC Code X'41'

RSP (MANDATORY) is a nonprinting graphic character which differs from space in that it is not a word delimiter for word underscoring.

Support Note

Required Space is used to prevent a line ending from being inserted (for example before the C in J. Carter) and for underscoring several words with one word underscore (which differs from using several word underscores in that the blank space is underscored, too).

Required Space functions exactly like and is treated as a graphic character.

The width of RSP is not expanded during justification.

Space (SP)

EBCDIC Code X'40'

SP (MANDATORY) is a nonprinting graphic character that is a word delimiter.

Support Note

The Space function is used to separate words.

Space functions like any other graphic character except it is a Word-Underscore delimiter and its width is expanded during justification.

Subscript (SBS)

EBCDIC Code X'38'

The SBS (OPTIONAL-EAA) control causes the output pointer to move vertically down a fraction (<1) of the single line increment for the active line density with no horizontal movement of the output pointer.

Support Note

Subscript is used in writing mathematical and chemical formulas, in simulating fractional line spacing such as .5, 1.5, or 2.5, and in constructing graphical representations and figures.

Subscript movement is an implementation-defined fraction x, where $0 < x < 1$ (rather than requiring 1/2 line movement). The fraction may vary from one Subscript to the next but the distance moved must be equivalent for the reversing Superscript. If multiple Subscripts are not supported, a Class 1 exception must be raised for the extra Subscript codes. If SEA allows continued processing (SEA AC=0), then the exception alternate action is to ignore the extra Subscript codes. Subscript codes are cancelled by matching Superscript codes. If multiple SBS and or SPS are not supported, Final-Form-Text Architecture recommends that a count of the multiple Subscripts be maintained, in order to determine correct vertical positioning when matching Superscripts are present to cancel the multiple Subscripts.

Successive Superscripts and Subscripts: The number of successive Superscripts or Subscripts that are supported varies. Limiting support to a single Superscript or Subscript is necessary for certain printers due to printer physical limitations. However, the architecture does not limit the use of Superscript or Subscript. Complex expressions involving multiple SBS or SPS should be constructed by the document creator (formatter or originator) to produce a final-form-text data stream; that is, as several discrete print lines, with appropriate double spacing, for example. Although overstrike problems may result, an operator who knows how to use multiple Superscripts and Subscripts has greater capability.

Hanging Index at Line Boundary: When an SBS was not cancelled by an SPS and a line-end code is encountered, a hanging index occurs.

In Final-Form-Text Architecture, line-ending codes do not cause a return to the baseline; that is, SBS codes will not be cancelled automatically at line end and a new baseline is established.

Hanging Index at Page Boundary: When an SBS is not cancelled by an SPS and a page-end code is encountered, a hanging index occurs. As with a line-ending code, terminating a Superscript or Subscript with page-ending codes places certain practical limitations on where page-end decisions can be made. However, due to the noncontiguous effect caused by paper boundaries, retaining the half index may not be noticed.

In Final-Form-Text Architecture, Form Feed cancels any outstanding Subscript or Superscript positioning.

Definition of Base Line: If multiple Subscripts are not supported, it is important to have a uniform definition of the base line.

When an NL, LF, or RNL is executed, the new base line is X lines below the current vertical position of the output pointer, where X is the line spacing currently in effect. This means that if the output pointer is off the old base line when the line-ending control is executed, the new base line will not be X lines below the old base line.

The disadvantages of this approach (compared with having the new base line always X lines below the old base line) are:

- It is not intuitive.
- It is necessary to look ahead a full line during processing, because the superscript part of the N+1st line could overlay the base line part of the Nth line, requiring reverse movement by the paper-handling mechanism.

The advantages of this approach are:

- It is compatible with existing products.
- It permits the use of a single Subscript in conjunction with a RNL to make a single 1.5 spaced line without setting line spacing twice.
- When a mistake is made, the error message occurs near the place where the error occurred. With the other approach, an error could easily cause a message several pages later, because all the text between the two places would be off the base line but the operator would not realize it.

Size of Superscript and Subscript: Historically, word processing products have implemented a vertical escapement of one-half index for SBS and SPS.

In Final-Form-Text Architecture, the vertical escapement for SBS and SPS is a fraction (<1) of the depth of a single line, where the fraction is implementation defined.

Substitute (SUB)

EBCDIC Code X'3F'

The SUB (MANDATORY) control is inserted in the text when invalid graphic code points are encountered.

Support Note

SUB identifies the place in the text where invalid data appeared.

SUB is assigned a print representation that is implementation dependent. The graphic (Underscore or Required Hyphen is widely used) selected should be unique.

Superscript (SPS)

EBCDIC Code X'09'

The SPS (OPTIONAL-EAA) control causes the output pointer to move vertically a fraction (<1) of the single line increment for the active line density with no horizontal movement of the output pointer.

Support Note

Superscript is used in writing mathematical and chemical formulas, in simulating fractional line spacing such as .5, 1.5, or 2.5, and in constructing graphical representations and figures.

Superscript movement is an implementation-defined fraction x, where $0 < x < 1$ (rather than requiring 1/2 line movement). The fraction may vary from one Superscript to the next (for example, x to the power (a to the power 2)) but the distance moved must be equivalent for the reversing Subscript. If multiple Superscripts are not supported, a Class 1 exception must be raised for the extra Superscript codes. If SEA allows continued processing (SEA AC=0), then the exception alternate action is to ignore the extra Superscript codes. Superscript codes are cancelled by matching Subscript codes. If multiple SBS or SPS are not supported, Final-Form-Text Architecture recommends that a count of the multiple Superscripts be maintained, in order to determine correct vertical positioning when matching Subscripts are present to cancel multiple Superscripts.

The additional support notes in "Subscript" on page 47 are applicable.

Syllable Hyphen (SHY)

EBCDIC Code X'CA'

The SHY (MANDATORY) control always prints as a hyphen.

Support Note

SHY is used to indicate the end of a syllable, so that adjust algorithms can create a line end at that point.

In Final-Form-Text Architecture, Syllable Hyphen functions like a graphic character.

Printed even at end of paragraph: Printing of a Syllable Hyphen may be suppressed when it is the last character of a paragraph, on the theory that it makes no sense in that position and therefore should not be printed. Final-Form-Text Architecture requires printing a Syllable Hyphen at the end of a paragraph, to call attention to the error condition that exists. This is desirable because it is unlikely that suppressing the Syllable Hyphen actually gives the desired result. A Syllable Hyphen is treated exactly like a Required Hyphen.

Underscore

EBCDIC Code X'6D'

Underscore is a graphic character that is used for underscoring.

Support Note

Underscore characters with backspace characters are used to underscore parts of words and for underscoring full words or phrases on devices that do not support Word Underscore.

Underscore is a Word Underscore delimiter.

Unit Backspace (UBS)

EBCDIC Code X'1A'

The UBS (OPTIONAL-EAA) control causes the output pointer to move left one escapement, as defined for the printer being used, when in proportional spacing mode.

Support Note

The purpose of UBS is to provide character alignment in proportional-spaced printing.

In proportional spacing, the definition of UBS is the same as for Backspace, except that the space moved is one escapement.

In fixed pitches, UBS does not move the output pointer.

UBS is a word-ending code for Word Underscore.

See notes in "Backspace" on page 40 and "Release Left Margin" on page 23.

Word Underscore (WUS)

EBCDIC Code X'23'

The WUS (MANDATORY) control underscores everything between itself and the preceding WUS delimiter.

Support Note

Word Underscore is used for underscoring. A WUS may follow each word to be underscored, or required spaces may separate the words with one WUS at the end. The latter method underscores the spaces as well as the words and prevents line endings from being inserted in the phrase during adjusting. It is often used for titles or headings.

The controls that delimit a WUS are as follows: BS, NL, HT, LF, IT, JTF, NSP, FF, RNL, RFF, SP, UBS, WUS, and CR (ZICR). A WUS is also delimited by the underscore character.

SEQUENCE OF CONTROLS AND EXECUTION

Both one-byte and multibyte controls can occur throughout the final-form-text data stream. However, some controls can occur only on a page or line boundary.

Controls Restricted to Page Boundaries

When specified, the PPM, SIC, SPPS, SPSU, and SVM controls must always occur before the start of text in a document or immediately after a Form Feed control or other multibyte control(s) that immediately follow(s) a Form Feed control. If any of these page boundary controls occurs elsewhere, a Class 2 exception is raised. If SEA allows continued processing, it is recommended that the control be processed as specified.

Controls Restricted to Line Boundaries

When specified, the SEA, SHM, STAB, SIL, SJM, SLS, and SSLD controls must always occur before the start of text in a document or immediately after a CR(ZICR), NL, RNL, FF, RFF, or other multibyte control(s) that immediately follow(s) a CR(ZICR), NL, RNL, FF, or RFF control. If any of these line boundary controls occurs midline, a Class 2 exception is raised. If SEA allows continued processing, it is recommended that the control be processed as specified.

Controls with No Occurrence Restrictions

The BOS, EOS, BUS, EUS, JTF, RLM, SFG, SCG, and one-byte controls can occur anywhere in a final-form-text data stream.

Initialization Requirements and Execution Considerations

The Final-Form-Text Architecture requires initialization of control values to the default values specified in the control description (and summarized in "Parameter Values and Default Values") before processing a final-form-text document. It is not a requirement that the environment be physically re-established; this requirement will be considered satisfied if the environment is logically re-established to the final-form-text default values before processing a final-form-text document. This is equivalent to Initialization Set 1 of the SIC control.

Subsequent final-form-text controls are executed immediately and no multibyte controls cause an automatic form feed, line feed, or new line. All final-form-text multibyte control values stay in effect until either the end of the document is reached or the same multibyte control, paired ending control for either BUS or BOS, or an SIC control resets the control value. The SFG control does not affect the setting of any previous STAB control but is used in resolving subsequent HT positions, if floating tabs were specified in the active STAB.

SCG must occur before the start of text in a final-form-text document if the text was created using a GCSGID other that 110 or a CPGID other than 256.

An automatic form feed is executed at the start of text in a final-form-text document (that is, the first code point that is not part of a X'2B' control).

A Form Feed control causes an immediate eject but does not cause a form feed until the first text character is encountered (that is, the first code point that is not part of a X'2B' control).

An automatic eject without a form feed is executed at the end of a final-form-text document.

An NL, RNL, or LF causes an immediate line feed but a NL, RNL, or CR (ZICR) does not cause a return to the left margin or indent level until the start of subsequent text (that is, the first code point that is not part of a X'2B' control).

PARAMETER VALUES AND DEFAULT VALUES

A summary of the final-form-text parameter values is listed below. For further detail on the meaning of the headings used in the table, please refer to "Final-Form-Text Term Definitions" on page 7.

FINAL-FORM-TEXT PARAMETER VALUES

Control	Parameter	Optional Values	Mandatory Values	Default Values	SIC Non-US Values	Exception Alternate Action
BOS	CHAR	X'40'-'FE'	-	-	-	-
	BYP	X'00-01',X'80'	X'01'	X'01'	X'01'	Note 5
	GCSGID	0-65534	0 and 110	0	0	Note 5
	CPGID	0-65534	0 and 256	0	0	Note 5
EOS	-	-	-	-	-	-
BUS	MODE	1	1	-	-	-
	BYP	X'00-01',X'80'	X'01'	X'01'	X'01'	Note 5
EUS	-	-	-	-	-	-
JTF	RE	0-32767	-	0	0	Note 4
	PR	0-100	-	100	100	Note 2
PPM	RS	0	-	0	0	-
	FC	0-2	-	1	1	Note 1
	SD	0-127	-	1	1	Note 1
	DDO	0 and 254	-	0	0	Note 1
	DD	0-20	-	1	1	Note 1
	Q	0-3	-	2	2	Note 1
	DX	0-2	-	1	1	Note 1
RLM	-	-	-	-	-	-
SEA	EC	0-4	0	-	-	Note 1
	AC	0-3	0 and 2	-	-	Note 1
SFG	GFID	1-65534	-	-	-	Note 5
	FWD	1-1440	120	-	-	Note 5
	FA	1-2	1	-	-	Note 5
SCG	GCSGID	1-65534	110	110	-	Note 5
	CPGID	1-65534	256	256	-	Note 5
SHM	LM	0-32767	0-Device Limit	2160	-	-
	RM	0-32767	-	10800	-	Note 2

Figure 3 (Part 1 of 2). Control Parameter Values

Control	Parameter	Optional Values	Mandatory Values	Default Values	SIC Non-US Values	Exception Alternate Action
STAB	FF	0-1	0-1	-	-	-
	ALn	0	0	-	-	-
	TABn	0-32767	0-Device Limit	-	-	Note 4
SIL	IL	0-255	0-14	0	0	Note 2
SIC	IS	1-2	1	1	1	IS=1
SJM	STATE	0-1	-	0	0	Note 1
	PER	0-100	-	100	100	Note 2
SLS	LS	1-8	2 and 4	2	2	Note 3
SPPS	Width	0-32767	-	12240	-	Note 4
	Depth	0-32767	-	15840	-	Note 4
SPSU	RS	0	-	0	0	-
	PFT	0-5	-	3	3	Note 1
	RS	0	-	0	0	-
	SI	1 to 60 characters	-	-	-	Note 1
SSLD	LD	1-1440	-	240	240	Note 3
SVM	TM	0-32767	0-Device Limit	1680	1680	-
	BM	0-32767	-	14400	14400	Note 4

Notes on Exception Alternate Action for unsupported optional parameter values:

Note 1: Use current parameter value.
Note 2: Use nearest supported parameter value.
Note 3: Use nearest supported smaller parameter value. If no smaller values are supported use next larger supported value.
Note 4: Use nearest supported larger parameter value. If no larger values are supported use next smaller supported value.
Note 5: The exception alternate action, if the specified value is not supported, is described in the support notes for the control.

Figure 3 (Part 2 of 2). Control Parameter Values

EXCEPTION HANDLING

The exception handling specifications are described below.

Exception Classes

Exceptions for the one-byte and multibyte controls are process detected as deviations from the syntax and semantics as described in each of the controls in this architecture.

Syntax Exceptions

The syntax exceptions are detected by examination of the data stream when the process to print or display lines is executed. Syntax exceptions for multibyte controls can be detected when the Control-Sequence-Prefix (CSP,X'2B'), Class and Type fields are valid. Detected invalid or unrecognized multibyte class and type codes are handled as Class 3 exceptions. Multibyte syntax exceptions include incorrect length values that may be smaller or larger than necessary to specify the required control parameters. Because the parameters are specified in positional order, syntactic evaluation of the parameters can only determine that the length field is sufficient to contain the required parameters and that their values are within the range specified for the control. Processes that have the capability to evaluate the parameter values should raise a Class 4 exception condition when a control parameter is detected as outside the executable range of values.

When an exception condition is detected and processed and the SEA Class and Action pair indicates processing is to continue, the data stream examination is resumed at the next byte following the last byte that is contained in the control as specified by the multibyte control count field. However, an attempt should be made to process all of the parameters in the control before resuming the data stream examination.

Semantic Exceptions

The semantic exceptions are detected when the controls are executed by the print or display device. Although a control is syntactically valid, when it is executed by the print or display process, an exception condition can occur because the control is incompatible with the environment in which the exception is detected. For example, one control may establish the limits of a presentation space and another control may specify the positioning of the text. While both may be syntactically correct, the control that specifies the positioning of the text could cause attempted printing outside the limits of the presentation space; this would be semantically incorrect and should raise a Class 1 exception condition at the time it is executed by the print or display process. Class 1 and 2 exceptions are semantic exceptions. Unrecognized one-byte code points (invalid or unassigned EBCDIC code point values) may cause information to be misinterpreted and are therefore Class 1 exceptions.

The following table summarizes for each control the exception conditions that can occur and the class of each condition.

Exception Conditions	Exception Classes			
	1	2	3	4
Any unrecognized X'2B' control	x	-	x	-
Any presentation space (page) overrun condition	x	-	-	-
Any unrecognized one-byte code	x	-	-	-

Unsupported One-Byte Controls	Exception Classes			
	1	2	3	4
Backspace	-	-	-	-
Bell	-	-	-	-
Carriage Return (ZICR)	-	-	-	-
Form Feed	-	-	-	-
Horizontal Tab	-	-	-	-
Indent Tab	-	-	-	-
Line Feed	-	-	-	-
New Line	-	-	-	-
Null	-	-	-	-
Numeric Space	-	x	-	-
Required Form Feed	-	-	-	-
Required Hyphen	-	-	-	-
Required New Line	-	-	-	-
Require Space	-	-	-	-
Space	-	-	-	-
Subscript	x	-	-	-
Substitute	-	-	-	-
Superscript	x	-	-	-
Syllable Hyphen	-	-	-	-
Unit Backspace	-	x	-	-
Word Underscore	-	-	-	-

"x" = exception can occur; "-" = not an exception case.

Figure 4 (Part 1 of 3). Exception Conditions and Classes

Unsupported Multibyte Controls, Parameters, or Values	Exception Classes			
	1	2	3	4
Begin Overstrike	x	-	-	-
CHAR	-	-	-	-
BYP	-	x	-	x
CGCSGID	x	-	-	x
End Overstrike	x	-	-	-
Begin Underscore	x	-	-	-
MODE	-	x	-	x
BYP	-	x	-	x
End Underscore	x	-	-	-
Justify Text Field	-	x	x	x
Page Presentation Media	x	-	x	-
SC,DDO,DD,Q,DX	-	-	-	x
FC,SD	x	-	-	x
Set Exception Action	-	-	x	x
Set FID thru GFID	-	-	-	-
GFID	-	x	-	x
FWD,FA	x	-	-	x
Set CGCSGID	x	-	-	x
Set Horizontal Margins	-	-	-	-
LM	x	-	-	x
RM	-	x	-	x
Set Horizontal Tab Stops	x	-	-	x
Set Indent Level	x	-	-	x
Set Initial Conditions	x	-	-	x
Set Justify Mode	-	x	x	x
Set Line Spacing	-	-	-	-
LS(if Ex Alt Act =smaller value)	-	x	-	x
LS(if Ex Alt Act =larger value)	x	-	-	x
Set Presentation Page Size	x	-	x	x
Set Print Setup	x	-	x	-
PFT	-	-	-	x
SI	x	-	-	x
Set Single Line Distance	-	-	-	-
LD(if EAA = smaller value)	-	x	-	x
LD(if EAA = larger value)	x	-	-	x
Set Vertical Margins	-	-	-	-
TM	x	-	-	x
BM	-	-	-	-

"x" = exception can occur; "-" = not an exception case.

Figure 4 (Part 2 of 3). Exception Conditions and Classes

NOTES:

1. Class 1 exceptions are conditions that may cause the loss of text information. Detection of a condition that may cause the resulting presentation position in the inline direction or the baseline direction to exceed the physical sheet or display width or depth is a Class 1 exception. Unsupported controls that may cause information to be misinterpreted are Class 1 exceptions. Superscript, for example, when used as an exponent in an arithmetic expression, if not supported, results in a loss of the explicit meaning of the expression.

2. Class 2 exceptions are conditions that may cause a change in the intended appearance of presented information but do not affect the interpretation of text or cause presentation space (page) overruns.

3. Class 3 exceptions are conditions that result from detecting a specific control class and type code that is not supported. The results of this class of exception can be both loss of information and alteration of the intended format of the text. Class 3 exceptions also result when a control class or type code is detected that is unrecognized and could, therefore, cause a loss of information.

4. Class 4 exceptions are conditions that arise as a result of attempting to use a control parameter value that is not recognized or is not supported. These Class 4 exceptions can cause a loss of information and may alter the intended appearance of formatted text.

Figure 4 (Part 3 of 3). Exception Conditions and Classes

Exception Action

The handling of exception conditions is described in the SEA control. If any optional Action parameters of the SEA control (AC=1 or AC=3) are not supported and the Action parameter in effect for unsupported parameter values is AC=0(accept), then the current Action parameter value for the class(es) specified in the SEA control just encountered remains unchanged, processing resumes, and a message or indicator is presented to the receiver.

When a message or indicator is presented to the receiver, the message or indicator must be printed or displayed with the document and should provide sufficient information to identify the problem and to determine where the exception occurred.

DOCUMENT PORTABILITY

Use of certain optional final-form-text controls, parameters, and parameter values in an interchange environment may affect data integrity or document appearance when these are not supported by a receiving process. The conditions that may cause such occurrences are described below.

Information Integrity Considerations

Warning: The conditions that may cause information loss or affect the information integrity of a final-form-text document when they are not supported by a receiver are:

- **PSM fonts**
- **Paper sizes greater than 8 1/2 inches in width and 11 inches in length**
- **Line spacing more dense than 6 lines per inch**
- **Character set**
- **Envelope feature**
- **Superscript and subscript**
- **Fonts specified by the final-form-text date stream that are smaller in width than the receiver supports.**

Document Appearance Considerations

The conditions that do not affect information integrity but will affect final-form-text document appearance when they are not supported by a receiver are:

- Line spacing other than single or double
- Multicolumn justification
- Mid-page font changes for emphasis
- Use of Unit Backspace for justification or alignment if the receiver does not support the control values and the same font size (including the same PSM character sizes)
- Justify Text Field or Set Justify Mode, if the receiver does not support the control.

FINAL-FORM-TEXT HARDWARE FEATURE REQUIREMENTS

In addition to requiring a common set of device controls, the final-form-text interface also requires support of a common set of hardware features.

- Coded Graphic Character Set Global ID 110-256 (GCSGID 110 and CPGID 256)
- Line density of 6 lines per inch
- Paper size of 8 1/2 inches in width and 11 inches in length
- At least 60 characters per line.

APPENDIX A. FONT SUMMARY

IDENTIFIERS FOR SFG PARAMETERS

A listing of fonts can be found in product publications. The following table lists the fonts that have been assigned font identifiers and the corresponding SFG parameter values.

Font style	GFID	FWD	FA
Advocate	01	144	1
Delegate	02	144	1
OCR-B	03	144	1
Polygo Pica	04	144	1
Orator(alias Rhetoric)	05	144	1
Light Italic 10	06	144	1
OCR-M	07	144	1
Scribe 10	08	144	1
Large Pica	09	144	1
Cyrillic 22	10	144	1
Courier 10 (alias Courier 72)	11	144	1
Prestige Pica	12	144	1
Artisan 10	13	144	1
Manifold (alias Artisan All Caps)	14	144	1
Bookface Academic	15	144	1
Latin 10 High Speed	16	144	1
1403 OCR	17	144	1
Courier Italic 10	18	144	1
OCR-A	19	144	1
PICA	20	144	1
Katakana Light	21	144	1
Printing & Publishing, 12 Number 3	22	144	1
Light Italic 10 Mod	23	144	1
OCR-CD	24	144	1
Presentor	25	144	1
Reserved	26-79		

Font style	GFID	FWD	FA
Scribe	80	120	1
Artisan 12(alias Artisan 72)	81	120	1
Auto Elite	82	120	1
Elite	83	120	1
Script	84	120	1
Courier 12	85	120	1
Prestige Elite	86	120	1
Letter Gothic	87	120	1
High Speed Latin 12	88	120	1
Large Elite	89	120	1
Dual Gothic	90	120	1
Light Italic 12	91	120	1
Courier 12 Italic	92	120	1
Polygo Elite	93	120	1
Diplomat	94	120	1
Adjutant	95	120	1
Olde World	96	120	1
Light Italic 12 Mod	97	120	1
Reserved	98-154		
Boldface Italic	155	120	2
Thesis	156	120	2
Title	157	120	2
Modern	158	120	2
Boldface	159	120	2
Essay	160	120	2
Arcadia	161	120	2
Essay Italic	162	120	2
Essay Bold	163	120	2
Reserved	164		
High Speed Latin PSM	165	120	2
Reserved	166-220		
Prestige 15	221	96	1
Gothic 15	222	96	1
Courier 15	223	96	1
Rotated Data 1 15	224	96	1
Scribe 15	225	96	1
Reserved	226-255		

APPENDIX B. PSM CHARACTER WIDTHS

The following table lists the character width of each PSM character when the SFG font attribute parameter value equals 2. The character widths are expressed in units of 1/60 of an inch. The characters listed with their corresponding hexadecimal values represent the graphic character set known as GCSGID 337 and CPGID 256. The characters indicated by an asterisk (*) adjacent to the hexadecimal value represent the graphic character set known as GCSGID 110 and CPGID 256.

HEX	UNITS	NAME	HEX	UNITS	NAME
16	5	Backspace			
1A	1	Unit Backspace			
36	5	Numeric Backspace			
40*	5	Space	60*	5	Minus Sign, Hyphen
41*	5	Required Space	61*	5	Slash
42	5	a circumflex	62	7	A circumflex
43	5	a diaeresis	63	7	A diaeresis
44	5	a grave	64	7	A grave
45	5	a acute	65	7	A acute
46	5	a tilde	66	7	A tilde
47	5	a angstrom	67	7	A angstrom
48	5	c cedilla	68	7	C cedilla
49	6	n tilde	69	7	N tilde
4A	5	Open Sq.Bracket	6A	5	Vertical Broken Line
4B*	5	Period	6B*	5	Comma
4C	5	Less Than Sign	6C*	5	Percent Sign
4D*	5	Left Parenthesis	6D*	5	Underscore
4E*	5	Plus Sign	6E	5	Greater Than Sign
4F	5	Exclamation Point	6F*	5	Question Mark
50	6	Ampersand	70	5	0 slash
51	5	e acute	71	6	E acute
52	5	e circumflex	72	6	E circumflex
53	5	e diaeresis	73	6	E diaeresis
54	5	e grave	74	6	E grave
55	3	i acute	75	4	I acute
56	3	i circumflex	76	4	I circumflex
57	3	i diaeresis	77	4	I diseresis
58	3	i grave	78	4	I grave
59	6	German Sharp S	79	5	Grave
5A	5	Close Sq.Bracket	7A*	5	Colon
5B	5	Dollar Sign	7B	5	Number Sign
5C	5	Asterisk	7C	5	At Sign
5D*	5	Right Parenthesis	7D	3	Apostrophe
5E*	5	Semicolon	7E*	5	Equal Sign
5F	5	Circumflex	7F	5	Quotation Mark

HEX	UNITS	NAME	HEX	UNITS	NAME
80	7	0 Slash	A0	6	Micro
81*	5	a	A1	5	Tilde
82*	6	b	A2*	5	s
83*	5	c	A3*	4	t
84*	6	d	A4*	6	u
85*	5	e	A5*	6	v
86*	4	f	A6*	7	w
87*	6	g	A7*	6	x
88*	6	h	A8*	6	y
89*	3	i	A9*	5	z
8A	7	European Open Quote	AA	5	Spanish Open Exclam.Point
8B	7	European Close Quote	AB	5	Spanish Open Question Mark
8C	6	d stroke	AC	7	D stroke
8D	6	y acute	AD	7	Y acute
8E	6	Small letter thorn	AE	7	Capital Thorn
8F	5	Plus Minus Sign	AF	5	Circle R
90	5	Degree, Angstrom	B0	5	Cent Sign
91*	3	j	B1	5	Pound Sign, Lira
92*	6	k	B2	5	Yen
93*	3	l	B3	7	Peseta
94*	7	m	B4	5	Florin, Gilder
95*	6	n	B5	5	Section Sign
96*	5	o	B6	5	Paragraph Sign
97*	6	p	B7	5	One Fourth
98*	6	q	B8	5	One half
99*	5	r	B9	5	Three Fourths
9A	5	a underscore	BA	5	Logical NOT
9B	5	o underscore	BB	5	Logical OR
9C	7	ae dipthong	BC	5	Overbar
9D	5	Cedilla	BD	5	Diaeresis
9E	7	AE dipthong	BE	5	Acute
9F	5	International currency symbol	BF	5	Double underscore

HEX	UNITS	NAME	HEX	UNITS	NAME
C0	5	Open brace	E0	5	Reverse slash
C1*	7	A	E1*	5	Numeric Space
C2*	7	B	E2*	6	S
C3*	7	C	E3*	7	T
C4*	7	D	E4*	7	U
C5*	6	E	E5*	7	V
C6*	6	F	E6*	7	W
C7*	7	G	E7*	7	X
C8*	7	H	E8*	7	Y
C9*	4	I	E9*	6	Z
CA*	5	Syllable Hyphen	EA	5	Square (superscript 2)
CB	5	o circumflex	EB	7	O circumflex
CC	5	o diaeresis	EC	7	O diaeresis
CD	5	o grave	ED	7	O grave
CE	5	o acute	EE	7	O acute
CF	5	o tilde	EF	7	O tilde
D0	5	Close brace	F0*	5	0
D1*	5	J	F1*	5	1
D2*	7	K	F2*	5	2
D3*	6	L	F3*	5	3
D4*	7	M	F4*	5	4
D5*	7	N	F5*	5	5
D6*	7	O	F6*	5	6
D7*	6	P	F7*	5	7
D8*	7	Q	F8*	5	8
D9*	7	R	F9*	5	9
DA	3	Dotless i	FA	5	Cube (superscript 3)
DB	6	u circumflex	FB	7	U circumflex
DC	6	u diaeresis	FC	7	U diaeresis
DD	6	u grave	FD	7	U grave
DE	6	u acute	FE	7	U acute
DF	6	y diaeresis	FF		Eight Ones

APPENDIX C. DOCUMENT PROFILE

The DIA document profile to be used for final-form-text documents is Document Profile Type 3. The structure defined by DIA is as follows:

LENGTH	ID	BASE SUBPROFILE		
	FORMAT	LENGTH	IDF	BASE
LL	X'CA0301'	LL	X'CA0401'	PARAMETERS

Figure 5. Document Profile

The base subprofile (the only subprofile required for final-form-text documents) contains self-identifying parameters that specify attributes of the final-form-text document. Each parameter starts with a two-byte binary field that specifies the length of the parameter (including the length field) followed by the IDF field.

The format specification and general requirements for a Type 3 Document Profile (including complete detail on the base subprofile parameters) are addressed in the profile architecture. Please see Document Interchange Architecture: Interchange Document Profile, for further information.

The final-form-text base subprofile must contain the following required parameters:

Document Name
Document Type = X'0002'
Profile CGCSGID

The final-form-text base subprofile may contain any of the following optional parameters:

Author
Copy List
Creation Date-Time
Document Date
File Cabinet Reference
Last Changed Date-Time
Owner
Subject

The Document CGCSGID parameter, defined as an optional parameter in the base subprofile of the Interchange Document Profile (IDP) Architecture specification, is invalid in a final-form-text profile. If present, the parameter is ignored.

The two conditional parameters (Revisable-Form-Text Parameters, System Code), defined in the base subprofile of the IDP architecture, are also invalid in a final-form-text profile, since the specified conditions are not applicable. If present, both of these parameters are ignored.

The use and processing of the profile is not defined by the Final-Form-Text Architecture.

GLOSSARY

This glossary includes words and phrases that have special meanings in the Final-Form-Text Architecture. The terms are defined as they are used in this book. The one-byte and multibyte controls are not included in the glossary. Descriptions of them can be found in the body of this manual. If you cannot find the term you are looking for, refer to the index or to the Vocabulary for Data Processing, Telecommunications, and Office Systems, GC20-1699.

adjust. A feature that automatically adjusts the line endings of text to comply, within the line-end zone, with the original margin setting or to changed settings, with or without editing.

baseline. The horizontal line connecting the bottoms of capital letters. Graphics may be placed on the baseline so that descenders are below the line and the remainder is above. An imaginary line upon which a sequence of graphic characters appear to rest.

character. A symbol used in presentation for a video display or printing. For example, a letter of the alphabet, a numeral, a punctuation mark, or any other symbol that represents information.

character box. The rectangular area that can be occupied by a character on the printed page or in the page image. The size of the rectangular area varies with the pitch of the character and the number of lines per inch.

character set. A set of different characters that is agreed to be complete for some purpose.

class. The indicator within an introducer that identifies a group of controls that have a common purpose or attribute.

code. A system of bit patterns to which specific graphic or control meanings have been assigned.

coded graphic character. A graphic character with its assigned code point.

coded graphic character set. A set of graphic characters with their assigned code points.

code page. A specification of code points for each graphic character in a set, or in a collection of graphic character sets. Within a given code page, a code point can have one and only one specific meaning.

code point. One of the bit patterns specified by a code.

control function. An action that affects the processing and the interpretation of data.

control sequence prefix (CSP). The introducer (first byte = X'2B') for all multibyte controls.

default value. A value that is pre-assigned and assumed when when no explicit assignment has been made in the document.

destination node. The OSN that provides services for attached source and recipient nodes.

DIA. Document interchange architecture.

distribution. In general, the function provided by DIA of

transporting information from a source node to one or more recipient nodes.

distribution document name. A unique identifier assigned to each distribution request.

distribution system. The collection of office system nodes, source nodes, and recipient nodes that are interconnected to form an office system network.

document. (1) A data medium and the data recorded on it, that generally has permanence, and that can be read by human or machine. (2) A unified collection of information pertaining to a specific subject or related subjects.

document content architecture. A family of data stream architectures that define and specify the form of information by describing the syntax and semantics of allowable elements in the data stream.

document content introducer. The DIU data stream component that identifies the beginning of the document content.

document interchange architecture (DIA). The specification of rules and data streams necessary to interchange information in a consistent, predictable manner.

document interchange unit (DIU). The basic unit of information exchanged between DIA processes.

document type. A classification that identifies the structure and format of a document.

EBCDIC. Extended Binary Coded Decimal Interchange Code.

edit. To create or modify the contents of a document. For example, to insert, delete, change, rearrange, or copy lines.

end node. A node that interfaces directly with either a source or a recipient.

end user. The ultimate source or destination of information flowing through a distribution system.

exception alternate action. A defined fallback action that a product executes when it does not support an optional control, parameter, or parameter value.

exception class. A code that indicates the severity and category of the exception condition detected by the receiving process.

final-form text. The final-form text is text that has already been formatted and is ready for presentation.

font. An assortment of type, all of one size and style.

format. A declaration of composition state values. For example, see SPPS. A set of specific conditions that determine the final position of text on a page.

formatter. A computer program that prepares a document for presentation. For example, the presentation may be on paper or a display screen.

CGCSGID. Coded Graphic Character Set Global ID, a concatenation of 2 two-byte numbers that specify the coded character set ID and code page ID.

graphic. The character image assigned to a graphic code.

graphic code. An EBCDIC code point, X'40' to X'FE'.

graphic location. The coordinates of the graphic's reference point.

hexadecimal. Pertaining to a number system based on 16, using the sixteen digits 0, 1, . . . 9, A, B, C, D, E, and F. For example, hexadecimal 1B equals decimal 27.

indent. To set or start typographical material to the right of the left margin.

initial value or condition. A value that is assumed for the parameters of the control functions until that parameter value is explicitly changed. Synonymous with default values.

IDP. Interchange document profile.

interchange document profile (IDP). A set of descriptors that identify and describe a document.

justification. The process of evenly distributing and inserting extra blank space between the words in an output line to cause the right-hand edge of the last word in the line to reach the right margin. As a result, the right-hand edge of each output line is aligned with the right margin.

line end. One or more controls, the combination of which reset the horizontal pointer to the current or active left margin.

line location. The y-coordinate of the line's baseline.

location. A point addressed by an x,y coordinate pair relative to the page image origin (x=0,y=0). X and y express the number of units between the respective axes and the point.

multibyte control. A variable length control sequence that is used to provide code extensions.

office system node. The DIA process that provides the services for attached source or recipient nodes.

one-byte controls. The control characters of EBCDIC whose occurrence in a particular context initiates, modifies, or stops a control function. A control character may have a graphic representation in some circumstances.

originating node. The office system node that provides services for attached source nodes.

OSN. Office system node.

output device. A machine used to print, display, or store the result of processing.

output line. A line of text produced by a text processor.

pitch. A number that represents the amount of horizontal space a character occupies on a line. For example, 10-pitch means 10 characters per inch, or each character is 0.1 (1/10) inch wide.

position. A rectangular space addressed by a single cardinal number relative to some boundary, usually the left margin. There is never a position zero. The rectangular space could be a graphic box, a 1/1440th inch box, or a box of any other unit.

presentation. The printing of the document on a printing device or the displaying of it on a video display.

process. A systematic sequence of operations to produce a specified result. Generally used in this publication to mean a computer program.

profile parameter. A field of a subprofile that identifies and describes the document.

proportional spacing. The spacing of characters in a printed line so that each character is allotted a space proportional to the character's width.

protocols. The set of rules that govern the operation of the functional units of a communication system.

recipient. An end user that receives information in an office system network.

recipient node. A DIA logical component that provides services on behalf of recipients.

reference point. The point that is the intersection of the left edge of the graphic box with the base of the graphic, ignoring descenders. The location of a graphic is the location of its reference point.

rightmost position. A position whose right edge is coincident with the right margin.

segmentation. The division of a DIU data stream component into two or more segments.

semantic. The meaning or interpretation of a language. A precise meaning and a single interpretation of the control functions defined in Final-Form-Text Architecture.

SNA. Systems network architecture

SNA character string (SCS). A character string composed of EBCDIC controls, optionally intermixed with end-user data.

source. An end user that request services in an office system network.

source node. A DIA logical component that provides services on behalf of sources.

subprofile. A set of profile parameters that describe the characteristics and attributes of a document.

syntax. The arrangement or structure of language. The format of the control functions defined in Final-Form-Text Architecture.

system code. An identifier associated with the originator of the document that is contained in a DIU document unit.

terminal. A device, usually equipped with a keyboard and some kind of display, capable of sending and receiving information over a communication channel.

text. Any sequence of graphic codes, one-byte controls, and multibyte controls.

text line. Any sequence of graphics and controls not containing a line end, followed by one or more contiguous line ends.

type. The indicator within an introducer that identifies the specific operation to be performed.

word processing. Pertaining to machines, systems, or processes, that provide: (a) efficient text entry techniques, (b) serial processing of text and control character strings, (c) final-form-text presentation (printed or displayed) for business communications.

INDEX

T

U

W

Document Content Architecture:
Final-Form-Text Reference
Order No. SC23-0757-0

READER'S COMMENT FORM

You may use this form to communicate your comments about this publication, its organization, or subject matter, with the understanding that IBM may use or distribute whatever information you supply in any way it believes appropriate without incurring any obligation to you. Your comments will be sent to the author's department for whatever review and action, if any, are deemed appropriate.

Note: *Copies of IBM publications are not stocked at the location to which this form is addressed. Please direct any requests for copies of publications, or for assistance in using your IBM system, to your IBM representative or to the IBM branch office serving your locality.*

Possible topics for comment are:
Clarity Accuracy Completeness Organization Coding Retrieval Legibility

If you wish a reply, give your name, company, mailing address, and date:

_____

_____

_____

_____

Note: Staples can cause problems with automated mail sorting equipment. Please use pressure sensitive or other gummed tape to seal this form.

What is your occupation? _____

Number of latest Newsletter associated with this publication: _____

Thank you for your cooperation. No postage stamp necessary if mailed in the U.S.A. (Elsewhere, an IBM office or representative will be happy to forward your comments or you may mail directly to the address in the Edition Notice on the back of the title page.)

SC23-0757-0

Reader's Comment Form

Cut or Fold Along Line

Fold and tape | Please Do Not Staple | Fold and tape

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST CLASS PERMIT NO. 40 ARMONK, NEW YORK

POSTAGE WILL BE PAID BY ADDRESSEE

International Business Machines Corporation
Department D31, Building 803
11400 Burnet Rd.
Austin, Texas 78758

Fold and tape | Please Do Not Staple | Fold and tape

IBM®

Document Content Architecture: Final-Form-Text Reference Printed in U.S.A. SC23-0757-0

Document Content Architecture:
Final-Form-Text Reference
Order No. SC23-0757-0

READER'S COMMENT FORM

You may use this form to communicate your comments about this publication, its organization, or subject matter, with the understanding that IBM may use or distribute whatever information you supply in any way it believes appropriate without incurring any obligation to you. Your comments will be sent to the author's department for whatever review and action, if any, are deemed appropriate.

Note: *Copies of IBM publications are not stocked at the location to which this form is addressed. Please direct any requests for copies of publications, or for assistance in using your IBM system, to your IBM representative or to the IBM branch office serving your locality.*

Possible topics for comment are:
Clarity Accuracy Completeness Organization Coding Retrieval Legibility

If you wish a reply, give your name, company, mailing address, and date:

_____

_____

_____

_____

Note: Staples can cause problems with automated mail sorting equipment.
Please use pressure sensitive or other gummed tape to seal this form.

What is your occupation? _____

Number of latest Newsletter associated with this publication: _____

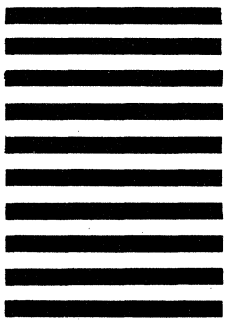
Thank you for your cooperation. No postage stamp necessary if mailed in the U.S.A. (Elsewhere, an IBM office or representative will be happy to forward your comments or you may mail directly to the address in the Edition Notice on the back of the title page.)

SC23-0757-0

Reader's Comment Form

Fold and tape | Please Do Not Staple | Fold and tape

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES



BUSINESS REPLY MAIL

FIRST CLASS PERMIT NO. 40 ARMONK, NEW YORK

POSTAGE WILL BE PAID BY ADDRESSEE

International Business Machines Corporation
Department D31, Building 803
11400 Burnet Rd.
Austin, Texas 78758

Fold and tape | Please Do Not Staple | Fold and tape

IBM®

Cut or Fold Along Line

Document Content Architecture: Final-Form-Text Reference Printed in U.S.A. SC23-0757-0

SC23-0757-0

Document Content Architecture: Final-Form-Text Reference Printed in U.S.A. SC23-0757-0

IBM®