SC21-7852-1
S5280-28

IBM 5280 Distributed Data System

DE/RPG Problem Determination Procedures for the Programmer

Program Number 5708-DE1

SC21-7852-1
S5280-28

IBM 5280 Distributed Data System

DE/RPG Problem Determination Procedures for the Programmer

Program Number 5708-DE1

Second Edition (June 1981)

This is a major revision of, and obsoletes, SC21-7852-0 and incorporates SN21-8196. Because the changes and additions are extensive, this publication should be reviewed in its entirety.

This edition applies to release 3, version 1, modification 0 of the IBM 5280 System DE/RPG (Program Product 5708-DE1) and to all subsequent versions and modifications. Changes are periodically made to the information herein; these changes will be reported in technical newsletters or in new editions of this publication.

Use this publication only for the purposes stated in the *Preface.*

It is possible that this material might contain reference to, or information about, programming, or services that are not announced in your country. Such references or information must not be construed to mean that IBM intends to announce such IBM products, programming, or services in your country.

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

This publication could contain technical inaccuracies or typographical errors. Use the Reader's Comment Form at the back of this publication to make comments about this publication. If the form has been removed, address your comments to IBM Corporation, Product Information Development, Department 997, Austin, Texas 78758. IBM may use and distribute any of the information you supply in any way it believes appropriate without incurring any obligation whatever. You may, of course, continue to use the information you supply.

This manual is intended to help the programmer and the service representative identify and isolate compiler problems or user program problems and to guide them to the appropriate action once the problem is determined.

This manual is divided into three chapters:

Chapter 1. Problem Determination identifies conditions that result in a compiler problem or a user program problem. This section identifies the problem and guides the service representative or programmer to the appropriate action. Also included is a list of information to send to IBM when a problem is reported.

Chapter 2. Debugging DE/RPG Programs describes the functions of the compiler and the organization, content, and logic flow of the object programs that are generated by the compiler. This section is intended mainly for the service representative, although the programmer can also use it to circumvent or investigate compiler problems.

Chapter 3. Debugging Options describes what is available on the system to aid in problem determination. Examples show how to interpret and analyze dumps.

RELATED PUBLICATIONS

- *IBM 5280 DE/RPG Reference Manual,* SC21-7787
- *IBM 5280 Message Manual,* GA21-9354
- *IBM 5280 Utilities Reference/Operation Manual,* SC21-7788
- *IBM 5280 System Control Programming Reference/Operation Manual,* GC21-7824
- *IBM 5280 Operator's Guide,* GA21-9364
- *IBM 5280 Functions Reference Manual,* GA21-9353
- *IBM 5280 Data Areas and Diagnostic Aids Handbook,* SY31-0595
- *IBM 5280 Communications Utilities Reference Manual,* SC34-0247
- *IBM 5280-3270 Emulation Reference Manual,* SC34-0384

PREFACE iii
Related Publications iii

CHAPTER 1. PROBLEM DETERMINATION **1**
Identifying DE/RPG Problems 1
Reporting DE/RPG Problems 3

CHAPTER 2. DEBUGGING DE/RPG PROGRAMS **5**
Compiler 5
Compiler Phases 5
Compiler Communications Area (CCA) 7
Shared Routines 7
Compiler Work Files 8
Compiler Error Handling 8
Compiler Module Descriptions 8
Object Program Organization 13
Literals/Prompts Table 17
Named Fields 17
Logical File Name Block 17
Dup/Store Table 17
Table Directory 17
User Data Table 17
Screen Format Control String Table 18
Printer and Diskette Edit Format Control Strings and Table 18
Subroutines 18
Return-to-Program Exit Code 21
Record Level Code 22
Calculations Object Code 22
Format Control Table 22
Z-Specification Driver 23
Termination Code 23
Initialization Code 23
File Translation and Alternate Collating Sequence Tables . . . 24
Logical Buffers 24
Physical Buffers 25
I/O Driver 25
Data Areas 25
Register Save Area 25
Partition Control Block 25
Logical Record Buffers 27
Physical Buffers 27
Input Mask Buffer 27
Output Mask Buffer 28
Format Control Table 28
Status Line Buffer Area 29
Job Statistics Counter Area 29
I/O Driver Parameter Block 30
Logical File Name Block 32
Partition Subroutine Stack 33
Example Program Source Listing 34

CHAPTER 3. DEBUGGING AIDS **45**
Compiler Debug Functions 45
Invoking the Compiler Debug Functions 45
Running the Dump and Trace Programs 47

GLOSSARY **49**

If you encounter a problem in compiling or executing a DE/RPG program, the information in this section can help you circumvent or solve the problem. If it does not and you decide to call your service representative to investigate the problem, this section can help you gather the information your service representative needs to solve the problem.

This section includes an *Identifying DE/RPG Problems* and a *Reporting DE/RPG Problems* description. The first description helps identify the type of DE/RPG problem that has occurred and what you can do to circumvent it, if possible. The second description provides information about the DE/RPG problem that you should gather before you call for service.

IDENTIFYING DE/RPG PROBLEMS

When a DE/RPG problem occurs, you can use the following series of questions to identify the cause:

Has the user program previously been compiled or run successfully?

NO YES

Consider what has changed. For example, have operating procedures changed, are new files being used, or have programs changed?

Is the current release of the DE/RPG compiler being used?
(The release number is printed on the first line of the source listing.)

YES NO

In addition to using the current release of the compiler, make sure the latest utilities licensed program diskette (5708-UT1) that applies to the compiler release, is used during system configuration. Add any PTFs (program temporary fixes) and recompile the program.

Have all IBM-supplied programs on the utilities licensed program diskette (5708-UT1) that apply to the current release of the compiler been used during system configuration?

YES NO

Use the current programs and recompile the user program.

Have any non-IBM-supplied modifications been made to the compiler or to the utilities licensed program diskette (5708-UT1)?

NO YES

If the compiler has been changed, use the current release and programs, and recompile the program. If the utilities licensed program diskette has been changed, use the current release and programs for system configuration.

Did the DE/RPG compiler terminate abnormally?

NO YES

Two conditions can occur whenever the compiler terminates abnormally:

One condition displays a system error, number 9999, along with the two-character module ID displayed on the status line.

The other condition displays DE/RPG and the two-character module ID.

In either case, record the module ID, address, and partition number displayed and report the condition to the service representative. In addition, record the size of the partition used.

Refer to *Reporting DE/RPG Problems* later in this section for a description of the information that you should gather before calling for service.

Did the DE/RPG compiler get in a loop while compiling a user program? (A loop is a set of instructions that executes repeatedly while a certain condition exists.)

NO YES

Report this condition to the service representative.

Refer to *Reporting DE/RPG Problems* later in this chapter for a description of the information that you should gather before calling for service.

Did the DE/RPG compiler generate any unexpected messages or errors?

NO YES

Review all warning messages and/or errors to ensure that they are not unexpected. Go to the source statement that produced the error and correct it. Recompile the program.

Error code descriptions are in the *DE/RPG Reference Manual.* If you cannot isolate the problem, refer to *Reporting DE/RPG Problems* described in this section, gather the information described there, and call for service.

Note: For service representatives, see *Chapter 3. Debugging Aids, Running the Dump and Trace Programs.*

Did the DE/RPG user program get in a loop during its execution, produce incorrect output, or unexpected messages at execution time? (A loop is a set of instructions that executes repeatedly while a certain condition exists.)

NO YES

If you cannot solve or circumvent the problem, refer to *Reporting DE/RPG Problems* later in this chapter, gather the information described there, and call for service.

Refer to *Reporting DE/RPG Problems* later in this chapter, gather the information described there, and call for service.

REPORTING DE/RPG PROBLEMS

When you identify a DE/RPG problem, refer it to IBM for service.

Gather the following information to help IBM personnel solve the problem.

Compile Time Error

Information to Gather	*How to Obtain It*
Original source program	SYSCOPY
Source program listing	SYSPRINT or a copy of the listing
Compile-time dump	Absolute dump at the time of failure
PTF log number of all changes made to the compiler and/or to the utilities licensed program diskette (5708-UT1)	SYSPTF-Patch Program (option 5, system history)
Level of the compiler	Source program listing
Level of the utilities licensed program diskette (5708-UT1)	Adhesive label on the diskette
Engineering change level of the machine	Displayed on the screen during IPL

Execution Time Error

Information to Gather	*How to Obtain It*
Original source program	SYSCOPY
User files required by the DE/RPG program	SYSCOPY
Execution-time dump	Absolute dump taken at the time of failure
DE/RPG object program	SYSCOPY
Description of processing environment	
Printed program output	
PTF log number of all changes made to the compiler and/or to the utilities licensed program diskette (5708-UT1)	SYSPTF-Patch Program (option 5, system history)
Level of the compiler	Source program listing
Level of the utilities licensed program diskette (5708-UT1)	Adhesive label on the diskette
Engineering change level of the machine	Displayed on the screen during IPL

For further information on the SYSCOPY utility, refer to the *Utilities Reference/ Operation Manual.*

For further information on the patch program, refer to the *System Control Programming Reference/Operation Manual.*

Chapter 2. Debugging DE/RPG Programs

The information contained in this chapter requires that the reader have a knowledge of the DE/RPG compiler. This information is intended mainly for the service representative; however, an experienced programmer can also investigate DE/RPG problems with it, before or instead of calling for service. Information about the compiler, the object program organization, and the data areas are described here.

COMPILER

The following information about the compiler is described here:

- Phases
- Work files
- Error handling
- Module descriptions

Compiler Phases

The compiler consists of the following six major phases necessary to compile a DE/RPG source program into an executable object program:

1. Enter
2. Diagnostic
3. Assign
4. Preassemble
5. Assemble
6. Object

As each module of each phase is brought into the compiler overlay area, the previous module is overlaid. The following chart (Figure 1) shows the sequence of execution and a summary of each phase. Figure 2 shows the layout of the compile-time partition.

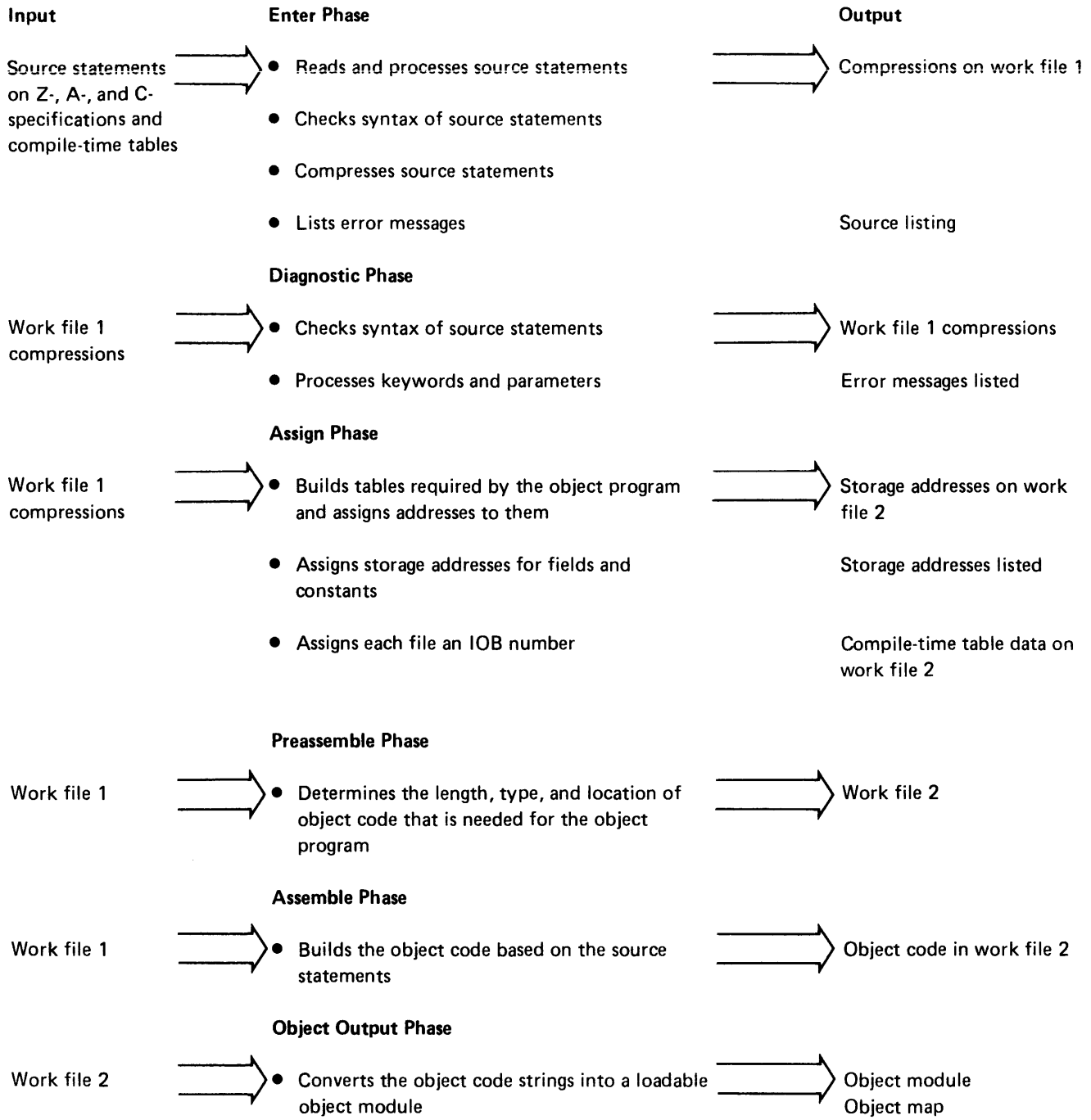


Figure 1. Execution Sequence of Compiler Phases

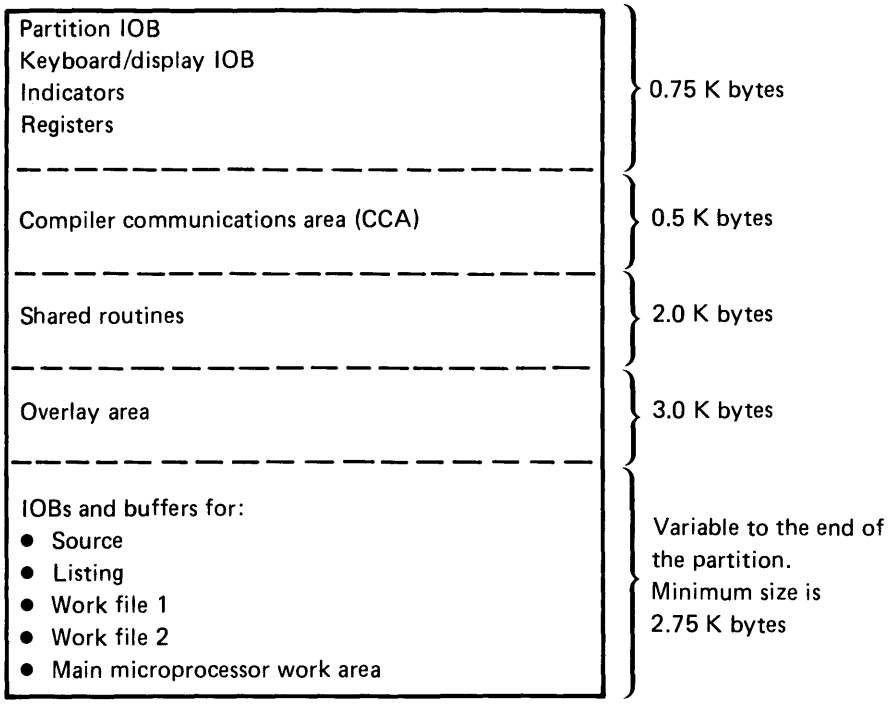


Figure 2. Compile-Time Partition Layout

Compiler Communications Area (CCA)

The CCA (compiler communications area) provides an area for information to be stored and passed from module to module during compilation.

The contents of the CCA are:

- Tables that point to each of the shared routines
- Compiler data set information
- Starting addresses of each compression type
- Selected information that pertains to the source program and the object program

Shared Routines

Shared routines are used by several modules of each compiler phase. SYSDERPG loads shared routines, which remain in the compile-time partition during most of the compilation.

Compiler Work Files

The compiler uses two work files. The first work file contains compressions of source statements (the compiler version of the source statements) from the Z-, A-, and C-specifications, and compile-time tables. Compressions are used by the compiler to determine the object code that is needed.

The second work file contains the object code that the compiler generates. During the object phase, the object code is moved from one work file to the other if the work file requires sorting; then the work file is written to the object file.

Compiler Error Handling

The compiler handles source errors during the enter and diagnostic phases.

The enter phase lists the error on the source listing immediately following the source statement that contains the error. Indicators record errors during the enter phase while the compressions are created from the source statements so that the error condition can be stored. Error compressions are updated with error numbers from the indicators that were previously set.

When the diagnostic phase completes, the error in the error compression is sorted and listed along with the severity of the error and the error message text.

Compiler Module Descriptions

The following text is a summary of each phase and the modules within each phase. The module identification name is found in the first 8 bytes of the partition IOB as the module executes its function. The module is overlaid with the next module called. (For a detailed description of the partition IOB, refer to the *Functions Reference Manual*.)

The compiler is stored on a diskette in two files; SYSDERPG and SYSCMPO. SYSCMPO is a partitioned data set and contains all modules described here except SYSDERPG.

Enter Phase

The modules in the enter phase perform initial processing of a DE/RPG source program. During this phase the source program is read and checked for proper syntax. The syntax check is not a complete check, further checks are done in modules of the diagnostic phases.

If a listing is requested, the source program is listed along with error messages for errors detected, and both are printed or written to a diskette data set.

Source statements are reformatted as compressions and written to a compiler work file for other phases to use.

The enter phase modules are listed in the following chart:

Module Identification	Partition IOB Program Name	Function
SYSDERPG	SYSDERPG	– Determines the device the compiler is loaded from. – Initializes the compiler work area, shared control routines, IOBs, and buffers. – Prompts for data set names, device IDs, and listing information. – Opens data sets. – Creates error compressions.
RGAC	DERPG AC	– Checks Z-specification source statements for errors. – Reads and compresses Z-specification source statements. – Writes these compressions to diskette.
RGAG	DERPG AG	– Checks A-specification source statements for errors. – Reads and compresses A-specification source statements. – Writes these compressions to diskette.
RGAH	DERPG AH	– Checks C-specification source statements for errors. – Reads and compresses C-specification source statements. – Writes these compressions to diskette.
RGAI	DERPG AI	– Checks compile-time data tables, file translation and alternate collating sequence table source statements for errors. – Reads and compresses these source statements. – Writes these compressions to diskette.
RGAJ	DERPG AJ	– Processes self-check source records. – Reads and compresses these source statements. – Writes these compressions to diskette.
RGEA	DERPG EA	– Loads and initializes shared routines for the remainder of the compile.

Diagnostic Phase

These modules process the compressions from the work file and provide further syntax checking for errors not found in the enter phase.

Keywords are prepared for the next phase (assign phase) by reordering and moving their parameters to parameter lists and reformatting the parameters.

The diagnostic phase modules are as follows:

Module Identification	Partition IOB Program Name	Function
RGEC	DERPG EC	– Checks syntax of keywords on the A-specifications.
RGED	DERPG ED	– Checks syntax of keywords on the A-specifications.
RGEF	DERPG EF	– Checks keyword compatibility with the A-specifications.
RGEH	DERPG EH	– Checks keyword compatibility and syntax on the A-specifications.
RGEJ	DERPG EJ	– Checks keyword parameter movement, syntax, and reserved word handling on the A-specifications.
RGEO	DERPG EO	– Checks job line keywords on the A-specifications.
RGEP	DERPG EP	– Checks entry and review lines on the Z-specifications.
RGEQ	DERPG EQ	
RGER	DERPG ER	
RGES	DERPG ES	– Checks C-specification fields.
RGFE	DERPG FE	– Checks keyword references on the A-specifications.
RGFG	DERPG FG	– Checks field positions and record lengths on the A-specifications.
RGFM	DERPG FM	– Sorts and prints line numbers and associated error message numbers.
RGFL	DERPG FL	– Sorts and prints error message text.
RGFN	DERPG FN	– Contains the text for error messages used by DERPG FL.
RGFS	DERPG FS	– Checks C-specification references.
RGFX	DERPG FX	– Generates self-check table and checks keyword specifications for self-check specifications.
RGFZ	DERPG FZ	– Checks compile-time tables.

Assign Phase

The modules in the assign phase assign the object program information (such as constants, data fields, and tables) to storage addresses in the object program and assigns IOB numbers to files.

The assign phase modules are listed in the following chart:

Module Identification	Partition IOB Program Name	Function
RGIB	DERPG IB	– Assigns literals to storage addresses.
RGIC RGIE	DERPG IC DERPG IE	– Assigns data areas to storage addresses.
RGIG	DERPG IG	– Assigns literals to storage addresses.
RGII	DERPG II	– Builds literal/prompt and AUXDUP tables, and table dope vectors.
RGIJ RGIK RGIL	DERPG IJ DERPG IK DERPG IL	– Assigns file IOB numbers and accumulates file and record usage information. Builds logical file name block.
RGIM	DERPG IM	– Assigns tables to storage addresses.
RGIO	DERPG IO	– Builds compile-time data tables.
RGIS	DERPG IS	– Assigns block numbers and label numbers.
RGIZ	DERPG IZ	– Checks for undefined names in the source program.

Preassemble and Assemble Phases

Preassemble modules determine the type, length, and location of the object code that is to be created by the assemble phase.

The assemble phase builds the object code that the preassemble phase has determined from the source program, and places that code in the object program.

The preassemble and assemble modules are listed in the following chart:

Module Identification	Partition IOB Program Name	Function
RGMA RGMB RGMC RGMD RGME RGMF	DERPG MA DERPG MB DERPG MC DERPG MD DERPG ME DERPG MF	– Builds screen format control strings for all records defined for the CRT file.

Module Identification	Partition IOB Program Name	Function
RGMG	DERPG MG	
RGMH	DERPG MH	
RGMI	DERPG MI	
RGNA	DERPGNA	– Builds printer and diskette format control strings.
RGNZ	DERPG NZ	– Builds a table of addresses for screen format and diskette/printer format control strings.
RGPA	DERPG PA	– Determines which subroutines are needed by the object program and includes them.
RGPC	DERPG PC	
RGQA	DERPG QA	– Builds the return-to-program exit code from the screen format control string.
RGQB	DERPG QB	
RGQC	DERPG QC	
RGQD	DERPG QD	
RGQE	DERPG QE	
RGQF	DERPG QF	
RGQG	DERPG QG	
RGQR	DERPG QR	– Builds the calculations entry to link the return-to-program exit code to calculations.
RGRA	DERPG RA	– Builds record level I/O control strings.
RGUA	DERPG UA	– Builds calculations code.
RGUB	DERPG UB	
RGVA	DERPG VA	
RG VB	DERPG VB	
RG VC	DERPG VC	
RGVD	DERPG VD	
RGVE	DERPG VE	
RGYA	DERPG YA	Builds format control table and partition control block.
RGYB	DERPG YB	Builds object code to link the following:
RGYD	DERPG YD	– Data entry driver and Z-specification driver.
RGYF	DERPG YF	– Job initialization and job termination.
RGYG	DERPG YG	– IOBs, buffers, file translate tables, and alternate
RGYH	DERPG YH	collating sequence tables for diskette, keyboard,
RGYI	DERPG YI	printer, and communications data sets.
RGYJ	DERPG YJ	

Object Phase

The object phase assembles the object code into an executable object module.

The object phase modules are listed in the following chart:

Module Identification	Partition IOB Program Name	Function
RGYZ	DERPG YZ	— Prints object program map and initializes partition IOB areas.
RGZC	DERPG ZC	— Writes a load module to a data set.

OBJECT PROGRAM ORGANIZATION

The first part of the partition area contains fixed addresses relative to the start of the partition. The addresses are hex 0000 for the partition IOB, hex 0040 for the logical I/O table, hex 0080 for the keyboard display IOB, and hex 0100 for indicators and registers.

For a description of the partition IOB, logical I/O table, keyboard/display IOB, indicators, and registers, see the *Functions Reference Manual* or the *Data Areas and Diagnostic Aids Handbook.*

Indicators and registers that are assigned for DE/RPG programs are described in the following charts.

Indicator Assignments

Indicator	Meaning when On
I0	File allocate at open time is not allowed
I1-I99	DE/RPG user indicators
I108	Keyboard external status disabled
I110	Foreground/background
I151	Enter (add) mode
I152	Attention/system request
I153	Open allocate
I192-I207	Test bits loaded into BR12 (all 16 indicators used this way)
Indicators for Modes	
I208	Enter mode
I209	Update mode
I210	Verify mode
I211	Rerun mode
I212	Prompt mode
I213	Print or copy process mode
I214	Print key
I215	Copy mode
I216	Search mode
I217	Search relative record
I218	Search content
I219	Search content forward
I220	Auto run, search content first pass mask, and updated statistics
I221	Search parameter error
I222	Record insert mode
I223	Cancel key pressed

Indicators for Transaction Data Set

I224	Beginning of extent
I225	Beginning of extent plus 1
I226	First pass record backspace
I227	First pass record advance
I228	End of data
I229	End of data plus 1
I230	End of extent

Indicators for I/O Functions in the Data Entry Driver

I231	Clear next buffer
I232	Read next record
I233	Read previous record
I234	Write previous record
I235	Write record to extend file

Copy Data Set Indicators

I236	Beginning of extent
I237	Beginning of extent plus 1
I238	First pass record backspace
I239	First pass record advance
I240	End of data
I241	End of data plus 1

Miscellaneous Indicators

I242	Enter was canceled by the common area
I243	Transaction file = 0; copy file = 1
I244	Page forward
I245	Next format
I246	Select format
I247	Record correct
I248	Record changed in record correct
I249	Record marked
I250	Erase function key invoked
I251	I/O complete
I252	External status pending
I253	End of data in verify mode
I254	Last record written (in copy mode); last record verified (in verify mode)

Register Assignments

Register	Meaning
BR10	Field changed indicator
BR11	Field changed indicator
BR12	Used to test bits by indicators I192-I207
BR18	Partition subroutine stack
BR32	Register save area
BR37	Partition control block address
BR38	Key accept address area
BR39	Number of inputs and enter commands address
BR40	Transaction file IOB address
BR41	Print file IOB address
BR42	Copy file IOB address
BR43	Keyboard/display IOB address
BR44	Current read file IOB address
BR45	Address of IOB passed to the open routine
BR46	Previous record buffer address
BR47	Current record buffer address
BR48	Next record buffer address
BR49	Mask output area address
BR50	Search mask area address
BR51	Format control table address
BR52	Current format pointer in format table
BR53	Status line buffer address
BR54	Job statistics counter address
BR55	Current screen format control string format number
BR56	Logical record length
BR57	Previous record number (first half)
BR58	Previous record number (second half)
BR59	Current record number (first half)
BR60	Current record number (second half)
BR61	Next record number (first half)
BR62	Next record number (second half)
BR63	Current read file IOB number
BR64	Number of records to insert
BR73	Format repeat count
BR74	Deleted records count
BR75	Next format ID
BR76	Displacement in the return-to-program exit table for the next exit
BR77	Parameter address from the I/O driver to the I/O management subroutines
BR78	Constant value 4 for return function
BR79	Current named field address

The following list shows how the object program is organized and the sequence in which it is generated. Following this list is a description of each item. (Because the user program determines which items are included in the object program, some of the items listed may not be generated.)

1. Literals and prompts constants
2. Named fields
3. Logical file name block
4. User data tables
5. Literal/prompt and dup/store tables
6. Table directories
7. Screen format control strings and table
8. Printer and diskette edit format control strings and table
9. Subroutines
10. Return-to-program exit code
11. Keyboard record level code
12. Printer and diskette record level code
13. Calculations code
14. Return-to-program exit code table
15. Format control table
16. Z-specification driver routine
17. Termination code
18. Initialization code
19. File translation and alternate collating sequence tables
20. Logical buffers
21. IOBs
22. Physical buffers

Literals/Prompts

The addresses of all literal and prompt constants are shown on the source listing.

Named Fields

The addresses of all named fields are shown on the source listing.

Logical File Name Block

The description of the logical file name block is listed in this chapter under *Data Areas.*

User Data Table

This table contains the data for the users tables.

Literal/Prompt and Dup/Store Tables

The address of the storage duplication table is at hex displacement 36 in the keyboard/display IOB. The index into the table is in the screen format control string. This table contains the address of fields that were named on the A-specification with keywords AUXDUP or AUXST. Access to this table is via the screen format control string.

The address of this table is at hex displacement 0D in the keyboard/display IOB. This address points to the literal/prompt table, which points to the actual literal/prompt. The address of each literal/prompt is also shown on the source listing.

Table Directory

The address of the table directory is in the partition IOB at hex displacement 18. The main microprocessor uses this address to access tables. The contents of the table are:

Hex Displ	Length in Bytes (Hex)	Description
0	2	Starting address of the table
2	2	Address of the last table entry
4	1	Length minus 1 of the table entry
5	1	Length of the bypass (see Note)
6	2	Number of table entries

Note: Bypass is used for alternate tables where the length is between the last byte of a search argument and the first byte of the next search argument.

Screen Format Control String Table

A screen format control string is built for each CRT record defined on the A-specification. These control strings provide the operations to the keyboard/display microprocessor to perform.

The address of the screen format control string table is in the keyboard/display IOB at hex displacement 79. This address points to the first byte of the screen format control string.

The keyboard/display microprocessor uses the keyboard format number in the Enter instruction as an index into the table.

Printer and Diskette Edit Format Control Strings and Table

Edit format control strings are generated for printer, diskette, and communications files referred to on the C-specification. The address of the format control string table is in the partition IOB at hex displacement 24.

Subroutines

Subroutines control I/O operations specified on the C-specification. Basically the I/O driver and I/O management routines control and perform I/O operations.

The I/O driver provides the link from the object code to the I/O management routines, which in turn provide the device microprocessors with the necessary information to move data to and from buffers in the object program.

The object program requests an I/O operation via a CALL instruction, which is followed by the 8-byte control block. The I/O driver calculates the IOB address using the logical I/O table, and uses the logical file name block to get the address of the I/O management routine.

The I/O management routine uses the I/O control code to determine the function to be performed by the device microprocessor. The I/O function is passed to the device microprocessor in an instruction via the device IOB.

The device microprocessor moves data to or from logical buffers and to or from physical buffers depending on the I/O operation.

The source listing contains the address of each included subroutine.

The following text lists all the possible subroutines used in a DE/RPG program and a brief description of each.

RG01 Keyboard External Status Processor

The keyboard external status subroutine processes external status conditions from the keyboard for the object program.

RG03 Keyboard/CRT I/O Management

The keyboard/CRT I/O management subroutine provides the interface to the keyboard/display device from calculations and from formats which specify the WRITE(*NO) keyword.

RG04 Magnetic Stripe Reader I/O Management

The magnetic stripe reader I/O management processes open, close, and read operations for the magnetic stripe reader.

RG20 Printer External Status Processor

The printer external status processor displays and retries error conditions for the printer, when appropriate, and calls the diskette external status subroutine for diskette errors when the printer output has been redirected to diskette. Other printer I/O errors are posted in the appropriate logical file block.

RG22 Printer I/O Management

The printer I/O management subroutine performs printer data set operations such as open, close, write a record, and control space and skip.

RG30 Diskette External Status

The diskette external status routine provides diskette error recovery during run time for files controlled by calculations.

RG31 Diskette External Status Processor

This diskette external status routine transfers control to the common area to process diskette external status conditions for data entry files (transaction files and copy files).

RG32 Diskette I/O Management

The diskette I/O management subroutine processes requests to diskette data sets, such as open and close. It updates, writes, reads, chains, and deletes records, as well as sets record limits.

RG33 Diskette I/O Management

The diskette I/O management provides diskette I/O operations for data sets that are keyed and indexed.

RG34 Diskette I/O Management

The diskette I/O management provides diskette I/O operations for multivolume data sets.

RG35 Diskette I/O Management

The diskette I/O management provides diskette I/O operations for data sets that are keyed, indexed, and multivolume.

RG36 Diskette I/O Management

The diskette I/O management provides diskette I/O operations for the transaction file.

RG40 Communications External Status Processor

The communications external status routine processes external status codes for SNA and BSC.

RG42 Communications Management

The communications I/O management subroutine performs communications data set operations such as open, close, write a record, read a record, and FEOD (force end of data).

RG47 IBM 3270 Emulation Communications Management

The IBM 3270 Emulation communications management subroutine performs communications data set operations such as open, close, write a record, and read a record.

RG50 and RG51 I/O Driver

The I/O driver provides the interface from the object code to the I/O management routines. RG50 is provided when no calculation controlled files are used in the program. RG51 is provided when a control string is passed in the parameter block.

RG80 Verify Error Display

The verify error display subroutine provides the interface to the common area to display error codes while in verify mode.

RG81 Error Display

The error display subroutine displays error codes (with or without help text) and marks the current data field and current record when the automark function is active.

RG82 Packed/Binary Data Conversion

The packed/binary data conversion subroutine performs data conversion for packed and/or binary fields.

RG83 Alternate Collating Sequence

The alternate collating sequence subroutine compares two alphameric characters for the collating sequence by using either the ASCII translate table or a user defined table.

RG84 Calculations Extended Precision Arithmetic

The calculations extended precision arithmetic subroutine processes all extended arithmetic for add, subtract, multiply, and divide operations.

RG85 Resolve Table Element Address

The resolve table element address subroutine computes the address of table elements by using the table directory and index numbers.

RG86 Physical Buffer Allocation

The physical buffer allocation subroutine provides buffer space for devices that require buffers.

RG99 File Close

The file close subroutine handles job termination by closing all DE/RPG files and calling CFA.

Return-to-Program Exit Code

This code handles extended edits for keyboard/display operations that cannot be handled by the screen format control strings. All named fields require exit code and the following keywords require exit code:

- ADD
- AUXST
- CHECK (BY BV Gxx Mxx)
- COMP
- ERROR
- EXSR
- INSERT

- LOOK
- RANGE
- RANGET
- RESET
- SETOF
- SETON
- SEQ
- SUB
- SUBST
- TADD
- TSUB
- XCHK

The return-to-program exit table address is in the partition control block. The partition control block address is in BR37.

Record Level Code

Keyboard

This code provides the interface between calculations code and keyboard operations. This code sets up the return-to-program exit code, initializes cursor positions, and clears the display line.

Printer/Diskette Record Level Code

This code provides the read and write operations specified on the A-specification. This code performs functions such as forms control (printer), record identification, interface to access methods on diskette, and formatting data.

Calculations Object Code

This code performs the operations that are specified on the C-specifications.

Format Control Table

The description of the format control table is in this chapter under *Data Areas.*

Z-Specification Driver

The Z-specification driver is generated from the Z-specification compressions. It interfaces with either the data entry driver in the common area when the transaction file is specified or the keyboard/CRT I/O management subroutine when a WRITE(*NO) is specified.

The Z-specification driver includes the format control table. The Z-specification driver passes a pointer to the format control table entry for the current format to the data entry driver. The Z-specification driver also clears the screen, sets the cursor, and controls format chaining.

The object code generated by the Z-specification driver controls the repeat count, format sequence, write function of the TFILE, production statistics, and calls the data entry driver or keyboard/CRT I/O management subroutine.

The address of the Z-specification driver is on the source listing. The contents of the format control table is in this chapter under *Data Areas.*

Termination Code

A DE/RPG program is terminated when the End of Job key is pressed or the EOJ keyword is encountered. Termination code, then performs the following functions:

- Closes files
- Update station statistics
- Goes to end of job
- Write job statistics
- Chains to the next job (EOJ keyword only)

These functions are primarily handled in the common area.

Initialization Code

Initialization code prepares the object program for execution. The main microprocessor starts executing initialization code when the load prompt response is executed. The address of the code is in the partition IOB at hex displacement 10 and 11 (instruction address). The first executable instruction is also on the listing.

Initialization code performs the following functions:

- Checks indicator 0 for an active request to dump or trace indicator
- Checks for the data entry driver routine in the common function area
- Initializes the stack pointer
- Attaches the keyboard and clears the screen if TFILE is present
- Initializes user fields
- Reads data tables
- Opens all files (except the copy file, print file, and communications file) pertaining to transaction files (calls I/O driver)
- Prompts for mode selections
- Initializes the station statistics
- Gets the system date from the system control block
- Calls Z-specification driver
- Allocates storage buffers for:
  - register save area
  - logical buffers for non-data-entry files
  - job statistics
  - status line
  - search mask areas

File Translation and Alternate Collating Sequence Tables

The file IOB contains the file translation table number that points to the table directory. The table directory contains the address of the file translation table.

The address of the alternate collating sequence table is in the partition control block.

Logical Buffers

Three logical buffers for records; previous, current, and next are provided for data entry. These buffers must have the same length with minimum size dependent on logical record length.

Two subroutines located in the common area, record advance and record backspace, exchange addresses to these buffers for record advance and record backspace operations.

The new addresses of the previous and current buffers is placed in the keyboard/display IOB.

Physical Buffers

Physical buffers are used for all files except the communications files and magnetic stripe reader files.

Double buffers are needed for transaction files and keyed files with WRITE specified. Separate buffers are needed for copy files and print files if copy and print IOBs are included.

I/O Driver

The I/O driver provides the interface from object code to I/O management routines for the required I/O operations. An 8-byte parameter list is input to the I/O driver for processing the I/O control strings.

The address of the I/O driver is on the source listing.

DATA AREAS

This section describes the data areas and control blocks used by the DE/RPG program and how they can be located in the object program.

Register Save Area

The register save area is a 128-byte area that is used by the common area.

The address of the register save area is in BR32.

Partition Control Block

The partition control block provides the link from a partition to the common area. The address of the partition control block is in BR37. The contents of the partition control block are described in the following chart:

Hex Displ	Length in Bytes (Hex)	Descriptions
0	2	Record length of the logical buffers for data entry
2	2	Record mark position (no mark = hex FFFF)
4	2	Verify record mark position (no mark = hex FFFF)
6	1	File exchange type: Hex 00 = basic 01 = H 02 = I
7	6	Record count for allocating files
0D	2	Display attributes

Hex Displ	Length in Bytes (Hex)	Descriptions
0F	1	Flags: Bit 0-1 = bypass production statistics 0 = prompt for writing production statistics 1-1 = auto mark 2-1 = clear screen 3-1 = write deleted record 4-1 = write record 5-1 = write record and bypass ENTR (data entry) 6-1 = no prompt for file open 7-1 = print file specified in the program
10	1	CFA2 Flags (Set to hex 00)
11	1	Flags: Bit 0-1 = calculations mode 1-1 = EXFMT is active 2-1 = magnetic stripe reader has data available 3-1 = EXFMT detected a function key 4-1 = SUBST determined 5-1 = first SEQ error detected 6-1 = not used 7-1 = keyboard external status enabled
12	1	Flags: Bit 0-1 = processing a format that specified WRITE(*NO) 1-1 = Enter instruction issued for the EXFMT operation has been canceled by the common area for ATTN, SYS REQ, or dump file open processing. 2-1 = Initialized deleted record insert count 3-1 = Force review mode tests for this record 4-1 = Error code returned with data 5-1 = End of Job and file closing has begun 6-1 = Abnormal End of Job 7-1 = Field modified
13	2	Number of records between deleted record insertions for transaction file (WRITE (*NO) = hex 0000)
15	2	Address of the record read by the I/O driver
17	2	Lowest address for buffer allocation
19	2	Lowest address available for buffer allocation
1B	2	Forward return to program address

Hex Displ	Length in Bytes (Hex)	Descriptions
1D	2	Backward return to program address
1F	1	I/O error indicator for EXFMT operations
20	2	Error code for a function key detected during an EXFMT operation
22	2	Absolute storage address of the alternate collating sequence table
24	2	Address of the EXFMT buffer

Logical Record Buffers

Three equal length logical buffers are required by a DE/RPG program for data entry. The minimum length of each buffer is determined by the logical record length. There are no alignment requirements nor need the buffers be adjacent.

The address of each logical buffer for the keyboard is in:

Previous: BR46

Current: BR47

Next: BR48

The address of logical buffers for other files are in the device IOB.

Physical Buffers

All files need physical buffers except the communications file and the magnetic stripe reader file. A double buffer is required for the transaction file and a keyed file with a WRITE specified. Separate buffers are required for the copy file and the print file.

The address of the physical buffer is in the device IOB.

Input Mask Buffer

The input mask buffer is a 78-byte area to store input masks for search content and search sequential content.

The address of the input mask buffer is in BR50.

Output Mask Buffer

The output mask buffer is an 80-byte area that follows the input mask buffer. This area stores the search mask that is passed to microcode for search content and search sequential content.

The complete input and output mask buffer area is also used to allocate data sets and write production statistics.

The last 11 bytes of the output mask buffer area is current record buffer when user inputs are accepted.

The address of this area is in BR49.

Format Control Table

The format control table contains one 10-byte entry for each format defined by a DE/RPG program. This table controls the sequence between formats on an entry format; the record IDs and format selection on a review format.

Format 0 is always the first entry in the table. The address of the format control table is in BR51. The address of the format currently being displayed is in BR52.

The contents of the format control table are described in the following chart:

Hex Displ	Length in Bytes (Hex)	Description
0	1	Bit 0-1 = Last entry in this table 1-1 = WRITE(*NO) specified on Z-specification statement for this format (used by Z-specification driver) 2-1 = Calculations reference (BEGSR specified on C-specification) 3 = Not used 4-7 = Repeat count for the current format
1	2	Format ID
3	1	Keyboard format number
4	2	Displacement into the return-to-program table, or the calculations entry point address
6	2	Address of format test code
8	1	Starting line number on the display
9	1	Clear line count on the display (hex FF for WRITE (*NO))

Status Line Buffer Area

The status line buffer area contains the 21 bytes (bytes 20 through 40) of the status line data. The address of this area is in BR53.

The contents of the status line buffer area is described in the following chart:

Hex Displ	Length in Bytes (Hex)	Description
0	1	Display control
1	5	Record number
6	1	Display control
7	1	Auto dup/skip indicator
8	1	Display control
9	1	Auto record advance indicator
A	1	Display control
B	2	Format ID
D	1	Display control
E	3	Mode of operation
11	1	Display control
12	1	Verify mark
13	1	Display control

Note: All display control bytes are initialized to hex 20.

Job Statistics Counter Area

The job statistics area is a 54-byte reserved area for job counters that are updated at mode select time, record advance time, and end-of-job time by the common area.

The address of the job statistics area is in BR54.

I/O Driver Parameter Block

The I/O driver parameter block follows a CALL instruction to the I/O driver routine.

Hex Displ	Length in Bytes (Hex)	Description
0	2	Address of the I/O control string
2	1	I/O control code
3	1	IOB number
4	1	Not used, set to hex 00
5	1	Not-found indicator
6	1	I/O error indicator
7	1	End-of-file indicator

I/O Control Codes

Code	Hex	Description
Close	01	Close data set
Open	02	Open data set
Exfmt	03	Execute format
Update	04	Update current record
Write	05	Write next record
Read-P	06	Read previous record
Read	07	Read next record
Chain-R	08	Chain to relative record
Chain-K	09	Chain to keyed record
Delete	0A	Delete current record
Delete-R	0B	Delete relative record
Delete-K	0C	Delete keyed record

Code	Hex	Description
Setll-R	0D	Set lower limit of relative control
Setll-K	0E	Set lower limit of keyed record
Wrtno	10	WRITE(*NO) to CRT
Feod	11	Force end of data

I/O Control String Commands

Each record control string begins with a begin-record command and ends with an end-record command. Each file group of control strings is preceded by a begin-file command, followed by an end-file command.

Command Code	Hex	I/O Operation Description
Noop	01	Continue with next command
Begin-file	02	Begin a file operation
End-file	03	End a file operation
Begin-record	04xxxx	Begin a record operation (numeric field table address) refer to the numeric field table
End-record	05	End a record operation
Execute	06	Execute a transfer
Recid	07xxxxxx	Record identifier and position
Screen	08xxxxxx	Screen formats control string number (index into screen format table and index into return-to-program start exit number table)
Format	09xx	Format number (index into format table)
Space	0Axx	Space the number of line
Skip	0Bxx	Skip to line number
Seton	0Cxx	Set indicator on
Setof	0Dxx	Set indicator off

Numeric Field Table

There is one numeric field table for each record in the following format:

Hex Displ	Length in Bytes (Hex)	Description
0	1	Length minus 1 of the field, or hex FF if end of table
1	2	Address of the numeric field

The logical file name block is the interface to the I/O management routine from the I/O driver routine. This block contains one entry for each file defined in the source program. The address of the block is in the data set name address field of the device IOB at hex displacement 14.

Logical File Name Block

Hex Displ	Length in Bytes (Hex)	Description
0	1A	File name (data set name)
1A	4	Device code
1E	E	Owner ID or COMM3270*
2C	2	I/O routine address
2E	2	Logical buffer address
30	2	Error code
32	1	Flags: Hex 80 = I/O error 40 = end of file 20 = record not found 10 = Internal error flag used by I/O management routines 0C = transaction or copy file 08 = transaction file 04 = copy file 02 = random access 01 = physical buffer has been allocated

*The following applies only to COMM3270 files:

Hex Displ	Length in Bytes (Hex)	Description
1E	B	3270 printer control block
29	1	3270 internal to user EBCDIC translation table high address
2A	1	3270 user EBCDIC to internal translation table high address
2B	1	3270 last AID byte sent to host

Hex Displ	Length in Bytes (Hex)	Description
33	1	Hex 80 = continued volume 40 = first volume 20 = read previous record in progress 10 = retry with next volume 08 = current record not valid 04 = record not found 02 = volume end of extent 01 = active volume
34	1	Flags: Hex 80 = current active volume when reading or updating 40 = index file positioned at end-of-data 20 = last volume of an offline multi-volume file (volume number = 99) 10 = user error indicator specified 0F = index file job number
35	1	Count of the volumes or the length of the LOGON parameter
36	2	Either the end of page indicator, LOGON address, or key table entry address

Partition Subroutine Stack

This stack is a system table with a maximum of 128 two-byte entries. Whenever a subroutine CALL instruction is executed, the address of the next sequential instruction is assumed to be the return address, and is stored in the stack. The address of the stack is in BR18.

EXAMPLE PROGRAM SOURCE LISTING

The source listing in the following figure (Figure 3) is from an example program in the *DE/RPG User's Guide.* A dump of this program is shown in Figure 4.

```
          DE/RPG COMPILER   V0M23

          *Source file.     PROG15
          *Object file.     OBJ64
00001     Z*****************************************************************
00002     Z*                                                               *
00003     Z*****************************************************************
00004     ZJ  COMBINA                                          TFILE(BILLMST)
00005     Z X1FIND      1E                          X2         WRITE(*NO)
00006     Z X2TOGETH    1E                          X1         WRITE(BILL)
00007     A           F GET            75          DEVICE(CRT) DSPSIZ(6 80)
00008     A           R FIND
00009     A                                  0002001'CUSTOMER NUMBER'
00010     A             NUMBER          5    I002017CHECK(DR) EXSR(GOGET)
00011     A           R TOGETH
00012     A                                  0001001'CUSTOMER NAME'
00013     A                                  0002001'ADDRESS'
00014     A             CUSTN           30   I001017INSERT(CUSNA)
00015     A             ADRES           30   I002017INSERT(ADDR)
00016     A             ITEMN            6  0I003001PMT(ENTER ITEM NUMBER)
00017     A             PRICE            6  2I004001PMT(ENTER COST)
00018     A           F BILLMST        75          DEVICE(DISK D1)
00019     A           R BILL
00020     A             ITEMN
00021     A             PRICE
00022     A             CUSTN
00023     A             ADRES
00024     A           F CUSMAST        65          DEVICE(DISK D1)
00025     A           R LOOKSE
00026     A             CUSNA           30
00027     A             ADDR            30
00028     A           K NUMBER           5
00029     C             GOGET     BEGSR
00030     C             NUMBER    CHAINLOOKSE               0102
00031     C                       ENDSR
00032
00033
00034
00035

      * ADDR  CONSTANT
      * 02F0  'CUSTOMER NUMBER'
      * 02FF  'CUSTOMER NAME'
      * 030C  'ADDRESS'
      * 0313  'ENTER ITEM NUMBER'
      * 0324  'ENTER COST'
      *
      * ADDR  NAME
      * 032E  NUMBER
      * 0333  CUSTN
      * 0351  ADRES
      * 036F  ITEMN
      * 0375  PRICE
      * 037B  CUSNA
      * 0399  ADDR
      *
      *LINE   ERROR
      *00012  1068
      *00014  1068
      *
      *002 errors appeared in this program
```

Figure 3 (Part 1 of 2). Source Listing

```
          DE/RPG COMPILER   V0M23

     *
     *ERROR MESSAGE TEXT
DE W*1068 CRT field is in the prompt line
     *



     *      OBJECT PROGRAM MAP
     *ROUTINE ENTRY POINTS
     *EP      RTN    DESCRIPTION
     *0670    RG99 - End of job processor
     *06C4    RG80 - Verify mode error display
     *0784    RG86 - Physical buffer allocation
     *0850    RG01 - Keyboard external status routine
     *0A7C    RG03 - KB/CRT I/O management routine
     *0BE8    RG30 - Diskette external status routine
     *0D54    RG33 - Diskette I/O management routine
     *17B4    RG51 - I/O driver - full function
     *
     *1DD4    Z-spec driver entry point
     *1F6C    Program entry point
     * 9,472  Is the program length.
```

Figure 3 (Part 2 of 2). Source Listing

Data Areas in the Dump Example		How to Find Data Areas in the Dump Example
1	Partition IOB pointers	System control block hex 00000
2	Diskette IOB pointers	System control block hex 00040
3	Printer IOB	System control block hex 00080
4	Partition IOB	Relative address hex 0000
5	Logical I/O table	Relative address hex 0040
6	Keyboard/display IOB	Relative address hex 0080
7	Indicators (I0–I255)	Relative address hex 0100
8	Binary registers (BR0–BR127)	Relative address hex 0100
9	Decimal registers (R0–R30)	Relative address hex 0100
10	Literals/prompts	Literals/prompts table and the source listing
11	Named fields	Source listing
12	Logical file name block for the TFILE IOB	TFILE IOB hex displacement 14
13	Logical file name block for the CUSMAST IOB	CUSMAST IOB hex displacement 14
14	Keyed index file table	Table directory
15	Literals/prompts table	Keyboard/display IOB hex displacement 0D
16	Dup/store table	Keyboard/display IOB hex displacement 36
17	Table directory	Partition IOB hex displacement 18
18	Logical buffer next	BR48
19	Logical buffer previous	BR46
20	Logical buffer current	BR47
21	EXFMT buffer (a work area used when EXFMT and WRITE(*NO) are specified)	Partition control block hex displacement 24
22	Dup/store area (used for INSERT)	Dup/store table
23	Screen format control string (keyboard)	Screen format control string table
24	Edit format control string (diskette)	Edit format control string table
25	Screen format control string table	Keyboard/display IOB hex displacement 79

Data Areas in the Dump Example	How to Find Data Areas in the Dump Example
26 Edit format control string table	Partition IOB hex displacement 24
27 RG99 End-of-job processor subroutine	Source listing
28 RG80 Verify mode error display subroutine	Source listing
29 RG86 Physical buffer allocation subroutine	Source listing
30 RG01 Keyboard external status subroutine	Source listing
31 RG03 KB/CRT I/O management subroutine	Source listing
32 RG30 Diskette external status subroutine	Source listing
33 RG33 Diskette I/O management subroutine	Source listing
34 RG51 I/O driver—full function subroutine	Source listing
35 Return to program exit code	Return to program exit code table
36 Return to program exit code table	Partition control block (BR37) hex displacement 1B
37 Z-specification driver format control table	BR51
38 Partition control block	BR37
39 Z-specification driver	Source listing
40 Review tests	Z-specification driver format control table hex displacement 06 into each 10 byte table entry
41 Initialization code (program entry point)	Source listing
42 Status line buffer	BR53
43 Logical buffer for CUSMAST	CUSMAST IOB hex displacement 0C
44 Register save area	BR32
45 Input mask buffer	BR50
46 Output mask buffer	BR49
47 Job statistics counter area	BR54
48 TFILE IOB	BR40 and logical I/O table
49 CUSMAST IOB	Logical I/O table
50 Physical buffer 1 for TFILE	TFILE IOB hex displacement 18

Data Areas in the Dump Example	How to Find Data Areas in the Dump Example
51 Physical buffer 2 for TFILE	TFILE IOB hex displacement 20
52 Physical buffer for CUSMAST	CUSMAST IOB hex displacement 18
53 Partition subroutine stack	BR18
54 Main microprocessor work area	Last 256 bytes of the partition

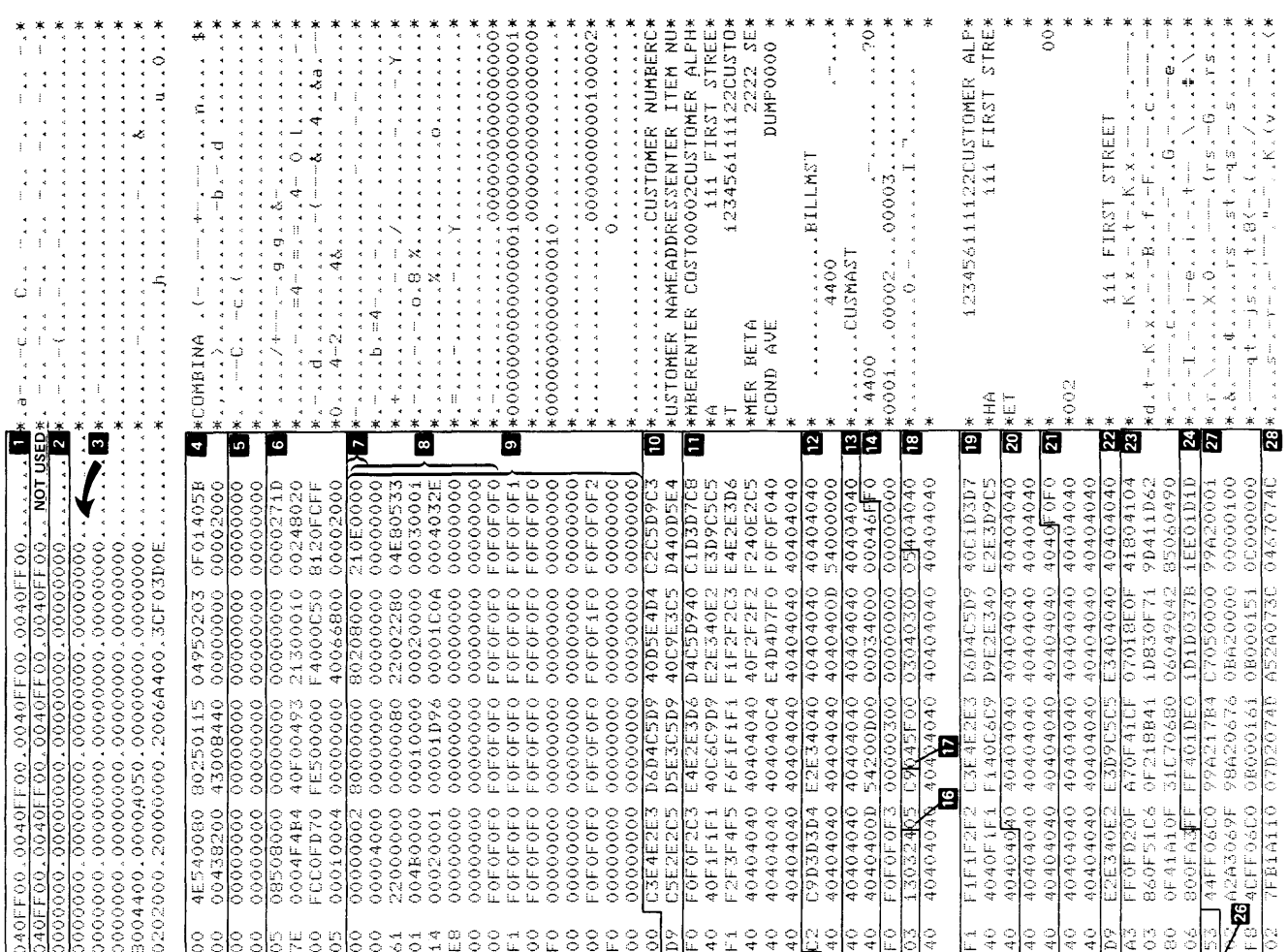


Figure 4 (Part 1 of 6). Example Program Dump

Figure 4 (Part 2 of 6). Example Program Dump

```
049E0 06E0 | 46B124D6 994201BC 0B000121 00000670  99AA01AC 97A80000 C709AAA8 6AAC210A  28
04A00 0700 | B501565C 00000720 69AC02F5 69AD01F4  C709AAA8 A19701AC 6AAC4405 000006E0
04A20 0720 | 6AAC2303 000006E0 A52A0740 0000074C  A52A0744 40A4FFFF 0000074C AE00A426
04A40 0740 | AC00A426 AC0026A4 AC009EA4 039E0754  A52A0748 99A00764 7FB1A010 7FA1A000
04A60 0760 | 0C000000 1CAC0399 05C9057E 01040000  00000000 00000000 00000000 00000000
04A80 0780 | 00000000 A2A0083C 99A0083C 7DA1A102  B8A02C14 B501A032 00000828 98A90008  29
04AA0 07A0 | A11701A8 93A80080 B8AA2C18 9BAA007F  A11601AA 03AA07CD B8AE4A17 91AE007F
04AC0 07C0 | 9BAEFF80 B9022C78 000007E0 B8AE2C18  9BAEFF80 B8A02C14 B5802C13 B101A032
04AE0 07E0 | 92A801AE B8AD2C78 9CAE01AC A11601AC  6DACA80D A3AE2C18 B7202C13 00000810
04B00 0800 | 92A801AC 6DACA808 90AE01AC A3AE2C20  9BAEFF80 90AE01AC A3AE4A17 99A0083C
04B20 0820 | 7FA1A000 0C000000 99A09905 A3A02C04  0B000119 00000670 00000670 00000002
04B40 0840 | 00890001 22800002 04270000 00000000  BB02560F 00000864 BB375626 00000864  30
04B60 0860 | 00000670 B5014A11 00000A0C A2A00A68  99A00A68 7DA1A102 B8265628 B828562A
04B80 0880 | B82A5641 9B2A0F00 A113012A B5085644  9D2A0800 BC2A5642 B8A1560F 9BA0000F
04BA0 08A0 | 6AA00445 6AA0054F 6AA00B56 B5804A11  000008D8 B5804A12 000008D8 0B000109
04BC0 08C0 | 99A00A68 7FA1A000 B3014A11 079E08D5  CD010000 0C010000 0BA10A14 000008C0
04BE0 08E0 | B8A15627 69A13303 69A03F02 99A20A02  0B00094C 0C000000 B8A15626 69A13103
04C00 0900 | 69A04802 99A20A11 0B00094C 0C000000  B8A15626 69A13103 69A03D02 99A20A29
04C20 0920 | 0B00094C 0C000000 B7804A12 00000934  B1104A12 99A00000 99A20A34 0B00094C
04C40 0940 | 0C000000 0B000109 0C000000 90A001A2  B8A1A000 68A0FF03 B1404A12 0B00015D
04C60 0960 | 000009B4 03A00988 B7804A12 00000990  03500989 40D0FFFF 984C0164 9970004B
04C80 0980 | B85E4A24 42A18000 0B000109 000009B4  B7404A11 000009B4 B5FF4A1F 000009B4
04CA0 09A0 | 9BA0FF7F 91A09500 A3A04A20 B1104A11  C7050000 0C000000 B8A04A1B 90A00198
04CC0 09C0 | B8A0A000 03A009DC 999E0000 B1014A11  0C0109D5 B720563F 0BA00000 91980002
04CE0 09E0 | 000008C0 B8A04A1D 93980002 90A00198  B8A0A000 03A00A00 B720563F 0BA00000
04D00 0A00 | 000008C0 B1204A11 000008C0 0C010A11  0C000000 094408E0 08F80910 09B809E4
04D20 0A20 | 090C0928 09440944 09440A04 09440944  09440944 01820000 03048506 008708FF
04D40 0A40 | FF091011 12131415 00000016 17180000  00040000 00008500 08820000 00192021
04D60 0A60 | 222300FF 2400A500 057E004B 00FF0001  02010080 053304E8 00000000 A2A00BD4  31
04D80 0A80 | 99A00BD4 7DA1A102 BB109A00 00000AAC  B1044A0F 0B000105 B3044A0F 72A16803
04DA0 0AA0 | 72A19B07 B1804A12 00000AE0 B7804A11  00000BA8 B5404A11 00000BB0 B1404A11
04DC0 0AC0 | 0B00012D 72A16A0F 99A00BE4 72A1A100  B9E76A0F B9406A11 99A40029 C711A400
04DE0 0AE0 | 99A00014 99A20014 CAA26AA0 73A1562C  B8A04A24 A3A0562C A3A0562E 99A2004B
04E00 0B00 | B8AB563E B3F2563E B180563E B8A44A0D  A3A45674 B8A59A04 A5A40BC5 6AA4FF03
04E20 0B20 | 99A60201 73A20001 CAA2A0A7 B8A79A07  6AA6FF0E B8989A08 B8A86808 440101A9
04E40 0B40 | B7804A12 99A80201 C708A800 B3404A12  A5A60BC9 36000000 B5404A12 00000B38
04E60 0B60 | B8A79A04 6AA6FF01 A5A60BCD 73A1572C  B3404A11 B3804A12 B3F2563E 98AA00F2
04E80 0B80 | A5AA0BD0 B8A19A00 6AA01004 99A00BE4  72A1A000 72A16B0F 0B000135 99A00BD4
04EA0 0BA0 | 7FA1A000 0C000000 46A126C2 00000BB4  46A126C3 994201AC 0B000121 00000BC0
04EC0 0BC0 | 00000670 3CA000A2 CF000000 21A00000  B100563E 0000E7F1 020104E8 01040000
04EE0 0BE0 | 00000000 00000000 A2A00D24 99A00D24  7DA1A102 B8A02C04 B8A22C14 99A60D34  32
04F00 0C00 | 99A80D50 B8AAA600 6EA0AA02 91A60002  6DA8A6FC 99A80D34 92A601A8 A19001A6
04F20 0C20 | B70CA232 00000C3C 69A70902 6AA60E01  00000C3C 0B00010D 00000D18 08A70C40
04F40 0C40 | 0CA80CD0 0CE80CE8 0CE80C88 0C600C88  0C880C64 0C6C0C90 0C7C0CF0 0D040000
04F60 0C60 | 00000D04 0B000784 00000D18 0B000784  B101A232 0B00010D 00000D18 B702A232
04F80 0C80 | 00000D18 00000D10 0B00010D 00000D18  B8452C62 A1930144 0B000149 00000D04
04FA0 0CA0 | 0B00010D 00000D18 B740A233 00000CC8  B510A233 00000CC0 B120A232 00000D08
04FC0 0CC0 | B140A232 00000D08 B108A233 00000D18  B580A233 00000CE0 B140A232 00000D08
04FE0 0CE0 | B108A233 00000D18 B120A232 00000D08  B702A232 00000C64 0B000119 00000670
05000 0D00 | 00000C64 B180A232 A3A0A230 00000D18  0B000119 00000670 99A00D24 7FA1A000
05020 0D20 | 0C000000 00000000 00000000 00000000  00000000 34123701 37023703 37063151
05040 0D40 | 32513215 32113730 34303432 37053799  00000000 A2A01776 99A01776 7DA1A102  33
05060 0D60 | 7FB1A110 B8A79A00 B8B39A01 B8B69A02  B8B59A04 A11701B4 B8B8B614 B8BBB834
05080 0D80 | 03BA0D9C 98A001BA A11101A0 91A00040  B8BCA002 9BBCFF80 B8BFDC14 93A60001
050A0 0DA0 | 0BA71758 03BA0DD8 B7F0BE32 00000DD4  B580BE32 B104B833 B720BE32 00000DC8
050C0 0DC0 | B102BE33 B104B833 A3BB9A01 A3BC9A02  00000DF8 B306BE33 B7F0B832 00000DF4
050E0 0DE0 | B580B832 B104B833 B520B832 B106B833  00000DF8 B306B833 99A01776 7FB1A010
05100 0E00 | 7FA1A000 0C000000 03BA0E14 A5BA1720  A5BA1720 A5B21720 A5B21720 0C000000
05120 0E20 | 4100FFFF B501B64C 00000E38 B502B64C  B104B64C 4000FFFF 03BA0EA0 B501BE32
05140 0E40 | 00000E4C 982C01BC 0B000784 99A00000  A3A0BC30 A5BA1754 04760E70 49A0029C
05160 0E60 | 7392BF1A 07000E71 B502BC4C B104BC4C  985A01BC 0B00014D B580BE32 00000F00
05180 0E80 | B701BE32 0B000784 B704BC4C 00000E98  98A801BC 0B000F04 B3FFBE33 B140BE33
051A0 0EA0 | B501B832 00000EB0 982C01B6 0B000784  99A00000 A3A0B630 B708B832 00000EC8
051C0 0EC0 | 0B000101 00000F00 985A01B6 0B00014D  B580B832 00000F00 B701B832 0B000784
051E0 0EE0 | B3FFB833 B104B833 B110B832 B140B833  B704B64C 00000F00 98A801B6 0B000F04
05200 0F00 | 0C000000 B8A0A814 B8A2A036 98A601A2  B0A7A846 91A60001 B8A5A846 91A40004
```

Figure 4 (Part 3 of 6). Example Program Dump

```
05220 0F20 | BB00A600 00000F44 98A801A6 91A80001  AE01A6A8 07670F45 92A201A4 92A601A4  33 |*-.w....-qy.wjy..-.wy.-.-ks.ukw.u*
05240 0F40 | 00000F20 A3A2A036 0C000000 B504B833  00000FD0 03BA0F65 B702B64C 00000FAC     |*....ts-.....-.-....).-.--.-(...-*
05260 0F60 | 00000F6C B8A0BC44 03A00FAC B8A0B60C  B0A0B644 93A00001 99A21796 B8A5B646     |*...%----.-.----.---l-..rs.o-v--*
05280 0F80 | A5A411DC B8A0B60C B8A2B61C B8A59A04  A5A41751 B0A0B644 93A00001 99A21796     |*vu.----.-s-.-v-.vu.----l-..rs.o*
052A0 0FA0 | B8A5B646 A5A416C0 07670FC1 A5B31738  B780B832 B110B832 B104B833 00000FDC     |*-v--vu.(.-.Av-..--.-.-.-.-....-*
052C0 0FC0 | 45A001A2 A5A411DC 99A09711 00000FD4  99A09713 A3A0B830 B180B832 0C000000     |*--.svu.-r-p....Mr-p.t--.---.....*
052E0 0FE0 | 03BA1068 0B0011A4 B8A2BE36 B8A0B60C  B0A0B644 93A00001 B8A5BC46 A5A416C0     |*.-.-...u-s-.--.---l-..-v--vu.(*
05300 1000 | 07651025 98A601BC 91A60067 98A401BC  91A40038 AE02A6A4 07651068 A5BA174C     |*.-..qw.-jw.-qu.-ju..-.wu.-.-v-.(*
05320 1020 | 00001068 A5BA1734 0B001704 B580BE32  00001108 B520BE32 00001064 0B001648     |*...-v-......--.....-.-.....-...-*
05340 1040 | B520B832 00001060 B580B832 00001108  99A09712 A3A0BE30 B180BE32 00001108     |*-.-....---.....r-p.t--.--.....*
05360 1060 | B320B832 B320BE32 0B00110C B580B832  00001108 B302B833 B110B832 07671091     |*-.-.-.-.....--.....-.-.-.-..-.j*
05380 1080 | 99A09715 A3A0B830 B140B832 B310B832  03BA1108 B8AFB638 B8B0B639 B8A2BC0C     |*r-p.t--.- -.-.-..-..--.---.-s-.*
053A0 10A0 | B8A5BC46 99A0001F 92A001A4 69A11001  93A00010 91A00290 A5A411DC 99A001AE     |*-v--r-..k-.u-..l-..j-.-vu.-r-.-*
053C0 10C0 | 90A201A4 91A20001 AC03A2A0 A5BA1740  B580B832 00001108 98A801BC 0B0011E0     |*-s.ujs..-.s-v-. ---.....qy.-...\*
053E0 10E0 | 98A401BC 91A40038 98A601BC 91A6003B  AE02A4A6 07671109 99A09715 A3A0BE30     |*qu.-ju..qw.-jw..-.uw.-..r-p.t--.*
05400 1100 | B140BE32 B310BE32 0C000000 B702B64C  00001180 0B0011A4 B8A2B836 B8A0B60C     |*- -.-.-.......-.-(...-...u-s-.---.*
05420 1120 | B0A0B644 93A00001 B8A5B646 A5A416C0  0765113D 0B00121C 00001164 A5B21734     |*---l-..-v--vu.(.-...........-v-..*
05440 1140 | 0B0016E4 B580B832 0000119C B520B832  00001164 99A09712 A3A0B830 B180B832     |*...U---....--.-.....-r-p.t--.---.*
05460 1160 | 0000119C B320B832 A5B31744 B580B832  0000119C 98A801B6 0B0011E0 00001188     |*...--.-.v-.---.....-qy.-...\...h*
05480 1180 | 0B00121C A5B31738 98A401B6 91A40038  98A601B6 91A6003B AE02A4A6 B104B833     |*....v-..qu.-ju..qw.-jw..-.uw-.-.*
054A0 11A0 | 0C000000 B8A0B60C B8A2B61C B8A59A04  A5A41751 B0A0B644 93A00001 B8A5B646     |*....---.-s-.-v-.vu.----l-..-v--*
054C0 11C0 | 99A2001F 92A201A4 69A31001 93A20010  91A20290 A5A411DC 0C000000 AC00A2A0     |*rs..ks.u-t..ls..js.-vu.-....-.s-*
054E0 11E0 | B8A4A814 B8A2A436 B8A0A80C B0A0A844  93A00001 B8A5A846 A5A416C0 07651205     |*-uy.-su.--y.--y-l-..-vy-vu.(.-..*
05500 1200 | A5A411DC 90A201A4 91A20001 98A001A8  91A00038 AC02A2A0 0C000000 98A601B6     |*vu.--s.ujs..q-.yj-..-.s-....qw.-*
05520 1220 | 91A60067 98A401B6 91A40038 AE02A6A4  07651238 A5B2174C 0C000000 03BA1291     |*jw.-qu.-ju..-.wu.-..v-.(......-.j*
05540 1240 | B502B833 00001280 B110B833 A5B21728  0B0016E4 B310B833 B740B832 000012F8     |*-.-....--.-.v-.....U-.-.- -....8*
05560 1260 | A5B21748 0B0016E4 B580B832 000012F8  B104B833 99A09714 A3A0B830 000012F8     |*v-.-...U---....8-.-.r-p.t--....8*
05580 1280 | 99A09713 A3A0B830 B180B832 000012F8  B502BE33 000012EC B110BE33 A5BA1728     |*r-p.t--.---....8-.-....--.-.v-..*
055A0 12A0 | 0B001704 B310BE33 B7E0BE32 000012D8  B740BE32 000012F8 A5BA1748 0B001704     |*....-.-.-\-.....Q- -....8v-.-....*
055C0 12C0 | B580BE32 000012F8 B104B833 99A09714  A3A0BE30 000012F8 0B001648 B720B832     |*---....8-.-.r-p.t--....8...--.-.*
055E0 12E0 | 000012F8 B320B832 00001298 99A09713  A3A0BE30 B180BE32 0C000000 03BA1331     |*...8-.-....qr-p.t--.---......-..*
05600 1300 | B502B833 00001320 A5B2172C 0B0016E4  B740B832 00001378 B104B833 00001378     |*-.-.....v-.....U- -....--.-....-*
05620 1320 | 99A09713 A3A0B830 B180B832 00001378  B502BE33 0000136C A5BA172C 0B001704     |*r-p.t--.---....--.-....%v-......*
05640 1340 | B7E0BE32 00001358 B740BE32 00001378  B104B833 00001378 0B001648 B720B832     |*-\-....-- -....--.-....-...--.-.*
05660 1360 | 00001378 B320B832 00001330 99A09713  A3A0BE30 B180BE32 0C000000 0B0016C4     |*...--.-.....r-p.t--.---........D*
05680 1380 | BB00B835 0B00171C B102B833 A5B21730  0B0016E4 B740B832 000013A4 B340B832     |*-.-......-.-.v-.....U- -....u- -.*
056A0 13A0 | B120B832 0C000000 03BA13BD B102B833  A5B21734 0B0016E4 000013D0 B102BE33     |*-.-.......-.--.-.v-.....U...}-.-.*
056C0 13C0 | A5BA1734 0B001704 B7E0BE32 0B001648  0C000000 B504B833 000013F0 A5B2173C     |*v-.......-\-....-....-.-....0v-..*
056E0 13E0 | B780B832 B110B832 B104B833 000013FC  99A09713 A3A0B830 B180B832 0C000000     |*---.-.-.-.-....-r-p.t--.---.....*
05700 1400 | 0B0016C4 BB00B835 0B00171C B102B833  A5B21730 0B0016E4 B740B832 0000142C     |*...D-.-......-.-.v-.....U- -.....*
05720 1420 | B340B832 B120B832 00001434 B7E0B832  A5B2173C 0C000000 03BA1455 B102B833     |*- -.-.-.....-\-.v-........-.--.-.*
05740 1440 | A5B21734 0B0016E4 B7E0B832 A5B2173C  00001478 B102BE33 A5BA1734 0B001704     |*v-.....U-\-.v-.....--.-.v-......*
05760 1460 | B7E0BE32 0000146C 00001478 0B001648  B7E0B832 A5B2173C 0C000000 0B0016C4     |*-\-....%...-...--\-.v-.........D*
05780 1480 | B102B833 A5B21730 0B0016E4 B7E0B832  000014B8 B740B832 000014A8 B340B832     |*-.-.v-.....U-\-....-- -....y- -.*
057A0 14A0 | B120B832 000014F4 B720B832 000014F4  B320B832 B110B832 B110B833 A5B21728     |*-.-....4-.-....4-.-.-.-.-.-.v-..*
057C0 14C0 | 0B0016E4 B310B833 B740B832 000014E4  B340B832 A5B21748 0B0016E4 B580B832     |*...U-.-.- -....U- -.v-.-....U---.*
057E0 14E0 | 000014F4 B710B832 B140B832 B104B833  B302B833 0C000000 03BA1585 B102B833     |*...4-.-.- -.-.-.-.-.........-.e-.-.*
05800 1500 | A5B21734 0B0016E4 B7E0B832 00001544  B720B832 00001634 B8A2B836 B8A5B646     |*v-.....U-\-....--.-.......-s-.-v--*
05820 1520 | 99A0001F 92A001A4 69A11001 93A00010  91A00290 A5A416C0 07651634 B320B832     |*r-..k-.u-..l-..j-.-vu.(.-..-.-.*
05840 1540 | B110B832 B110B833 A5B21728 0B0016E4  B310B833 B740B832 00001570 B340B832     |*-.-.-.-.v-.....U-.-.- -....--- -.*
05860 1560 | A5B21748 0B0016E4 B580B832 00001634  B710B832 B140B832 B104B833 B302B833     |*v-.-...U---.....-.- -.-.-.-.-.*
05880 1580 | 00001634 B102BE33 A5BA1734 0B001704  B7E0BE32 000015D0 B720BE32 00001634     |*....-.-.v-.......-\-....}-.-.....*
058A0 15A0 | B8A2BE36 B8A5BC46 99A0001F 92A0C1A4  69A11001 93A00010 91A00290 A5A416C0     |*-s-.-v--r-..k-.u-..l-..j-.-vu.(*
058C0 15C0 | 07651634 B320BE32 B110BE32 000015F0  0B001648 B580B832 00001634 B720B832     |*.-..-.-.-.....0...----.......-.-.*
058E0 15E0 | 000015F0 B320B832 B110BE32 00001624  B110BE33 A5BA1728 0B001704 B310BE33     |*...0-.-.-.-......-.-.v-.......-.-.*
05900 1600 | B7E0BE32 00001624 B740BE32 00001634  B340BE32 A5BA1748 0B001704 B580BE32     |*-\-......- -......- -.v-.-....---.*
05920 1620 | 00001634 B710BE32 B140BE32 B302BE33  B104B833 0C000000 99A09999 A3A0B830     |*....-.-.- -.-.-.-.......r-rrt--.*
05940 1640 | B180B832 0C000000 B8A0BC0C 98A201A0  B0A0BC44 B0A1BC46 B8AFA001 B8B0A002     |*---.....---.qs.-------------.---.*
05960 1660 | BB00B835 0B00171C A5B21730 0B0016E4  B580B832 000016BC B740B832 0000168C     |*-.-.....v-......U---....-- -....-*
05980 1680 | B340B832 B180B832 000016BC B520B832  000016BC 6EA0A209 B8A0B60C B0A0B644     |*- -.---....--.-....->-s.---.-----*
059A0 16A0 | 93A00001 B8A5BC46 A5A416C0 076716BC  99A09711 A3A0B830 B180B832 0C000000     |*l-..-v--vu.(.-.-r-p.t--.---.....*
059C0 16C0 | AE00A0A2 99AE0000 46A2FFFF 67A29203  10A2A210 17B292A2 A7B201AE A79201B0     |*-.-sr-..-s--sk..ss..-ksx-.-xk.-*
059E0 16E0 | 0C000000 2D000000 A5B216F8 A5B216FC  A5B21700 0C000000 251016E4 253016F5     |*.........v-.8v-.-v-..........U...5*
05A00 1700 | 36000000 2D000000 A5BA1718 A5BA16FC  A5BA1700 0C000000 25101704 00001638     |*.........v-..v-.-v-...............*
05A20 1720 | 23500000 2240FF00 2000FF07 2000FF09  2080FFAF 2080FF92 30500008 3540FF08     |*.&... -...-...-..---.---k.&... -.*
05A40 1740 | 3140FF08 31500008 2600FF00 2680FF00  3CA000A2 2B40FFA0 0E080E20 16380F4C     |*. -..&....-..--..-.s. --.......(*
```

Figure 4 (Part 4 of 6). Example Program Dump

Figure 4 (Part 5 of 6). Example Program Dump

Figure 4 (Part 6 of 6). Example Program Dump

COMPILER DEBUG FUNCTIONS

Debug functions are available as an aid for debugging complex DE/RPG problems. These functions of the compiler should be used by service representatives only, normally under the guidance and direction of field support personnel. This description is for reference purposes only.

Debug functions are used during compilation to halt on the load of a module and to dump main storage, compressions, and object text.

A debug function requires an additional 1 K byte in the partition. Ten functions can be specified for each compilation.

Invoking the Compiler Debug Functions

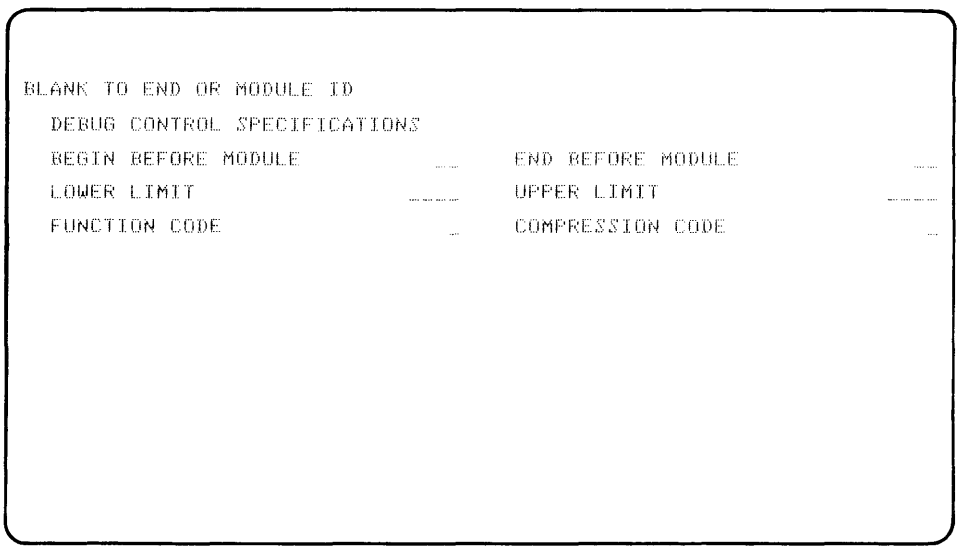
Functions are invoked by responding to prompts for a compilation.

1. When the list options prompt is displayed, select option 7 instead of options 1, 2, 3, or 4.

2. Press the Error Reset key when error code 9400 is displayed. (This redisplays the list options.)

3. Select one of the list options; then select the device for the output.

The following lines are displayed:



Specify the functions you want performed by entering information into the fields. Lower and upper limits define the controls on these functions.

Three prompts are displayed on line 2 as you specify the required information. The first prompt (BLANK TO END or MODULE ID) means that *no entry* ends the debug function; or *enter* the module ID to continue. The module ID must be 'EO' or greater.

The next prompt lists the dump function codes and meanings.

The last prompt lists the compression codes and meanings.

Some fields of the function code have different meanings depending on the function to be performed. The function descriptions that follow contain only the entries that are valid for that function.

Function Code H = HALT

- Begin before module — The compiler halts when the request to load this module is made.
- End before module — The compiler will continue to halt for all module load requests up to and including this module.

Function Code D = Dump Main Storage

- Begin before module — The first dump is taken before the module is loaded.
- End before module — The last dump is taken before this module is loaded.
- Lower limit — The lower limit hex address of main storage. (Defaults to 0.)
- Upper limit — The upper limit hex address of main storage. (Defaults to 0.)

Function Code C = Dump Compressions

- Begin before module — The first dump is taken before the module is loaded.
- End before module — The last dump is taken before this module is loaded.
- Lower limit — The source statement number (in decimal) of the first compression to be dumped. If a value of 0000 is entered, all compressions of the specified type are dumped.
- Upper limit — The source statement number (in decimal) of the last compression to be dumped.

- Compression code – The character that identifies the type of compression to dump:

  0 = Z-compressions
  1 = A-compressions
  2 = C-compressions
  3 = Table compressions
  4 = Error compressions
  5 = Debug compressions
  6 = Module compressions

Function Code T = Dump Text

- Begin before module – Start dumping text at this module.

- End before module – Stop dumping text before this module has loaded.

RUNNING THE DUMP AND TRACE PROGRAMS

When unexpected results occur while executing a user program, use the dump or the trace function to isolate the problem. When the trace function is to be used, you must first use the Patch program in order to reserve IOB space to execute dump/trace. Reserve IOB space by modifying location hex 100 to 80. By doing this, indicator 0 is set on. Otherwise, if indicator 0 is off, the IOB space is used for other purposes.

For a description of how to use the Patch program, refer to the *System Control Programming Reference/Operation Manual.* For a description of how to use the dump and trace functions, refer to the *Functions Reference Manual* or the *Data Areas and Diagnostic Aids Handbook.*

ASCII: American National Standard Code for Information Interchange.

BR: Binary register.

BSC: Binary synchronous communications.

CCA: Compiler communications area.

compiler communications area (CCA): A portion of main storage where information is saved for use by other modules during compilation of a program. Each module can access the information in the CCA and can also pass the information on to the succeeding phase(s).

compressions: The compiler version of source statements contained on a work file that determines the object code required.

CRT: Cathode-ray tube.

DE/RPG: Data Entry with RPG subroutines.

displ: Displacement.

dup: Duplicate.

external status: A condition encountered by an I/O device whenever that device cannot resolve, for example, an error condition or a certain object code instruction. The unresolved condition is brought to the attention of the main microprocessor for it to resolve.

hex: Hexadecimal.

I: Indicator.

ID: Identification.

I/O: Input/output.

IOB: Input/output control block.

IPL: Initial program load.

K: Storage capacity of 1024 bytes.

KB: Keyboard.

microprocessor: A processing unit that is microprogram-controlled and performs internal machine operations.

overlay area: A technique of repeatedly using the same portion of main storage for all compiler phases during the compilation of a program. As each phase of the compiler is brought into the overlay area, the previous phase is overlaid.

PTF: Program temporary fix.

relative addressing: A means of addressing instructions and data areas by designating their location in relation to the location counter or to some symbol.

return-to-program exit code: Object program code that processes advanced edit functions such as self-check digit, range check, and arithmetic checks between fields for keyboard/display operations.

SNA: Systems network architecture.

Index

absolute dump 3
alternate collating sequence subroutine 21
alternate collating sequence table 24
assemble phase 11
assign phase 10

binary register assignments 15
buffers
  input mask 27
  logical 24, 27
  output mask 28
  physical 24, 27
  status line 29

calculations
  extended precision arithmetic subroutine 21
  object code 22
communications external status subroutine 20
communications management subroutine 20
compile time dump 3
compile time error information 3
compile time partition 7
compiler
  communications area (CCA) 7
    definition 49
  debug functions 45
  error handling 8
  module descriptions 8
  phases 5
  work files 8
compressions 8
control codes 30
control string commands 31
copy indicators 14

data areas 25
data areas in dump example 36
data entry driver indicators 14
debugging aids 45
debugging programs 5
diagnostic phase 9
directory, table 17
diskette
  edit format control strings 18
  external status subroutines 19
  I/O management subroutines 19
  record level code 22
dump and trace programs 47
dump example 39
dump, absolute 3
dup/store table 17

edit format control strings 18
enter phase 8
error display subroutine 20
error handling 8
example program
  dump 39
  source listing 34
execution sequence of phases 6
execution time dump 4
execution time error 4
exit code requirements 21
external status
  communications 20
  definition 49
  diskette 19
  keyboard 18
  printer 19

file close subroutine 21
file translation table 24
format control table 28

glossary 49

I/O control codes 30
I/O control string commands 31
I/O driver 25
I/O driver parameter block 30
I/O driver subroutines 20
I/O function indicators 14
identifying problems 1
indicator assignments 13
initialization code 23
input mask buffer 27

job statistics counter area 29

keyboard external status subroutine 18
keyboard record level code 22
keyboard/CRT I/O management subroutine 19

literals/prompts table 17
logical buffers 24, 27
logical file name block 32

magnetic stripe reader I/O management subroutine 19
microprocessor definition 49
miscellaneous indicators 14
mode indicators 13
module descriptions 8
module identification 9-12

named fields 17
numeric field table 32

object code, calculations 22
object phase 12
object program organization 13
object program sequence 16
output mask buffer 28
overlay area definition 49

partition control block 25
partition IOB program name 9-12
partition layout 7
partition subroutine stack 3
patch program 3, 47
phase summary 6
physical buffer allocation subroutine 21
physical buffers 25, 27
preassemble phase 11
printer edit format control strings 18
printer external status subroutine 19
printer I/O management subroutine 19
printer record level code 22
problem determination 1
PTF log number 3

record level code 22
register assignments 15
register save area 25
release number 1
reporting problems 3
resolve table element address subroutine 21
return to program exit code 21
  definition 49
running dump and trace programs 47

screen format control string table 18
sequence of phase execution 6
shared routines 7
source errors 8
source listing example 34
status line buffer 29
subroutines 18
SYSCOPY 3
SYSPRINT 3
SYSPTF 3

table
  alternate collating sequence 24
  directory 17
  dup/store 17
  file translation 24
  format control 28
  literals/prompts 17
  numeric field 32
  screen format control string 18
  user data 17
termination code 23
trace program 47
transaction data set indicators 14

user data table 17
utilities licensed program diskette 1

verify error display subroutine 20

work files 8

Z-specification driver 23

READER'S COMMENT FORM

Please use this form only to identify publication errors or request changes to publications. Technical questions about IBM systems, changes in IBM programming support, requests for additional publications, etc, should be directed to your IBM representative or to the IBM branch office nearest your location.

Error in publication (typographical, illustration, and so on). **No reply.**

Page Number *Error*

Inaccurate or misleading information in this publication. Please tell us about it by using this postage-paid form. We will correct or clarify the publication, or tell you why a change is not being made, provided you include your name and address.

Page Number *Comment*

IBM may use and distribute any of the information you supply in any way it believes appropriate without incurring any obligation whatever. You may, of course, continue to use the information you supply.

Name _____

Company or Organization _____

Address _____

_____

● No postage necessary if mailed in the U.S.A.

SC21-7852-1

Cut Along Line

Fold and tape | Please do not staple | Fold and tape

NO POSTAGE
NECESSARY IF
MAILED IN THE
UNITED STATES

BUSINESS REPLY MAIL

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

POSTAGE WILL BE PAID BY ADDRESSEE

IBM CORPORATION
Information Design and Development
Department 997
11400 Burnet Road
Austin, Texas 78758

Fold and tape | Please do not staple | Fold and tape

IBM

International Business Machines Corporation

General Systems Division
4111 Northside Parkway N.W.
P.O. Box 2150
Atlanta, Georgia 30055
(U.S.A. only)

General Business Group/International
44 South Broadway
White Plains, New York 10601
U.S.A.
(International)

International Business Machines Corporation

General Systems Division
4111 Northside Parkway N.W.
P.O. Box 2150
Atlanta, Georgia 30055
(U.S.A. only)

General Business Group/International
44 South Broadway
White Plains, New York 10601
U.S.A.
(International)

IBM 5280 DE/RPG Problem Determination Procedures for the Programmer (File No. S5280-28) Printed in U.S.A. SC21-7852-1