Communications Research Centre

COMMUNICATIONS CANADA
CRC
LIBRARY – BIBLIOTHÈQUE

TELIDON

VIDEOTEX PRESENTATION LEVEL PROTOCOL: AUGMENTED PICTURE DESCRIPTION INSTRUCTIONS

by

C.D. O'BRIEN, H.G. BOWN, J.C. SMIRLE,
Y.F. LUM AND J.Z. KUKULKA

IC

LKC
TK
5102.5
.R48e
#709
c.2

CRC TECHNICAL NOTE NO. 709-E

Government of Canada
Department of Communications

Gouvernement du Canada
Ministère des Communications

OTTAWA, FEBRUARY 1982

Canada

COMMUNICATIONS RESEARCH CENTRE

DEPARTMENT OF COMMUNICATIONS
CANADA

Industry Canada
Library - Queen
MAR 2 0 2013
Industrie Canada
Bibliothèque - Queen

TELIDON

VIDEOTEX PRESENTATION LEVEL PROTOCOL: AUGMENTED PICTURE DESCRIPTION INSTRUCTIONS

by

C.D. O'Brien, H.G. Bown, J.C. Smirle, Y.F. Lum and J.Z. Kukulka

(Information Technology and Systems Branch)

Technical Editor: A. Kwan

Illustrations by P. McLenachan

CRC TECHNICAL NOTE NO. 709-E

February 1982
OTTAWA

CAUTION
This information is furnished with the express understanding that:
Proprietary and patent rights will be protected.

T E L I D O N

VIDEOTEX PRESENTATION LEVEL PROTOCOL: AUGMENTED PICTURE DESCRIPTION INSTRUCTIONS

Table of Contents

Table of Contents
(Cont'd)

Table of Contents
(Cont'd)

List of Figures

List of Tables

T E L I D O N

VIDEOTEX PRESENTATION LEVEL PROTOCOL: AUGMENTED PICTURE DESCRIPTION INSTRUCTIONS

By

C.D. O'Brien, H.G. Bown, J.C. Smirle,
Y.F. Lum and J.Z. Kukulka

ABSTRACT

The Telidon Videotex system is a method by which information and transactional services can be accessed from information sources by the general public. By the use of a domestic home television receiver typically augmented by a micro-computer controlled interface device, a user can access pages of graphical and textual information over public common carrier communications facilities including the telephone network, a cable television line or encoded into unused space in a broadcast television signal. In order to transmit information to a Telidon terminal at a minimum bandwidth, and in a manner independent of the type of communications channel, a coding scheme has been devised which permits the encoding of a picture into the geometric drawing elements which compose it. These "Picture Description Instructions" are an alpha-geometric coding model and are based on the primitives of POINT, LINE, ARC, RECTANGLE, POLYGON and INCREMENT. Text is encoded as (ASCII) characters along with a supplementary table of accents and special characters. A mosaic shape table is included for compatibility.

Substantial agreement has emerged in North America about the coding scheme to be used for Videotex services. This agreement is based on an extension to the CCITT Recommendations S.100 and F.300. This document provides a detailed specification of the coding scheme as well as a description of the principles which make it independent of communications channel and display hardware. The associated CRC Technical Notes 697 and 699 present a general description of the design philosophy of Telidon as well as of the technical description of the alpha-geometric Videotex coding scheme which formed part of the basis for the international standards CCITT Recommendations S.100 and F.300 and the Canadian field trial standard. This document serves as an input to the Canadian Standards Association working group on Videotex standards.

1. INTRODUCTION

Videotex is a name used internationally to represent a class of home and business information services which disseminate information or provide for transactional services from public information suppliers. The system makes use of the home television set as an essential still-picture display medium where the consumer may select what is displayed. An electronic module can be added to the television set to allow it to assemble and display an image consisting of characters and pictorial drawings. The information to be displayed is typically received either from a data connection over a telephone line or from encoded data in the unused scan lines of an over-the-air television or cable television signal.

There are a number of such public access information systems under development in various countries which are designed to operate over different communication lines and on various types of terminals. Unfortunately, the major problem with many of these systems is that the limitations of the particular display hardware, especially with respect to resolution, are reflected in the communications protocols they utilize. The protocol presented in this document is based primarily on alpha-geometric coding principles and is independent of the type of display hardware or communications channel.

Rapid advances in the state of the art in electronics in the past decade have changed the original design constraints for Videotex systems. The cost of memories has dropped tremendously with the result that high resolution pictures may now be stored and displayed on home television sets or other display apparatus at a moderate cost. In addition, the introduction of inexpensive micro-computers has allowed sufficient sophistication to be programmed into a terminal so that a high level, efficient communications code could be utilized.

The basic design approach for the Canadian Videotex system is based on the premise of independence from particular hardware display apparatus limitations such as the resolution of current display techniques. The concept of forward and backward compatibility is of central importance. Forward compatibility means that the communication codes must be designed in such a way that future terminals will be able to access old data. This allows for growth and means that a future terminal with higher resolution can accept low resolution or otherwise limited data from an established data base. Backward compatibility, which is much more difficult to achieve, means that an installed inventory of terminals can receive and decode all future command formats in an intelligent manner. Again using resolution as an example, a picture which is communicated in high resolution should appear as a low resolution picture on older or less expensive terminals and as a high resolution picture on those terminals equipped to handle high resolution. The broadcast colour television signal format is a good example of a design for both forward and backward compatibility. At one time only monochrome television sets and signals existed, but now the situation is more complex. Colour sets can receive both black and white signals as well as colour signals (forward compatibility). Monochrome television sets can receive black and white signals, and when receiving colour signals, they interpret them as black and white (backward compatibility). Technically, this compatibility was very difficult to achieve with television signals, but it has allowed the introduction of colour television without the need to replace the

installed inventory of television sets. The same type of growth potential must be designed into the Videotex communications protocol from the start.

For the above reasons, the Videotex Presentation Level Protocol includes the Picture Description Instructions which have been defined in terms of the geometric primitives of POINT, LINE, ARC, RECTANGLE, POLYGON and INCREMENT, and which describe the structure of the entity to be drawn. For example, a line is drawn by specifying its endpoints. It is the responsibility of the terminal to decode this high level description and to draw the best line possible between the two endpoints. On a high resolution display, finer increments are used to draw the same line that would be displayed in a coarse manner on a low resolution display. In this way, both forward and backward compatibility is achieved because drawing commands are described in terms of geometric primitives rather than in terms of some parameter of the display hardware. The accuracy with which coordinate positions are specified to describe these geometric primitives is reflected in the communications code, but this can be dealt with in a compatible manner by allowing the resolution of this description to be varied and by truncating the coordinate description to the accuracy which can be handled by the terminal.

The term "alpha-geometric" is used in general to describe drawing information encoded in the Picture Description Instruction format, the details of which will be presented later in this document.

The geometric coding scheme was originally developed at the Communications Research Centre of the Department of Communications, Government of Canada, after many years of research into image communications. The specification was set down in Communications Research Centre Technical Note 699 and was accepted by the Canadian Videotex Consultative Committee as the national Videotex field trial standard. This coding scheme was presented to the CCITT Study Groups I and VIII and resulted in the alpha-geometric model being included in the coding recommendation S.100 and the service recommendation F.300. Based on this international recognition major information service operators in North America are developing their service offerings. The American Telephone and Telegraph company has released a specification for the Videotex Presentation Level Protocol (PLP) which they have adopted. The PLP was developed with a comprehensive technical understanding of the coding scheme presented by the Canadian Department of Communications and it contains several extensions to the geometric coding scheme above the level of functionality standardized in CCITT Recommendation S.100 as well as other functions from CCITT Recommendations S.100 and S.61. This document presents the details and rationale behind the protocol for text and graphics communications towards which a consensus is emerging in North America.

There are many applications of this standardized protocol beyond Videotex public access information systems. Because of the independence of the coding scheme from display hardware limitations, this common coding approach can be used for describing pictures for all types of computer graphics terminals such as calligraphic vector displays, storage tube displays, raster graphic displays or flat panel matrix displays. The geometric coding scheme can be used as the display-independent coding for computer graphics applications. Similarly, the CCITT has provided for a compatible coding structure for the Teletex business communications service and the Videotex Presentation Level Protocol can provide advanced Presentation level facilities to Teletex. The next section will describe the various related text and graphic communication services.

2. SERVICE

Videotex is a generic term used to describe a family of information and transaction services for home or business use wherein a user can select information consisting of both textual and graphical data for presentation on his terminal. There are various different service scenarios evolving which utilize many different communications methods. To name a few, there are voice grade telephone lines, cable television lines, optical fibre communication channels, over-the-air broadcast television and digital communications lines such as packet switched channels. These various types of communication channels vary widely with respect to error rate, data rate, bandwidth and the capability to reverse the channel to transmit data back from the terminal to the information source. Standardized terminology has not yet been developed to describe all these services. In fact, some of these services tend to blend together. The general term Videotex* is often used to encompass all such text and graphic communications systems even though specific services may have distinct service names.

Interactive Videotex is a service based on the use of a two-way communications line between a Videotex terminal and a central data bank. Broadcast Videotex (Teletext) is a service based on the use of one-way communications channels such as over-the-air or cable television signals where the entire repertoire of data pages is continuously transmitted and where the user's terminal waits for and selects the desired page. Except for the nature of the interactive dialogue for information retrieval, the time delay and the amount of information available, both services operate in essentially the same way as far as the user is concerned.

For information retrieval, a subscriber to Videotex would be able to select from a number of pages of material which he or she may wish to display and view. An information supplier would provide a central data bank of information covering a wide range of topics. Typically a subscriber selects which information is to be presented by keying the appropriate selection into a calculator-like keypad or typewriter-like keyboard attached to the television set. As an example, a user who wishes to know the latest sports scores selects the sports page which then presents a menu consisting of a list of sports to choose from. By keying in the menu selection for hockey, the current hockey scores might be presented.

A Videotex system consists of an information source, a communication system to distribute the information data, and a terminal to interpret and display the data requested by a susbscriber. Because there will be a large number of terminals in many locations such as homes, offices and schools, the terminal is the most important component of the entire Videotex system with respect to both capabilities and cost. It is envisaged that a number of manufacturers will

---

*The term "Teletext" (not yet adopted by the CCIR) is commonly referred to as a broadcast television Videotex service. It is different from "Teletex", a term adopted officially by CCITT to define a specific type of terminal-to-terminal text communications service. This is an area of confusion which has been recognized by both the CCITT and the CCIR and reconciliation of terminology is a matter for further study.

produce terminals for Videotex of varying levels of cost and sophistication. A high quality terminal capable of very fine resolution may find use in an office situation, while a low cost minimum-feature terminal may be of greatest demand in the cost-sensitive home consumer market. A range of terminals will provide the public with choice. A manufacturer is free to produce any level of sophistication in a terminal he desires as long as the terminal is fully compatible with the Presentation Level Protocol. The various levels of implementation of certain features such as resolution, character sets, filling of rectangular areas and polygons, etc., will be described later. Although the Presentation Level Protocol does not change for different communications media, there could be differences in the manner of receiving data, in error coding, in data flow control and in transmitting interaction commands back to the host data base computer. These differences must be considered in the specifications for the interface of a terminal to a communication line.

For the purpose of this document, the relative merits of various communications means will not be discussed, although the existence of important systems factors, such as the loading on telephone exchanges and the need for error protection in cable and off-air systems are noted. Rather, the ISO layered system architectural model will be presented which illustrates how the various aspects of a total communications protocol for given services can be made to interwork.

2.1 Communications Means

The common aspect to all Videotex-like text and graphic systems is the identical coding format for the information content. The manner in which data is communicated to the home or business terminal has no effect on the PLP commands themselves but does affect the form of the interactive dialogue for the user. Communications over a narrow-bandwidth voice grade telephone line is predicated on the use of one line per terminal so the response to interactions by the user is very good. However, images take time to build up because the data transfer rate is usually 1200 bits per second. Communications over a subchannel of a cable television system offers a higher bandwidth path into the home, but the path is a broadcast channel into all homes with the result that statistical queueing of requests is required. Responses to interactions would be slower but pictures could build up very quickly. Some cable television systems provide a reverse channel for two-way operation, but if this is not available, another channel over a different medium would be required to transmit interaction commands back to the data base host computer. Standards for these various system aspects exist or are under development for different communications systems.

Several unused lines in the flyback (vertical blanking) period of an over-the-air broadcast television signal may be used to transmit images by encoding the information in this unused bandwidth. This technique is commonly called Broadcast Videotex or TELETEXT. Because no reverse channel is possible over the air, a continuous stream of a small number of pages is sent in a cyclic sequence. A terminal waits until the page it has requested is transmitted and then captures the page and displays it. The limitations are that only a small number of information pages may be accessed and long delays may be incurred in waiting for a page to be presented. A detailed specification for the Broadcast

Videotex (Teletext) transmission technique is given by the Department of Communications Regulatory Service Broadcast Specification BS-14.

Other communications means such as digital data services over the telephone local loop and optical fibre data channels are also capable of providing such services to home or business. Of particular interest are combined communications means, for example, the transmission of individual pictures over a circuit-switched medium such as a telephone line; and the transmission of large blocks of data, whole "magazines", over a broadcast facility such as a cablevision sub-channel for later off-line use.

2.2 Data Bases

Vast data bases of information to be presented on a Videotex terminal would be stored on central computers run by information supplier companies. The cost of establishing and maintaining these data bases will be a major portion of the cost of an entire Videotex system. If the PLP is used as the communication protocol, then compatibility with various types, generations and resolutions of terminals is achieved. Terminal compatibility is important but there are other valuable uses of the PDI codes. The Picture Description Instructions in the PLP are a compact form of describing images and, therefore, are an ideal method of encoding pictures for the data base. Storing PDIs in the data base also simplifies the task of the host computer because all it needs to do upon receiving a request is to select the proper page of information and to dump the stored data to the output channel.

Pages of information in the data base must be generated and maintained by the information supplier company. The fact that the PDI commands are based on geometric primitives such as LINE, ARC and POLYGON means that generation and editing of pictures is also based on geometric primitives. A LINE, POLYGON or other primitive may be added to, deleted from, or otherwise modify a picture. If two areas overlap and one is deleted or moved, the background can be easily re-written so that the information supplier can manipulate his data with a minimum of effort. Furthermore, because the PDIs are described in terms of endpoint and vertex coordinates, mathematical transformations such as translation, rotation and scaling can be provided. The use of the PDI commands in the data base provides both flexibility in manipulating pictures and compact data storage.

2.3 Related Services

There are a large number of information distribution and computation services using remote terminals existing at present which could be enhanced by the addition of graphics in a hardware independent manner. These services are as diverse as communicating word processor machines, a message oriented Teletex service or the communication of terminals with a central computing facility. For most of these text-oriented services a de facto standard has developed with ASCII as the presentation layer standard. Millions of such "Teletype"(R) - like printing and "Video display units" exist and it is a fundamental requirement that Videotex terminals incorporate the functions of these basic business terminals.

(R) "Teletype" is the registered trademark of Teletype Corporation.

In order for a Videotex terminal or a basic business terminal augmented with graphics to interwork with these existing applications it is necessary for ASCII* (American Standard Code for Information Interchange) to be the basic and default text coding standard. Such a Videotex terminal is usable in any such general application. A general business terminal will be able to access a Videotex data base displaying the textual information and, if it supports some minimal aspects of the International Standard for Code Extension (ISO 2022), it can ignore any information it cannot interpret.

* ASCII is equivalent to CSA standard Z243.4-1973.

3. CODING PHILOSOPHY

The Presentation Level Protocol encodes text and graphic information in such a way as to enable it to be easily communicated. Independence of display or communications hardware constraints is achieved by using simple geometric Picture Description Instructions as the basis of the coding scheme.

The standard for encoding text and graphic information is but one of a number of interconnected standards required to provide a Videotex or similar communications service. In terms of the architecture defined in ISO's multilayer reference model of Open Systems Interconnection, the Presentation Level Protocol forms the sixth or Presentation layer.

The ISO layered system architecture is a seven-layered assembly of inter-related protocols required to define an entire communications system. Each layer covers an independent aspect of a communications system in such a way that other protocols may be substituted at various layers in order to operate over different media. For example, the physical layer standard defined for communications over telephone wires is a modem specification, while the physical layer standard for transmission over a television broadcast channel is an electrical radio tranmission specification. Both the Interactive Videotex and Broadcast Videotex (Teletext) services make use of the same Presentation Level Protocol, at layer 6, while having wholely different transmission protocols in layers 1 to 5.

The layered system architecture permits communications protocols to be defined in such a manner that many interrelated applications may interwork. Through the work of ISO and CCITT a unified structure is emerging in which compatible protocols can be defined. An example is the packet switched data transmission protocol presented in CCITT Recomendation X.25 which defines a data transmission service totally independent of the coding format or application of the data transmitted over it. Such a protocol is intended to interconnect with different application data presentation coding and service scenarios and is therefore an open system. The layered system architecture allows Open Systems Interconnection (OSI).

In the layered communications architecture, similar functions are grouped together to form distinct layers. Each standardized layer provides a set of services to the layer above it by performing functions internal to the layer and by utilizing services provided by the layer below it. Since all inter-layer interfaces are well defined, each layer can be considered independent of the particular implementations of other layers. The following diagram illustrates the seven functionally separate layers of the OSI model.

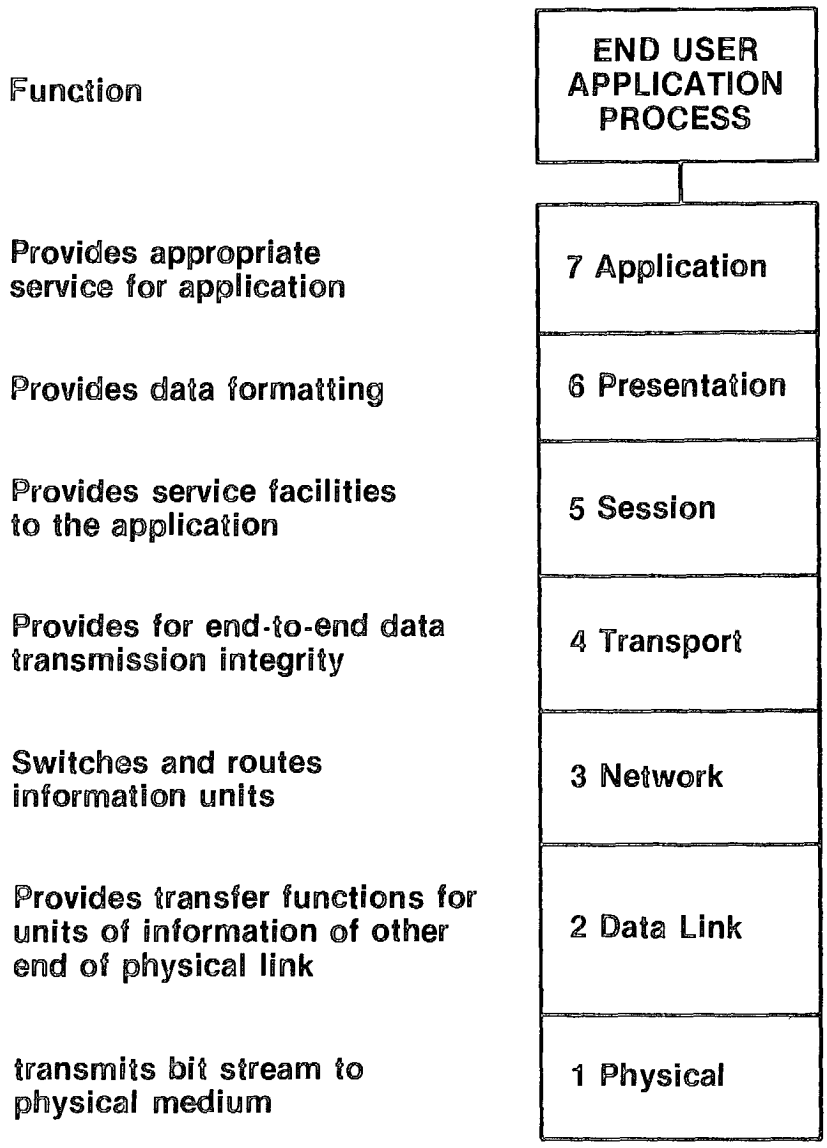


Figure 3.1 - The seven functionally separate layers of the ISO OSI Model

The seven layers may be viewed in two major groupings. Layers 1 to 4 concern the transference of data while layers 5 to 7 concern how the data is processed and used.

The Physical Layer provides mechanical, electrical and procedural functions in order to establish, maintain and release physical connections.

The Data Link Layer provides a data transmission link across one or several physical connections. Error correction, sequencing and flow control are performed in order to maintain data integrity.

The Network Layer provides routing, switching and network access considerations in order to make invisible to the transport layer how underlying transmission resources are utilized.

The Transport Layer provides an end-to-end transparent virtual data circuit over one or several tandem network transmission facilities.

The Session Layer provides the means to establish a session connection and to support the orderly exchange of data and other related control functions for a particular communication service.

The Presentation Layer provides the means to represent and interpret the information in a data coding format in a way that preserves its meaning. The detailed coding formats for the scheme described in this document provide the basis of a Presentation Level Protocol for Videotex and related applications.

The Application Layer is the highest layer in the reference model and the protocols of this layer provide the actual service sought by the end user. As an example, the information retrieval service commands of a Videotex application form part of the application layer protocol.

Although there is need to define certain standardized application level protocols for information retrieval and other basic applications, the presentation layer and lower layer protocols should be defined in such a way so as to be application independent. There are a large number of services which are also associated with Videotex information retrieval such as "Teleshopping" which have unique application level interaction scenarios. In "Teleshopping" a user answers a number of questions in order to fill out an order form and then transmits the transaction data unit to the host data base computer. This application has much in common with all transaction oriented business data processing services. In fact it is very difficult to define a unique videotex application.

As has been indicated previously, various different transmission media may be used to effect the same service. The coded information retains its meaning regardless of whether it is broadcast or transmitted over a bidirectional data channel. It is the responsibility of the session layer protocol to provide service features appropriate to the transmission service, taking into account such factors as the fact that Broadcast Videotex (Teletext) is an inherently one-way transmission medium. A number of different standards are required for the transmission and session layers of these various services.

Since the information coding protocol at the presentation layer preserves the meaning of the information messages in a manner independent of the service or transmission media by which it is delivered, it forms the bridge between many diverse applications.

3.1 Presentation Coding Structure

The most fundamental standard for the information industry is the Presentation Level Protocol, because it forms a common coding scheme for describing all text and graphic information regardless of the service or application. Data which has been encoded in terms of the Presentation Level (or layer) Protocol can be used in Broadcast or Interactive Videotex, as well as many related services and in applications as diverse as word processing and computer graphics. The universal, long standing ASCII standard encodes textual information in a manner independent of the terminal device or the application. The Presentation Level Protocol presented in this document builds upon the existing ASCII standard by including a supplementary character set and a DRCS (Dynamically Redefinable Character Set) capability and includes the Picture Description Instructions and block mosaics to encode pictorial graphics information.

The Presentation Layer Protocol makes use of the existing standards at the presentation layer of the OSI model, such as the well known ASCII standard for text and the techniques for code extension standardized by ISO. The relationships of these character coding standards and the manner in which they are woven into a unified Presentation Level Protocol will be described in some detail in this section.

There are a number of existing standards in wide use at the presentation level which encompass the coding of text and graphics information, and which must be taken into account when developing a unified presentation protocol. The most fundamental is the North American standard ASCII code table. ASCII stands for the American Standard Code for Information Interchange (ANSI X3.4) and has almost universally been adopted by the computer communications industry for the assignment of codes to character coded text. In addition there is an International Reference Version (IRV) code table described in ISO Standard 646 and, ISO Standard 2022 presents the method of extending the functionality of character coded sequences. The CCITT Recomendations S.100 and F.300 covering Videotex standardize the fundamentals of encoding graphical (pictorial) information, and have been used as the bases of the unified Presentation Level Protocol.

The national standard code for text in Canada is CSA Z243.4-1973 which is identical to ASCII. The ASCII standard is identical to the International Reference Version for all characters including versions of the nationally assignable characters, except that the monetary symbol has been chosen to be "$" (position 2/4), and the tilde sign (~) is in place of the overline (‾) in position 7/14.

There is an evolutionary process which goes on in the development of standards. Certain standards are developed to cover certain needs and then their range of applicability is extended as new needs emerge. As new standards are developed they have to be harmonized with existing standards in order to maintain compatability with existing equipment.

Harmonization does not mean incorporating previous standards without any changes, as that would impose an unreasonable restriction. Minor alterations to existing standards are sometimes introduced in order to make room for the added functionallity introduced by the new standards. Since existing inventories of equipment exist, it is not possible to extend or add new standards without harmonization and backward compatability. A prime example of harmonization exists in the extension to the ASCII (ISO 646, CCITT A.5) character code table to include accented characters for all Latin based alphabets.

3.1.1 Text coding

In the character coded method of describing text, particular character codes are identified by an eight bit code sequence in which seven of the bits are used as an index into a 128 character code table and the eighth bit is used for parity error checking or extention to another code table of 128 characters, as will be described later. The text standard consists primarily of the code table describing the shapes of characters.

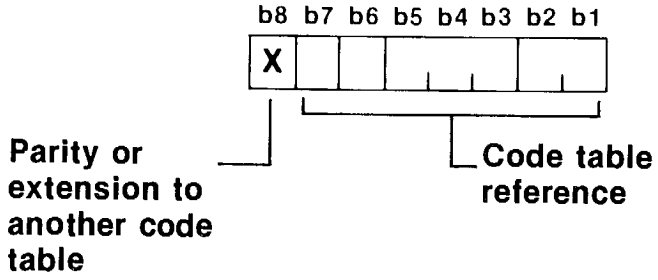


Figure 3.2 Text Coding Format

The character code table is normally represented as a table of eight columns and sixteen rows with bits b7, b6 and b5 addressing the columns and bits b4, b3, b2 and b1 addressing the rows. This general format is used throughout this document. In diagrams, the bits are numbered b1 to b8 with b8 occupying the most significant position. See Figure 3.2.

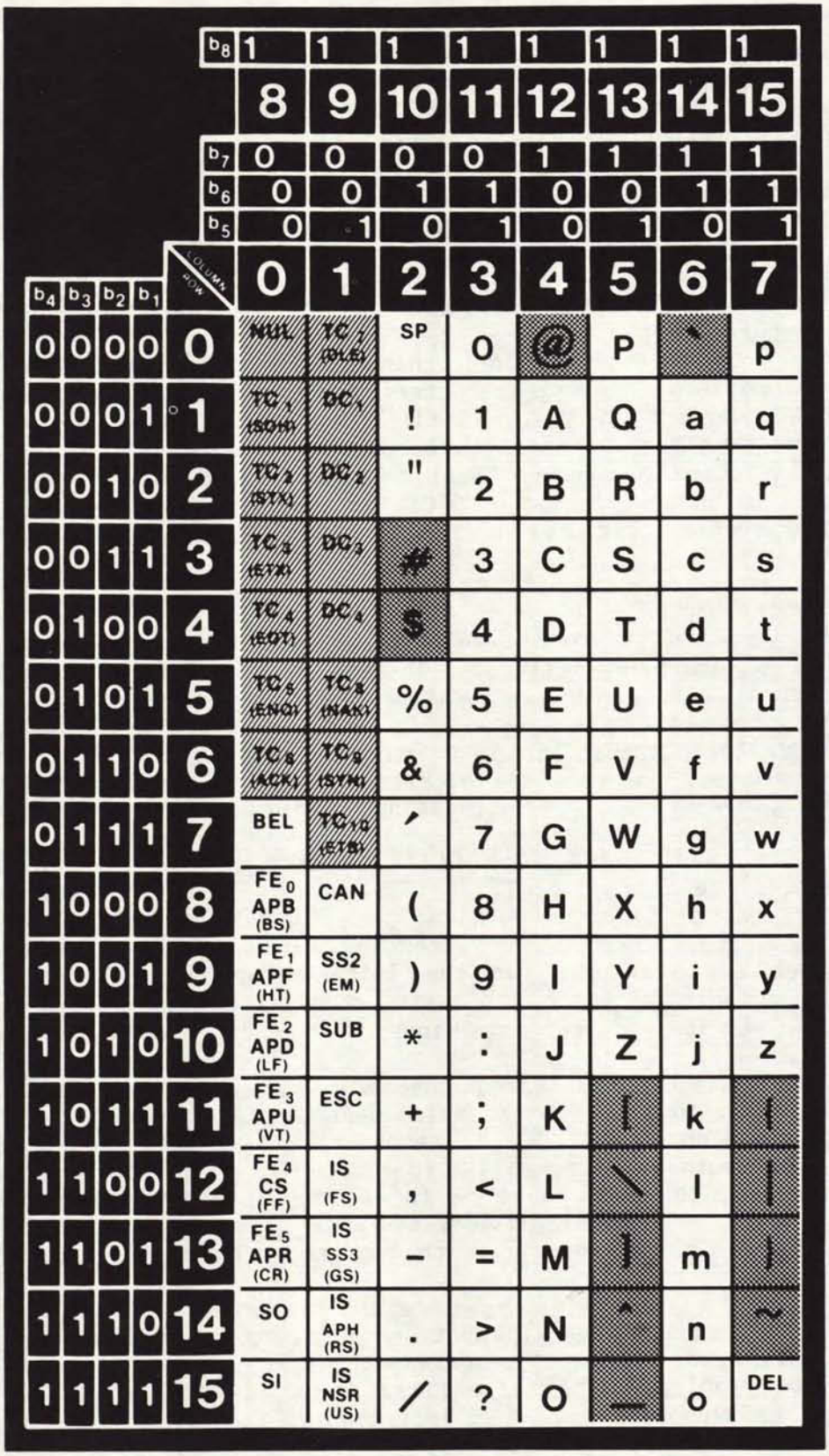
Figure 3.3 illustrates the ASCII code table. The table is subdivided into two areas. The first area occupying columns 0 and 1 contains control characters such as "carriage return" (CR) and "line feed" (LF) and is known as the "C0" (C zero) set. The second area occupying columns 2 to 7, contains by default the printable alpha-numeric characters A-Z etc., and is known as the "G0" (G zero) set. The characters which are national variants of the International Reference Version (IRV) code table are indicated by shading. The remainder of the graphics and control repertoire will be discussed later.

The control characters in the C0 set can be further subdivided into two classes. The first are those characters which are termed transmission control characters and device control characters, indicated here by hatched lines. The transmission control characters (TC1-TC10) are reserved for the use of transmission protocols at the lower levels of the layered architectural model. The device control characters (DC1 - DC4) are used to control device parameters. Typically the characters DC1 and DC3 are used as stop/start flow control codes in many transmission protocols. The transmission control characters, the device control characters and NUL (0/0) are not used by any presentation layer process.

The remaining control characters in the C0 set are available within the Presentation Layer Protocol. These codes can be classified as Format Effector (FE) codes, Information Separator (IS) codes, and Code Extension control codes as well as the Bell (BEL) code. The Code Extension control codes operate in accordance with the code extension standard ISO 2022. Since this standard has recently

been revised, as well as new control code definitions in CCITT Recommendation S.100 having been specified, there is a need for some harmonization. The CCITT has introduced in its recommendation the single shift invocation codes SS2 and SS3 for G2 and G3 code tables as replacements for the little-used code (EM) and information Group Separator code (GS). The CCITT recommendation has also broadened the definitions of the Format Effector (FE) codes so that they cover more generalized devices. This new terminology is indicated on the figure and the details of these new definitions will be covered in a later section.

Reset functions may cross layer boundaries in terms of the open systems layered architecture, and may cause an effect at the Presentation and Application as well as the transmission layers. In particular, the C0 control character CAN (position 1/8) may be interpreted in various ways by transmission level protocols, however its effect is defined in the Presentation Layer Protocol, for those situations in which this reset function is communicated up from the lower layers.



b_4 b_3 b_2 b_1	ROW / COLUMN	0	1	2	3	4	5	6	7
		b_8 1	1	1	1	1	1	1	1
		8	9	10	11	12	13	14	15
		b_7 0	0	0	0	1	1	1	1
		b_6 0	0	1	1	0	0	1	1
		b_5 0	1	0	1	0	1	0	1
0 0 0 0	0	NUL	TC_7 (DLE)	SP	0	@	P	`	p
0 0 0 1	1	TC_1 (SOH)	DC_1	!	1	A	Q	a	q
0 0 1 0	2	TC_2 (STX)	DC_2	"	2	B	R	b	r
0 0 1 1	3	TC_3 (ETX)	DC_3	#	3	C	S	c	s
0 1 0 0	4	TC_4 (EOT)	DC_4	$	4	D	T	d	t
0 1 0 1	5	TC_5 (ENQ)	TC_8 (NAK)	%	5	E	U	e	u
0 1 1 0	6	TC_6 (ACK)	TC_9 (SYN)	&	6	F	V	f	v
0 1 1 1	7	BEL	TC_{10} (ETB)	'	7	G	W	g	w
1 0 0 0	8	FE_0 APB (BS)	CAN	(	8	H	X	h	x
1 0 0 1	9	FE_1 APF (HT)	SS2 (EM)	)	9	I	Y	i	y
1 0 1 0	10	FE_2 APD (LF)	SUB	*	:	J	Z	j	z
1 0 1 1	11	FE_3 APU (VT)	ESC	+	;	K	[	k	{
1 1 0 0	12	FE_4 CS (FF)	IS (FS)	,	<	L	\	l	|
1 1 0 1	13	FE_5 APR (CR)	IS SS3 (GS)	-	=	M	]	m	}
1 1 1 0	14	SO	IS APH (RS)	.	>	N	^	n	~
1 1 1 1	15	SI	IS NSR (US)	/	?	O	_	o	DEL

Figure 3.3 ASCII Code Table

The ASCII code table largely caters to the needs of the English language. For other Latin alphabet based languages where accented letters are an integral part, separate code tables have historically been designed for each linguistic group. The primary text code table for Latin alphabet based languages is shown with the thirteen characters whose meanings and symbols alter among national versions, indicated by shading. This variance between code tables could potentially cause great difficulty in communication between nations whose terminals have implemented different code tables.

In order to alleviate the problems associated with differing code tables, the CCITT in cooperation with ISO has recommended a supplementary character set which contains all the necessary special characters for all Latin based alphabets as well as a method for accenting characters known as the composition method, in which a non-spacing accent character is first transmitted from the supplementary character set and then the character to be accented is transmitted. This method has the great advantage of having total backward compatibility because older, simpler terminals which only implement ASCII or the IRV simply ignore the supplementary accent or special character.

A crude method of accenting characters without the need to implement national code tables had been in use in ASCII and the IRV for some time, by making use of the code FE0 (BS/APB) which would effect a backspace and allow punctuation/ accent marks to be added to letters. This method did not work well on many classes of terminals. Not only was backspace not implemented on many terminals but the defaults were not guaranteed. In addition there are legitimate non-accenting uses for the punctuation/accent marks which have developed. This method has now been abandoned. However, the backspace code and the punctuation/ accent mark codes remain in ASCII and the International Reference Version (IRV).

As can be readily seen the CCITT primary set is a subset of the International Reference Version (IRV) of ISO 646, and therefore also a subset of ASCII. That is, the primary set contains some empty positions as indicated by shading in Figure 3.3. Coding for accented characters is obtained from the supplementary set and the code for the letter from the primary set. The supplementary code table of accents, diacritical signs and special characters for Latin based alphabets is illustrated in Figure 3.4.

In typical usage an accented character would require three bytes to encode. For example, in a seven-bit environment with the primary set designated in its default position as G0 and the secondary set as G2, the coding for an accented character would be thus: an SS2 (position 1/9 in the C0 set) would start the sequence invoking the code table G2 for a single access. The accent would then be specified followed by the primary character. In such a manner the letter é would be coded SS2 ´ e, that is, three characters from code table positions 1/9, 4/2, 6/5.

While this method may be somewhat inefficient, it has the great merit of catering to all the needs of the Latin based languages by just the two primary and supplementary code tables. The technique appears to be general enough to be extendible to other alphabets and, most importantly, it degrades gracefully. If a supplementary table has not been defined, or the code combination is beyond the repertoire implemented on a particular terminal type, then the accent would be missing but the primary character would remain.

The composition coding method for accented characters can be expected to be universally used for text communication in the future. However, there currently exists a large number of terminals throughout the world which use national code tables; this is especially true of the ASCII code table in North America. Most of these terminals are "dumb" terminals in that they do not support the ISO code extension techniques.

Since a very large number of terminals exist in North America which implement only ASCII, and yet in international communications only the character codes indicated by the CCITT are guaranteed, a harmonization problem exists. In order to be compatible with the large number of data sources which use only ASCII, a North American Videotex terminal must implement ASCII as the default G0 text set. It should also implement the supplementary character set as per the CCITT and ISO method. The information provider or data supplier should restrict the generation of information to those character codes which are guaranteed on all terminals within his sphere of interest both in North America and throughout the world.

The implementation of the Latin based alpha-numeric character set is therefore to be as described below:

User Terminal Transmit	User Terminal Receive
a) [ASCII + Supp] Note (1)	a) [ASCII + Supp] Note (2) b) CCITT Primary + Supp
Data Base Transmit	**Data Base Receive**
a) [CCITT Primary + Supp]	a) [ASCII + Supp] Note (2) (Normally from user terminals) b) [CCITT Primary + Supp] (Normally from IPS or from another VHS)

Explanations:

(i) Supplementary Set refers to Figure 3.4.

(ii) For the CCITT Primary Set see Figure 3/S.100 of Reference 9.

Note 1 (a) $ (dollar sign) will be coded as 2/4 (Primary).
The code 2/4 (Supplementary) will not be used.

(b) # (number sign) will be coded 2/3 (Primary).
The code 2/6 (Supplementary) will not be used.

(c) ` (grave accent) will be coded as 4/1 (Supplementary), if used as an accent. The symbol (`) will be coded as 6/0 (Primary) if used for purposes other than as an accent.

(d) ^ (circumflex) will be coded as 4/3 (Supplementary) if used as an accent. The symbol (^) will be coded as 5/14 (Primary) if used for purposes other than as an accent.

(e) ~(tilde) will be coded as 4/4 (Supplementary) if used as an accent. The symbol (~) will be coded as 7/14 (Primary) if used for purposes other than as an accent.

Note 2 (a) Positions 2/4 (Primary) and 2/4 (Supplementary) will be interpreted as $ (dollar symbol).
(b) Positions 2/3 (Primary) and 2/6 (Supplementary) will be interpreted as # (number symbol).
(c) All accent symbols in column 4 (Supplementary) will be "non-spacing" and are normally used to create accented alphabets, using the Composition Method.
(d) the Symbols (˜), (ˆ) and (~) Primary Set are normal spacing characters and will not be used to create accented alphabets.

The designation sequence for the CCITT primary code table, when it is assigned by ISO, will be of use in controlling the transcoding of data in gateways. Since terminal manufacturers will have to define an actual effect for each character in the code table in a real terminal implementation, there need be no distinction at the terminal level between the ASCII code table and the CCITT primary text code table which is a subset.

The supplementary table of accents, diacritical signs and special characters is illustrated in Figure 3.4 This table is based on CCITT Recommendation S.61. However, it includes some additional characters proposed by CCIR, ISO and other organizations. Those additional characters which are not yet included in a CCITT recommendation are indicated by a subscript 1 in the Figure. Manufacturers should make use of the entire table for introduction of the service bearing in mind which characters have been established in CCITT recommendations.

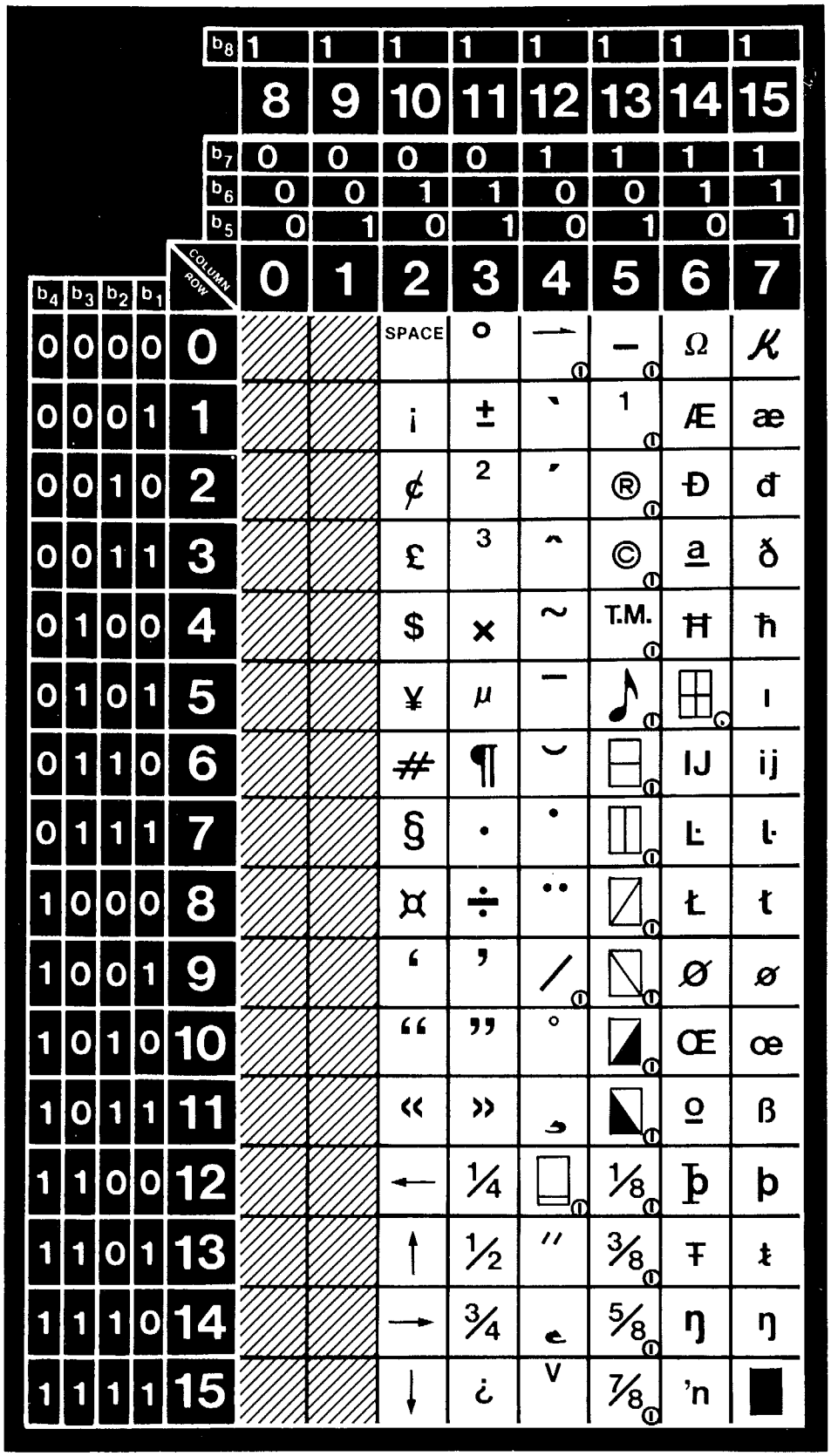


Figure 3.4 Supplementary Character Table

3.1.2 Geometric Pictorial Coding

The code table technique of defining the functionality of communications codes has been extended to cover pictorial information by the definition of the Picture Description Instructions.

These interpretations for the code table consist of sequences of codes to describe graphical operations. The code set is subdivided into two fields, one for operation codes (opcodes) and the other for the numeric data operands associated with an opcode (See Figure 4.1). A code sequence to perform a drawing operation consists of an opcode to draw a POINT, LINE, ARC, RECTANGLE, or POLYGON, followed by a variable-length sequence of data bytes used to encode numeric parameters for the command. This numeric coding is formed out of codes from table columns 4, 5, 6 or 7, and consists of the least significant six bits of each byte for these code table positions.

Another way of interpreting these code table assignments is to examine the bit patterns of the codes. A PDI code consists of an 8-bit data byte which may include a bit for parity error checking or extension to an additional code table. There are, therefore, seven bits of data in each byte.

The format for PDI drawing commands is a 6-bit code field and a 1-bit flag field occupying the 7-bit data field in each byte, as illustrated in Figure 3.5.

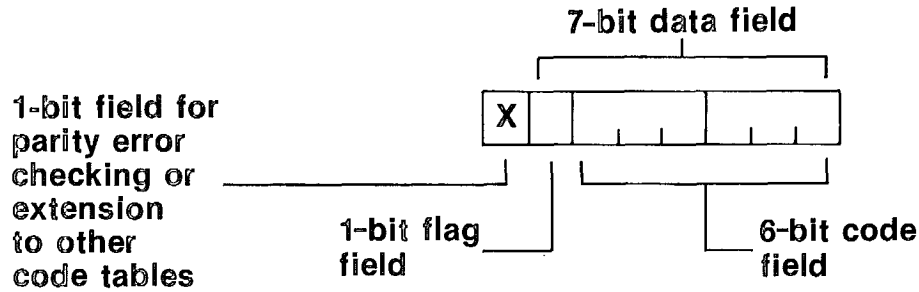


Figure 3.5 PDI Drawing Command Format

The flag field of the command is used to indicate whether the byte represents a command opcode or data associated with the command. The flag field is 0 for opcodes and 1 for numeric data. The number of bytes in the code sequence associated with a particular drawing command is determined from the flag field. A command's domain begins with its opcode byte and terminates with the start of the next code sequence. A command sequence is terminated by an opcode introducing the next drawing sequence or by any other presentation layer code not from the numeric data section of the PDI code table. Transmission layer control codes should not affect the range of a PDI command sequence as they should be removed from the data stream by lower-level processes.

The field of the opcode is further subdivided into a descriptor field and a facilities field (see Figure 3.6). The descriptor field contains the numeric

identifier of each of the eight possible opcodes. The remaining two bits in the facilities field are used for describing optional forms of the instruction. The flag bit (bit 7) equals 0 to indicate an opcode and bit 6 equals 1 to restrict opcodes to columns 2 and 3 from the code table.

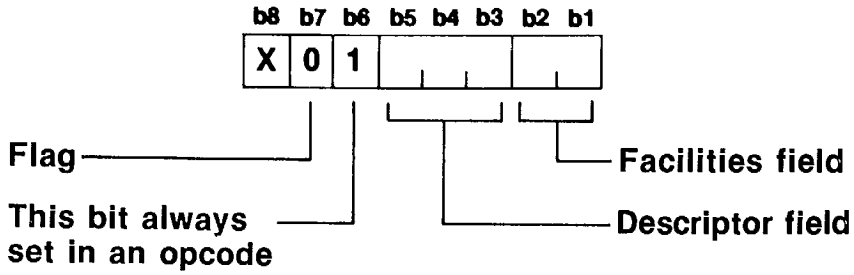


Figure 3.6 Format of a PDI Opcode Byte

There are a possible total of eight opcodes, each having four variants defined by the two facilities bits. The eight opcodes are: POINT, LINE, ARC, RECTANGLE, POLYGON, INCREMENTAL and two of CONTROL. The first five opcodes in this list are the alpha-geometric drawing primitives while the remaining opcodes are used to augment or control these basic drawing commands. A SET position may be encoded within the basic drawing primitives in order to increase transmission efficiency. The SET position describes the start of the drawing operation in Cartesian coordinates (X,Y), on the screen.

The functions of the opcodes are summarized as follows:

POINT - sets the drawing point to any position in the display space and optionally displays a dot.

LINE - draws a line based on its end points.

ARC - draws a circular arc based on the endpoints of the arc and a point on the arc. The endpoints of the arc may optionally be joined by a chord and the area so defined filled in. If more points are given, they define a higher level ARC, a curvilinear line defined by a SPLINE function. A circle is described as an arc whose endpoints coincide and whose intermediate point (with the endpoints) defined the diameter.

RECTANGLE - draws a rectangular outline or fills in an area of specified length and width.

POLYGON - draws a polygonal outline or fills in the circumscribed area based on a series of defined vertices.

INCREMENTAL - draws a point, line or polygon in an incremental manner.

CONTROL - provides control over the modes of the drawing commands. One of its major functions is to set up a value or colour of an object.

Data to be used with a drawing command immediately follows the opcode byte and is distinguished by having the flag bit set to "1". Any number of groups of coordinates (horizontal (x), vertical (y) and depth (z)*) or other data may follow a drawing command. The drawing command may be re-executed for each group of data so that, for example, a series of concatenated lines may be specified by one opcode command and followed by the appropriate number of (dx,dy) coordinate pairs. Repeated SET and POINT data can be used to point-plot a graph and repeated RECTANGLE data can be used to draw histograms. Those drawing commands such as POLYGON and ARC (spline), which use a variable amount of data, end their data list upon the next opcode or other presentation level code not from the numeric data section of the PDI code table.

PDIs have been defined to be independent of the physical resolution of the display media. This accommodates future technological advancements in display apparatus without obsoleting current terminals and data banks defined at a lower resolution.

The data associated with a primitive drawing command is coordinate information. A Cartesian coordinate or displacement in two's complement notation is specified as a signed number, usually specified to 9 bits of accuracy including one sign bit. The origin (0,0) of the Cartesian coordinate system is the lower left hand corner of the visible display screen.

The coordinate specifications are based on a Cartesian 0 to 1 numbering scheme with positions being specified as fractions of this range. This is independent of the physical resolution of the apparatus which may use a television set for display with on the order of 256 positions of resolution in the horizontal direction, or a high resolution display apparatus of 1024 positions or any other resolution. Any drawing commands or text display operations which attempt ot draw outside of the valid 0 to 1 Cartesian area are in error. Drawing specifications outside this absolute area, either directly specified in the negative coordinate space below 0 or specified by relative displacements, are in error. The effect on the terminal of such an erroneous condition is undefined.

Coordinate specifications may be described to several levels of accuracy because they are represented as fractions of the visible drawing area. Unnecessary least significant bits are eliminated by truncation when the specification is to a greater accuracy than can be handled by the terminal. The numbering scheme 0 to 1 is a single-ended open range, that is, the point 0 is part of the valid drawing area but the 1 is inaccessible.

Display screens with non-square visible areas map into the square drawing area such that (0,0) remains in the lower left hand corner. On a television-like screen with a 4:3 aspect ratio, this corresponds to 0.00 to 0.99... in the x axis and 0.00 to approximately 0.75 in the y axis. Drawing commands into the entire square 0 to 1 grid are permissible but only the circumscribed 4:3 area is visible. The 4:3 aspect ratio need not be exactly adhered to as it is the entire active television signal which is defined on the 4:3 ratio including the border

---

* the depth (z) coordinate specification is only meaningful on terminals with a capability for operating in 3-D mode. The complete specification of 3-D mode operation is for future study.

areas. For example, a manufacturer might build a terminal based on a matrix of 256 x 200 picture elements, which can display up to approximately 0.78 in the y axis. The amount of the display area which can actually be viewed on a consumer-grade television set depends primarily on the overscan characteristics of the set. This effect will be discussed later.

The exact number of scan lines, picture elements or other basic parameters must be derived from the hardware for any particular terminal implementation; and for ease in building the hardware, it must bear a simple relationship to the geometric coordinate system. The number of characters which may be displayed readably on the screen, the number of scan lines of the North American 525 line TV system and the aspect ratio of 4:3 do not divide evenly. It is permissible for a terminal manufacturer to deviate slightly, by no more than 5%, from the aspect ratio in order to accommodate restrictions in the organization of pixel memory and/or character display hardware.

The default range of the PDI number system is 9 bits occupying 3 bytes of data to describe an x,y coordinate position as illustrated below. It is envisaged that this 3-byte description would be used to communicate the majority of Videotex pictures.

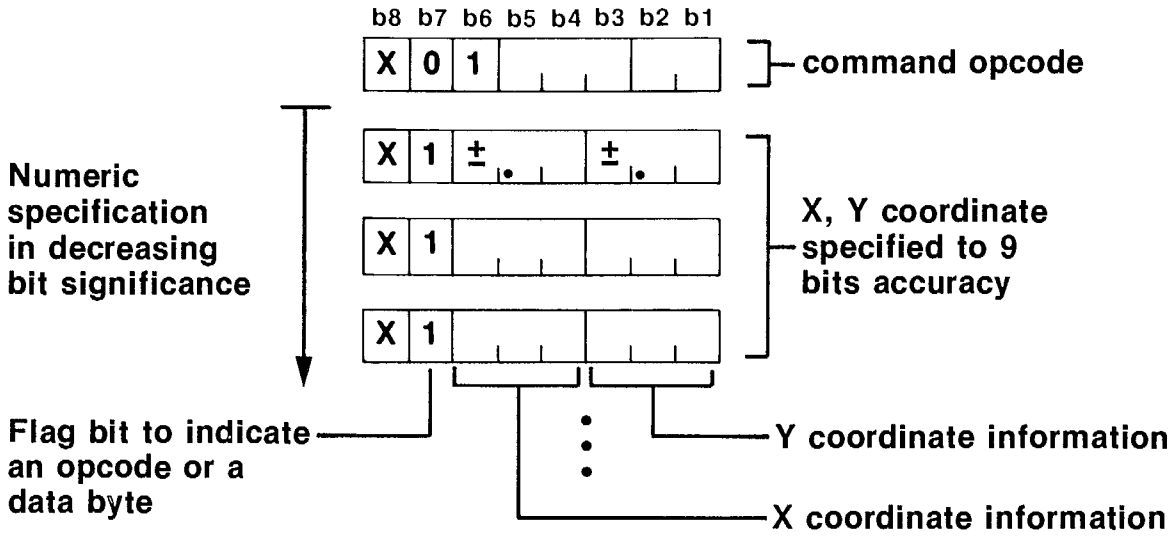


Figure 3.7 2-D MODE

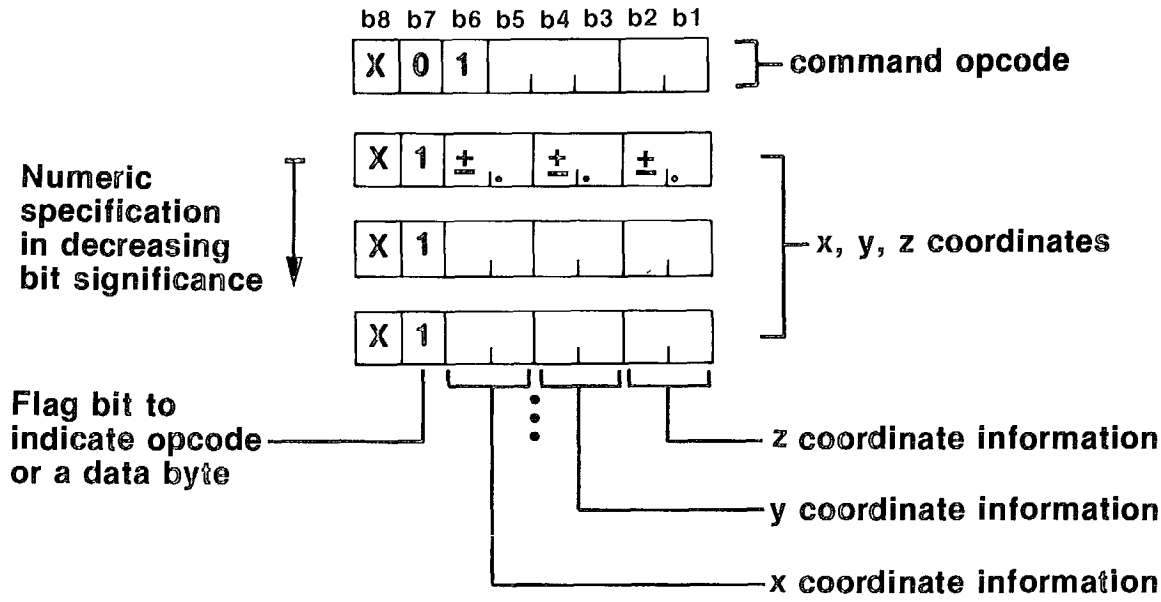


Figure 3.8 3-D MODE

For images that need to be specified to greater resolution than can be specified in a number range of 9 bits (+ 256) an additional data byte may be added to the coordinate specifications extending the range to 12 bits. Other higher or lower number ranges may be defined (see Figures 3.7 and 3.8). This number system in no way restricts the terminal manufacturer from implementing any resolution he feels is cost effective at the basic hardware level. The number system merely defines the manner by which pictures are described so that there is display hardware independence in the PDI code. The manufacturer would convert coordinate information from the PDI coordinate system to his working coordinate system. If the hardware number system bears a simple relation to the PDI number system, this would be a trivial conversion. These coding options are made available so that both information suppliers and terminal manufacturers can make these trade-offs in a compatible manner.

3.1.3 Other Pictorial Coding

The CCITT in its Recommendations S.100 and F.300 present several mutually exclusive optional methods of coding data for a Videotex service. Two of these methods, the alpha-geometric method and the alpha-mosaic method are currently defined in some detail. Other options such as the alpha-photographic option (point-by-point facsimile-like) are left for further study by the CCITT recommendation.

In order to provide international compatibility via at least a gateway function, it is desirable to have a minimum capability within each terminal to permit data conversion from these other coding methods. To take the alpha-photographic method of coding Videotex data as an example, a minimum point-by-point drawing capability is provided in the INCREMENTAL POINT command. Since the INCREMENTAL command, like all PDI commands, is defined in a resolution and hardware independent manner, it is possible to create an automatic gateway conversion program which will generate INCREMENTAL POINT data from alpha-photographic data coded in any coding scheme.

In the alpha-mosaic approach, the display screen is divided into a fixed number of character positions, typically 20 rows by 40 character positions per row for a total of 800 characters for a 525 line television system. Each position is fixed, somewhat like the squares of a chessboard. Coarse graphics is accomplished by subdividing each character position into six sub areas in order to define 72 x 80 positions on a European 625 line TV display or 60 x 80 on a North American 525 line TV display. The major problem is not that the resolution of these display terminals is low, but rather that it is tied to the hardware resolution limitations of the particular terminal technology.

There are two alpha-mosaic coding schemes defined in the CCITT recommendations of Videotex. These two methods, the so-called serial and parallel attribute schemes make use of entirely different coding formats to define the attributes (such as drawing colour) pertaining to the display. Recent work in CEPT (Conference of European Postal and Telephone Administrations) has produced rules for interworking of these two schemes thereby defining what happens when both schemes are implemented in one terminal or when a gateway is constructed.

In addition to the two alpha-mosaic schemes defined in CCITT recommendations there are other extremely popular, but incompatible mosaic coding schemes used in the personal and small business computer industry and operating on a different character density, typically 16 rows by 32 characters.

The Presentation Level Protocol contains a basic mosaic shape table of the 2 x 3 sub shapes used within each character cell which may be used in conjunction with the alpha-geometric attribute coding. This permits data conversion from any alpha-mosaic scheme using the 2 x 3 sub shapes by an automatic gateway function. Different character densities may be accommodated by text scaling.

3.1.4 Redefinable Primitives

The Presentation Level Protocol defines one other basic method of encoding data. However, this method is very open ended and can be used for a wide variety of purposes. The technique of defining a section of the picture description code to be a "sub-picture" which may be reused several times, is well known in the Computer Graphics field. In the Presentation Layer Protocol this feature may be provided and is called a MACRO drawing primitive.

A MACRO-PDI consists of a sequence of presentation level commands stored under a single character label. When a character from the MACRO code table is transmitted, the sequence is executed. A control sequence is used to delimit the sequence of codes which are buffered to form a MACRO, and an entire G table of 96 character positions is reserved for MACRO names.

A Dynamically Redefinable Character Set (DRCS) capability is provided along with the MACRO capability in order that character sets primarily for non-Latin based alphabets or special alphabets may be defined. DRCS is the TEXT case of the MACRO facility and differs from the macro capability in that the sequences defining DRCS shapes are pre-executed; that is, they are executed at definition time. At execution time, TEXT attributes apply.

For the definition of these redefinable primitives, any presentation level code sequence may be used. It is even permissible for MACRO's to invoke MACROs. When used in conjunction with the WAIT command and the modification of colours using a colour look up table, very sophisticated animations may be produced.

3.2 Presentation Code Environment

Various existing and enhanced standards have been woven together to form the Presentation Layer Protocol. To understand the Presentation Layer Protocol completely it is necessary to look at the way in which these codes interweave.

ISO Standard 2022 describes the method of code extention to be used with character coded protocols. It describes the method by which code tables may be "designated" and "invoked". A byte of coded data acts as a pointer into a combined code table consisting of a C control code table and a G set (or text code table). In most applications there are not enough functions (or characters) available in a single G set, so provision has been made in the structure to permit G sets to be switched.

The G and C sets currently in use are called the current G and current C sets. There are four G sets and two C sets which are designated at any one time; that is, any one of the four tables G0, G1, G2 or G3 could be invoked as the current G set by an invocation sequence. Invocation sequences may be either locking or non-locking, that is, a locking sequence establishes a new current G set while a non-locking sequence establishes a G set for only a single reference. The default state has G0 as the current in-use G set. The G0, G1, G2 and G3 tables act as slots into which the tables of meanings may be designated. In the default state G0 contains the ASCII character code table, G1 the PDI code table, G2 the code table of supplementary accents and diacritical signs and G3 the mosaic table. A designation sequence is used to establish a new meaning to a code table slot. For example a designation sequence would be used to establish the table of MACRO PDIs as the G3 code table and an invocation sequence to make it the current in-use G set. Designation sequences must be registered with ISO.

The meaning of each of the code tables may be altered by "designating" a set of interpretations to it from the "Repertory". For example, an alternate text alphabet such as the Inuit symbols or the Greek letters could be designated as G0. A particular terminal would be able to accomplish this only if it contained that particular set of code table interpretations in its repertory. If a set is not in the repertory known to a particular terminal, the designation sequence has a reasonable default effect such as designating a blank code table.

It is the responsibility of the application to ensure (or assume) that sufficient capabilities have been implemented to handle its case. No defaults are guaranteed; however, careful design and judicious use of code extension in an application can ensure that default scenarios work. For example, it is suggested that in all applications the supplementary table of accents, diacritical marks and special characters be designated as a G2 set and be invoked by the non-locking invocation code SS2. This will ensure that if the application makes use of terminals which don't implement the supplementary set, the primary text will appear properly only without accents. For similar reasons, it is suggested that the primary ASCII text code table always appear as G0. It is also possible to designate entirely different functions to the code tables, such as designating the MACROs to G3 or the MOSAIC codes to G1 in order to facilitate interworking. However, the default designation positions should be used whenever possible.

In many simple applications there is never a need to make use of a designation sequence because the defaults for the service suffice.

The C0 set is always in use since it contains the code extension control codes and an alternate C0 set cannot be designated easily without altering the entire code extension structure. Since the C0 code specified in ASCII and the IRV differs from that recommended by CCITT for Videotex there is a need for some harmonization. Since only one C0 set can easily be accommodated within a terminal, the C0 set described in CCITT Recommendation S.100 has been incorporated in the Presentation Layer Protocol. The characters which differ are used quite rarely by conventional protocols so the Presentation Layer Protocol remains applicable to far greater than just Videotex services.

The ISO Standard 2022 for code extension has recently been revised to cover both eight and seven bit coding environments. While the seven bit environment can operate as a direct subset of the eight bit environment, the eight bit environment provides the additional capability of having two G sets as well as both the C0 and C1 sets in use simultaneously with added transmission efficiency. The eighth bit b8 is used to address either the right or left hand side of the expanded code table.

It is necessary to establish within the session level and transmission protocol whether the eighth bit will be used for parity error checking or for extension to another code table. In other words, the choice of the eight bit or seven bit code environment at the presentation level is established by the lower layers of protocol for a particular service, or by prior arrangement. A terminal manufacturer should accommodate both 8-bit and 7-bit presentation level coding, and can do this easily because seven level coding is a subset of eight level coding and is therefore always supported.

Parity or other forms of basic error protection are a level 2 data link layer function. Odd parity is required in the Broadcast Videotex (Teletext) transmission technique in order to guarantee data bit transmissions which help to maintain synchronization in certain classes of decoders. In order to maintain compatibility, odd parity should be used in those Interactive Videotex scenarios which make use of parity error protection.

The following subsection describes the code extension structure in more detail.

3.2.1 Code Extension

This section presents the complete code extension structure within which the various code tables operate. The techniques of the ISO Standard 2022 on code extension are employed; and the reader is referred to that standard for additional details which may be required to interwork with non Videotex-like applications. The ISO work item on code extension is still open; however, a complete set of invocation and designation sequences have been established for this service consistant with the best available knowledge of the current work in ISO. Designation sequences for code tables must go through a lengthy registration procedure. Since the sequences defined in this document have been chosen carefully, they may be used with some confidence for the establishment of introductory services.

The coding scheme is described below for both the seven and eight bit environment.

In both cases, data codes are formatted into 32-character control (C) sets and 96-character graphic (G) sets. The contents of these sets apply to both the 7-bit and 8-bit modes without modification. These sets are manipulated, for the purpose of providing a virtual address space larger than the 128 or 256 characters available in a 7-bit or 8-bit code, via the code extension techniques defined below.

3.2.1.1 Code extension in a 7-bit environment

Figure 3.9 illustrates the 7 bit code table. 128 character positions are available at one time organized into a C (control) and a G set. The G set is by default G0, which contains as its default the ASCII text characters. The C set is C0 containing the harmonized CCITT C0 control functions discussed earlier. The designation of other control sets to C0 is defined in the ISO Standard 2022, but its use in Videotex and related applications is left for further study.

The in-use G set may be selected from any of the four active code tables G0, G1, G2 and G3. If there is a requirement to use more code tables than those provided by the default states of G0, G1, G2 and G3 it is necessary to designate these code tables to G0, G1, G2 or G3 by means of the appropriate escape sequences. Each subsequent use of a shift function invokes the corresponding currently designated set.

A single C1 set is described for this Presentation Layer Protocol, however ISO 2022 provides the capability of designating alternate C1 sets. The use of this capability especially with respect to interworking with other services is left for further study. The designation sequences take the form ESC, (I), (F) where I is the intermediate character and F is the final character. The intermediate character determines which set is to be changed (re-designated). The table below shows the assignment of I characters to the sets to be designated.

Intermediate character	Set to be desiganted
2/1	C0
2/2	C1
2/8 or 2/12	G0
2/9 or 2/13	G1
2/10 or 2/14	G2
2/11 or 2/15	G3

Note that there are two I characters that will result in the re-designation of each of the four G sets. Only the first in each pair is defined for the designation presently used. The remaining I character is reserved by the ISO registration body to enlarge the number of sets which may be registered.

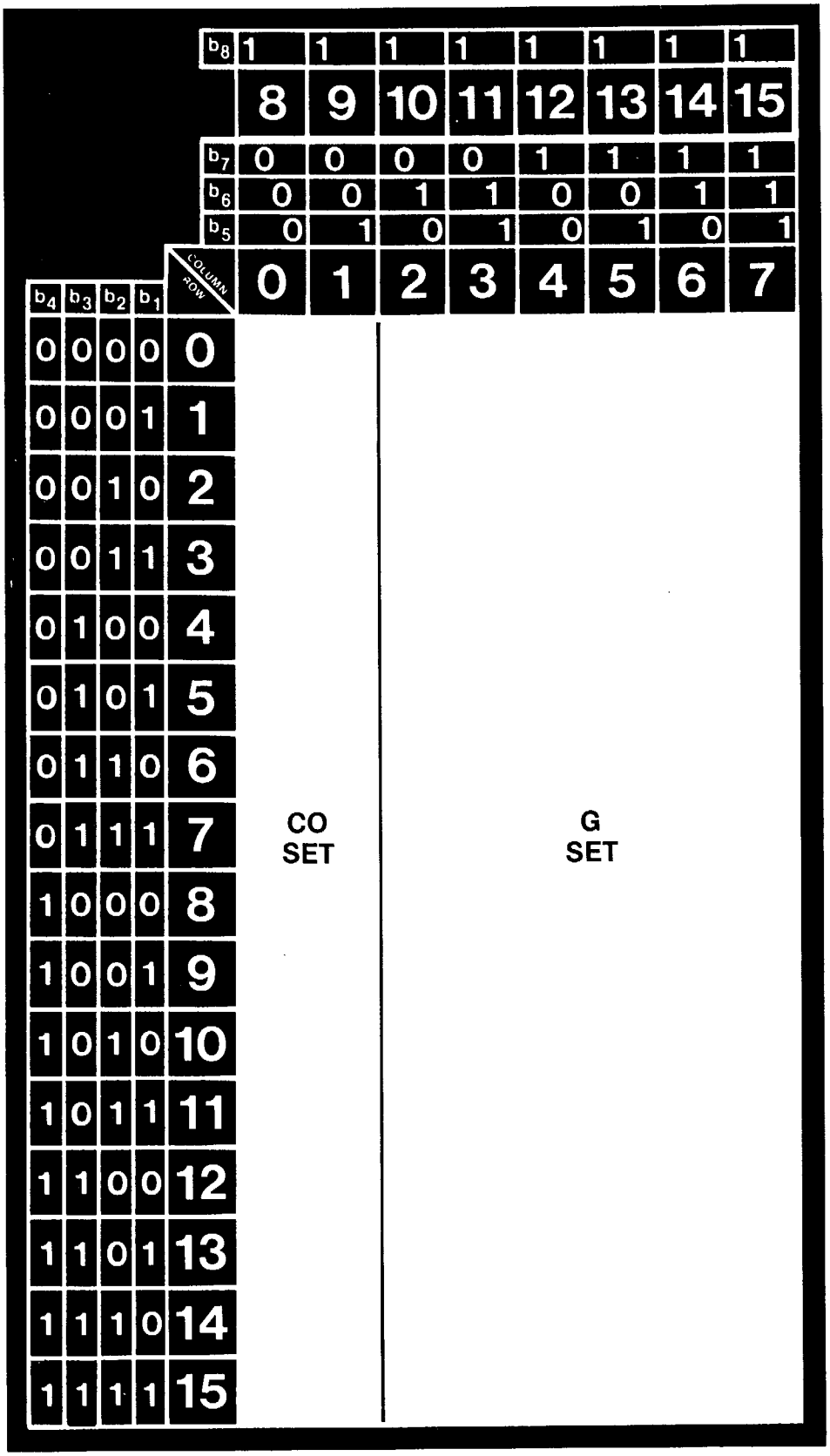


Figure 3.9 7-bit in-use table

The table below describes the final F characters to be used with designation sequences for this service. These characters are either registered through ISO or proposed to be registered as such.

Table 3.1
Final Character (F) Assignments

Graphic Set	(F)
ASCII Alpha-numerics	4/2
Mosaics	7/13
Supplementary Graphic Characters	7/12
DRCS	7/11
PDI Code Set	5/7
Macro-PDI Name Set	7/10

Figure 3.10 illustrates all the invocation and designation functions for a seven bit environment as described in ISO 2022.2 (as per May 1981). The shift functions used in this Presentation Layer Protocol have the following coding:

Mnemonic	Name	Code
SI	Locking Shift 0	SI (0/14)
SO	Locking Shift 1	SO (0/15)
LS2	Locking Shift 2	ESC (1/11), 6/14
LS3	Locking Shift 3	ESC (1/11), 6/15
SS2	Single Shift 2	SS2 (1/9)
SS3	Single Shift 3	SS3 (1/13)

The locking shift functions SI, SO, LS2 and LS3 invoke the current G0, G1, G2 or G3 set respectively into the in-use table. The single shift functions SS2 and SS3 are used to invoke the G2 or G3 set respectively in a non-locking manner. The single shift invocations alter the meaning of the immediately following bit combination (byte) only and ascribes to it the meaning of the corresponding bit combination of the G2 or G3 set.

In some situations where an alternate C0 set is used, the non-locking shift codes SS2 and SS3 will not be available. ISO provides alternate coding for the non-locking shift functions as specified in ISO 6429; however, consideration of this point is beyond the scope of this document.

Note that the PDI code set can only be single shifted into the in-use table in those cases where the PDI command is not to be followed by an associated numeric operand. Characters from the C1 set are accessed via two character escape sequences of the form ESC, (F), where F is the desired character from the C1 set. The character F must have a bit combination corresponding to column 4 or 5 of the 7-bit code table. The in-use table automatically reverts to its former state after the C1 command is executed. (Note that although the C1 controls consist of single characters, some commands may initiate multiple byte operations.) If any of the G sets are re-designated via a three character escape sequence while it is in the in-use table, the new code interpretations are simultaneously invoked; that is, a locking shift is not required for the change to take effect.

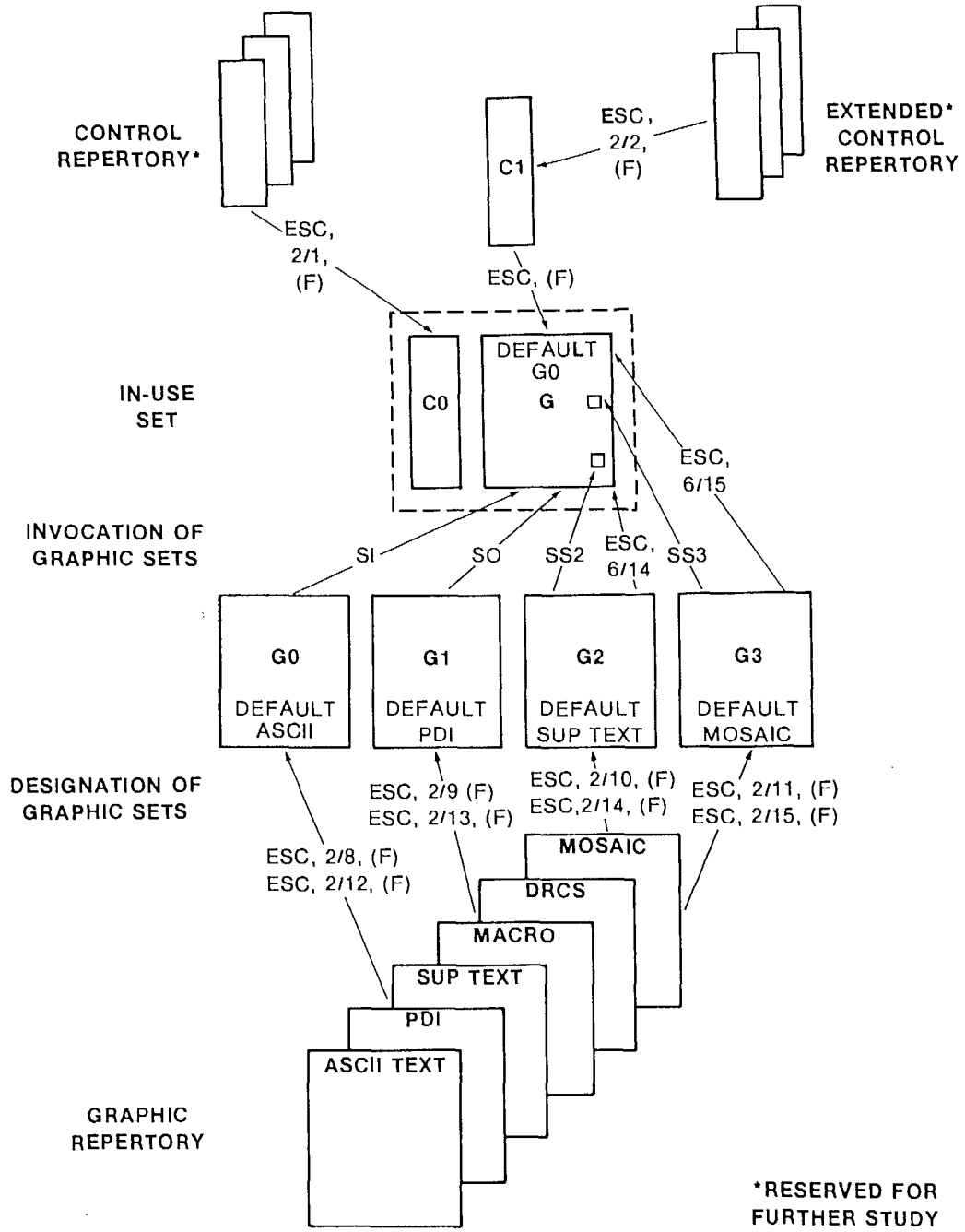


Figure 3.10 Code Extension in a 7-Bit Environment

Upon initialization, the ASCII alphanumerics are designated as the G0 set and the G0 set is invoked into the in-use table. The PDI codes are designated as the G1 set, the Supplementary Graphics Characters are designated as the G2 set, and the Mosaics are designated as the G3 set.

3.2.1.2 Code extension in an 8-bit environment

An 8 bit operating environment is obtained by using one additional bit of each byte of the combinations of the 7-bit code environment. The characters of the 7-bit code environment are assigned to the 128 bit combinations the eighth bit of which is ZERO. In this way, the 7-bit environment forms a defined and integral subset of the 8-bit environment. The 128 additional bit combinations, the eighth bit of which is equal to ONE, form the remainder of the 8-bit environment. All of the designation and invocation sequences in a 7-bit environment may operate in an 8-bit environment including the invocation sequence for the C1 set; however the 8-bit environment provides additional designation and invocation sequences. The 256 character in-use table is defined as shown in Figure 3.11. The table itself is organized into sixteen columns of sixteen rows, with bits 1 through 4 defining the row number and bits 5 through 8 defining the column number. The in-use table always contains the C0 set in columns 0 and 1, and the C1 set in columns 8 and 9. Columns 2 through 7, which by convention will be called the GL (G-left) area, can accommodate any of the four active G sets (G0, G1, G2, G3). Columns 10 through 15, which will be called the GR (G-right) area, can accommodate the G1, G2, or G3 sets.

The same designation sequences defined in an 7-bit environment apply to an 8-bit environment, however, the available shift sequences are increased to include invocations to the right hand side of the in use table. These additional invocation functions have the following coded representations:

Mnemonic	Name	Code
LS1R	Locking Shift 1 Right	ESC (1/11), 6/11
LS2R	Locking Shift 2 Right	ESC (1/11), 6/12
LS3R	Locking Shift 3 Right	ESC (1/11), 6/13

The SI character is used to invoke, in a locking manner, the G0 set into GL. Note that G0 cannot be invoked into GR. Since ASCII Text characters by convention and by default are designated as G0, this ensures that a very simple terminal can at least interpret text.

If any of the G sets are re-designated via a three character escape sequence while it is in the in-use table, the new interpretations are simultaneously invoked; that is, a locking shift is not required for the change to take effect.

Upon initialization, the ASCII alphanumerics are designated as the G0 set and the G0 set is invoked into GL. The PDI set is designated as the G1 set and the G1 set is invoked into GR. The Supplementary Graphics Characters are designated as G2 and the Mosaics are designated as G3.

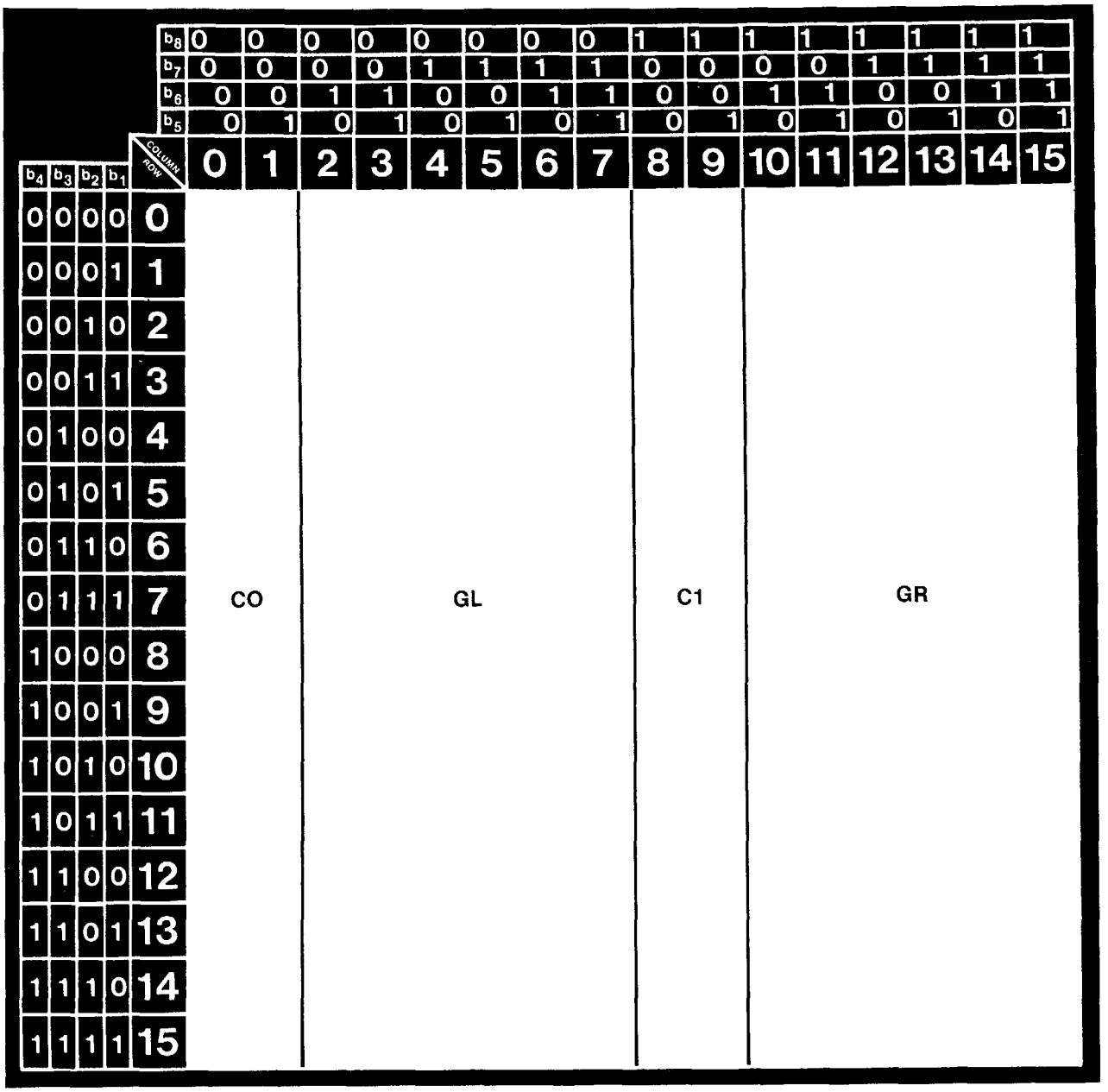


Figure 3.11 8-bit in-use table

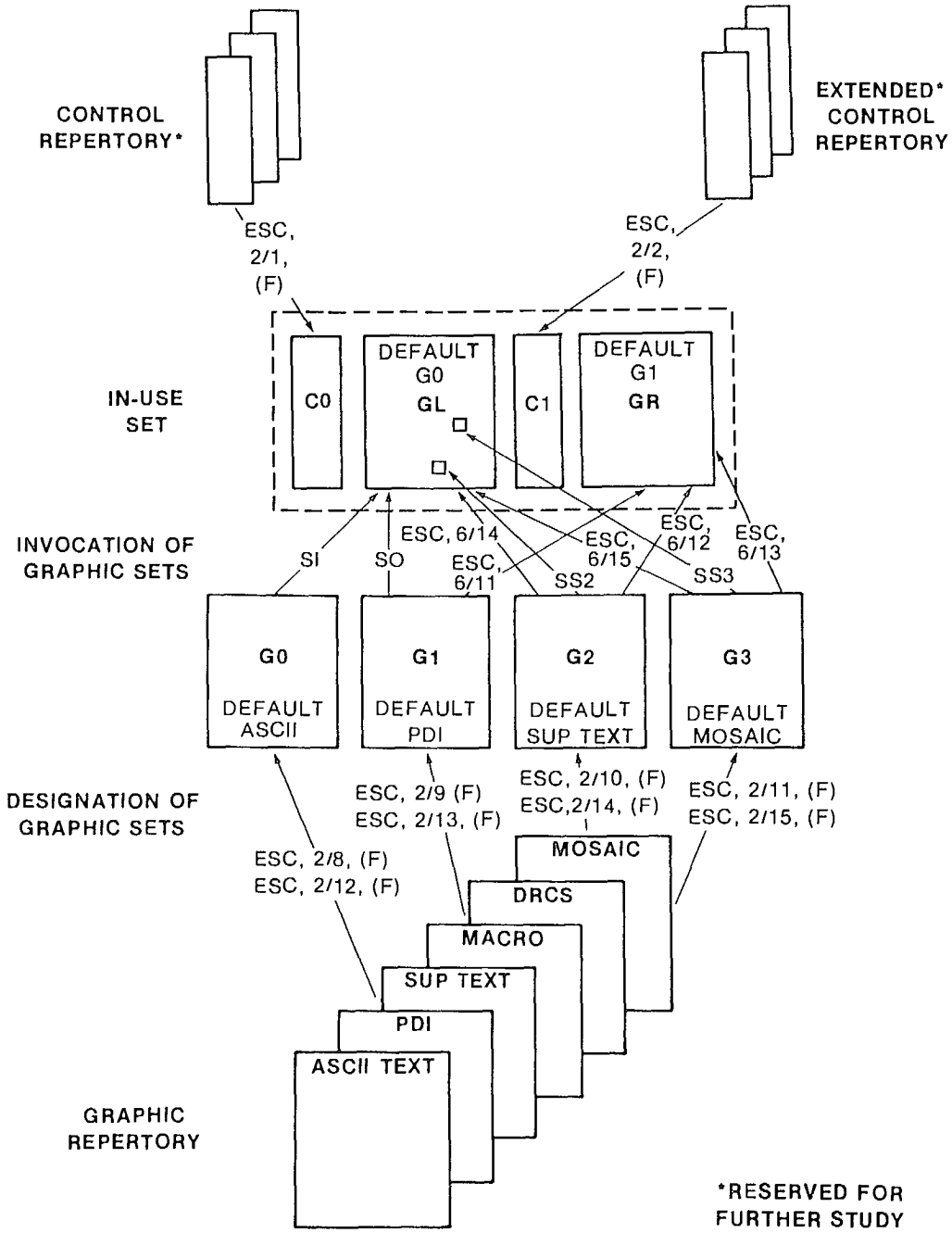


Figure 3.12 Code Extension in an 8-bit Environment

3.3 Display Considerations

The Presentation Layer Protocol defines a coding scheme which is independent of the hardware display apparatus used. The geometric drawing commands are defined in terms of an abstract unitary coordinate space, and the text oriented commands are defined in terms of a variable character size which permits any number of characters to be displayed on the screen. Practically, various types of display apparatus have different capabilities which affect the displayed picture. In order to provide the terminal manufacturer with the flexibility to build a terminal in the most cost effective manner, certain freedoms and defaults have been deliberately defined within the PLP.

3.3.1 Overlaying

There is no special positional dependence upon the order in which drawing primitives are presented in the geometric coding approach. Pictures coded with PDIs are built up out of a sequence of drawing commands with each geometric primitive superimposed over previous ones. For example, a RECTANGLE command may be drawn to display an area the full size of the screen as a background. A subsequent POLYGON command would draw over the background RECTANGLE and a subsequent LINE command could overwrite a portion of the POLYGON. In this manner, pictures are built up in layers out of commands from the geometric primitive drawing set.

Manufacturers are free to implement terminal equipment by whatever technologies they deem economical. This includes the freedom to implement the G1 graphics drawing display hardware and the G0 and G2 character drawing hardware by different technologies. Because of this freedom the interaction of layering between characters and graphics primitives is deliberately undefined.

The registration between alphanumeric text, geometric picture, and photographic pictures is not guaranteed to be exact, nor is the superposition of these drawing modes defined. This permits a manufacturer to build an overlaying character hardware generator or to implement characters by software. When a manufacturer varies from absolute registration it is desirable that he minimizes the distance between the physical image of each character and its defined position within the unit screen.

By not strictly specifying the registration or superposition of characters or of incrementally defined pictures to the geometrically drawn picture, the manufacturer gains certain freedoms in the construction of a terminal. For example, a manufacturer might build a low resolution terminal with 128 pixels of resolution in width. At this resolution it is impossible to define characters within a bit plane memory, so a hardware character generator would be required. This character generator output would be mixed with the geometric picture, the characters in superposition overlaying the geometric picture.

3.3.2 Text Sizes

Most home television sets have significant overscan which means that only a portion of the display raster area is visible. Several studies have been performed, particularly by SMPTE (Society for Motion Picture and Television Engineers) of the overscan on home television receivers (see ref. 18 and 19).

According to these SMPTE studies, it is estimated that 10% margins are required on each side of the television raster area to accommodate overscan. This provides a rather severe constraint on displaying textual information. In order to achieve 40 characters in a row, close spacing between characters is required. Since the publication of the SMPTE studies, "ultra rectangular" television screens have become common in home televisions. This marginally tends to relax the overscan constraint, but there have been no recent comprehensive studies to determine what the overscan limits are on the current population of television equipment. Experience with field equipment indicates that terminal manufacturers should design within 10% overscan boundaries.

Extensive psychological investigations have been carried out in Canada to determine the readability of various character densities. The display of 40 characters per row is quite acceptable with 5% margins, although with 10% margins, 40 characters per row is somewhat cramped. In the vertical direction, 20 rows of characters can be displayed with adequate inter row spacing, providing a minimum space for the descender portion of a character. These investigations have shown that readability is an important parameter. It is also important in Canada to leave sufficient room for accents. A person reads by recognizing words only if the customary visual clues such as descending characters are present. If text is crowded too close together then a person reads slowly by deciphering words character by character. Due to spacing and readability considerations, the basic default character density for use on North American 525 line television systems is 40 characters per row and 20 rows per screen within the SMPTE safe title area (of 10% overscan margins on all sides of the television picture). A 21st row (at the top of the screen) may be transmitted to a terminal, for use such as presenting less essential service information, but it cannot be guaranteed to be visible on all television sets.

On European 625 line television systems 40 characters per row and 24 rows per screen with an optional 25th sevice row can be displayed within the television set overscan limits. This same number of rows can also be displayed on a North American 525 line studio or commercial display monitor which does not require overscan margins, and 80 characters per row can also be presented on some such monitors. Essentially the conventional business terminal character density of 80 characters per row and 24 rows per screen can be provided on a PLP terminal driving a higher grade monitor.

If a terminal is constructed so that an NTSC composite video colour signal at baseband or RF is used to interface a television display monitor, then the readability of the text is further degraded. Only 32 characters per row are clearly visible and a character density of 32 characters per row and 16 rows per screen should be considered using the variable text size capability of the Presentation Level Protocol. By the combination of using specialized generation circuitry and the establishment of good practice guidelines which limit the colour combinations which may be used by an information provider, 40 characters by 20 rows can be made minimally acceptable.

4. DETAILED DESCRIPTION

This section provides a detailed description of the Presentation Level Protocol in a narrative form. Figure 4.1 illustrates the drawing and control commands available in the geometric mode. The Geometric mode, the Mosaic mode, the DRCS and MACRO capabilities as well as the C0 and C1 control sets are described, however the basic text and code extension capabilities covered in section 3 are not presented.

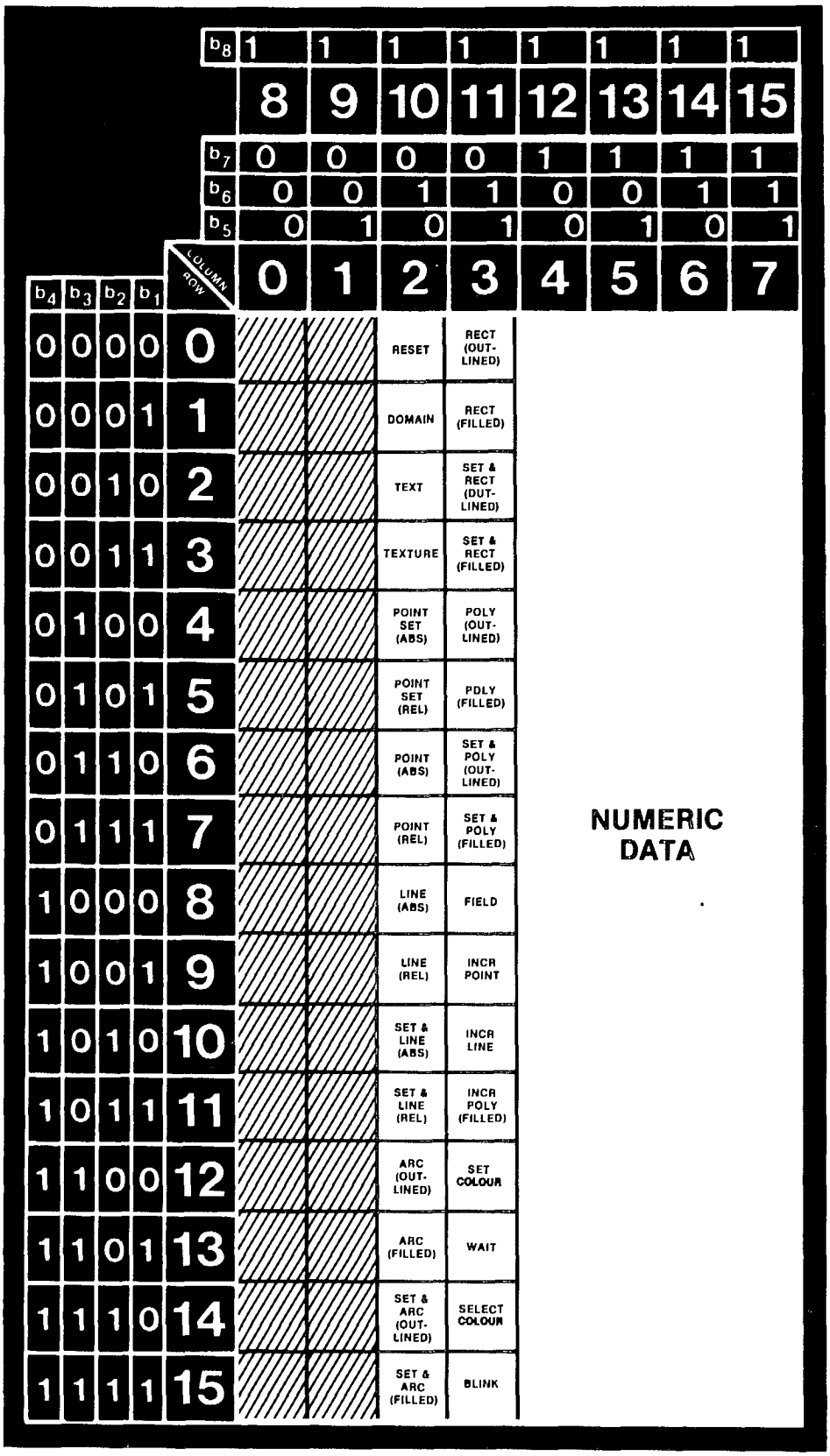


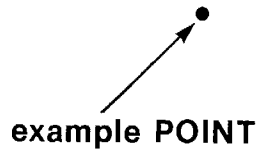
Figure 4.1 PDI Code Set

4.1 Picture Descriptors

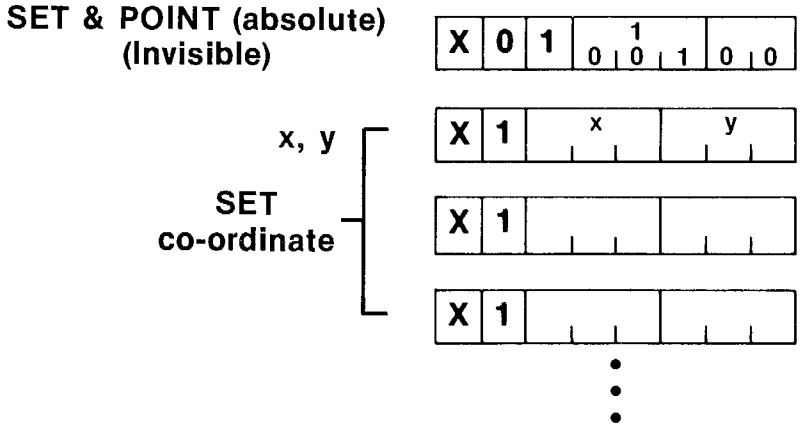
4.1.1 POINT

The POINT opcode is used to perform the two most basic geometric drawing operations, that of establishing the coordinate at which to commence drawing and that of drawing a point. An (x,y) coordinate pair must always be specified with this command to set the drawing position (SET). Optionally, a dot may be drawn (i.e. made visible) at the specified x,y coordinate position (POINT). The (x,y) coordinate may either be specified as an absolute position or as a relative displacement from the previous drawing position.

A series of coordinate positions following a POINT opcode may be used to draw a point by point graph.

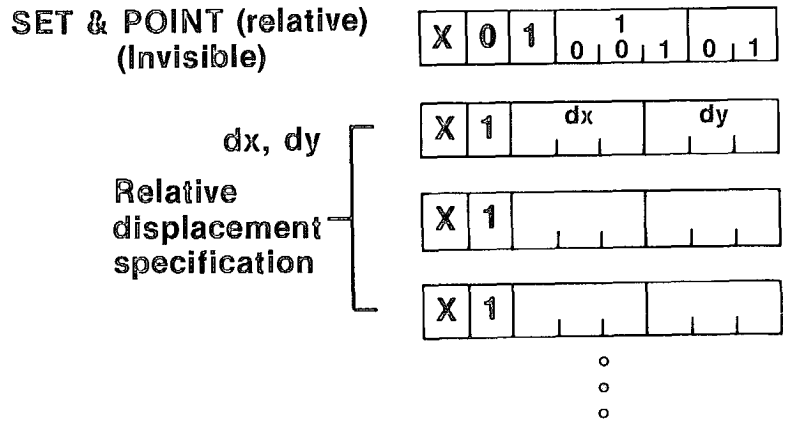


4.1.1.1 SET & POINT (absolute, invisible)



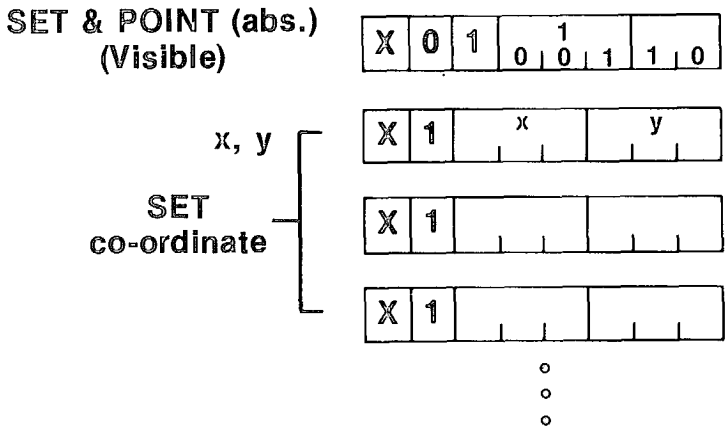
This opcode sets the drawing position to the absolute coordinates specified (x,y). A dot is not drawn.

4.1.1.2 SET & POINT (Relative, Invisible)



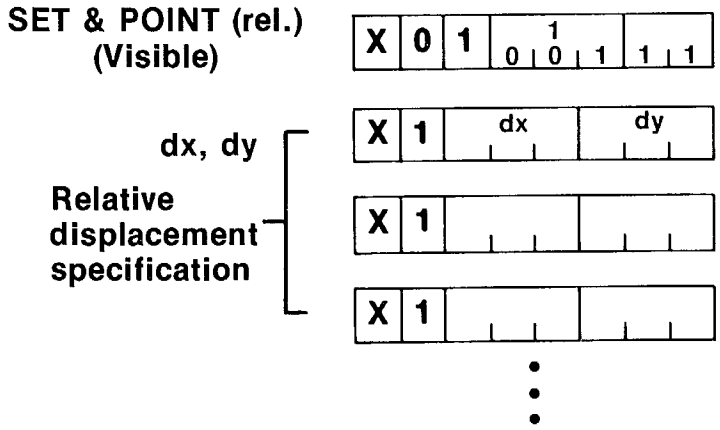
This opcode sets the drawing position to the coordinates obtained by adding the displacement specified to the current drawing position. A dot is not drawn.

4.1.1.3 SET & POINT (absolute, visible)



This opcode sets the drawing position to the absolute coordinate specified, (x,y) and draws a dot at that point whose size is determined by the logical PEL size, and whose colour is determined by the in-use colour. (See section 4.2.2 for a description of logical pel).

4.1.1.4 SET & POINT (relative, visible)

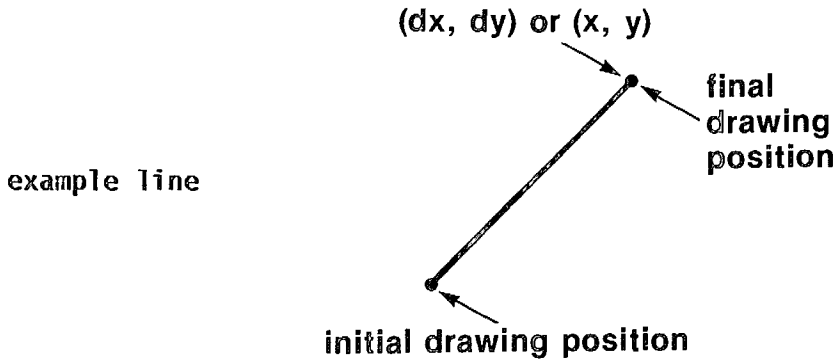


This opcode sets the drawing position to the coordinates obtained by adding the displacement specified (dx,dy) to the current drawing position, and draws a dot at that point whose size is determined by the logical pel size, and whose colour is determined by the in-use colour.

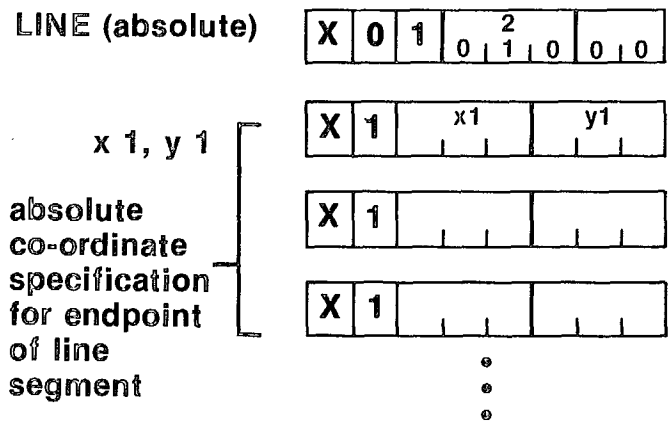
4.1.2 LINE

The second most basic geometric drawing operation is the LINE opcode. The direction and length of a line is specified by the endpoints. The initial drawing position of a line drawing operation may be either explicitly specified within the LINE opcode or interpreted as being the final position of the previous drawing opcode. The final drawing position for a line segment may be either specified as a relative displacement from the initial position or as an absolute (x,y) co-ordinate. The line is drawn in the in-use colour(s), has a width which is determined by the logical pel size, and a texture determined by the line current texture attribute.

The LINE opcode may be used to draw a line graph from a table of numbers described as absolute or relative coordinates in the same manner as the POINT opcode.

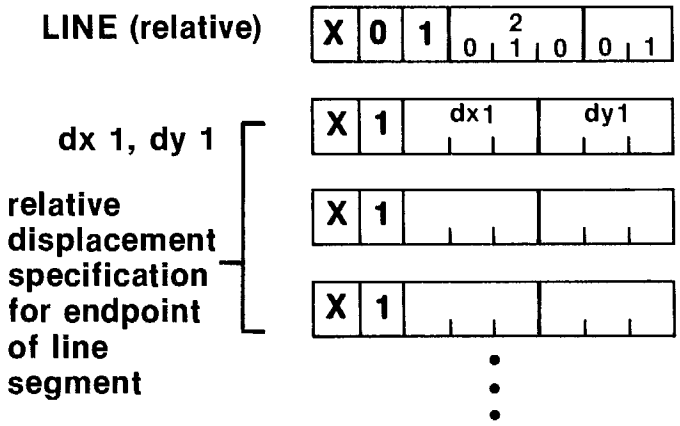


4.1.2.1 LINE (absolute)



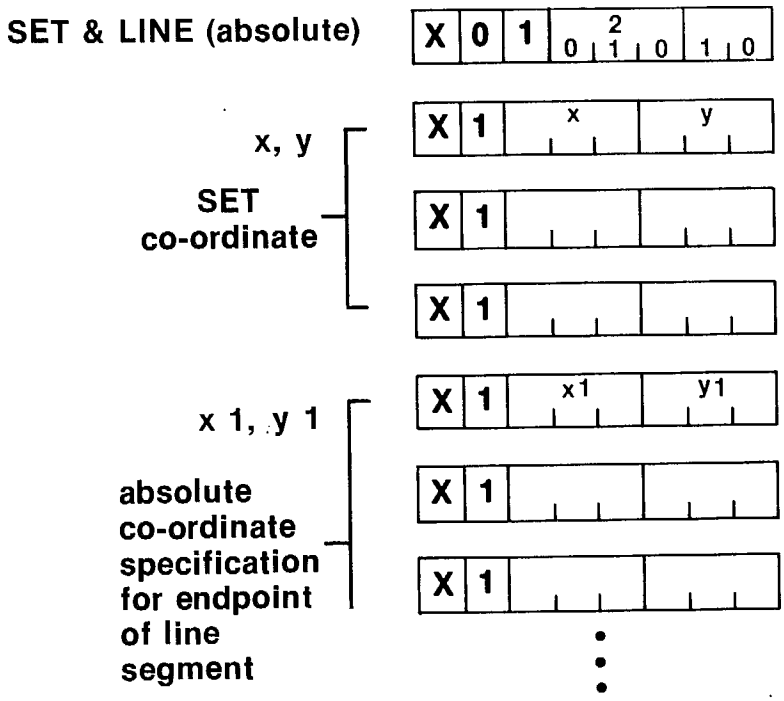
This opcode draws a line between the current drawing point and the endpoint which is specified in absolute coordinate (x,y).

4.1.2.2 LINE (relative)



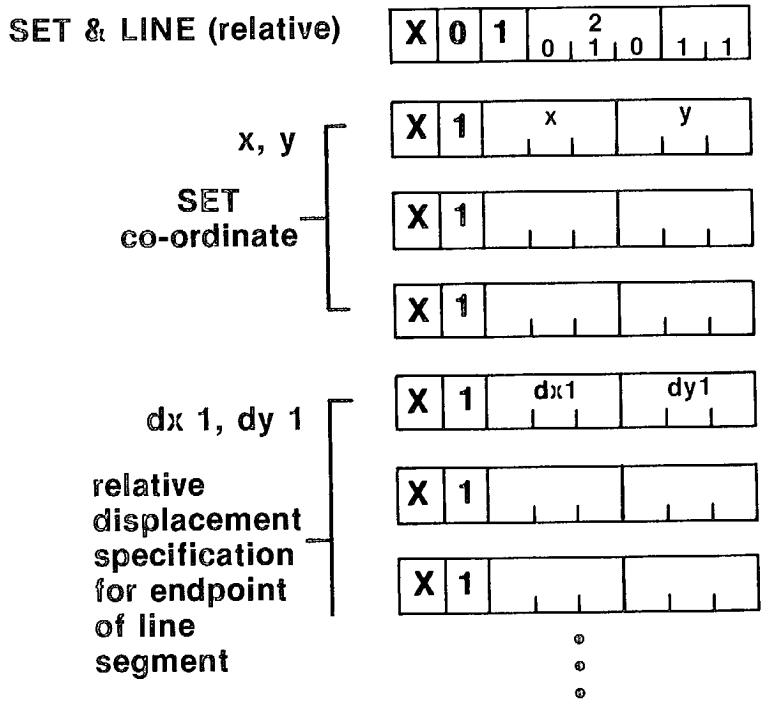
This opcode draws a line between the current drawing point and the endpoint which is specified as a relative displacemnt (dx, dy) to the current drawing point.

4.1.2.3 SET & LINE (absolute)



This opcode draws a line between the first set of coordinate (x,y) and the second set (x1, y1) both of which are specified in absolute coordinates.

4.1.2.4 SET & LINE (relative)



This opcode draws a line between the first set of coordinates (x,y) which is specified absolutely, and the final position which is specified as a relative displacement (dx1, dy1) to the first set of coordinates.

4.1.3 ARC

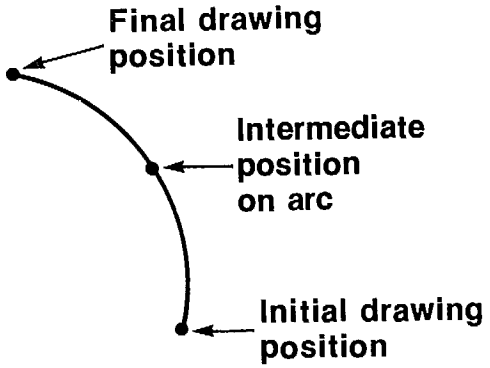
The ARC geometric drawing operation provides the capability of drawing circles, segments of circles and curvilinear splines. An arc is drawn from an initial drawing position to a final drawing position through an intermediate point on the arc. The initial drawing position may be either explicitly specified within the ARC opcode or interpreted as being the final position of the previous drawing opcode. The intermediate position on the arc is described as a relative displacement from the initial drawing position. The final drawing position is also specified as a relative displacement from the intermediate position on the arc. It is good practice, in order to minimize error, to always specify the intermediate point on the arc as being approximately midway between the start and end points.

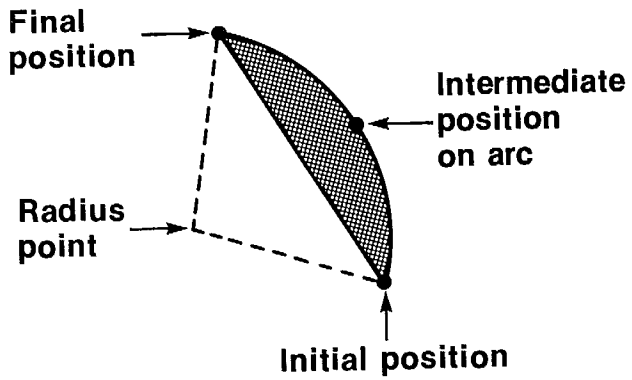
The current drawing position at the completion of drawing an arc is the endpoint specified. Drawing a circle results when the start and endpoint are coincident. For the definition of a circle, the point on the arc defines the diameter of the circle and therefore is the midpoint between the start and endpoint. If the three drawing points are co-linear, a line or lines result from the start to the endpoint through the other point. If the endpoint is omitted, it is taken to be coincident with the start point and a circle is drawn.

If more data is given after the endpoint specification, a curvilinear spline is drawn through the points specified. The exact algorithm for the curvilinear spline is reserved for future study.

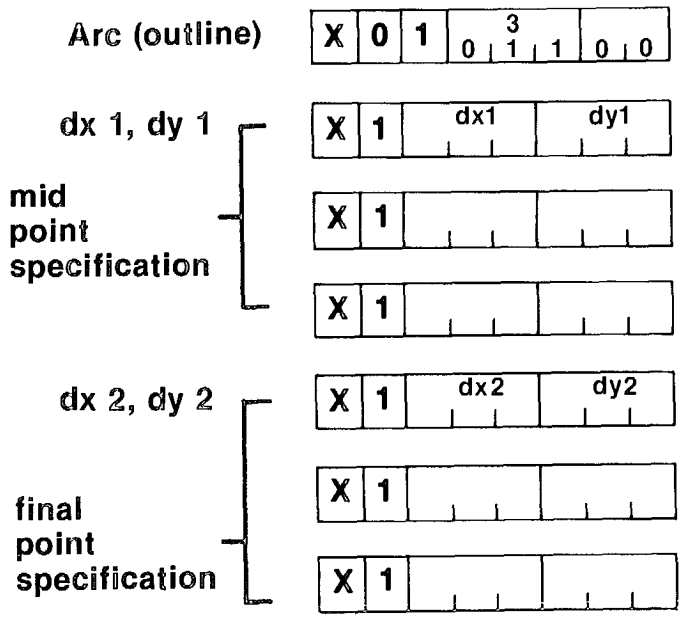
The arc is drawn in a line which has a texture given by the current line texture attribute (see Section 4.2.4).

The area enclosed by an ARC opcode may be filled in the texture pattern as defined by the current texture attribute. The form of the filled-in area is the area enclosed by the ARC command and the chord joining the two endpoints of the arc. The chord is not considered a part of the arc and, as such, is not highlighted if highlight mode is selected (see section 4.2.4 "TEXTURE" for a discussion of highlight mode).





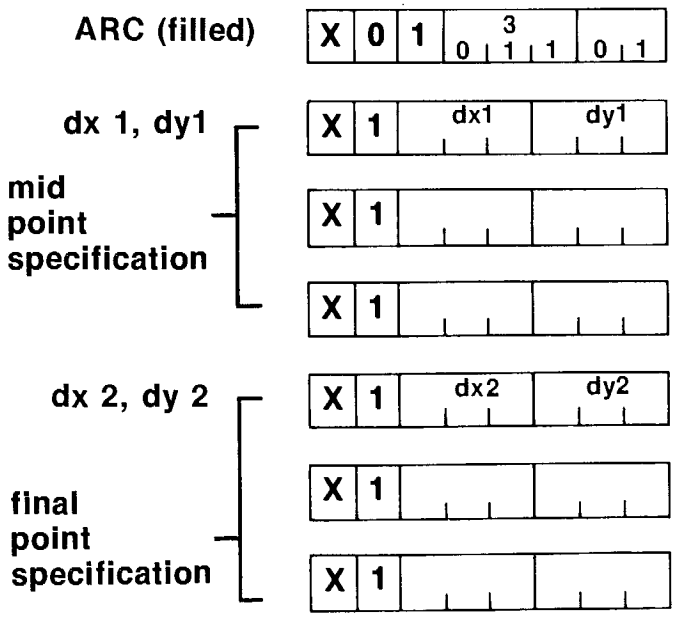
4.1.3.1 ARC (outline)



This opcode causes an arc to be drawn; the initial point position is the current drawing point, the intermediate point position is the first block of coordinate data specified as a relative displacement (dx1,dy1) from the initial point, and the final point position is the second block of coordinate data specified as a relative displacement (dx2,dy2) from the intermediate point.

The arc is not filled.

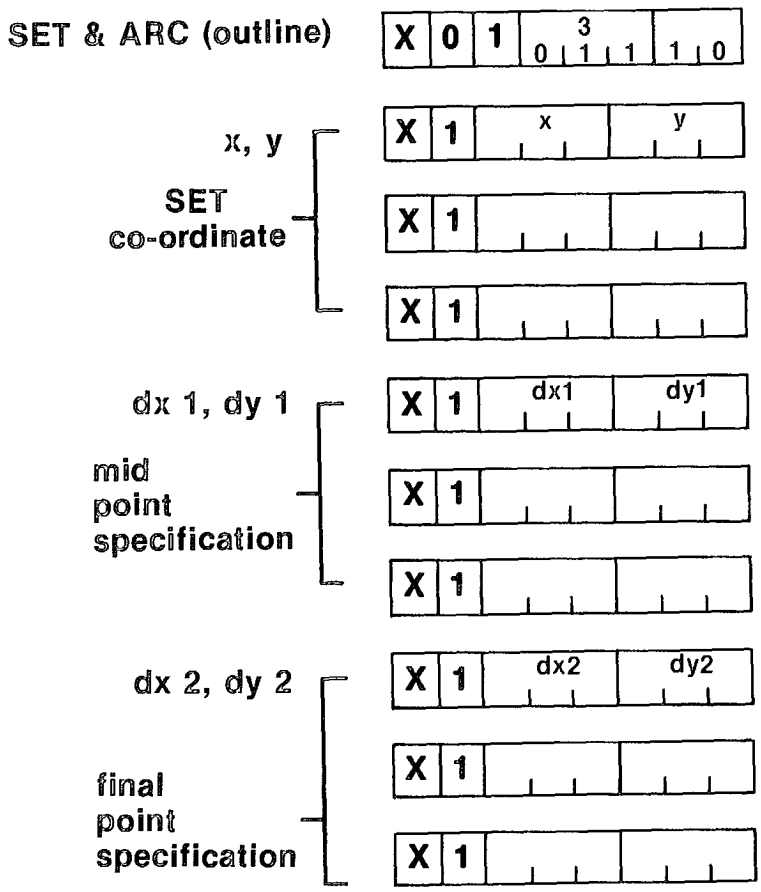
4.1.3.2 ARC (filled)



This opcode causes an arc to be drawn; the initial point position is the current drawing point, the intermediate point position is the first block of coordinate data specified as a relative displacement (dx1,dy1) from the initial point, and the final point position is the second block of coordinate data specified as a relative displacement (dx2,dy2) from the intermediate point.

The initial and final drawing positions are joined by a chord and the resulting figure is filled in the current colour(s) with the current texture pattern.

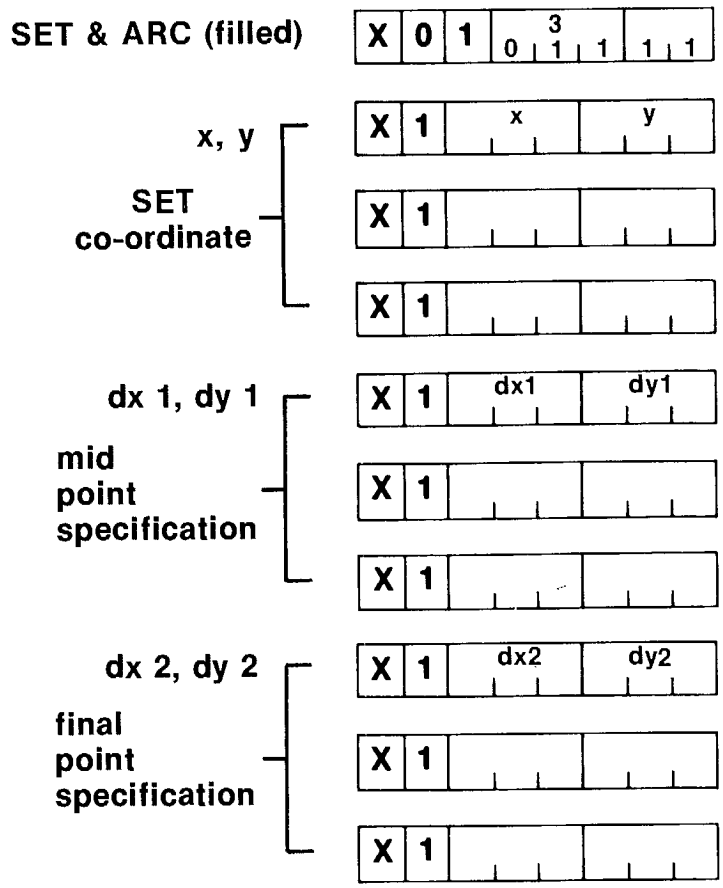
4.1.3.3 SET & ARC (outline)



This opcode causes an arc to be drawn; the initial point is the first block of coordinate data specified in absolute coordinates (x,y) the intermediate point position is the second block of coordinate data specified as a relative displacement (dx1, dy1) from the initial point and the final point position is the third block of coordinate data specified as a relative displacement (dx2,dy2) from the intermediate point.

The arc is not filled.

4.1.3.4 SET & ARC (filled)



This opcode causes an arc to be drawn; where the initial point is the first block of coordinate data specified in absolute coordinates (x,y), the intermediate point position is the second block of coordinate data specified as a relative displacement (dx1, dy1) from the initial point, and the final point position is the third block of coordinate data specified as a relative displacement (dx2,dy2) from the intermediate point.

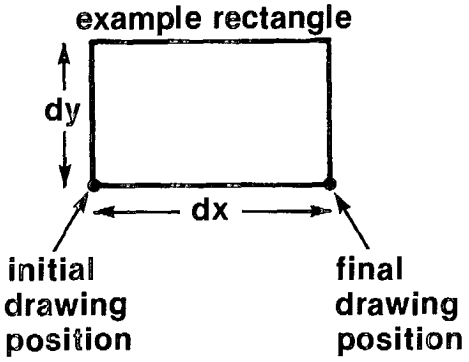
The initial and final drawing positions are joined by a chord and the resulting figure is filled in the current colour(s) with the current texture pattern.

4.1.4. RECTANGLE

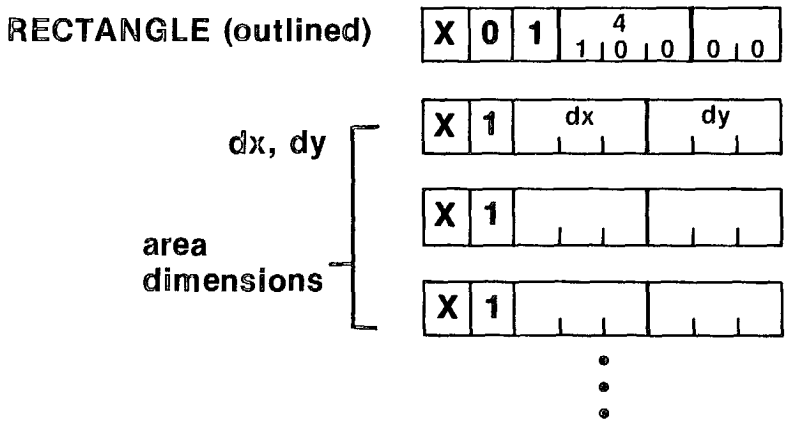
The RECTANGLE geometric drawing operation provides the capability of drawing a rectangular area of width dx and height dy. The initial drawing position of a RECTANGLE drawing operation may be either explicitly specifed within the RECTANGLE opcode or interpreted as being the final position of the previous drawing opcode. The final drawing position of a RECTANGLE opcode is the initial drawing position altered in x only, by the amount of the dx displacement.

A RECTANGLE may be either filled or outlined. For outlined RECTANGLES, the line texture which is specified by the TEXTURE command applies. For filled RECTANGLES, the area is filled in the current colour(s) with the texture pattern specified in the TEXTURE command, and the perimeter may be optionally highlighted.

The RECTANGLE opcode may be used to draw a histogram from a table of numbers representing relative dy and dx displacements in the same manner as the POINT and LINE opcode graph plot mode.



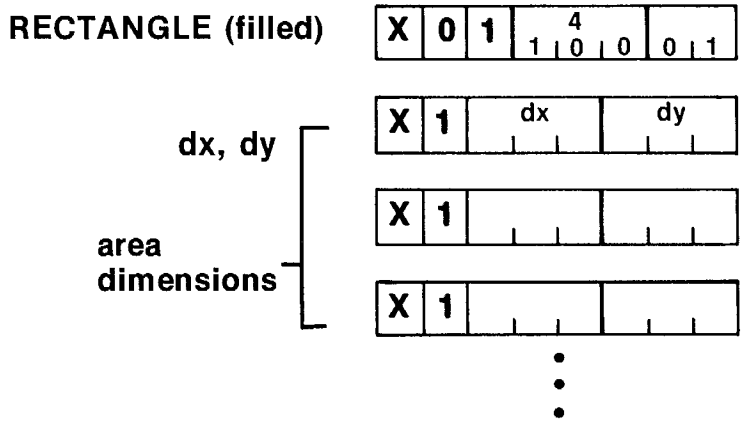
4.1.4.1 RECTANGLE (outlined)



This opcode causes a rectangle to be drawn; the initial drawing position is the current drawing point and the width and height (dx,dy) are given as the first block of coordinate data.

The rectangle is not filled.

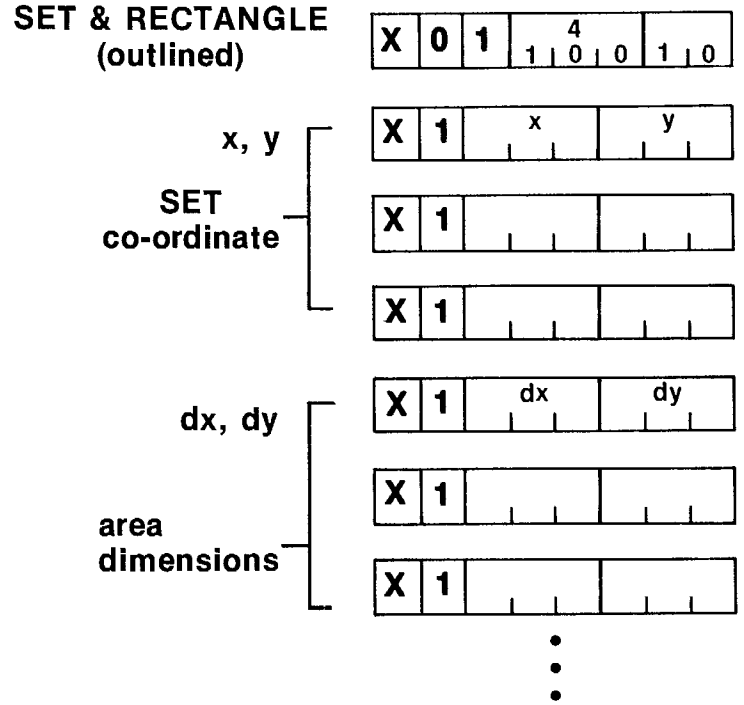
4.1.4.2 RECTANGLE (filled)



This opcode causes a rectangle to be drawn; the initial drawing position is the current drawing point and the width and height (dx,dy) are given as the first block or coordinate data.

The rectangle is filled in the current colour(s) with the current texture pattern.

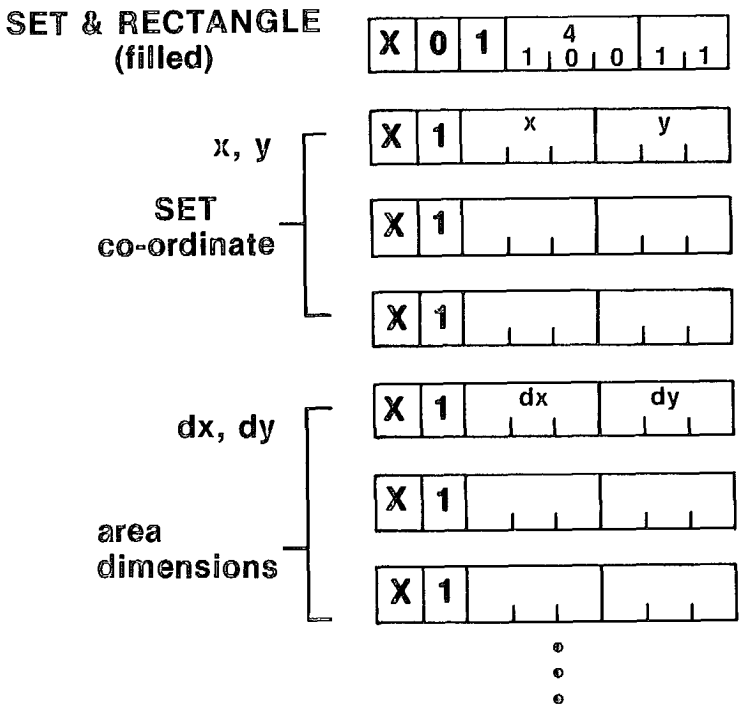
4.1.4.3 SET & RECTANGLE (outlined)



This opcode causes a rectangle to be drawn; the initial drawing position is specified in absolute coordinates (x,y) as the first block of coordinate data, and the width and height (dx,dy) are given as the second block of coordinate data.

The rectangle is not filled.

4.1.4.4 SET & RECTANGLE (filled)



This opcode causes a rectangle to be drawn; the initial drawing position is specified in absolute coordinates (x,y) as the first block of coordinate data, and the width and height (dx,dy) are given as the second block of coordinate data.

The rectangle is filled in the current colour(s) with the current texture pattern.

4.1.5 POLYGON

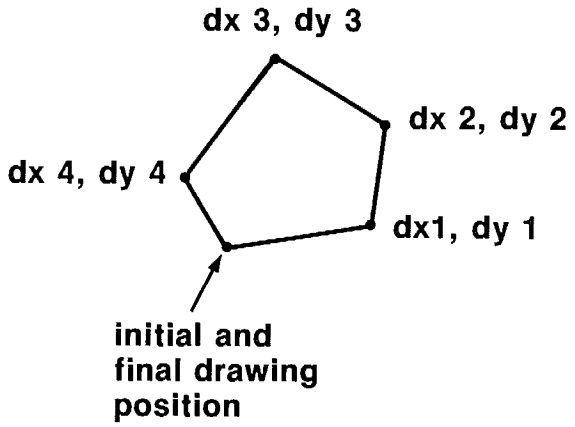
The POLYGON geometric drawing operation provides the capability of drawing a general polygonal area with the specified vertices. A POLYGON is specified as a series of (x,y) coordinates of the vertices about the perimeter of the polygon. Each (dx,dy) coordinate pair represents a relative displacement from the last vertex (a relative displacement of magnitude 0 is ignored). There is implicit closure between the initial drawing position and the last vertex specified so that the final drawing position is identical with the initial drawing position.

A POLYGON must enclose a single area; that is, no line joining two consecutive vertices may cross any other line joining two consecutive vertices.

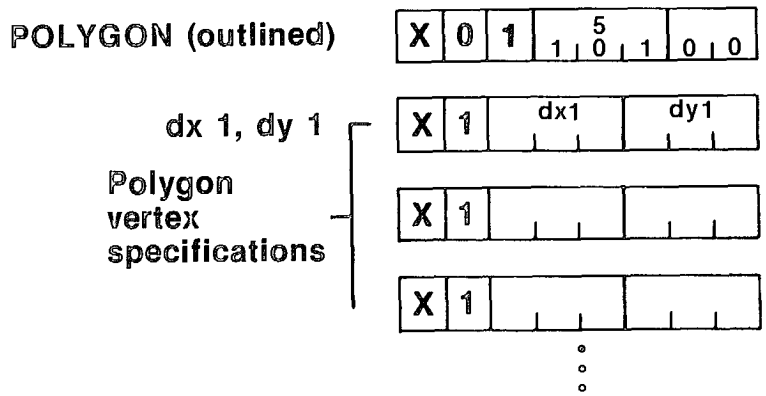
A polygon may be either filled or outlined. For outlined polygons, line texture is as specified in the TEXTURE PDI. For filled polygons, the area enclosed is filled in the current colour(s) with the texture pattern specified in the TEXTURE PDI, and the perimeter may be optionally highlighted.

The number of vertices describing a polygon is determined by the amount of data following the POLYGON opcode. The polygon is not drawn until all the vertices have been communicated to the terminal and a termination code such as another opcode or any other presentation level code not from the numeric data field of the PDI set has been received. The maximum number of vertices permitted to describe a polygon is terminal dependent.

Example Polygon



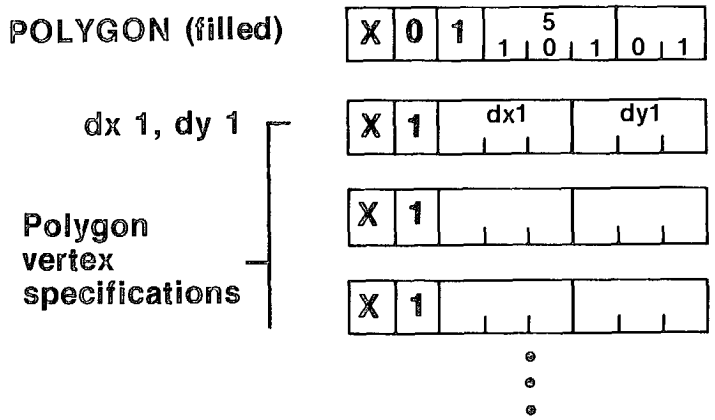
4.1.5.1 POLYGON (outlined)



This opcode causes a polygon to be drawn; the initial drawing position is the current drawing point and subsequent vertex coordinates are specified as relative displacements (dx1,dy1) from the previous vertex coordinate.

The polygon is not filled.

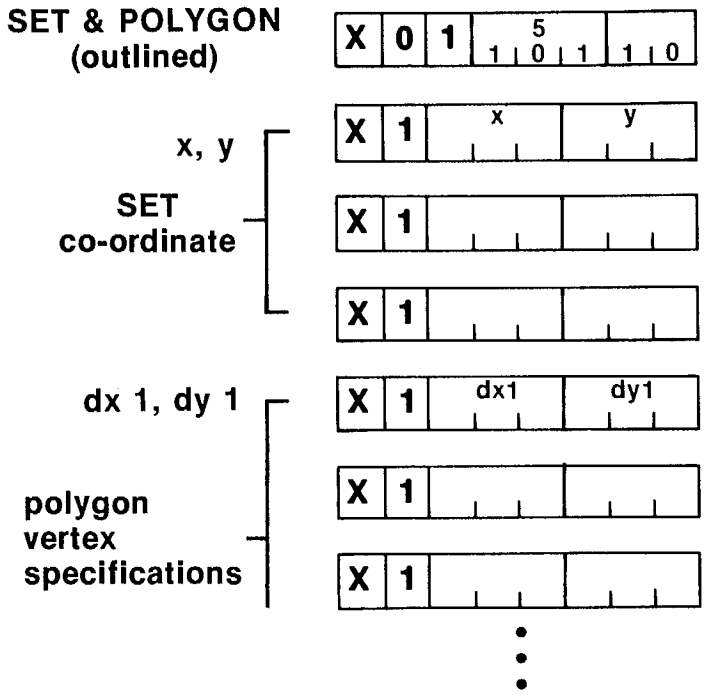
4.1.5.2 POLYGON (filled)



This opcode causes a polygon to be drawn; the initial drawing position is the current drawing point and subsequent vertex coordiantes are specified as relative displacements (dx1,dy1) from the previous vertex coordinate.

The polygon is filled in the current colour(s) with the current texture pattern.

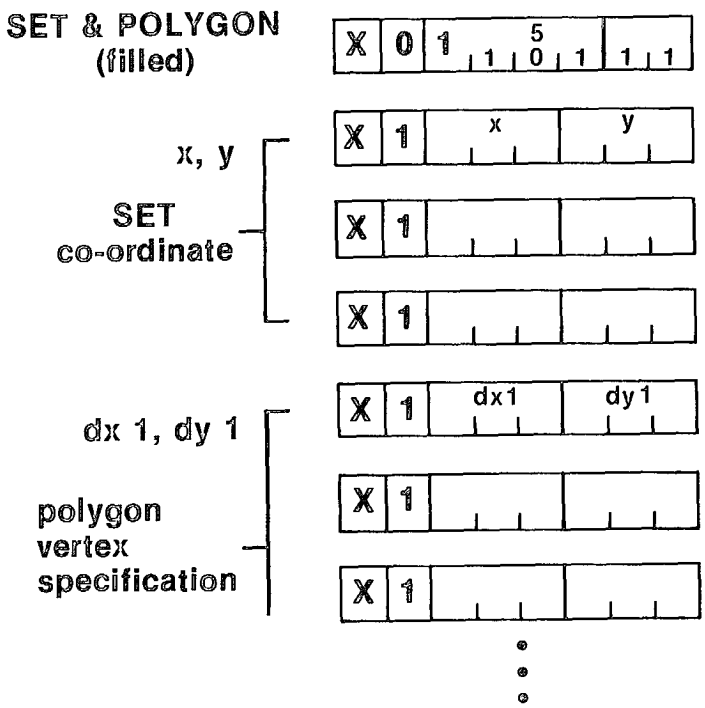
4.1.5.3 SET & POLYGON (outlined)



This opcode causes a polygon to be drawn; the initial drawing position is specified in absolute coordinates (x,y) as the first block of coordinate data, and subsequent vertex coordinates are specified as relative displacements (dx1,dy1) from the previous vertex coordinates.

The polygon is not filled.

4.1.5.4 SET & POLYGON (filled)



This opcode causes a polygon to be drawn; the initial drawing position is specified in absolute coordinates (x,y) as the first block of coordinate data, and subsequent vertex coordinates are specified as relative displacements (dx1,dy1) from the previous vertex coordinates.

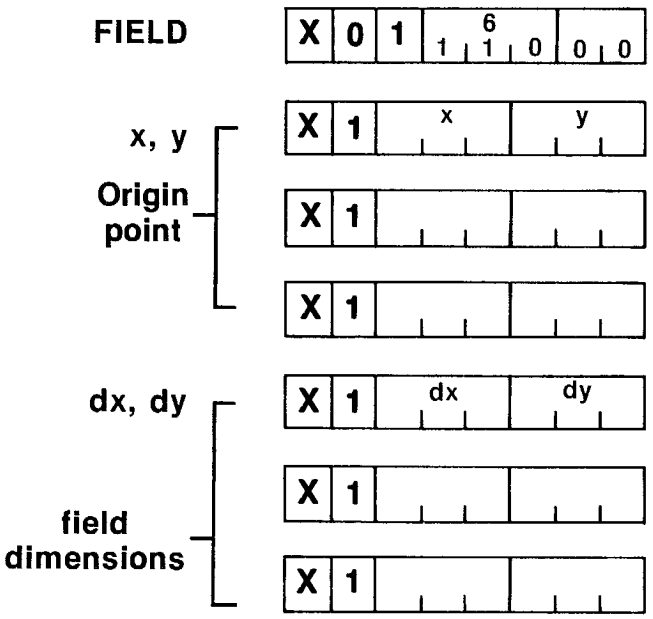
The polygon is filled in the current colour(s) with the current texture pattern.

4.1.6. INCREMENTAL

The INCREMENTAL opcodes allow for the specification of complex images in a compact manner. There are four INCREMENTAL opcodes namely FIELD, INCREMENTAL POINT, INCREMENTAL LINE and INCREMENTAL POLYGON (filled).

The image may be of a photographic nature (FIELD and INCREMENTAL POINT PDIs) or may consist of complex lines such as signatures (INCREMENTAL LINE), or filled polygons such as logos or other symbols (INCREMENTAL POLYGON).

4.1.6.1 FIELD



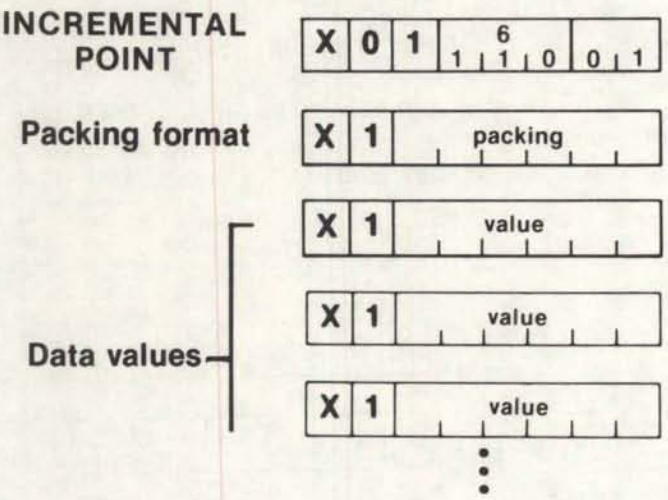
This opcode is used to define the active drawing area used by the INCREMENTAL POINT PDI (the active drawing area is also used for columnated text and for unprotected fields).

The origin point of the field is specified in absolute coordinates (x,y) as the first block of coordinate data. The next block of coordinate data gives the field dimensions, width and height (dx,dy).

The current drawing point is set to the origin of the field after the FIELD PDI has been executed.

If no data bytes follow the FIELD opcode, the active drawing area is set to the full unit screen.

4.1.6.2 INCREMENTAL POINT



The INCREMENTAL POINT opcode is a photographic drawing primitive which allows an image to be described in a point by point or otherwise encoded manner in a similar way to the operation of a facsimile machine. An INCREMENTAL POINT drawing command stores colour values in a field of logical pels. A typical hardware display apparatus consists of an array of pixels organized in some manner and of a definition which varies from low resolution such as 64 by 48 elements in the x and y dimensions respectively, to high resolution such as 1024 by 768 or greater. The organization of the pixel memory could be a character-oriented mosaic approach, a raster-organized bit-plane approach or other. The INCREMENTAL POINT opcode is set up in an independent manner from the particular hardware implementation methods and it is the responsibility of the implementation on each type of terminal to realize the picture described by the INCREMENTAL POINT opcode as best it can on a particular type of hardware with the condition that the picture must in all cases lie within the defined active drawing area (field).

Example INCREMENTAL POINT PICTURE

The first data byte following the opcode is the packing format, an unsigned integer that determines the number of consecutive bits to be taken from following data bytes to make up a single colour specification. A value of 0 is taken to mean 64 bits. A packing value of 1 is possible and will take on a colour value of green in colour mode 0 (see section 4.2.6 "SELECT COLOUR" for a discussion of colour mode). This is useful for use in DRCS downloading of shapes during which colour attributes are ignored.

The remaining data bytes contain the colour specifications, stored sequentially without regard to byte boundaries with the high-order bit b6 and the low-order bit b1. In colour mode 0, these colour specifications are treated as actual colours, organized in triplets in the order G, R, B, G, R, B... etc., where the first G, R, B, triplet gives the most significant bits of the colour specification. Note that it is possible to specify a packing format which is not an integer multiple of 3 in which case the three primary colours will not be specified to equal accuracy.

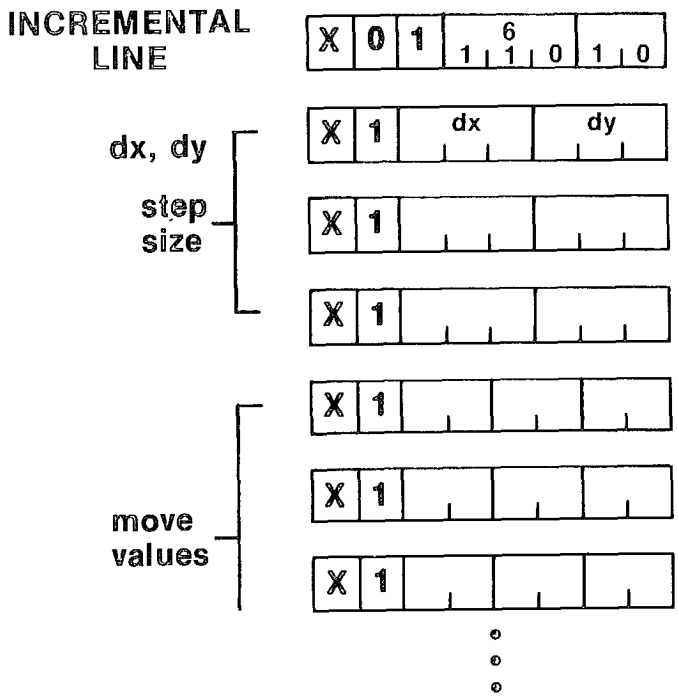
In colour modes 1 and 2, these colour specifications are addresses into the colour look-up table.

The assignment of colour specifications to logical pels is as follows. The first colour specification is deposited into all pixels lying under the logical pel at the current drawing point (if one or both of the dimensions of the logical pel are set to 0, then the dimension(s) are reset - for the duration of the execution of the INCREMENTAL POINT command only - to the smallest specifiable positive value in the current number domain). The drawing point is then shifted over to the right a distance dx equal to the width of the logical pel (note if dx is negative, the drawing point is shifted to the left). All pixels lying under this new position have the next colour specification deposited into them. The drawing point is shifted again and so on. Pixels are assigned colour specifications until the drawing point meets or exceeds the vertical boundary of the active drawing area. When this happens, the drawing point is repositioned to the opposite boundary and the Y value is incremented by the height of the logical pel, dy (i.e. the drawing point moves up if dy is positive, down if dy is negative). If the horizontal boundary is met or exceeded, as well, the y-value is left constant and the image lying within the active drawing area is scrolled a distance equal to -dy. Also note that when the vertical boundary is met (i.e., left or right side), any remaining bits in the current byte of the value operand are ignored and the interpretation of bits after the drawing point has been repositioned to the opposite vertical boundary resumes at the next complete byte, starting with b6.

When all data has been exhausted, the drawing point is set to the origin of the active drawing area.

If at the time an INCREMENTAL POINT PDI is received, there is no currently defined active drawing area, or if the current drawing point does not lie within the defined active drawing area, then the INCREMENTAL POINT PDI is considered to be in error and is ignored.

4.1.6.3 INCREMENTAL LINE



The INCREMENTAL LINE geometric drawing operation provides the capability of compactly describing an image consisting of a series of short line segments drawn in the current colour and line texture.

John Doe

Example INCREMENTAL LINE PICTURE

The first block of data gives the step size parameters, dx and dy, as a signed parameter.

The last block of data gives the move values as an indefinite number of bytes, each of which contains three two-bit nibbles in the numeric data field which are interpreted b6 to b1. The interpretation of these two-bit nibbles is as follows:

Table 4.1 INCREMENTAL LINE MOVE VALUES

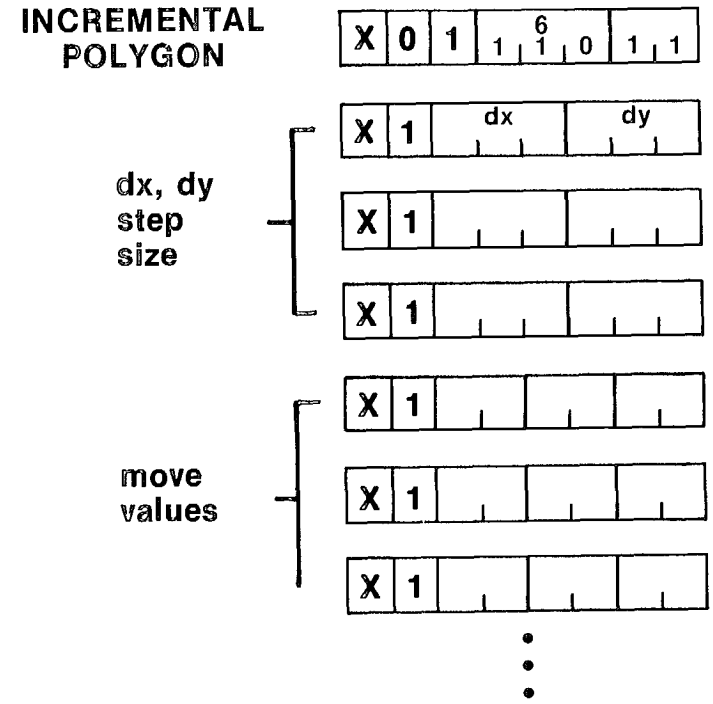
Nibble value	Interpretation
0 0	Interpret the following nibble as a "modify parameter" instruction (see below)
0 1	advance the drawing point a distance dx in the x direction and optionally draw a line
1 0	advance the drawing point a distance dy in the y direction and optionally draw a line
1 1	advance the drawing point a distance dx in the x direction and dy in the y direction and optionally draw a line.

If the draw flag is on, then every time the drawing point is stepped, a line in the current line texture and colour is drawn joining the current drawing point (after the step operation) with the previous drawing point (before the step operation). If the draw flag is off, then no line is drawn after the step operation. When a nibble value of (0,0) is encountered (Table 4.1), the next nibble is interpreted as a "modify parameter" instruction as shown in Table 4.2. The nibble following that is interpreted as a step operation (Table 4.1). Note that the draw flag is initially on whenever the INCREMENTAL LINE opcode is encountered. When the value operand is terminated, the current drawing point is left at the final drawing point.

Table 4.2 INCREMENTAL LINE MODIFY PARAMETER INSTRUCTIONS

Nibble value	Modify parameter instruction
0 0	change the state of the draw flag (on to off if originally on, or off to on if originally off)
0 1	change sign of dx
1 0	change sign of dy
1 1	change sign of dx and dy

4.1.6.4 INCREMENTAL POLYGON (Filled)



The INCREMENTAL POLYGON geometric drawing operation provides the capability of compactly describing a polygon drawn with a series of short line segments and filled in the current colour and texture pattern. The highlight attribute also applies to this PDI.

Example INCREMENTAL POLYGON PICTURE

The interpretation of the operand data following the opcode is exactly as for the INCREMENTAL LINE opcode with the following exceptions:

1) the draw flag is always on

2) should a "modify parameter" instruction (Table 4.2) to change the state of the draw flag (0,0) be encountered, it is treated as a null, and the following nibble is interpreted as the "modify parameter" instruction (Table 4.2).

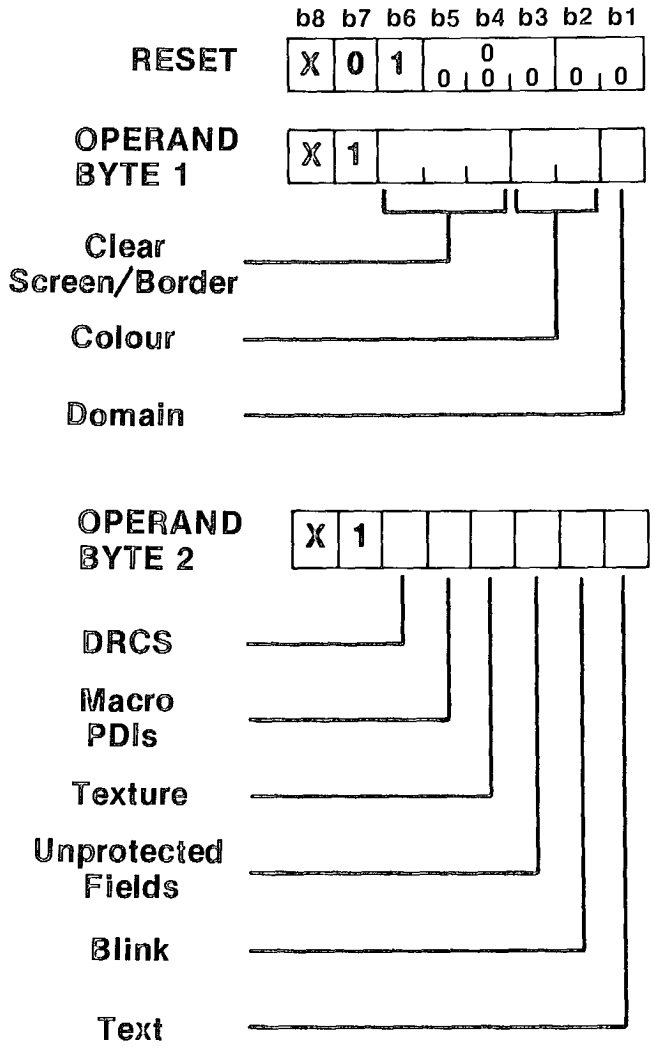
3) the final drawing point is implicitly taken as the initial drawing point.

4) the resulting figure is filled in the current colour, texture pattern and according to the highlight attribute.

Note that the INCREMENTAL POLYGON PDI must enclose a single area (similar to the POLYGON PDI, section 4.1.5)

4.2 Control Opcodes

4.2.1 RESET



This opcode is used to selectively reset parameters as described below. It takes a 2-byte operand.

4.2.1.1 Operand Byte 1 of RESET

Bits 4, 5, 6, of byte 1 clear the screen and/or border to the following colours.

b6	b5	b4	Colour
0	0	0	no action
0	0	1	screen to black
0	1	0	screen to current drawing colour
0	1	1	border to black
1	0	0	border to current drawing colour
1	0	1	screen and border to current drawing colour
1	1	0	screen to current drawing colour and border to black
1	1	1	screen and border to black

The border surrounds the display area and may only be set to one colour at a time. (Screen and display area are synonymous.)

Bits b2 and b3 of byte 1 modify the colour mode and/or current drawing colour as follows:

b3	b2	Colour mode
0	0	no action
0	1	select colour mode 0, initialize implicit colour map if it exists
1	0	select colour mode 1 and set colour map to default colours
1	1	select colour mode 1, set colour map to default colours and set the in-use drawing colour to white

If bit b1 of byte 1 equals 1, the domain parameters are reset to their default values. If b1 is 0, the domain parameters are unchanged.

4.2.1.2 Operand Byte 2 of RESET

If bit b1 of byte 2 equals 1, the cursor is sent to its home position (top left character position on the screen) and all text parameters are reset to their default values. If b1 is 0, the text parameters and the cursor position are unchanged.

If bit b2 of byte 2 equals 1, all blink processes are terminated. If b2 is 0, then blink processes are not affected.

If bit b3 of byte 2 equals 1, all unprotected fields are changed to protected status but the contents are unaffected. If b3 is 0, unprotected fields are left unaffected.

If bit b4 of byte 2 equals 1, all texture attributes are set to their default values. The four programmable TEXTURE MASKS are not cleared. If b4 is 0, current texture attributes are not changed.

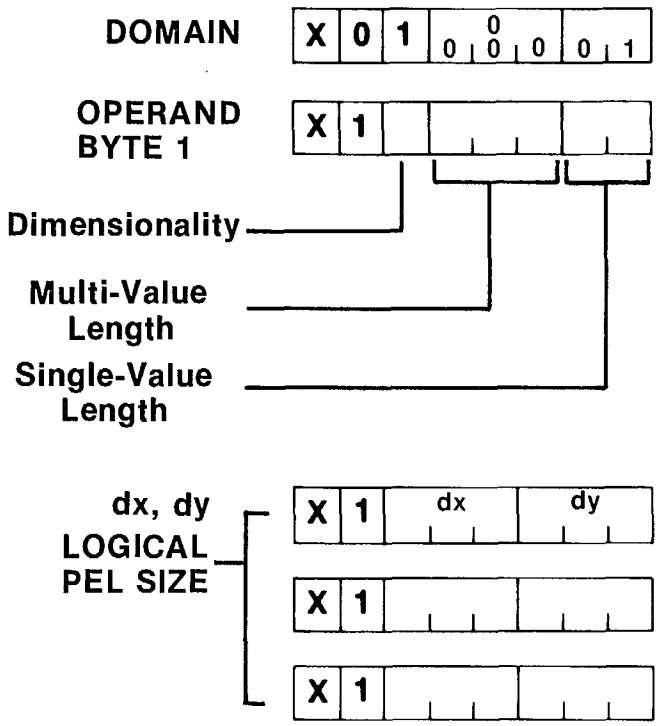
If bit b5 of byte 2 equals 1, all MACRO - PDIs are cleared. This includes TRANSMIT-MACROS. If b5 is 0, MACRO - PDIs are unaffected.

If bit b6 of Byte 2 equals 1, all DRCS characters are cleared to zeros. If b6 is 0, the DRCS characters are unaffected.

If the RESET command is received with no operands, it is interpreted as if it had been sent with all bits in both bytes set equal to 1. If only the first byte is received, the second byte is then interpreted as if it had been received with all zeros. If more than two data bytes are received, the additional byte(s) are ignored.

For descriptions of TEXTURE MASKS, TRANSMIT-MACRO and DRCS, see sections 4.3.3, 4.3.1 and 4.3.2 respectively.

4.2.2 DOMAIN



The domain command permits the numbered domain for coordinate data, Red-Green-Blue (R-G-B) colour specifications and for colour map addresses to be specified. As well it allows the size of the logical pel to be defined.

Bits b1 and b2 of byte 1 determine the number of bytes to be used in single-value format operands according to the following table:

b2	b1	single-value operand length (bytes)
0	0	1 (default value)
0	1	2
1	0	3
1	1	4

Colour map addresses are expressed as single-value operands.

Bits b3, b4, and b5 of byte 1 determine the number of bytes to be used in multi-value format operands according to the following table:

b5	b4	b3	Multi-value operand length (bytes)	
0	0	0	1	
0	0	1	2	
0	1	0	3	(default value)
0	1	1	4	
1	0	0	5	
1	0	1	6	
1	1	0	7	
1	1	1	8	

Coordinate data and R-G-B colour values are expressed as multi-value operands.

If bit b6 of byte 1 equals 0, the 2-dimensional (x,y) mode is specified. This is the default mode. If this bit equals 1, the 3-dimensional (x,y,z) mode is indicated. This mode has not yet been defined.

The coordinate data following byte 1 of the operand is interpreted to be the width (dx) and height (dy) of the logical pel. The default size of the logical pel is dx = dy = +0. (i.e., dimensionless drawing point, but the visible display would be equal to a single display-pixel dimension).

The concept of logical pel provides a controllable stroke width capability (analogous to the width of a paint brush) for use with the POINT, LINE, ARC, RECTANGLE, POLYGON and INCREMENT drawing PDIs. These drawing operations are defined so as to affect all display pixels lying under any portion of the logical pel as it is mapped to the display screen. The geometric alignment of the drawing point within the logical pel is:

lower left corner if both dx and dy are positive;

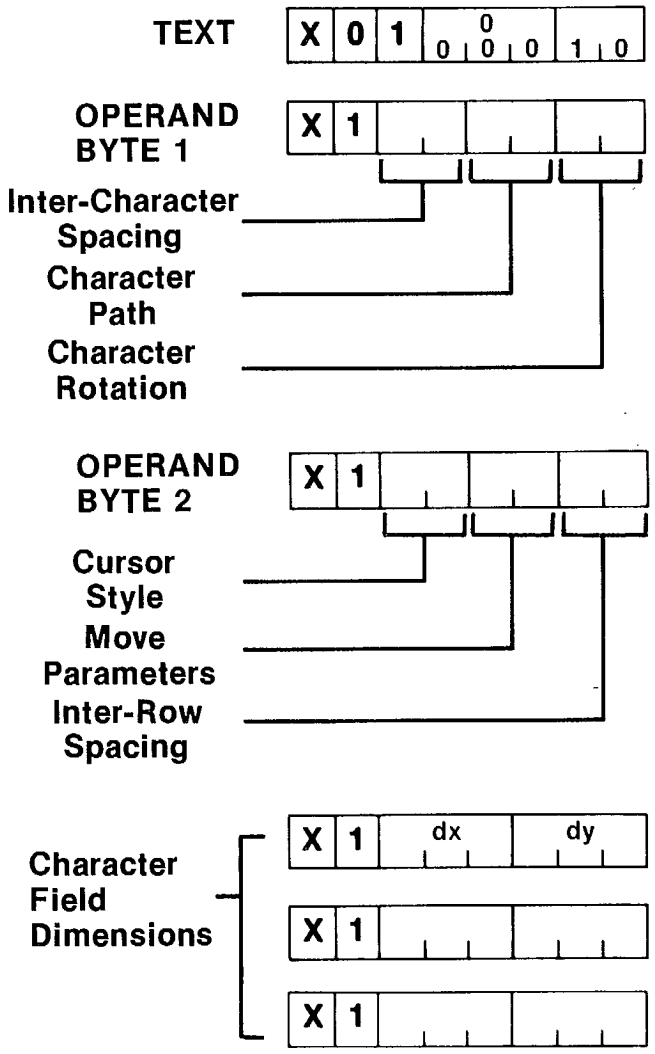
lower right corner if dx is negative and dy is positive;

upper left corner if dx is positive and dy is negative, and

upper right corner if both dx and dy are negative.

If additional bytes follow the logical pel size data byte(s), they are ignored.

4.2.3 TEXT



This opcode is used to modify parameters which describe the manner in which text characters are presented.

Bits b1 and b2 of byte 1 are used to specify character rotation as shown below:

b2	b1	Rotation
0	0	0° (default value)
0	1	90°
1	0	180°
1	1	270°

Bits b3 and b4 of byte 1 determine the direction of the character path as shown in the table below:

b4	b3	Character path movement
0	0	right (default)
0	1	left
1	0	up
1	1	down

Bits b5 and b6 of byte 1 are used to determine the distance the text cursor is moved (in multiples of the character field dimension lying parallel to the character path) after a character is displayed or after a backspace or horizontal tab character is received. This is known as the inter-character spacing and is as defined in the table below:

b6	b5	inter-character spacing
0	0	1 (default value)
0	1	1.25
1	0	1.5
1	1	proportional spacing

With proportional inter-character spacing, the spacing is determined by the width of the character. The exact definition of proportional spacing is left to the terminal manufacturer.

If the cursor goes past the edge of the unit screen or active drawing area:

a) For the case of fixed inter-character spacing:
an automatic Carriage Return (CR), and Line Feed (LF) are executed;
b) For the case of proportional spacing:
an automatic CR and LF are executed only if the character to be deposited does not fit into the dimension available.

Bits b1 and b2 of byte 2 determine the spacing between rows of characters in multiples of the character field dimension lying perpendicular to the character path according to the following table:

b2	b1	inter-row spacing
0	0	1 (default value)
0	1	1.25
1	0	1.5
1	1	2

Bits b3 and b4 of byte 2 are used to define the relationship between movement of the text cursor and movement of the graphics drawing point as shown in the table below:

b4	b3	move attribute
0	0	move together (default)
0	1	cursor leads
1	0	drawing point leads
1	1	move independently

If the cursor and drawing point are set to move together, then whenever the text cursor is moved (such as when characters are displayed) the graphics point follows it; and, conversely, whenever the drawing point is moved (such as with a drawing PDI) the text cursor is also moved to the same location.

If the cursor is defined as leading, then the drawing point will follow the text cursor but not vice versa.

If the drawing point is set to lead the text cursor, then the text cursor will follow the movements of the drawing point but not vice versa.

If the drawing point and the cursor are set to move independently, then movement of one will not effect the position of the other.

Movement of the drawing point should never cause the text cursor to be located such that any part of the character field indicated by the text cursor would fall outside the unit screen. Should such a situation arise, the text cursor will be moved as close as possible to the drawing point without violating the above condition.

The alignment of the drawing point and of the text cursor is such that the drawing point corresponds to the left end of the underscore cursor, the bottom left corner of the block cursor, the center of the crosshair cursor and the center of the custom cursor (undefined).

Bits b5 and b6 of byte 2 are used to determine the display style of the cursor as follows:

b6	b5	cursor style
0	0	underscore (default style)
0	1	block
1	0	cross hair
1	1	custom

The text cursor is located in the position in which the next text character is to be deposited. The underscore is a single line the width of the current character field at the bottom of the character field. The block cursor is a solid block whose size is the size of the current character field. The cross-hair cursor consists of a vertical line and a horizontal line which intersect at the center of the character field and whose height and width are equal to the height and width of the current character field. The definition of the shape of the custom cursor is left to the terminal manufacturer.

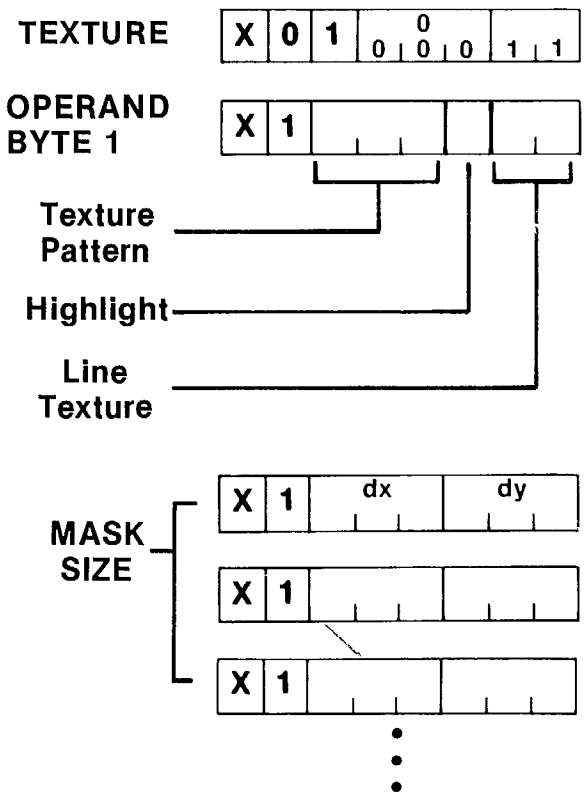
The block of coordinate data following the first two bytes of the operand give the width (dx) and height (dy) of the character field.

If dx is negative, the character pel patterns are reflected about the vertical center axis of the character field. If dy is negative, the character pixel patterns are inverted about the horizontal center axis of the character field.

If the character field dimensions are omitted from the operand, then the current character field dimensions remain unchanged.

The default dimensions of the character field are dx = 0.025 and dy = approximately 0.04.

4.2.4 TEXTURE



This opcode is used to set texture attributes used to control the drawing of lines, the highlighting of filled areas and the texture patterns used to fill filled areas.

Bits b1 and b2 of byte 1 are used to set the line texture attribute, which determines the style of lines and outlines (but not highlights) drawn with the LINE, ARC, RECTANGLE, POLYGON and INCREMENTAL LINE drawing PDIs. The size of the dot and of the inter-dot spacing are set equal to the size of the logical pel. The height of the dash is equal to the height of the logical pel. The length of the dash and the inter-dash spacing is equal to three times the width of the logical pel. All end points of lines and arcs and all vertices of rectangles and polygons must be plotted regardless of the line texture used. If the logical pel size is 0, solid lines will always be drawn.

b2	b1	line texture
0	0	solid (default)
0	1	dotted
1	0	dashed
1	1	dot dashed

If bit b3 of byte 1 is equal to 1, then all filled rectangles, arcs, polygons and incremental polygons are drawn in highlighted mode. In this mode, perimeters are drawn in black in colour Modes 0 and 1, and in the background colour in Colour Mode 2. The default state of this attribute is no highlight (b3 = 0).

Bits b4, b5 and b6 are used to select the texture pattern to be used in filling rectangles, arcs, polygons and incremental polygons according to the following table:

b6	b5	b4	texture pattern
0	0	0	solid (default pattern)
0	0	1	vertical hatching
0	1	0	horizontal hatching
0	1	1	cross hatching
1	0	0	mask A
1	0	1	mask B
1	1	0	mask C
1	1	1	mask D

The width and spacing of hatching lines in the vertical hatching pattern is equal to the width of the logical pel. The height and spacing of hatching lines in the horizontal hatching pattern is equal to the height of the logical pel.

The programmable texture masks are defined using the DEF TEXTURE command as described in section 5.3.3.

The block of coordinate data following the first byte of the operand specifies the mask size (dx, dy) to be used in the step-and-repeat process. This process takes the selected texture mask, scales it to the specified mask size, logically covers the given object with contiguous copies of the mask and then deposits the in-use colour(s) in all pixels indicated by the mask pattern. This process takes as its initial reference, the origin (0,0) point of the unit screen in order that registration of the pattern be maintained across figures at any given mask size.

The default mask size is dx = 0.025, dy = approximately 0.04 (the default character field size). The sign bits of dx and dy are used to reflect or invert respectively the mask pattern within the mask field in a manner similar to reflection or inversion of text character fields.

If the mask size operand is not present within the texture PDI, then the current mask size is not changed.

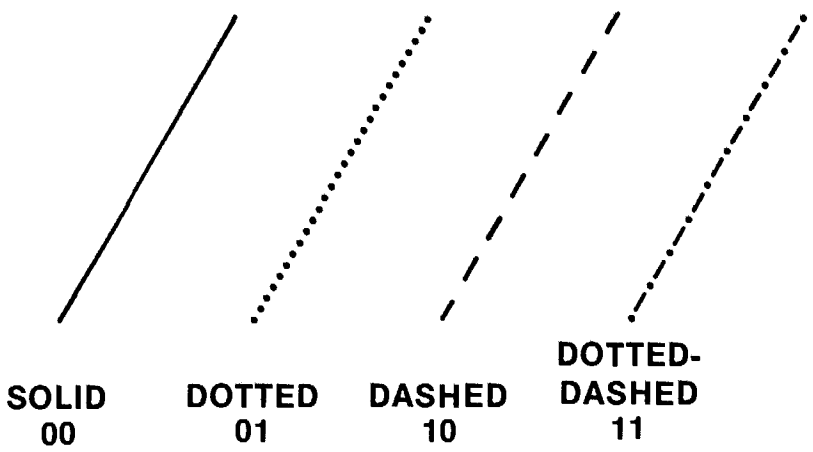
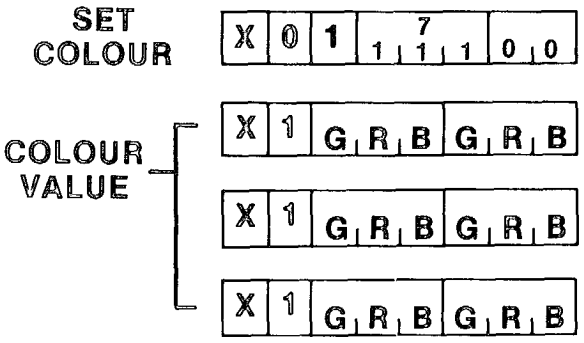


Figure 4.2 LINE TEXTURES

4.2.5 SET COLOUR



The SET COLOUR command is used to specify colour values accessed by subsequent drawing opcodes and by text and mosaics. The "colour value" operand is used to define a colour according to the scheme illustrated below.

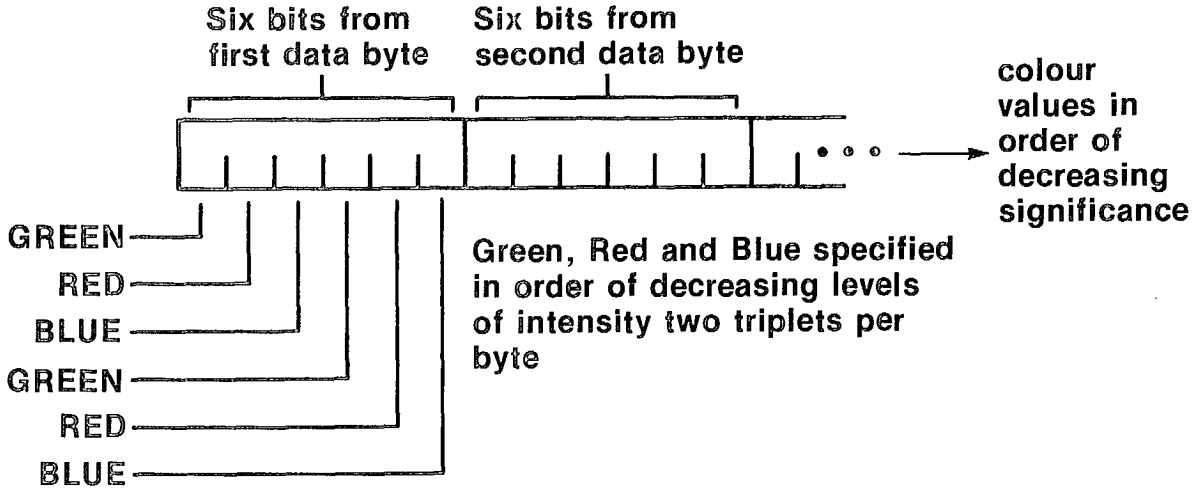


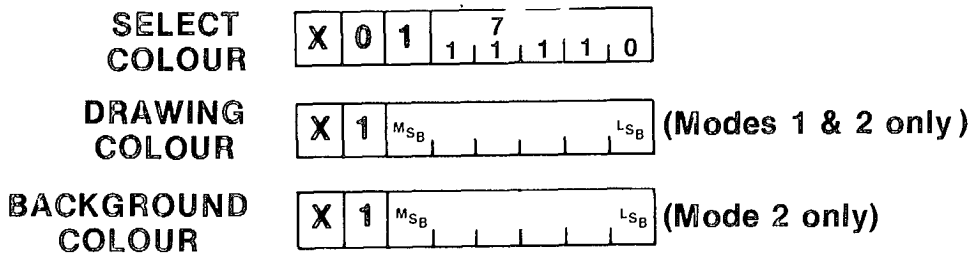
Figure 4.3 Colour Encoding Scheme

In Colour Mode 0, (see SELECT COLOUR command) the colour specified as the operand for this command is used to set the actual colour used in subsequent text and graphics drawing commands. Note that a colour specification of G = R = B = 0 corresponds to "transparent", in which lower order planes would show on the display. These planes could correspond to memory planes or to an analog television signal. In the absence of any lower order planes, the "transparent" colour is treated as "black".

In Colour Modes 1 and 2, the colour specification given is loaded into the colour map at the address indicated by the in-use drawing colour (which must have been selected previously with the SELECT COLOUR command). If the maximum number of bits that the colour map can accommodate is smaller than the number of bits provided by the SET COLOUR operand, the operand is truncated and only the most significant bits are used. If the maximum number of bits that the colour map can accommodate is larger than the number of bits provided by the SET COLOUR operand, the operand is padded with 0s.

If more data follows this operand; i.e., there are more bytes given than the number of bytes specified for a multi-value operand (as specified by the DOMAIN command), then the SET COLOUR command is implicitly repeated with the address of the colour entry automatically incremented prior to the execution of the new SET COLOUR command. The in-use drawing colour is not affected by this implicitly executed command.

4.2.6 SELECT COLOUR



The SELECT COLOUR opcode is used to select the colour mode and to select the in-use drawing colours for modes 1 and 2.

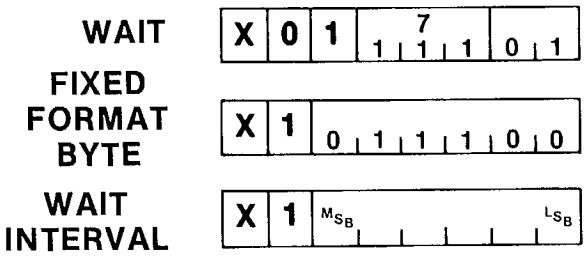
If the SELECT COLOUR opcode is followed by no operand data, then colour mode 0 is selected. In this mode, the in-use colour must be explicitly defined by specifying the R-G-B values with the SET COLOUR command.

If this opcode is followed by a one single-value operand, then Colour Mode 1 is selected. The operand indicates the colour map address which gives the in-use drawing colour to be used in this mode.

If the SELECT COLOUR opcode is followed by two single-value operands then Colour Mode 2 is selected. In this case, the first byte of data gives the colour map address to be used for graphics and text. The second byte of data gives the background colour which is used for text characters. In this mode, text characters are drawn in the current drawing colour while the background colour occupies the remainder of the character field. The background colour is also used as the highlight colour for filled areas and for the alternating colour in the line and area texture patterns. If both bytes are identical, then the in-use drawing colour is unchanged and only the in-use background colour is changed to the specified value.

Colour mode 0 is the default colour mode, white being the default in-use colour. A default set of colours for the colour map is described in Appendix E (for terminals allowing explicit manipulation of the map).

4.2.7 WAIT

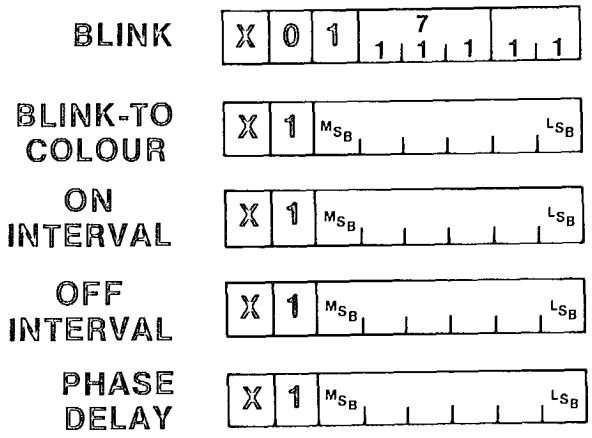


The WAIT command (which is coded as a CONTROL (STATUS) sub-command (see section 4.4), is used to cause a delay in processing of a specific time.

The first byte of operand data following the WAIT opcode must follow the format given in the diagram shown above. The next byte of operand data gives the time delay in units of 1/10 of a second (64 binary coded values). Only bits b1 through b6 are used for this purpose. If any additional data bytes follow, they are treated as additional periods of waiting time, each period being specified independently by each data byte.

When multiple WAITS are used to generate long delays, any user input will terminate subsequent consecutive WAITS described in the same WAIT command (i.e., the entire operand of the currently executing WAIT command is flushed).

4.2.8 BLINK



The BLINK opcode is used to cause a colour map entry to periodically alternate between two colours.

The mechanism for performing this is a blink process. This process periodically overwrites the contents of the current in-use drawing colour (the "blink-from" colour) and substitutes another entry in the colour map which is called the "blink-to" colour. The blink-to colour is activated for a period of time known as the ON interval. The blink-from colour is activated for a period of time known as the OFF interval. The ON and OFF intervals alternate with each other. A phase delay may also be specified, this being a delay in the start of the ON interval referenced to the start of the ON interval of the last active blink process defined. If multiple blink processes have ON or OFF intervals that expire simultanteously, they are processed sequentially starting with the most recently defined blink process and ending with the latest defined blink process. In this case, each blink process takes as its input the colour map that resulted from the previously executed blink process.

The first single-value operand following the BLINK opcode is the blink-to colour specification, specified as a colour-map address. The next fixed format operand is the ON interval specified in units of 1/10th of a second. Only bits b6 through to b1 are used for this specification. In a similar manner, the next fixed format operand specifies the OFF interval. The fourth single-value operand specifies the phase delay, also in units of 1/10th of a second. If this byte is omitted, a phase delay of 0 is indicated.

Defining a blink process on a pair of blink-to and blink-from colours automatically terminates any previously defined blink process operating on the same pair of colours. If no operands follow the blink opcode then all blink processes utilizing the current in-use colour as the blink-from colour will be terminated.

If additional data follows a completely specified blink process, then the blink command is implicitly repeated with the address of the blink-from colour being automatically incremented prior to the execution of the new opcode. The in-use drawing colour is not affected by this incrementing.

4.3 C1 Control Set

The C1 control set (see Figure 4.4) is used to allow control over text format and also to create MACRO-PDIs, DRCS characters, programmable texture masks and unprotected fields.

4.3.1 MACRO-PDIs: General

The MACRO-PDI facility provides the facility for a string of presentation level characters to be stored within the terminal and to be subsequently executed via a single MACRO call. MACROs are called by simply designating the MACRO-PDI name set into the current G set and then by sending one of the 96 characters (2/0, 2/1, ... 7/15) in that G set (which are named M0 through M95). It is also possible to execute MACROs from a key on the keyboard, if the manufacturer provides such a feature, MACRO may be nested by including a MACRO calling sequence within a MACRO definition. The application level must assume responsibility to avoid infinite looping when MACROs are nested.

4.3.1.1 DEF MACRO

This C1 control is used to define a MACRO-PDI. The character following this control is interpreted as the name of the MACRO; e.g., the character (2/0) would cause MACRO M0 to be defined. All characters subsequent to this character are stored (but not executed) within the terminal under the specified MACRO name. Definition of the MACRO terminates upon receipt of one of the following C1 controls: DEF MACRO, DEFP MACRO, DEFT MACRO, DEF DRCS or END. Neither the terminating control character nor its preceding ESC character (in a 7-bit environment) is stored as part of the MACRO.

Definition of a MACRO replaces any previously existing MACRO under the same name. A null MACRO definition; i.e., a MACRO definition in which there are no characters between the MACRO name and the terminating C1 character (or its preceding ESC character), causes that MACRO to be deleted. Definiton of a MACRO-PDI is independent of whether the MACRO-PDI set is invoked or not (though it must, of course, be invoked in order to actually execute the MACRO).

All MACROs may be simultaneously deleted with the RESET command.

4.3.1.2 DEFP MACRO

The operation of this control character is identical to that of the DEF MACRO command except that:

1) incoming characters that make up the MACRO definition are simultaneously executed as well as being stored;
2) the DEFP MACRO cannot be used to define a MACRO that calls itself.

4.3.1.3 DEFT MACRO

The DEFT MACRO control character is used to define a TRANSMIT-MACRO. TRANSMIT-MACROs, when called, are not executed, but are transmitted back to the host computer in their entirety. This could, for example, be used to provide a programmable-function key capability if one or more keys on the keyboard were capable of causing the execution of MACROs.

TRANSMIT-MACROs are defined and deleted in a manner similar to that described for normal MACRO-PDIs, and they share the same 96 MACRO names.

4.3.2 DEF DRCS

The DEF DRCS command is used to start the downloading operation for one of the DRCS characters of which a total of 96 are permitted. This is followed by the bit combination representing the DRCS character being defined. Next comes any valid stream of presentation-level code. The DRCS downloading operation is terminated when an END, DEF MACRO, DEFP MACRO, DEFT MACRO, DEF TEXTURE, or another DEF DRCS command is received. When the current DRCS downloading operation is terminated by a DEF DRCS command, the next character of the DRCS G set (i.e., in the sequence 2/0, 2/1, ... 2/15, 3/0, 3/1 .....) is defined by the presentation level code immediately following the DEF DRCS command. (This is the only time the DEF DRCS character is not followed by the DRCS character to be defined.) If the DEF DRCS name character is followed by an END command, the buffer space allocated to that character is freed.

Definition of a DRCS set is independent of whether the set is invoked or not (though it must, of course, be invoked in order to actually display the character).

The presentation level code defining the DRCS character is executed within the unit screen upon being received by the terminal. It is not, however, displayed on the screen, but rather is used to modify a separate storage buffer which is mapped from the unit screen. The storage buffer is one bit deep (i.e., on or off only), has a width equal to the current character width (in pixels) and a height equal to the current character height (in pixels). This buffer will be used in a manner similar to the permanently stored ASCII character pixel patterns and will be subject to the same attributes (text size, in-use colour, etc.).

The buffer is initially cleared to all 0s upon receipt of the DEF DRCS command. All presentation level code is then executed, causing the appropriate bits in the DRCS buffer to be set to ones to define the DRCS shape. SET COLOUR, SELECT COLOUR and BLINK will be executed, but will have no effect on the DRCS character definition. If the text size is changed, it will have an effect upon the storing of following text characters within the DRCS buffer, but it will not affect the size of the DRCS buffer since it has already been allocated.

After the downloading sequence has been terminated, the terminal reverts to the normal procedure of mapping the unit screen to the display screen with the drawing point reset to (0,0).

The entire DRCS character set may be cleared using the RESET command. Should a RESET that clears the DRCS be received during a downloading operation, it will clear the character pixel pattern definition in progress, but the downloading operation will continue.

4.3.3 END

End terminates the current DEF MACRO, DEFP MACRO, DEFT MACRO, DEF DRCS or DEF TEXTURE operation.

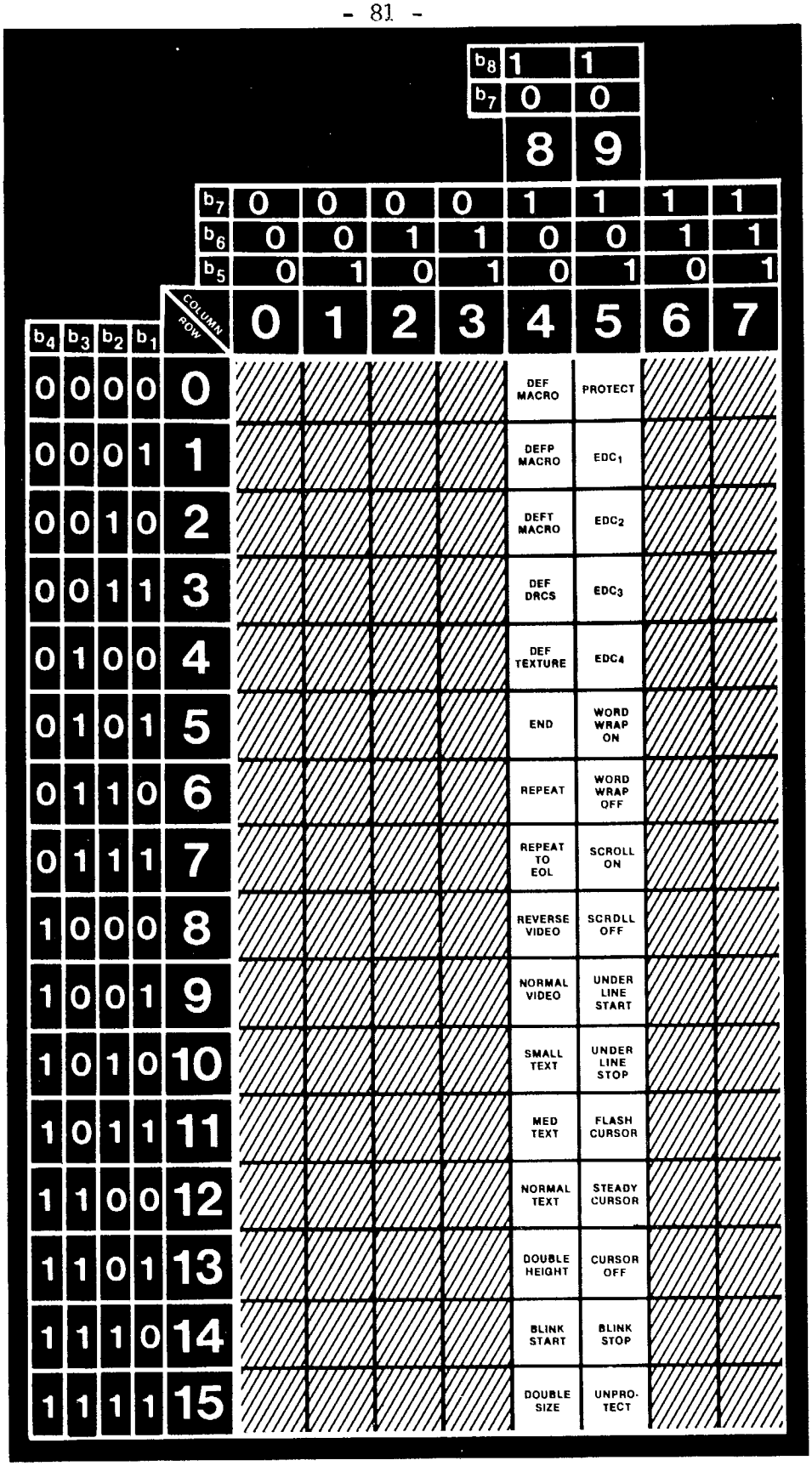


Figure 4.4 C1 Control Set

4.3.4 DEF TEXTURE

This command is used to define one of the four programmable texture masks described in section 4.2.4.

The DEF TEXTURE character is immediately followed by one of the following bit combinations: (4/1), (4/2), (4/3), (4/4) which causes selection of mask A,B,C or D respectively to be defined. Any existing texture pattern associated with the specified mask is deleted. The mask may also be cleared by sending an END command at this point. Presentation level code follows, which describes the texture mask in the same manner as DRCS characters. At the end of the downloading sequence, the terminal reverts to the normal procedure of mapping the unit screen to the display screen with the drawing point reset to (0,0).

The TEXTURE masks may be cleared via the RESET command.

4.3.5 PROTECT/UNPROTECT

Unprotected fields are rectangular areas on the display into which the user may enter or edit data for possible subsequent transmission to the host. The method of entering and editing data into these fields is left to the terminal manufacturer.

By default, the entire display screen is protected; i.e., it is not possible for the user to enter or alter data on the screen. However, if the UNPROTECT command is given, the active drawing area (as defined by the FIELD command described in section 4.1.6.1) is made unprotected and the user may subsequently add or alter data within that field. If no active drawing area is currently defined, then the entire display screen is made unprotected. Any number of unprotected fields may be defined (subject to terminal-dependent memory limitations) by defining an active drawing area via the FIELD PDI and then issuing the UNPROTECT command. Should an UNPROTECT command result in unprotecting an area which partially or wholly lies within an already unprotected field, the already unprotected field is made protected (without affecting the displayed contents) before the new area is made unprotected. This prevents unprotected fields from overlapping.

The user may enter or locally edit information (text or graphics) into an unprotected field. The host may also add information into this field which may subsequently be edited by the user. When the user initiates a transmission to the host, the information entered into the unprotected field(s) is transmitted conforming to the Presentation Level Protocol described in this document. The transmit presentation process and its associated states must be maintained separately from the receive presentation process. In transmission, the FIELD PDI containing the coordinates of the origin of the unprotected field as well as its dimensions is transmitted followed by the contents of the field transmitted according to the Presentation Level Protocol defined in this document , and then by the END command. When more than one unprotected field is present, the order of transmission is top left to bottom right.

Unprotected fields may be re-protected via the PROTECT command. This command causes all unprotected fields of which any portion lies within the active drawing area to be made protected. The entire screen is protected if no active drawing area is currently defined. The RESET command may be used to protect all currently defined unprotected fields.

4.3.6 Text Controls

This set of 20 controls allows simple textual functions to be implemented, some of which may also be performed via the TEXT or BLINK PDIs.

4.3.6.1 REPEAT

This command causes to last text character received to be repeated a specific number of times. Bits b6 to b1 of the character following the REPEAT character are interpreted as the repeat count. This repeat count character must be one of the characters in columns 4 though 7 of the in-use table otherwise the command is considered in error and is not executed.

4.3.6.2 REPEAT TO EOL

When this command is encountered, the last TEXT character received is repeated until the last character position along the current character path is reached. If any portion of the text cursor lies within the currently defined active drawing area when this command is encountered, then characters are repeated only up to the last character position along the current character path within the active drawing area.

4.3.6.3 REVERSE VIDEO

This command cause the terminal to enter reverse video mode in which any text characters accessed are "complemented" within the current character field prior to their display. By "complemented" is meant that the pixels surrounding the character shape are turned on instead of those pixels describing the character shape itself.

4.3.6.4 NORMAL VIDEO

This command causes the terminal to exit reverse video mode. This is the default state.

4.3.6.5 SMALL TEXT

This command causes the dimensions of the character field to be set as follows:

dx = 0.0125
dy = (approx.) 0.04

This allows a 20 row X 80 column (nominal) character display.

4.3.6.6 MEDIUM TEXT

This command causes the dimensions of the character field to be set as follows:

dx = 0.03125
dy = (approx.) 0.047

This allows a 16 row X 32 column character display.

4.3.6.7 NORMAL TEXT

This command causes the dimensions of the character field to be set to their default value; that is:

dx = 0.025
dy = (approx.) 0.04

This allows a 20 row X 40 column character display.

4.3.6.8 DOUBLE HEIGHT

This command causes the dimensions of the character field to be set as follows:

dx = 0.025 (default)
dy = (approx.) 0.08 (double default height)

This allows a 10 row X 40 column character display.

4.3.6.9 DOUBLE SIZE

This command causes the dimensions of the character field to be set as follows:

dx = 0.05 (double default width)
dy = (approx.) 0.08 (double default height)

This allows a 10 row X 20 column character display.

4.3.6.10 WORD WRAP ON

This command causes the terminal to enter word wrap mode. In this mode, text is buffered into words. A word is displayed on the current line on the screen only if the entire buffered word will fit into the space remaining on the current line within the unit screen (or active drawing area, should the text cursor lie within that area). If the word does not fit into the space remaining on the current line, then the cursor is repositioned beginning at the first character position on the next line within the screen or area, and the word is displayed. The SPACE character should be dropped if the last word on the line is terminated with a SPACE that does not fit on that line.

For the purposes of this section, a word is defined as being an accumulation of text characters beginning with any valid text character and terminating with:

the SPACE character or optionally
one of: ! " $ % ( ) [ ] < > { } ^ * + - / , . : ; = ? _ ~
or
a mosaic set character
or
a PDI opcode
or
a format Effector control character such as Bell, Information Separation Characters (FS, RS and US)
or
any character which causes the length of the word to equal the maximum length of a line.

4.3.6.11 WORD WRAP OFF

This command causes the terminal to exit word wrap mode. In this mode (the default mode) all text is broken on character boundaries whenever an automatic carriage return and line feed are executed.

4.3.6.12 SCROLL ON

In this mode a line feed or vertical tab command which would advance the cursor out of the screen or active drawing area (if it should be in such an area) causes the entire display within the screen or area to be scrolled. Scrolling occurs pixel-by-pixel in a direction perpendicular to the character path and far enough to bring the next intended character location just into the screen or drawing area.

Data scrolled out of the screen or active drawing area may be buffered at the discretion of the terminal manufacturer.

4.3.6.13 SCROLL OFF

In this mode a line feed or vertical tab command which would advance the cursor out of the screen or active drawing area (if it should be in such an area) causes the cursor to be repositioned to the opposite edge of the screen or area such that the character field so defined lies entirely within the screen or area.

4.3.6.14 UNDERLINE START

This command causes the terminal to enter underline mode. In this mode, all ASCII, supplementary graphics and DRCS characters have a line added to their pixel patterns. The line appears at the bottom of the character field and extends across the entire width of the field. Its thickness is determined by the vertical dimension (dy) of the logical pel size.

All characters from the Mosaic set, in this mode, are displayed in separated graphics mode; i.e., the six mosaic elements composing each character are reduced horizontally by the width of the logical pel (dx), and vertically by the

height of the logical pel (dy) with the reduced mosaic elements being left and bottom justified within the normal element area, the remainder of the area being considered as "background".

4.3.6.15 UNDERLINE STOP

This command causes the terminal to exit underline mode. While in this mode (the default mode), characters from the Mosaic set are displayed in contiguous mode; i.e., the six mosaic elements composing each mosaic character completely span their normal element areas.

4.3.6.16 FLASH CURSOR

This command turns on the text cursor display and causes it to flash intermittently.

4.3.6.17 STEADY CURSOR

This command turns on the text cursor display and maintains it steadily visible.

4.3.6.18 CURSOR OFF

This command causes the text cursor to be made invisible (default mode). Note that the cursor still functions and moves as usual, it is just not visible while in this mode.

4.3.6.19 BLINK START

This command creates a blink process in which: the blink-from colour is the current in-use drawing colour; the blink-to colour is nominal black in Colour Modes 0 and 1, or the in-use background colour in Colour Mode 2; the on and off intervals are at the discretion of the manufacturer; and the phase delay is 0.

In colour mode 0 the C1 blink command establishes an automatic process which implicitly defines a blinking colour. Changing the current colour causes the current colour to cease being the blinking colour.

4.3.6.20 BLINK STOP

This command turns off any currently active blink processes utilizing the current in-use colour as the blink-from colour.

4.3.6.21 EDC1, EDC2, EDC3 AND EDC4 (Extended Device Controls)

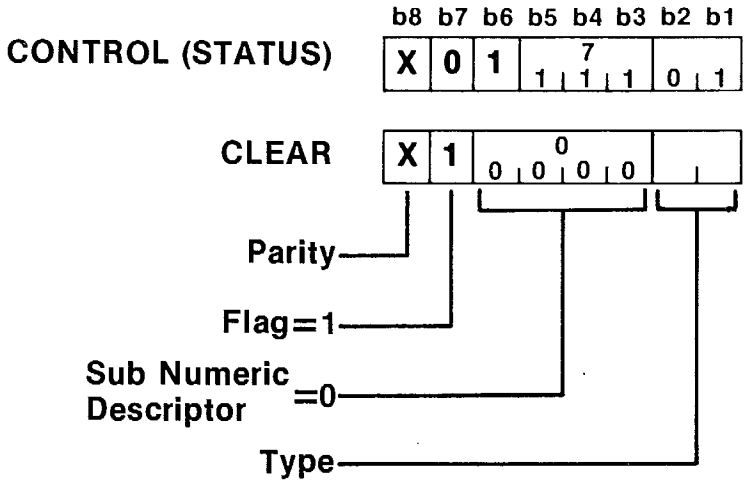
The precise meanings of EDC1, EDC2, EDC3, EDC4 are reserved for further study.

4.4 Control (Status) Sub-Commands

The Control (Status) group of sub-commands perform a subset of the control functions which can be done via the CONTROL PDIs already discussed in section 4.2. These sub-commands are included in order to harmonize the new standard with CCITT Recommendation S.100.

Only the level of functionality described in S.100 is included here. However, it is desirable to implement the Control (Status) command to the level described in the Telidon Field Trial Specification (CRC Technical Note 699) in order to maintain compatibility with existing Telidon Field Trials. Refer to document 699 for a detailed specification of these codes.

4.4.1 CLEAR



Control (Status) Clear

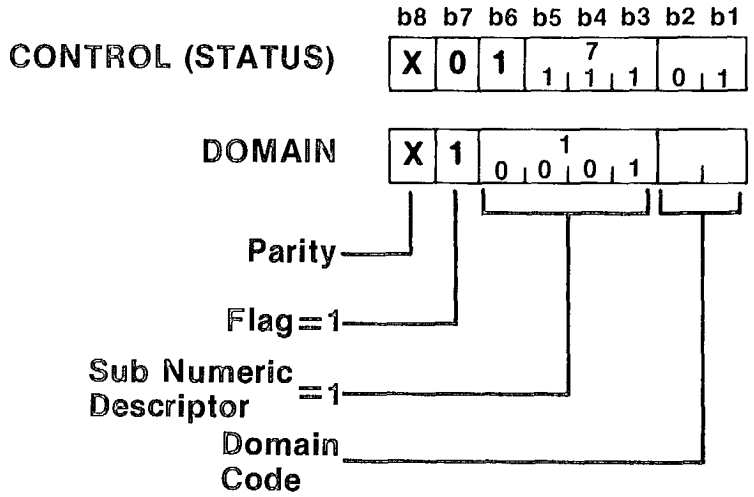
The CLEAR Status Command provides general control over all states of the terminal by permitting general erasure of the entire display screen and also control to reset all the other states of the terminal to their initial conditions.

The "type" bits (b1 and b2) determine the specific action to be taken as follows:

b2	b1	Action
0	0	Clear the screen to black and reset the text cursor and drawing point to HOME position (i.e., the top left character position on the screen).
0	1	Clear the screen to transparent and reset the text cursor and drawing point to HOME position.
1	0	Clear the screen to black, reset the text cursor and drawing point to HOME position and reset all terminal states to their default conditions.
1	1	Clear the screen to the in-use colour and reset the text cursor and drawing point to HOME position.

If any additional data bytes follow, they are ignored.

4.4.2 DOMAIN



Control (Status) Domain

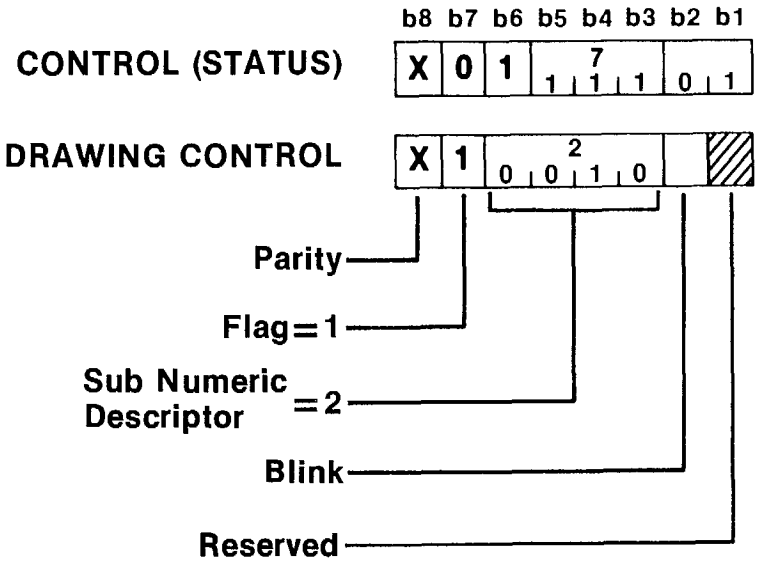
The DOMAIN Status Command permits the number of bytes to be used for multi-value operands to be specified.

The DOMAIN code (bits b2 and b1) are interpreted as follows.

b2	b1		No. of bits used to specify each coordinate	No. of bytes for multi-value operands
0	0	(default)	9	3
0	1		12	4
1	0		15	5
1	1		18	6

If additional data bytes follow a DOMAIN control command, they are ignored.

4.4.3 DRAWING CONTROL



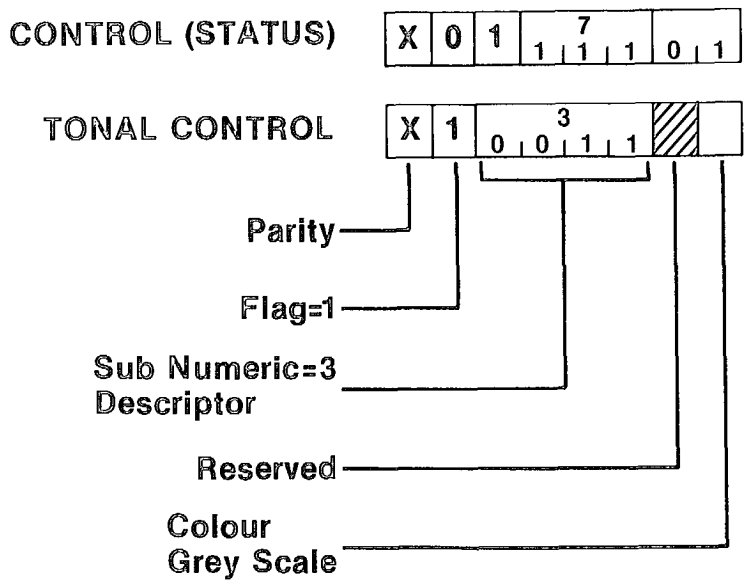
Control (Status) Drawing Control

If bit b2 of the DRAWING CONTROL sub-command is 1, then BLINK mode is activated. If this bit is 0, then BLINK mode is turned off by reverting to the previous in-use colour.

The BLINK mode of the terminal allows particular objects drawn on the display screen to be flashed in a repetitive manner for emphasis. Any item can be blinked by setting the blink status flag before drawing the object. The end of BLINK status may be indicated by another DRAWING CONTROL Status Command in which the BLINK status bit is cleared or by specifying another colour. In general an object of any colour or grey scale level may be blinked, but in some implementations, blinking may be restricted to white, so that any object which is specified to blink will appear in the colour white.

If any additional data bytes follow a DRAWING CONTROL Status Command they are ignored.

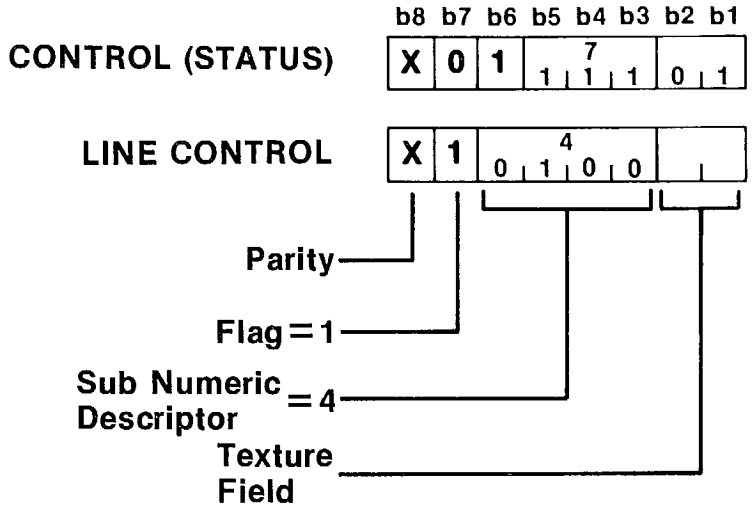
4.4.4 TONAL CONTROL



Control (Status) Tonal Control

The Control (Status) TONAL CONTROL defines the colour or greyscale value accessed by subsequent drawing operations. This value may be used to represent either colour or greyscale value for a particular picture element as defined by the TONAL CONTROL Status command. Bit 1 of the TONAL CONTROL is initially 0 indicating that all subsequent drawing operations would produce coloured lines, rectangles, text, etc. If bit 1 of the TONAL CONTROL is set to 1, all subsequent drawing operations would produce picture drawings in which the value would be interpreted as a greyscale level. Both coloured and greyscale graphical drawing entities may co-exist in the same picture. The drawing operations to create the coloured and greyscale picture are separated by TONAL CONTROL status Commands.

4.4.5 LINE CONTROL



Control (Status) Line Control

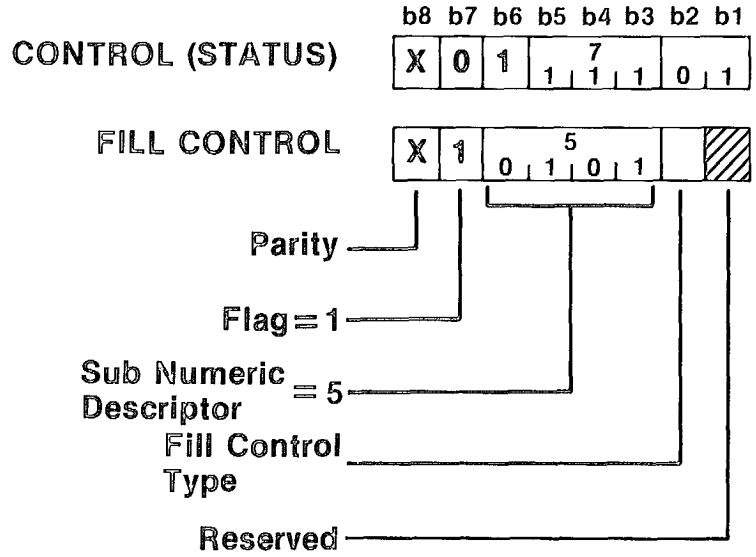
Line texture controls the form in which LINEs or outlines for a RECTANGLE, POLYGON or ARC are drawn. The following line textures are available.

b2	b1	line texture
0	0	solid
0	1	dotted
1	0	dashed
1	1	dotted-dashed

The LINE CONTROL status is defined in a two bit texture field consisting of bits b2 and b1 from the LINE CONTROL Status Command. The default line texture is a solid line. If the LINE CONTROL Status Command is issued, subsequent lines are drawn with the specified line texture. The line texture pattern is referenced to the absolute coordinate grid of the display screen so that the texture pattern aligns between drawing commands.

If any additional data bytes follow, they are ignored.

4.4.6 FILL CONTROL

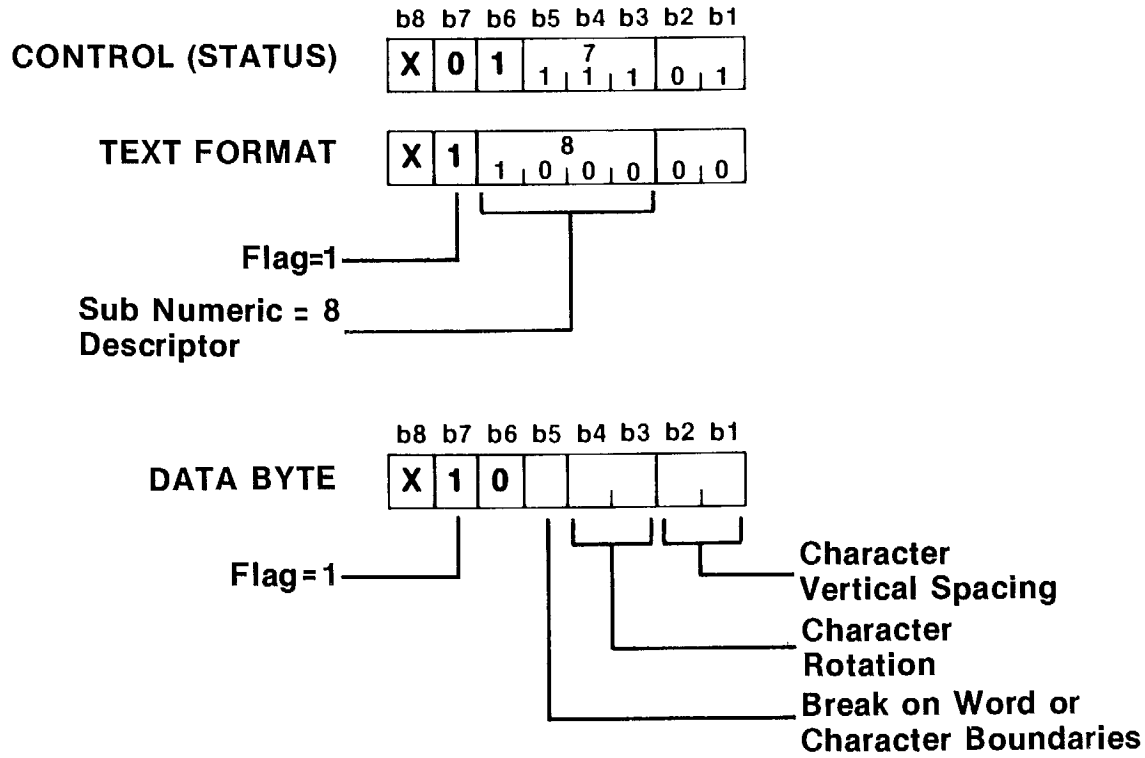


Control (Status) Fill Control

The FILL CONTROL Status Command establishes the highlight mode for filled areas.

If bit b2 is 0, then areas are filled with no highlighting. If bit b2 is 1, then areas are filled and their perimeters are highlighted in black, (colour modes 0 and 1), or with the background colour (colour mode 2).

4.4.7 TEXT FORMAT



Control (Status) TEXT FORMAT

The Control (Status) TEXT FORMAT command allows control over the method in which TEXT is displayed.

Character strings may automatically "wrap around" when the margin of the screen is reached. There are two forms of this wraparound. In the first or default mode (bit b5 = 0) character strings are broken between characters at the end of a character path. In the second mode (bit b5 = 1) character strings are broken between words (see section 4.3.5.10).

Character rotation is specified in a 2-bit field (b4 & b3) which allows for 4 directions of rotation, as shown in Figure A.8.

Bits b2 and b1 specify vertical character spacing in units of character height as follows:

b2	b1	vertical character spacing
0	0	1
0	1	1.5
1	0	2.0
1	1	2.5

4.5 Mosaic

A Mosaic drawing capability is essentially a coarse photographic technique in which each picture element is coded to take on the attribute of a specifically coded shape. There are many possible coding algorithms to define different mosaic systems, however the shape which has been standardized in CCITT Recommendation S.100 is the 2 X 3 sub-element code which makes use of text character cell structure of the display. The algorithm which determines which of the sub-elements is on is derived from the bit combinations of the code table reference.

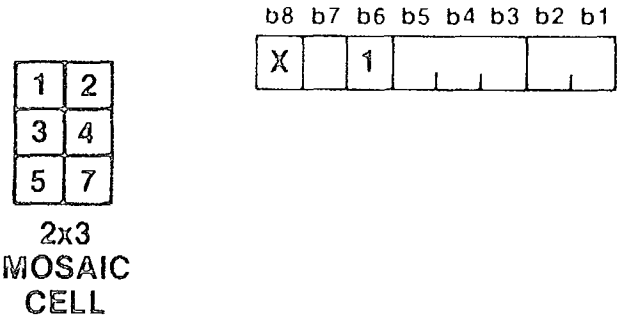


Figure 4.5 Mosaic Sub-element Encoding

The code combination 1011111 also implies all sub-elements on. See Figure A.11 for a code table representation of the mosaic scheme.

The Mosaic drawing capability may be used for those applications for which they are appropriate. It also provides a harmonization with CCITT Recommendation S.100 in that the inclusion of the shape table permits direct transcoding from data encoded in that form. The variable character density feature of the PLP permits the Mosaic code table to be used at various densities such as 40 x 20, 40 x 24, 32 x 16 or other densities, creating compatibility with a large number of standard and non-standard mosaic coding schemes.

4.6 C0 Control Set

This section describes the C0 control set (see Figure 4.6) which occupies columns 0 and 1 of the 7 and the 8-bit in-use tables.

Invocation of other control sets into the C0 set is reserved to future study.

4.6.1 NUL

Null (0/0) may be used as a fill character. It has no effect upon the terminal.

4.6.2 Transmission Control Codes

The transmission control codes; i.e., SOH (0/1), STX (0/2), ETX (0/3), EOT (0/4), ENQ (0/5), ACK (0/6), DLE (1/0) NAK (1/5), SYN (1/6) and ETB (1/7) are reserved for use at protocol levels lower than the presentation level.

4.6.3 BEL

The Bell (0/7) character is used to ring a bell or sound a beeper within the terminal.

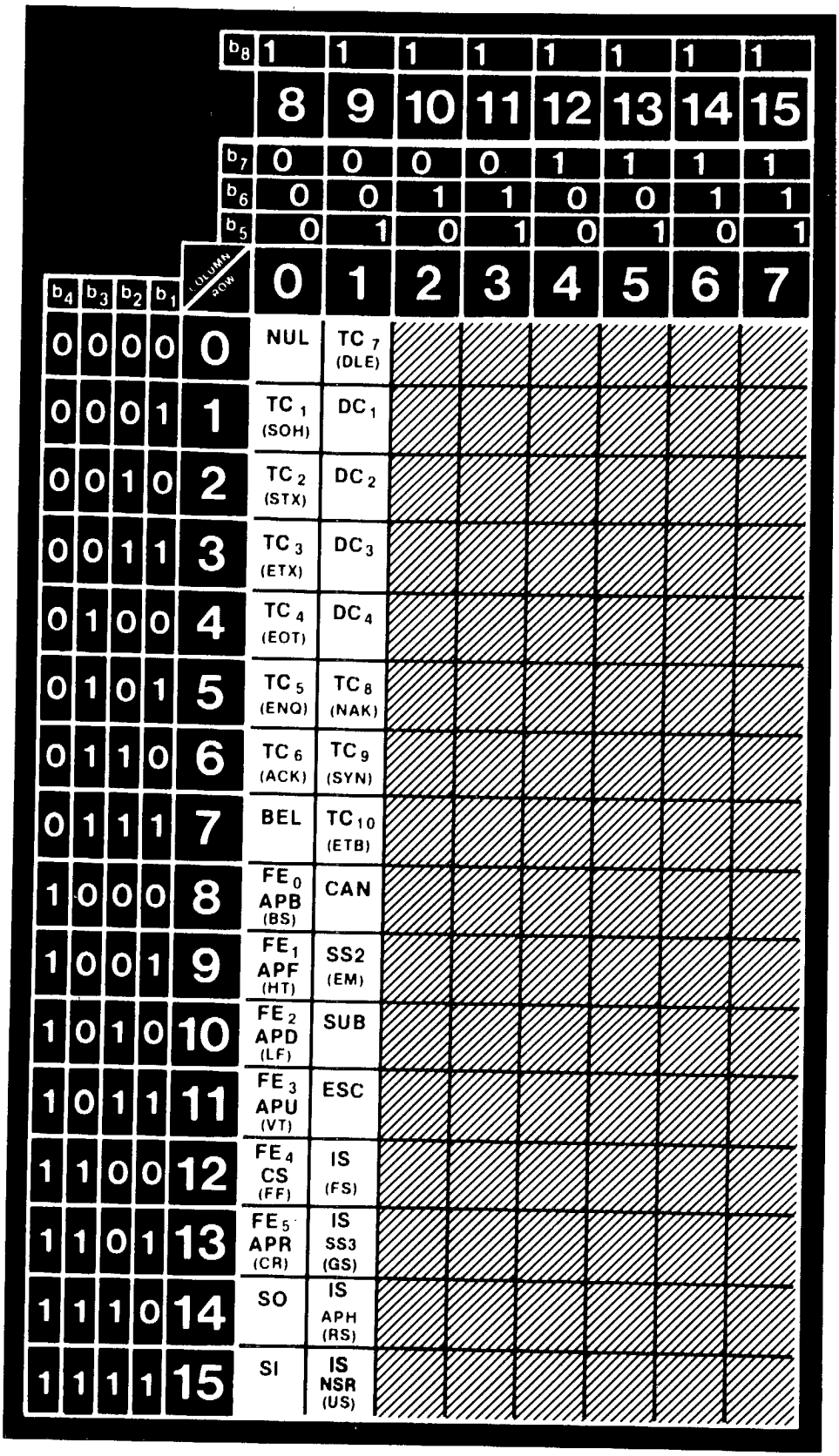
4.6.4 Format Effector Characters

4.6.4.1 APB

Active Position Backward (0/8) or Backspace (BS) is used to position the text cursor a distance equal to the dimension of the character field lying parallel to the character path in the direction opposite to the character path (i.e., 180° from the direction of the character path). If such a movement would cause the edge of the unit screen (or active drawing area should the cursor lie within such an area) to be crossed, then the cursor is instead positioned at the opposite edge (along the character path) of the screen or active drawing area and an automatic Active Position Up (APU) is executed.

4.6.4.2 APF

Active Position Forward (0/9) or Horizontal Tab (HT) is used to position the text cursor a distance equal to the dimension of the character field lying parallel to the character path in the direction of the character path. If such a movement would cause the edge of the unit screen (or active drawing area should the cursor lie within such an area) to be crossed, then the cursor is instead positioned at the opposite edge (along the character path) of the screen or active drawing area and an automatic Active Position Down (APD) is executed.



b_8	1	1	1	1	1	1	1	1
	8	9	10	11	12	13	14	15
b_7	0	0	0	0	1	1	1	1
b_6	0	0	1	1	0	0	1	1
b_5	0	1	0	1	0	1	0	1
b_4 b_3 b_2 b_1 / COLUMN / ROW	0	1	2	3	4	5	6	7
0 0 0 0 0	NUL	TC_7 (DLE)						
0 0 0 1 1	TC_1 (SOH)	DC_1						
0 0 1 0 2	TC_2 (STX)	DC_2						
0 0 1 1 3	TC_3 (ETX)	DC_3						
0 1 0 0 4	TC_4 (EOT)	DC_4						
0 1 0 1 5	TC_5 (ENQ)	TC_8 (NAK)						
0 1 1 0 6	TC_6 (ACK)	TC_9 (SYN)						
0 1 1 1 7	BEL	TC_{10} (ETB)						
1 0 0 0 8	FE_0 APB (BS)	CAN						
1 0 0 1 9	FE_1 APF (HT)	SS2 (EM)						
1 0 1 0 10	FE_2 APD (LF)	SUB						
1 0 1 1 11	FE_3 APU (VT)	ESC						
1 1 0 0 12	FE_4 CS (FF)	IS (FS)						
1 1 0 1 13	FE_5 APR (CR)	IS SS3 (GS)						
1 1 1 0 14	SO	IS APH (RS)						
1 1 1 1 15	SI	IS NSR (US)						

Figure 4.6 C0 Control Set

4.6.4.3 APD

Active Position Down (0/10) or Line Feed (LF) is used to position the text cursor a distance equal to the dimension of the character field lying perpendicular to the character path in a direction perpendicular to the character path (-90°). If such a movement would cause the edge of the unit screen (or active drawing area should the cursor lie within such an area) to be crossed, then special action is taken, which is dependent on whether a not scroll mode is in effect (see sections 4.3.5.12 and 4.3.5.13).

4.6.4.4 APU

Active Position Up (0/11) or Vertical Tab (VT) is used to position the text cursor a distance equal to the dimension of the character field lying perpendicular to the character path in a direction perpendicular to the character path (90°). If such a movement would cause the edge of the unit screen (or active drawing area should the cursor lie within such an area) to be crossed, then special action is taken which is dependent on whether or not scroll mode is in effect (see sections 4.3.5.12 and 4.3.5.13).

4.6.4.5 CS

Clear Screen (0/12) or Form Feed (FF) is used to position the text cursor to the upper left character position on the screen. In colour modes 0 and 1, this control code also clears the screen. In colour mode 2 it clears the screen to the current in-use "background" colour.

4.6.4.6 APR

Active Position Reset (0/13) or Carriage Return (CR) is used to position the text cursor to the first character position within the unit screen (or within the active drawing area should the cursor lie within such an area) along the character path.

4.6.5 SO/SI

4.6.5.1 SO

Shift Out (0/14) is used to invoke the G1 set into the in-use table (see sections 3.2.1.1 and 3.2.1.2).

4.6.5.2 SI

Shift In (0/15) is used to invoke the G0 set into the in-use table (see sections 3.2.1.1. and 3.2.1.2).

4.6.6 Device Control Codes

The device control codes; i.e., DC1(1/1), DC2(1/2), DC3(1/3), DC4(1/4) are reserved for use at protocol levels lower than the presentation level.

4.6.7 CAN

The Cancel code (1/8) is used to terminate processing of all currently executing MACRO PDIs. Execution is resumed at the next presentation character following the terminated MACRO call. The effect of the CAN code is immediate, i.e., it is not put at the end of any existing queue of unprocessed presentation level code.

4.6.8 Single Shift Codes

4.6.8.1 SS2

Single Shift 2 (1/9) is used to invoke the G2 set into the in-use table in a non-locking manner (see sections 3.2.1.1 and 3.2.1.2).

4.6.8.2 SS3

Single Shift 3 (1/13) is used to invoke the G3 set into the in-use table in a non-locking manner (see sections 3.2.1.1. and 3.2.1.2).

4.6.9 SUB

SUB (1/10) has no effect at the presentation level.

4.6.10 ESC

Escape ESC (1/11) is used for code extension. See sections 3.2.1.1 and 3.2.1.2.

4.6.11 FS

FS (1/12) has no effect at the presentation level.

4.6.12 APH (RS)

Active Position Home (APH) (1/14) or Record Separator (RS) is used to position the text cursor to the upper left character position in the screen.

4.6.13 NSR (US)

The Non-Selective Reset (NSR) or Unit Separator (US) (1/15) character is used to reset the code environment to a known initial state. In particular it resets the DOMAIN, TEXT, TEXTURE and other parameters as well as colour mode to their default values and establishes the default designated tables and in-use G table. An alternate use of the Non-Selective Reset capability is for compatibility with other Mosaic systems. Non-Selective Reset is used to introduce a special control sequence to position the input cursor. The NSR character is followed by two bytes from columns 4 through 7 only of the code table. Bits b6 through b1 of the first and second bytes define the row and column address respectively to which the cursor should be moved, with row 0 column 0 numbered from the upper left of the display screen. If the two bytes following the NSR character are both not from columns 4 through 7 of the code table, they are ignored and the cursor position remains unchanged.

5. GUIDELINES

The Presentation Level Protocol described in this document provides a coherent flexible coding scheme in which pictures may be encoded in a manner independent of the hardware dependencies of particular terminal configurations. This provides manufacturers with the flexibility necessary in order to develop cost effective products and service offerings. It is expected that ranges of terminals will develop in order to address different segments of the market and to cover related areas. However the existence of ranges of terminals causes a problem for the information provider, because from the information source point of view it would be simpler if all terminals displayed all pictures identically. In practise there needs to be a balance between the freedoms given to an information supplier and those given to a manufacturer. This section outlines some of the implementation considerations and presents guidelines for manufacturers to construct terminals for Videotex and related services, and guidelines for information providers to create data which can be viewed on a wide range of terminals covering many related applications.

Any freedom given to a manufacturer is a constraint to the information provider and vice versa. The nature of the information industry and the equipment manufacturing industry is quite different. In the equipment manufacturing industry it is important to have the flexibility to permit the continued improvement and cost reduction of equipment. Old techniques which may have been cost effective at one time become obsolete and equipment manufacturers would prefer to retire that equipment. The problem of coping with obsolete equipment has traditionally been left to service providers in the data communications and computing industry.

The information provider in the newly emerging electronic publishing industry finds itself in a very difficult position. He/she must acknowledge the fact that the equipment manufacturer needs to have the freedom to innovate, thereby reducing the cost of equipment, because as the cost of terminal equipment goes down the number of potential customers increases and with it increases the potential readership for this information. However the information provider cannot tolerate a fragmented market. If he has to produce different information for different types of terminals the economies of scale in publishing for a mass audience are lost.

The solution of the problem of technological obsolescence in communications systems is to establish flexible communications protocols. It is on this premise that the Videotex Presentation Level Protocol and especially the alpha-geometric Picture Description Instructions were designed. Compatibility with installed inventories of television, Videotex and computer and communications equipment has been achieved by building a defaulting structure into the Presentation Level Protocol.

As an analogy we should examine television systems again. The trans mission protocol has been designed in such a way that a colour television signal defaults to a monochrome picture on a monochrome television set. However an "information provider", in this case an advertiser, who wishes to produce an advertisement for television to sell laundry soap which "makes blues bluer", will have difficulty getting his point across on that percentage of television sets which show only monochrome pictures. The information provider creates information

to address the largest segment of the market he can address with the maximum impact message. This trade off will vary from message to message and will vary with the composition of the market place with time.

Table 5.1 is a chart derived from the CCITT service Recommendation F.300 which illustrates some of the parameters which may vary from terminal to terminal in Videotex applications. As is readily seen, the situation is for more complex than for analog television. Besides the variability of colour, there are many other parameters which may vary between implementations.

The basic principal which must be followed in all implementations is that all of the basic drawing primitives must be implemented. Manufacturers are free to implement any attribute to any level of capability.

There are several different information presentation techniques which have been woven together to form the Presentation Level Protocol. These are the alphanumeric text, and the geometric, mosaic and photographic techniques. It is important that a manufacturer fully implements all the primitives for each technique included in the terminal design. For example, for the text technique of encoding information it is important that the terminal implement the entire ASCII set of characters. It would be disastrous if only the consonants were implemented with no vowels. The supplementary set of accents, diacritical marks and special characters need not necessarily be implemented as they are not basic and accented characters default back to the primary unaccented character forms.

In a similar manner all the geometric primitives, POINT, LINE, ARC, RECTANGLE, and POLYGON, must be implemented in any terminal which supports the geometric mode. Attributes which modify the drawing primitives such as colour, crosshatching etc. may be implemented or may default to simple states. System parameters such as resolution may vary from implementation to implementation, however, the basic drawing primitives must always be available.

The INCREMENTAL POINT primitive can only be implemented on certain classes of hardware and is therefore not guaranteed to always be available. In general this command provides the capability of providing a point by point photographic like mode and is especially useful for transcoding of facsimile or photographic data encoded by other coding schemes.

The mosaic mode makes use of a mosaic shape table, which when implemented should be implemented in its entirety. This mode may be used for those applications for which mosaic is considered cost effective. The full table defined is the union of both of the mosaic code tables defined in CCITT Recommendation S.100 and is therefore most useful for transcoding from other mosaic coding schemes.

A text-only ASCII terminal can be considered to be compatible with the PLP if it also implements the basic procedures of the ISO 2022.2 code extension procedure. This is extremely important for interworking with related text communications and data processing services, however it does not serve as basis upon which to build a Videotex electronic information publishing industry. Certain basic capabilities must be available in every Videotex terminal in order for information providers to have a profile of their readership.

TABLE 5.1

Display attributes
(adapted from CCITT F.300)

Attribute	Degree of implementation: decreasing capability to the right				
Colour	Any number of colours	Any 16 colours displayed at one time, of unlimited choice	6 pre-defined colours + black and white + 6 grey shades	6 shades of grey + black and white	black and white
Flashing	More complex implementations including phase control	Alternate between foreground and background colour	Alternate between foreground colour and black	Alternate between white and black	steady
Line texture	More choices of texture		Dashed, dotted or solid		Solid
Surface filling	More choices of filling patterns	Filled with solid colour or cross-hatched lines		Filled with solid colour	Not filled (outlined)
Character sizes	More choices of size	16 sizes : 8 geometrically similar size each with 2 heights	4 sizes : 2 heights and 2 widths	2 sizes: 2 heights with fixed width	one size
Underline	Underline			No underline	
Character orientation	More choices of orientation		text rotated through 0, 90°, 180° or 270°		No rotation

Every Videotex terminal compatible with the Presentation Level Protocol should implement at least the minimum text and geometric modes, however it is strongly suggested that the basic mosaic shape table be included for compatibility and that the INCREMENTAL POINT be implemented on any terminal which has the hardware to support it.

As a reference level of implementation, in order that information providers can estimate the profile of terminal equipment capabilities available in their readership a reference level of implementation is suggested in which all G sets listed below are included:

- alphanumeric (ASCII) text;
- supplementary text;
- geometric PDI codes table;
- macro PDI name set;
- mosaic code table;
- DRCS name set.

Several basic guidelines for the design of terminal equipment are outlined below:

- All basic drawing primitives for a particular drawing mode must be implemented if that mode is implemented.

- The level of implementation of the attributes which modify the drawing primitives (such as colour etc.) may vary from terminal to terminal, however reasonable defaults must always be provided.

- All parameters for drawing commands or attribute specifications which are specified as continuous variables may be defaulted to specific implementation dependent discrete values, which in the limiting case may be only one discrete value, the default value, for the attribute.

The following sub-sections discuss in more detail a number of topics for which implementation guidelines are required for the manufacturer of equipment or for the information provider.

Although it is difficult to give order to these topics they will be organized roughly in terms of those which affect text, geometric, mosaic, photographic and overall modes.

5.1 TEXT

The complete ASCII text code table including the $ as a monetary symbol must form the basic primitives of the TEXT mode. With due consideration to the cost of implementation of Videotex terminals, a multiplicity of independent character sizes may not be implemented on many terminals. Text size has a continuous variable as a parameter and in many practical implementations, especially those using character generator hardware, will default to a range of discrete sizes. The continuous text specification would map to the nearest available size.

The minimum number of text sizes which should be supported include normal text (40 x 20), medium text (32 x 16), as well as double size for each of these sets and double height for the four sets which result. It is also desirable to support 24 lines of text by reducing font spacing on the 40 x 20 normal size. If a terminal is incapable of handling a text size because it is too small it is up to the manufacturers to default this in the most suitable manner.

Figure 5.1 below illustrates a range of text sizes which can be established based on only two character shape, tables which differ by a factor of approximately 1.5 and the double and half size of each. This provides a psychologically pleasing approximation to a continuous range of character sizes.

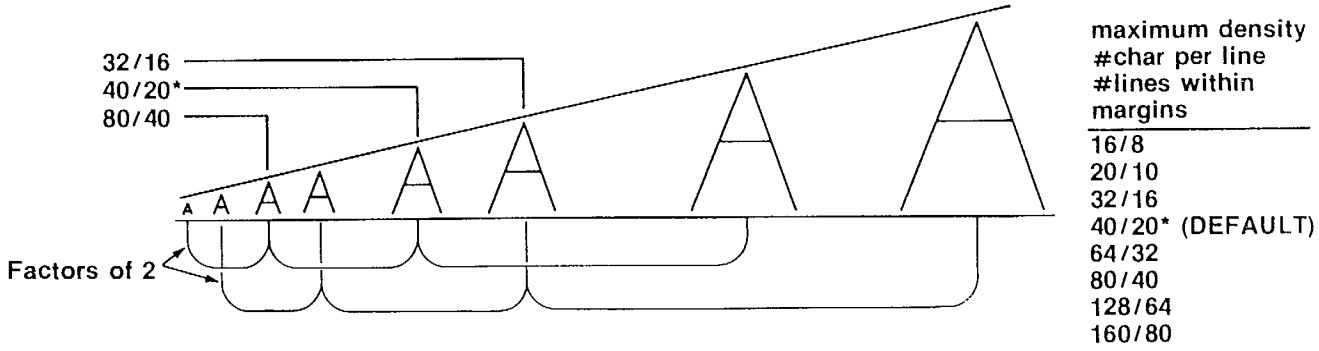


Figure 5.1 - Discrete Text Sizes

For the purpose of text definition by DRCS, scroll algorithms etc., the text character origin is located at the bottom left of the full character font, below any danglers, and not at the base of the upper case character.

Word wrap mode permits words to be kept together by causing a wordwrap to occur at the delineation of a word. The sophistication of the word delineation algorithm is up to the manufacturer and may in some terminals be destined for unilingual special applications and may include syllabic hyphenation. Psychological investigations have indicated that confusion sometimes results from word break on all punctuation characters and further study is required in this area. Word break is considered acceptable if it occurs on a SPACE character end of line or entering drawing mode only. Word wrap should not take effect until the character after the word break character is received.

The supplementary character set of accents, diacritical marks and special symbols cannot be guaranteed on all text terminals, however, it is recommended that for Videotex terminals at least the following list of non-spacing supplementary graphics characters should be implemented:

˙ (4/1), ˚ (4/2), ˆ (4/3), ˜ (4/4), ¨ (4/8), ¸ (4/11)

In addition, these spacing characters should be implemented:

¡ (2/1), ¿ (3/15), « (2/11), » (3/11)

The C1 control set permits several attributes to be defined for text display which may not be available on all forms of hardware. This is especially true of those terminal implementations in which hardware character generators are used and where these features may not be available in the hardware. For some terminal implementations the capability to alter character rotation, character path, inter character spacing and inter-row spacing from the default values may not be available for all character sizes or may not be available at all.

It is good practice for information providers to restrain from the use of these special text capabilities except for those situations where they are used to simply embellish the information and where the default effect on a terminal would be acceptable.

A cursor capability is not trivial to implement on some types of display hardware which make use of an inexpensive slow microprocessor. Also with the presence of continuous scaling the cursor can be of any size. This may be impractical and uneconomical on some types of hardware character generators. The cursor capability should be implemented if possible, however its prime purpose is as an aid to interactive input. This feature should not be depended upon by the information provider in designing application level interaction dialogues. Similarly either the underline capability or the reverse video capability may not be available especially on some character generator based display apparatus.

5.1.1 DRCS

The DRCS (Dynamically Redefinable Character Set) capability relates closely to the Text capability. However, discrete character sizes do not necessarily imply the same discrete DRCS sizes. In a minimum configuration terminal which implements DRCS it is expected that only one DRCS size may be definable at one time.

The manner in which DRCS characters are defined is independent of the physical display hardware. The Picture Description Instructions or other drawing mode is used to define a picture of the shape of the DRCS character in terms of the full 0 to 1 range of the coordinate system; that is, the 0 to 1 number system related to the DRCS character cell. The DRCS character cell memory is effectively a small coarse bitplane memory into which a picture of the character is drawn as if it was a full size bit plane memory. The character field command is used to define the aspect ratio and coarseness of the character definition; for example, a character which is 10 pels high and 6 wide would be defined on a coordinate system of 0 to 1 in the Y axis and 0.0 to 0.6 in the X axis.

Since the DRCS character is stored as a bit pattern at low resolution it is not possible to scale the character at the time it is displayed on the screen, except by factors of 2, without encountering quantization error which would introduce distortions. For this reason many terminal implementations would only support character size scaling in factors of 2 for DRCS.

DRCS characters may be defined at any font size within specified implementation limits. It is suggested that the font limits be selected such that a full 96 character set may be defined within an amount of RAM likely to be available in most terminals. A minimum DRCS storage memory of (8 x 16 physical maximum definition = 16 bytes/character =) 1536 bytes is recommended.

If the character is defined in a font which is larger than the specified limit, the font would be reduced by factors of two until it lies within the given limits and the mapping will take place in the reduced font.

The memory available within any particular terminal implementation is limited and would most likely be shared between such memory intensive functions as MACRO and DRCS. At a minimum the Information Provider would only be able to assume that sufficient memory was available for DRCS or MACRO or other memory intensive functions and that using these functions simultaneously would reduce the amount of memory available for the other functions. DRCS should not be used to define pictorial images by defining shapes instead of characters, unless the full continuous size character scaling is implemented, because hardware dependencies would result.

5.1.2 Unprotected Fields

Unprotected fields are basically a special form of record activation. The intent of unprotected fields is to provide a data entry facility. The full requirements of unprotected fields require keeping track of anything drawn through an unprotected field. There can be any quantity of data stored within the unprotected field within memory limitations of the terminal.

The following restrictions are acceptable:

a) An unprotected field may be limited to contain only text entered from the terminal keyboard or keypad or received from the host.

b) Once data entry is started in a particular unprotected field, the character size and orientation is fixed.

c) Characters will be drawn in fixed character positions. These character positions are defined to align with the size of text at the time of definition of the unprotected field.

d) Local editing functions on some terminals may be limited to a simple line delete -- clear entire entry; or a rub out -- erase single character.

For certain services it is necessary to be able to establish the default for field protection and also the form of transmission of the field dimensions to the host. This must be defined within the protocols for other layers.

5.2 Geometric Mode

For Videotex terminals the geometric mode should always be implemented as it forms the basis of the flexible hardware independent code structure. The basic primitives of POINT, LINE, ARC, RECTANGLE and POLYGON must always be implemented in order to provide Information Providers with a consistent set of tools with which to build information pages. Only the two dimensional mode is currently defined for geometric commands. In order to allow for the graceful introduction of a three dimensional capability without rendering obsolete existing apparatus, it is suggested that three dimensional commands be interpreted as two dimensional commands with the Z axis ignored.

5.2.1 Spline Form of the ARC Command

The spline form of the ARC command provides the capability of drawing a smooth arc like curve through a series of points. Further study is required to determine which spline algorithm is best suited, however the B-spline algorithm discussed in Technical Note 689 can serve as a suitable starting point for current implementations. In a simpler terminal which does not implement the SPLINE form of the ARC command the spline case should effectively be treated as a polygon.

The filled spline is treated in the same manner as ARC in that to close the spline curve, a straight line is draw from the start to the end point. Unfilled spline is treated as a series of concatenated lines, but is not closed.

5.2.2 Logical Pel

The LOGICAL PEL capability permits points, lines and other primitives to be drawn at various widths as well as providing the necessary information for defining the INCREMENTAL POINT logical pel size and the texture pattern size. A drawing primitive is defined by an algorithm which follows as closely as possible a standard geometric path. For example a LINE is a locus of points following a straight line algorithm between two specified coordinates. The physical picture elements (Pels) through which the infinitely small locus point pass would be turned on. The LOGICAL PEL specification allows the locus point to take on specific dimensions thereby acting as a wider "brush" which turns on additional pixels as it traverses its geometric path and generates the effect of line width.

In minimum configuration terminals it may not be cost effective to perform the necessary calculations to maintain uniform line width on arcs or even in some instances to support the line width attribute. The LOGICAL PEL definition should, nonetheless, be recognized and interpreted as it provides information which is needed for other drawing attributes.

5.2.3 Fill

The capability to fill a polygon or other geometric shape is a feature intrinsicly dependent upon the type of display hardware upon which the terminal apparatus is constructed. For Videotex applications it is expected that the fill capability would always be available, however for many computer graphic applications the fill capability may have to default to an outline or some other default state on a caligraphic, storage tube or other display apparatus. Information providers should consider the fill feature to always be available in Videotex systems.

The capability to highlight a filled geometric shape with a contrasting or additional outline is a simple useful feature which is expected to be available in Videotex terminals. However both the highlight feature as well as the capability to fill with textured patterns may not be implemented on extremely low cost and possibly low resolution terminals.

5.2.4 Texture Masks

For those terminals which implement the texture mask capability, texture sizes would be defined in a similar manner to DRCS although the maximum dimensions may be slightly larger.

It is desirable for harmonization that there be eight default texture masks, the last four being represented by:

5. a dot pattern (precise pattern to be defined by the manufacturer);
6. 45 degree lines with positive slope;
7. 45 degree lines with negative slope;
8. 45 degree crosshatching.

The last four patterns may be redefined by the host and are not affected by logical pel size. The default patterns would be restored on receipt of a texture reset command. This provides harmonization with the CCITT S.100 specification, and the Telidon field trial PDI specification as represented in Technical Note 699. These four default harmonization patterns are not guaranteed on all terminals which implement the PLP, nor is the capability to redefine these patterns guaranteed on minimum implementations. Some minimum implementations may not support the texture attribute to any more than its default value so that fill patterns are either solid or unfilled. Terminal manufacturers should endeavour to ensure that different texture at least produce distinguishably different results.

5.2.5 INCREMENTAL

The INCREMENTAL commands provide a short hand method of describing specific highly repetitive operations. These features essentially do not use any additional display resources than the basic geometric drawing commands of POINT, LINE, ARC, RECTANGLE and POLYGON; however, there is an important practical difference. The Incremental commands have their primary use in encoding a large number of adjacent drawing entities, that is, adjacent point in an INCREMENTAL POINT command in a photographic like manner or adjacent short lines in the INCREMENTAL LINE command or short sides of a polygon in an INCREMENTAL POLYGON command. The Incremental commands only have practical use where sufficient resolution is supported in the terminal hardware. On low resolution terminal implementations or teminal designs which cannot support independent adjacent colours, the Incremental commands may not be implemented as they would have no practical use. The INCREMENTAL LINE and POLYGON commands are an efficient way of encoding short lines and short sided polygons; however, they do not provide additional display facilities. In particular the same physical limitations on the LINE and POLYGON command apply to the INCREMENTAL LINE and POLYGON commands.

5.2.6 Calculation Errors

The error in calculation of the geometric path or the coordinate end positions should never be more than + of a physical pel for a given terminal apparatus. This is in order to prevent accumulation of errors for concatenated commands. An exception may occur for a circular arc command where additional error may be tolerated (in the geometric path but not in at end points), in order that the circular arc appear more circular.

5.2.7 Filling of Polygons

Because the geometric coding scheme is independent of resolution it is possible to build terminals which display pictures to any level of resolution. Truncation of least significant bits will cause some minor perterbations in the shape of the geometric entity; however, this error should be less than 1/2 pixel in magnitude. A special case arises in filled polygons, where a valid polygon with a unique inside and outside may be filled at one resolution; but, at a lower resolution, the narrow channel may close (have 0 width) due to truncation and round off error. The effect is exaggerated by the use of the scale operation on information provider equipment. A good practise guideline for information providers is to take care not to produce such polygons. However, a terminal should be able to fill a polygon which contains a topologically single inside area comprised of several sub areas joined by narrow 0 width channels.

5.2.8 Maximum Number of Polygon Vertices

The maximum number of vertices permitted to describe a polygon is related to terminal buffer size. It is suggested to limit the maximum number of vertices in a polygon to 256.

5.3 Mosaic

The Mosaic shape table illustrated in Figure A.12 has been provided for those applications where Mosaic coding techniques may have some application and especially to permit transcoding of data bilaterally through gateways. The Geometric drawing commands can easily be transcoded into Mosaic drawing commands albeit with some degradation in resolution. The Mosaic shape table in the PLP makes transcoding in the other direction possible. The capability of positioning the text cursor by a control character sequence using the IS4 (US, position 1/15) is included for compatibility with other Mosaic schemes. It is not recommended the Information Providers make use of this command as the imbedded reset will cause compatibility problems.

5.4 Overall Modes

Several of the implementation and usage guidelines which cover several modes of operation are discussed below.

5.4.1 Resolution

The physical resolution of the display apparatus in spatial, colour and other dimensions often defines the level of capability which can be implemented in a given terminal and is dependent upon the techniques used to construct a terminal. Freedom from hardware dependence in resolution is one of the prime design criteria of the Presentation Level Protocol. When resolution is decreased obviously fine detail is lost. An Information Provider must take care not to place a lot of essential information content in fine detail.

5.4.2 Colour

The dimension of colour, like resolution is continuously variable and it may be implemented in a terminal by discrete states. The simplist implementations make use of only the six colours of RED, GREEN, BLUE, MAGENTA, CYAN and YELLOW plus BLACK and WHITE. This can be achieved very simply electronically by simply grounding the cathodes for each electron gun on the television picture tube, however it does not provide a very rich range of colours with which the Information Provider can work.

Addition colour capability can be implemented in a terminal by several techniques, the most powerful of which is the colour map (colour look-up table) technique which has been used for some time in the computer graphics industry. The PLP provides the capability of defining colour in three ways, all of which can make use of colour look up tables (LUTs).

Colour maps may be utilized in two ways in the Presentation Level Protocol, that is the colour map locations may be defined implicitly as colours are made current in colour mode 0 or they may be referred explicitly by colour map address in colour mode 1. The prime purpose of accessing the colour map explicitly is to manipulate it and a display effect such are animation by substitution, which manipulates the colour map, cannot be defaulted for a terminal which doesn't implement a colour map. It is therefore suggested as a guideline to information providers to use colour mode 0 to define general drawing colours and to use colour mode 1 to define colour states for manipulation or other special colour map dependent purposes.

Colour Maps may not be implemented by all terminal hardware, a fact which must always be kept in mind by information providers. If colours are defined in colour mode 0 then the colour map (LUT) can be used implicitly if it exists or a default colour selected. The SET COLOUR and SELECT COLOUR commands in colour mode 1 and 2 permit the colour map to be explicitly defined and modified. On terminal implementations without colour look up tables, manipulation commands to explicitly manipulate colour map entries will have no effect. Information Providers should remember that this will be the case and to design their pictures accordingly. Because of the implicit Black in the RESET command and the default White drawing colour, Black and White or their equivalent positions in the colour LUT are always defined. If these positions in the table are overwritten, then those colours will change.

It should be noted that the algorithm for colour map definition described in appendix E is just for convenience, as these colour values are not guaranteed to be available on all terminals.

Colours are specified as binary fractions of the intensity range for the primary colours Green, Red and Blue. In order that the truncation process of least significant bits not bias the colour range to the less saturated shades, it is desirable that a manufacturer normalizes colour intensity fractions to the number of bits transmitted when a partial fraction specification occurs.

5.4.3 Blink

There are two forms of the BLINK command which primarily serve two different purposes. The basic form of the BLINK command is really a timed manipulation of the colour map. This command is very powerful in creating special effects, however it may not operate on all terminals because of its dependence on colour map hardware.

The other form of the BLINK command is primarily used for attention gathering. This form of blink therefore causes all subsequent information to flash at a terminal dependent fixed rate. The manipulation form of blink requires the use of a colour map and refers to all pixels of a given content, whether they are drawn before or after the blink process is put into effect. This form is especially useful for creating animation sequences and it permits the definition of phase delay and timed intervals so that semi-animation effects may be created. In some implementations the to/from colours will be accepted but the on/off intervals and phase delays will not be processed.

Both the BLINK control command and the C1 form of the BLINK command permit the explicit manipulation form of the Blink command to be specified, however in colour mode 0 the C1 form of the BLINK command defines the simple attention gathering form of the BLINK command where all subsequent drawing material is drawn in a blinking manner until another COLOUR command or BLINK command is specified.

Specifically what occurs in this case is that in colour mode 0 a C1 BLINK command creates a implicit new blinking colour or a reference to an existing blinking colour. This provides harmonization with S.100 and the Telidon field trial specification 699, and may in many simple implementations be the only Blink capability available. The C1 BLINK OFF command is used to stop a Blink process from flashing on and off, not to terminate the range of a BLINK command.

5.4.4 Scroll

The capability to SCROLL parts or all of the display screen is a feature which has great utility but which may not be very efficient on some terminal implementations. A software scroll pel by pel on a bit plane memory may be unacceptably slow, however no faster techniques may be available on some forms of hardware.

Some terminal manufacturers may wish to implement a locally controlled page mode. Page mode would be implemented by a repositioning of the drawing point to the opposite corner of the active drawing area. Page mode is assumed for "non scroll" operations and in cases where a scroll cannot (will not) be performed (i.e. some non full screen scroll).

Scrolling may only be supported on a full screen basis on some implementations and it is expected that only vertical scroll hardware would exist on many terminal implementations. Information providers should expect that the only efficient scroll operation to exist for a large number of terminal is vertical scroll and even that is not guaranteed. However, it is guaranteed to Information Providers that in all cases even in partial screen scrolls, either a scroll operation or a page mode operation will be available.

It is a good practice for an information provider to verify that the number of characters drawn will fit on a line of a specified text size in order not to cause any inadvertent scroll or page operations. This is especially true when proportional spaced text of non-standard size is used with the wrap-around feature. It is a good practice for the Information Provider equipment to indicate to the user when defaults have been applied (i.e. when a font has been mapped to a default character size).

To avoid scrolling due to a CRLF sequence on the last line on the screen, wrap-around (execution of the CRLF or new page) does not occur until an attempt is actually made to draw past a character border. Because of the variation in terminal hardware the default state for SCROLL is left to the manufacturer.

5.4.5 MACRO

The MACRO feature provides the capability of encoding sequences of PLP data to be executed upon command. The MACRO feature is a very useful one, however it requires storage in the terminal which may not be available on very simple terminals. For those terminals which support the MACRO and the associated transmit MACRO and Programmable Function Key features it is suggested at least one kilobyte of memory be made available for macro storage in a shared manner with the Texture Mask feature DRCS and other memory intensive functions. It is suggested that the high-numbered MACROs be reserved for general MACRO usage and the low-numbered MACROs be used for programmable function keys and TRANSMIT-MACROs.

The MACRO capability is very powerful but must be used with care by the information provider. It is possible to establish sequences in which MACROs delete themselves, invoke themselves or designate other tables to replace the inuse MACRO name table. All of these situations can be legal under certain circumstances so it is impossible for the terminal to test for such cases or even for looping situations. The information provider is advised to take care in using the MACRO capabilities. The terminal manufacturer is advised to implement MACROs by chained return addresses rather than stacked return addresses, in order that looping, in which a MACRO invokes itself directly or indirectly, would not cause catastrophic results.

5.4.6 Border

The Presentation Level Protocol provides the capability of establishing a colour for a border area around the 0 to 1 unit drawing screen. This border capability is completely optional and when implemented it may not have the full colour range of the rest of the display. It is suggested, however, that Information Providers define border colours to produce more pleasing pictures on those terminals which support the feature.

5.4.7 Errors

Errors should be indicated to the user, however, the exact type of error indicator(s) is terminal dependent. The terminal software may indicate not only transmission errors but macro definition errors, and identify when buffer areas are full.

5.4.8 Interaction Dialogue

Several factors affecting the interaction dialogue are discussed in Appendix C. Although the data which is transmitted to the terminal and to the host must comply with the Presentation Level Protocol, there are several other factors which must be considered in design of terminal apparatus. It is necessary to provide some guidance in this area to terminal manufacturers so that terminals can be constructed for both the Videotex service and the many related services.

In order to be compatible with regular business data terminals there should be no distinction made between text oriented input devices. Also the communications channel from the user through the terminal to the host should be or appear to be independent of the channel from the host through the terminal and display to the user. The ASCII keys of the keypad transmitted to the host should be a subset of the standard ASCII keyboard. There should be no difference between what is received by the terminal as an input character from either the keypad or keyboard. Local function keys are additional to that.

If a break key is supported, it should be a standard break key; i.e., it should cause transmission of a binary 0 for time t, where t is between 150 ms and 200 m sec.

In record activation, processing may be halted when a key is struck, echoing performed and processing continued when an activation character is typed. If a terminal is capable of supporting drawing and echoing simultaneously, it may do so.

A "Loop Back" feature where the data stream transmitted from the terminal to the host computer is automatically, character by character echoed to the terminal is a feature provided by several types of communications networks and interface devices in an attempt to provide a simple form of local echo capability.

Loop back (whether locally or remotely generated) may cause the current host-terminal data stream to become distorted, due to the potential mixing of data from the host and echoed data. Because of the nature of the ISO 2022 code extension technique this could produce erroneous results. Information service operators are advised to configure systems to avoid the use of a "Loop back" feature.

5.4.9 Code Extension Techniques

The ISO Standard 2022, is currently a draft international standard, and although no changes are expected, it is possible that minor alignment with other standards may be required. Some of the invocation and designation sequences have not yet been registered and therefore there is some doubt. In particular the LS2R LS3R invocation sequences are under some debate. It would be prudent for terminal manufacturers to implement the codes given in this document as well as the current alternate proposals.

6. EXTENSIBILITY

Great care has been used in the design of the Picture Description Instructions to free the coding scheme from the constraints of particular hardware implementation techniques, so that more sophisticated terminals may be introduced in the future. Room to grow has been deliberately designed into the system with respect to the resolution, speed and sophistication of the terminal.

It is impossible to attempt to predict all future avenues by which a Videotex service can be extended, but it is necessary to provide means by which extensions can take place.

The code extension techniques described in section 3.2 allows alternate G sets of character codes and C (control) codes to be defined. The code tables currently available in the PLP are fully defined; that is, there are no code positions in the table which are open to reinterpretation from implementation to implementation. The manner in which the PLP can be extended is by adding whole new G and C sets.

Some of the avenues of expansion are outlined below. Additional character sets may be added to terminals as defined primary and supplementary sets for such languages as Inuit, Greek or Arabic. Although this functionality can be achieved by using the DRCS capability, it will be advantageous to add such code tables to terminals for particular regions or applications.

A feature which could reduce the load on the communications component and central data base computer of a Videotex system, is the ability for a terminal to receive a number of pages of information in a block transfer and to examine this "magazine" off-line from the central computer. A terminal would require secondary storage for pages composed of PDI commands. Current developments in bubble memory and other secondary storage media make this an attractive avenue of development for Videotex systems.

An enhancement of the above feature would be the off-loading of even more tasks to increasingly more sophisticated terminals. The term "telesoftware" has been used to describe the dynamic loading of programs into terminals. This would have particular application to Computer Aided Instruction and video games. Great care must be taken to ensure that any "telesoftware" programming system be compatible for all types of terminals. The definition of a microprocessor independent, low level, efficient code for telesoftware is of prime importance for general access. Recent developments in computer science have led to the emergence of efficient pseudo machine languages which could form the basis of a universal telesoftware protocol which is completely microprocessor independent.

A far more sophisticated feature for a home or office Videotex terminal is the capability of communicating directly from terminal to terminal. The Communications Research Centre has been involved in research in this area for some time and has identified techniques by which only a minimum amount of information needs to be communicated between the terminals. To this end, Picture Manipulation Instructions have been suggested as a portion of the CONTROL commands, so that pictures may be modified simultaneously at many terminals.

In many applications it is necessary to be able to do more than describe a picture and present it on a display screen. Many applications require that a picture be interacted with and that updates, erasures and modifications be made to the displayed picture. In the Videotex situation it is necessary to be able to generate pictures for the display screen in the home. The generation and modification of displayed pictures is normally the business of an information provider organization such as a newspaper company etc., but the function could also be done in the home in some situations, thus making every man his own publisher. In many industrial display applications the manipulation and updating of a previously displayed picture is a very common operation.

Picture Manipulation Instructions (PMI) are under study to allow a file of descriptive commands to be established in the display terminal so that selective erasure by undrawing or other manipulation may be performed on the picture displayed on the screen.

An extension which is receiving considerable study both in Canada and elsewhere is the addition of a Photographic mode to a Videotex service. This would permit photograph like (or facsimile like) point by point defined pictures to be transmitted along with geometric, mosaic or textual data. The INCREMENTAL POINT capability provides a rudimentary Photographic feature which is desirable in terminals in order to permit simple photographic images to be drawn and to allow for transcoding from a more sophisticated scheme. The design of a more comprehensive photographic specification is still for further study, however, it appears that hardware independence is possible by basing the photographic picture element array on the 0-1 unit screen principle. The flow of data may be best encoded as a binary stream of data without reference to the character coded format used for other coding schemes. An Escape sequence could be used to enter photographic mode and a special bit combination flag used to return to character coded mode. The compression of photographic data by techniques such as transform coding is currently under study, however no clearly superior transform technique has yet emerged.

7. CONCLUDING REMARKS

Electronic Publishing is developing in society as a new and unique information media which will bring with it changes in the amount of information available to the individual. This may be as revolutionary to society as the invention of the printing press or the development of the telephone, radio or television. In effect it brings together the pictorial capabilities of television, the information storage capabilities of the computer and the universal communications access of the telephone.

Videotex is but one part of the emergence of Electronic Publishing. Rapid advances are occuring in parallel in data packet communications, in transactional data services, Teletex, in advanced telephone switching systems, in video storage (video disk and tape), in personal microcomputers, in fibre optics, in direct satellite to home broadcast and in many other areas which will all affect the way one receives information about the world around him.

At the same time as new services such as Videotex are developing the technology upon which they are being constructed is rapidly changing. Dramatic decreases in the cost of electronic memory is rapidly making the bit plane memory technique for construction of terminal apparatus the most cost effective means. The key to handling such extensive change in the developing electronic information market place lies in the flexibility of the Presentation Level Protocol. Since the standard is based on mathematical principals rather than on the hardware capabilities of particular display apparatus various service scenarios on terminal configurations are possible. In this way the coding of information can be done in a common manner for all Electronic Publishing applications. The Presentation Level Protocol will form the base of the information industry.

Chapter 5 of this document presented a number of guidelines which illustrated how information providers or equipment manufacturers could make use of the flexibility of the PLP. Information providers may feel that the flexibility is too great during the emerging stages of the Videotex industry. On the other hand it is necessary that the full flexibility be available so that minimum cost terminals can be made available and so that the many related services can be accommodated. The capabilities which are required for introductory broadcast and interactive Videotex services must be developed between the information providers, the service operators and the equipment manufacturers.

The major problems which continue to exist in the emergence of Electronic Publishing services are not the problems of how to code data but are the more practial problems of how to generate it, how to store it and how to distribute it. Many field trials are currently underway in Canada to test the methods of distributing Videotex information. Much more study is necessary on how to organize data banks and integrate electronic means into the existing news and information gathering media. One of the most important areas which need study is pricing policy. There are very few existing Videotex services operating in the world today however there are some major differences in pricing policy which are worth examining as these services develop. It seems that public acceptance is far greater for services priced on a subscription basis or paid for through advertizing than on a pay per page basis.

As the new electronic publishing media develop they will effect social changes which are of great importance to society. Government policies need to be developed to maintain privacy for the individual and establish universal availability. Advertizing tax write off and other policies will require study in order to establish Canadian content in the domestic data generation market and to control transborder data flow. The social and policy areas of the Electronic Publishing revolution require urgent study so that these emerging services will develop within an equitable social structure.

APPENDIX A
DRAFT STANDARD

This section of the document is intended to provide a succinct specification of the Presentation Level Protocol in a manner suitable for forming the basis of an International Standard. The style of this section is modelled along that of the CCITT Recommendation S.100. No extraneous detail is provided and there is no explanation of material covered in other related standards such as the ASCII text standard, ISO 646, ISO 2022.2 and ISO 6937. The material in this section encompasses the specification for alpha-geometric Videotex as well as a unified graphic code table for alpha-mosaic Videotex as described in CCITT Recommendation S.100. A narrative specification is presented in section 4 covering the same material.

A.1 General

The Presentation Layer Protocol is a set of code operating in either the 7 or 8 bit environment. The code is defined to include the code table positions 2/0 and 7/15 within the definitions of the individual code tables. TEXT is defined by CSA Standard Z 243.4-1973 (identical to ASCII) and is designated by ESC I 4/2 where I defines the table G0, G1, G2, G3 to which the code set is designated. The other presentation modes available are the geometric Picture Description Instructions, a table of Mosaic shapes, a Macro subpicture capability and a Dynamically Redefinable Character Set (DRCS) capability for defining alternative text code sets.

A.1.1 Description of alpha-geometric

In the alpha-geometric coding, the display is composed of pictorial drawings that are defined in terms of geometric primitives transmitted to the terminal as drawing commands.

A.1.2 Designation of geometric codes

The designation of the geometric code table into a G set is the sequence ESC I 5/7.

A.1.3 Geometric primitives

The pictorial coding scheme is based on geometric primitives. Each drawing primitive is specified in terms of cartesian coordinates to describe the positions, end-points, or vertices of each drawing operation. Geometric drawings are defined in terms of the drawing primitives: POINT, LINE, ARC, RECTANGLE, POLYGON, and INCREMENTAL.

A.1.4 Drawing position

Drawings are positionally independent, therefore drawing primitives may overlay each other redefining the drawing at that position.

A.1.5 Coordinate system

The coordinate specifications are defined based on a Cartesian 0 to 1 numbering scheme.

The numbering system is referenced to the display screen and consists of coordinates ranging from 0 to 1 in both X and Y axes, with coordinate values being specified as fractions of this range. The coordinates are encoded in two's complement notation and specified as signed numbers to a minimum accuracy of 3 bits, including the sign bit. Increased accuracy is obtained by additional increments of 3 bits. Unused least significant bits are truncated when the coordinates are defined to a greater accuracy than can be handled by the terminal.

Display screens with non-square visible areas map into the square drawing area number system so that the origin (0,0) remains in the lower left hand corner within the screen margins. On a television-like display with a 4:3 aspect ratio, this corresponds to a range of 0 to 0.99... in the X axis and 0 to approximately 0.75 in the Y axis. Drawing commands addressing the entire square 0 to 1 grid are permissible but only the inscribed 4:3 area is visible.

A.1.6 Picture resolution

Any number of physical picture elements may be implemented. Hence, picture resolution is at the discretion of terminal manufacturers.

A.1.7 Logical Pel

The logical pel is a geometric construct which defines the size of the drawing point. This affects the stroke width of graphics primitives and the size of the dot for the INCREMENTAL POINT and POINT commands.

A.2 Drawing commands

A.2.1 General

Drawing commands consist of Operational codes (opcodes) and their associated data parameters.

Opcodes describe the types of drawing operation. Following the Opcode byte are one or more blocks of additional bytes of data to describe one or more (X,Y) coordinate positions. Each block of data for the (X,Y) coordinates normally (by default) contains 3 bytes (9 bits accuracy), however from 1 to 8 bytes may be used depending on the degree of resolution desired.

Figure A.1 describes the code table for the opcodes and numeric data bytes.

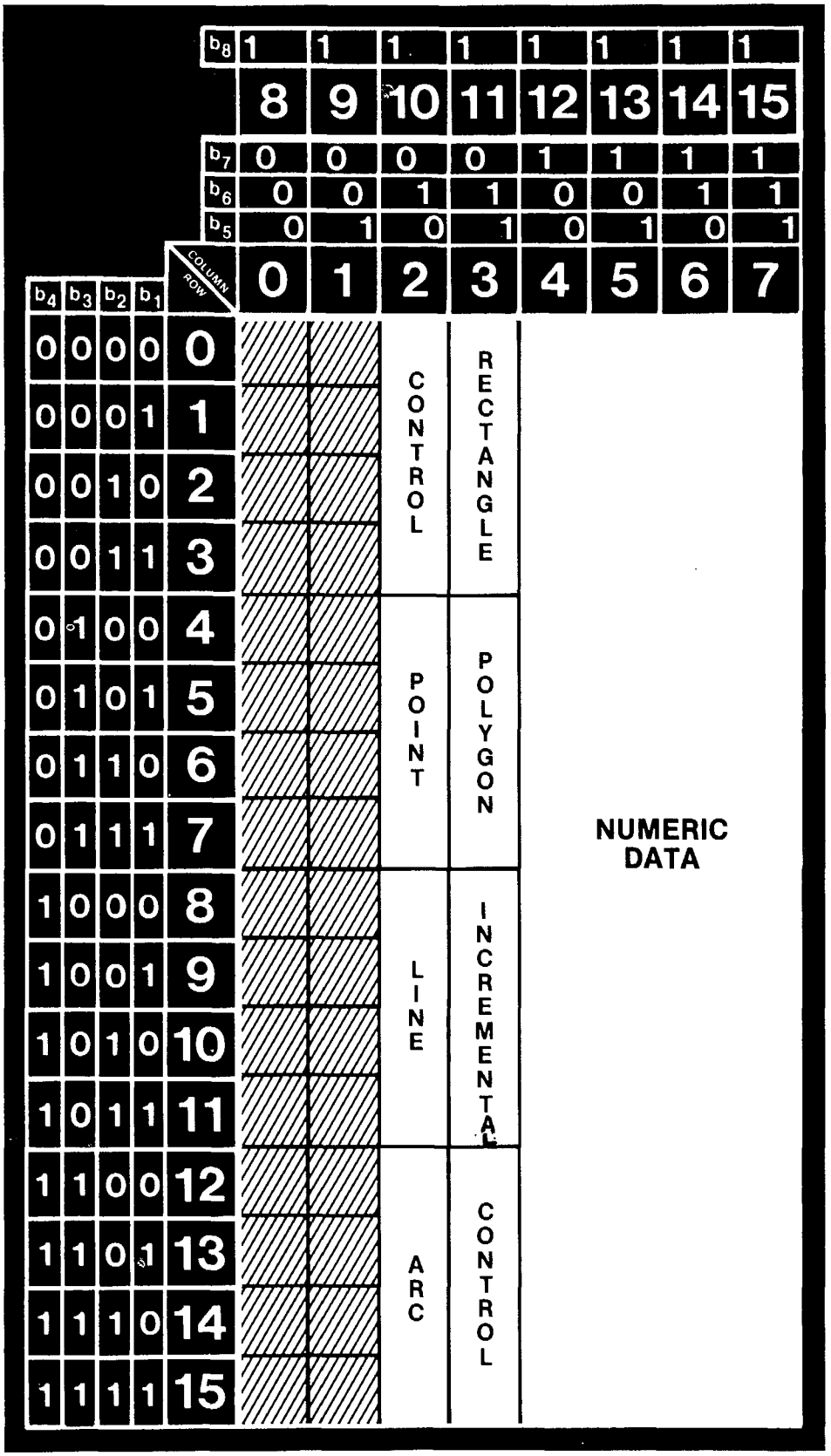


Figure A.1 Operation Code and Data Field Assignments

A.2.2 Opcode byte

The structure of the Opcode byte is as shown in Figure A.2.

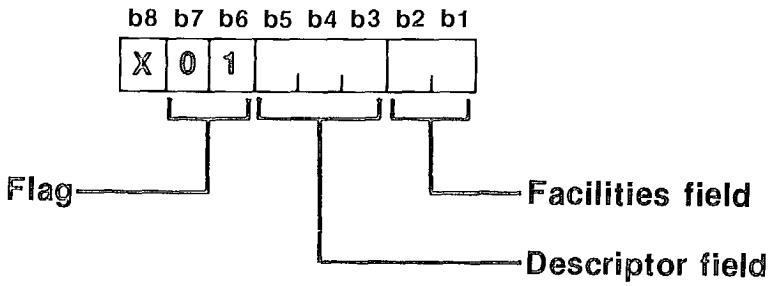


Figure A.2 8 bit Opcode Byte

A.2.3 Opcode definitions

A.2.3.1 POINT sets the drawing point to any position in the display space and optionally draws a dot.

A.2.3.2 LINE draws a line based on the two given end points.

A.2.3.3 ARC draws a circular arc based on three points, which are the start point, a point on the arc and the end point of the arc. A circle results when the start and end points are coincidental and the point on the arc defines the opposite end of the diameter. If only 2 points are transmitted then a circle is drawn in which the end point is assumed to be identical to the first point. If additional numeric data is transmitted beyond the end point of the arc, then a curvilinear spline is drawn, with each operand treated as a point on the spline and given as a relative displacement from the last point. The arc (spline) may be either in outline, or the area enclosed by the arc (spline) and the chord may be filled.

A.2.3.4 RECTANGLE draws a rectangle based on a specified width and height. The rectangle may be in outline or may be a filled-in area.

A.2.3.5 POLYGON draws a closed polygon of arbitrary shape specified by the vertices. The polygon may be in outline or may be a filled-in area. The maximum number of vertices is limited to 256.

A.2.3.6 INCREMENTAL draws an image consisting of either sequences of points, sequences of lines or filled polygons described in a compact incremental manner.

A.2.3.7 CONTROL provides control over the modes or attributes of the drawing commands.

A.2.4 Opcode facilities

Each geometric promitive opcode has four variants; these are defined by the facility bits (b2 and b1) as shown in Figure A.3. Facility field interpretations are as below.

OPCODE	b8	b7 b6 FLAG	b5 b4 b3 DESCRIPTOR FIELD	b2 = 0	b2 = 1	b1 = 0 (FACILITY FIELD)	b1 = 1 (FACILITY FIELD)
CONTROL	X	01	000				
POINT	X	01	001	INVIS	VIS	ABS	REL
LINE	X	01	010	JOIN	SET	ABS	REL
ARC	X	01	011	JOIN	SET	OUTLINE	FILL
RECTANGLE	X	01	100	JOIN	SET	OUTLINE	FILL
POLYGON	X	01	101	JOIN	SET	OUTLINE	FILL
INCREMENTAL	X	01	110				
CONTROL	X	01	111				

INVIS = Invisible
VIS = visible

ABS = Absolute
REL = Relative

Figure A.3 Opcode facilities

If b2 is binary 1

a) POINT — a visible point is drawn on the display screen.

b) LINE, ARC, RECTANGLE, POLYGON — the initial drawing position is specified within the data bytes as absolute (X,Y) coordinates, i.e, the initial point is Set.

If b2 is binary 0

a) POINT — an invisible point is located on the display screen.

b) LINE, ARC, RECTANGLE, POLYGON — the initial drawing position is the same point as the final drawing position of the previous opcode, i.e., the current drawing is joined to the previous drawing.

If b1 is binary 1

a)	POINT	the (dx, dy) coordinates are relative displacements to the preceding coordinate specifications.
b)	LINE	the (dx, dy) coordinates for the final drawing position of a line segment are relative displacements from the initial drawing position of that line segment.
c)	ARC, RECTANGLE, POLYGON	the interior areas established are filled.

If b1 is binary 0

a)	POINT	the (X,Y) coordinates of the point are absolute values.
b)	LINE	the (X,Y) coordinates of the final drawing position of the line segment are absolute values.
c)	ARC, RECTANGLE, POLYGON	the drawings are outlined.

A.3 Opcode numeric Data

The numeric data bytes associated with an opcode immediately follow the opcode byte and are recognized when the flag bit (b7) is binary 1. Any number of blocks of data bytes defining pairs of coordinates or drawing displacements may follow the drawing opcode. Any presentation level code other than from the numeric data portion of the code table terminates the sequence of data blocks. Transmission level codes have no effect at the presentation level as they should have been removed by lower layer processes.

The default number of data bytes that forms a block that defines a pair of x,y coordinates is three. The structure of the data block is shown in Figure A.4.

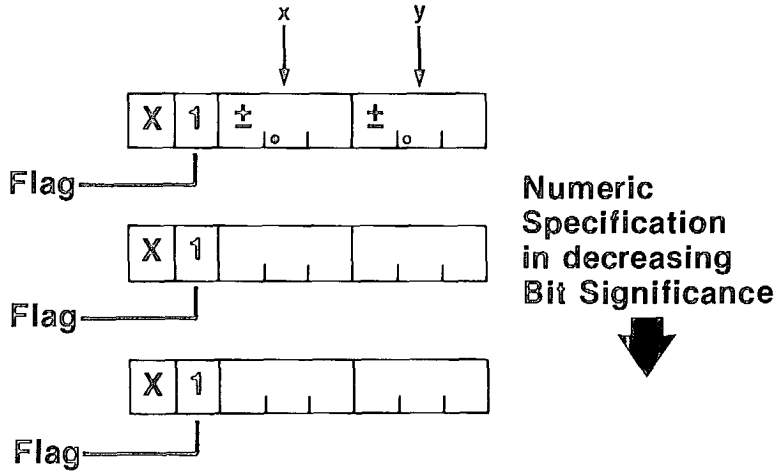


Figure A.4 Structure of a Block of 3 data bytes

A.4 Repeated opcode operation

For each of the POINT, LINE and RECTANGLE opcodes, repeated drawing operations will automatically be effected if the numerical data field following the opcode byte contains more than one complete set of coordinate specifications. The repeated drawing feature allows concatenated drawings to be effected without having to repeat the opcode itself.

A.5 INCREMENTAL Opcodes

The INCREMENTAL opcodes provide the capability of compactly describing images consisting of sequences of points, lines, or filled polygons. The four INCREMENTAL opcodes are shown in figure A.5

OPCODE	b8 -	b7 - b6 - FLAG	b5 - b4 - b3 - DESCRIPTOR FIELD	b2 - b1 - FACILITY FIELD
FIELD	X	01	110	00
INCREMENTAL POINT	X	01	110	01
INCREMENTAL LINE	X	01	110	10
INCREMENTAL POLYGON	X	01	110	11

Figure A.5 INCREMENTAL Opcodes

A.5.1 FIELD

The FIELD PDI establishes a rectangular active drawing area (field) on the screen which is used by commands such as the INCREMENTAL POINT PDI. The first block of data following the opcode gives the coordinates of the origin of the rectangular active drawing area. The next (and last) block of data gives the dimensions of the field as dx and dy. After a FIELD PDI is executed, the current drawing point is set to the origin point of the area.

A.5.2 INCREMENTAL POINT

INCREMENTAL POINT PDI allows the definition of a photographic type image by describing the colours of successive logical pels within the active drawing area field. The order of the logical pels is (starting from the current drawing point) left to right (or right to left if dx is negative), and up (or down if dy is negative) as shown in figure A.6. The n + 1 pixel causes a scroll of the field to occur in the -dy direction.

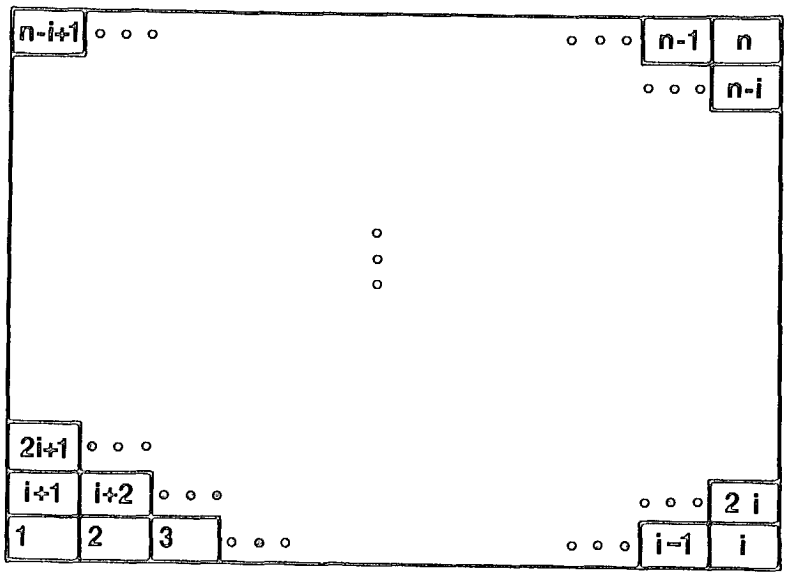


Figure A.6 Ordering of n logical pels for INCREMENTAL POINT (dx positive, dy positive case)

The first data byte following the INCREMENTAL POINT opcode is the packing counter, an unsigned integer giving the number of consecutive bits to be used to make up a single colour specification (1 to 64) where 0 implies 64. The remaining bytes give the colour specifications, high order bit (b6) to low order bit (b1) within the numeric data fields as colours or as colour map addresses, depending on the colour mode.

Bits are concatenated without regard to byte boundaries except when the drawing point meets or exceeds the boundary of the active drawing area. In this case, any remaining bits in the byte currently being interpreted are discarded and interpretation is resumed at the next complete byte.

A.5.3 INCREMENTAL LINE

This command allows a series of short line segments of width dx and/or height dy to be drawn. The first block of data associated with the command gives the step size parameters as dx and dy. The remaining bytes are interpreted as

three two bit nibbles per byte in the numeric data field (b6 through b1) which are interpreted left to right. These two bit nibbles cause the following actions to take place:

b2	b1	ACTION
0	0	Interpret the following nibble as a modify parameter instruction
0	1	move the drawing point a step in X of dx and optionally draw a line between the two points
1	0	move the drawing point a step in Y of dy and optionally draw a line between the two points
1	1	move the drawing point a step in X of dx and a step in Y of dy and optionally draw a line between the two points.

When a nibble value of (0,0) is encountered, the next nibble causes parameter changes as follows:

b2	b1	MODIFY PARAMETER INSTRUCTION
0	0	toggle the state of the draw flag (on or off) (default ON)
0	1	negate dx
1	0	negate dy
1	1	negate dx and dy

A.5.4 INCREMENTAL POLYGON

The interpretation of the operand data following the INCREMENTAL POLYGON opocde is exactly as for the INCREMENTAL LINE opcode except that the resulting figure is filled in the current colour and the final drawing point is taken as the initial drawing point. The draw flag is not used and the perimeter may not fold over on itself.

A.6 Geometric control opcodes

A.6.1 General

The CONTROL opcodes establish the display attributes and drawing states of the terminal for subsequent pictorial drawing, text or other presentation level commands. The eight CONTROL opcodes are given in Figure A.7.

OPCODE	b8 -	b7 - b6 - FLAG	b5 - b4 - b3 - DESCRIPTOR FIELD	b2 - b1 - FACILITY FIELD
RESET	X	01	000	00
DOMAIN	X	01	000	01
TEXT CONTROL	X	01	000	10
TEXTURE	X	01	000	11
SET COLOUR	X	01	111	00
WAIT	X	01	111	01
SELECT COLOUR	X	01	111	10
BLINK	X	01	111	11

Figure A.7 Control Opcodes

RESET is used to selectively reinitialize the drawing state and attribute parameters and to perform the function of clearing the display screen and other defined tables.

DOMAIN is used to establish operand parameter length and the logical pel size.

TEXT is used to control parameters related to the attributes of TEXT characters.

TEXTURE provides control of texture attributes that determine the method of filling areas for subsequent drawing commands and that determine the texture for lines and outlines.

SET COLOUR specifies actual colour values for use in drawing commands or for insertion into the colour map.

WAIT causes a delay of a specific time in processing data.

SELECT COLOUR is used to select the in-use drawing colour and the current colour mode.

BLINK allows control over a blink process to manipulate the colours displayed. A colour map is used to assign colour numbers to colour values.

A.6.2 ATTRIBUTES

A number of drawing attributes may be applied to the drawing commands and where appropriate to the other text and graphic commands. Attributes are defined by appropriately coded sequences as described below. Once an attribute is defined, it remains valid until the attribute is redefined or cleared to its default state. In the implementation of attributes, the level of sophistication and complexity are left to the discretion of the implementer.

A.6.3 DOMAIN

The integer value of bits b1 and b2 of the first byte of the operand (plus one) gives the length of subsequent single-value operands (such colour map table addresses) in bytes. The integer value of bits b3, b4, b5, of the first byte of the operand (plus one) gives the length of subsequent multi-value operands (such as coordinates specifications) in bytes.

Bit 6 of the first byte of the operand gives the dimensionality of the coordinate specification. Zero indicates two dimensional (2D) mode, while one indicates three dimentional (3D) mode. The definition of 3D mode is reserved for further study.

The subsequent bytes of the operand define the logical pel size as dx and dy. If an operand is shorter than the specified operand length then the operand is padded with zeros. If the operand is longer than the specified operand length then the command is repeated with the subsequent numeric data taken as the new operand.

A.6.4 TEXT

Bits 1 and 2 of the first byte of the operand determine the rotation of text characters as shown in the table below and in figure A.8.

b2	b1	Rotation
0	0	0° (default)
0	1	90°
1	0	180°
1	1	270°

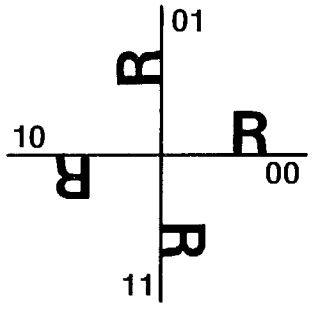


Figure A.8 Character Rotation

Bits 3 and 4 of byte 1 determine the direction of the text cursor path as follows:

b4	b3	cursor movement
0	0	right (default)
0	1	left
1	0	up
1	1	down

Bits 5 and 6 of byte 1 determine the inter-character spacing in units of the character field dimension lying parallel to the character path as follows:

b6	b5	inter-character spacing
0	0	1 (default)
0	1	1.25
1	0	1.5
1	1	proportional spacing

Bits 1 and 2 of the second byte determine the inter-row spacing in units of the character field dimension lying perpendicular to the character path as follows:

b2	b1	inter-row spacing
0	0	1 (default)
0	1	1.25
1	0	1.5
1	1	2

Bits 3 and 4 of the second byte determine the relationship between movement of the text cursor and movement of the graphics drawing point as follows:

b4	b3	move parameters
0	0	move together (default)
0	1	cursor leads
1	0	drawing point leads
1	1	move independently

Bits 5 and 6 of byte 2 determine the cursor display style as follows:

b6	b5	cursor style
0	0	underscore (default)
0	1	block
1	0	cross hair
1	1	custom (manufacturer defined)

The remaining bytes specify the dimensions of the character field dx (width) and dy (height).

A.6.5 TEXTURE

Bits 1 and 2 of the first byte of the operand determine the line texture attributes as follows:

b2	b1	line texture
0	0	solid (default)
0	1	dotted
1	0	dashed
1	1	dot dashed

The line texture pattern is referenced to the absolute coordinate grid of the display screen so that the texture pattern aligns between drawing commands.

If bit 3 of the first byte is 1, then filled rectangles, arcs and polygons are highlighted by explicitly drawing the perimeter. If bit 3 of the first byte is 0, then there is no highlight.

Bits 4, 5, and 6 determine the texture pattern for filled rectangles, arcs, and polygons as follows:

b6	b5	b4	texture pattern
0	0	0	solid (default)
0	0	1	vertical hatching
0	1	0	horizontal hatching
0	1	1	cross hatching
1	0	0	mask A
1	0	1	mask B
1	1	0	mask C
1	1	1	mask D

The four texture masks, A, B, C and D are dynamically redefinable.

The remaining bytes give the width (dx) and height (dy) of the mask size to be used in the step and repeat process which copies the mask pattern into the entire fill area.

A.6.6 SET COLOUR

In colour mode 0, this opcode specifies the colour attribute of the drawings (or text) that follows, implicity defining a colour map.

In colour modes 1 and 2, this command is used to explicitly load colour values into the colour map at the address indicated by the in-use drawing colour (previously selected with the SELECT COLOUR command).

The number of data bytes is variable and the sequence is terminated on the appearance of another opcode. Less significant bits for colour information are truncated where they are not used. The bit assignments of the data bytes are shown in Figure A.9.

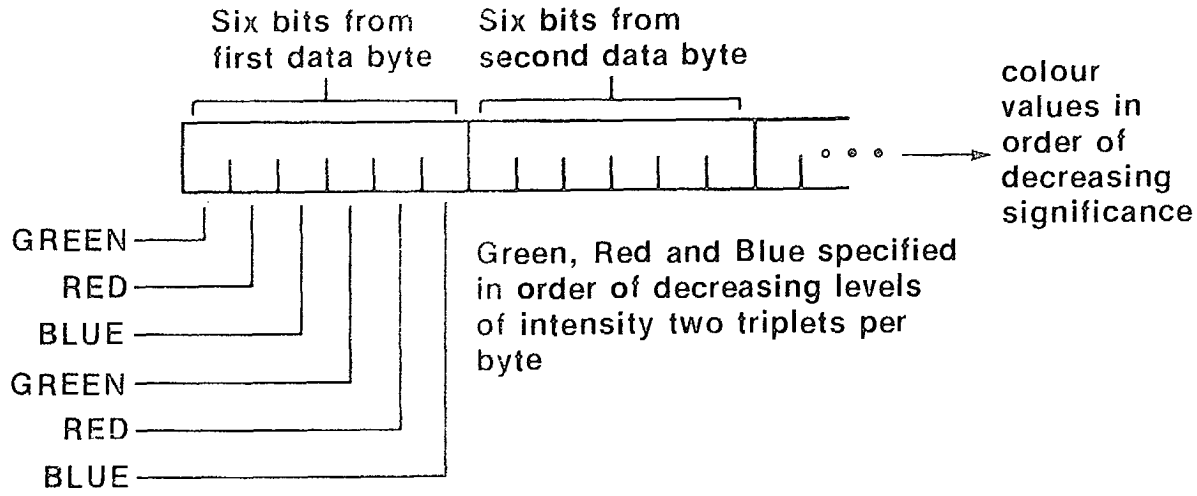


Figure A.9 SET COLOUR Operand

A.6.7 WAIT

This command causes a delay of a specific time in processing and display.

The length of wait is specified in tenths of a second by the first associated parameter byte (6 bits for up to 6.3 seconds). Each additional parameter byte causes an additional delay.

A.6.8 SELECT COLOUR

This opcode permits the in-use colour to be explicitly selected from the colour map and permits the colour mode to be set.

If no operands follow the SELECT COLOUR opcode, then the default colour mode 0 the default mode is selected.

If one single-value operand follows, then colour mode 1 is indicated, and the in-use drawing colour is set to the colour map address indicated by the single value operand.

If two single-value operands follow, then colour mode 2 is indicated. The in-use drawing colour is determined by the first operand and the in-use background colour for text is determined by the second operand (which points also to a colour map address).

A.6.9 BLINK

This opcode permits the colour map to be manipulated with respect to timed events.

The first byte is a single-value operand which points to the colour map entry of the "blink to" colour (the "blink from" colour is the current in-use drawing colour).

The next byte gives the blink "on" interval in tenths of a second.

The following byte gives the blink "off" interval in tenths of a second.

The last byte gives the phase delay in tenths of a second measured from the beginning of the next "on" interval of the most recently received active blink process.

A.6.10 RESET

If b1 of the first byte is 1, then the domain parameters are reset to their default values.

Bits 2 and 3 of byte 1 determine the colour mode and/or the in-use colour as follows:

b3	b2	
0	0	no action
0	1	select colour mode 0 and initialize implicit colour map
1	0	select colour mode 1 and set colour map to default colours
1	1	select colour mode 1, initialize colour map and set in-use drawing colour to white

Bits 4,5 and 6 of byte 1 clear the screen and/or border areas as follows:

b6	b5	b4	
0	0	0	no action
0	0	1	clear screen to black
0	1	0	clear screen to in-use drawing colour
0	1	1	clear border to black
1	0	0	clear border to in-use drawing colour
1	0	1	clear screen and border to in-use drawing colour
1	1	0	clear screen to in-use drawing colour-border to black
1	1	1	clear screen and border to black

If b1 of the second byte is 1, the text parameters are set to their default values and the cursor is sent to the home position (top left character position on the screen).

If b2 of the second byte is 1, all currently active blink processes are terminated.

If b3 of the second byte is 1, all unprotected fields are made protected.

If b4 of the second byte is 1, the texture attributes are set to their default values.

If b5 of the second byte is 1, all currently defined MACRO-PDIs are cleared.

If b6 of the second byte is 1, the DRCS is cleared (all character positions are set to the space character).

If no operand follows the RESET command, a complete reset is indicated (i.e., as if both operand bytes were set to all ones).

If one operand follows the RESET command, the second byte is padded with zeros.

A.6.11 CONTROL (STATUS) sub-commands

In order to harmonize with the existing CCITT Recommendation S.100, an alternate method of defining several of the control commands may be implemented on some terminals. These commands produce the same effects or combination of effects as the control opcodes. They are merely alternate ways of accessing the same functions, which have been retained for compatibility.

The WAIT control opcode may be interpreted as a control status code which introduces a number of sub-commands. The numerical data byte following the opcode indicates the sub-command. The WAIT function itself operates as one of the sub-commands.

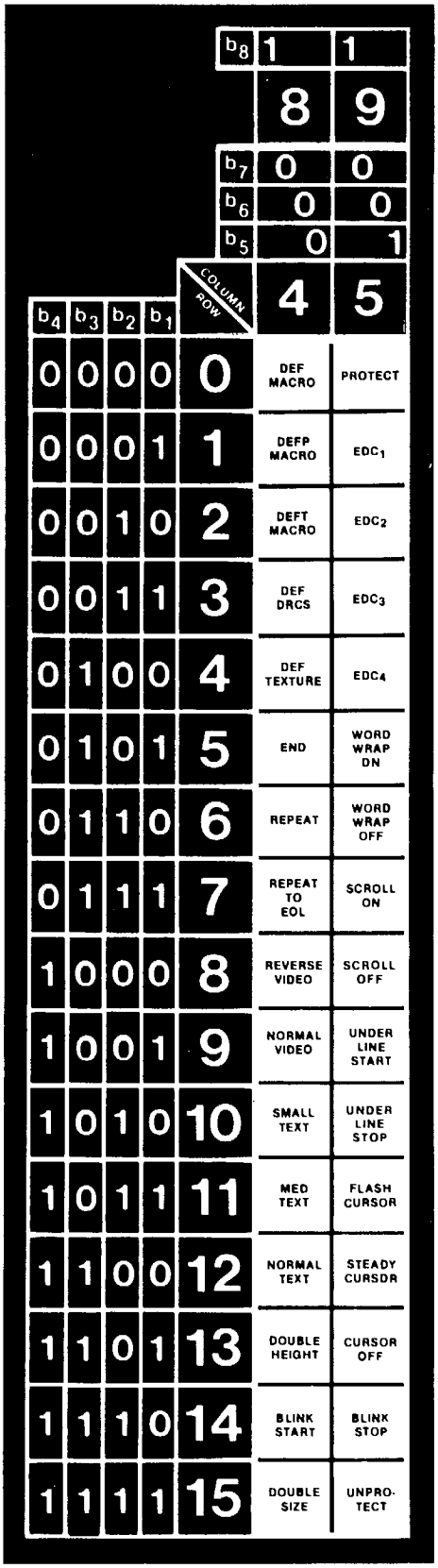


Figure A.10 C1 Control Set

A.7 C1 Control Set

The C1 control set provides additional control facilities, such as text control and MACRO PDI, DRCS and texture mask down-loading capabilities. The code table is illustrated in Figure A.10, and each of the control commands are described below.

A.7.1 DEF MACRO

The MACRO capability provides the facility of storing arbitrary strings of presentation level coded characters in such a way so that the string can be executed by the invocation of a single character from the 96 character MACRO table.

a) Definition of a MACRO function begins with the invocation of the C1 control function DEF MACRO followed by a single character (2/0, 2/1, ..., 7/15) of the MACRO name (M0 through M95).

b) A sequence of presentation layer codes follows in which a MACRO may not recursively call itself. A null sequence implies deletion of the MACRO.

c) Definition of the MACRO ends upon receipt of one of the following: C1 control character sequences, DEF MACRO, DEFP MACRO, DEFT MACRO, DEF DRCS, END.

A.7.2 DEFP MACRO

This C1 control is the same as the DEF MACRO control except that it simultaneously stores and executes the incoming characters that make up the MACRO. The DEFP MACRO cannot be used to define a MACRO that calls itself.

A.7.3 DEFT MACRO

This C1 control is the same as the DEF MACRO control, except that when called, it is not executed but is transmitted back to the host computer, thus providing a programmable function key capability.

A.7.4 DEF DRCS

a) Down-loading of one of 96 DRCS characters begins with the invocation of this C1 control to be followed by the bit combination of the location of the DRCS character (2/0, 2/1, ... 7/15). Existing DRCS shapes associated with the identified DRCS character will be deleted.

b) Next comes any legal sequence of presentation layer codes which define the DRCS shape being down-loaded. The codes are executed when received and the resultant DRCS shape is mapped into a 1-bit deep DRCS character buffer.

c) The down-loading is terminated when any one of the following commands is received: END, DEF MACRO, DEFP MACRO, DEFT MACRO, DEF TEXTURE, DEF DRCS. In the special case of terminating by a DEF DRCS sequence,

the next DRCS character location becomes implicit and need not be transmitted to the terminal (i.e. if down-loading for current position 2/5 is terminated by DEF DRCS, then the next down-loading sequence will automatically define the DRCS shapes for position 2/6).

d) A DRCS character can be deleted by sending the END C1 control immediately after step (a) above. All DRCS characters can be simultaneously deleted with the RESET command.

e) When the down loading process is completed, the drawing point is set to (0,0).

A.7.5 DEF TEXTURE

Begin definition of a programmable texture mask. The mask code (A (4/1), B (4/2), C (4/3), D (4/4)) is next, followed by presentation level code. The definition is terminated with an END control code or another DEF control code.

A.7.6 END

END terminates the current DEF MACRO, DEFP MACRO, DEFT MACRO, DEF DRCS or DEF TEXTURE operation.

A.7.7 REPEAT

The last text character received is repeated a specified number of times. Bits b6 through b1 of the next character received specify the repeat count.

A.7.8 REPEAT to EOL

The last TEXT character received is repeated to the last character position along the current character path, within the screen or active drawing area.

A.7.9 REVERSE VIDEO

Enter reverse video mode. Any TEXT character received subsequent to this control code is complemented within the current character field prior to its display.

A.7.10 NORMAL VIDEO

Enter normal video mode. The action of the REVERSE VIDEO control code is terminated.

A.7.11 SMALL TEXT

The horizontal dimension of the character field (dx) is set to 0.0125 to allow an 80 character per row display. The vertical dimension (dy) is set approximately to 0.04.

A.7.12 MED TEXT

The character field dimensions are set to dx = 0.03125, dy = 0.047 (approx.) to provide a 16 row by 32 character display.

A.7.13 NORMAL TEXT

The character field dimensions are set to their default values of dx = 0.025, dy = 0.0375 (approx.) to provide a 20 row by 40 character display.

A.7.14 DOUBLE HEIGHT

The vertical dimension of the character field is set to twice its default size. The horizontal dimension of the character field is set to its default size.

A.7.15 DOUBLE SIZE

The horizontal and vertical dimensions of the character field are set to twice their respective default sizes.

A.7.16 WORD WRAP ON

Enter word wrap mode. All text is broken on word boundaries at end of line conditions. A word boundary is delineated by a space character, a format effector control character, a drawing command from the geometric or mosaic sets, a maximum line length or optionally a punctuation character.

A.7.17 WORD WRAP OFF

Enter word wrap mode. All text is broken at character boundaries at end of line conditions.

A.7.18 SCROLL ON

Enter scroll mode. If an APD or an APU would cause the text cursor to be advanced past the edge of the active drawing area, the entire contents of the display are scrolled to bring the cursor back within the active drawing area.

A.7.19 SCROLL OFF

Exit scroll mode. If an APD or an APU would cause the text cursor to be advanced past the edge of the active drawing area it is instead moved to the opposite edge of the active drawing area.

A.7.20 UNDERLINE START

Enter underline mode. All characters received from the alpha-numeric, supplementary graphics or DRCS set will be displayed underlined. The mosaic set will be displayed in separated graphics mode.

A.7.21 UNDERLINE stop

Exit underline mode.

A.7.22 FLASH CURSOR

The cursor display is turned on and caused to flash intermittently.

A.7.23 STEADY CURSOR

The cursor display is turned on and is made steadily visible.

A.7.24 CURSOR OFF

The cursor display is made invisible.

A.7.25 UNPROTECT

Cause the active display area (as defined by the FIELD PDI) to be defined as an unprotected field.

A.7.26 PROTECT

Causes all unprotected fields of which any portion lies within the active drawing area to be made protected.

A.7.27 BLINK START

Establish an automatic and implicit Blink process. Any text or graphics codes received while in this mode will cause the resulting text or graphic to flash intermittently.

A.7.28 BLINK STOP

Disable all blink processes both implicit and explicit.

A.7.29 EDC1, EDC2, EDC3 EDC4

The meanings of the extended device controls are reserved for future standardization.

A.8 Default conditions

The default conditions of the attributes for the alpha-geometric coding scheme are summarized below:

colour mode:	0
In-use colour:	white
Single-value length (colour map address):	1 byte
Multi-value length (coordinate data):	3 bytes
Dimensionality:	2 D
Logical pel size:	dx = +0, dy = +0
Text rotation:	0° (horizontal)
Character path:	to the right
Inter-Character spacing:	1 (width of current character field)
Inter-row spacing	1 (height of current character field)
Move parameters:	Text cursor and graphics drawing point move together
Cursor Style:	Underscore
Cursor display:	off
Character field dimensions:	dx = 0.025, dy = 0.04 (approx.)
Line texture:	solid
Texture pattern:	solid
Highlight:	no highlight
Texture mask size:	dx = 0.025, dy = 0.04 (approx.)
Underline mode:	off
Word wrap mode:	off

A.9 MOSAICS

A basic Mosaic shape table is included in the presentation level protocol. This table is illustrated in Figure A.12 and is a union of the two Mosaic shape tables standardized in CCITT Recommendation S.100. This provides the capability of using Mosaic drawing techniques for those situations where they may be appropriate, as well as providing the basic shapes necessary for transcoding data from other Mosaic systems into the Presentation Level Protocol. By use of the variable character density feature of the PLP it is possible to handle mosaic data at 40 x 20, 40 x 24, 32 x 16 or any other density.

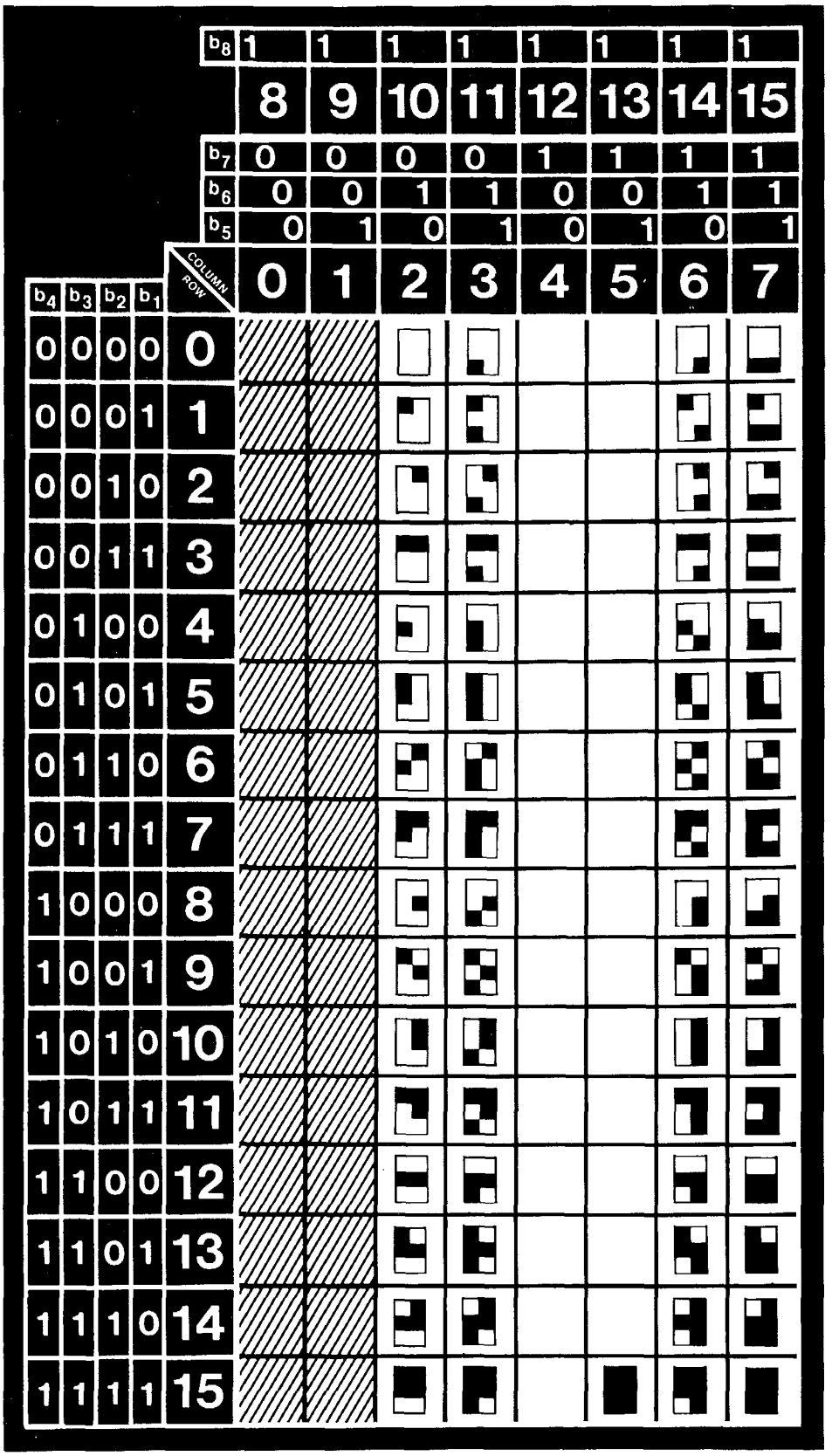


Figure A.11 Mosaic Shape Table

A.10 C0 Text Cursor Control Codes.

C0 Character Name	Code Table Position	Effect on Text Cursor Location	Action on Text Cursor Crossing Unit Screen or Active Drawing Area	Other Action
APB(BS)	(0/8)	More in the opposite direction to the character path a distance equal to the dimension of the character field lying parallel to the character path	Position the cursor to opposite edge of screen or drawing area (along the character path) and execute an automatic VT	n/a
APF(HT)	(0/9)	Move in the direction of the character path a distance equal to the dimension of the character field lying parallel to the character path	Position the cursor to opposite edge of screen or drawing area (along the character path) and execute an automatic LF	n/a
APD(LF)	(0/10)	Move in the direction perpendicular to the character path (-90°) a distance equal to the dimension of the character field lying perpendicular to the character path	Special action taken dependent on whether scroll mode is in effect or not	n/a
APU (VT)	(0/11)	Move in the direction perpendicular to the character path (90°) a distance equal to the dimension of the character field lying perpendicular to the character path	Special action taken dependent on whether scroll mode is in effect or not	n/a
CS(FF)	(0/12)	Position cursor to upper left character position on screen	n/a	clear screen (colour modes 0 and 1) or clear screen to background colour (colour mode 2)
APR(CR)	(0/13)	Position cursor to first character position within the unit screen (or drawing area) along the character path	n/a	n/a
APH(RS)	(1/14)	Position cursor to upper left character position on screen	n/a	n/a

APPENDIX B - DESCRIPTIONS OF CHARACTERS

B.1 Coded Representations and Descriptions

In tables B.1 to B.10, the column labelled "Coded Representation" specifies the codes representation of the graphic characters and symbols for the case of a 7-bit code. It is assumed for this description that, in the eight bit case, the elements of the primary set are represented by bit combinations in the range 2/1 to 7/14.

The symbol "S" is used in to identify a bit combination representing an element of the supplementary set.

Depending on the code extension techniques used, a bit combination representating an element of the primary or the supplementary set may have to be preceded by a code extension function invoking the character set concerned.

All characters in column 4 (or column 12) in the supplementary set are taken as non-spacing.

Table B.1 - Latin Alphabetic Characters

GRAPHIC	NAME or DESCRIPTION	CODED REPRESENTATION
a	lower case a	6/1
A	upper case A	4/1
á	lower case a with acute accent	S 4/2 6/1
Á	upper case A with acute accent	S 4/2 4/1
à	lower case a with grave accent	S 4/1 6/1
À	upper case A with grave accent	S 4/1 4/1
â	lower case a with circumflex accent	S 4/3 6/1
Â	upper case A with circumflex accent	S 4/3 4/1
ä	lower case a with diaeresis or umlaut mark	S 4/8 6/1
Ä	upper case A with diaeresis or umlaut mark	S 4/8 4/1
ã	lower case a with tilde	S 4/4 6/1
Ã	upper case A with tilde	S 4/4 4/1
ă	lower case a with breve	S 4/6 6/1
Ă	upper case A with breve	S 4/6 4/1
å	lower case a with ring	S 4/10 6/1
Å	upper case A with ring	S 4/10 4/1
ā	lower case a with macron	S 4/5 6/1
Ā	upper case A with macron	S 4/5 4/1
ą	lower case a with ogonek	S 4/14 6/1
Ą	upper case A with ogonek	S 4/14 4/1
æ	lower case æ dipthong	S 7/1
Æ	upper case Æ dipthong	S 6/1
b	lower case b	6/2
B	upper case B	4/2

Table B.1 (continued) - Latin Alphabetic Characters

GRAPHIC	NAME or DESCRIPTION	CODED REPRESENTATION
c	lower case c	6/3
C	upper case C	4/3
ć	lower case c with acute accent	S 4/2 6/3
Ć	upper case C with acute accent	S 4/2 4/3
ĉ	lower case c with circumflex accent	S 4/3 6/3
Ĉ	upper case C with circumflex accent	S 4/3 4/3
č	lower case c with caron	S 4/15 6/3
Č	upper case C with caron	S 4/15 4/3
ċ	lower case c with dot	S 4/7 6/3
Ċ	upper case C with dot	S 4/7 4/3
ç	lower case c with cedilla	S 4/11 6/3
Ç	upper case C with cedilla	S 4/11 4/3
d	lower case d	6/4
D	upper case D	4/4
ď or d'	lower case d with caron	S 4/15 6/4
D	upper case D with caron	S 4/15 4/4
đ	lower case d with stroke	S 7/2
Đ	upper case D with stroke, Icelandic eth	S 6/2
ð	lower case eth, Icelandic	S 7/3
e	lower case e	6/5
E	upper case E	4/5
é	lower case e with acute accent	S 4/2 6/5
É	upper case E with acute accent	S 4/2 4/5
è	lower case e with grave accent	S 4/1 6/5

Table B.1 (continued) - Latin Alphabetic Characters

GRAPHIC	NAME or DESCRIPTION	CODED REPRESENTATION
È	upper case E with grave accent	S 4/1 4/5
ê	lower case e with circumflex accent	S 4/3 6/5
Ê	upper case E with circumflex accent	S 4/3 4/5
ë	lower case e with diaeresis or umlaut mark	S 4/8 6/5
Ë	upper case E with diaeresis or umlaut mark	S 4/8 4/5
ě	lower case e with caron	S 4/15 6/5
Ě	upper case E with caron	S 4/15 4/5
ė	lower case e with dot	S 4/7 6/5
Ė	upper case E with dot	S 4/7 4/5
ē	lower case e with macron	S 4/5 6/5
Ē	upper case E with macron	S 4/5 5/5
ę	lower case e with ogonek	S 4/14 6/5
Ę	upper case E with ogonek	S 4/14 4/5
f	lower case f	6/6
F	upper case F	4/6
g	lower case g	6/7
G	upper case G	4/7
ǵ	lower case g with acute accent	S 4/2 6/7
ĝ	lower case g with circumflex accent	S 4/3 4/7
Ĝ	upper case G with circumflex accent	S 4/3 6/7
ğ	lower case g with breve	S 4/6 6/7
Ğ	upper case G with breve	S 4/6 4/7
ġ	lower case g with dot	S 4/7 6/7
Ġ	upper case G with dot	S 4/7 4/7

Table B.1 (continued) - Latin Alphabetic Characters

GRAPHIC	NAME or DESCRIPTION	CODED REPRESENTATION
Ģ	upper case G with cedilla	S 4/11 4/7
h	lower case h	6/8
H	upper case H	4/8
ĥ	lower case h with circumflex accent	S 4/3 6/8
Ĥ	upper case H with circumflex accent	S 4/3 4/8
ħ	lower case h with stroke	S 7/4
Ħ	upper case H with stroke	S 6/4
i	lower case i	6/9
I	upper case I	4/9
í	lower case i with acute accent	S 4/2 6/9
Í	upper case I with acute accent	S 4/2 4/9
ì	lower case i with grave accent	S 4/1 6/9
Ì	upper case I with grave accent	S 4/1 4/9
î	lower case i with circumflex accent	S 4/3 6/9
Î	upper case I with circumflex accent	S 4/3 4/9
ï	lower case i with diaeresis or umlaut mark	S 4/8 6/9
Ï	upper case I with diaeresis or umlaut mark	S 4/8 4/9
ĩ	lower case i with tilde	S 4/4 6/9
Ĩ	upper case I with tilde	S 4/4 4/9
İ	upper case I with dot	S 4/7 4/9
ī	lower case i with macron	S 4/5 6/9
Ī	upper case I with macron	S 4/5 4/9
į	lower case i with ogonek	S 4/14 6/9
Į	upper case I with ogonek	S 4/14 4/9

Table B.1 (continued) - Latin Alphabetic Characters

GRAPHIC	NAME or DESCRIPTION	CODED REPRESENTATION
ij	lower case ij ligature	S 7/6
IJ	upper case IJ ligature	S 6/6
ı	lower case i without dot	S 7/5
j	lower case j	6/10
J	upper case J	4/10
ĵ	lower case j with circumflex accent	S 4/3 6/10
Ĵ	upper case J with circumflex accent	S 4/3 4/10
k	lower case k	6/11
K	upper case K	4/11
ķ	lower case k with cedilla	S 4/11 6/11
Ķ	upper case K with cedilla	S 4/11 4/11
ĸ	lower case k, Greenlandic	S 7/0
l	lower case l	6/12
L	upper case L	4/12
ĺ	lower case l with acute accent	S 4/2 6/12
Ĺ	upper case L with acute accent	S 4/2 4/12
ľ or l'	lower case l with caron	S 4/15 6/12
Ľ or L'	upper case L with caron	S 4/15 4/12
ļ	lower case l with cedilla	S 4/11 6/12
Ļ	upper case L with cedilla	S 4/11 4/12
ł	lower case l with stroke	S 7/8
Ł	upper case L with stroke	S 6/8
ŀ	lower case l with middle dot	S 7/7
Ŀ	upper case L with middle dot	S 6/7

Table B.1 (continued) - Latin Alphabetic Characters

GRAPHIC	NAME or DESCRIPTION	CODED REPRESENTATION
m	lower case m	6/13
M	upper case M	4/13
n	lower case n	6/14
N	upper case N	4/14
ń	lower case n with acute accent	S 4/2 6/14
Ń	upper case N with acute accent	S 4/2 4/14
ñ	lower case n with tilde	S 4/4 6/14
Ñ	upper case N with tilde	S 4/4 4/14
ň	lower case n with caron	S 4/15 6/14
Ň	upper case N with caron	S 4/15 4/14
ņ	lower case n with cedilla	S 4/11 6/14
Ņ	upper case N with cedilla	S 4/11 4/14
ŋ	lower case eng, Lapp	S 7/14
Ŋ	upper case eng, Lapp	S 6/14
n	lower case n with apostrophe	S 6/15
o	lower case o	6/15
O	upper case O	4/15
ó	lower case o with acute accent	S 4/2 6/15
Ó	upper case O with acute accent	S 4/2 4/15
ò	lower case o with grave accent	S 4/1 6/15
Ò	upper case O with grave accent	S 4/1 4/15
ô	lower case o with circumflex accent	S 4/3 6/15
Ô	upper case O with circumflex accent	S 4/3 4/15
ö	lower case o with dieresis or umlaut mark	S 4/8 6/15

Table B.1 (continued) - Latin Alphabetic Characters

GRAPHIC	NAME or DESCRIPTION	CODED REPRESENTATION
Ö	upper case O with dieresis or umlaut mark	S 4/8 4/15
õ	lower case o with tilde	S 4/4 6/15
Õ	upper case O with tilde	S 4/4 4/15
ő	lower case o with double acute accent	S 4/13 6/15
Ő	upper case O with double acute accent	S 4/13 4/15
ō	lower case o with macron	S 4/5 6/15
Ō	upper case O with macron	S 4/5 4/15
œ	lower case œ ligature	S 7/10
Œ	upper case Œ ligature	S 6/10
ø	lower case o with slash	S 7/9
Ø	upper case O with slash	S 6/9
p	lower case p	7/0
P	upper case P	5/0
q	lower case q	7/1
Q	upper case Q	5/1
r	lower case r	7/2
R	upper case R	5/2
ŕ	lower case r with acute accent	S 4/2 7/2
Ŕ	upper case R with acute accent	S 4/2 5/2
ř	lower case r with caron	S 4/15 7/2
Ř	upper case R with caron	S 4/15 5/2
ŗ	lower case r with cedilla	S 4/11 7/2
Ŗ	upper case R with cedilla	S 4/11 5/2
s	lower case s	7/3

Table B.1 (continued) - Latin Alphabetic Characters

GRAPHIC	NAME or DESCRIPTION	CODED REPRESENTATION
S	upper case S	5/3
ś	lower case s with acute accent	S 4/2 7/3
Ś	upper case S with acute accent	S 4/2 5/3
ŝ	lower case s with circumflex accent	S 4/3 7/3
Ŝ	upper case S with circumflex accent	S 4/3 5/3
š	lower case s with caron	S 4/15 7/3
Š	upper case S with caron	S 4/15 5/3
ş	lower case s with cedilla	S 4/11 7/3
Ş	upper case S with cedilla	S 4/11 5/3
ß	lower case sharp s, German	S 7/11
t	lower case t	7/4
T	upper case T	5/4
ť or t'	lower case t with caron	S 4/15 7/4
Ť	upper case T with caron	S 4/15 5/4
ţ	lower case t with cedilla	S 4/11 7/4
Ţ	upper case T with cedilla	S 4/11 5/4
ŧ	lower case t with stroke	S 7/13
Ŧ	upper case T with stroke	S 6/13
þ	lower case thorn, Icelandic	S 7/12
Þ	upper case thorn, Icelandic	S 6/12
u	lower case u	7/5
U	upper case U	5/5
ú	lower case u with acute accent	S 4/2 7/5
Ú	upper case U with acute accent	S 4/2 5/5

Table B.1 (continued) - Latin Alphabetic Characters

GRAPHIC	NAME or DESCRIPTION	CODED REPRESENTATION
ù	lower case u with grave accent	S 4/1 7/5
Ù	upper case U with grave accent	S 4/1 5/5
û	lower case u with circumflex accent	S 4/3 7/5
Û	upper case U with circumflex accent	S 4/3 5/5
ü	lower case u with diaeresis or umlaut mark	S 4/8 7/5
Ü	upper case U with diaeresis or umlaut mark	S 4/8 5/5
ũ	lower case u with tilde	S 4/4 7/5
Ũ	upper case U with tilde	S 4/4 5/5
ŭ	lower case u with breve	S 4/6 7/5
Ŭ	upper case U with breve	S 4/6 5/5
ű	lower case u with double acute accent	S 4/13 7/5
Ű	upper case U with double acute accent	S 4/13 5/5
ů	lower case u with ring	S 4/10 7/5
Ů	upper case U with ring	S 4/10 5/5
ū	lower case u with macron	S 4/5 7/5
Ū	upper case U with macron	S 4/5 5/5
ų	lower case u with ogonek	S 4/14 7/5
Ų	upper case U with ogonek	S 4/14 5/5
v	lower case v	7/16
V	upper case V	5/16
w	lower case w	7/7
W	upper case W	5/7
ŵ	lower case w with circumflex accent	S 4/3 7/7
Ŵ	upper case W with circumflex accent	S 4/3 5/7

Table B.1 (continued) - Latin Alphabetic Characters

GRAPHIC	NAME or DESCRIPTION	CODED REPRESENTATION
x	lower case x	7/8
X	upper case X	5/8
y	lower case y	7/9
Y	upper case Y	5/9
ý	lower case y with acute accent	S 4/2 7/9
Ý	upper case Y with acute accent	S 4/2 5/9
ŷ	lower case y with circumflex accent	S 4/3 7/9
Ŷ	upper case Y with circumflex accent	S 4/3 5/9
ÿ	lower case y with diaeresis or umlaut mark	S 4/8 7/9
Ÿ	upper case Y with diaeresis or umlaut mark	S 4/8 5/9
z	lower case z	7/10
Z	upper case Z	5/10
ź	lower case z with acute accent	S 4/2 7/10
Ź	upper case Z with acute accent	S 4/2 5/10
ž	lower case z with caron	S 4/15 7/10
Ž	upper case Z with caron	S 4/15 5/10
ż	lower case z with dot	S 4/7 7/10
Ż	upper case Z with dot	S 4/7 5/10

Table B.2 Decimal digits

GRAPHIC	NAME or DESCRIPTION	CODED REPRESENTATION
1	digit 1	3/1
2	digit 2	3/2
3	digit 3	3/3
4	digit 4	3/4
5	digit 5	3/5
6	digit 6	3/6
7	digit 7	3/7
8	digit 8	3/8
9	digit 9	3/9
0	digit 0	3/0

Table B.3 Currency signs

GRAPHIC	NAME or DESCRIPTION	CODED REPRESENTATION
¤	general currency sign	S 2/8
£	pound sign	S 2/3
$	dollar sign	2/4; S 2/4
c	cent sign	S 2/2
¥	yen sign	S 2/5

Table B.4 Punctuation marks

GRAPHIC	NAME or DESCRIPTION	CODED REPRESENTATION
	space	2/0; S 2/0
!	exclamation point	2/1
¡	inverted exclamation point	S 2/1
"	quotation mark	2/2
'	apostrophe, acute accent	2/7
(	opening parenthesis (left)	2/8
)	closing parenthesis (right)	2/9
¸	comma (cedilla)	2/12
_	underline	5/15
-	hyphen, minus sign	2/13
.	period (decimal point)	2/14
/	slant (solidus)	2/15
:	colon	3/10
;	semicolon	3/11
?	question mark	3/15
¿	inverted question mark	S 3/15
«	angle quotation marks left	S 2/11
»	angle quotation marks right	S 3/11
‘	single quotation mark left	S 2/9
’	single quotation mark right	S 3/9
“	double quotation marks left	S 2/10
”	double quotation marks right	S 3/10

Table B.5 Arithmetic signs

GRAPHIC	NAME or DESCRIPTION	CODED REPRESENTATION
+	plus sign	2/11
±	plus/minus sign	S 3/1
<	less than sign	3/12
=	equals sign	3/13
>	greater than sign	3/14
÷	divide sign	S 3/8
x	multiply sign	S 3/4

Table B.6 Subscripts and Superscripts

GRAPHIC	NAME or DESCRIPTION	CODED REPRESENTATION
1	Superscript 1	S 5/1
2	Superscript 2	S 3/2
3	Superscript 3	S 3/3

Table B.7 Fractions

GRAPHIC	NAME or DESCRIPTION	CODED REPRESENTATION
1/2	fraction one half	S 3/13
1/4	fraction one quarter	S 3/12
3/4	fraction three quarters	S 3/14
1/8	fraction one eighth	S 5/12
3/8	fraction three eights	S 5/13
5/8	fraction five eights	S 5/14
7/8	fraction seven eights	S 5/15

Table B.8 Miscellaneous Symbols

GRAPHIC	NAME or DESCRIPTION	CODED REPRESENTATION
#	number sign	2/3; S 2/6
%	percent sign	2/5
&	ampersand	2/6
*	asterisk	2/10
@	commercial at	4/0
[	opening bracket (left)	5/11
\	reverse slant (solidus)	5/12
]	closing bracket (right)	5/13
{	opening brace	7/11
|	vertical line	7/12
}	closing brace	7/13
µ	micro sign	S 3/5
Ω	ohm sign	S 6/0
°	degree sign	S 3/0
º	ordinal indicator, masculine	S 6/11
ª	ordinal indicator, feminine	S 6/3
§	section sign	S 2/7
¶	paragraph sign, pilcrow	S 3/6
·	middle dot	S 3/7
←	leftward arrow	S 2/12
→	rightward arrow	S 2/14
↑	upward arrow	S 2/13
↓	downward arrow	S 2/15

Table B.8 (continued) Miscellaneous Symbols

GRAPHIC	NAME or DESCRIPTION	CODED REPRESENTATION
®	registered sign	S 5/2
©	Copyright sign	S 5/3
TM	Trade mark sign	S 5/4
♪	musical note	S 5/5
`	grave accent	6/0
^	circumflex	5/14
~	Tilde (high line)[1]	7/14
▨	diagonal[2]	S 5/8**
▨	reverse diagonal[3]	S 5/9**
◪	filled diagonal[4]	S 5/10**
◪	filled reverse diagonal[5]	S 5/11**
⊞	cross[6]	S 6/5**
⊟	full horizontal line[7]	S 5/6**
▯	full vertical line[8]	S 5/7**
—	horizontal bar[9]	S 5/0**

Notes:

1. "Tilde" may be represented by a "high line" on some implementations.

2. The "diagonal" extends from the lower left corner of the character field to the upper right corner of the character field.

3. The "reverse diagonal" extends from the lower right corner of the character field to the upper left corner of the character field.

** the coded representations of these symbols have not yet been officially adopted by a standard body.

Table B.8 (continued) Miscellaneous Symbols

Notes:

4. The "filled diagonal" is a filled triangle occupying the lower right half of the character field.

5. The "filled reverse diagonal" is a filled triangle occupying the lower left half of the character field.

6. The "cross" character is equivalent to overlaying the "full horizontal line" character with the "full vertical line" character.

7. The "full horizontal line" extends from the middle of the left edge of the character field to the middle of the right edge of the character field.

8. The "full vertical line" extends from the middle of the bottom edge of the character field to the middle of the top edge of the character field.

9. The typical graphic representation of "horizontal bar" is a horizontal line at about the middle of a capital letter, and that of "high line" is a horizontal line at about the top of a capital letter.

Table B.9 Diacritical Marks

GRAPHIC	NAME or DESCRIPTION	CODED REPRESENTATION
´	acute	S 4/2, 2/0 (2/7)
`	grave	S 4/1, 2/0 (6/0)
^	circumflex	S 4/3, 2/0 (5/14)
~	tilde	S 4/4, 2/2 (7/14)
¨	diaeresis or umlaut	S 4/8, 2/0
¸	cedilla	S 4/11, 2/0
˛	ogonek	S 4/14, 2/0
˘	breve	S 4/6, 2/0
ˇ	caron	S 4/15, 2/0
˝	double acute	S 4/13, 2/0
˙	dot	S 4/7, 2/0
¯	macron	S 4/5, 2/0
˚	ring	S 4/10, 2/0

Table B.10 Miscellaneous Non-Spacing Characters

GRAPHIC	NAME or DESCRIPTION	CODED REPRESENTATION
	non-spacing underline	S 4/12**
	vector overbar	S 4/0**
	non-spacing solidus	S 4/9**

Note:

The graphic representation of "vector overbar" is that of a vector arrow at just above the top of a capital letter.

B.2 Use of Diacritical Marks

This Figure B.1 shows the combinations of diacritical marks and basic letters used in Latin-based languages. See reference 10 for more information.

BASIC LETTER	ACUTE ACCENT	GRAVE ACCENT	CIRCUMFLEX ACCENT	DIAERESIS OR UMLAUT MARK	TILDE	CARON	BREVE	DOUBLE ACUTE ACCENT	RING	DOT	MACRON	UMLAUT MARK	CEDILLA	OGONEK
a A	á Á	à À	â Â	ä Ä	ã Ã		ă Ă		å Å		ā Ā	ä Ä		ą Ą
b B														
c C	ć Ć		ĉ Ĉ			č Č				ċ Ċ			ç Ç	
d D						ď Ď								
e E	é É	è È	ê Ê	ë Ë		ě Ě				ė Ė	ē Ē	ë Ë		ę Ę
f F														
g G	ǵ		ĝ Ĝ				ğ Ğ			ġ Ġ			ģ	
h H			ĥ Ĥ											
i I	í Í	ì Ì	î Î	ï Ï	ĩ Ĩ					İ	ī Ī			į Į
j J			ĵ Ĵ											
k K													ķ Ķ	
l L	ĺ Ĺ					ľ Ľ							ļ Ļ	
m M														
n N	ń Ń				ñ Ñ	ň Ň							ņ Ņ	
o O	ó Ó	ò Ò	ô Ô	ö Ö	õ Õ			ő Ő			ō Ō	ö Ö		
p P														
q Q														
r R	ŕ Ŕ					ř Ř							ŗ Ŗ	
s S	ś Ś		ŝ Ŝ			š Š							ş Ş	
t T						ť Ť							ţ Ţ	
u U	ú Ú	ù Ù	û Û	ü Ü	ũ Ũ		ŭ Ŭ	ű Ű	ů Ů		ū Ū	ü Ü		ų Ų
v V														
w W			ŵ Ŵ											
x X														
y Y	ý Ý		ŷ Ŷ	ÿ Ÿ								ÿ Ÿ		
z Z	ź Ź					ž Ž				ż Ż				

Figure B.1 Use of Diacritical Marks

APPENDIX C
CONSIDERATIONS FOR OTHER PROTOCOL LAYERS

Complete Codes - The entire coding scheme including all the default designations comprises a complete code. That is, the Presentation Layer Protocol is a consistent self contained code structure and not just a number of independently defined code tables. Other complete codes can be defined for other services which are beyond the range of applicability of the Presentation Layer Protocol and these complete codes may interwork with the Presentation Layer Protocol. For the purposes of interworking a possible escape sequence could introduce the PLP. This sequence would not normally transmitted but may be required for a gateway or operation in which interworking between protocol is required.

The code tables defined in the Presentation Level Protocol make use of all 96 code table locations. This differs slightly from the normal operation of "Text only" code tables where the code table position 2/0 is reserved for a SPACE function and position 7/15 is reserved for a DELETE function. Because of the nature of the Geometric, Mosaic and MACRO pictorial coding methods this convention acts as a coding restriction. This restriction is overcome within the rules of ISO 2022 by establishing within the complete code the entire conventional coding scheme with the code tables defined as 96 characters. For code tables registered with 94 characters, such as the ASCII text table or the Supplementary Text table, the characters SPACE and DELETE are added to the appropriate corners. A fully filled in character matrix is also defined in order to establish a display representation for the DELETE character.

Other Protocol Layers - The field trials of Videotex in Canada to date have experimented with various transmission means and interaction scenarios. The nature of the interactive dialogue varies with the type of transmission media, however, it is expected that a common protocol will develop permitting common terminals to communicate over the different communications means. Further study is required in this area as Videotex systems mature.

The interaction dialogue for videotex services should parallel as closely as possible that used in conventional text business terminals in order to interwork with the largest number of related applications. In fact the PLP can be considered as a superset of the conventional text oriented ASCII coding scheme. In the same way the interaction should incorporate conventional terminal dialogue techniques. Although the interactive protocol may vary from service to service or require more features due to the nature of different transmission media, there are several basic functions which would be common to all such protocols. Some of these functions are outlined in this section together with suggested coding for some functions. Other functions must be coded in a manner dependent on the nature of the service, such as being coded as bits in a packet oriented Broadcast Videotex (Teletext) header structure. The definition of coding for these functions will be left to the various service specifications.

The session level of the Open Systems Layered Architecture provides the means to establish a session connection and to support the orderly exchange of data and other control functions for a particular service. In order to do this it must establish and maintain the session dialogue as well as address the desired presentation entity. The session level dialogue requires that certain features be provided in an Interactive Videotex service. The session level dialogue and service functions for the Broadcast Videotex service are provided in Department of Communications Broadcast specification BS-14.

If the coding of dialogue functions are not established within the bit assignments of a particular transmission protocol, then presentation level codes must be assigned to accommodate these functions. These codes must be assigned in a non-interfering manner with the Presentation Layer Protocol, and therefore require the definition of a set of Escape sequences. The exact coding of these sequences is left to further study.

Symmetric Dialogue - The dialogue should be totally symmetric, that is the communications channel from the terminal to the host computer or other terminal should be independent of the channel to the terminal. In general anything typed on the keyboard, keypad or other device should be transmitted "up" the communications link and anything received should be interpreted and decoded by the terminal. The PLP applies independently in both directions. However, in practice the data which is transmitted usually comes only from the default ASCII text code table or the presentation portion of the C0 set. This is because in general only a keypad or keyboard device is used. A keypad is a direct subset of a keyboard. The only difference is that fewer keys are available. The activation character (see below) if record activation is used is the carriage return character FE5 (CR/APR) position 0/13 in the C0 control code table, just as it is for the keyboard, and it is always transmitted if it is typed. If data is automatically transmitted such as a result of a programmable function key, it too must be in the format of the PLP, and this transmission will not interrupt a sequence of data being transmitted nor will it be interrupted by any transmission; that is, the requested sequence will be transmitted as a single block of data. If other than full duplex transmission media are used, it is the responsibility of the lower layers of the transmission system to establish a half duplex protocol.

Data-forwarding - There are several ways in which the interaction dialogue may be manifested, just as there are several types of dialogue for conventional text oriented business terminals. Data-forwarding, that is transmission, may occur either on a single character or on a buffered basis. For single character data-forwarding each character is transmitted as it is typed. For record data-forwarding a buffer is accumulated. Upon typing the data-forwarding character the entire buffer together with the data-forwarding character is transmitted.

The characters which are typed may be either echoed for display on the screen by the remote computer or locally within the terminal in a similar manner to the loopback local echo mode used in many conventional terminals. Obviously local echo is intended for use with record data-forwarding, however a dialogue with local echo and character data-forwarding is used by some host computers.

A third mode, a forms entry mode may be established using the unprotected field commands. This is not the same as a local echo mode but is rather a special mode for forms entry which behaves in a similar but not identical way to local echo. When unprotected fields are established for the display screen they set up multiple record data-forwarding buffers. Data-forwarding (transmission) of all of the multiple buffers occurs at once when the data-forwarding character is typed. Order is defined with respect to the position of the start of the buffer on the screen, from top left to bottom right.

Editing capabilities for these multiple record data-forwarding buffers may vary from terminal to terminal, however it is necessary to provide some basic editing capabilities which are guaranteed to exist on all terminals in order that consistent Application Level scenarios can be designed and Information Providers can provide instruction pages in their databases. This basic editing capability proposed is described below.

Four local control keys which would move the text entry cursor position ahead one position, back one position, ahead one buffer or back one buffer would permit the active entry position to be moved about the record data-forwarding buffers without entering these characters in the buffers. Typing additional characters overwrites the data already entered at the specific active position. Typing to the end of the buffer causes the active entry position to progress to the next buffer. The Carriage Return character FE5 (CR/APR) is the data-forwarding character.

In order to interwork with conventional business terminal applications it is desirable for terminals to support all of these permutations of the dialogue described above. Control sequences may in the future be defined to switch states, however often it is necessary to be able to establish a state without the transmission of special codes in order to interact with many conventional applications. The default states should therefore be defined by switch setting on a terminal.

Although there are several combinations of these parameters, interworking with virtually all conventional business terminal applications can be achieved by establishing a single switch to set the defaults to:

Remote echo	Local Echo
Character Data-forwarding	Record Data-forwarding

The third meaningful state is local echo/character data-forwarding which may also be provided by switch settings on some terminals. Many conventional business applications would not be able to interpret the FIELD commands which are provided for in the PLP preceding the transmission of a record activation buffer and the END control command following it. It may also be desirable to provide a capability at a terminal to turn this feature off if the forms mode is to be used for other than Videotex applications.

The WAIT command provides a capability of delaying the execution of PLP commands. This affects the interaction dialogue especially in remote echo situations. In order to resolve any interaction delay problems, the WAIT command should be terminated upon any character being input by a user, and that character is handled according to the current mode.

Enquiry - The capability of enquiring from a terminal its current state or some indication of its capability, is very difficult. Certain basic functions can be provided, but providing a sophisticated polling feature would entail a complex protocol to ensure that the status of the terminal did not change between the enquiry and when the application process in the host computer receives the data. Such procedures are beyond the scope of a simple dialogue. Enquiring about the level of capability of a terminal is a procedurally dangerous operation because it tends to cause the generation of as many Application Level scenarios as there are possible features of terminals. It is tempting to simply provide information to the Application level and let it solve all differences between terminals by establishing special cases. This doesn't solve problems but merely defers them. Since the Presentation Layer Protocol has been designed to degrade gracefully to lower levels of capability, it is best to reserve the use of enquiry sequences for this purpose.

An enquiry sequence can be used to great advantage to ease the "Log-on", session establishment, scenario for a particular service. By transmitting an enquiry character ENQ (0/5) to a terminal it can respond with the Log-on sequence for that service. However, another problem develops here. In much the same manner as television sets are required to have tuners which provide equal access to all stations, it is expected that laws will develop which prevent terminals for general public sale to have a bias toward one Information Broker or another. Since terminals should not identify their owner directly, but merely a billing number in order to ensure consumer privacy, it is necessary that the enquiry function, if implemented on a terminal, be similar to a repertory telephone dialer. For those simple terminals which don't implement an automatic enquiry function, they could echo a suitable indication on the screen so that a billing number can be typed in.

ABORT - In order to provide a capability to the user to stop the display of data, a display "abort" scenario must be provided. This feature is primarily the responsibility of the application process operating in the host data base computer, however; the terminal must provide the required functionality for the feature to operate over various different commmunications means including configurations which may have significant delays. Delays over all types of networks are the basic probem. If a user wishes to abort the transmission of a page of data he enters the appropriate command, waits for it to be transmitted to the host computer, waits for host computer to respond and when the host computer has finally stopped sending data to the terminal the user must still wait until the terminal processes all of the data in its input buffer. It is this last delay which is most annoying to the user, but any technique by which the terminal would automatically clear the input buffer would not work because the terminal cannot tell when the host stops transmission to the terminal and there would always be some characters which would arrive out of context after the buffer was cleared. Inserting special flag characters may solve one part of the problem, but it creates another. Conventional business terminal applications do not transmit any special flag characters, and therefore the terminal would be restricted to only information retrieval Videotex. Since "Action Pages" and other Videotex advanced features more closely resemble conventional applications it is necessary even for Videotex alone to find a more general solution.

A technique which requires a simple code facility to be provided from the transmission levels can be used quite effectively. It will permit any abort scenario to be developed at the application level, without restricting compatibility with other services.

The basic function required is for the terminal to clear its input buffer upon receipt of an "interrupt" control character. For a character oriented transmission level communications protocol this code would be chosen to be DC4 position 1/4 in the C0 set. This code is used to clear the input buffer and it is processed immediately upon receipt, in the same manner as the other device control (DC) codes. In fact this function could be used at level 2 to break flow control deadlock situations. At the session level an "abort" signaled by the user would cause the host computer to send a DC 4 command to the terminal where it would clear the input buffer. The host computer would follow the DC 4 command with a command to reset the code environment such as IS2 (NSR/US), position 1/15 in the C0 set and whatever other commands are required to set the terminal into the state the application process requests.

The "abort" signal transmitted to the host computer would be dependent upon the particular application process and could range from a special sequence of characters to type for a convention business application or to a situation in which typing any input aborts the current page being transmitted for an information retrieval situation.

APPENDIX D - DEFINITION OF TERMS

Absolute coordinates - an ordered pair of signed numbers between -1 and 1 (non-inclusive) that specify the new location of the drawing point with respect to the origin of the unit screen. Note that only positive absolute co-ordinate specifications lie within the unit screen.

Application level - the highest of seven protocol levels defined by ISO's Reference Model of Open System Interconnection. In this case, the application level (or layer) comprises the collection of information services known as Videotex.

ASCII - American Standard Code for Information Interchange, a table of text character codes. The Canadian standard CSA Z234.4-1973 is identical to ASCII.

Attribute - a settable parameter to be applied to subsequent TEXT characters or geometric graphic primitives.

Border area - the area of the pysical display screen which is outside of the addressable display area.

C set - stands for control set. There are two control sets, C0 and C1, each of which comprises 32 character positions arranged in 2 columns by 16 rows.

CCIR - International Radio Consultative Committee, an agency of the International Telecommunication Union (I.T.U.) which develops recommendations on radio systems.

CCITT - International Telephone and Telegraph Consultative Committee, an agency of the ITU, which develops recommendations on communication systems.

Character field - the rectangular display area within which a TEXT character is defined.

Code extension - techniques for expanding the absolute character address space of a byte oriented code into a larger virtual address space.

Code table - the set of unambiguous rules that define the mapping between received bit combinations and presentation level characters.

Colour map address - an ordinal number associated with each pel in a stored digital image that determines the address in the colour map at which the actual colour value of that pel can be found. (This is sometimes abbreviated to simply "colour" when it can be done unambiguously.)

Colour map - a look up table which is used during scan conversion of the digital image that converts colour entry addresses into actual colour values.

colour value - an entry in a colour map that indicates the actual colour of the pel to be displayed.

cursor - a logical indicator of the screen position at which the next TEXT character is to be deposited. This position may or may not be marked by a cursor symbol.

DRCS - Dynamically Redefinable Character Set. The DRCS contains definable characters whose pel patterns can be downloaded from the host, or can be locally generated within a terminal.

Designate - to identify a given set from the graphics repertory as either the G0, G1, G2, or G3 set.

Display area - the addressable area of the physical display screen onto which the unit screen is mapped.

Drawing point - a logical indicator of the screen position at which the next geometric graphic primitive will commence execution. This is not normally marked by a drawing point symbol.

Escape sequence - a two, three or more byte code extension sequence beginning with the ESC character. A three character escape sequence contains an intermediate character (I) and ends with a final character (F), and is used primarily to designate a set of 94 or 96 character code sets as one of the four active G sets. Two character escape sequences contain only a final character (F) and are one method by which code sets are invoked into the in-use table.

Final character - the last character of an escape sequence.

G set - There are four G sets, G0, G1, G2, and G3, each of which comprises 94 or 96 character positions arranged in 6 columns by 16 rows.

Geometric graphic primitive - a locally stored picture drawing algorithm that can be called via a specified opcode and associated operand(s).

Graphic repertory - the collection of available graphic tables that are subject to designation as one of the G sets.

In-use - refers to the code sets or attributes that will be used to interpret or be applied to subsequently received commands.

Intermediate character - the character which occurs between the ESCAPE character and the final character is an escape sequence.

Invoke - to bring one of the four active G sets into the in-use code table.

ISO - International Organization for Standardization.

Layer - terminology adopted by ISO to describe each individual module of the reference model for Open System Interconnection (OSI). (The terms "level" and "layer" are used interchangeably).

Link level - the second of seven protocol levels defined by ISO's Reference Model of Open Systems Interconnection. The link level (or layer) is primarily responsible for establishing, maintaining, and clearing an error free communication path between network entities.

Locking shift - an invocation of a code set into the in-use table that remains in effect until another code set is invoked in its place.

Macro PDI - a locally stored string of presentation code represented with a single character name. When the macro name is used, the locally stored string is processed in its place.

Mosaic primitive - a rectangular matrix of six elements, each of which may be made up of multiple pels, which is processed as TEXT but can be used to construct block style graphic images.

Opcode - a one byte, presentation level character that initiates the execution of a locally stored geometric primitive or control operation. An opcode may be followed by one or more operands.

Operand - a single or multiple byte string from the numeric data field of the PDI code set that is used to specify control, attribute, or coordinate parameters required by the opcode.

PDI - Picture Description Instruction. A PDI is composed of an opcode followed by one or more operands and constitutes an executable picture drawing or control command.

Pel (pixel) - Picture Element. The physical picture element (pixel) is the smallest displayable unit on a given display device. The logical picture element (pel) is a geometric construct associated with the drawing point whose size determines the stroke width of graphics primitives. Although the terms pixel and pel are synonymous in common usage, throughout this document, pixel is used for physical picture elements and pel for logical picture elements.

Presentation level - the sixth of seven protocol levels defined by ISO's Reference Model of Open Systems Interconnection. The presentation level (or layer) is primarily responsible for the encoding of text, graphic, and display control information.

Protocol - a set of formats, rules, and procedures governing the exchange of information between peer processes (levels).

Relative coordinates - an ordered pair of signed numbers between -1 and 1 (non-inclusive) that specify (in two's complement arithmetic) either the new location of the drawing point with respect ot the old location of the drawing point, when used within a geometric primitive PDI, or the dimensions of a given field when used with one of the control commands.

Session level - the fifth of seven protocol levels defined by ISO's Reference Model of Open Systems Interconnection. The session level (or layer) is primarily responsible for the binding and unbinding of two presentation entities.

Single shift - an invocation of a code set into the in-use table that effects only the interpretation of the next character received. Interpretation then automatically reverts to the previous contents of the table. (This is also referred to as non-locking shift.)

TEXT - pre-defined pel patterns which, when called, are drawn with a set of pre-selected attributes at positions on the screen indicated by the cursor.

Unit screen - the virtual display address space within which all PDIs are executed and TEXT characters are deposited. The dimensions of the unit screen are 0 to 1 in the horizontal (X), vertical (Y), and depth (Z) dimensions. (The last is only defined in 3-D mode.)

APPENDIX E

On terminal implementations employing an explicit colour map, it is possible to define a default set of colours. The method for calculating these default colours is described in this section. It is important to note that these default colours are not guaranteed to be implementable on all types of hardware.

If N is the number of bits in the colour map address and M is the number of bits in the colour values [M = 3(N-1)], then the default colours are defined as follows. The first half of the colour map defines $2^N/2$ uniformly-spaced grey levels (R=G=B) ranging from black (transparent) to white. The second half of the colour map will consist of $2^N/2$ hues equally spaced around the hue circle shown in Figure E.1 and starting at 0°.

The formula used to calculate the value of R, G and B for the second (non-grey scale) half of the default colour map is as follows:

If h = desired hue ($2^N/2$ equally spaced hues starting at 0° and running counter-clockwise)

Pn = value of the nearest primary to h (eg. R, G or B)

Pi = value of the primary of intermediate distance from h

P_f = value of the furthest primary from h

ang (h) = angle of desired hue

ang (Pn) = angle of nearest primary to h

ang (Pi) = angle of the primary of intermediate distance from h.

Then, the values for the primaries, Pn, Pi and P_f is given by:

Pn = 1 (all bits set equal to 1)

$$Pi = \left| \frac{\text{ang (h)} - \text{ang (Pn)}}{60°} \right|$$

P_f = 0 (all bits set equal to 0)

The hue circle showing 8 default colours is given in Figure E.1.

The actual RGB binary values for the entire colour map for this example with 9 bits of accurary in the colour values is shown in Table E.1.

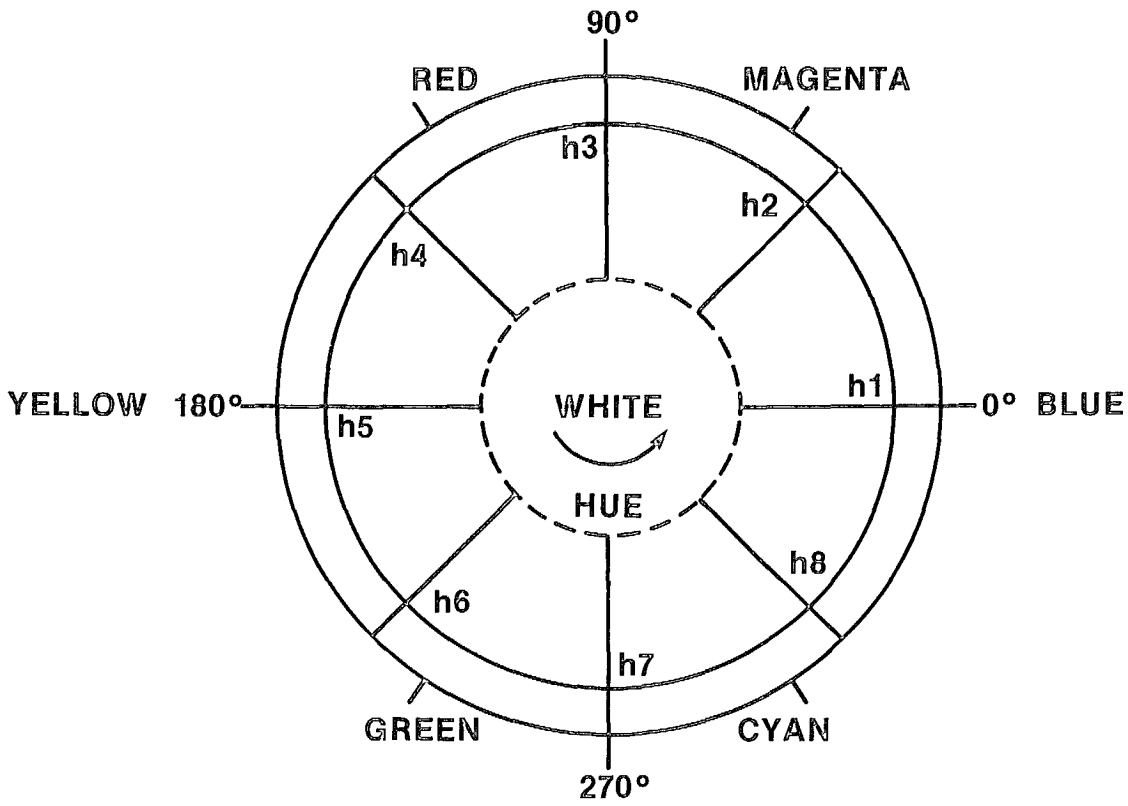


Figure E.1 - The hue circle showing an example selection of 8 default colours (h1, h2, ... h8)

Map address	Colour Values R	G	B	Colour
0 0 0 0	000	000	000	transparent (or black)
0 0 0 1	001	001	001	|
0 0 1 0	010	010	010	grey
0 0 1 1	011	011	011	scale
0 1 0 0	100	100	100	|
0 1 0 1	101	101	101	|
0 1 1 0	110	110	110	|
0 1 1 1	111	111	111	white
1 0 0 0	000	000	111	|
1 0 0 1	101	000	111	colour
1 0 1 0	111	000	100	hues
1 0 1 1	111	010	000	|
1 1 0 0	111	111	000	|
1 1 0 1	010	111	000	|
1 1 1 0	000	111	100	|
1 1 1 1	000	101	111	|

Table E.1 - Default Colour Map for 4 bits accuracy in addresses (N=2) and 9 bits accuracy in values (M=3)

REFERENCES

1. ANSI X3.4, "American National Standard Code for Information Interchange (ASCII)," 1977.

2. Bell System, "Videotex Standard: Presentation Level Protocol", May 1981 (American Telephone & Telegraph Company AT&T).

3. H.G. Bown and P.E. Allard, "On the Generation and Representation of Line Drawings", CRC Technical Note No. 689, Department of Communications, Canada, February 1978.

4. H.G. Bown, C.D. O'Brien, W. Sawchuk, and J.R. Storey, "Picture Description Instructions (PDI) for the Telidon Videotex System", CRC Technical Note No. 699, Department of Communications Canada, November 1979.

5. Canadian Standards Association, "7-Bit Coded Character Sets for Information Processing Interchange", CSA Standard Z243.4-1973.

6. CBS Television Network, "North American Broadcast Teletext Specification", June 1981.

7. CCIR, Report AD/11, "Characteristics of Teletext Systems" Document 11/5001-E, October 1981.

8. Department of Communications, Canada, Telecommunications Regulatory Service, Broadcast Specification BS-14, June 1981.

9. CCITT Recommendation S.100, "International Information Exchange for Interactive Videotex", Yellow Book, Volume VII.2, Geneva, 1980.

10. CCITT Recommendation F.300, "Videotex Service", ibid., Geneva, 1980.

11. CCITT Recommendation S.61, "Character Repertoire and Coded Character Sets for the International Teletex Service", ibid., Geneva, 1980.

12. CCITT Recommendation S.62, "Control Procedures for the Teletex Service", ibid., Geneva, 1980.

13. ISO/DIS 2022.2, "Information Processing - ISO 7-Bit and 8-Bit Coded Character Sets - Code Extension Techniques", Document TC 97/SC2 N1131, May 1981.

14. ISO 646, " 7-Bit Coded Character Set for Information Processing Interchange", July 1, 1973.

15. ISO/DP7498, "Data Processing - Open Systems Interconnection - Basic Reference Model", Document TC 97/SC16 N719, October 1981.

16. ISO/DIS 6937/2, "Coded Character Sets for Text Communication, Part 2: Latin Alphabetic and Non-Alphabetic Graphic Characters", Document TC 97 SC2 N1144, June 1981.

17. S. Shlien and P.E. Allard, "FIR Filtering Approach for the Generation of Smooth Curves on a Graphics Terminal", Computer Graphics and Image Processing, Academic Press, November 1981.

18. "Status Report of the Graphics Standards Planning Committee", Computer Graphics quarterly report of SIGGRAPH-ACM, Vol. 13, No. 3, August 1979.

19. "SMPTE Recommended Practice, (R.P. 27.3, 1972): Specifications for Safe Action and Safe Title Areas, Test Pattern for Television Systems", Society of Motion Picture and Television Engineers.

20. "Television Receiver Picture-Area Losses", Charles L. Townsend, Journal of SMPTE, Vol. 66, No. 12, Dec., 1957.

21. "Report on Home Receiver Image Area Test", R.J. Zavada, Journal of the SMPTE, Vol. 83, No. 4, April 1974.

CRC DOCUMENT CONTROL DATA

1. ORIGINATOR: Department of Communications/Communications Research Centre

2. DOCUMENT NO: CRC Technical Note No. 709-E

3. DOCUMENT DATE: February 1982

4. DOCUMENT TITLE: Telidon
Videotex Presentation Level Protocol:
Augmented Picture Description Instructions

5. AUTHOR(s): C.D. O'Brien, H.G. Bown, J.C. Smirle, Y.F. Lum and J.Z. Kukulka

6. KEYWORDS: (1) PDI
(2) Telidon
(3) Videotex

7. SUBJECT CATEGORY (FIELD & GROUP: COSATI)

14 Methods and Equipment

14 07 General Concepts

8. ABSTRACT: Refer to front of this Technical Note for Abstract.

9. CITATION:

LKC
TK5102.5 .R48e #709
c.2
Telidon : videotex
presentation level protocol
: augmented picture
description instructions

DATE DUE
DATE DE RETOUR

CARR McLEAN 38-296

CRC LIBRARY/BIBLIOTHEQUE CRC
TK5102.5 R48e #709 c. b

INDUSTRY CANADA / INDUSTRIE CANADA
212147

Government of Canada
Department of Communications

Gouvernement du Canada
Ministère des Communications