Update Package

UPD4031-13A

for

PL/I Subset G Reference Guide
IDR4031

December 1984

This Update Package, UPD4031-13A, is Update 3 for the April 1980 edition of the PL/I Subset G, IDR4031. Pages that have been changed or added are listed on the next page.

A vertical bar in the margin of the table of contents indicates an addition since the last printing.

Technical Publications Department
500 Old Connecticut Path
Framingham, MA 01701

Update Package, UPD4031-13A

Pages to add:

Insert Pages	After Page
G-1 through G-14	F-8

Pages to change:

Insert Pages	In Place of Pages
i through viii	i through vi
X-1 through X-22	X-1 through X-20

PL/I Subset G Reference Guide

IDR4031

First Edition

Updated for Revision 19.4

by

Robert L. Dimmig

This guide documents the software operation of the Prime Computer and its supporting systems and utilities as implemented at Master Disk Revision Level 19.4 (Rev. 19.4).

Prime Computer, Inc.
500 Old Connecticut Path
Framingham, Massachusetts 01701

COPYRIGHT INFORMATION

PRINTING HISTORY — PL/I Subset G Reference Guide

Edition	Date	Number	Software Release
First Edition	April 1980	IDR4031	17.0
Update 1	December 1980	PTU2600-075	18.0
Update 2	July 1982	PTU2600-08	19.0
Update 3	December 1984	UPD4031-13A	19.4

Changes made to the text since the last printing are identified by vertical bars in the margin.

CUSTOMER SUPPORT CENTER

Prime provides the following toll-free numbers for customers in the United States needing service:

1-800-322-2838 (within Massachusetts)
1-800-541-8888 (within Alaska)
1-800-651-1313 (within Hawaii)
1-800-343-2320 (within other states)

HOW TO ORDER TECHNICAL DOCUMENTS

Obtain an order form, a catalog, and a price list from one of the following:

U.S. Customers

Software Distribution
Prime Computer, Inc.
1 New York Ave.
Framingham, MA 01701
(617) 879-2960 X2053

Prime Employees

Communications Services
MS 15-13, Prime Park
Natick, MA 01760
(617) 655-8000 X4837

Customers Outside U.S.

Contact your local Prime subsidiary or distributor.

CONTENTS

APPENDICES

APPENDIX G

UPDATE FOR REVISION 19.4

This document is the third update to the First Edition of the PL/I Subset G Reference Guide (IDR4031). It includes features released at Master Disk Revision 19.4 (Rev. 19.4) plus some documentation corrections.

This document should be inserted as the last appendix, Appendix G, in your book.

UPDATED COMPILER OPTIONS

The changes in compiler options for Rev. 19.4 include:

- New syntaxes
- New options that have been added to implement new functionality
- New abbreviations for most options

These changes were implemented to make Prime's translator products more standardized in their user interface, as well as among themselves and with other Prime software products.

New Syntaxes

New syntaxes have been introduced to replace the older forms of some existing options. For example, the new syntax -OPTIMIZE [n]", where n is the level of optimization to be performed, has replaced the the older syntax of the options -OPTIMIZE and -NOOPTIMIZE.

Forming Negated Options: The construction of negated options is now done by prefixing the name of the positive form with -NO_ (for example, -MAP and -No_MAP). To form the abbreviated name of a negated option, prefix the positive abbreviated name with -N (in this case, -MAP and -NMAP).

Specification of Options on Command Line: Processing of compiler options previously allowed duplicate and conflicting options on the same command line (for example, specifying an option twice, or specifying -PRODUCTION and -NOPRODUCTION for the same compiler invocation). In these occurrences of conflicting options, the one that had been specified last on the command line was the option that took control. Duplicate and conflicting options are now considered obsolete, and may cause fatal errors in future software revisions.

New Options

At Rev. 19.4, a number of new compile-time options replace current compiler options. The compiler will issue warnings if you should use the outdated forms, and the obsolete forms may cause fatal errors in future software revisions.

The updated options are:

```
-32I
-64V
-ALLOW_PRECONNECTION/-NO_ALLOW_PRECONNECTION
-BIG/-NO_BIG
-BINARY/-NO_BINARY
-COPY/-NO_COPY
-DEBUG/-NO_DEBUG
-ERRLIST/-NO_ERRLIST
-ERRTTY/-NO_ERRTTY
-EXPLIST/-NO_EXPLIST
-FRN/-NO_FRN
-FULL_HELP
-FULL_OPTIMIZE
-HELP
-INPUT pathname
-LCASE/-UPCASE
-LISTING [file]/-NO_LISTING
```

Updated options (continued)

-MAP/-NO_MAP
-NESTING/-NO_NESTING
-OFFSET/-NO_OFFSET
-OPTIMIZE [n]
-OVERFLOW/-NO_OVERFLOW
-PRODUCTION/-NO_PRODUCTION
-RANGE/-NO_RANGE
-SILENT (with decimal argument)
-SOURCE pathname
-SPACE and -TIME
-STATISTICS/-NO_STATISTICS
-STORE_OWNER_FIELD/-NO_STORE_OWNER_FIELD
-XREF/-NO_XREF

Following is a brief description of each option. The full name is listed along with the name of its opposite, negated form, if it has one. Proper abbreviations are indicated. Abbreviations that are now considered obsolete have been listed as such.

-32I
The -32I option generates 32I-mode code. There is no abbreviation for this option. Obsolete forms: -3,-32.

-64V
The -64V option generates 64V-mode code. There is no abbreviation for this option. Obsolete forms: -6, -64.

-ALLOW_PRECONNECTION/-NO_ALLOW_PRECONNECTION
You may specify the -ALLOW_PRECONNECTION option to permit the preconnection of a listing output to a pre-opened file unit 2, or a binary output to a pre-opened file unit 3. These preconnections must be made with the PRIMOS commands LISTING and BINARY, which are discussed in the PRIMOS Commands Reference Guide. If you specify the negative form of the compiler option, the compiler will always open and close the listing and binary files (and will use dynamic file units). The negative form is the default. Accepted abbreviations: -APRE for -ALLOW_PRECONNECTION, and -NAPRE for -NO_ALLOW_PRECONNECTION.

-BIG/-NO_BIG
The -BIG option can be used to handle arrays that span segment boundaries, if the compiler itself does not detect that these segment boundaries have been spanned. The negated option, -NO_BIG, is the default. Accepted abbreviations: -NBIG for -NO_BIG, no abbreviation for -BIG. Obsolete forms: -NOB, -NOBI, -NOBIG.

-BINARY [file]/-NO_BINARY
The -BINARY option specifies that a binary object file will be produced. The -NO_BINARY option replaces -BINARY NO, which is considered obsolete, to specify that no binary file is to be produced. The default is to produce a binary file. Accepted abbreviation: -B [file] for -BINARY [file] and -NB for -NO_BINARY. Obsolete forms: -BI, -BIN, -BINA, -BINAR, -NOB, -NOBI, -NOBIN, -NOBINA, -NOBINARY.

-COPY/-NO_COPY
To prevent constants from being changed by a called procedure, the -COPY option will make a copy of the constant before it is passed. The -NO_COPY option specifies that such a copy need not be made, on the assumption that the programmer has been careful not to pass constants as actual parameters to formals that get modified. -COPY is the default. Accepted abbreviations: -COP for -COPY, -NCOP for -NO_COPY.

-DEBUG/-NO_DEBUG
The -DEBUG option generates the full debugger (DBG) functionality code. Accepted abbreviations: DBG, NDBG. Obsolete forms: -DE, -DEB, -DEBU, -NOD, -NODE, -NODEB, -NODEBU, -NODEBUG.

-ERRLIST/-NO_ERRLIST
The -ERRLIST option prints a listing of errors only. Accepted abbreviations: -ERRL for -ERRLIST, -NERRL for -NO_ERRLIST. Obsolete forms: -ERRLI, -ERRLIS, -NOERRL, -NOERRLI, -NOERRLIS, -NOERRLIST.

-ERRTTY/-NO_ERRTTY
The -ERRTTY option prints error messages at user terminal. Accepted abbreviations: -ERRT for -ERRTTY, -NERRT for -NO_ERRTTY. Obsolete forms: -ERRTT, -NOE, -NOERRT, -NOERRTT, -NOT, -NOTT, -NOTTY, -NOTTYD, -NOTTYDI, -NOTTYDIA, -NOTTYDIAG, -NOTTYDIAGS.

-EXPLIST/-NO_EXPLIST
The -EXPLIST option prints a listing, including assembler-like output. Accepted abbreviations: -EXP for -EXPLIST, -NEXP for -NO_EXPLIST. Obsolete forms: -EX, -EXPL, -EXPLI, -EXPLIS, -NOEX, -NOEXP, -NOEXPL, -NOEXPLI, -NOEXPLIS.

-FRN/-NO_FRN
The -FRN option generates special code to improve the accuracy of single-precision floating-point calculations (by doing a floating round on store instructions). Accepted abbreviations: -NFRN for -NO_FRN. Obsolete forms: -F, -FR, -NOF, -NOFR, -NOFRN.

-FULL_HELP
This option is similar to the -HELP option, except that in addition to the usage summary, a description of each compiler option is given. Accepted abbreviation: -FH.

-FULL_OPTIMIZE
This option indicates that the highest possible optimization should be used. The program listing will display the optimization level raised by the use of this option. Accepted abbreviation: -FOPT.

-HELP
The -HELP option will produce information on how to use the invoked compiler, including a list of compiler options. However, descriptions of the options are not provided. The user is referred to the system HELP command to obtain full information about the installed compiler, and to the -FULL_HELP option for full information about the invoked compiler. Accepted abbreviation: -H.

Note: If the user makes a mistake in specifying the options, or gives only the name of the compiler on the command line, then the user will be referred to the -HELP option after the error message has been given.

-INPUT pathname
This is an alternate way of specifying the source of the input file. If pathname is TTY, then input will come from the user terminal. Accepted abbreviation: -I pathname. Obsolete forms: -IN, -INP, -INPU.

-LCASE/-UPCASE
These compiler options control the compiler's automatic conversion to uppercase. When -UPCASE is used, everything in the source program, except literals, is translated to uppercase; when -LCASE is used, the source program (except for reserved words) is left as is. With -LCASE a program can have two data items (X and x, for example) which are treated as separate items. The -UPCASE option is the default. Accepted abbreviations: -UP for -UPCASE, -LC for -LCASE. Obsolete forms: -LCA, -LCAS, -U, -UPC, -UPCA, -UPCAS.

-LISTING [file]/-NO_LISTING
The -LISTING option is used both to specify that a source file listing is to be produced and, optionally, to specify where the listing is to go. The -NO_LISTING option takes the place of -LISTING NO, which is considered obsolete, to specify that no listing file is to be produced. The negative option is the default. Accepted abbreviations: -L for LISTING [file], -NL for -NO_LISTING. Obsolete forms: -LI, -LIS, -LIST, -LISTI, -LISTIN.

-MAP/-NO_MAP
The -MAP option produces information similar to the -XREF option, but without the source line references. When you specify -MAP, you will automatically have a listing file produced. Conversely, when you specify -L, you will automatically get the map information. If

you do not want your listing file to have the map information, you need to specify -L with -NO_MAP, as -MAP is the default. Accepted abbreviations: -MA for -MAP, -NMA for -NO_MAP. Obsolete form: -NO_M.

-NESTING/-NO_NESTING
The -NESTING option adds nesting level numbers in the program listing. Accepted abbreviations: -NE for -NESTING, -NNE for -NO_NESTING. Obsolete forms: -NONE, -NONES, -NONEST, -NONESTI, -NONESTIN, -NONESTING.

-OFFSET/-NO_OFFSET
The -OFFSET option appends an offset map to the listing file. Its use also implies that -L must be specified on the command line. For each statement in the source program, the offset map gives the offset in the object file of the first machine instruction generated for that statement. The -NO_OFFSET option means that no offset map is created, and is the default. Accepted abbreviations: -OFF for -OFFSET, -NOFF for -NO_OFFSET. Obsolete forms: -O, -OF, -OFFS, -OFFSE, -NOOF, -NOOFF, -NOOFFS, -NOOFFSE,-NOOFFSET.

-OPTIMIZE [dec-integer]
This option is the same as the existing optimize option except that it may be followed by a decimal integer that specifies an optimization level. If the decimal integer is not specified, the default value is 2. To turn optimization off, -OPT 0 should be used. -NO_OPTIMIZE IS now considered to be obsolete. Accepted abbreviation: -OPT [dec-integer]. Obsolete forms: -OP, -OPT1, -OPT2, -OPT3, -NOOP, -NOOPTIMIZE.

Following is a brief description of what types of optimizations are performed at each level. Each optimization level performs all the optimizations of the next lower level, plus those that are listed. Note that the functionality associated with some levels at Revision 19.4 may change in the future, and that more levels of optimization may be added at later revisions.

0: Perform no optimizations. This level replaces the option -NOOPTIMIZE.

1: Pattern replacement.

2: Common subexpression elimination.

3: Loop invariant removal.

-OVERFLOW/-NO_OVERFLOW
The -OVERFLOW option enables integer overflow conditions. The negated option, -NO_OVERFLOW, is the default. Accepted abbreviations: -OVF for -OVERFLOW, -NOVF for -NO_OVERFLOW. Obsolete forms: -OVE, -OVER, -OVERF, -OVERFL, -OVERFLO, -NOOV, -NOOVE, -NOOVER, -NOOVERF, -NOOVERFL, -NOOVERFLO, -NOOVERFLOW.

-PRODUCTION/-NO_PRODUCTION
These options are alternative options controlling code for Prime's Source Level Debugger, DBG. The -PRODUCTION option is similar to -DEBUG, except that no statement information is produced. With -PRODUCTION, you may not breakpoint at individual statements, nor step through your program. The execution time of a program compiled with -PRODUCTION will be better than the execution time of a program compiled with -DEBUG. The -NO_PRODUCTION option is the default. Accepted abbreviations: -PROD for -PRODUCTION, and -NPROD for -NO_PRODUCTION. Obsolete forms: -P, -PR, -PRO, -PRODU, -PRODUC, -PRODUCT, -PRODUCTI, -PRODUCTIO, -NOP, -NOPR, -NOPRO, -NOPROD, -NOPRODU, -NOPRODUC, -NOPRODUCT, -NOPRODUCTI, -NOPRODUCTIO, -NOPRODUCTION.

-RANGE/-NO_RANGE
These compiler options control runtime subscript range checking. When -RANGE is specified on the compile line, any variable array subscript is checked for out-of-range conditions, and if any occurs, a runtime error is issued and the program halts. The use of -RANGE will increase both the compile time and execution time of a program, and therefore should be used only as a debugging tool. The -NO_RANGE option will suppress this capability, and is the default option. Accepted abbreviations: -RA for -RANGE, and -NRA for -NO_RANGE. Obsolete forms: -R, -RAN, -RANG, -NOR, -NORA, -NORAN, -NORANG, -NORANGE.

-SILENT [n]
This option has been modified to accept a decimal argument, which indicates an error severity level. When n is specified, all errors with that severity and less are NOT reported. The default value of n is 1. Accepted abbreviation: -SI. Obsolete forms: -SIL, -SIL1, -SIL2, -SIL3, -SILE, -SILEN, -SILENT, -SILENT1, -SILENT2, -SILENT3.

-SOURCE pathname
This is an alternate way of specifying the source of the input file. If pathname is TTY, then input will come from the user terminal. Accepted abbreviation: -S pathname. Obsolete forms: -SO, -SOU, -SOUR, -SOURC.

-SPACE and -TIME
The -SPACE option specifies that the size of the optimized code (space) is to be given preference over the speed of the optimized code (time) in optimization consideration. The opposite of -SPACE is -TIME, which means that optimization is to favor time over space. The -TIME option is the default. There are no abbreviations for either of these two options.

-STATISTICS/-NO_STATISTICS
The -STATISTICS option displays compilation statistics at the terminal. The -NO_STATISTICS option suppresses compile time statistics, which may be generated by the -STATISTICS option. The negative option is the default. Accepted abbreviations: -STAT for -STATISTICS, -NSTAT for -NO_STATISTICS. Obsolete forms: -STATI,

-STATIS, -STATIST, -STATISTI, -STATISTIC, -NOST, -NOSTA, -NOSTAT, -NOSTATI, -NOSTATIS, -NOSTATIST, -NOSTATISTI, -NOSTATISTIC, -NOSTATISTICS, -TO, -TOT, -TOTA, -TOTAL, -TOTALS.

-STORE_OWNER_FIELD/-NO_STORE_OWNER_FIELD

The -STORE_OWNER_FIELD option causes the identity of the current program to be stored in a known place for use by traceback routines, such as DMSTK. This option is the default. Using the -NO_STORE_OWNER_FIELD option will not save this information, but will save a small code sequence for extremely time-critical programs. Accepted abbreviations: -SOF for -STORE_OWNER_FIELD, and -NSOF for -NO_STORE_OWNER_FIELD. Obsolete forms: -STO, -STOR, -STORE, -STORE_, -STORE_O, -STORE_OW, -STORE_OWN, -STORE_OWNE, -STORE_OWNER, -STORE_OWNER_, -STORE_OWNER_F, -STORE_OWNER_FI, -STORE_OWNER_FIE, -STORE_OWNER_FIEL, -NOSTO, -NO_STOR, -NO_STORE, -NO_STORE_, -NO_STORE_O, -NO_STORE_OW, -NO_STORE_OWN, -NO_STORE_OWNE, -NO_STORE_OWNER, -NO_STORE_OWNER_, -NO_STORE_OWNER_F, -NO_STORE_OWNER_FI, -NO_STORE_OWNER_FIE, -NO_STORE_OWNER_FIEL, -NSO, -NO_OW, -NO_OWN, -NO_OWNE, -NO_OWNER, -NO_OWNERI, -NO_OWNERID.

-XREF/-NO_XREF

The -XREF option will produce a listing with cross references of data and procedure names. Accepted abbreviations: -XR for -XREF, -NXR for -NO_XREF. Obsolete forms: -X, -XRE, -NOX, -NOXRE, -NOXREF.

New Abbreviations

A number of abbreviations for compiler options have been modified. The old abbreviations will still be accepted at Revision 19.4, but will not be accepted at later revisions.

Following is a list of PL1G compiler options, and their official abbreviations. Default options are preceded by an asterisk (*).

Option	Abbreviation
-32I	(none)
* -64V	(none)
-ALLOW_PRECONNECTION	-APRE
* -NO_ALLOW_PRECONNECTION	-NAPRE
-BIG	-(none)
* -NO_BIG	-NBIG
* -BINARY [file]	-B [file]
-NO_BINARY	-NB
* -COPY	-COP
-NO_COPY	-NCOP
-DEBUG	-DBG
* -NO_DEBUG	-NDBG
-ERRLIST	-ERRL
* -NO_ERRLIST	-NERRL
* -ERRTTY	-ERRT
-NO_ERRTTY	-NERRT
-EXPLIST	-EXP
* -NO_EXPLIST	-NOEXP
-FRN	-FRN
* -NO_FRN	-NFRN
-FULL_HELP	-FH
-FULL_OPTIMIZE	-FOPT
-HELP	-H
-INPUT pathname	-I pathname
-LCASE	-LC
-LISTING [file]	-L [file]
* -NO_LISTING	-NL
* -MAP	-MA
-NO_MAP	-NMA
-NESTING	-NE
* -NO_NESTING	-NNE
-OFFSET	-OFF
* -NO_OFFSET	-NOFF
* -OPTIMIZE [dec-integer]	-OPT [dec-integer]
-OVERFLOW	-OVF
* -NO_OVERFLOW	-NOVF
-PRODUCTION	-PROD
* -NO_PRODUCTION	-NPROD
-RANGE	-RA
* -NO_RANGE	-NRA
* -SILENT [dec-integer]	-SI [dec-integer]
-SOURCE pathname	-S pathname
-SPACE	(none)
-STATISTICS	-STAT

Option (continued)	Abbreviation
* -NO_STATISTICS	-NSTAT
* -STORE_OWNER_FIELD	-SOF
-NO_STORE_OWNER_FIELD	-NSOF
* -TIME	(none)
* -UPCASE	-UP
-XREF	-XR
* -NO_XREF	-NXREF

NEW FUNCTIONALITY

Locked MIDAS + Records

The ON ERROR statement can now be used to catch the raised condition when a locked record of a MIDAS file is referenced. The corresponding value returned by the built-in function ONCODE is 1130, and it can be used to distinguish the special condition from other error conditions.

Installation and Build Procedures

PL1G is now supplied with two directories: PL1G and PL1G_LIBRARY. In order to use PL1G, the user must run two install files, PL1G>PL1G.INSTALL.COMI and PL1G_LIBRARY>PL1G_LIBRARY.INSTALL.COMI.

Obsolete Options

This is the last revision that will accept the -OBDATA and -NOOBDATA options. They are the same as the -EXPLIST and -NO_EXPLIST options, which are the more acceptable forms.

BIND: A NEW LINKING/LOADING UTILITY

An important new feature you should know about at Rev. 19.4 is Prime's new, more efficient program loading utility named BIND. If you are using a Rev. 19.4 or higher system, it would be advantageous for you to use this new loader. Prime's former loader, SEG, is still available to all users. However, in this update and future revisions of this book, BIND, not SEG, will be documented and used in examples. If you want more information on SEG, you should consult the SEG and LOAD Reference Guide.

Complete information on Prime's new loader, BIND, is contained in the new Programmer's Guide to BIND and EPFs. This book is also released at Rev. 19.4. In this update, however, a minimum amount of BIND information will be presented to get you started.

What is BIND?

BIND is Prime's new dynamic linking/loading utility for loading V-mode and I-mode programs. The executable files (runfiles) that BIND creates from the object (binary) files produced by Prime's high-level languages are called EPFs (Executable Program Formats).

Used for all virtual addressing mode programs, BIND makes it easier for you to build and maintain software. Unlike SEG, BIND creates dynamic EPF runfiles. This means that these runfiles can execute in any segment, or segments, of PRIMOS and do not need to use the same segment each time they are invoked. These runfiles can be assigned by PRIMOS at runtime, instead of load time, into free segments of your address space. A runfile can also be interrupted, and then resumed with the PRIMOS RESUME command.

SEG, on the other hand, creates static runfiles. That is, with SEG, all segment locations used by a program are assigned at load time by PRIMOS, and the runfile uses the same segment every time it executes. Whereas EPFs can be suspended and restarted at the PRIMOS level without intervening programs overwriting and destroying the suspended program, SEG's runfiles cannot.

One of BIND's most convenient features is that you can actually create an EPF on one PRIMOS command line. That is, the entire loading process can be entered on just one line.

Another advantage of BIND is the ability to invoke certain programs via DBG, without causing the program and Debugger to compete for the same storage space. As described in Appendix M of the Source Level Debugger User's Guide (UPD4033-21A), when SEG and DBG are called upon to process programs that use MIDAS+ or DBMS, the resulting storage overlap may lead to runtime errors. Although it is possible to reload the program and use A/SYMBOL commands to sidestep the problem, BIND's dynamic EPF runfiles avoid this problem altogether.

Using BIND

After compiling your program in the usual way, with the suffix .PL1G appended to the source file, you can use BIND in one of two ways:

- Directly on one PRIMOS command line
- Interactively by invoking subcommands of BIND

Invoking on One Command Line: To run BIND from the PRIMOS command line, enter the command:

```
BIND [epf-filename] -LO[object-file...] -LI PL1GLB -LI
```

When using BIND on a PRIMOS command line, you must precede each argument (BIND command) with a hyphen (-). The epf-filename is the name of your runfile (EPF). This name need not be the name of your source program. If the epf-filename is missing, BIND uses the name of the first loaded object file.

A loading session using one PRIMOS command line looks like this:

```
OK, BIND TEST -LOAD TEST TEST.2  -LIBRARY PL1GLB -LIBRARY
[BIND rev 19.4]
BIND COMPLETE
OK,
```

In the example shown above, the -LOAD argument is used to load the main program, TEST, and any other binary file, in this case, TEST.2. The -LIBRARY PL1GLB argument loads the PL1G library, and the -LIBRARY command loads the system library.

When the bind is complete, BIND saves your runfile (EPF), with the default name epf-filename.RUN, in the directory to which you are attached. If you do not specify an epf-filename, BIND automatically adds the suffix .RUN to the first object file that you load and saves the runfile in your directory. In the example above, the name of the runfile is:

```
TEST.RUN
```

Invoking Interactively: To invoke BIND interactively, enter:

```
BIND
```

at PRIMOS command level. BIND then asks you with a colon (:) prompt to load your files and libraries. An interactive loading session looks like this:

```
OK, BIND TEST
[BIND rev 19.4]
: LOAD TEST
: LOAD TEST.2
: LIBRARY PL1GLB
: LIBRARY
BIND COMPLETE
: FILE
OK,
```

BIND creates your runfile in the directory to which you are attached and names it TEST.RUN. The FILE command saves the EPF runfile and returns you to PRIMOS command level.

Executing Runfiles Created with BIND: To execute a runfile created with BIND, enter the command:

```
RESUME filename
```

Other commands: Three other useful BIND commands are QUIT, MAP, and HELP. The QUIT command returns you to PRIMOS immediately without saving the current EPF. Using the MAP command with the -UNDEFINED option identifies unresolved references if you do not receive a "BIND COMPLETE" message. HELP can give you online help on the commands if you run into trouble while using BIND. For more information see the Programmer's Guide to BIND and EPFs.

Where to Substitute BIND for SEG

In the PL/I Subset G Reference Guide, there is one reference to SEG. To update your book, you should note the following page, where SEG is referenced, then substitute BIND information in place of SEG. The SEG reference appears on the following page:

1-3

Index

W

X