GC27-6972-1

Systems

OS/VS Graphic Programming Services (GPS) for IBM 2260 Display Station (Local Attachment)

Program Numbers	5741-GA1
	5752-GA1

IBM

Second Edition (September 1983)

This is a reprint of GC27-6972-0 incorporating changes released in the following Technical Newsletters:

GN27-1392 (dated 29 September 1972)
GN33-3056 (dated 31 July 1978)

This edition applies to Release 1 of GAM – Graphic Programming Services and to all subsequent releases until otherwise indicated in new editions or Technical Newsletters.

Changes are continually made to the information herein; before using this publication in connection with the operation of IBM systems, consult the *IBM System/360 Bibliography,* GC20-0360-0, the *IBM System/370 Bibliography,* GC20-0001-2, and the current SRL Newsletter for the editions that are applicable and current.

This publication was prepared for production using an IBM computer to update the text and to control the page and line format. Page impressions for photo-offset printing were obtained from an IBM 1403 Printer using a special print train.

References in this publication to IBM products, programs, or services do not imply that IBM intends to make these available in all countries in which IBM operates.

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

A form for readers' comments has been provided at the back of this publication. If the form has been removed, address comments to IBM United Kingdom Laboratories Limited, Product Publications, Mail Point 95, Hursley Park, Winchester, Hampshire SO21 2JN, England. IBM may use or distribute whatever information you supply in any way it believes appropriate without incurring any obligation to you.

PREFACE

This publication provides the reader with instructions and guidance in writing application programs for the IBM 2260 Display Station (Local Attachment) to run under OS/VS, using GAM - Graphic Programming Services (GPS). The macro instructions and control program routines which constitute GPS, are described in detail. Coding examples with accompanying explanations are included to illustrate uses of the macro instructions.

To take full advantage of the operating system and the programming services provided for the 2260 display station, the reader should have completed a basic course in assembler language coding or have equivalent knowledge. He should also be familiar with the contents of the following publications:

- IBM System/360 Component Description: IBM 2260 Display Station: IBM 2848 Display Control, GA27-2700
- IBM Data Processing Glossary, GC20-1699
- IBM System/370 System Summary, GA22-7001
- OS/VS and DOS/VS Assembler Language, GC33-4010
- OS/VS1 Supervisor Services and Macro Instructions, GC24-5103
- OS/VS2 Supervisor Services and Marco Instructions, GC28-0683
- OS/VS1 JCL Reference, GC24-5099
- OS/VS2 JCL Reference, GC28-0692
- OS/VS Linkage Editor and Loader, GC26-3813

For information to help determine and solve problems that involve display programs that use graphic programming services (GPS) for the 2260 display station, see:

- OS/VS Problem Determination Aids and Messages and Codes for GPS and GSP, GC27-6974

ILLUSTRATIONS

The programming support described in this publication may be used with one or more IBM 2260 Display Stations connected to an IBM System/370 central processing unit. Each display station is associated with an IBM 2848 Display Control, which is connected directly to the processing unit through either a multiplexer channel or a selector channel.

In this configuration, which is referred to as the "IBM 2260 Local," and in the "IBM 2260 Remote" configuration described below, the display station may be located anywhere within a 2000-foot cable distance from the display control.

By using the IBM 2701 Data Adapter Unit and appropriate common carrier facilities, display controls may be located any distance from the central processing unit. This configuration, which is referred to as the "IBM 2260 Remote," is supported under OS/VS BTAM.

CONFIGURATIONS AND FEATURES

There are two models of the IBM 2260 display station, Models 1 and 2. Each is a cathode-ray tube (CRT) display and is available with or without an alphameric, numeric, or data entry keyboard. The optional keyboard makes man-machine communication possible on a continuing basis at the display station. Other special features available with the IBM 2260 Local are Line Addressing and Nondestructive Cursor.

The IBM 2848 Display Control contains the interface control, a character generator, and buffer storage for the display stations. This buffer storage retains the video data to maintain display regeneration on the CRT. The data formats and number of characters that may be displayed on the face of the CRT, as well as the number and model of display stations that may be attached to a single display control, depend on which model of the display control is used, as shown in Figure 1.

Models 21 and 22 of the IBM 2848 provide special capabilities for aiding data entry applications. These include extra keyboard character buffering and input/output commands that cause the keyboard of the addressed display station to remain mechanically locked to gain the attention of the operator. Models 21 and 22 may be used only in combination with an IBM 2260 Model 2.

In addition, one IBM 1053 Printer Adapter Unit may be attached to each display control. This feature allows an IBM 1053 Printer to be used at a central location to document and record periodic system transactions. Like the display stations, the 1053 printer may be located up to 2000 feet from the display control.

The programming services described in this publication support all of the optional features of both the IBM 2260 and the IBM 2848 described in detail in IBM System/360 Component Description: IBM 2260 Display Station; IBM 2848 Display Control.

IBM 2848 Display Control					
Characteristic	Model 1	Model 2	Model 3	Model 21	Model 22
Model No. of Display Stations to be Used	2	2	1	2	2
Max. No. of Display Stations	24	16	8	24	16
No. of Lines of Characters	6	12	12	6	12
No. of Characters Per Line	40	40	80	40	40
Total No. of Characters	240	480	960	240	480

Figure 1. Display control characteristics

PROGRAMMING SUPPORT

Operating system programming support for the IBM 2260 Display Station includes macro instructions and control routines. They are used for two functions: input/output control and attention handling. The macro instructions function as part of the application program; the control routines function as part of the control program. This programming support constitutes a graphics access method for the 2260 display station.

The following paragraphs contain brief descriptions of the graphic programming services that are explained in detail in later sections. Appendixes include a sample program and CCW formats.

INPUT/OUTPUT CONTROL

Input/output control macro instructions are used for data transmission and control functions, for creating data control blocks (DCBs), and for establishing and terminating system communication between problem programs and display stations. Input/output control routines create channel command words (CCWs), issue supervisor calls to execute channel programs, and control data transmission between main storage and display station buffer storage.

ATTENTION HANDLING

Attentions are asynchronous (unscheduled) input/output interruptions of the CPU. They are caused by entries at a keyboard attached to a 2260 display station.

Basic Attention Handling

Basic attention handling facilities automatically detect the occurrence of 2260 attentions, and the control program routes them to the specified application program routines for handling. Through a set of macro instructions provided by graphic programming services (SAEC, SPAR, ATTNINQ, and DAR), the application programmer defines his attention handling routines and makes them accessible to the control program.

Express Attention Handling

Express attention handling facilities enable the application programmer to check for the occurrence of 2260 attentions through a set of macro instructions provided by graphic programming services (ANALYZ and GSERV). Detection of attentions is not automatic; the application program must check periodically for their occurrence, so that attentions are not lost. When attentions are detected, control is transferred to the specified application program routines. (Express attention handling is described in Appendix C.)

ERROR HANDLING

Synchronous errors (that is, input/output errors that occur during channel operations) are handled by the IBM 2260 Standard Error routine. This module tests the status and sense bytes for synchronous error conditions and provides appropriate error-recovery procedures for specific error situations.

The IBM 2260 Standard Error routine provides a message to the operator whenever operator intervention is required, or whenever a bus out check or equipment check occurs. These messages appear in the OS/VS1 and OS/VS2 system messages manuals.

PROBLEM DETERMINATION

For information to help determine and solve problems that involve display programs that use graphic programming services (GPS) for the 2260 display station, see OS/VS Problem Determination Aids and Messages and Codes for GPS and GSP. This manual, which is intended mainly for display operators and application programmers, includes:

Display operator aids for image problems, interaction problems, messages, job termination, and collecting problem data.

Application programmer aids, including summaries of completion and return codes, explanations of write-to-programmer messages issued by GPS, procedures for using the Generalized Trace Facility (GTF), and descriptions of problem-related data to be collected.

SYSTEM GENERATION

The 2260 programmer defines his graphic configuration to the operating system by using appropriate system generation macro instructions. System generation requirements are described in the OS/VS1 and OS/VS2 system generation reference manuals.

OS/VS2

The graphic programming services described in this publication can be used with OS/VS2. When special considerations apply to the use of these services in a VS2 environment, these considerations are noted at the end of the section to which they apply.

FORMAT ILLUSTRATION CONVENTIONS

Each macro instruction discussed in this publication is illustrated by the following coding format:

Name	Operation	Operand	Comments
Symbol or blank	Macro name	None, or one or more operands separated by commas	

The coding format illustrations use the following conventions:

- Upper case (capital) letters, numbers, and punctuation marks must be coded by the programmer exactly as shown. Exceptions to this convention are brackets, [ ], braces, { }, and ellipses, ... These three exceptions are never coded.
- Lower case (small) letters represent variables for which specific information or specific values must be substituted by the programmer when coding.
- Items within brackets, [ ], are optional. They may be omitted at the programmer's discretion.
- Braces, { }, indicate that one operand from the group must be chosen unless a default option is indicated.
- Underscoring indicates a default option; if no operand is coded, the underscored operand is assumed.
- The vertical stroke (|) signifies exclusive "or." For example, A|B means the programmer may write A or B. Alternative operands are also indicated by vertical grouping within braces { }.
- An ellipsis, ... , denotes that the preceding syntactical unit (enclosed in braces or brackets) can be written any number of times.
- Operands are separated by commas.
- All operands are written on one line of a coding sheet (unless a continuation card is used); they may appear on two or more lines in the format illustration.
- The last operand is always followed by a blank.

Positional operands must be coded in the order shown in the appropriate format illustration. Keyword operands may be written in any order, but must follow all positional operands. Commas must appear in place of omitted operands preceding the last-specified positional operand.

To determine how the operands are to be coded when substitution is required, a Summary of Operands is provided in Appendix D. This appendix contains all macro instructions and indicates the method of writing operands. The summary is designed so it can be opened and the macro instructions displayed while the descriptions are read. The classifications used in the summary are:

SYM
is any symbol valid in the assembler language.

DEC INT
are any decimal digits up to the value indicated in the associated macro instruction description. If both SYM and DEC INT are checked, an absolute expression is also allowed.

REGISTER
is always coded within parentheses as follows:

(2-12) - one of general registers 2 through 12, previously loaded with the right-adjusted value or address specified in the associated macro instruction description. The unused high-order bits must be set to zero. The register may be designated by either an absolute expression or a symbolic name enclosed in parentheses.

(1) - general register 1, previously loaded as indicated above. The register can be designated only as (1).

RX TYPE
any address that is valid in a RX-type instruction (for example, LA) may be designated.

A-TYPE ADCON TYPE
any address that may be written in an A-type address constant may be designated.

Indexing or register notation cannot be used if the L form of a macro instruction is used or if the macro-form (MF) operand is omitted; only relocatable expressions, absolute expressions, or codes may be used. When using the L-form of a macro instruction, the name field should contain a symbolic reference.

Unless otherwise noted, numeric values indicated as macro instruction operands are decimal integers, and they need not be preceded by leading zeros.

Unless otherwise noted, omission of the macro-form (MF) operand causes a macro instruction to perform both L and E functions, that is, to derive a parameter list from all the operands specified and, at execution time, to execute the macro instruction using that list.

LINKAGE CONVENTIONS

Linkage conventions described in _OS/VS Supervisor Services and Macro Instructions_ apply to the programming support described here. Information about register assignments for specific linkage functions can be found in that publication.

TERMINOLOGY

In this publication, "display station buffer storage" refers to the buffer storage physically located in the display control unit, and the unqualified term "register" means "general register."

IBM Data Processing Glossary contains a glossary of terms used in this and other OS/VS publications.

Input/output control is achieved through macro instructions and associated control routines which together initiate input/output operations associated with data display. The input/output macro instructions and their functions for the IBM 2260 Display Station are listed in Figure 2.

Note: The address of the application program save area must be loaded into register 13 before any input/output macro instruction is issued.

BACKGROUND INFORMATION

To help the reader better understand a discussion of input/output operations, the following paragraphs briefly describe the functional relationships among the various system control blocks associated with 2260 input/output operations under the operating system.

Normally, each data control block (DCB) is established by a DCB macro instruction and completed by execution of an OPEN macro instruction. The DCB, which contains a reference to a DD statement (see below), may be associated with one display station or with a group of display stations connected to a single 2848 display control. More than one DCB may be completed by a single OPEN macro instruction. The OPEN macro instruction creates a data extent block (DEB) in protected main storage. Included in the DEB is a table of address pointers to specific unit control blocks (UCBs) associated with the data control block. One UCB is specified for each display station and for each 1053 printer attached to a 2848 display control. The UCB is used to describe its associated device to the control program.

Some input/output macro instructions contain a "unit" operand. The "unit" operand refers to a device within a group of devices that can be named during system generation. This operand is used as an index to reference the desired element in the DEB list of UCB pointers. The value of the "unit" operand, expressed as a decimal integer, must not exceed the number of devices specified on the data definition (DD) card for the data set.

A DD statement can refer to any display station or 1053 printer singly or to any group of display stations (up to 24) together with a 1053 printer connected to the same display control. All units of a group need not be specified in one DD statement; subsequent DD statements can refer to the unspecified units in that group. Information about data definition specifications can be found in OS/VS JCL Reference.

Mnemonic	Macro Instruction Name	Function
DCB	Data Control Block	Establishes the data control block (DCB).
OPEN	Open Data Control Block	Initializes one or more data control blocks.
GREAD	Read Character Data	Transmits data from the display station buffer storage to main storage.
GWRITE	Write Character Data	Transmits data from main storage to the display station buffer storage or to a 1053 printer.
GCNTRL	Erase Display Station Buffer Storage	Erases data from the display station buffer storage.
CLOSE	Close Data Control Block	Disassociates one or more data control blocks.
WAIT	Wait for One or More Events	Relinquishes control of the CPU until the completion of one or more input/output operations.

Figure 2. Input/output macro instructions

The value of the "unit" operand decremented by 1 is placed in the data event control block (DECB) when it is generated during macro expansion ("unit" - 1 = UCB index). At execution time, the input/output control routine places this value into the input/output block (IOB). This completes the linkage between the input/output macro instruction and the selected UCB, which in turn points to the device associated with the macro instruction.

INPUT/OUTPUT MACRO INSTRUCTIONS

The macro instructions in this group implement 2260 input/output functions, which include defining certain control blocks and controlling data transmission between main storage and the display station buffer storage. These macro instructions provide linkage to the input/output control routine that creates the IOB, constructs channel programs, and issues calling sequences to execute the channel programs.

All input/output macro instructions except the DCB macro instruction may use the L- and E-forms of macro definition. If all operands except the MF=L or MF=E keyword operand are written, the input/output function is executed using the specified operands. If any required operand is missing at execution time the task is terminated. Detailed descriptions of these forms and their use are found in OS/VS Data Management Macro Instructions.

A 32-byte DECB is formed during macro expansion, and the parameters derived from the macro instruction operands are inserted into appropriate positions in the DECB. This data is utilized by the input/output control routine to generate the CCWs. The format of the DECB for the IBM 2260 Local is shown in Figure 3.

DCB -- Data Control Block Macro Instruction

The DCB macro instruction for the IBM 2260 Local reserves space for the data control block and establishes its format (see Figure 4). In addition to the basic information required in the DCB, space is also reserved for addresses of the input/output control routine and the input/output block, which are supplied at open time. One DCB services a maximum of 24 display stations and one 1053 printer attached to the same display control. The format of the DCB macro instruction is:

Name	Operation	Operand
[symbol]	DCB	DSORG=GS,MACRF=code [,GTYPE={BASIC|EXPRESS}] [,DDNAME=ddname] [,EXLST=exit list address] [,GNCP=number of blocks] [,POLST=poll list address]

DSORG=GS
specifies that a graphic data control block is defined. This operand is required for macro expansion.

MACRF=
specifies the type or types of macro instructions to be used, where R indicates GREAD, W indicates GWRITE, and C indicates GCNTRL. The following are

Offset from DECB Name (bytes)	DECB Field	Associated I/O Macro Instruction Operand
+00	Event control block (ECB)	
+04	Type code	type
+08	Data control block (DCB) address	dcb address
+12	Area address	area address
+16	Code for incorrect length or permanent read error	
+17	Residual count from CSW if incorrect length	
+20	Length (byte count)	length
+24	(Reserved)	
+28	Index to reference an element in the DEB table of address pointers to UCBs associated with the DCB	unit
+29	Zero (the low order bit of this byte is set to one when data entry type codes are specified)	
+30	Zeros	
+31	Zeros	

Figure 3. Format of the DECB

Offset from DCB name (bytes)	DCB Field
+0-13	Reserved
+14	GTYPE--Specifies basic or express attention handling procedures
+15-25	Reserved
+26-27	DSORG--Specifies a graphic data control block
+28-31	Reserved
+32	GNCP--Maximum number of I/O macro instructions issued before a WAIT macro instruction is issued
+33-35	POLST--Address of list of DCB addresses used in express attention handling
+36	Reserved
+37-39	EXLST--Address of application program exit list for exceptional conditions
+40-47	DDNAME--Corresponds to name of data definition (DD) statement
+48-49	Reserved
+50-51	MACRF--Types of I/O macro instructions used in problem program

Figure 4. Data control block elements

the eight valid codes for the MACRF operand.

R
W
RC
WC
(R,W)
(RC,W)
(R,WC)
(RC,WC)

GTYPE=
specifies whether basic or express attention handling procedures are to be employed. If this operand is omitted, express attention handling is assumed.

DDNAME=
specifies the name of the DD statement describing the data set being processed. DDNAME corresponds exactly to the contents of the Name field of the DD statement provided in the job control language. It can be defined by as many as eight characters, the first of which must be a letter. DDNAME furnishes a logical connection between the DD statement and the data control block of the application program.

EXLST=
specifies the address of the exit list in the application program as defined in OS/VS Data Management Macro Instructions. If EXLST is not specified, a value of zero is assumed.

GNCP=
specifies the maximum number of input/output macro instructions that can be issued before a WAIT macro instruction (that is, the number of input/output blocks constructed at open time). The value of GNCP must be from 1 to 99 at execution time. If the value of GNCP is not specified, a value of one is assumed. The programmer may specify a value of zero at assembly time, but must respecify the value within the above range at execution time. If the value zero is present at open time, the job is abnormally terminated.

POLST=
is the address of the location in main storage, reserved by the application programmer, where a list of DCB addresses is constructed by the OPEN macro instruction. This list of DCB addresses, called a poll list, is used with express attention handling facilities by the OPEN, CLOSE, GSERV, and ANALYZ macro instructions as described in Appendix C of this publication. The first entry in the poll list points to the last entry. (It must originally be set to zeros.) As additional devices are opened, the DCB addresses are added to the poll list. The length of the area needed to construct the list is n+1 fullwords, where "n" is the number of devices for which the DCB is opened. If the application programmer wishes to open multiple DCBs using the same OPEN macro instruction, he must use the same POLST operand with each DCB macro instruction he issues. He may use the same POLST operand or different POLST operands when issuing other OPEN macro instructions. If more than one poll list is created, each requires a separate ANALYZ macro instruction. If POLST is not specified, a value of zero is assumed. This operand is used only with express attention handling.

PROGRAMMING NOTES: The application programmer may create a DCB at assembly time by using the DCB macro instruction. Data definition (DD) card information may be supplied by the programmer at execution time. The GNCP operand may be specified at execution time by a DD card, provided GNCP is set to zero at assembly time.

The application programmer can create his own DCB by using a constant area. In this case he may specify both the DSORG and GNCP operands in a DD card. By ensuring that this area contains the same information at the same locations as the DCB created by the DCB macro expansion (see Figure 4), the programmer can effectively create a DCB. The DD name of the DD card must be identical to the DD name set up in this area (offset 40 through 47). For information on coding of the DD card, refer to OS/VS JCL Reference.

The data control block assembled from a DCB macro instruction will not have names associated with the individual fields that comprise the control block. To refer or gain access to the fields in the data control block, the application programmer can write a DCBD macro instruction as follows:

```
     DCBD  DSORG=GS
```

The name field must be left blank, and only the DSORG operand is specified.

The DCBD macro instruction generates a DSECT statement that provides a symbolic name for the fields within a data control block. Each field is defined so that with proper initialization of base registers, the programmer can refer to any or all fields of one or more data control blocks. The macro instruction expands as follows:

```
+*                      DCB SYMBOLIC DEFIN-
+*                       ITION FOR GRAPHICS
+*                       WITH APPENDAGES

+IHADCB   DSECT

+         ORG  IHADCB+0

+*                      GRAPHIC DEVICE
+*                       INTERFACE

+         DS   BL4
+         DS   BL4
+         DS   BL4
+DCBBRSA  DS   AL2
+DCBGTYPE DS   BL1
+         DS   BL1
+DCBBFRST DS   AL2
+DCBBFRSZ DS   AL2
```

```
+*                      COMMON

+         ORG  IHADCB+20
+         DS   BL6
+DCBDSORG DS   BL2
+DCBIOBAD DS   A

+*                      FOUNDATION EXTENSION

+         ORG  IHADCB+32
+DCBGNCP  DS   0BL1
+DCBPOLST DS   A
+DCBEXLST DS   A

+*                      FOUNDATION BEFORE
+*                       OPEN

+         ORG  IHADCB+40
+DCBDDNAM DS   CL8
+DCBOFLG  DS   BL1
+DCBIFLG  DS   BL1
+DCBMACR  DS   BL2

+*                      FOUNDATION AFTER
+*                       OPEN

+         ORG  IHADCB+40
+DCBTIOT  DS   AL2
+DCBMACRF DS   BL2
+DCBIFLGS DS   0BL1
+DCBDEBAD DS   A
+DCBOFLGS DS   0BL1
+DCBGIOCR DS   A
```

For more information, see OS/VS Data Management Macro Instructions.

OPEN -- Open Data Control Block Macro Instruction

The OPEN macro instruction for the IBM 2260 Local initializes one or more data control blocks that establish a communication path between the application program and the display station. OPEN performs the following functions:

Indicates that the display station is "open" by setting the appropriate bit in the DCB.

Fills in and initializes appropriate fields in the DCB.

Acquires main storage to initialize a chain of IOBs and associated channel programs. Linkage is provided in the IOB to its CCW list and to the next IOB in the chain. The number of IOBs constructed is one greater than the number specified by the GNCP operand, unless GNCP is equal to 99. (No more than 99 IOBs may be constructed for each DCB.) The additional IOB is constructed in order to always ensure the availability of an IOB, since IOBs are not constructed dynamically. A pointer to the top of the chain of IOBs is placed in the DCB.

- Loads the graphic I/O control routine and places its address in the DCB.

- Performs initialization functions required by attention handling facilities. If the POLST operand was specified in the DCB macro instruction, a DCB address list (poll list) is constructed. OPEN provides one POLST entry for every display station that the user wants polled. This function applies only to 2260s equipped with an optional keyboard feature.

The format of the OPEN macro instruction is:

Name	Operation	Operand
[symbol]	OPEN	(dcb address [,,dcb address]...) [,MF={L|(E,parameter list name)|(E,(1))}]

dcb address
specifies the addresses of the DCBs to be initialized for the desired graphic device(s). Any number of DCB addresses may be specified. This allows parallel opening of DCBs and their associated data sets. If more than one DCB address is specified, the "dcb address" operands must be separated by two commas. The second comma indicates omission of optional operands not applicable to graphic programming services.

MF=L
specifies that expansion of the macro instruction is to consist of a parameter list only. The name assigned to the parameter list is the symbolic name specified in the Name field of this OPEN macro instruction.

MF=(E,parameter list name)
specifies that the macro instruction is to perform the open function using a parameter list created by another OPEN macro instruction. The open function is executed for each "dcb address" operand in the list. The "parameter list name" operand must correspond exactly to the name specified in the Name field of the OPEN macro instruction used to construct the parameter list.

MF=(E,(1))
performs the same function as (E, parameter list name). By writing the special register notation (1), the programmer specifies that he has loaded the address of the parameter list into register 1 before execution of the macro instruction.

PROGRAMMING NOTES: If a task attempts to open a DCB that was already opened by another task, the task is abnormally terminated. If a task attempts to open a graphics DCB for a nongraphics device, the task is abnormally terminated. If a task attempts to have OPEN construct a number of IOBs that is not in the range 1 through 99, the task is abnormally terminated. Write-to-programmer messages are issued for these error conditions. For message explanations and programmer error checks, see OS/VS Problem Determination Aids and Messages and Codes for GPS and GSP.

If the "dcb address" operand of the OPEN macro instruction specifies a DCB that was already opened (by the same task), no action results. Opening when a corresponding DD statement has not been provided causes no error indication; however, an attempt to use the data set will result in an abnormal termination of the task.

If the device specified on the DD card is not a graphic device, an invalid UCB results, and the task is abnormally terminated.

EXAMPLES: In the second and third examples, DCB1 and DCB2 are initialized using the parameter list constructed by the first example. In the third example, the location of the parameter list has been loaded into register 1.

```
EXAMP1    OPEN    (DCB1,,DCB2),MF=L
EXAMP2    OPEN    MF=(E,EXAMP1)
EXAMP3    OPEN    MF=(E,(1))
```

GREAD -- Read Character Data Macro Instruction

The GREAD macro instruction transmits data from the display station buffer storage to main storage. There are six possible read operations performed by the GREAD macro instruction depending upon the "type" operand specified. The format of the GREAD macro instruction is:

Name	Operation	Operand
[symbol]	GREAD	decb name ,type ,dcb address ,[length] ,area address [,unit] [,MF={L|E}]

decb name

is the name assigned to the data event control block generated in the macro expansion. This operand is required for macro expansion.

type

specifies the type of read operation. This operand is required for proper macro expansion. The following are the valid forms of the operand (each "type" is described following the discussion of all other operands related to the GREAD macro instruction):

DSM
Read Display Station Manual Input Message for the 2260.

DSMW
Read Display Station Manual Input Message Without Keyboard Restore for the 2260.

SMI
Read Display Station Short Manual Input Message for the 2260.

SMIW
Read Display Station Short Manual Input Message Without Keyboard Restore for the 2260.

DSB
Read Full Display Station Buffer Storage for the 2260.

DSBW
Read Full Display Station Buffer Storage Without Keyboard Restore for the 2260.

dcb address

is the address of the data control block associated with the graphic device.

length

is the number of data bytes to be read.

area address

is the address of the main storage location into which the first byte of data will be read.

unit

is a decimal integer; it is used to create an index to the desired element in the DEB table (address pointers to UCBs associated with the DCB). The indicated UCB contains the address of the desired device. The value of "unit" must be in the range 1 to N, where N is the number of devices specified on the data definition card for this data set. N cannot exceed 25 (the maximum configuration is 24 display stations and one 1053 Printer per display control).

Caution: When using register notation, the contents of the register, at execution, must be one less than the value of the "unit" operand described above.

Note: The device indicated by "unit" cannot be the 1053 printer.

Omission of the "unit" operand results in the value one being assigned to "unit", indicating the first element in the DEB table.

MF=L

specifies that expansion of the macro instruction is to consist of a parameter list only. The name assigned to the parameter list is the name specified by the "decb name" operand of this GREAD L-form macro instruction.

MF=E

specifies that the macro instruction is to perform the read operation using a parameter list constructed by another GREAD macro instruction. The first operand in the E-form macro instruction must correspond exactly to the name of the parameter list. The parameter list may be updated at execution time by specifying new values in any of the operands preceding the MF=E keyword operand.

A "type" operand, specified in an L-form macro instruction, may be modified by a corresponding E-form macro instruction. Other operands required for execution that were not specified in an L-form macro instruction should be specified in the corresponding E-form macro instruction.

Note: A unique name must be assigned to each parameter list defined. Use of the same DECB name for more than one DECB results in a multiple definition error at assembly time.

If neither the MF=L nor MF=E keyword operand is written, all required operands must be coded to achieve executable macro expansion. Omission of the "decb name" operand prevents macro expansion; omission of other operands produces error messages on the assembly listing, but does not terminate macro expansion.

TYPES OF READ OPERATIONS: The six types of read operations for the 2260 are described below.

DSM (Read Display Station Manual Input Message): This "type" operand causes character data in the selected display station buffer storage to be transferred to main storage by the display control. DSM can be used only if an optional keyboard feature is available. Starting with the first display position following the start symbol, character data will be sequentially retrieved and transferred until the end of message is detected or the channel byte count is reduced to zero. The start symbol is then cleared. Data on the same line and to the right of a new line symbol is not transferred. The new line symbol is transferred. If the check symbol is present, it is transferred. The end of message code is not transferred to main storage. The deletion of the start symbol is the only change to the CRT display; the data is not changed or destroyed by the read operation.

One channel command word is generated by GREAD, "type" DSM: the Read Display Station Manual Input Message CCW, which initiates the read operation.

DSMW (Read Display Station Manual Input Message Without Keyboard Restore): The operation caused by this "type" operand, which can only be used with IBM 2848 Models 21 and 22, is identical to DSM, except that the keyboard remains mechanically locked at the completion of the operation.

One channel command word is generated by GREAD, "type" DSMW: the Read Display Station Manual Input Message Without Keyboard Restore CCW, which initiates the read operation.

SMI (Read Display Station Short Manual Input Message): The operation caused by this "type" operand is identical to DSM, except that the start symbol is not erased when the end of message is detected or the channel byte count is reduced to zero. Consequently, keyboard lockout time is substantially less for the SMI "type" operand than for the DSM "type" operand. This is an important consideration for applications involving a high rate of data input and multiple 2260s attached to one 2848.

One channel command word is generated by GREAD, "type" SMI: the Read Display Station Short Manual Input Message CCW, which initiates the read operation.

SMIW (Read Display Station Short Manual Input Message Without Keyboard Restore): The operation caused by this "type" operand, which can only be used with IBM 2848 Models 21 and 22, is identical to SMI, except that the keyboard remains mechanically locked at the completion of the operation.

One channel command word is generated by GREAD, "type" SMIW: the Read Display Station Short Manual Input Message Without Keyboard Restore CCW, which initiates the read operation.

DSB (Read Full Display Station Buffer): This "type" operand causes a maximum of 240, 480, or 960 data bytes to be transferred, depending upon the size of the addressed display station buffer. The first symbol transferred is the first symbol in the upper left-hand corner of the CRT, and the last symbol transferred is the last symbol in the lower right-hand corner of the CRT. If no character is found in a displayable position, a space is transferred. If the addressed buffer has the optional nondestructive cursor feature, the cursor position is not indicated by the data transferred. If the addressed buffer has a standard destructive cursor feature, the cursor position is indicated by the location of the end of message code. (For more information about the cursor features and the end of message code, see the 2260 component description.) The check and start symbols are transferred if they are present. The codes are:

bit	0	1	2	3	4	5	6	7
Destructive cursor	0	1	1	0	1	0	1	0
Check symbol	0	1	1	1	1	1	1	1
Start symbol	0	1	0	0	1	0	1	0

The CRT is erased after the operation, and the cursor is placed in the first cursor position (upper left-hand corner) of the CRT.

Multiple channel command words are generated by GREAD, "type" DSB. First, the Read Full Display Station Buffer CCW is issued, which initiates the read operation. Second, the Erase Display Station Buffer Storage CCW is issued, which deletes all character data from the display station buffer storage and resets the cursor to the first display position of the CRT.

DSBW (Read Full Display Station Buffer Without Keyboard Restore): The operation caused by this "type" operand, which can only be used with IBM 2848 Models 21 and 22, is identical to DSB, except that the keyboard remains mechanically locked at the completion of the operation.

Multiple channel command words are generated by GREAD, "type" DSBW. First, the Read Full Display Station Buffer Without Keyboard Restore CCW is issued, which initiates the read operation. Second, the Erase Display Station Buffer Storage Without Keyboard Restore CCW is issued, which deletes all character data from the display station buffer storage and resets the cursor to the first display position of the CRT.

PROGRAMMING NOTES: If the channel byte count in the CCW reaches zero, a stop sequence is generated by the channel. During a read operation, receipt of the stop sequence terminates transmission of data from the display control to main storage. The display control continues reading data from the display station buffer storage but does not transfer data to main storage. Detection of the end of message when using DSM causes the display control to restore the keyboard of the selected display station and delete the start symbol. When SMI is used, the keyboard is restored and the start symbol is not erased. When DSB is used, the display is erased at completion of the operation, and the cursor is reset. If an error is detected during execution of DSB, the erase operation is not performed. The application programmer, therefore, should ensure that the correct length is specified if he wants the erase operation to be performed.

An unspecified "length" operand or zero entry results in the insertion of a byte count of 960 in the generated CCW. This assures that the stop sequence will not be encountered on read operations. If the "length" operand is incorrectly specified, an error message is printed on the assembly listing.

ERROR CONDITIONS: If incorrect length occurs during execution, the count from the CSW is placed in its appropriate location in the DECB (DECB+17 through 19), and a hexadecimal code of 43 is posted in the DECB (DECB+16). Since this case is not considered an error, a hexadecimal code of 7F is also posted in the high-order byte of the ECB. If a permanent read error occurs, the screen is erased, the message ERROR ON READ is displayed, an error code of 41 is posted in the high-order byte (byte 0) of the ECB, and a hexadecimal code of 45 is posted in the DECB (DECB+16). Automatic retry is not provided, and the handling of the error is entirely the responsibility of the application program.

Before processing the data obtained from a read operation, the application program should check the high-order byte of the ECB to determine whether the operation was successful. The program should then check bytes 16 through 19 of the DECB for more information about the completion condition. If the DECB is to be used for more than one input/output operation, the program must zero DECB byte 16 before execution of the next input/output operation. The indications and conditions present when determining the completion condition, and the read operation affected, are shown in Figure 5.

Indication	Condition	Read Operation
1. ECB=7F DECB+16=0 DECB+17=not used	Normal Completion	DSM, DSMW, SMI, SMIW, DSB, DSBW
2. ECB=41 DECB+16=45 DECB+17=not used	Permanent Error	DSM, DSMW, SMI, SMIW, DSB, DSBW
3. ECB=7F DECB+16=43 DECB+17=0 CSW count=0	Long Length Record (The CSW count reached zero before the end of message symbol was detected) (The CSW count reached zero before the full buffer was read)	DSM, DSMW, SMI or SMIW DSB or DSBW
4. ECB=7F DECB+16=43 DECB+17=count CSW count not equal to initial count and not zero	Short Length Record (The end of message symbol was encountered before the CSW count reached zero) (The full buffer was read before the CSW count reached zero)	DSM, DSMW, SMI or SMIW DSB or DSBW
5. ECB=7F DECB+16=43 DECB+17=initial count	a. The ENTER key was depressed when the start symbol was not on the screen b. A DSM was given when the ENTER key had not been depressed c. The ENTER key was depressed before any data was entered (the start symbol is followed immediately by the cursor)	DSM, DSMW, SMI or SMIW

Figure 5. Completion indications and conditions

After execution of the macro instruction, the program should check the hexadecimal return code in register 15 to determine whether execution was successful. Return codes and their meanings are:

Return Code	Meaning
00	Normal
10	DCB address is not in the DECB
14	DCB addresses in the DECB and the IOB are not equal
1C	Type code is not in the DECB
24	Area address is not in the DECB
28	DCB addresses in the DECB and the DEB are not equal
2C	Unit index in the DECB exceeds units specified in the DD record
3C	Length in the DECB is negative

Write-to-programmer messages are issued for the error conditions. For message explanations and programmer error checks, see OS/VS Problem Determination Aids and Messages and Codes for GPS and GSP.

EXAMPLES: In the following example, a DECB named DECB1 is to be generated as part of the macro expansion. The "type" operand specifies that a DSM operation for the data set associated with the data control block INDCB will be performed. A length of 100 bytes will be read into an area in main storage called INAREA. The value of the "unit" operand is one. In example 1, L-form is specified, which generates the parameter list. Example 2 shows coding when positional operands are omitted in an L-form GREAD. Example 3 shows the E-form GREAD with the length changed to 150 bytes.

```
EXAMP1 GREAD     DECB1,DSM,INDCB,100,
                 INAREA,1,MF=L
EXAMP2 GREAD     DECB1,DSM,MF=L
EXAMP3 GREAD     DECB1,,,150,MF=E
```

Register Notation: The programmer may specify register notation, in which case the address of the DECB must be loaded into register 1 before execution of the macro instruction. Other positional operands may be loaded into other registers (2-12). For example:

```
EXAMP1 LA        1,DECB
       GREAD     (1),MF=E
or
EXAMP2 LA        1,DECB1
       LA        2,INDCB
       LA        3,INAREA
       GREAD     (1),,(2),,(3),MF=E
```

When expressing the "unit" operand in register notation, the value loaded into the register is the same as the UCB index (unit-1) and is placed in the high-order byte of the register.

GWRITE -- Write Character Data Macro Instruction

The GWRITE macro instruction transmits data from main storage to the display station buffer storage or to a 1053 printer. There are eight possible write operations performed by the GWRITE macro instruction depending upon the "type" operand specified. The format of the GWRITE macro instruction is:

Name	Operation	Operand
[symbol]	GWRITE	decb name ,type ,dcb address ,[length] ,area address [,unit] [,MF={L|E}]

decb name
is the name assigned to the data event control block generated in the macro expansion. This operand is required for macro expansion.

type
specifies the type of write operation. This operand is required for proper macro expansion. The following are the valid forms of the operand (each "type" is described following the discussion of all other operands related to the GWRITE macro instruction):

DSB
Write Display Station Buffer Storage for the 2260, or Write 1053 Printer Buffer Storage for the 1053.

DSBW
Write Display Station Buffer Storage Without Keyboard Restore for the 2260.

EBW
Erase and Write Display Station Buffer Storage for the 2260.

EBWW
Erase and Write Display Station Buffer Storage Without Keyboard Restore for the 2260.

LNE
Write Display Station Line Address for the 2260.

LNEW
Write Display Station Line Address Without Keyboard Restore for the 2260.

EWL
Erase and Write Display Station Line Address for the 2260.

EWLW
Erase and Write Display Station Line Address Without Keyboard Restore for the 2260.

dcb address
is the address of the data control block associated with a graphic device.

length
is the number of bytes of display data to be written. This includes the new line character and the line address control byte when used.

area address
is the address of the main storage location from which the first byte of display data is to be written.

unit
is a decimal integer; it is used to create an index to the desired element in the DEB table (address pointers to UCBs associated with the DCB). The indicated UCB contains the address of the desired device. The value of "unit" must be in the range 1 to N, where N is the number of devices specified on the data definition card for this data set. N cannot exceed 25 (the maximum configuration is 24 display stations and one 1053 printer per display control).

CAUTION: When using register notation, the content of the register, at execution, must be one less than the value of the "unit" operand described above.

Note: The 1053 printer can be indicated by "unit" only when the "type" operand is written as DSB.

Omission of the "unit" operand results in the value one being assigned to "unit", indicating the first element in the DEB table.

MF=L
specifies that expansion of the macro instruction is to consist of a parameter list only. The name assigned to the parameter list is specified by the "decb" operand of this GWRITE L-form macro instruction.

MF=E
specifies that the macro instruction is to perform the write operation using a parameter list constructed by another GWRITE macro instruction. The first operand in the E-form macro instruction must correspond exactly to the name of the parameter list. The parameter list may be updated at execution time by specifying new values in any of the operands preceding the MF=E operand.

A "type" operand, specified in an L-form macro instruction, may be modified by a corresponding E-form macro instruction. Other operands required for execution that were not specified in an L-form macro instruction should be specified in the corresponding E-form macro instruction.

Note: A unique name must be assigned to each parameter list defined. Use of the same DECB name for more than one DECB results in a multiple definition error at assembly time.

If neither the MF=L nor MF=E keyword operand is written, all required operands must be coded to achieve executable macro expansion. Omission of the "decb name" operand prevents macro expansion; omission of other operands produce error messages on the assembly listing, but do not terminate macro expansion.

TYPES OF WRITE OPERATIONS: The eight types of write operations for the 2260 are described below.

DSB (Write Display Station Buffer Storage): This "type" operand causes sequential bytes of character data to be transmitted to the display control in either of two modes depending upon whether the address is specified as display station or 1053 printer.

Upon recognition of the display station address by the display control, character data is placed into the display station buffer storage starting at the cursor symbol position. The cursor is automatically advanced to the next cursor position. The cursor is placed in the first display position (upper left-hand corner) of the CRT when a displayable character is inserted into the last display position of the CRT.

When the 1053 printer address is recognized by the display control, incoming character data is placed into contiguous print positions of 1053 buffer storage and the 1053 print operation is initiated. A new line character should be inserted at least after every 130th character; otherwise, overprinting results.

If a write error occurs during execution of a GWRITE, "type" DSB, operation, the system error recovery procedure erases the screen completely before initiating a retry.

One channel command word is generated by GWRITE, "type" DSB: the Write Display Station Buffer Storage CCW, which initiates the write operation.

DSBW (Write Display Station Buffer Storage Without Keyboard Restore): The operation caused by this "type" operand, which can only be used with IBM 2848 Models 21 and 22, is identical to DSB, except that the keyboard remains mechanically locked out at the completion of the operation.

One channel command word is generated by GWRITE, "type" DSBW: the Write Display Station Buffer Storage Without Keyboard Restore CCW, which initiates the write operation.

EBW (Erase and Write Display Station Buffer Storage): This "type" operand causes the display control to delete all character data from the display station buffer storage, and to reset the cursor to the first display position (upper left-hand corner) of the CRT. The display data is then placed into display station buffer storage, starting with the position at which the cursor is reset.

Two channel command words are generated by GWRITE, "type" EBW. First, the Erase Display Station Buffer Storage CCW is issued to delete all character data from the display station buffer storage and reset the cursor to the first display position of the CRT. Second, the Write Display Station Buffer Storage CCW is issued to initiate the write operation.

EBWW (Erase and Write Display Station Buffer Storage Without Keyboard Restore): The operation caused by this "type" operand, which can only be used with the IBM 2848 Models 21 and 22, is identical to EBW, except that the keyboard remains mechanically locked at the completion of the operation.

Two channel command words are generated by GWRITE, "type" EBWW. First, the Erase Display Station Buffer Storage Without Keyboard Restore CCW is issued to delete all character data from the display station buffer storage and reset the cursor to the first display position of the CRT. Second, the Write Display Station Buffer Storage Without Keyboard Restore CCW is issued to initiate the write operation.

LNE (Write Display Station Line Address): This "type" operand, which can only be used if the line addressing special feature is available, causes the display control to transmit sequential bytes of character data into the display station buffer storage under channel byte count control. The starting location in the display station buffer storage, as indicated in the first data byte, is interpreted as a control byte. This byte specifies the line to which the display data will be transmitted (see Figure 6). The cursor is moved to the first display position of the line indicated and display data is placed into display station buffer storage starting at the position of the cursor symbol. The cursor is automatically advanced to the next display position.

LNE performs essentially the same operation as DSB. The basic difference is that information in the first data byte (a hexadecimal value in the range F0 through FB), which must be supplied by the user, specifies the line to which the display data will be transmitted.

One channel command word is generated by GWRITE, "type" LNE: the Write Display Station Line Address CCW, which initiates the write operation.

LNEW (Write Display Station Line Address Without Keyboard Restore): The operation caused by this "type" operand, which can only be used with IBM 2848 Models 21 and 22, is identical to LNE, except that the keyboard remains mechanically locked out at the completion of the operation.

One channel command word is generated by GWRITE, "type" LNEW: the Write Display Station Line Address Without Keyboard Restore CCW, which initiates the write operation.

Bit Positions 0 1 2 3 4 5 6 7	Hex. Equiv.	Selected Line
1 1 1 1 0 0 0 0	F0	one
1 1 1 1 0 0 0 1	F1	two
1 1 1 1 0 0 1 0	F2	three
1 1 1 1 0 0 1 1	F3	four
1 1 1 1 0 1 0 0	F4	five
1 1 1 1 0 1 0 1	F5	six
1 1 1 1 0 1 1 0	F6	seven
1 1 1 1 0 1 1 1	F7	eight
1 1 1 1 1 0 0 0	F8	nine
1 1 1 1 1 0 0 1	F9	ten
1 1 1 1 1 0 1 0	FA	eleven
1 1 1 1 1 0 1 1	FB	twelve

Figure 6. Format of first data byte for line addressing operations

EWL (Erase and Write Display Station Line Address): This "type" operand, which can be used only if the line addressing special feature is available, causes the display control to delete all character data from the display station buffer storage, and resets the cursor to the first display position (upper left-hand corner) of the CRT. The cursor is moved to the first display position of the specified line, and the display data is placed into the display station buffer storage starting at the position of the cursor symbol. The cursor is automatically advanced to the next display position.

EWL executes the same operation as LNE, except that an erase operation is performed first.

Two channel command words are generated by GWRITE, "type" EWL. First, the Erase Display Station Buffer Storage CCW is issued to delete all character data from the display station buffer storage and reset the cursor symbol. Second, the Write Display Station Line Address CCW is issued to initiate the write operation.

EWLW (Erase and Write Display Station Line Address Without Keyboard Restore): The operation caused by this "type" operand, which can only be used with the IBM 2848 Models 21 and 22, is identical to EWL, except that the keyboard remains mechanically locked at the completion of the operation.

Two channel command words are generated by GWRITE, "type" EWLW. First, the Erase Display Station Buffer Storage Without Keyboard Restore CCW is issued to delete all character data from the display station buffer storage and reset the cursor symbol. Second, the Write Display Station Line Address Without Keyboard Restore CCW is issued to initiate the write operation.

ERROR CONDITIONS: A normal completion is indicated by a hexadecimal code of 7F in the high order byte of the ECB (byte 0). If a permanent error occurs, a hexadecimal code of 41 is posted in this byte.

After execution of the macro instruction, the program should check the hexadecimal return code in register 15 to determine whether execution was successful. Return codes and their meanings are:

Return Code	Meaning
00	Normal
10	DCB address is not in the DECB
14	DCB addresses in the DECB and the IOB are not equal
1C	Type code is not in the DECB
24	Area address is not in the DECB
28	DCB addresses in the DECB and the DEB are not equal
2C	Unit index in the DECB exceeds units specified in the DD record
3C	Length in the DECB is negative

Write-to-programmer messages are issued for the error conditions. For message explanations and programmer error checks, see OS/VS Problem Determination Aids and Messages and Codes for GPS and GSP.

EXAMPLES: In the following examples, a DECB named DECB1 is to be generated as part of the macro expansion. The "type" operand specifies a DSB operation for the data set associated with the data control block OUTDCB. One hundred bytes of data will be written from an area in main storage called OUTAREA. The value of the "unit" operand is 1. In Example 1, L-form is specified to generate the parameter list. Example 2 shows coding for an L-form GWRITE macro instruction when positional operands are omitted. Example 3 shows coding for an E-form GWRITE macro instruction with the length changed to 150 bytes.

```
EXAMP1 GWRITE    DECB1,DSB,OUTDCB,100,
                 OUTAREA,1,MF=L

EXAMP2 GWRITE    DECB1,DSB,MF=L

EXAMP3 GWRITE    DECB1,,,150,MF=E
```

Register Notation: The programmer may specify register notation. The address of the DECB must be loaded into register 1 before execution of the macro instruction. Other positional operands may be loaded into other registers (2-12). For example:

```
EXAMP1 LA        1,DECB1
       GWRITE    (1),MF=E
```

or

```
EXAMP2 LA        1,DECB1
       LA        2,OUTDCB
       LA        3,OUTAREA
       GWRITE    (1),,(2),,(3),MF=E
```

When expressing the "unit" operand in register notation, the value loaded into the register is the same as the UCB index (unit-1), and is placed in the high-order byte of the register.

GCNTRL -- Erase Display Station Buffer Storage Macro Instruction

The GCNTRL macro instruction erases data from the display station buffer storage. There are two possible erase operations performed by the GCNTRL macro instruction depending upon the "type" operand specified. The format of the GCNTRL macro instruction is:

Name	Operation	Operand
[symbol]	GCNTRL	decb name ,type ,dcb address [,unit] [,MF={L|E}]

decb name
: is the name assigned to the data event control block. This operand is required for macro expansion.

type
: specifies the type of control operation. This operand is required for proper macro expansion. The following are the valid forms of the operand (each "type" is described following the discussion of all other operands related to the GCNTRL macro instruction):

ERS
: Erase Display Station Buffer Storage for the 2260.

ERSW
: Erase Display Station Buffer Storage Without Keyboard Restore for the 2260.

dcb address
: is the address of the data control block associated with the graphic device.

unit
: is a decimal integer; it is used to create an index to the desired element in the DEB table (address pointers to UCBs associated with the DCB). The indicated UCB contains the address of the desired device. The value of "unit" must be in the range 1 to N, where N is the number of devices specified on the data definition card for this data set. N cannot exceed 25 (the maximum configuration is 24 display stations and one 1053 Printer per display control).

Note: The device indicated by the "unit" operand cannot be the 1053 printer, since an erase operation is invalid for the 1053 printer.

Omission of the "unit" operand results in a value of one being assigned to "unit", indicating the first element in the DEB table. If "unit" is incorrectly specified, an error message is printed on the assembly listing.

MF=L
: specifies that expansion of the macro instruction is to consist of a parameter list only. The name assigned to the parameter list is the name specified by the "decb name" operand of this GCNTRL L-form macro instruction.

MF=E
: specifies the macro instruction is to perform the erase function using a parameter list constructed by another GCNTRL macro instruction. The first operand in the E-form macro instruction must correspond exactly to the name of the parameter list. The parameter list may be updated at execution time by specifying new values in any of the operands preceding the keyword operand.

A "type" operand, specified in a L-form macro instruction, may be modified by a corresponding E-form macro instruction. Other operands required for execution that were not specified in an L-form macro instruction should be specified in the corresponding E-form macro instruction.

Note: If neither the MF=L nor MF=E keyword operand is written, all required operands must be coded to achieve executable macro expansion. Omission of the "decb name" operand prevents macro expansion; omission of other operands produce error messages on the assembly listing, but do not terminate macro expansion.

TYPES OF CONTROL OPERATIONS: The two types of control operations for the 2260 are described below.

ERS (Erase Display Station Buffer Storage): This "type" operand causes the display station to delete all character data from the display station buffer storage. The cursor symbol is placed in the first display position of the CRT (upper left-hand corner), and the remainder of the display is blank.

GCNTRL, "type" ERS, generates one channel command word, the Erase Display Station Buffer Storage Control CCW, which initiates the erase operation.

ERSW (Erase Display Station Buffer Storage Without Keyboard Restore): The operation caused by this "type" operand, which can only be used with IBM 2848 Models 21 and 22, is identical to ERS, except that the keyboard remains mechanically locked out at the completion of the operation.

One channel command word is generated by GCNTRL, "type" ERSW: the Erase Display Station Buffer Storage Control Without Keyboard Restore CCW, which initiates the erase operation.

ERROR CONDITIONS: A normal completion is indicated by a hexadecimal code of 7F in the high order byte of the ECB (byte 0). If a permanent error occurs, a hexadecimal code of 41 is posted in this byte.

After execution of the macro instruction, the program should check the hexadecimal return code in register 15 to determine whether execution was successful. Return codes and their meanings are:

Return Code	Meaning
00	Normal
10	DCB address is not in the DECB
14	DCB addresses in the DECB and the IOB are not equal
1C	Type code is not in the DECB
24	Area address is not in the DECB
28	UCB addresses in the DECB ano the DEB are not equal
2C	Unit index in the DECB exceeds units specified in the DD record

Write-to-programmer messages are issued for the error conditions. For message explanations and programmer error checks, see OS/VS Problem Determination Aids and Messages and Codes for GPS and GSP.

EXAMPLES: In the coding below, a DECB named DECB1 is to be addressed. The "type" operand specifies that an ERS operation for the data set associated with the data control block OUTDCB will be performed. The value of the "unit" operand is one. L-form is specified in Example 1. Example 2 shows coding for an L-form GCNTRL macro instruction when positional operands are omitted. Example 3 shows coding for an E-form GCNTRL macro instruction when a positional operand is changed.

```
EXAMP1 GCNTRL     DECB1,ERS,OUTDCB,1,
                  MF=L

EXAMP2 GCNTRL     DECB1,ERS,MF=L

EXAMP3 GCNTRL     DECB1,,,4,MF=E
```

Register Notation: The programmer may specify register notation. In such cases, he must load the address of the DECB into register 1 before execution of the macro instruction. Other positional operands may be loaded into other registers (2-12). For example:

```
EXAMP1 LA         1,DECB1
       GCNTRL     (1),MF=E
or

EXAMP2 LA         1,DECB1
       LA         2,4
       SLL        2,24
       GCNTRL     (1),,,(2),MF=E
```

When expressing the "unit" operand in register notation, the value loaded into the register is the same as the UCB index (unit-1), and is placed in the high-order byte of the register.

WAIT -- Wait for One or More Events

The WAIT macro instruction relinquishes control of the CPU when the problem program has no further processing to do and must wait for the completion of one or more input/output operations. For a complete description of the WAIT macro instruction, see OS/VS Supervisor Services and Macro Instructions.

Name	Operation	Operand
[symbol]	WAIT	[number of events,] {ECB=address {ECBLIST=address}

number of events
: maximum is 255. Zero is an effective NOP instruction; one is assumed if the operand is omitted. The number of events must not exceed the number of event control blocks.

ECB=
: is the address of the event control block representing the single event that must occur before processing can continue. Valid only if the number of events is one or is omitted.

ECBLIST=
: is the address of a main storage area containing one or more consecutive fullwords on a fullword boundary. Each fullword contains the address of an event control block; the high-order bit in the last word must be set to one to indicate the end of the list. The number of event control blocks must be equal to or more than the specified number of events.

CLOSE -- Close Data Control Block Macro Instruction

The CLOSE macro instruction disassociates one or more data control blocks from the application program, thereby breaking the communication path between the application program and the display station. CLOSE performs the following functions:

- Indicates that the DCB for the display station is "closed" by setting the appropriate bit in the DCB.

- Releases main storage acquired at open time, including the chain of IOBs and associated channel programs, as well as storage acquired for attention handling functions.

- Clears the fields in the DCB initialized at open time.

- Deletes attention routines loaded by the "open" functions. (Basic attention handling only)

- Removes entries from the DCB address list (poll list) constructed at open time. (Express attention handling only)

The format of the CLOSE macro instruction is:

Name	Operation	Operand
[symbol]	CLOSE	(dcb address [,,dcb address]...) [,MF={L|(E,parameter list name)|(E,(1))}]

dcb address
: specifies the addresses of the DCBs for the graphic device(s) being closed. Any number of DCB addresses may be specified. If more than one DCB address is specified, the "dcb address" operands must be separated by two commas. The second comma indicates omission of an optional operand not applicable to graphic programming services.

MF=L
: specifies that expansion of the macro instruction is to consist of a parameter list only. The name assigned to the parameter list is the name specified in the Name field of this CLOSE macro instruction.

MF=(E,parameter list name)
: specifies that the macro instruction is to perform the close functions using a parameter list constructed by another CLOSE macro instruction. The close function is executed for each "dcb address" operand in the list. "Parameter list name" must correspond exactly to the name specified in the Name field of the CLOSE macro instruction used to construct the parameter list.

MF=(E,(1))
: performs the same function as (E, parameter list name). By writing the special register notation (1), the programmer signifies that he will load the address of the parameter list into register 1 before macro instruction execution.

PROGRAMMING NOTES: If a task attempts to close a DCB that was not opened within the same task, the task is abnormally terminated.

If the "dcb address" operand specifies a DCB that was already closed, no action results. If the operand does not specify the address of a DCB, the task is abnormally terminated.

MODIFYING THE DECB DURING EXECUTION

Required parameters in the DECB may be modified during execution by using an E-form macro instruction. The following programming considerations apply:

1. Register notation may be specified. The register is loaded with the appropriate parameter before execution of the macro instruction.

2. In using register notation, registers 2 through 12 may be used for operands other than the "decb name" operand. For this operand, register 1 must be used.

3. If the "type" operand is changed, it should be specified only by using an E-form macro instruction containing the new "type" code. Register notation is not allowed for the "type" operand.

Upon completion of an input/output operation, the completion flag in the event control block is set to one. If an ECB is used for more than one input/output operation, it is the application programmer's responsibility to set the completion flag to zero before execution of the second input/output operation and all subsequent input/output operations that refer to that ECB. The following illustration shows the position of the completion flag in the ECB.

```
    ┌───┬───┬───┬─────────────────────────┐
ECB │   │ 1 │   │                         │
    └───┴───┴───┴─────────────────────────┘
bits  0   1   2                          31
```

Assume, for example, that a data event control block named DECB1 is to be used for two read operations. The second GREAD macro instruction can be preceded by the following line of code to clear the completion flag in the ECB.

```
     NI    DECB1,X'BF'
```

OS/VS2 CONSIDERATIONS

The following rules apply to the use of input/output macro instructions in a VS2 environment, where basic attention handling facilities are used.

- A graphic device that has been opened by one task must be closed by the same task. Any task attempting to open or close a device that has been opened by another task, and not subsequently closed, will be abnormally terminated. (Note that this applies to the graphic device, not the DCB.)

- Within a single task, a graphic device may be opened more than once in succession (that is, without being closed after each open). A different DCB must be used each time, however.

- A task may issue input/output macro instructions (GREAD, GWRITE, GCNTRL) for a DCB that has been opened by a subtask or by an originating task. However, if the task that opened the DCB closes the DCB before the input/output function is completed, the task issuing the input/output macro instruction may be abnormally terminated. Synchronization of open, close, and input/output functions is the application programmer's responsibility.

The operator at the 2260 display station can present information to the system through a keyboard. Input from the keyboard results in an attention (an unscheduled input/output interruption). In order for such input to be meaningful, there must exist a routine or routines to be entered when an attention occurs.

Under basic attention handling, the Input/Output Supervisor automatically detects the occurrence of an attention and routes control through control routines to the correct application program routine. The application programmer must supply information specifying which routine is available to handle an attention from a particular display station. The macro instructions described in this section provide the necessary communication between the application program and the Input/Output Supervisor. These macro instructions give the programmer a high degree of control over the attention handling mechanism. The macro instructions and their functions are listed in Figure 7.

FUNCTIONS OF BASIC ATTENTION HANDLING

Basic attention handling provides the facilities to:

- Detect the occurrence of attentions automatically.
- Route these attentions to the application program attention handling routines.
- Queue succeeding attentions, if necessary, until the specified routine is ready to process them.
- Wait for expected attentions, optionally continuing selective processing in the meantime.

Particular advantages of basic attention handling are:

- Attentions are not lost if the routine designed to handle them is responding to an earlier attention at the time they are received.
- Processing may be overlapped with attention input or operator response time or both.
- System overhead is reduced, since there is no need to continually check for attentions.

BACKGROUND INFORMATION

The depression of the ENTER key on a 2260 keyboard causes the control program to interrupt the currently processing routine (which may be an attention routine) and to determine which, if any, routine has been specified to handle attentions from that device. If an attention routine has not been specified, the attention data is lost and control returns to the application program. If an attention routine is available, the attention data is placed in the programmer-specified area (COMAREA), and the attention routine is given control. To be available for entry upon receipt of a 2260 attention, a routine must have been designated by a SAEC macro instruction and the associated GACB must have been specified by a SPAR macro instruction and not subsequently deleted by a DAR macro instruction. If the routine specified to

Mnemonic	Macro Instruction Name	Function
SAEC	Specify Asynchronous Entry Conditions	Defines an attention handling routine and specifies the exact conditions that will cause entry to the routine.
SPAR	Specify Attention Routine	Makes attention handling routines available for system use.
DAR	Delete Attention Routine	Makes attention handling routines unavailable for system use.
ATTNINQ	Attention Inquiry	Allows communication between an attention handling routine and the control program.

Figure 7. Basic attention handling macro instructions

process the attention is the routine in control, the attention data is held until the current processing is completed.

Note: Issuing an attention from a 2260 causes the keyboard on that 2260 to become inoperative until the attention is processed either by an attention routine or by an ATTNINQ macro instruction. A GREAD or a GWRITE (not DATA ENTRY) macro instruction must be used to make the keyboard operative.

BASIC ATTENTION HANDLING MACRO INSTRUCTIONS

Macro instructions in this group allow the application programmer to define his program's attention handling capability to the system. The programmer can designate one attention handling routine to service attentions from all the graphic devices engaged in a task, or he can provide separate routines for individual devices. The SAEC, SPAR, and DAR macro instructions are used to establish the status of his attention handling routines. The ATTNINQ macro instruction is used to select an attention from a specified display station.

SAEC -- Specify Asynchronous Entry Conditions Macro Instruction

The SAEC macro instruction defines an attention handling routine by creating a parameter list called a graphic attention control block (GACB). The GACB contains information that identifies the attention handling routine, relates it to a 2260, specifies the entry point of the routine, and specifies the type or types of attentions that are to cause entry to the routine. The GACB can be referred to subsequently by the symbolic name of the SAEC macro instruction. The format of the SAEC macro instruction is:

<table>
<tr><th>Name</th><th>Operation</th><th>Operand</th></tr>
<tr><td>[symbol]</td><td>SAEC</td><td>[EP={entry point address|0}]
[,DCB=dcb address]
[,COMAREA=communication area address]
[,ATTNTYP={({A|S|R},2260)|SAVE|RESTORE}]
[,MF={L|(E, gacb address)}]</td></tr>
</table>

EP=

specifies the entry point of an attention handling routine. If this operand is omitted or specified equal to zero (these are equivalent), any attention for the routine will be held but not processed, until such time as an entry point is supplied or until the attention is examined by the ATTNINQ macro instruction. Subsequent attentions for this routine received from other 2260 stations are held until the first attention is processed.

Note: The 2848 display control locks the keyboard on the 2260 until the pending attention from that device is processed.

DCB=

specifies the address of a DCB associated with the display station, or group of display stations, from which attentions are to be serviced.

COMAREA=

specifies the address of a programmer-defined area in main storage, aligned on a fullword boundary and at least 16 bytes long, which is to be used by the control program to pass attention information to the attention routine.

ATTNTYP=

specifies the type of attention that will cause entry to the attention routine.

A, S, or R

specify, respectively, whether the information that follows is to be added to or subtracted from or is to replace the existing ATTNTYP field in the GACB. If this operand is omitted from an E-form of the macro instruction, R is assumed.

2260

specifies that the routine is to service attentions originating from a 2260.

SAVE, RESTORE

specify, respectively, that the contents of the ATTNTYP field of the GACB are to be stored or are to be replaced with the mask saved by a previous SAEC macro instruction with an ATTNTYP=SAVE operand.

Note: The codes A, S, R, SAVE, and RESTORE are provided for compatibility with other graphic devices, which may allow for more than one type of attention.

MF=L

specifies that the macro instruction is to construct a GACB utilizing the specified operands. Operands omitted from the L-form cause the associated fields to be set to zeros. If neither the L- nor E-form is specified, L is assumed. A, S, R, SAVE, and RESTORE are not relevant when the L-form of this macro instruction is used. Omission of the A, S, or R operand need not be indicated with a comma.

MF=E

causes the GACB established by an L-form of this macro instruction to be modified, as indicated by the other operands. If an operand is omitted, the corresponding field of the GACB is not affected. A comma must show the omission of the A, S, or R operand.

PROGRAMMING NOTES: The values specified for the COMAREA, DCB, ATTNTYP, and EP operands of the SAEC macro instruction are inserted in the first, second, fourth, and fifth words of the GACB, respectively. All should have been defined and the DCB must have been opened before a SPAR macro instruction referring to this GACB is issued.

The GACB, created by the SAEC macro instruction, is 14 words long. The first 5 words of the GACB have the following format:

GACB	+0	COMAREA address
	+4	DCB address
	+8	Reserved
	+12	ATTNTYP address
	+16	EP address

By setting the EP field of the GACB equal to zero, the programmer may queue attention data and process it in a background routine (as opposed to an attention handling routine). All attention data for GACBs with zero EP fields is placed on the same queue. The attention data may be accessed when desired by means of the ATTNINQ macro instruction.

The EP field may be set to zero at assembly time (by means of the SAEC macro instruction) or at any time during execution. If it is set to zero at execution time, any attentions received prior to setting EP equal to zero will be processed as described under "Background Information"; any received after EP has been set to zero will be queued.

The first four words of the COMAREA have the following format:

COMAREA	+0	IX	RESERVED	TYPE
	+4	RESERVED		
	+8	RESERVED		
	+12	RESERVED		
		PROGRAMMER-DEFINED AREA		

Field Contents

RESERVED These fields are used by the control program.

IX If multiple 2260s are assigned to a single DCB, this byte is set to an integer from 1 through 25 to indicate which device originated the attention that the routine has currently available for processing. The value of IX corresponds to the position of the device in the device list supplied by the DD card and placed in the DEB by an OPEN macro instruction. (IX functions in the same manner as the "unit" operand of the GREAD, GWRITE, and GCNTRL macro instructions.) If one 2260 is assigned to a single DCB, the byte is set to zero (hexadecimal 00).

TYPE This is always a hexadecimal 07, signifying a 2260 attention.

The user may add other fields to the COMAREA; see the suggestion under "Application Program Attention Handling Routine."

If an attention routine is to serve more than one device, and these devices have been specified in separate SAEC macro instructions, its executable code must be reenterable.

The DCB address in a GACB should not be changed without first deleting the attention routine with a DAR macro instruction. After changing the DCB address, the routine must be respecified with a SPAR macro instruction.

EXAMPLES:

```
GACBX1   SAEC   EP=AR1,DCB=GRAPHD1,
                ATTNTYP=2260,
                COMAREA=AREA1
```

The GACB established by this macro instruction can be referred to by the symbol GACBX1. The GACB defines an attention routine whose entry point is AR1. This routine is able to service attentions from the 2260 display station or stations associated with the identified DCB, and it will receive attention information in the COMAREA identified by the symbol AREA1. Because the macro-form (MF) is not specified, L-form is assumed.

```
EX2      SAEC   DCB=GRPHDCB2,EP=(3),
                MF=(E,GACBX1)
```

This macro instruction changes the DCB address and the attention routine entry point in the GACB defined in the first example.

Note: If this routine has been previously specified (with a SPAR macro instruction), it must be deleted (with a DAR macro instruction) before changing the DCB address. After issuing the SAEC macro instruction to change the DCB address, the attention routine must then be respecified before it is available to the system.

SPAR -- Specify Attention Routine Macro Instruction

The SPAR macro instruction makes previously defined attention handling routines available for system use by establishing control program references to their GACBs. It also assigns priority if multiple attention routines are defined. The format of the SPAR macro instruction is:

Name	Operation	Operand
[symbol]	SPAR	(gacb address [,gacb address]...) [,PRTY=integer] [,MF={L|(E,gacb list address)}]

gacb address
specifies the address of a graphic attention control block (GACB), defined by a SAEC macro instruction, that is to be made available to the system.

PRTY=
is a decimal integer from 0 through 127, used to determine priority among attention routines. Priority increases with numeric value (127 gives highest priority). No attention handling routine may have control while a higher-priority routine for the same device has any processing to do.

MF=L
specifies the construction of a list of GACB addresses that can be used by an E-form of either a SPAR or DAR macro instruction.

MF=E
specifies that the list of GACBs referred to by the "gacb list address" operand is to be modified as indicated by the "gacb address" operands and that the GACBs in the list are then to be specified, that is, made available for system use. The list cannot be lengthened or shortened by the E-form of this macro instruction.

gacb list address
specifies the address of a list of GACBs established by an L-form SPAR macro instruction.

PROGRAMMING NOTES: The PRTY operand is provided primarily for compatibility with certain other graphic devices that are capable of originating more than one type of attention. For such devices, the PRTY operand permits a routine designed to handle attentions of a particular type to interrupt another, less important, attention routine that handles attentions of another type from the same device. The programmer establishes the relative "importance" of routines for this purpose by use of the PRTY operand. This usage is not possible on the 2260, which can originate only a single type of attention. Therefore, the PRTY operand can normally be omitted.

When the PRTY operand is omitted, a priority of zero is assigned.

If more than one attention routine is available for any one 2260 at any time, and if these routines have been assigned the same numeric priority (for example, by omission of the PRTY operand), the routine referred to last in the last SPAR macro instruction has the highest priority.

All attention handling routines take precedence over any background routine.

An attention routine may be interrupted and immediately reentered if the same routine is defined in separate SPAR macro instructions for two or more display stations or if the routine is used by more than one task. Such a routine must be reenterable (that is, it must not modify itself in any way).

If the same GACB is referred to more than once in a SPAR macro instruction, references after the first are treated as no-operations. A GACB address of zero, in a list of GACB addresses, will also be treated as a no-operation.

ERROR CONDITIONS: Certain error conditions will stop the processing of a list of GACBs and cause an immediate return. Upon return to the program, register 1 will contain the address of the GACB associated with the error condition, and register 15 will contain one of the following return codes to indicate the type of error:

Return Code	Meaning
04	DCB is not a graphics DCB
08	GACB was already specified through a SPAR macro instruction

0C	Parameter list is too short to contain GACB addresses
10	DCB is not open
14	(OS/VS2 only) SPAR macro instruction is not issued by the task that opened the DCB

Write-to-programmer messages are issued for the error conditions. For message explanations and programmer error checks, see OS/VS Problem Determination Aids and Messages and Codes for GPS and GSP.

EXAMPLES:

EX1 SPAR (GACBX2)

This macro instruction establishes control references to the routine defined in a GACB identified by GACBX2.

EX2 SPAR (GACBX1,NAME3,TAG6),MF=L

This macro instruction constructs a list containing the three GACB addresses specified.

EX3 SPAR (,,GACBX3),MF=(E,EX2)

This macro instruction causes the parameter list constructed by EX2 to be modified, replacing the third GACB address (TAG6) with GACBX3. Control references are then established for the three GACBs referred to by the list.

DAR -- Delete Attention Routine Macro Instruction

The DAR macro instruction is the inverse of the SPAR macro instruction. It makes attention handling routines unavailable for system use by deleting the control program references to the GACBs established by SPAR. The format of the DAR macro instruction is:

Name	Operation	Operand
[symbol]	DAR	(gacb address [,gacb address]...) [,MF={L|(E, gacb list address)}]

gacb address
specifies the address of a GACB defining the attention handling routine to be deleted.

MF=L
specifies formation of a list of the GACB addresses written in the macro instruction.

MF=E
specifies that the list of GACBs referred to by the "gacb list address" operand is to be modified by the "gacb address" operands and that all attention routines referred to in the list are then to be deleted, that is, made unavailable for system use.

gacb list address
specifies the address of a list of GACB addresses constructed by an L-form SPAR or DAR macro instruction.

PROGRAMMING NOTES: The CLOSE function will delete all attention routines defined for the associated DCB. However, an attempt to issue a CLOSE macro instruction from an attention routine will result in abnormal termination of the task. Similarly, an attempt to delete an active attention routine will result in an error condition.

ERROR CONDITIONS: Certain error conditions will stop the processing of a list of GACBs (or of one GACB) and cause an immediate return. Upon return to the program, register 1 will contain the address of the GACB associated with the error condition, and register 15 will contain a return code to identify the error condition as follows:

Return Code	Meaning
04	DCB address is not in the GACB
08	GACB was not specified through a SPAR macro instruction
0C	Parameter list is too short to contain GACB addresses
10	Specified attention handling routine is still active
14	(OS/VS2 only) DAR macro instruction is not issued by the task that issued the SPAR macro instruction
18	DCB is not open
1C	DCB is not a graphics DCB
20	Attention handling environment is not basic

Write-to-programmer messages are issued for the error conditions. For message explanations and programmer error checks, see OS/VS Problem Determination Aids and Messages and Codes for GPS and GSP.

ATTNINQ -- Attention Inquiry Macro Instruction

The ATTNINQ macro instruction establishes communication between an attention handling routine and the control program to permit the routine to select an attention from a specified display station or from any display station. This allows the attention handling routine to wait until the attention occurs, to relinquish control to the mainline program until the attention occurs, to branch to a specified address if specified attention information is pending, or to clear all attentions pending for this routine. In the case of CLEAR, the attentions are released, but the 2260 display station must be manually unlatched. The format of the ATTNINQ macro instruction is:

<table>
<tr><th>Name</th><th>Operation</th><th>Operand</th></tr>
<tr><td>[symbol]</td><td>ATTNINQ</td><td>gacb address,
MODE={R|W|(C,branch address)|CLEAR}
[,UNIT={integer|ALL}]
[,MF={L|(E,parameter list address)}]</td></tr>
</table>

gacb address
: specifies the address of the associated GACB.

MODE=
: specifies one of the four modes of inquiry, as follows:

R
: relinquish mode. The attention handling routine is to relinquish control until attention information from the display station the routine is servicing (specified by the UNIT operand) is available for that routine. When this information is available, processing resumes with the next sequential instruction. This mode is treated by the system as a W (wait) mode if the ATTNINQ macro instruction is issued from a routine entered by means of a LINK macro instruction, or if the EP field of an associated GACB has been set to zero.

W
: wait mode. The attention handling routine is to be placed in a wait state until attention information is available from the display station specified by the UNIT operand. While the attention routine is in a wait state, attentions from other display stations can be serviced by other attention routines or another task can be in control. When the wait condition is satisfied, and control is returned to the attention routine, processing continues with the next sequential instruction.

C,branch address
: condition mode. If attention information from the device specified by the UNIT operand is presently pending for processing by this attention handling routine, a branch is made to the location specified by the "branch address" operand. Otherwise, processing continues with the next sequential instruction.

CLEAR
: clear mode. All attentions pending for this attention handling routine are released, and processing continues with the next sequential instruction. The keyboard on any 2260 display station must be manually unlatched if an attention issued by it has been cleared by an ATTNINQ macro instruction.

UNIT=integer
: specifies a decimal integer from 1 through 25 and corresponds to the position of the device in the device list supplied by a DD statement, where multiple devices are assigned to a single DCB. It is used as an index to select one of the several UCB addresses appended to the data extent block (DEB). The index value corresponding to the 1053 printer may not be specified. (This operand functions in the same manner as the "unit" operand of the GREAD and GWRITE macro instructions.)

UNIT=ALL
: specifies that an attention will be accepted from any 2260 assigned to the DCB.

If the UNIT operand is omitted, ALL is assumed.

MF=L
: specifies that the expansion of the macro instruction is to consist of a parameter list only. The symbol written in the Name field is assigned to the parameter list.

MF=E
: causes the parameter list, established by the L-form of the macro instruction and identified by the parameter list address operand, to be modified by the preceding operands and causes the attention inquiry request to be executed.

parameter list address
: specifies the address of a parameter list established by an L-form ATTNINQ macro instruction.

PROGRAMMING NOTES: The ATTNINQ macro instruction may be issued within an attention routine or, if the EP field of the associated GACB is set to zero, may be issued from a background routine. Standard register conventions must be observed.

The UNIT operand must be used in conjunction with the W or C mode if there are multiple devices for the associated DCB.

It is meaningless in conjunction with the CLEAR or R mode.

Under R, W, or C mode, information pertaining to the attention that satisfies the inquiry is made available in the COMAREA.

ERROR CONDITIONS: During execution, certain error conditions will terminate processing of the ATTNINQ macro instruction and cause an immediate return to the program. Upon return, register 15 contains a return code to identify the error condition as follows:

Return Code	Meaning
10	Parameter list contains no GACB address
14	EP2 field of the GACB is not zero
18	GACB was not specified through a SPAR macro instruction
1C	(OS/VS2 only) ATTNINQ macro instruction is not issued by the task that issued the SPAR macro instruction
20	Mode field in the parameter list is invalid
24	Internal dequeue error occurred
28	Specified attention handling routine is already in a wait state
2C	Branch address is not in the parameter list

Write-to-programmer messages are issued for the error conditions. For message explanations and programmer error checks, see OS/VS Problem Determination Aids and Messages and Codes for GPS and GSP.

EXAMPLES:

```
EX1  ATTNINQ  GACBX4,MODE=R
```

This macro instruction causes control by the attention routine to be relinquished until the desired attention occurs.

```
EX2  ATTNINQ  GACBX4,MODE=W,UNIT=5
```

This macro instruction causes the task to enter a WAIT state until attention information from the display station which occupies the fifth position on the device list is available.

```
EX3  ATTNINQ  GACBX4,MODE=(C,BRANCHA),
              UNIT=5
```

This macro instruction causes a branch to be taken to location BRANCHA if an attention from the display station which occupies the fifth position on the devices list is pending.

```
EX4  ATTNINQ  GACBX4,MODE=CLEAR
```

This macro instruction causes all attentions pending for this routine to be released.

APPLICATION PROGRAM ATTENTION HANDLING ROUTINE

The following notes may be helpful to an application programmer writing a program designed for operator-machine communication.

Establishing the Attention Routine

An attention handling routine is defined by means of a SAEC macro instruction, which establishes a graphic attention control block (GACB) containing the information provided in the SAEC macro instruction. In the GACB are:

1. The address of the entry point of the associated attention routine.
2. The address of a DCB associated with the device to be serviced.
3. The address of a communication area (COMAREA) to be used to pass attention information to the attention routine.

The GACB thus provides a link between a display station or group of display stations and an attention routine.

Although defined, an attention routine is not available for use until it has been "specified" by means of a SPAR macro instruction. This establishes control program references to it.

The DAR macro instruction can be used to disable an attention routine that is no longer needed, although the associated DCB remains open. The SPAR and DAR macro instructions may be thought of as analogous to OPEN and CLOSE.

Processing the Attention

Upon entry to an attention handling routine, register 1 contains the address of the GACB. In the first word of the GACB is the address of the communication area, or COMAREA. If more than one device is assigned to the DCB, the first byte of the COMAREA indicates the device that originated the attention.

Use of the communication area allows the programmer to pass information between the housekeeping/background routine and the attention routine or routines. Each display station may be associated with its own communication area, and the attention data for each display station may be recorded independently of that for other display units. One of the fields in the communication area can be an ECB where completion of attention processing can be posted. The attention routine can post the appropriate ECB when it completes attention processing for that display station.

Using the information available in the COMAREA and the DCB, the attention handling routine can perform necessary calculations, issue appropriate input/output commands, and do whatever else is required to respond to the attention. If further communication from the display station operator is required, an ATTNINQ macro instruction can be used to relinquish control or enter a wait state until the desired attention occurs, or to set up a conditional branch based on its availability.

When processing of the last attention of a communication sequence has been completed, attentions (for this routine) can be cleared by means of an ATTNINQ, mode CLEAR, macro instruction before passing control to the interrupted routine by means of the RETURN macro instruction.

The following is a brief outline of functions that might be performed by a housekeeping/background routine and by an attention handling routine.

Housekeeping/Background Routine

1. Open the DCB for the display station or stations.
2. Establish the attention routine by means of the SAEC and SPAR macro instructions.
3. Issue input/output commands.
4. Make sure that input/output operations were started successfully.
5. Wait for posting of completion in the ECB or perform background processing, which may or may not be related to the attention routine.
6. Close the DCB.
7. Indicate task completion by issuing the RETURN macro instruction.

Attention Routine

1. Perform standard entry functions: save registers, establish addressability, etc.
2. Perform operations based on the attention information.
3. Issue input/output commands.
4. Make sure that input/output operations were started successfully.
5. Wait for the next attention by means of the ATTNINQ macro instruction.
6. After the last attention is serviced, post completion in the ECB and return control to the background routine by means of the POST and RETURN system macro instructions.

Synchronizing the Application Program Routines

Synchronization of the housekeeping/ background routine and the attention handling routine is the application programmer's responsibility. He may use the WAIT system macro instruction to delay execution of the housekeeping/background routine until completion has been posted in an ECB by the application program attention routine. The attention routine may either explicitly post the ECB by means of the POST system macro instruction or, by means of an application program convention, locate the appropriate ECB by means of the graphic attention control block and then post completion. The attention routine passes control back to the housekeeping/background routine by means of the RETURN macro instruction. Since completion has been posted, the housekeeping routine is taken out of its wait state, and execution continues. This cycle, the passing of control to the attention routine when an attention occurs and the returning of control to the interrupted (housekeeping/background) routine when completion is posted, continues as long as the attention routine is defined and as long as attentions occur.

OS/VS2 CONSIDERATIONS

The following rules apply to the use of basic attention handling facilities in a VS2 environment.

In order for the application program attention handling routines to be available to a device, they must be defined (by means of an SAEC macro instruction) and specified (by a SPAR macro instruction) within the task that opens that device or within one of its subtasks. A SPAR macro instruction issued from an inappropriate task will return with a return code of hexadecimal 14.

Each GACB must be deleted (by means of DAR) by the task in which it was specified (by SPAR). A DAR macro instruction from another task will cause a return with a return code of hexadecimal 14. An attempt by one task to close a DCB for which one or more GACBs are still specified by another task will result in the abnormal termination of the specifying task.

An ATTNINQ macro instruction for any GACB may be issued only by the task that specified (by means of SPAR) that GACB; otherwise, the ATTNINQ macro instruction returns control with a return code of hexadecimal 1C.

An application program attention routine may serve more than one task. However, a separate GACB must be established for each task-device-routine combination. The application programmer is responsible for ensuring that the task containing the attention routine is not terminated while other tasks have active (currently specified) GACBs referring to that routine.

APPENDIX A: SAMPLE PROGRAM

Coding shown in Figure 8 is intended for use with two IBM 2260 Display Stations (Local Attachment), equipped with the alphameric keyboard feature. A 1053 printer is not needed.

For the purposes of this sample program, it is assumed that the first seven characters of the manual input message contain a code ('PROCESS') that determines whether the routine processes the message and stores the results or, in the absence of the code, ends the job and displays results derived from the processing of each individual request.

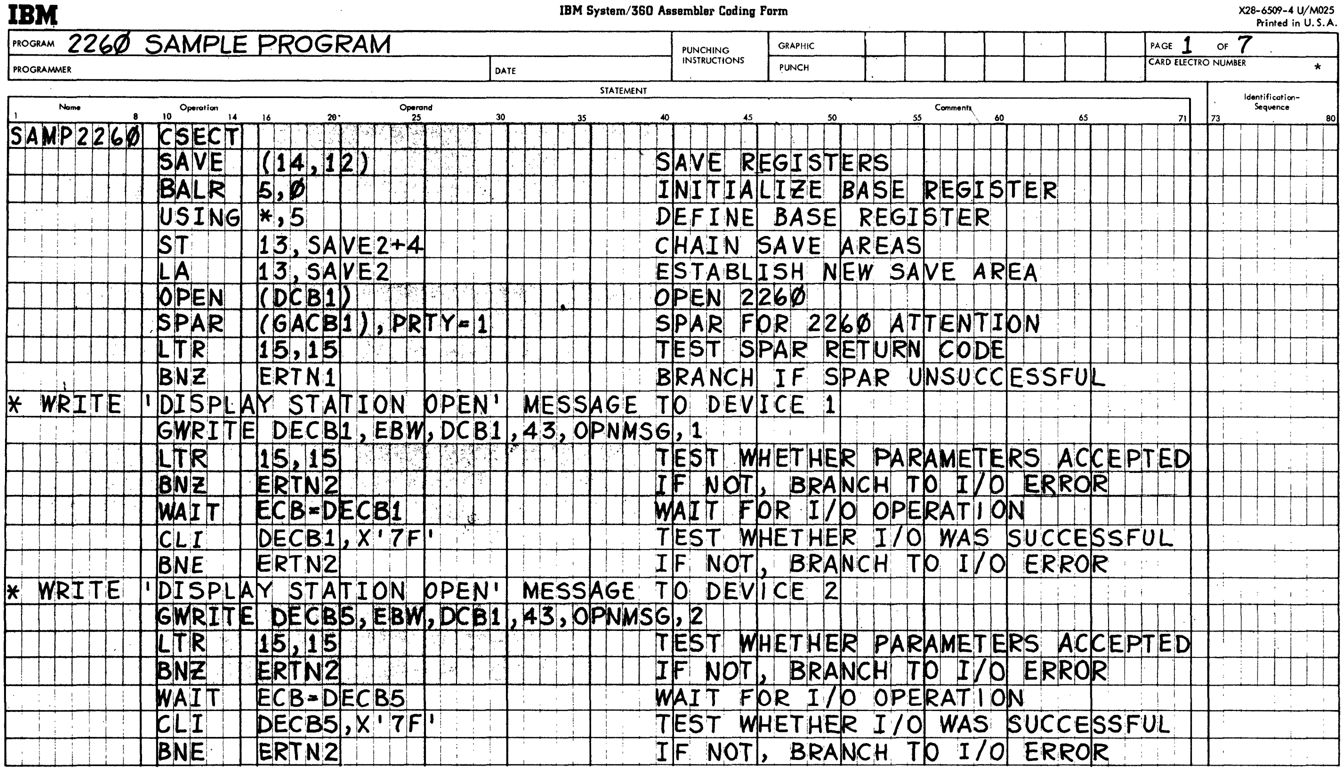
The purpose of each element of the program is explained by the comments in the illustration. The following paragraphs note several graphic programming considerations illustrated by the program.

Opening the DCB: The DCB for the 2260s must be opened before references are made to it.

Use of the GREAD Macro Instruction: The GREAD macro instruction shown in the coding generates a Read Display Station Manual Input (DSM) operation.

Use of the GWRITE Macro Instruction: The GWRITE macro instruction shown issues an Erase and Write Display Station Buffer Storage (EBW) operation to clear the storage and write a message to the operator.

Use of the Basic Attention Handling Facilities: All application program attention handling routines should consider themselves as subroutines, and as such, standard subroutine linkage should be observed. The COMAREA should be saved if there is any possibility that the attention routine presently in control may be interrupted during execution by another attention routine that shares the same COMAREA.



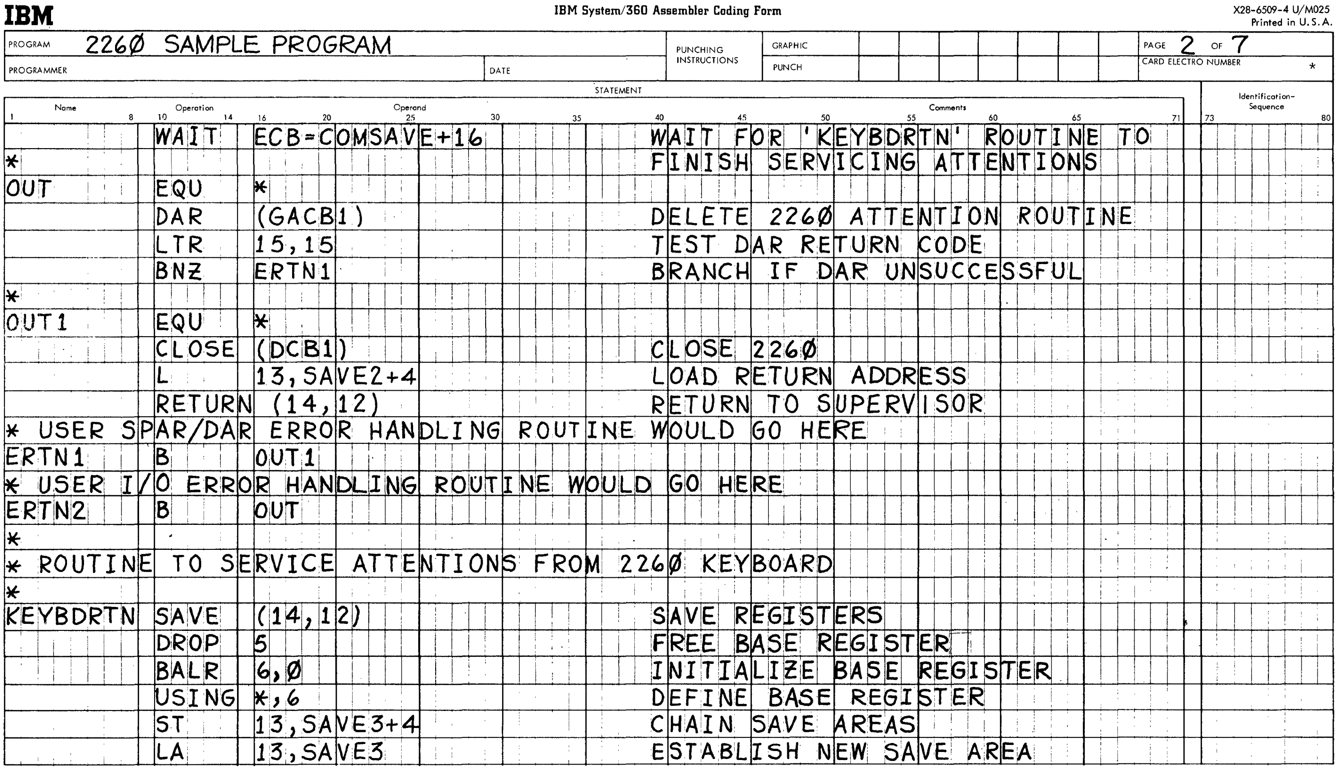
IBM — IBM System/360 Assembler Coding Form — X28-6509-4 U/M025 Printed in U.S.A.

PROGRAM 226Ø SAMPLE PROGRAM — PUNCHING INSTRUCTIONS — GRAPHIC — PUNCH — PAGE 1 OF 7

PROGRAMMER — DATE — CARD ELECTRO NUMBER *

```
SAMP226Ø CSECT
         SAVE  (14,12)                  SAVE REGISTERS
         BALR  5,Ø                      INITIALIZE BASE REGISTER
         USING *,5                      DEFINE BASE REGISTER
         ST    13,SAVE2+4               CHAIN SAVE AREAS
         LA    13,SAVE2                 ESTABLISH NEW SAVE AREA
         OPEN  (DCB1)                   OPEN 226Ø
         SPAR  (GACB1),PRTY=1           SPAR FOR 226Ø ATTENTION
         LTR   15,15                    TEST SPAR RETURN CODE
         BNZ   ERTN1                    BRANCH IF SPAR UNSUCCESSFUL
* WRITE 'DISPLAY STATION OPEN' MESSAGE TO DEVICE 1
         GWRITE DECB1,EBW,DCB1,43,OPNMSG,1
         LTR   15,15                    TEST WHETHER PARAMETERS ACCEPTED
         BNZ   ERTN2                    IF NOT, BRANCH TO I/O ERROR
         WAIT  ECB=DECB1                WAIT FOR I/O OPERATION
         CLI   DECB1,X'7F'              TEST WHETHER I/O WAS SUCCESSFUL
         BNE   ERTN2                    IF NOT, BRANCH TO I/O ERROR
* WRITE 'DISPLAY STATION OPEN' MESSAGE TO DEVICE 2
         GWRITE DECB5,EBW,DCB1,43,OPNMSG,2
         LTR   15,15                    TEST WHETHER PARAMETERS ACCEPTED
         BNZ   ERTN2                    IF NOT, BRANCH TO I/O ERROR
         WAIT  ECB=DECB5                WAIT FOR I/O OPERATION
         CLI   DECB5,X'7F'              TEST WHETHER I/O WAS SUCCESSFUL
         BNE   ERTN2                    IF NOT, BRANCH TO I/O ERROR
```

Figure 8 (part 1 of 7). Coding for sample program

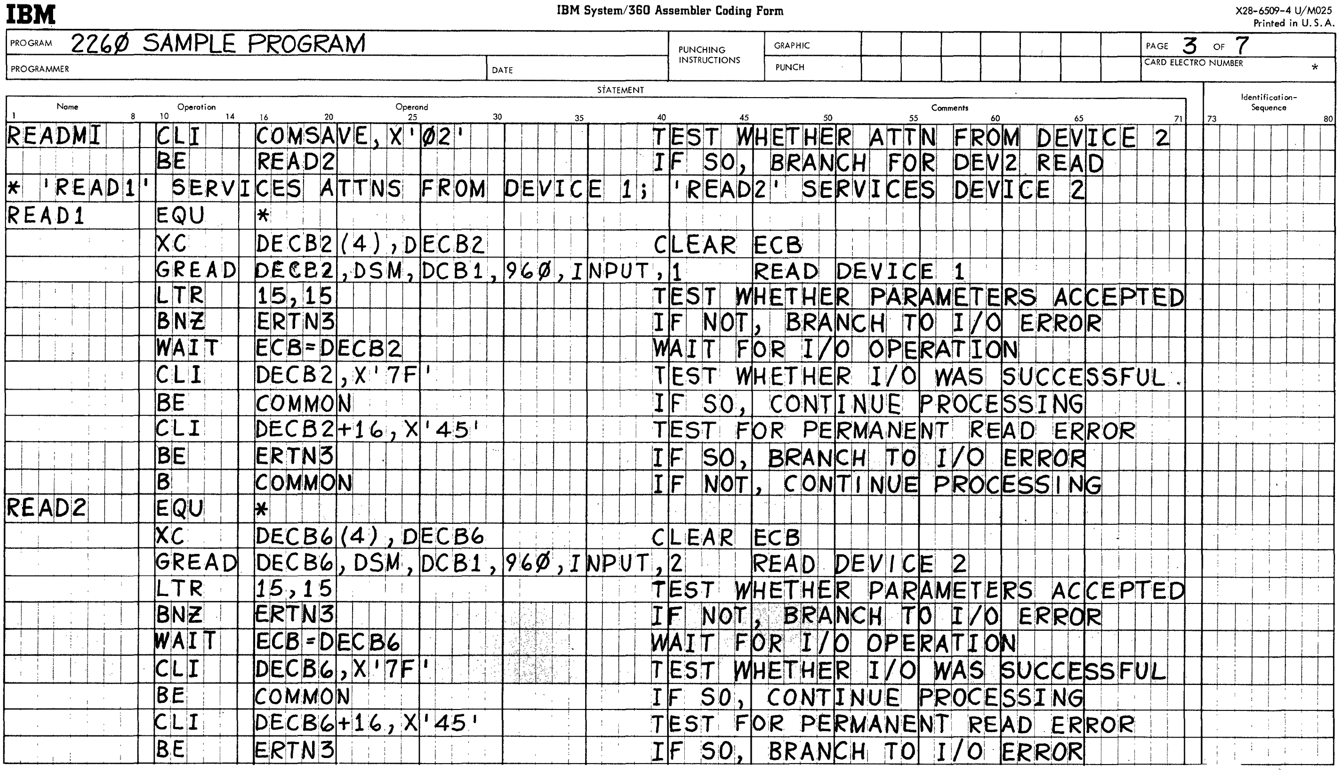


IBM — IBM System/360 Assembler Coding Form — X28-6509-4 U/M025 Printed in U.S.A.

PROGRAM 226Ø SAMPLE PROGRAM — PAGE 2 OF 7

```
         WAIT  ECB=COMSAVE+16                WAIT FOR 'KEYBDRTN' ROUTINE TO
*                                            FINISH SERVICING ATTENTIONS
OUT      EQU   *
         DAR   (GACB1)                       DELETE 226Ø ATTENTION ROUTINE
         LTR   15,15                         TEST DAR RETURN CODE
         BNZ   ERTN1                         BRANCH IF DAR UNSUCCESSFUL
*
OUT1     EQU   *
         CLOSE (DCB1)                        CLOSE 226Ø
         L     13,SAVE2+4                    LOAD RETURN ADDRESS
         RETURN (14,12)                      RETURN TO SUPERVISOR
* USER SPAR/DAR ERROR HANDLING ROUTINE WOULD GO HERE
ERTN1    B     OUT1
* USER I/O ERROR HANDLING ROUTINE WOULD GO HERE
ERTN2    B     OUT
*
* ROUTINE TO SERVICE ATTENTIONS FROM 226Ø KEYBOARD
*
KEYBDRTN SAVE  (14,12)                       SAVE REGISTERS
         DROP  5                             FREE BASE REGISTER
         BALR  6,Ø                           INITIALIZE BASE REGISTER
         USING *,6                           DEFINE BASE REGISTER
         ST    13,SAVE3+4                    CHAIN SAVE AREAS
         LA    13,SAVE3                      ESTABLISH NEW SAVE AREA
```

Figure 8 (part 2 of 7). Coding for sample program

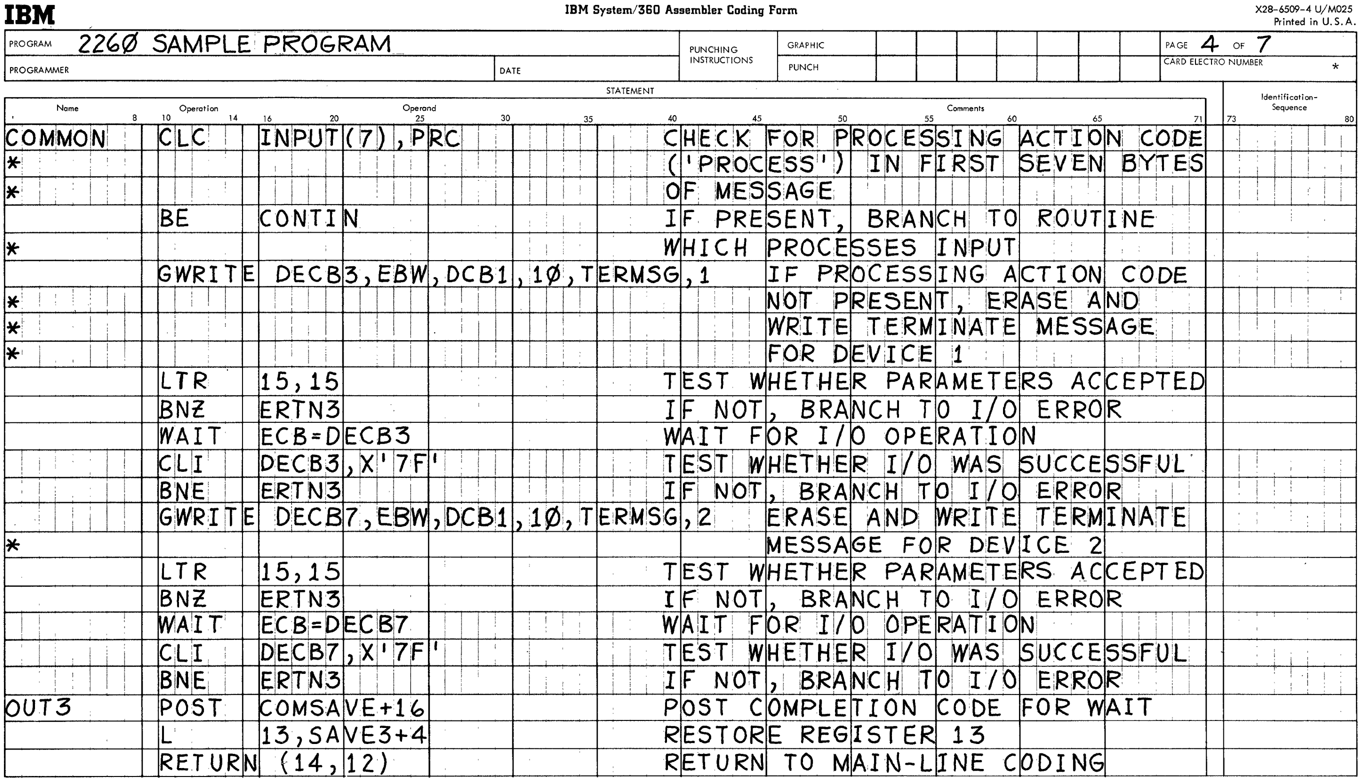


IBM — IBM System/360 Assembler Coding Form — X28-6509-4 U/M025 Printed in U.S.A.

PROGRAM 226Ø SAMPLE PROGRAM — PAGE 3 OF 7

```
READMI   CLI   COMSAVE,X'Ø2'                 TEST WHETHER ATTN FROM DEVICE 2
         BE    READ2                         IF SO, BRANCH FOR DEV2 READ
* 'READ1' SERVICES ATTNS FROM DEVICE 1; 'READ2' SERVICES DEVICE 2
READ1    EQU   *
         XC    DECB2(4),DECB2                CLEAR ECB
         GREAD DECB2,DSM,DCB1,96Ø,INPUT,1    READ DEVICE 1
         LTR   15,15                         TEST WHETHER PARAMETERS ACCEPTED
         BNZ   ERTN3                         IF NOT, BRANCH TO I/O ERROR
         WAIT  ECB=DECB2                     WAIT FOR I/O OPERATION
         CLI   DECB2,X'7F'                   TEST WHETHER I/O WAS SUCCESSFUL.
         BE    COMMON                        IF SO, CONTINUE PROCESSING
         CLI   DECB2+16,X'45'                TEST FOR PERMANENT READ ERROR
         BE    ERTN3                         IF SO, BRANCH TO I/O ERROR
         B     COMMON                        IF NOT, CONTINUE PROCESSING
READ2    EQU   *
         XC    DECB6(4),DECB6                CLEAR ECB
         GREAD DECB6,DSM,DCB1,96Ø,INPUT,2    READ DEVICE 2
         LTR   15,15                         TEST WHETHER PARAMETERS ACCEPTED
         BNZ   ERTN3                         IF NOT, BRANCH TO I/O ERROR
         WAIT  ECB=DECB6                     WAIT FOR I/O OPERATION
         CLI   DECB6,X'7F'                   TEST WHETHER I/O WAS SUCCESSFUL
         BE    COMMON                        IF SO, CONTINUE PROCESSING
         CLI   DECB6+16,X'45'                TEST FOR PERMANENT READ ERROR
         BE    ERTN3                         IF SO, BRANCH TO I/O ERROR
```

Figure 8 (part 3 of 7). Coding for sample program

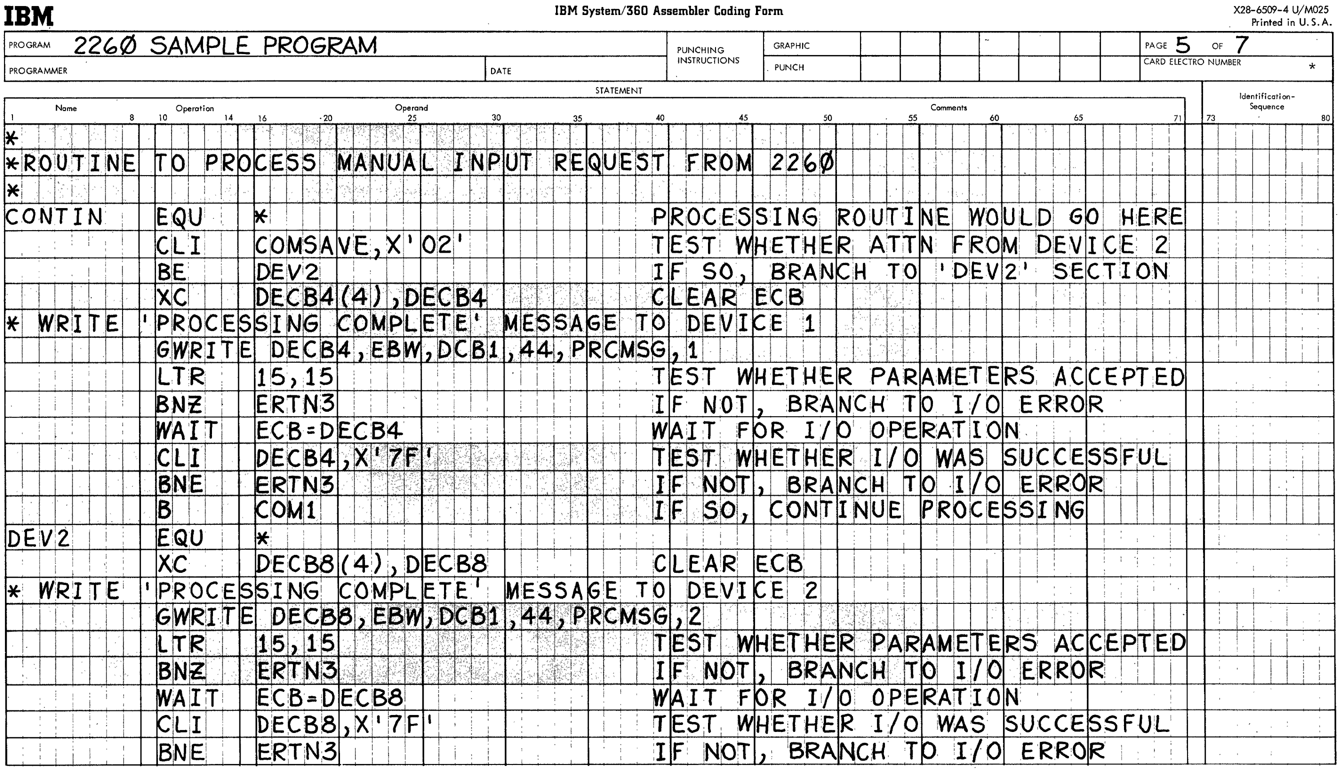


IBM — IBM System/360 Assembler Coding Form — X28-6509-4 U/M025 Printed in U.S.A.

PROGRAM 226Ø SAMPLE PROGRAM — PAGE 4 OF 7

```
COMMON   CLC   INPUT(7),PRC            CHECK FOR PROCESSING ACTION CODE
*                                      ('PROCESS') IN FIRST SEVEN BYTES
*                                      OF MESSAGE
         BE    CONTIN                  IF PRESENT, BRANCH TO ROUTINE
*                                      WHICH PROCESSES INPUT
         GWRITE DECB3,EBW,DCB1,1Ø,TERMSG,1 IF PROCESSING ACTION CODE
*                                           NOT PRESENT, ERASE AND
*                                           WRITE TERMINATE MESSAGE
*                                           FOR DEVICE 1
         LTR   15,15                   TEST WHETHER PARAMETERS ACCEPTED
         BNZ   ERTN3                   IF NOT, BRANCH TO I/O ERROR
         WAIT  ECB=DECB3               WAIT FOR I/O OPERATION
         CLI   DECB3,X'7F'             TEST WHETHER I/O WAS SUCCESSFUL
         BNE   ERTN3                   IF NOT, BRANCH TO I/O ERROR
         GWRITE DECB7,EBW,DCB1,1Ø,TERMSG,2 ERASE AND WRITE TERMINATE
*                                           MESSAGE FOR DEVICE 2
         LTR   15,15                   TEST WHETHER PARAMETERS ACCEPTED
         BNZ   ERTN3                   IF NOT, BRANCH TO I/O ERROR
         WAIT  ECB=DECB7               WAIT FOR I/O OPERATION
         CLI   DECB7,X'7F'             TEST WHETHER I/O WAS SUCCESSFUL
         BNE   ERTN3                   IF NOT, BRANCH TO I/O ERROR
OUT3     POST  COMSAVE+16              POST COMPLETION CODE FOR WAIT
         L     13,SAVE3+4              RESTORE REGISTER 13
         RETURN (14,12)                RETURN TO MAIN-LINE CODING
```

Figure 8 (part 4 of 7). Coding for sample program

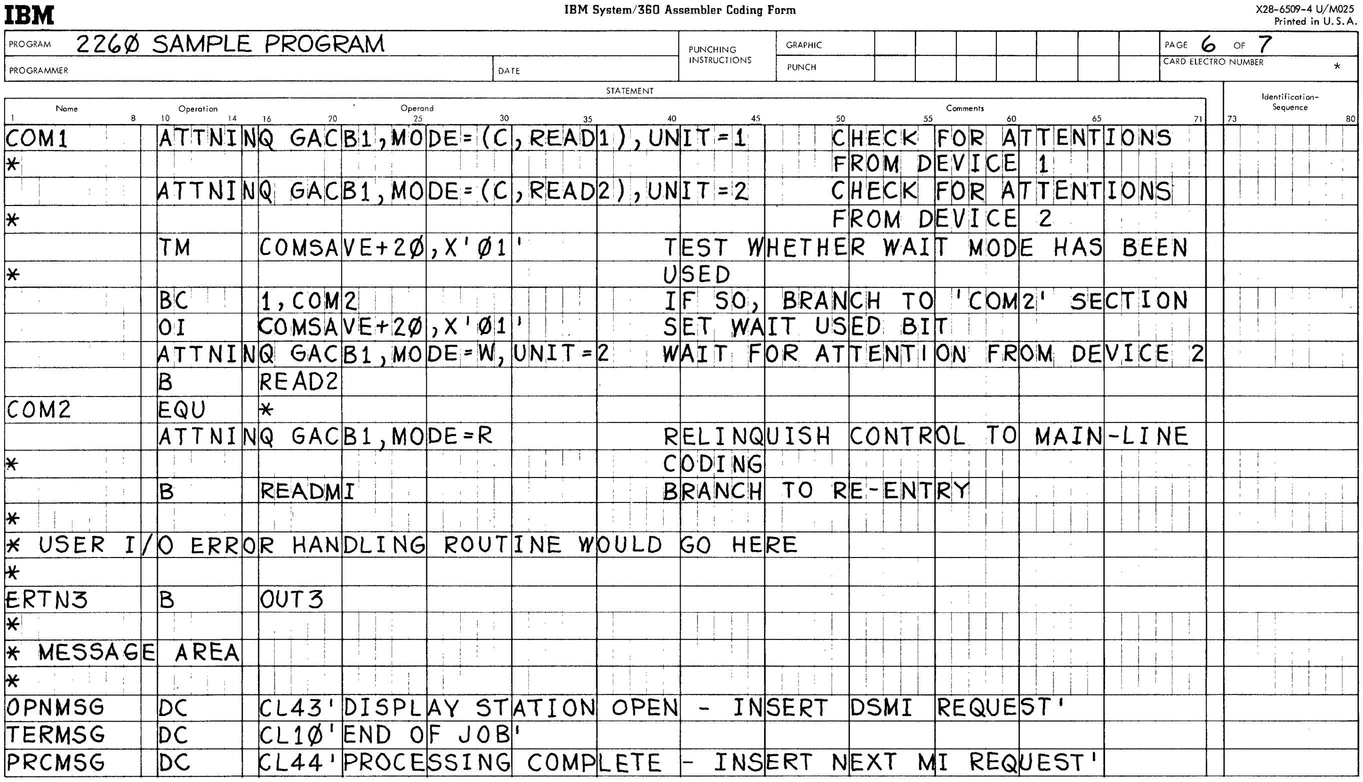


IBM — IBM System/360 Assembler Coding Form — X28-6509-4 U/M025 Printed in U.S.A.

PROGRAM 226Ø SAMPLE PROGRAM — PAGE 5 OF 7

```
*
*ROUTINE TO PROCESS MANUAL INPUT REQUEST FROM 226Ø
*
CONTIN   EQU   *                       PROCESSING ROUTINE WOULD GO HERE
         CLI   COMSAVE,X'02'           TEST WHETHER ATTN FROM DEVICE 2
         BE    DEV2                    IF SO, BRANCH TO 'DEV2' SECTION
         XC    DECB4(4),DECB4          CLEAR ECB
* WRITE 'PROCESSING COMPLETE' MESSAGE TO DEVICE 1
         GWRITE DECB4,EBW,DCB1,44,PRCMSG,1
         LTR   15,15                   TEST WHETHER PARAMETERS ACCEPTED
         BNZ   ERTN3                   IF NOT, BRANCH TO I/O ERROR
         WAIT  ECB=DECB4               WAIT FOR I/O OPERATION
         CLI   DECB4,X'7F'             TEST WHETHER I/O WAS SUCCESSFUL
         BNE   ERTN3                   IF NOT, BRANCH TO I/O ERROR
         B     COM1                    IF SO, CONTINUE PROCESSING
DEV2     EQU   *
         XC    DECB8(4),DECB8          CLEAR ECB
* WRITE 'PROCESSING COMPLETE' MESSAGE TO DEVICE 2
         GWRITE DECB8,EBW,DCB1,44,PRCMSG,2
         LTR   15,15                   TEST WHETHER PARAMETERS ACCEPTED
         BNZ   ERTN3                   IF NOT, BRANCH TO I/O ERROR
         WAIT  ECB=DECB8               WAIT FOR I/O OPERATION
         CLI   DECB8,X'7F'             TEST WHETHER I/O WAS SUCCESSFUL
         BNE   ERTN3                   IF NOT, BRANCH TO I/O ERROR
```

Figure 8 (part 5 of 7). Coding for sample program

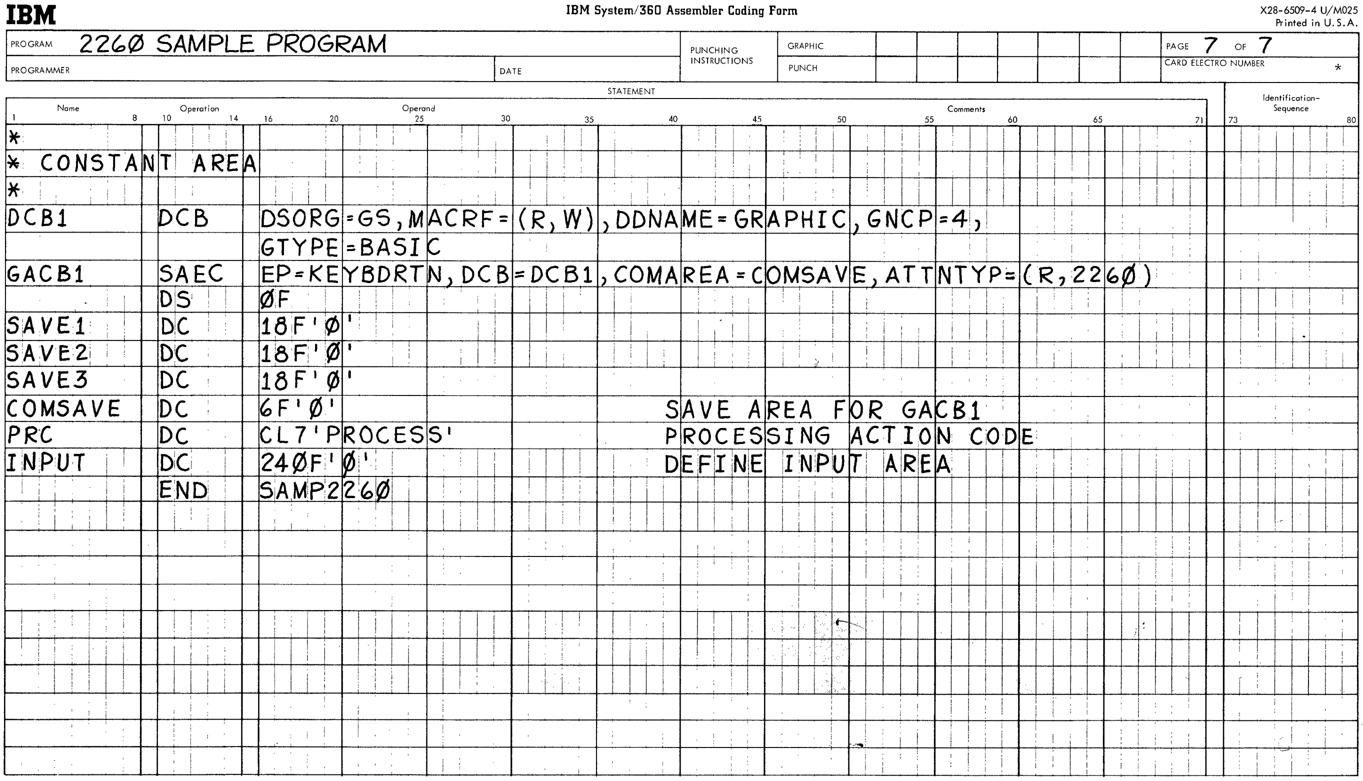


IBM — IBM System/360 Assembler Coding Form — X28-6509-4 U/M025 Printed in U.S.A.

PROGRAM 2260 SAMPLE PROGRAM — PAGE 6 OF 7 — CARD ELECTRO NUMBER *

```
COM1     ATTNINQ GACB1,MODE=(C,READ1),UNIT=1     CHECK FOR ATTENTIONS
*                                                FROM DEVICE 1
         ATTNINQ GACB1,MODE=(C,READ2),UNIT=2     CHECK FOR ATTENTIONS
*                                                FROM DEVICE 2
         TM    COMSAVE+20,X'01'        TEST WHETHER WAIT MODE HAS BEEN
*                                      USED
         BC    1,COM2                  IF SO, BRANCH TO 'COM2' SECTION
         OI    COMSAVE+20,X'01'        SET WAIT USED BIT
         ATTNINQ GACB1,MODE=W,UNIT=2   WAIT FOR ATTENTION FROM DEVICE 2
         B     READ2
COM2     EQU   *
         ATTNINQ GACB1,MODE=R          RELINQUISH CONTROL TO MAIN-LINE
*                                      CODING
         B     READMI                  BRANCH TO RE-ENTRY
*
* USER I/O ERROR HANDLING ROUTINE WOULD GO HERE
*
ERTN3    B     OUT3
*
* MESSAGE AREA
*
OPNMSG   DC    CL43'DISPLAY STATION OPEN - INSERT DSMI REQUEST'
TERMSG   DC    CL10'END OF JOB'
PRCMSG   DC    CL44'PROCESSING COMPLETE - INSERT NEXT MI REQUEST'
```

Figure 8 (part 6 of 7). Coding for sample program



IBM — IBM System/360 Assembler Coding Form — X28-6509-4 U/M025 Printed in U.S.A.

PROGRAM 2260 SAMPLE PROGRAM — PAGE 7 OF 7 — CARD ELECTRO NUMBER *

```
*
* CONSTANT AREA
*
DCB1     DCB   DSORG=GS,MACRF=(R,W),DDNAME=GRAPHIC,GNCP=4,
               GTYPE=BASIC
GACB1    SAEC  EP=KEYBDRTN,DCB=DCB1,COMAREA=COMSAVE,ATTNTYP=(R,2260)
         DS    0F
SAVE1    DC    18F'0'
SAVE2    DC    18F'0'
SAVE3    DC    18F'0'
COMSAVE  DC    6F'0'                   SAVE AREA FOR GACB1
PRC      DC    CL7'PROCESS'            PROCESSING ACTION CODE
INPUT    DC    240F'0'                 DEFINE INPUT AREA
         END   SAMP2260
```

Figure 8 (part 7 of 7). Coding for sample program

APPENDIX B: CCW'S ASSOCIATED WITH INPUT/OUTPUT CONTROL

The following are formats of CCWs generated when GREAD, GWRITE, and GCNTRL macro instructions are used.

GREAD DSM CCW FORMAT

```
Command   Data                                              Byte
 Code    Address              Flags*                        Count
┌──────┬────────────┬───┬───┬───┬───┬───┬──────────────┬────────────┐
│  02  │            │   │   │   │   │   │              │            │
└──────┴────────────┴───┴───┴───┴───┴───┴──────────────┴────────────┘
0     7 8         31  32  33  34  35  36                48        63
```

GREAD DSMW CCW FORMAT

```
Command   Data                                              Byte
 Code    Address              Flags*                        Count
┌──────┬────────────┬───┬───┬───┬───┬───┬──────────────┬────────────┐
│  12  │            │   │   │   │   │   │              │            │
└──────┴────────────┴───┴───┴───┴───┴───┴──────────────┴────────────┘
0     7 8         31  32  33  34  35  36                48        63
```

GREAD SMI CCW FORMAT

```
Command   Data                                              Byte
 Code    Address              Flags*                        Count
┌──────┬────────────┬───┬───┬───┬───┬───┬──────────────┬────────────┐
│  0A  │            │   │   │   │   │   │              │            │
└──────┴────────────┴───┴───┴───┴───┴───┴──────────────┴────────────┘
0     7 8         31  32  33  34  35  36                48        63
```

GREAD SMIW CCW FORMAT

```
Command   Data                                              Byte
 Code    Address              Flags*                        Count
┌──────┬────────────┬───┬───┬───┬───┬───┬──────────────┬────────────┐
│  1A  │            │   │   │   │   │   │              │            │
└──────┴────────────┴───┴───┴───┴───┴───┴──────────────┴────────────┘
0     7 8         31  32  33  34  35  36                48        63
```

GREAD DSB CCW FORMATS

```
Command   Data                                              Byte
 Code    Address              Flags*                        Count
┌──────┬────────────┬───┬───┬───┬───┬───┬──────────────┬────────────┐
│  06  │            │   │ 1 │   │   │   │              │    0002    │
└──────┴────────────┴───┴───┴───┴───┴───┴──────────────┴────────────┘
0     7 8         31  32  33  34  35  36                48        63

Command   Data                                              Byte
 Code    Address              Flags*                        Count
┌──────┬────────────┬───┬───┬───┬───┬───┬──────────────┬────────────┐
│  07  │            │   │   │ 1 │   │   │              │    0002    │
└──────┴────────────┴───┴───┴───┴───┴───┴──────────────┴────────────┘
0     7 8         31  32  33  34  35  36                48        63
```

*
Bit 33 -- chain command flag
Bit 34 -- suppress length indication flag

GREAD DSBW CCW FORMATS

```
Command    Data                                                Byte
  Code    Address          Flags*                             Count
┌──────┬────────────┬───┬───┬───┬───┬───┬───────────────┬───────────────┐
│ 16   │            │   │ 1 │   │   │   │               │               │
└──────┴────────────┴───┴───┴───┴───┴───┴───────────────┴───────────────┘
0     7 8          31  32  33  34  35  36                 48           63
```

```
Command    Data                                                Byte
  Code    Address          Flags*                             Count
┌──────┬────────────┬───┬───┬───┬───┬───┬───────────────┬───────────────┐
│ 17   │            │   │   │ 1 │   │   │               │               │
└──────┴────────────┴───┴───┴───┴───┴───┴───────────────┴───────────────┘
0     7 8          31  32  33  34  35  36                 48           63
```

GWRITE DSB CCW FORMAT

```
Command    Data                                                Byte
  Code    Address          Flags*                             Count
┌──────┬────────────┬───┬───┬───┬───┬───┬───────────────┬───────────────┐
│ 01   │            │   │   │ 1 │   │   │               │               │
└──────┴────────────┴───┴───┴───┴───┴───┴───────────────┴───────────────┘
0     7 8          31  32  33  34  35  36                 48           63
```

GWRITE DSBW CCW FORMAT

```
Command    Data                                                Byte
  Code    Address          Flags*                             Count
┌──────┬────────────┬───┬───┬───┬───┬───┬───────────────┬───────────────┐
│ 11   │            │   │   │ 1 │   │   │               │               │
└──────┴────────────┴───┴───┴───┴───┴───┴───────────────┴───────────────┘
0     7 8          31  32  33  34  35  36                 48           63
```

GWRITE EBW CCW FORMATS

```
Command    Data                                                Byte
  Code    Address          Flags*                             Count
┌──────┬────────────┬───┬───┬───┬───┬───┬───────────────┬───────────────┐
│ 07   │            │   │ 1 │ 1 │   │   │               │      0002     │
└──────┴────────────┴───┴───┴───┴───┴───┴───────────────┴───────────────┘
0     7 8          31  32  33  34  35  36                 48           63
```

```
Command    Data                                                Byte
  Code    Address          Flags*                             Count
┌──────┬────────────┬───┬───┬───┬───┬───┬───────────────┬───────────────┐
│ 01   │            │   │   │ 1 │   │   │               │               │
└──────┴────────────┴───┴───┴───┴───┴───┴───────────────┴───────────────┘
0     7 8          31  32  33  34  35  36                 48           63
```

GWRITE EBWW CCW FORMATS

```
Command    Data                                                Byte
  Code    Address          Flags*                             Count
┌──────┬────────────┬───┬───┬───┬───┬───┬───────────────┬───────────────┐
│ 17   │            │   │1  │ 1 │   │   │               │      0002     │
└──────┴────────────┴───┴───┴───┴───┴───┴───────────────┴───────────────┘
0     7 8          31  32  33  34  35  36                 48           63
```

```
Command    Data                                                Byte
  Code    Address          Flags*                             Count
┌──────┬────────────┬───┬───┬───┬───┬───┬───────────────┬───────────────┐
│ 11   │            │   │   │ 1 │   │   │               │               │
└──────┴────────────┴───┴───┴───┴───┴───┴───────────────┴───────────────┘
0     7 8          31  32  33  34  35  36                 48           63
```

*
Bit 33 -- chain command flag
Bit 34 -- suppress length indication flag

GWRITE LNE CCW FORMAT

```
Command   Data                                                  Byte
 Code     Address              Flags*                           Count
┌───────┬─────────────┬───┬───┬───┬───┬───┬───────────────┬─────────────┐
│  05   │             │   │   │ 1 │   │   │               │             │
└───────┴─────────────┴───┴───┴───┴───┴───┴───────────────┴─────────────┘
0      7 8          31  32  33  34  35  36              48            63
```

GWRITE LNEW CCW FORMAT

```
Command   Data                                                  Byte
 Code     Address              Flags*                           Count
┌───────┬─────────────┬───┬───┬───┬───┬───┬───────────────┬─────────────┐
│  15   │             │   │   │ 1 │   │   │               │             │
└───────┴─────────────┴───┴───┴───┴───┴───┴───────────────┴─────────────┘
0      7 8          31  32  33  34  35  36              48            63
```

GWRITE EWL CCW FORMATS

```
Command   Data                                                  Byte
 Code     Address              Flags*                           Count
┌───────┬─────────────┬───┬───┬───┬───┬───┬───────────────┬─────────────┐
│  07   │             │   │ 1 │ 1 │   │   │               │    0002     │
└───────┴─────────────┴───┴───┴───┴───┴───┴───────────────┴─────────────┘
0      7 8          31  32  33  34  35  36              48            63

Command   Data                                                  Byte
 Code     Address              Flags*                           Count
┌───────┬─────────────┬───┬───┬───┬───┬───┬───────────────┬─────────────┐
│  05   │             │   │   │ 1 │   │   │               │             │
└───────┴─────────────┴───┴───┴───┴───┴───┴───────────────┴─────────────┘
0      7 8          31  32  33  34  35  36              48            63
```

GWRITE EWLW CCW FORMATS

```
Command   Data                                                  Byte
 Code     Address              Flags*                           Count
┌───────┬─────────────┬───┬───┬───┬───┬───┬───────────────┬─────────────┐
│ 17    │             │   │1  │ 1 │   │   │               │    0002     │
└───────┴─────────────┴───┴───┴───┴───┴───┴───────────────┴─────────────┘
0      7 8          31  32  33  34  35  36              48          63

Command   Data                                                  Byte
 Code     Address              Flags*                           Count
┌───────┬─────────────┬───┬───┬───┬───┬───┬───────────────┬─────────────┐
│ 15    │             │   │   │ 1 │   │   │               │             │
└───────┴─────────────┴───┴───┴───┴───┴───┴───────────────┴─────────────┘
0      7 8          31  32  33  34  35  36              48          63
```

GCNTRL ERS CCW FORMAT

```
Command   Data                                                  Byte
 Code     Address              Flags*                           Count
┌───────┬─────────────┬───┬───┬───┬───┬───┬───────────────┬─────────────┐
│  07   │             │   │   │ 1 │   │   │               │    0002     │
└───────┴─────────────┴───┴───┴───┴───┴───┴───────────────┴─────────────┘
0      7 8          31  32  33  34  35  36              48            63
```

GCNTRL ERSW CCW FORMAT

```
Command   Data                                                  Byte
  Code    Address              Flags*                           Count
┌───────┬─────────────┬───┬───┬───┬───┬───┬───────────────┬─────────────┐
│  17   │             │   │   │ 1 │   │   │               │    0002     │
└───────┴─────────────┴───┴───┴───┴───┴───┴───────────────┴─────────────┘
0      7 8          31  32  33  34  35  36              48            63
```

*
Bit 33 -- chain command flag
Bit 34 -- suppress length indication flag

The operator at the 2260 display station can present information to the system through a keyboard. Keyboard input results in an attention interruption. Express attention handling requires that the application program check for the occurrence of an attention. When an attention is detected, attention information is routed to the appropriate application program attention handling routine.

Macro instructions are provided to communicate the occurrence of attentions to the application program. The macro instructions and their functions are listed in Figure 9.

BACKGROUND INFORMATION

When an attention from a 2260 occurs, the application program is interrupted and control goes to the control program. The control program sets a bit in the graphic control byte (GCB) associated with the display unit from which the attention was received. Control then returns to the application program at the point of interruption.

To determine whether or not an attention has occurred, the application program must include an attention analysis provision to periodically inspect the bits in the GCBs. This attention analysis can be done by using either the ANALYZ macro instruction or a programmer-written substitute. If the application programmer elects to perform his own attention analysis, he must employ the GSERV macro instruction to reset bits in the GCBs.

EXPRESS ATTENTION HANDLING MACRO INSTRUCTIONS

The macro instructions in this group are provided to help the application programmer service attentions from the 2260.

ANALYZ -- Graphic Attention Analysis Macro Instruction

The ANALYZ macro instruction performs attention analysis to determine the occurrence of and the reason for an attention from a keyboard entry on the 2260. This is done by a polling technique that inspects the bits in the GCBs. The ANALYZ macro instruction also resets the GCB (by calling on the GSERV macro instruction), places coded information into an output area defined by the application programmer, and gives control to the appropriate application program routine to handle the attention.

To employ the ANALYZ macro instruction, the application programmer must:

1. Include the ANALYZ macro instruction statement in his application program.

2. Provide a contiguous list of DCB addresses and UCB index factors that indicate the UCB containing the appropriate GCB from which an attention is possible. This list is provided at open time if the POLST operand is specified in the DCB macro instruction.

3. Specify the entry address of the routine to handle the keyboard attention.

4. Reserve eight bytes (beginning on a fullword boundary) in his program area into which the output of ANALYZ can be placed.

Mnemonic	Macro Instruction Name	Function
ANALYZ	Graphic Attention Analysis	Determines and analyzes the occurrence of an attention from a display station. The bits in the GCB are reset before control is given to an application program routine to handle the attention.
GSERV	Graphic Attention Service	Resets the bits in the GCBs associated with attentions. This macro instruction must be employed if an application programmer elects to perform his own attention analysis instead of using ANALYZ.

Figure 9. Express attention handling macro instructions

The format of the ANALYZ macro instruction is:

Name	Operation	Operand
[symbol]	ANALYZ	polst address, pointer address [,MF={L|(E,parameter list name)|(E,(n))}]

polst address
: is the address of a four-byte reserved area followed by a list of DCB addresses and UCB index factors that indicate the UCB containing the appropriate GCB. Each is aligned on a fullword boundary. The reserved area is used by ANALYZ to point to the last DCB address in the list. The list is constructed at open time if the POLST operand is specified in the DCB macro instruction. The format of the list is:

polst	Reserved Area	
	UCB Index	DCB Address
	UCB Index	DCB Address

pointer address
: is the symbolic address of the first word of a five-word field located on a fullword boundary in the application program area. The contents of the field are shown in Figure 10.

Note: The application programmer must insert zeros in three of the five words indicated in Figure 10. Word two of the pointer table must contain a valid routine address if either optional keyboard feature is used.

Word	Contents
1	(Not used. Programmer must specify zeros.)
2	Address of the application program Keyboard Attention routine.
3	(Not used. Programmer must specify zeros.)
4	(Not used. Programmer must specify zeros.)
5	Address of the first byte in the output area (see "Output" below).

Figure 10. Contents of the pointer table

MF=L
: specifies that expansion of the macro instruction is to consist of a parameter list only. The name of the parameter list is the symbolic name specified in the "Name" field of this ANALYZ macro instruction. The "polst" and "pointer" operands may only be written as relocatable expressions.

MF=(E, parameter list name)
: specifies that the macro instruction is to perform the ANALYZ functions using a parameter list constructed by another ANALYZ macro instruction. The "parameter list name" operand must correspond exactly to the name of the parameter list.

MF=(E,(n))
: performs the same function as (E, parameter list name). By writing register notation (in the range 2 through 12), the programmer signifies that he will load the address of the parameter list into the indicated register before the macro instruction is executed.

OPERAND FORM: The following are the allowable operand forms for the ANALYZ macro instruction.

1. If the MF keyword operand is omitted or specified as L, the "polst address" and "pointer address" operands must be written as relocatable expressions.

2. If the MF=E keyword operand is specified, the "polst address", "pointer address", and "parameter list name" operands can be written as relocatable expressions; as implied or explicit, indexed or nonindexed addresses; or in register notation. For example:

```
LISTNAME  ANALYZ  POLST,POINTER,MF=L
          ANAYLZ  POLST1,MF=(E,LISTNAME)
```

or

```
          LA      2,POLST1
          LA      3,LISTNAME
          ANALYZ  (2),MF=(E,(3))
```

ERROR CONDITIONS: If either the "polst address" or "pointer address" operand is missing at assembly time, a message is printed on the assembly listing indicating the omission, and macro generation continues. If either parameter is missing, the macro instruction does not perform its intended function.

OUTPUT: The output of the ANALYZ macro instruction is placed in an eight-byte output area whose address is in word five of the pointer table. This area must begin on a fullword boundary. Output consists of the following, in the order listed:

- Reserved (four bytes)

- UCB address index factor (one byte)

- Address of DCB (three bytes)

<u>ROUTINE EXITS</u>: If inspection of the DCB list finds no attention indication, a code of hexadecimal 04 is placed in the return code register, and control is returned to the next sequential instruction in the application program. If an attention is detected, the macro instruction determines the address of the DCB, resets the GCB, stores output data in the output area, and gives control to the appropriate application program attention handling routine with the address of the output area in register 1 and the return address in register 14. The application programmer must save and restore registers in his attention handling routine.

Note that the application program attention handling routine, upon completion, does not return control to ANALYZ. Control may be returned from the attention handling routine to the next sequential instruction after ANALYZ by use of the RETURN macro instruction.

When ANALYZ is executed, it starts initially at the top of the poll list; but after the initial keyboard attention has been processed, starts at the point of previous exit from the poll list.

A return to the application program next sequential instruction is made (1) if no attention occurred, or (2) if the address of a 2260 keyboard attention routine is not provided in word two of the pointer table, or (3) if the address of an output area is not provided in word five of the pointer table. The attention is lost if no keyboard attention routine is provided. The return code register contains a hexadecimal code to indicate the reason for this return, as follows:

Contents of Return Code Register	Meaning
00	Normal
04	No attention occurred
08	An asynchronous error attention occurred
0C	Poll list address or attention routine list address is not in the parameter list
10	Output area address is not in the attention routine list

Write-to-programmer messages are issued for the error conditions. For message explanations and programmer error checks, see <u>OS/VS Problem Determination Aids and Messages and Codes for GPS and GSP</u>.

```
* MAIN ROUTINE PROCESSING
          .
          .
          .
          ANALYZ    POLST,POINTER
          .
          .
          .
POLST     DC        F'0'
          DS        2F
POINTER   DC        F'0'
          DC        A(KBRTN)
          DC        2F'0'
          DC        A(OUTAREA)
OUTAREA   DC        2F'0'
```

Figure 11. Graphic attention analysis routine, coding example

<u>EXAMPLE</u>: Figure 11 illustrates applicable coding in a typical IBM 2260 Local graphic job that uses the ANALYZ macro instruction.

POLST is the symbolic address of the first fullword entry in a list of DCB addresses and UCB index factors. Note that the first word in the list is set to zeros. POINTER is the symbolic address of the first entry in a five-word table that contains the entry address of the application program attention handling routines and the address of an output area (OUTAREA).

<u>SUGGESTIONS FOR THE APPLICATION PROGRAMMER</u>: The application programmer opens the graphic device, thereby setting up the appropriate control blocks. A suggested initialization procedure is the execution of a channel program, after the device is open, to inform an operator at the display station that the graphic program is now executing. Polling might then be started immediately by the application program if information resulting from attentions is to be processed.

During the period that an attention might logically be expected from a display station, polling should be done periodically. The timing of this function may be handled either by employing the ANALYZ macro instruction at appropriate times during execution of the problem program, or by use of a Set Interval Timer (STIMER) system macro instruction which causes the System/370 interval timer to employ the ANALYZ macro instruction at programmer-specified intervals. (See <u>OS/VS Supervisor Services and Macro Instructions</u> for the format and use of the STIMER macro instruction.) However, if the timer is used for establishing the interval between poll cycles, it is not available for establishing other task intervals.

```
         .
         .
         .
         OPEN     DCB1
         STIMER   REAL,POLADR,
         .        DINTVL=TIME
         .
         .
* MAIN ROUTINE PROCESSING
         .
         .
         .
         CLOSE    DCB1
         .
         .
         .
         DS       0D
TIME     DC       Z'00000020'
POLADR   ANALYZ   POLST,POINTER
* IF NO ATTENTION, RETURN TO MAIN RTN
         .
         .
         .
```

Figure 12. Implementation of polling

Note: If the application programmer wishes, he may wait for an attention by repeatedly executing the polling routine (that is, branching to the ANALYZ statement). It should be noted, however, that such a technique would defeat one of the primary purposes of a system having multiprogramming capabilities.

The coding of Figure 12 shows how the System/370 interval timer can be used to establish a polling interval. After the DCB for the device has been opened, use of the STIMER macro instruction provides an automatic transfer to the polling routine after a programmer-specified interval of real time (0.2 seconds in Figure 12). Until the interval has elapsed, processing continues in the main routine. Action taken by the polling routine, ANALYZ, depends upon whether or not an attention occurred. Control is passed to the application program attention handling routine if there was an attention or is returned to the instruction following the ANALYZ macro instruction if an attention did not occur.

GSERV -- Graphic Attention Service Macro Instruction

The GSERV macro instruction resets GCBs associated with attentions, so that an attention may not be handled more than once. This macro instruction is used by ANALYZ to reset the bits in the GCBs. If the application programmer elects to perform his own attention analysis, GSERV must be used to reset (that is, zero) bits in the GCBs. The format of the GSERV macro instruction is:

Name	Operation	Operand
[symbol]	GSERV	{list item address, gcb bit|(1)}

list item address
    is the symbolic address of a four-byte field, aligned on a fullword boundary, in the programmer's list of DCB addresses and UCB index factors. This list is constructed at open time if the POLST operand is written in the DCB macro instruction. The four-byte field contains the address of the DCB and the UCB index factor that indicates the appropriate UCB for the device.

```
      ┌─────┬───────────┐
      │UCB  │DCB Address│list item
      │index│           │ address
      └─────┴───────────┘
bit   0    7 8         31
```

gcb bit
    specifies which bit in the GCB is to be reset. The value of "gcb bit" must be 1 for the 2260.

If register notation is used, the parameters must have been loaded into register 1 before execution of the macro instruction. The format for loading the information into the register is as follows:

```
      ┌───────┬─────────┐
      │gcb bit│list item│ register 1
      │       │ address │
      └───────┴─────────┘
bit   0      7 8       31
```

To define the bit to be reset, the programmer codes the high-order byte in the register as follows:

Type of Attention	GCB Bit to be Reset	Code
Keyboard Entry	7	DC X'01'

The address that comprises the three low-order bytes may be obtained from the poll list created by the OPEN macro instruction if the POLST operand was specified in the DCB macro instruction.

If the UCB address is invalid or if the device is not a graphic device, the job is abnormally terminated.

After GSERV has reset the GCB bit, control is returned to the next sequential instruction of the calling program.

PROGRAMMING NOTES: The graphic control byte is set to zero when the first DCB is opened for any display station.

APPENDIX D: SUMMARY OF OPERANDS

The Summary of Operands, shown in this appendix, contains all of the macro instructions and indicates the allowable ways of writing the operands. The summary is designed so that it can be opened with all the macro instructions displayed when the descriptions are being read. The classifications used in the summary are as follows:

SYM
  is any symbol valid in the assembler language.

DEC INT
  are any decimal digits up to the value indicated in the associated macro instruction description. If both SYM and DEC INT are checked, an absolute expression is also allowed.

REGISTER
  is always coded within parentheses as follows:
  (2-12) - one of general registers 2 through 12, previously loaded with the right-adjusted value or address specified in the associated macro instruction description. The unused high-order bits must be set to zero. The register may be designated symbolically or with an absolute expression.

  (1) - general register 1, previously loaded as indicated above. The register can be designated only as (1).

Note: Register notation may be used with E-form macro instructions only.

RX TYPE
  any address that is valid in a RX-type instruction (e.g., LA) may be designated.

A-TYPE ADCON TYPE
  any address that may be written in an A-type address constant may be designated.

Note:

- Underscoring indicates a default option; that is, if no operand is coded, the underscored operand is assumed.
- Braces,{ }, indicate that one operand from a vertical or horizontal group must be chosen unless a default option is indicated.
- The vertical stroke (|) separates alternative operands within a horizontal group.

MACRO INSTRUCTION	OPERANDS	WRITTEN AS					
		SYM	DEC INT	Register (*)		RX type	A-type Adcon type
				(2-12)	(1)		
ANALYZ	polst address	refer to macro description					
	pointer address	refer to macro description					
	MF= L	written as shown					
	(E,	written as shown					
	parameter list name)	x		x			
ATTNINQ	gacb address			x		x	
	MODE= R	written as shown					
	W	written as shown					
	(C,	written as shown					
	branch address)			x			x
	CLEAR	written as shown					
	UNIT=	1 through 25 or ALL					
	MF = L	written as shown					
	(E,	written as shown					
	parameter list address)			x		x	
CLOSE	dcb address			x			x
	MF= L	written as shown					
	(E,	written as shown					
	parameter list name)	x			x		
DAR	gacb address			x			x
	MF= L	written as shown					
	(E,	written as shown					
	gacb list address			x		x	
DCB	DSORG=GS	written as shown					
	MACRF=	{R | W | RC | WC | (R,W) | (RC,W) | (R,WC) | (RC,WC) }					
	GTYPE=	{BASIC | EXPRESS }					
	DDNAME=ddname	x					
	EXLST=exit list address						x
	GNCP=number of blocks		x				
	POLST=poll list address						x
GCNTRL	decb name	x			x		
	type	{ERS | ERSW }					
	dcb address			x			x
	unit		x				
	MF=	{L | E }					
GREAD	decb name	x			x		
	type	{DSM |DSMW | SMI |SMIW | DSB | DSBW }					
	dcb address			x			x
	length		x	x			
	area address			x			x
	unit		x	x			
	MF=	{L | E }					
GSERV	list item address	refer to macro description					
	gcb bit	refer to macro description					
GWRITE	decb name	x			x		
	type	{DSB | DSBW | EBW | EBWW | LNE |LNEW | EWL | EWLW }					
	dcb address			x			x
	length		x	x			
	area address			x			x
	unit		x	x			
	MF=	{L | E }					
OPEN	dcb address			x			x
	MF= L	written as shown					
	(E,	written as shown					
	parameter list name)	x			x		

MACRO INSTRUCTION	OPERANDS	WRITTEN AS					
		SYM	DEC INT	Register (*)		RX type	A-type Adcon type
				(2-12)	(1)		
SAEC	EP= {entry point address			x		x	
	0		x				
	DCB=dcb address			x		x	
	COMAREA=communication area address			x		x	
	ATTNTYP=	{({A | S | R },2260) | SAVE | RESTORE }					
	MF=	{L | E }					
SPAR	gacb address			x			x
	PRTY=		x				
	MF= L	written as shown					
	(E,	written as shown					
	gacb list address)			x		x	

(*) Register notation may be used with E-form macro instructions only.

Indexes to systems reference manuals are consolidated in OS/VS Master Index, GC28-0602. For additional information about any subject listed below, refer to other publications listed for the same subject in the master index.

Where more than reference is given, the major reference is first.

OS/VS Graphic Programming
Services (GPS) for
IBM 2260 Display Station
(Local Attachment)

GC27-6972-1

READER'S COMMENT FORM

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

How did you use this publication?

□ As an introduction　　□ As a text (student)

□ As a reference manual　　□ As a text (instructor)

□ For another purpose (explain)____

____

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

Cut or Fold Along Line

What is your occupation?____

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

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

Thank you for your cooperation. No postage stamp necessary if mailed in the U.S.A. (Elsewhere, an IBM office or representative will be happy to forward your comments.)

Your comments, please . . .

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

Fold Fold

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST CLASS PERMIT NO. 40 ARMONK, N.Y.

POSTAGE WILL BE PAID BY ADDRESSEE:

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

Fold Fold

IBM®

Cut or Fold Along Line

GC27-6972-1

OS/VS Graphic Programming Services (GPS) for IBM 2260 Display Station (Local Attachment) Printed in U.S.A. GC27-6972-1