MICRO FOCUS | SERENA

StarTool® FDM

StarBat Option

Product version: 7.8 Patch 3

Publication date: February 2022

Table of Contents

Welcome to the StarTool FDM StarBat Option

This section contains the following information:

About This Book

This document describes the StarBat Option of StarTool® FDM (File and Data Manager) Version 7.8.

StarTool FDM StarTool FDM is a multi-purpose file and data management utility for IBM® mainframe systems. It provides an ISPF-based, menu-driven, integrated interface to a variety of editors and data management tools for PDS/PDSE, VSAM, direct-access, IMS, and Db2 files. A TSO command-line interface is also supported. A variety of available tools may be configured to specific user needs with separately licensed product options.

StarBat Option The StarBat Option provides a batch-mode interface for bulk changes to data sets. StarBat is a separately licensed, optional feature of StarTool FDM.

Audience This document is intended for use by IBM mainframe systems programmers and application specialists already familiar with the functions and uses of StarTool FDM.

Before You Begin

New Information Change bars in the left margin (shown at left) identify substantive changes to this publication since StarTool FDM Version 7.6.3.

Corrections and Technical Support The Readme file contains updates and corrections to this manual issued after the publication date. It also provides contact information for Micro Focus Customer Support.

Conventions

Terminology Throughout this document:

- z/OS refers to the z/OS™ and OS/390® IBM® operating systems.
- StarTool FDM may also be referenced as StarTool or FDM.

Typographic Conventions

The following textual conventions are used throughout this document to highlight special information:

Convention	Meaning
Bold	Panel title or field name.
Italics	Introduces new terms, sets off important information, or marks document titles.
UPPERCASE	Indicates keys or key combinations; for example, the ENTER key.
Bright blue	Clickable cross-reference or active hyperlink.
`Monospaced`	JCL, source code, or message text. Also used for member names, file names, and commands if these are not clear from context.
`MONOSPACED UPPERCASE`	Required value or literal in code or JCL parameter.
`monospaced lowercase` *Examples:* ■ `yyyy/mm/dd` ■ `C'aa'` ■ `B'nn'` ■ `D'nn'` ■ `X'nn'`	Pattern for a field value or parameter you specify. Number of characters is significant. Upper-case characters are literals. Lower-case characters are placeholders that indicate data type, where: `y` = year `m` = month `d` = day `a` = alphanumeric `n` = numeric `x` = other or mixed `?` = one-character wild card `*` = *n*-character wild card Punctuation other than wild cards must be reproduced in the position shown. *Examples:* ■ International calendar date with four-character year, two-character month, and two-character day separated by required slashes, such as `2010/01/01`. ■ Alphanumeric character string in user-readable form, two characters long, such as `C'K9'` ■ Binary number, two digits long, where `n = 0` to `1`, such as `B'10'` ■ Decimal number, two digits long, where `n = 0` to `9`, such as `D'10'` ■ Hexadecimal number, two digits long, where `n = 0` to `F`, such as `X'C1'`
`monospaced italics`	Descriptive placeholder for value or parameter you specify, but not a pattern; for example, *filename*.
Square braces `[ ]`	Optional parameter or choice of values. May be nested.
Vertical bar `|`	Inside braces, a vertical bar separates mutually exclusive parameter choices or values.
Ellipsis `. . .`	Optional repetitions of a pattern in a list.

Convention	Meaning
Greater-than symbol >	Separates items in a chain of menu or command selections on a GUI client. For example, **Start > All Programs > Micro Focus > *product_name***.

Documentation

All StarTool FDM documentation is provided in Adobe® Portable Document Format (PDF) and may be downloaded from the following Micro Focus website:

https://www.microfocus.com/support-and-services/documentation/

No user ID or password is needed to access the documentation.

Related Publications

Available StarTool FDM publications include:

Title	Description
StarTool FDM Installation Guide	System requirements, installation instructions, and configuration information for StarTool FDM.
SER10TY User's Guide	Installation information for SER10TY licensing software and instructions on how to apply license key SERtificates.
StarTool FDM Quick Reference	Overview of StarTool FDM commands, with syntax details for frequently used functions. Includes PEDIT and StarBat subcommands.
StarTool FDM User's Guide	StarTool FDM concepts and facilities, with instructions for using the ISPF-based menu-driven interface.
StarTool FDM Command Reference	TSO command-line syntax and parameter reference, organized alphabetically. Interactive subcommands included for major functions.
StarTool FDM System Services	Advanced reference to operating system calls used by StarTool FDM.
StarTool FDM StarBat Option	Batch-mode interface for bulk changes to data sets. StarTool FDM functions invoked by JCL procedures.
StarTool FDM Db2 Option	StarTool FDM data management functions for Db2 relational database tables, columns, and rows, with SQL processing support.
StarTool FDM IMS Option	StarTool FDM data management functions for IMS hierarchical database files and structures.
StarTool FDM Extended Compare Option	Integrated file comparison utility based on Micro Focus Comparex. Data file versus text file comparison logic.
StarTool FDM Messages	Consolidated message reference for base product and all licensed product options, with error recovery recommendations.

Using the PDF Documentation

To view PDF files, use Adobe® Reader®, which is freely available from Adobe on the World Wide Web at *http://www.adobe.com*. Reader Version 7.0.5 or higher is recommended.

TIP Be sure to download the *full version* of Reader. The more basic version does not include the cross-document search feature.

This section highlights some of the main Reader features. For more detailed information, see the Adobe Reader online help system.

The PDF manuals include the following features:

- **Bookmarks.** All of the online manuals contain predefined bookmarks that make it easy for you to quickly jump to a specific topic. By default, the bookmarks appear to the left of each online manual.
- **Links.** Cross-reference links within an online manual enable you to jump to other sections within the manual and to other manuals with a single mouse click. These links appear in blue.
- **Printing.** While viewing a manual, you can print the current page, a range of pages, or the entire manual.
- **Advanced search.** Starting with Version 6, Adobe Reader includes an advanced search feature that enables you to search across multiple PDF files in a specified directory. (This is in addition to using any search index created by Adobe Catalog—see step 3 below.)

To search across multiple PDF documents at once, perform the following steps (requires Adobe Reader Version 6 or higher):

1. In Adobe Reader, select Edit > Search (or press CTRL+F).
2. In the text box, enter the word or phrase for which you want to search.
3. Select the **All PDF Documents in** option, and browse to select the folder in which you want to search. (If you have a document open that has an Adobe Catalog index attached, you can leave the **In the index named...** option selected to search across all the manuals in the index.)
4. Optionally, select one or more of the additional search options, such as **Whole words only** and **Case-Sensitive**.
5. Click the **Search** button.

NOTE Optionally, you can click the **Use Advanced Search Options** link near the lower right corner of the application window to enable additional, more powerful search options. (If this link says **Use Basic Search Options** instead, the advanced options are already enabled.) For details, see Adobe Reader's online help.

Chapter 1

StarBat Overview and Installation

The StarBat Option of StarTool FDM is an batch program that uses standard JCL and SYSIN control statements to manipulate data files in bulk in a batch processing environment. Bulk data manipulation is supported on disk or tape media.

This section introduces the features, functions, and requirements of the StarBat Option.

Features and Components

Using StarBat, you can perform bulk file operations in batch mode. Control statements in your JCL or in a data set pointed to by the SYSIN DD statement specify the functions you want performed.

For example, with StarBat you can:

- Copy, update, or print an entire file or specific records within the file that satisfy certain conditions.
- Copy selected records or portions of records from one data set type to another.
- Add or enlarge data fields for a particular record type.
- Reformat date fields for a large number of files at once.

StarBat Functions

There are two main uses for StarBat: to manipulate members of a data set and to manipulate records in a file or data set member.

Manipulate Members of a Data Set

Different StarBat functions can copy, print and update members of a data set. These functions are COPYMBR, PRINTMBR, PRINTCHRMBR, PRINTHEXMBR, and UPDATEMBR.

Manipulate Records in a Member or Data Set

This is by far the most common use of the StarBat. This allows for copying, deleting, and inserting of records. Additionally, you can print records as text, hexadecimal, or both. You can total and reformat the data in the records by using COPYALL, COPYREC, COPYREV, COPYSOME, EXCLUDEREC, MULTICOPY, PRINT, PRINTALL, PRINTCHR, PRINTCHRALL,

PRINTCHRREV, PRINTHEX, PRINHEXALL, PRINTHEXREV, PRINTREV, SKIP, SKIPREV, TOTAL, UPDATEALL, and UPDATEREC.

These functions fall into five categories.

- **Copying data —** There are occasions when you need to create a subset of a data file and, by combining the various copy functions, the exclude function and the conditional parameters, you can develop sophisticated jobs to manipulate records to create new data files.
- **Deleting data —** Whether it is deleting bad records or dropping records because they are no longer required, StarBat has the ability to do this effortlessly. In combination with conditional parameters, you can edit a file in batch mode without creating a special purpose program to do it.
- **Inserting data —**In addition to copying and deleting records, StarBat can insert new records if required. You can add new record types to a test data file so that you can test new functionality in a program.
- **Printing data —**StarBat allows you to print data sets and members in text mode (which is suitable for source files) and in hexadecimal mode (which is more suited to data files). This is invaluable in testing a program because it clearly shows the exact contents of the files.
- **Changing data —** One of the most powerful features of StarBat is its ability to modify the content of a record. Using commands similar to ISPF edit commands, it is possible to change whole files or individual records according to some defined pattern. These changes include the ability to increase the size of a field, as might be needed with converting a date format from `MMDDYY` to `YYYYMMDD`.

StarBat Commands

StarBat support the following commands:

Function	Short	Description
COPYALL	CA	Copies all records of a data set
COPYMBR	CM	Copies members conditionally based on contents
COPYREC	CR	Copies data and reports record totals
COPYREV	CPR	Copies data in reverse order
COPYSOME	CS	Copies selected records but applies all changes like COPYALL
EXCLUDEREC	XR	Eliminates unwanted records in a copy
FPRINT		Prints members to a file
MULTICOPY	MC	Copies data to one or more output data sets
PRINT	P	Prints records in alphanumeric format with record statistics
PRINTALL	PA	Prints all records of a data set
PRINTCHR	PC	Prints records in alphanumeric format
PRINTCHRALL	PCA	Prints all records of a data set

Function	Short	Description
PRINTCHRMBR	PCM	Prints members conditionally based on contents
PRINTCHRREV	PCR	Prints records in alphanumeric format and reverse order
PRINTHEX	PH	Prints records in vertical hexadecimal format
PRINTHEXALL	PHA	Prints all records of a data set
PRINTHEXMBR	PHM	Prints members conditionally based on contents
PRINTHEXREV	PHR	Prints records in vertical hexadecimal format and reverse order
PRINTMBR	PM	Prints members conditionally based on contents
PRINTREV	PRR	Prints records in alphanumeric format and reverse order
SKIP	S	Moves the current record pointer forward
SKIPREV	SKR	Moves the current record pointer backward
TOTAL	T	Reads input records processing all parameter groups for SUM
UPDATEALL	UA	Updates all records of a data set
UPDATEMBR	UM	Updates members conditionally based on contents
UPDATEREC	UR	Updates records in place

Parameter Descriptions

The following keyword parameters are used in a standard way across one or more StarBat commands.

Parameter	Short	Description
ABEND	AB	Controls EOJ processing when abnormal condition occurs
AND	IF	Creates a logical AND condition check (used with IF)
CHANGE	C	Changes only the first instance of data in a record
CHANGEALL	CA	Changes all occurrences of data in a record
COPYOVER	CO	Controls the replacement of identically named output members
EXCLUDEREC	XR	Controls the # of records to bypass in an EXCLUDEREC function
EXPAND	EX	Expands records at a specified location

Parameter	Short	Description
IF	AND	Selects records to process based on data contents
MAP		Defines how lines are printed in the COPYBOOK format. The listing output is sent to the SYSLIST DD.
MAXRECIN	MRI	Controls the number of records to input
MAXRECOUT	MRO	Controls the maximum number of records to output
MEMBER	M	Specifies a member name to process in a PDS
MEMBERS	MS	Specifies a group of members to process in a PDS with a mask
MOVE	MV	Moves data into the record
NEWMBR	NM	Gives a new name to an output PDS member
NEWMBRS	NMS	Names multiple new members of an output PDS using a mask
OPTIONS	OP	Controls StarBat processing options
OR		Used with the IF parameter to indicate an OR condition
OVERALL	OA	Replaces all occurrences of data in a record with other data
OVERLAY	OL	Replaces the first instance of data in a record with new data
PADCHAR	PAD	Specifies a padding character for uninitialized parts of a record
PRINT	P	Prints records in alphanumeric format with record statistics
PRINTCHR	PC	Prints records in alphanumeric format
PRINTHEX	PH	Prints records in vertical hexadecimal format
PRINTLPI	PL	Specifies the number of lines per inch for print output pages
RBA		Processes VSAM data beginning at a relative byte address
RDW		Controls the inclusion of the record descriptor word
SELECT	S	Processes every nth record
STARTKEY	SK	Processes VSAM data beginning with a generic key
STOPIF	ST	Stops processing a function when a record satisfies a condition
SUM		Accumulates the contents of specified fields
WRITE	W	Writes a record to one or more output files

Requirements

StarBat installation prerequisites are largely the same as those for StarTool FDM in general. However, as a separately licensed product option, a separate license key is required. The following information should assist your installation planning.

Software Environment

StarBat runs under IBM z/OS. In addition, the following environments should be available:

- ISPF and ISPF/PDF (Version 4.1 or above)
- TSO/E
- SER10TY license management software

The Readme file provides the latest information concerning software requirements and supported versions.

Storage Requirements

Region Size

StarTool FDM in general performs best in region size of 3072K or larger. However, a 2048K region is usually sufficient to execute most StarBat jobs.

Data Set Organization

StarBat supports most mainframe disk data organizations and access methods, including partitioned data sets (PDSs), extended partitioned data sets (PDSEs), sequential data sets, VSAM data sets, and direct files. ISAM data sets are not supported. StarBat supports tape files with sequential organization. Any record format is acceptable. Character, hexadecimal, packed decimal, and binary data formats are all supported.

NOTE StarBat assumes that any partitioned data set with record format U is a load library. Other libraries are usually called "source libraries" in this manual.

Large Block Interface (LBI)

The StarBat Option supports the Large Block Interface (LBI) feature of DFSMSdfp for its primary tape input and output files using the `DD`*xx* and `DD`*xx*`0` DD statements. The LBI enables tape block sizes larger than 32K bytes. The actual block size depends on the tape drive control unit, and is either 65,535 or 262,144 bytes.

Use of the LBI increases the virtual storage requirements of StarBat jobs. Customers should evaluate their current `REGION=` parameters to allow for growth in buffer space requirements when using the LBI.

The LBI feature is not supported on disk devices, and StarBat does not support block sizes larger than 32,760 bytes for any other DD statement.

Files created by the MULTICOPY WRITE parameter do not support the LBI feature.

Installing and Licensing StarBat

The underlying code for StarBat is installed automatically with the base product for StarTool FDM. However, activating this feature requires an additional license. Contact your sales representative or Micro Focus Customer Support to obtain a license key for StarBat.

Optional license keys may be applied to an existing installation without reinstalling code. This is done using Micro Focus's licensing software, SER10TY. For instructions, refer to the *SER10TY User's Guide*, which is included on the distribution media with StarTool FDM.

Customizing StarBat

Certain global features of StarBat behavior may be customized permanently. StarBat-specific customization options are summarized here for reference.

Multivolume Tape Processing

Multivolume record counts can be enabled for StarBat by customizing the StarTool FDM options configuration module PDS#OPT4.

The #INITIAL macro in PDS#OPT4 takes a parameter keyword called SPFVOMSG that turns on record counts for each tape volume in a multivolume data set. Both input and output files are processed if this option is enabled. If this option is disabled, only the total record count for all volumes in the multivolume file is supplied.

Parameter	Description	Values	Default
SPFVOMSG	Provide individual record counts for each volume during multivolume tape processing in StarBat.	Y = Enable N = Disable	N

See the *StarTool FDM Installation Guide* for instructions on how to customize the PDS#OPT4 member.

Return Code Processing

Return code processing in StarBat may be modified by customizing the StarBat return code table SZFRCTAB. Return code customization for StarBat is independent of any customizations performed for the base StarTool FDM product in PDS#OPT4. See Appendix B, "Return Code Processing in StarBat" for details.

Chapter 2

Control Statement Syntax

Invoke StarBat with a JCL EXEC statement and any desired runtime parameters. Identify the data sets to be manipulated with standard DD statements. Then read in StarBat control statements as SYSIN data from the JCL job stream at runtime.

JCL Syntax to Invoke StarBat Jobs

The following example shows the JCL used to invoke a typical StarBat job. Italics indicate values supplied by the user. StarBat control statements are highlighted in bold type.

```
//jobname  JOB (job_card_parameters)
//stepname EXEC PGM=STARBAT,REGION=2048K,PARM='starbat_parm'
//STEPLIB  DD   DSN=steplib_dsn,DISP=SHR
//DD03     DD   DSN=input_dsn,DISP=SHR
//DD03O    DD   DSN=output_dsn,DISP=SHR
//SYSPRINT DD   SYSOUT=*
//SYSLIST  DD   SYSOUT=*
//SYSTOTAL DD   SYSOUT=*
//SYSIN    DD   *
DD03 COPYALL=(1,0,C'PGM=TESTPROG',C'PGM=PRODPROG'),
     PRINTCHR=100
```

The usual rules for EXEC and DD statements apply.

- The EXEC statement tells the operating system the name of the program to be executed — in this case, `PGM=STARBAT`.
- A STEPLIB or JOBLIB statement is required unless StarBat is present in your system's LINKLIST.
- JCL DD statements are required to describe input and output data sets to StarBat. The *xx* value can be any number from `00` through `99`. JCL DD names match the labels on the StarBat control statements that apply to the named data sets. Note that input data sets may be concatenated.

StarBat's primary output is directed to the SYSPRINT DD statement. This statement is optional; it is allocated dynamically if it is not present. StarBat echoes control statements and error, status, and completion messages to this DD name. StarBat expects this to be a data set with standard data control block attributes for a listing file, namely: `RECFM=FBA`, `LRECL=133`, and `BLKSIZE` a multiple of 133. Note that an LRECL of 80 through 132 will work but will truncate some print lines.

If you specify the PRINT or PRINTHEX StarBat functions, include the SYSLIST DD statement. If it is omitted, it is allocated dynamically if needed. This data set has the same DCB attributes as the SYSPRINT data set.

If you specify the SUM option, include the SYSTOTAL DD statement. If it is omitted, all SYSTOTAL output is redirected to the SYSPRINT data set. This data set also has the same data control block attributes as the SYSPRINT data set.

A DD*xx*O statement is required for COPYREC and EXCLUDEREC functions. The *xx* value must match the *xx* value in the input data set defined on the corresponding DD*xx* JCL DD statement. If necessary, basic data control block information such as record format (RECFM), logical record length (LRECL), and block size (BLKSIZE) will be copied from the input data set to the output data set when the data set is opened.

Additional output DD statements with any desired name may be specified for the MULTICOPY function. The MULTICOPY WRITE parameter names the output DD statement and causes output records to be written when the control statement is processed.

The SYSIN statement is used to read StarBat control statements.

StarBat Control Statements

Control statements must be 80 bytes long and all 80 columns can be used. If no control statements are found, StarBat defaults to a COPYREC function for every pair of corresponding input and output data set DD names (//DD*xx* and //DD*xx*O).

StarBat control statements consist of a data set identifier, a function name, and one or more parameters separated by commas. Comments are allowed following the first parameter and subsequent parameters in a parameter list. Comments may also stand alone on a separate line.

In the following example, DD01 is the data set identifier, PRINTCHR is the function name, IF=(10,EQ,C'A') is the parameter, and the remainder of the statement is comments.

```
DD01 PRINTCHR IF=(10,EQ,C'A') /*Print if 'A' in column 10*/
```

Data Set Identifier

The data set identifier identifies which data set in the JCL DD statement is being processed. The format of the data set identifier is DD*xx* where *xx* is from 00 to 99. The number references the //DD*xx* DD statement, which specifies the input data set to be processed. DD*xx* must begin in column 1 of the control statement. If the function is COPY or EXCLUDEREC, DD*xx* also references the //DD*xx*O DD statement representing the output data set.

Function Identifier

The function identifier identifies the action that you want to perform on the data. It is always positioned after the data set identifier with one or more spaces.

Parameters

Keyword parameters define the scope of the requested StarBat function. They follow the function identifier and are separated from it by one or more spaces. Multiple parameters are allowed, separated by commas, and may continue onto multiple lines.

Each parameter is followed by an equal sign and a single value or a comma-delimited value list bracketed by parentheses. Parameter values in a value list follow a specific order and contain the starting column of the operand, an operator, and data fields required for the operation (for example, a comparison value for a compare operation).

Start Location

A start location specifies the location at which data can be found in the input record. A start location can be coded in one of two ways: actual start location and relative start location.

Actual start location is the exact column number where the data in the record is positioned. The number can be from 1 to 32767, but it cannot be larger than the record size.

Relative start location indicates the position relative to the current location. StarBat supports both input relative start location and output relative start location.

When an input record is first read by StarBat, the relative location is set to the beginning of the record. Parameters such as IF, CHANGE, and OVERLAY change this location value. Specify a relative start location by placing a plus sign or a minus sign before a number. For example, to reference a location 10 bytes before the current location, use –10; to reference a location 5 bytes after the current location, use +5. Output relative location is supported when using the MOVE parameter with the COPY or MULTICOPY functions. When data is moved to an output location, the output relative location moves to the next available output position.

Operators

Operators are EQ, NE, GT, LT, GE, and LE. Use them to compare data with an input record.

StarBat also supports bit value comparisons using operators AO (all ones), AZ (all zeroes), NO (not ones), and MX (mixed ones and zeroes) with the IF, OR, CHANGE, or OVERLAY parameters.

When the exact location of the compare data is unknown, use the length element in place of the operator element. In this case, an equal comparison is assumed.

To scan the entire input record, use a length of `0`. Otherwise, the length value must be at least one greater than the length of the compare data; also, the sum of the current input location and the length value must be less than or equal to the record length. Length cannot be greater than 255.

The IF, OR, CHANGE and OVERLAY parameters are changed to scanning parameters when you use the length element. Therefore, they change the relative start input position as previously mentioned.

Data

StarBat supports character, hexadecimal, binary, and packed data. The data element can be used as compare data, as replacement data, or as literal data.

Enclose character data in single or double quotes. When used as compare data, enter more than one value with commas in between, indicating an OR condition, as long as they are enclosed in single quotes. For example, IF=(10,EQ, C'TEST,PROD') checks if the record contains the characters TEST or PROD in column 10. Use a duplication factor to

avoid coding repetitive data elements. For example, instead of coding C'TEST,PROD,TEST,PROD,', you could code 2C'TEST,PROD,'.

Alphanumeric data is case-sensitive and is not translated to upper case. To search for data in both upper and lower case, use the T character string. For example, to search for PROD, Prod, or proD, use T'Prod'.

All character data must fit into a single control statement; continuations are not supported.

Enter packed data with any valid length that fits on a single control statement (continuations are not supported). You must enclose all numeric digits in single quotes. StarBat calculates the data length using the data entered. Enter a plus or minus before each string of numbers; an unsigned packed number is considered positive. For example, P'12' and P'+12' are equivalent.

Enter binary data with a beginning H character for halfword binary numbers or an F character for fullword binary numbers. Enclose all numeric digits in single quotes. You can enter a plus or minus before each string of numbers; an unsigned binary number is considered positive. For example, F'12' and H'-12'.

Enter hexadecimal data with any length that fits on a single control statement (continuations are not supported). You must enclose all hexadecimal digits in single quotes. StarBat calculates the data length using the data entered and it must always have an even number of digits. You can also use a numeric duplication factor to avoid coding repetitive data elements. For example, 3X'003F' and X'003F003F003F' are equivalent.

Comments

Code StarBat comments on the command line after the first parameter in the parameter string. Multiple comments may be interspersed with parameters. Comments may also stand alone in the StarBat job stream.

Comments in StarBat control statements observe the following conventions.

- All comments begin with the characters /* (slash asterisk) and may optionally be terminated with the characters */ (asterisk slash). If not terminated, the comment continues to the end of the statement.
- If the first two bytes of a command contain the comment delimiters /*, the entire command is treated as a comment.
- A comment may be inserted within a command anywhere following the command name and the first data string parameter or MEMLIST parameter.
- A comment may not appear within a data string.
- A keyword parameter must be separated from any following comment by a delimiter such as a space, comma, or parenthesis.
- Multiple comments are permitted within a command.

For example, the following comments are valid in StarBat commands:

```
FIND 'ABCDEF' ASIS /* COMMENTS */
FIND 'ABCDEF' /* COMMENTS */ ASIS
FIND 'ABCDEF' ASIS /* COMMENTS */ OR('FEDCBA') /*MORE COMMENTS*/ OASIS
FIND 'ABCDEF' ASIS OR(/* comments */'FEDCBA') OASIS /*MORE COMMENTS
```

The following comments are invalid in StarBat commands:

```
FIND 'ABCDEF' ASIS/* COMMENTS */
FIND 'ABCDEF'/* COMMENTS */ ASIS
FIND 'ABCDEF' ASIS /* COMMENTS */ OR('FEDCBA')/*MORECOMMENTS*/ OASIS
```

Runtime EXEC Parameters

StarBat accepts certain runtime parameters on the EXEC JCL statement that modify program behavior during the current execution. Only one parameter is accepted at a time.

Use the following syntax to submit an EXEC parameter:

`//`*`stepname`* `EXEC PGM=STARBAT,PARM='`*`starbat_parm`*`'`

where:

- *`stepname`* is the name of the StarBat job step, and
- *`starbat_parm`* is one of the parameter values listed in the table below.

StarBat EXEC Parameter	Description
BSAM	For customers who cannot use QSAM processing. Turns off EXCP processing for sequential and partitioned data sets.
PRINTRCT	Prints the current StarBat return code table.
TSO	Allows execution in a foreground address space. Also converts page breaks to triple line breaks in the SYSLIST output. (Normally, each member output to the SYSLIST begins on its own page.)

Chapter 3

Data Set Processing

StarBat supports sequential, direct, VSAM, and partitioned data sets. ISAM data sets are not supported.

Sequential Data Sets

StarBat can create any type of sequential data set as output. Input concatenated data sets on unlike devices are supported but these data sets must have similar characteristics. When processing concatenated input data sets, StarBat treats all of these data sets as one logical input data set; no reports are provided on the individual data sets in the concatenation aside from a record count for each concatenated data set.

Direct Data Sets

StarBat uses BSAM to read direct data sets. If RKP is greater than zero, you do not need to perform any special action to copy the data and the imbedded keys. If, however, RKP is zero, you will need to use a MOVE parameter with a negative relative location to reference the key portion because the key is not also present in the data.

For example, if you have a record with a 20 byte key and an 80 byte data portion, use the following MOVE parameter to copy both the key and data portion of the record to an output record:

```
DD01   COPYREC    MOVE=(1,100,-20)
```

VSAM Data Sets

StarBat supports RRDS, KSDS, and ESDS data sets. Linear VSAM data sets are supported but the CISIZE has to be 4K.

Partitioned Data Sets

StarBat supports PDS and PDSEs. To process a specific member, use the MEMBER parameter. To process multiple members, either use multiple control statements with the MEMBER parameter (only one MEMBER parameter is allowed for each control statement), or use the MEMBERS parameter specifying a member mask. To process the entire data set, do not specify any MEMBER or MEMBERS parameter.

If the PDS and PDSEs are concatenated as input and if you use the MEMBER parameter, StarBat processes each data set in the order of concatenation and stops processing the function as soon as the member is found.

There are a few restrictions when processing load library members:

- Load PDSE members cannot be used as output in a copy.
- The MULTICOPY function does not support scatter-loaded, overlay, or note-listed modules.
- The block size of the input data set must be equal to or less than the output data set.
- The main module associated with an alias member is always copied at the same time an alias is copied.

Chapter 4

Functions

Following is a brief description of each supported function.

Function	Short	Description
COPYALL	**CA**	Copies all records of a data set
COPYMBR	**CM**	Copies members conditionally based on contents
COPYREC	**CR**	Copies data and reports records copied
COPYREV	**CPR**	Copies data in reverse order
COPYSOME	**CS**	Copies selected records but applies all changes like COPYALL
EXCLUDEREC	**XR**	Eliminates unwanted records in a copy
FPRINT		Prints members to a file
MULTICOPY	**MC**	Copies data to one or more output data sets
PRINT	**P**	Prints records in alphanumeric format with record statistics
PRINTALL	**PA**	Prints all records of a data set
PRINTCHR	**PC**	Prints records in alphanumeric format
PRINTCHRALL	**PCA**	Prints all records of a data set
PRINTCHRMBR	**PCM**	Prints members conditionally based on contents
PRINTCHRREV	**PCR**	Prints records in alphanumeric format and reverse order
PRINTHEX	**PH**	Prints records in vertical hexadecimal format
PRINTHEXALL	**PHA**	Prints all records of a data set
PRINTHEXMBR	**PHM**	Prints members conditionally based on contents
PRINTHEXREV	**PHR**	Prints records in vertical hexadecimal format and reverse order
PRINTMBR	**PM**	Prints members conditionally based on contents
PRINTREV	**PRR**	Prints records in alphanumeric format and reverse order
SKIP	**S**	Moves the current record pointer forward
SKIPREV	**SKR**	Moves the current record pointer backward
TOTAL	**T**	Reads input records processing all parameter groups for SUM
UPDATEALL	**UA**	Updates all records of a data set

Function	Short	Description
UPDATEMBR	**UM**	Updates members conditionally based on contents
UPDATEREC	**UR**	Updates records in place

COPYREC, COPYSOME, and COPYALL

Copies one, some, or all records in a member and reports a count of records output. If it is required to copy records from one data set into another, use the COPYREC (**CR**) function. With the addition of qualifiers it is possible to control the records selected for copying. Additionally, in the process of copying, you can modify the data by expanding or contracting fields and changing the data within fields. Use the COPYSOME (**CS**) function to process multiple conditional updates like COPYALL while copying selected records like COPYREC.

The following example copies up to *1000* records that contain *USA* anywhere after position 256:

```
DD01 COPYREC IF=(256,0,C'USA'),MAXRECOUT=1000
```

The following example copies the input data set and replaces the first *USA* in the first three positions with *DOM*:

```
DD01 COPYREC OVERLAY=(1,3,C'USA',C'DOM')
```

The following example copies every other record that contains *USA* in the first three positions:

```
DD01 COPYREC SELECT=2,IF=(1,3,C'USA')
```

The following example copies records containing an 8 or 9 and warp each record with YYDDD or YYMMDD format. If a record has both formats, only the YYDDD format will be warped.

```
DD01 COPYREC   IF=(12,EQ,C'8'),
              WARP=(16,C,YYDDD,ADD=5Y),
              IF=(22,EQ,C'9),
              WARP=(26,C,YYMMDD,ADD=5Y)
```

The following example copies all records and warps all 8 and 9 dates:

```
DD01 COPYALL   IF=(12,EQ,C'8'),
              WARP=(16,C,YYDDD,ADD=5Y),
              IF=(22,EQ,C'9),
              WARP=(26,C,YYMMDD,ADD=5Y)
```

The following example copies records containing an 8 or 9 and warps all dates on these records:

```
DD01 COPYSOME  IF=(12,EQ,C'8'),
```

```
WARP=(16,C,YYDDD,ADD=5Y),
IF=(22,EQ,C'9),
WARP=(26,C,YYMMDD,ADD=5Y)
```

COPYMBR

Use COPYMBR (**CM**) to copy one or more dataset members based on their contents.

Use the NEWMBRS keyword with COPYMBR to add a common prefix to member names. Indicate the prefix with a parameter value in the form XXX+, where XXX are the characters to be used as the prefix and the '+' indicates this is a prefix. If the length of the prefix plus the length of the original member name is greater than eight bytes, the member is skipped.

The following example copies all members whose names begin with STAR containing *USA* anywhere in the member:

```
DD01 COPYMBR MEMBERS=STAR-,IF=(1,0,C'USA')
```

NOTE COPYMBR – If you specify a single member only, and the input is a sequential data set rather than a PDS, COPYMBR operates much as COPYREC and copies the input file. If you specify multiple members in this situation, only a single copy of the input data set is copied. While inconsistent with the processing by COPYMBR, this allows customers to change data sets in JCL without changing the associated StarBat control statements.

COPYREV

The **COPYREV** (**CR)** function copies records from a sequential or VSAM data set in reverse order. This means that the first record copied will be the last record from the input data set followed by other records from the input data set moving toward the front of the data set.

The following example copies records except those containing *RED* anywhere in reverse order:

```
DD01 COPYREV IF=(1,0,C'RED')
```

The following example copies up to 1000 records that contain *USA* anywhere after position 256 in reverse order:

```
DD01 COPYREV IF=(256,0,C'USA'),MAXRECOUT=1000
```

EXCLUDEREC

The **EXCLUDEREC** (**XR)** function excludes (or eliminates) records during a copy; EXCLUDEREC selects records that would be bypassed by COPYREC.

The following example copies all records except those containing *RED* anywhere:

```
DD01 EXCLUDEREC IF=(1,0,C'RED')
```

The following example copies all records except the first *100* containing *RED* anywhere:

```
DD01 EXCLUDEREC IF=(1,0,C'RED'),EXCLUDEREC=100
```

The following example copies the input data set and drops any record that contains an A in position 1 or contains a value higher than *CAA* in location *112*. Processing stops after copying *20* records:

```
DD01 EXCLUDEREC IF=(1,EQ,C'A'),OR=(112,GT,C'AA'),MAXRECOUT=20
```

MULTICOPY

The **MULTICOPY** (**MC)** function copies data to one or more output data sets. With the MULTICOPY function, you can direct output records to several output data sets and add, delete, or modify records being copied to the output data sets at the same time.

To control the destination of the records, use the **WRITE** parameter to specify the output DDNAME.

Output records for MULTICOPY are created piece by piece with MOVE or by copying the input record. If you use the MOVE parameter, the output area is initialized to the PADCHAR value and not reset between different WRITE parameters.

There are several restrictions on the data that can be output with a MULTICOPY function:

- Output to load members is supported; however, JCL indicating PDS(MEMBER) notation is not supported. Instead, use only the PDS name in the JCL and indicate the member name with the MEMBER, MEMBERS, NEWMBR, or NEWMBRS parameters.
- Overlay, scatter-loaded, and note-listed load members are not supported.
- IBM's Large Block Interface is not supported for files created by the MULTICOPY WRITE parameter.
- Default DD names should not be used with the `WRITE=` parameter of the StarBat MULTICOPY command. When the MULTICOPY `WRITE=` parameter uses the same, default DD name (usually `DDxx0`) as a prior command that leaves the file open for output, an error condition results. This could occur, for example, if a SKIP command precedes the MULTIWRITE command.

The following example shows how to create two identical output data sets in `OUT1` and `OUT2`:

```
DD01 MULTICOPY WRITE=(OUT1,OUT2)
```

The following example copies a file, repeats any record with a *C'5'* and changes it to a *C'6':*

```
DD01 MULTICOPY WRITE=OUT1,IF=(20,EQ,C'5'),MOVE=(1,0,1),
             MOVE=(20,C'6'),WRITE=OUT1
```

The following example adds a STEPLIB JCL statement after any EXEC statement:

```
DD01 MULTICOPY WRITE=NEWJCL,IF=(1,20,C' EXEC '),MOVE=(1,80C' '),
            MOVE=(1,C'//STEPLIB  DD  DISP=SHR,'),
             MOVE=(+0,C'DSN=MYHILEV.MYMIDLEV.MYLOWLEV'),WRITE=NEWJCL
```

PRINT, PRINTALL, PRINTMBR, and PRINTREV

Print (**P**) prints a data set or a portion of a data set in alphanumeric format with identifying information such as record number, RBA, and record length. Normally, a column scale is printed with each record; however, if you request OPTIONS=SHORT, the column scale is produced only after each page header.

- PRINTMBR (**PM**) prints members based on their contents.
- PRINTALL (**PA**) processes multiple conditional updates (IF OVERLAY, IF CHANGE, IF MOVE, IF WARP) while printing records.
- PRINTREV (**PRR**) prints sequential or VSAM data sets in reverse order.

The following example prints the first 10 records:

```
DD01 PRINT MAXRECOUT=10
```

The following example prints the first 20 records containing RED in position 12 replaced with BLU. The input data set is not actually changed; use this function to see changes before running an UPDATEREC:

```
DD01 PRINT OVERLAY=(12,EQ,C'RED',C'BLU'),MAXRECOUT=20
```

The following example prints the first 100 positions of the first 20 records that contain RED in position 12:

```
DD01 PRINT IF=(12,EQ,C'RED'),MAXRECOUT=20,MOVE=(1,100,1)
```

The following example prints the first 100 positions of the last 20 records with RED in position 12 in reverse order:

```
DD01 PRINTREV IF=(12,EQ,C'RED'),MAXRECOUT=20,MOVE=(1,100,1)
```

PRINTCHR, PRINTCHRALL, PRINTCHRMBR and PRINTCHRREV

PRINTCHR (**PC**) prints a data set or a portion of a data set in alphanumeric format with no additional record information. To print a ruler at the top of each page to help identify the columns used, specify the `OPTIONS=SHORT` parameter.

PRINTCHRMBR (PCM) prints members based on their contents.

PRINTCHRALL (**PCA**) processes multiple conditional updates (IF OVERLAY, IF CHANGE, IF EXPAND, IF MOVE, IF WARP) while printing records.

PRINTCHRREV (**PCR**) prints sequential or VSAM data sets in reverse order.

The following example lists the first *10* records:

```
DD01 PRINTCHR MAXRECOUT=10
```

The following example lists the first *20* records containing *RED* in position *12* replaced with *BLU*. The input data set is not actually changed; use this function to see changes before running an UPDATEREC:

```
DD01 PRINTCHR OVERLAY=(12,EQ,C'RED',C'BLU'),MAXRECOUT=20
```

The following example lists the first *100* positions of the first *20* records that contain *RED* in position *12*:

```
DD01 PRINTCHR IF=(12,EQ,C'RED'),MAXRECOUT=20,MOVE=(1,100,1)
```

The following example lists the last *10* records in reverse order:

```
DD01 PRINTCHRREV MAXRECOUT=10
```

PRINTHEX, PRINTHEXALL, PRINTHEXMBR, and PRINTHEXREV

PRINTHEX (**PH**) prints records in a vertical hexadecimal format. The report produced also supplies additional useful information such as the record number and the record length. Normally, a column scale is printed with each record; however, if you request OPTIONS=SHORT, the column scale is only produced after each page header.

PRINTHEXMBR (PHM) prints members based on their contents.

PRINTHEXALL (PHA) processes multiple conditional updates (IF OVERLAY, IF CHANGE, IF EXPAND, IF MOVE, IF WARP) while printing records.

PRINTHEXREV (PHR) prints sequential or VSAM data sets in reverse order.

To print the first *10* records of a file in vertical hexadecimal:

```
DD01 PRINTHEX MAXRECOUT=10
```

The following example prints, in vertical hexadecimal, the first **20** records containing *RED* in position *12* replaced with *BLU*. The input data set is not actually changed; use this function to see changes before using the update functions.

```
DD01 PRINTHEX OVERLAY=(12,EQ,C'RED',C'BLU'),MAXRECOUT=20
```

This example prints, in vertical hexadecimal, up to *100* positions of the first *20* records with *RED* in position *12*:

```
DD01 PRINTHEX IF=(12,EQ,C'RED'),MAXRECOUT=20,MOVE=(1,100,1)
```

The following example prints, in vertical hexadecimal, the last *10* records in reverse order:

```
DD01 PRINTHEXREV MAXRECOUT=10
```

SKIP and SKIPREV

SKIP (**S**) moves the current record pointer forward, and **SKIPREV** (**SKR**) moves it backward. This allows large groups of records to be excluded from processing.

When using these functions, remember the following:

- You may not follow a SKIP function with an UPDATEREC function because the data set must be closed and reopened while maintaining data set positioning.
- You cannot follow a SKIP function with a reverse function like COPYREV.
- You cannot follow a SKIPREV function with a forward operation like COPYREC.

The following example skips the first *100* records before a PRINT function:

```
DD01 SKIP MAXRECIN=100
DD01 PRINT IF=(10,EQ,C'RED')
```

The following example skips the first *100* records before an UPDATEREC function:

```
DD01 UPDATEREC  MAXRECIN=100
DD01 UPDATEREC OVERLAY=(10,EQ,C'RED',C'BLU')
```

The following example skips the last *100* records before a PRINTREV function:

```
DD01 SKIPREV MAXRECIN=100
DD01 PRINTREV IF=(10,EQ,C'RED')
```

TOTAL

TOTAL (**T)** reads input records processing all parameter groups for SUM. The TOTAL function creates simple reports to validate data in a data set. The results generated from the TOTAL function and the SUM parameter are directed to the SYSTOTAL DDNAME if it is present; otherwise, the results are directed to the SYSPRINT data set. You can add comment statements to the input control statements; they will be printed with the TOTAL function results.

StarBat produces a two-line data set identification message after the comment and before the accumulation with the following output identifier:

FOLLOWING TOTALS DEVELOPED FROM for the first line
the input data set name and volume name on the second line.

The following example accumulates all packed numbers beginning in column *43*:

```
DD01 TOTAL  SUM=(43,'This is the packed data set total')
```

The following example accumulates all character numbers beginning in column *43* for type *M* records:

```
DD01 TOTAL  IF=(22,EQ,C'M'),SUM=(43,6,C,'This is the character data
set total')
```

UPDATEREC, UPDATEMBR, and UPDATEALL

UPDATEREC (**UR**) updates records in place. Combine this function with any of the parameters used to modify data such as the CHANGE and OVERLAY parameters.

NOTE This updates the data set in place. To preview your changes before committing them, use the PRINT functions instead to show the changes that will be made and then re-run the job replacing the PRINT function with the UPDATEREC function.

UPDATEMBR (**UM**) updates members based on their contents.

UPDATEALL (**UA**) processes multiple conditional updates (IF OVERLAY, IF CHANGE or IF WARP) while processing all records.

The following example shows potential changes before performing an actual UPDATEREC:

```
DD01 PRINT  OVERLAY=(1,0,C'RED',C'BLU')
```

The following example changes the first occurrence of *RED* anywhere in a record to *BLU*:

```
DD01 UPDATEREC OVERLAY=(1,0,C'RED',C'BLU')
```

The following example changes all occurrences of *RED* to *BLU* and all occurrences of *WHI* to *ANY*:

```
DD01 UPDATEALL OVERALL=(1,0,C'RED',C'BLU'),
                OVERALL=(1,0,C'WHI',C'ANY')
```

Chapter 5

Parameters

Specify parameters on function statements to limit the data processed and to control the function.

Descriptions of the parameters, in alphabetical order, are described in the table below.

Parameter	Short	Description
ABEND	**AB**	Controls EOJ processing when an abnormal condition occurs
AND	**IF**	Creates a logical AND condition check (used with IF)
CHANGE	**C**	Changes only the first instance of data in a record
CHANGEALL	**CA**	Changes all occurrences of data in a record
COPYOVER	**CO**	Controls the replacement of identically named output members
EXCLUDEREC	**XR**	Controls the # of records to bypass in an EXCLUDEREC function
EXPAND	**EX**	Expands records at a specified location
IF	**AND**	Selects records to process based on data contents
MAP		Defines how lines are printed in the COPYBOOK format. The listing output is sent to the SYSLIST DD.
MAXRECIN	**MRI**	Controls the number of records to input
MAXRECOUT	**MRO**	Controls the maximum number of records to output
MEMBER	**M**	Specifies a member name to process in a PDS
MEMBERS	**MS**	Specifies a group of members to process in a PDS with a mask
MOVE	**MV**	Moves data into the record
NEWMBR	**NM**	Gives a new name to an output PDS member
NEWMBRS	**NMS**	Names multiple new members of an output PDS using a mask
OPTIONS	**OP**	Controls StarBat processing options
OR	**(none)**	Used with the IF parameter to indicate an OR condition
OVERALL	**OA**	Replaces all occurrences of data in a record with other data
OVERLAY	**OL**	Replaces the first instance of data in a record with new data
PADCHAR	**PAD**	Specifies a padding character for uninitialized parts of a record

Parameter	Short	Description
PRINT	**P**	Prints records in alphanumeric format with record statistics
PRINTCHR	**PC**	Prints records in alphanumeric format
PRINTHEX	**PH**	Prints records in vertical hexadecimal format
PRINTLPI	**PL**	Specifies the number of lines per inch for print output pages
RBA	**(none)**	Processes VSAM data beginning at a relative byte address
RDW	**(none)**	Controls the inclusion of the record descriptor word
SELECT	**S**	Processes every nth record
STARTKEY	**SK**	Processes VSAM data beginning with a generic key
STOPIF	**ST**	Stops processing a function when a record satisfies a condition
SUM	**(none)**	Accumulates the contents of specified fields
WRITE	**W**	Writes a record to one or more output files

Parameters are grouped according to type, as follows:

Action	Changes data (CHANGE, EXPAND, MOVE, OVERLAY, OVERALL, SUM, WARP and WRITE)
Limit	Specifies record count limits (EXCLUDEREC, MAXRECIN, MAXRECOUT and SELECT)
Print	Prints records as they are processed (PRINT, PRINTCHR and PRINTHEX).
Selection	Selects records based on their contents (AND, IF, and OR)
Control	Defines basic conditions during execution (all other parameters such as ABEND and STOPIF)

There are several restrictions on parameter and function combinations:

- Use the EXCLUDEREC parameter only with the EXCLUDEREC function.
- Use the WRITE parameter only with the MULTICOPY function.
- Do not use the PRINTHEX, PRINTCHR and PRINT parameters with the PRINT functions.
- Do not use the EXPAND, MOVE and MAXRECOUT parameters with the UPDATE functions.
- Do not use the NEWMBR, NEWMBRS and COPYOVER parameters with PRINT functions, SKIP**,** SKIPREV or UPDATEREC.

Many StarBat parameters process numeric character data and packed decimal data. Numeric characters must be in zoned decimal (hexadecimal X'F0' through X'F9') and the last byte can be signed positive (X'C0'), negative (X'D0') or unsigned (X'F0').

Packed decimal numbers must contain valid numeric numbers and the sign must be positive (with either X'0C' or X'0F') or negative (X'0D'). In most cases, you can enter a data length of 0 for packed decimal numbers and StarBat will determine the length dynamically by scanning for the sign digit in each field as it is processing each record.

ABEND

The ABEND (AB) parameter controls how the end of job processing should be handled when an abnormal condition occurs during job execution.

```
ABEND=0/1/2
```

ABEND=0	Issues a return code at normal end of job.
ABEND=1	Issues a U0012 ABEND dump when an I/O error occurs. This is the default.
ABEND=2	Issues a user ABEND when a non-zero return code is encountered.

In the following example, a user ABEND of U0008 occurs if the character string `*ALIAS-*` is not found in the input data set and no records are written to the output data set.

```
DD01 MULTICOPY ABEND=2,IF=(1,0,C'ALIAS-').MOVE=(19,0,80),WRITE=FILE1
```

AND

The AND parameter selects records to be processed by the function being executed. The AND parameter and the IF parameter have identical meanings and format. Normally, AND is used after an IF parameter to improve readability.

CHANGE

The CHANGE (C) parameter works like the CHANGE command in ISPF edit. It replaces the value in *string-1* with the value in *string-2*. *String-1* and *string-2* can be different lengths and the data will be shifted left or right depending on the size of the second string (see NOTE: below). CHANGE only modifies the first occurrence of the matching data in the record, as with ISPF. To change multiple occurrences, use the similar CHANGEALL parameter.

CHANGE=(*start,length/operator,string-1,string-2*)

start	defines the position of the record where the search is to begin. The first byte of the record is position 1.
length	specifies how many characters to search. Specify 0 if you want to search the remainder of the record.
operator	an alternative to length is to specify a relational operator (such as EQ).

string-1	is a binary, character, hexadecimal or packed string representing data to be searched for at this location.
string-2	is a binary, character, hexadecimal or packed string representing data to be replaced at this location.

NOTE When CHANGE replaces a data field by a shorter field, StarBat shifts contiguous characters to the left until a blank character is found and inserts blanks to adjust for the missing characters. If no blanks are found, blanks are inserted at the end of the record.

For example:

CHANGE=(1,5,C'1234',C'X')

```
Record 1: 12345 AB1234GH
Record 2: 123456ABCDEFGH
```

becomes:

```
Record 1: X5    AB1234GH
Record 2: X56ABCDEFGH
```

When CHANGE replaces a data field by a longer field, StarBat shifts contiguous characters to the right compressing multiple blanks to single blanks until all characters fit into the record.

NOTE If no blanks are found using the COPYREC or MULTICOPY function for fixed length output records or using the UPDATE functions, truncation occurs if the data extends beyond the record boundary.

For example:

CHANGE=(11,3,C'USA',C'DOMESTIC')

```
Record 1: AA19980101USA       CA94010-1904
Record 2: AA19880101USA  CA94010-1904
```

becomes:

```
Record 1: AA19980101DOMESTIC  CA94010-1904
Record 2: AA19880101DOMESTIC CA94010-1904
```

It is important to remember when using the CHANGE (or OVERLAY) parameters that they operate on the input data records; therefore, specify any MOVE functions after the CHANGE (or OVERLAY) parameter because MOVE operates on the output record.

CHANGEALL

The CHANGEALL (CA) parameter is the same as the CHANGE parameter except that all occurrences are modified and the second parameter must be a length value (not an operator like EQ). See "CHANGE" on page 35 for additional details.

`CHANGEALL=(start,length,string-1,string-2)`

start	defines the position of the record where the search is to begin. The first byte of the record is position 1.

length	specifies how many characters to search. Specify 0 if you want to search the remainder of the record.
string-1	is a binary, character, hexadecimal or packed string representing data to be searched for at this location.
string-2	is a binary, character, hexadecimal or packed string representing data to be replaced at this location.

COPYOVER

Use the COPYOVER (CO) parameter to specify if identically named output members are to be replaced when copying from one PDS to another.

```
COPYOVER=YES/NO
```

COPYOVER=YES	Default, replaces the output PDS member if it already exists.
COPYOVER=NO	Do not replace the output PDS member if it already exists.

The following example specifies that existing members are not to be replaced:

```
DD01 COPYREC COPYOVER=NO
```

EXCLUDEREC

Use the EXCLUDEREC (XR) parameter with the EXCLUDEREC function to control the number of records to exclude.

```
EXCLUDEREC=n
```

where n is a number from 1 to 999999999. It represents the number of records to be excluded.

The following example replaces all occurrences of '*MAIL*' with '*IMA1*' and omits the first record that contains '*IMA1-M*'.

```
DD01 EXCLUDEREC OVERALL=(1,0,C'MAIL',C'IMA1'),IF=(1,EQ, C'IMA1-
M'),EXCLUDEREC=1
```

EXPAND

Use the EXPAND (EX) parameter to expand date and data fields. EXPAND is similar to MOVE but the original record is preserved to the extent possible. Normally, Character type is used for date expansions of any format because after a date is padded on the right with blanks, StarBat can still read it using its input picture and rewrite it with a new picture.

Column numbers referred to by EXPAND are the original column numbers in the data; similarly, references to column numbers by other StarBat parameters should also reference the original column numbers because StarBat adjusts these column numbers dynamically.

EXPAND=(*column,type,size,newsize*)

column	is the location where the data item begins. Any valid actual or relative location may be used.
type	is B for binary, P for packed, C for character or N for character numeric. Character type pads the original item on the right with blanks; other types of field expansion add numeric padding to the left.
size	is the number of bytes used by the original data item.
newsize	is the number of bytes to be used by this data item. Newsize can be smaller or larger than size.

The following example expands a packed data field and accumulates the next character field:

```
DD01 COPYREC  EXPAND=(21,P,4,8),SUM=(25,5,C,'The next field')
```

The following example expands a date field and converts a character date format from YYDDD to CCYYDDD:

```
DD01 COPYREC  EXPAND=(21,C,5,7),WARP=(21,C,YYDDD,OUTPIC=CCYYDDD)
```

The following example expands a date field and converts a packed date format from YYDDD to CCYYDDD:

```
DD01 COPYREC  EXPAND=(21,C,3,4),WARP=(21,P,YYDDD,OUTPIC=CCYYDDD)
```

The following example expands a binary field and a numeric character field and accumulates the fields:

```
DD01 COPYREC
     EXPAND=(12,B,2,4),EXPAND=(21,N,4,9),SUM=(12,4,B),SUM=(21,9,C)
```

IF

The IF parameter selects records to be processed by the function being executed. The IF parameter and the AND parameter have identical meanings and format. Normally, AND is used after an IF parameter to improve readability.

There are two IF syntax forms to test for data contents or valid numerics:

```
IF=(start,length/operator,string,...)     /* data content test   */
IF=(start,length,[duplicate]type,...)        /* valid numeric test*/
```

start	defines the position of the record where the search is to begin. The first byte of the record is position 1.
length	specifies how many characters to search. Specify 0 if you want to search the remainder of the record.
operator	an alternative to length is to specify a relational operator (such as EQ).

string	an alternative to length is to specify a relational operator (such as EQ).is a binary, character, hexadecimal or packed string representing data to be searched for at this location.
[*duplicate*] type	is an optional duplication number followed by EQN and EQP to check for valid numeric character or packed decimal or NEN and NEP to check for invalid numeric character or packed decimal.

NOTE You can enter multiple parameter sets in an IF parameter separated by commas to obtain multiple logical OR tests.

In the following example, all records without a hex *'F0F1'* in column 1 are written to file *FILE1.*

```
DD01 MULTICOPY IF=(1,NE,X'F0F1'),WRITE=FILE1
```

The following example copies all records, but in those records having an '*ME*' in position 1 AND a '*QE*' in column 3, position 10 is replaced with a '*P*'.

```
DD01 COPYALL IF=(1,EQ,T'ME'),IF=(3,EQ,C'QE'),OVERLAY=(10,C'P')
```

The example above can also be coded using the AND parameter:

```
DD01 COPYALL IF=(1,EQ,T'ME'),AND=(3,EQ,C'QE'),OVERLAY=(10,C'P')
```

The following example directly updates the input data set. If the data at position *60* for a length of *5* is not numeric, position *60* is replaced with five zeros.

```
DD01 UPDATEREC IF=(60,5,NEN),OVERLAY=(60,C'00000')
```

In the following example, all records with invalid packed numbers are copied to the output data set:

```
DD01 COPYREC IF=(44,0,NEP),PRINTHEX=4
```

This example copies input records with a *1* in position *10* and a *2* in position *20*, or an *A* in position *3* and a *B* in position *6*. Since it is a COPYREC, only position *30* is warped; COPYSOME or COPYALL warps both positions.

```
DD01 COPYREC
IF=(10,EQ,C'1'),AND=(20,EQ,C'2'),WARP=(30,C,CCYYMMDD,ADD=5Y),
PRINT=3,

IF=(3,EQ,C'A'),AND=(6,EQ,C'B'),WARP=(40,C,CCYYMMDD,ADD=10D),
MAXRECOUT=30
```

The following example prints, in vertical hex, all records that contain odd EBCDIC numbers in column *14*:

```
DD01 PRINTHEX IF=(14,AO,X'F1')
```

The following example prints, in vertical hex, all records that contain even EBCDIC numbers in column *14*:

```
DD01 PRINTHEX IF=(14,GE,X'F0'),IF=(14,NO,X'01')
```

The following example prints, in vertical hex, all records that contain an EBCDIC *0, 4* or *8* in column *14*:

```
DD01 PRINTHEX IF=(14,AO,X'F0'),IF=(14,AZ,X'03')
```

MAP

The MAP=() parameter is used to define how lines are printed in the COPYBOOK format. The listing output is sent to the SYSLIST DD.

The LRECL of the file limits the output line size to 160 characters.

```
DDnn FPRINT MAP=(copymbr {, OptionA{, OptionB} }

{,SHOW=(..., ...)}

{,HIDE=(..., ...)}

{,FORMAT=(..., ...)} )
```

copymbr	Specifies the Copybook member name. It is from 1 to 10 characters depending on the library used, and is read from the COPYBOOK DD.
OptionA	Specifies the default language type: ASSembler, COBol, or PL1. Only the first three characters are required. If not set, the type is set based on the records read.
OptionB	Defines the type of library: MVS, PANvalet or LIBrarian. Only the first three letters are required. MVS PDS/PDSE is assumed if nothing is entered.

FORMAT	Specifies the following options. If not used, the column headings contain the default values as listed below. DEFAULT sets the header to NAMES, PICT, TYPE, COL, NUM. NAMES|NONAMES controls the display of column name. PICT|NOPICT controls the display of the column's picture. TYPE|NOTYPE controls the display of the column's type. COL|NOCOL controls the display of the column's relative number, starting at 01. NUM|NONUM controls the display of the record number column. LEV1|NOLEV1 controls whether the LEVEL 1 variable is displayed. LEV88|NOLEV88 controls whether LEVEL 88 items are displayed. HEX|NOHEX controls an additional two lines of the HEX value of the data fields below the formatted data.
HIDE	Suppresses column printing. This is followed by 1 or more column names or column numbers, or the ALL option.
SHOW	Selects hidden columns for printing. This is followed by 1 or more column names or column numbers, or the ALL option.

MAXRECIN

The MAXRECIN (MRI) parameter specifies the number of input records to be read before terminating function processing.

```
MAXRECIN=n
```

where n is the maximum records to be input. Any number from 0 to 999999999 can be used where 0 is all records.

The following example copies the first *100* records from the input data set to the output data set:

```
DD01 COPYREC MAXRECIN=100
```

MAXRECOUT

The MAXRECOUT (MRO) parameter controls the number or records to be printed or written before processing stops. Use MAXRECOUT to limit the number of output records or

extend the number of printed records past the default of 250. The MAXRECOUT parameter is ignored for the update functions.

```
MAXRECOUT=n
```

where n is the maximum records to output. Any number from 0 through 999999999 may be used where 0 is all records.

The following example prints, in vertical hex, the first *30* records that contain the string TEST RECORD in column *23*:

```
DD01 PRINTHEX IF=(23,EQ,C'TEST RECORD'),MAXRECOUT=30
```

MEMBER

The MEMBER (M) parameter specifies a member name to process in a PDS. If neither MEMBER nor MEMBERS is specified for a PDS, all members of the data set will be processed.

```
MEMBER=memname/(member1,member2,…)
```

memname	specifies the member to be processed. To specify multiple members, use parentheses with MEMBER; use the MEMBERS parameter; or use multiple control statements with MEMBER and the same DD*xx* name.

The following example copies a single member.

```
DD01 COPYREC MEMBER=MEMBERA
```

The following example prints three different members in vertical hex.

```
DD01 PRINTHEX MEMBER=MEMBERA
DD01 PRINTHEX MEMBER=MEMBERB
DD01 PRINTHEX MEMBER=MEMBERC
```

The following example prints three different members in vertical hex.

```
DD01 PRINTHEX MEMBER=(MEMBERA,MEMBERB,MEMBERC)
```

MEMBERS

The MEMBERS (MS) parameter specifies a group of members to process in a PDS with a mask. If neither MEMBER nor MEMBERS is specified for a PDS, all members of the data set will be processed.

```
MEMBERS=ALL/maskname
```

ALL	specifies that all members are to be selected.

maskname	specifies the members to be processed using a mask of up to eight characters which is matched with member names from the data set. Several mask examples are shown below; for more information, see *"Member Name Forms" in the StarBat Command Reference*.

The following example copies all members whose names begin with *STAR*.

```
DD01 COPYREC MEMBERS=STAR
```

Dash (--)	copy all members whose names begin with STAR and contain 98 in positions seven and eight.

```
DD01 COPYREC MEMBERS=STAR--98
```

Combination (*)	copy all members whose names begin with STAR and contain 98 elsewhere.

```
DD01 COPYREC MEMBERS=STAR*98
```

Range (:)	copy all members in the alphabetic range whose names begin with *ABC* through *DEF999999*.

```
DD01 COPYREC MEMBERS=ABC:DEF
```

Pattern (/)	copy all members whose names contain *STAR* and *98* anywhere.

```
DD01 COPYREC MEMBERS=STAR/98
```

MOVE

The MOVE (MV) parameter builds an output record by moving data to it. You can use data from control statements or from input records depending on which form of the MOVE parameter you use.

NOTE MOVE cannot be used with the update functions.

There are two MOVE syntax forms that move data from a control statement or from an input record:

```
MOVE=(toposition,data)           Comment: data from control statement
MOVE=(toposition,length,from-loc)   Comment: data from input record
```

toposition	defines the starting position in the output record to which the data is to be moved. Absolute and relative positions can be used. 'n' means at position 'n', '+n' means 'n' bytes further down the record, '-n' means 'n' bytes previous in the record and '+0' means in the next available position.

data	is a binary, character, hexadecimal or packed string representing data to be moved to the output record.
length	defines the number of bytes to be moved. The number can be in the range 1 to 255. A value of 0 indicates that the remainder of the input record is to be moved.
fromposition	defines the starting position in the input record from which the data is to be moved. Absolute and relative positions can be used as for toposition; '+0' references the input relative location or the last scan position.

The output record area is initially set to X'00' by default but this can be overridden by the PADCHAR parameter. A data value set in the output record remains set until new data is placed there; the output record does not need to be reinitialized.

The following example moves *REDRED* to the first six positions of an output record.

```
DD01 COPYREC MOVE=(1,2C'RED')
```

The following example prints the first *100* characters of each input record.

```
DD01 PRINT MOVE=(+0,100,+0)
```

The following example substitutes *RED* for the first three positions of each record.

```
DD01 COPYREC MOVE=(1,C'RED'),MOVE=(+0,0,4)
```

NEWMBR

NEWMBR (NM) names a new member of an output PDS.

```
NEWMBR=memname
```

where memname specifies a new name to be assigned to the copied member.

The following example copies *MEMBERA* to an output data set and rename it to *MEMBERCC*.

```
DD01 COPYREC MEMBER=MEMBERA,NEWMBR=MEMBERCC
```

An existing member is replaced unless you specify the COPYOVER=NO parameter.

```
DD01 COPYREC MEMBER=MEMBERA,NEWMBR=MEMBERCC,COPYOVER=NO
```

NEWMBRS

NEWMBRS (NMS) assigns new names to a group of output PDS members.using a mask.

```
NEWMBRS=maskname
```

maskname	*maskname* can be from 1 to 8 characters long. Characters in the member name are replaced by characters from the maskname unless the character in the maskname contains a minus sign; the character in that position will be retained. Any blanks in the resulting member names are discarded.

The following example selects all members prefixed with TST and renames them to SYSxxA. For example, member *TST3333* is renamed to *SYS33A3*. Members that contain the same name in the output data set will not be replaced.

```
DD01 COPYMBR MEMBERS=TST,NEWMBRS=SYS--A,COPYOVER=NO
```

This example copies those members prefixed with TST and containing '*ALPHA*' in column 1 in the data, and renames the selected members to PARM…

```
DD01 COPYMBR MEMBERS=TST,IF=(1,EQ,C'ALPHA'),NEWMBRS=PARM
```

OPTIONS

OPTIONS (OP) controls StarBat processing options for file I/O and print functions.

```
OPTIONS=LONG/SHORT/JCL/MULTI/NOMULTI
```

OPTIONS=LONG	default, a column scale is to be printed for PRINT and PRINTHEX after each data record.
OPTIONS=SHORT	a column scale is only to be printed at the top of each PRINT, PRINTCHR and PRINTHEX page.
OPTIONS=XPAGE	Print output in SHORT form and suppress page breaks. Applies to PRINT, PRINTALL, PRINTMBR, PRINTCHR, PRINTCHRALL, PRINTCHRMBR, PRINTHEX, PRINTHEXALL, and PRINTHEXMBR.
OPTIONS=JCL	logical JCL processing for DD, EXEC, JOB, PROC and SET JCL statements.
OPTIONS=MULTI	after an end of file condition, a function with the same DDNAME is to reread the data set.
OPTIONS=NOMULTI	default, after an end of file condition, another read notes another end of data set condition.

The following example prints a column scale only on each page.

```
DD01 COPYREC OPTIONS=SHORT,MEMBER=MEMBERA,PRINTHEX=500
```

The following example uses logical JCL processing for this control statement group.

```
DD01 COPYREC OPTIONS=JCL,IF=(1,0,C'VOL=SER='),
     CHANGE=(1,0,C'UNIT=TAPE3',C'UNIT=TAPE77')
```

The following example causes StarBat to reread the data set if the end of the data set is encountered.

```
DD01 COPYREC OPTIONS=MULTI,PRINTHEX=500
```

OR

Provides for those situations where a number of conditions need to be grouped together with the OR operator. There are two OR syntax forms to test for data contents or valid numerics:

```
OR=(start,length/operator,string,...)     /* data content test   */
OR=(start,length,[duplicate]type,...)     /* valid numeric test  */
```

start	defines the position of the record where the search is to begin. The first byte of the record is position 1.
length	specifies how many characters to search. Specify 0 if you want to search the remainder of the record.
operator	an alternative to length is to specify a relational operator (such as EQ).
string	is a binary, character, hexadecimal or packed string representing data to be searched for at this location.
[*duplicate*]type	is an optional duplication number followed by EQN and EQP to check for valid numeric character or packed decimal or NEN and NEP to check for invalid numeric character or packed decimal.

NOTE You can enter multiple parameter sets separated by commas in an OR parameter to obtain multiple logical OR tests.

In this example, if position *40* contains *'COPYRIGHT'* or *'247B'* anywhere from column *2* through the end of the record, copy the record to the file identified in the JCL by *FILEA*.

```
DD01 MULTICOPY IF=(40,EQ,C'COPYRIGHT'),OR=(2,0,C'247B'),WRITE=FILEA
```

This example copies all records that contain a packed *'117400477'* or *'516988271'* in position *11*.

```
DD01 COPYREC IF=(11,EQ,{'117400477'),OR=(11,EQ,P'516988271'),
MAXRECOUT=0
```

This example copies all records, but if column 1 contains a hex *'22722D'* OR a hex *'28888C'*, replace column 1 with a hex *'99999C'*.

```
DD01 COPYALL
IF=(1,EQ,X'22722D',1,EQ'X'28888C'),OVERLAY=(1,X'99999C')
```

The example above is equivalent to:

```
DD01 COPYALL
IF=(1,EQ,X'22722D'),OR=(1,EQ,X'28888C'),OVERLAY=(1,X'99999C')
```

The following example copies all type *1* records or records with invalid packed numbers to the output data set:

```
DD01 COPYREC IF=(12,EQ,C'1'),OR=(44,0,NEP),PRINTHEX=4
```

This example copies input records with a *1* in position *10* and a *2* in position *20* or an *A* in position *3* and a *B* in position *6.* Since it is a COPYREC, only position *30* is warped; COPYSOME or COPYALL warps both positions.

```
DD01 COPYREC IF=(10,EQ,C'1'),AND=(20,EQ,C'2'),
WARP=(30,C,CCYYMMDD,ADD=5Y),PRINT=3,

IF=(3,EQ,C'A'),AND=(6,EQ,C'B'),WARP=(40,C,CCYYMMDD,ADD=10D),
MAXRECOUT=30
```

OVERLAY

The OVERLAY (OL) parameter replaces data at a particular location with new data in a variety of ways. Data in one location can be overlaid with literal data unconditionally, according to some conditional data content in a location, or according to some conditional data content in another separate location.

There are three OVERLAY syntax forms:

- Replace by location, as in

  OVERLAY=(*position,newstring*)

- Replace by condition, as in

  OVERLAY=(*position,length/operator,string-1,string-2*)

- Replace at alternate location, as in

  OVERLAY=(*position,length/operator,string-1,toposition,string-2*)

position	defines the starting position in the output record to which the data is to be moved. Absolute and relative positions can be used. 'n' means at position 'n', '+n' means 'n' bytes further down the record, '-n' means 'n' bytes previous in the record and '+0' means in the next available position.
newstring	is a binary, character, hexadecimal or packed string representing data to be placed in the specified location.
length	specifies how many characters to search. Specify 0 if you want to search the remainder of the record.
operator	an alternative to length is to specify a relational operator (such as EQ).
string-1	is a binary, character, hexadecimal or packed string representing data to be searched for at this location.
toposition	defines the starting position in the output record to which the data is to be moved. Absolute and relative positions can be used. 'n' means at position 'n', '+n' means 'n' bytes further down the record, '-n' means 'n' bytes previous in the record and '+0' means in the next available position.
string-2	is a binary, character, hexadecimal or packed string representing data to be replaced at this location.

NOTE OVERLAY replaces existing data with the new data at the specified location without shifting data.

Indicate bit operations by specifying a modifier as the first character of string-2; O indicates an OR, M indicates NOT-AND or binary minus, A indicates AND and E indicates an exclusive OR.

The following example copies all records and places *REDRED* in the first 6 positions of each record.

```
DD01 COPYREC OVERLAY=(1,2C'RED')
```

The following example replaces the first A found in column *10* through *19* with a *B*.

```
DD01 COPYREC OVERLAY=(10,10,C'A',C'B')
```

The following example finds *RED* beginning in position *6* and overlay position *10* with an asterisk.

```
DD01 COPYREC OVERLAY=(6,EQ,C'RED',10,C'*')
```

The following example clears the zone digits where ABCD is found with an AND bit operation.

```
DD01 COPYREC OVERLAY=(23,20,C'ABCD',AX'0F0F0F0F')
```

OVERALL

The OVERALL (OA) parameter replaces data at a particular location in the same way as OVERLAY does; however, OVERALL replaces all occurrences of the string and it only supports a length field. For more information, see "OVERLAY" on page 47.

There are two OVERALL syntax forms:

- Replace by condition, as in

  OVERALL=(*position,length,string-1,string-2*)

- Replace at alternate location, as in

  OVERALL=(*position,length,string-1,toposition,string-2*)

position	defines the starting position in the output record to which the data is to be moved. Absolute and relative positions can be used. 'n' means at position 'n', '+n' means 'n' bytes further down the record, '-n' means 'n' bytes previous in the record and '+0' means in the next available position.
length	specifies how many characters to search. Specify 0 if you want to search the remainder of the record.
string-1	is a binary, character, hexadecimal or packed string representing data to be searched for at this location.

topositio n	defines the starting position in the output record to which the data is to be moved. Absolute and relative positions can be used. 'n' means at position 'n', '+n' means 'n' bytes further down the record, '-n' means 'n' bytes previous in the record and '+0' means in the next available position.
string-2	is a binary, character, hexadecimal or packed string representing data to be replaced at this location.

The following example replaces all *REDs* found in columns *10* through *19* with *BLU*s.

```
DD01 COPYREC OVERALL=(10,10,C'RED',C'BLU')
```

PADCHAR

Use the PADCHAR (PAD) parameter to initialize an output area with the specified character. The default is binary zeros (X'00').

PADCHAR=C'*c*'/X'*nn*'

C'*c*'	*c* is any single character value.
X'*nn*'	*nn* is any valid hexadecimal value.

If a record has *10* blanks beginning in position *11*, move the indicated data and pad the end of the records with periods.

```
DD01 COPYREC IF=(11,EQ,10C' '),MOVE=(1,5,28),PADCHAR=C'.'
```

PRINT

The PRINT (P) parameter prints a specified number of records in a simple character format.

NOTE The PRINT parameter is not intended for use with the print functions; use it with any of the other functions that process the data set.

PRINT=*n*

where n is any number from 0 to 999999999. Specify 0 to print all records.

The following example prints the first *10* records that contain the string *RED* anywhere.

```
DD01 COPYREC IF=(1,0,C'RED'),PRINT=10
```

PRINTCHR

The PRINTCHR (PC) parameter prints a specified number of records in an alphanumeric format.

NOTE PRINTCHR is not intended for use with the print functions; use it with any of the other functions that process the data set.

```
PRINTCHR=n
```

where n any number from 0 to 999999999 can be used. Specify 0 to print all records.

The following example prints the first *10* records that contain the string *RED* anywhere.

```
DD01 COPYREC IF=(1,0,C'RED'),PRINTCHR=10
```

PRINTHEX

The PRINTHEX (PH) parameter prints a specified number of records in vertical hexadecimal format.

NOTE PRINTHEX is not intended for use with the print functions; use it with any of the other functions that process the data set.

```
PRINTHEX=n
```

where n is any number from 0 to 999999999. Specify 0 to print all records.

The following example prints in vertical hex the first *10* records that contain the string *RED* anywhere.

```
DD01 COPYREC IF=(1,0,C'RED'),PRINTHEX=10
```

PRINTLPI

The PRINTLPI (PL) parameter controls the number of lines written to a page for the SYSLIST output data set.

```
PRINTLPI=6/8
```

PRINTLPI=6	default, specifies 6 lines to an inch and allows up to 58 lines in a SYSLIST output.
PRINTLPI=8	specifies 8 lines to an inch and allows up to 78 lines in a SYSLIST output.

The following example prints the first *10* records containing the string *RED* with as many as *78* lines on a page.

```
DD01 PRINTHEX IF=(1,0,C'RED'),PRINTCHR=10,PRINTLPI=8
```

RBA

Use the RBA parameter for VSAM data sets. For ESDS and KSDS, this defines the Relative Byte Address. For RRDS, this defines the Relative Record Number.

RBA=*nn*/X'*nn*'

nn	*nn* is any decimal value
X'nn'	*nn* is any valid hexadecimal value.

The example below lists records beginning at relative byte address *800, which can also be specified* in hex as X'*320*'.

```
DD01 PRINTCHR RBA=800
```

RDW

The RDW (Record Descriptor Word) parameter specifies how to address the RECFM=V RDW word.

RDW=0/1/2/3

If the RDW parameters that you have coded in JCL do not work as you expect after you have installed FDM 7.7.1.03 or later releases, you will need to change the JCL control parameters for StarBat. The RDW parameter controls how and if the RDW in the records is processed.

NOTE The RDW parameter does not affect the processing of fixed-length or undefined input records.

If input records are of variable length, the records are moved to a processing area as they are read and padded with the fill character as needed. The RDW affects processing as follows:

RDW Value	Processing
0	The RDW is included in the record that is passed to the action routine. (Action routines change data, and include the processing that results from CHANGE, EXPAND, MOVE, OVERLAY, OVERALL, SUM, WARP, and WRITE parameters.) Therefore, you must code *start*, *to-position*, and *from-location* values that take the RDW (4 bytes in each record) into account. That is, the *start*, *to-position*, and *from-location* in the IF and MOVE statements must account for the RDW: • IF=(*start,length/operator,string*,...) • MOVE=(*to-position,length,from-location*) If RDW=0, the RDW will be included in a PRINT output request. (RDW=0 is the default.)
1	Same as RDW=0, but the RDW will not be included in a PRINT output request.

RDW Value	Processing
2	The RDW is not included in the record that is passed to the action routine. Code *location* values without taking the RDW into account. The RDW will not be included in a PRINT output request. RDW=2 does not affect the default output data set records or the MULTICOPY WRITE= data set records.
3	Same as RDW=2. RDW=3 is kept for compatibility purposes only.

If RDW=0 or 1, the variable input record has the RDW. The output record will also have the RDW and the *start*, *to-position*, and *from-location* values used in the commands must account for it. The output record format determines the use of the RDW when the record is written, as follows:

- For variable output records, the RDW is used as is, but is recomputed based on the output record's length.
- For fixed output records, the RDW is removed and the record is written with the fill character as needed.
- For undefined output records, the RDW is used to set the length of the record, but is removed from the record.

If variable input records do not have the RDW (RDW=2 or 3), the length of the output record is based on the length of the input record if the length of the output record has not been modified by a MOVE or EXPAND parameter. If the length of the output record has been modified by a MOVE or EXPAND parameter, its length is based on the last byte that has been moved.

SELECT

Use the SELECT (S)parameter to select every nth occurrence of a record for processing.

```
SELECT=n
```

where n is the number of records in an interval. Any number from 1 to 999999999 can be used.

The following example copies all records and tests every third record for a '*PBCE*' in column 3. If a match is found, '*PBCE*' is replaced by four zeroes.

```
DD01 COPYALL SELECT=3,OVERLAY=(3,EQ,C'PBCE',C'0000')
```

The following example only processes records containing '*Z145*' in column 3. Every third record satisfying the conditional statement is copied; '*Z145*' is replaced with '*ZZZZ*'.

```
DD01 COPYREC IF=(3,EQ,C'Z145'),SELECT=3,OVERLAY=(3,C'ZZZZ')
```

In this example, every third record is selected, and if the selected record contains a '*PBCE*' in column *3*, copy the record and replace '*PBCE'* with *'ZZZZ*'.

```
DD01 COPYREC SELECT=3,IF=(3,EQ,C'PBCE'),OVERLAY=(3,C'ZZZZ')
```

STARTKEY

Specify the STARTKEY (SK) parameter to initiate VSAM KSDS processing at a specific or generic key.

```
STARTKEY=C'cc'/X'hh'
```

STARTKEY=C'*cc*'	is any character string that specifies the beginning KSDS key value.
STARTKEY=X'*hh*'	is any hexadecimal string that specifies the beginning KSDS key value.

The following example copies VSAM records after positioning to generic key *RED*:

```
DD01 COPYREC STARTKEY=C'RED'
```

STOPIF

Use the STOPIF (ST) parameter to terminate processing before the end of data. STOPIF ends execution of the current process if the condition associated with the parameter is satisfied.

```
STOPIF=(start,length/operator,string,...)
```

start	defines the position of the record where the search is to begin. The first byte of the record is position 1.
length	specifies how many characters to search. Specify 0 if you want to search the remainder of the record.
operator	an alternative to length is to specify a relational operator (such as EQ).
string	is a binary, character, hexadecimal or packed string representing data to be searched for at this location.

NOTE You can code multiple conditions within one STOPIF parameter; these are evaluated using logical OR tests. Multiple contiguous STOPIF parameters result in logical AND tests for STOPIF conditions.

The following example copies all records until '*RECORDS READ*' is found in position 8. The record containing '*RECORDS READ*' is not copied.

```
DD01 COPYREC STOPIF=(8,EQ,C'RECORDS READ')
```

In this example, the first DD01 control statement copies records until *VOL001* is found in column 1. The second DD01 control statement begins with the *VOL001* record and stops copying records with *VOL300*.

```
DD01 COPYREC STOPIF=(1,EQ,C'VOL001')
DD01 COPYREC STOPIF=(1,EQ,C'VOL300')
```

The following example skips to the record containing a hex '*033198*' in column 1 or '*IMAGE*' in column 12. This record and all records after it are copied.

```
DD01 SKIP STOPIF=(1,EQ,X'033198',12,EQ,C'IMAGE')
DD01 COPYREC
```

SUM

Use the SUM parameter to add up any fields containing character, packed, or binary numbers. If there is a SYSTOTAL data set, then the totals go to that data set. Otherwise, they go to the SYSPRINT data set

```
SUM=(position[,'description])     /* for packed decimal data      */

SUM=(position,length,datatype[,'description'])  /* for character or
binary data */
```

position	The beginning position of the data. This can be an actual or a relative start position.
length	The length of the data to be totaled. For character data, specify a length of 1 to 15; for binary data, specify a length of 1 to 4; for packed data, its length is calculated by StarBat.
datatype	The type of data. For character data, specify C; for binary data, specify B; for packed data, you do not need to specify the type.
description	This is optional. Enter a maximum of 25 characters enclosed in single quotes. If a description is present, displays as report headers; otherwise, the control statement is used as a report header.

The following example selects records that contain a packed decimal '*9802'* in position *5*, then adds the packed field at position *12.* These selected records are printed in vertical format in hex.

```
DD01 PRINTHEX IF=(5,EQ,P'9802'),SUM=(12,'HIST TOTAL')
```

The example below begins processing a VSAM KSDS data set with the record at location *x'00433C'* and then adds the data at position *73*. Processing stops if the key is greater than *x'00433C'*.

```
DD01 TOTAL KEY=X'00433C',
          STOPIF=(1,GT,X'00433C'),
          SUM=(073,8,C,'TRAVEL EXPENSE REPORT')
```

WRITE

The MULTICOPY function only uses the WRITE (W) parameter to control when output is written to the named DD statements. The WRITE parameter can direct output to any number of output DD statements; if the same DDNAME is used more than once, multiple copies of the output record are written.

StarBat does not impose a limit on the number of separate output files when using the MULTICOPY function; however, the region size of the user can limit the total amount of storage available to StarBat. In particular, large values for the BLKSIZE and the BUFNO parameters limit the number of output files that can be simultaneously open.

There are several restrictions on the data that can be output with a MULTICOPY function:

- Output to load members is supported; however, JCL indicating PDS(MEMBER) notation is not supported. Instead, use only the PDS name in the JCL and indicate the member name with the MEMBER, MEMBERS, NEWMBR or NEWMBRS parameters.
- Overlay, scatter-loaded and note-listed load members are not supported.

  `WRITE=`*`ddname1`*`/(`*`ddname1`*`,`*`ddname2`*`,...)`

  where ddnamex is the DDNAME in the JCL containing the name of the output data set.

The following example shows how to create two identical output data sets in **//**OUT1 and **//**OUT2:

```
DD01 MULTICOPY WRITE=(OUT1,OUT2)
```

The following example copies a file and repeats any record with a *C'5*' and changes it to a *C'6*':

```
DD01 MULTICOPY WRITE=OUT1,IF=(20,EQ,C'5'),MOVE=(1,0,1),
             MOVE=(20,C'6'),WRITE=OUT1
```

The following example adds a STEPLIB JCL statement after any EXEC statement:

```
DD01 MULTICOPY WRITE=NEWJCL,IF=(1,20,C' EXEC '),MOVE=(1,80C' '),
            MOVE=(1,C'//STEPLIB  DD  DISP=SHR,'),
             MOVE=(+0,C'DSN=MYHILEV.MYMIDLEV.MYLOWLEV'),WRITE=NEWJCL
```

Chapter 6

Logic

Consecutive IF parameters represent a logical AND condition. In other words,
IF=(23,EQ,C'TEST RECORD'),IF=(10,0,C'RED') has the same effect as
IF=(23,EQ,C'TEST RECORD'),AND=(10,0,C'RED')

This example copies all records containing *TEST RECORD* in position *23* and *RED* anywhere after position *10*:

```
DD01 COPYREC IF=(23,EQ,C'TEST RECORD'),IF=(10,0,C'RED')
```

This example is equivalent because AND and IF are identical in meaning:

```
DD01 COPYREC IF=(23,EQ,C'TEST RECORD'),AND=(10,0,C'RED')
```

Logical OR Conditions

Code multiple data comparisons within an IF, AND, or OR parameter. A logical OR condition is assumed between multiple data comparisons; you can code logical OR conditions in multiple different formats as shown below.

The following example copies all records containing *TEST RECORD* in position *23* or *RED* anywhere after position *10:*

```
DD01 COPYREC IF=(23,EQ,C'TEST RECORD',10,0,C'RED')
```

The following example is identical in effect to the previous example:

```
DD01 COPYREC IF=(23,EQ,C'TEST RECORD'),OR=(10,0,C'RED')
```

The following example copies up to *10* records containing *TEST RECORD* in position *23* and *RED* anywhere after position *10* or the first *20* records containing *A* in position *3* and *B* in position *6*:

```
DD01 COPYREC IF=(23,EQ,C'TEST
RECORD'),AND=(10,10,C'RED'),MAXRECOUT=10,
             IF=(3,EQ,C'A'),AND=(6,EQ,C'B'),MAXRECOUT=20
```

Processing Multiple Selection Parameters

The IF, AND, and OR parameters are called selection parameters because they select records for processing. Limit parameters like MAXRECIN and MAXRECOUT can follow the selection parameters to apply processing record limits and action parameters like CHANGE; or, you can specify WARP to modify or inspect data fields. In addition, you can specify print parameters like PRINTHEX to selectively print records and control parameters like ABEND; or, you can use STARTKEY to select basic execution choices.

Specify complex selection criteria using multiple selection parameters. Multiple selection parameters are handled in groups where a group of selection parameters starts with the first selection parameter and ends after any associated limit, action and print parameters that follow the selection parameters.

For example;

```
DD01 COPYREC
IF=(10,EQ,C'1'),AND=(20,EQ,C'2'),WARP=(30,C,CCYYMMDD,ADD=5Y),
PRINT=3,

IF=(3,EQ,C'A'),AND=(6,EQ,C'B'),WARP=(40,C,CCYYMMDD,ADD=10D),
MAXRECOUT=30
```

In this example, there are two groups of selection parameters. The first group consists of the first IF parameter and the first AND parameter and is followed by the first WARP action parameter and the PRINT parameter. The second group consists of the second IF parameter and the second AND parameter and is followed by the second WARP action parameter and the MAXRECOUT limit parameter.

Assume a logical OR condition between groups of selection parameters that are followed by limit, action or print parameters. This means that if an input record contains data that matches any one of the groups of selection parameters then that input record is selected for processing. In the example above, records that contain a 1 in column 10 and a 2 in column 20 are copied as well as records that contain an A in column 3 and a B in column 6.

If an input record contains data that matches a group of selection parameters, then the limit, action and print parameters that follow that group are processed and StarBat skips to the next processing group. Whether StarBat actually processes those parameters depends on the name of the current function. If the function has an ALL or SOME suffix as a part of its name (for example, COPYSOME or COPYALL versus COPYREC), StarBat processes all selection groups. Otherwise, any subsequent selection groups are bypassed after the first selection group is satisfied.

In the example above, since COPYREC is the function, and an input record had a 1 in column 10 and a 2 in column 20, and an A in column 3 and a B in column 6, then the record is copied, but only column 30 is warped. If COPYSOME or COPYALL is used, then both column 30 and column 40 are warped.

Selecting Members by Content

To select members by their contents, use a MBR form of a function name (for example, COPYMBR, PRINTMBR, PRINTHEXMBR, PRINTCHRMBR or UPDATEMBR), conditioned by an `IF` statement. Members that meet the selection criteria are processed by the function.

With positive conditional operators such as `EQ`, `GT`, and `LT`, the IF statement evaluates all records within a single member using implied logical `OR`. Consequently, ***only one record*** needs to meet the selection criteria for the entire member to be processed by the function conditioned by `IF`.

For example, the following statement copies all members containing at least one record with the literal character value `TEST RECORD` beginning in column 23:

```
DD01 COPYMBR IF=(23,EQ,C'TEST RECORD')
```

With negative conditional operators such as NE, the IF statement evaluates all records within a single member using implied logical AND. Consequently, ***all records*** in the member must meet the selection criteria in order for the member to be processed by the function conditioned by IF.

For example, the following statement copies a member only if none of the records in that member contain the character literal //STEPLIB beginning in column 1.

```
DD01 COPYMBR IF=(1,NE,C'//STEPLIB')
```

Selecting Members by Name

Select members of a PDS by their name or by a mask that represents their name. Use the MEMBER and MEMBERS parameter to achieve this.

Enter the MEMBER parameter only once on a control statement. To specify multiple members, use parentheses with MEMBER; use the MEMBERS parameter; or use multiple control statements with MEMBER and the same DD*xx* name.

The following example copies a single member.

```
DD01 COPYREC MEMBER=MEMBERA
```

The following example prints two different members.

```
DD01 PRINTHEX MEMBER=MEMBERA
DD01 PRINTHEX MEMBER=MEMBERC
```

The following example prints two different members.

```
DD01 PRINTHEX MEMBER=(MEMBERA,MEMBERB)
```

The following example copies all members whose names begin with *STAR*.

```
DD01 COPYREC MEMBERS=STAR
```

The following example prints all members whose names begin with *STAR* and contain Z in position eight.

```
DD01 PRINTHEX MEMBERS=STAR---Z
```

OPTIONS=JCL Processing

StarBat supports **OPTIONS=JCL,** which means that StarBat determines JCL control statements and continuations; and each statement is processed logically. StarBat recognizes DD, EXEC, JCLLIB, JOB, OUTPUT, PROC, SET JCL statements and their logical continuations.

If a line in the member is not one of these statements or its continuation, StarBat uses normal change and update processing rules for that single line. Following lines can then

be searched or changed in either JCL or normal mode. JCL statements must begin with double slashes (//) in columns 1 and 2. Comment statements are considered a part of the concatenation.

Thus, the following is considered a single statement:

```
//STEPABC  EXEC  PGM=STARBAT,PARM='BATCH',
//* THIS IS A COMMENT IN THE MIDDLE OF A CONTINUED STATEMENT
//  TIME=4,REGION=4096K
```

To search a JCL structure, use the IF or OR statements. You can also search and update JCL statements with the CHANGE, CHANGEALL, OVERLAY and OVERALL statements. Be sure to use a length of zero for the search length of these parameters if you want to search the entire JCL statement.

The OVERLAY and OVERALL parameters update data but do not shift data columns, therefore, they can be used without problems by any of the COPY operations, MULTICOPY or the UPDATE operations.

CHANGE and CHANGEALL can cause data shifting if the search and replacement strings differ in length. If the string is shortened, StarBat follows normal JCL rules and is able to update any of the COPY, MULTICOPY or UPDATE operations. If the string is expanded, StarBat adjusts the JCL statement to fit the statement in the new string using normal JCL rules.

JCL expansion is performed using the following logic to minimize changes in JCL statements:

1 If the string fits, the JCL text is expanded to the right on the changed JCL line.

2 If it does not fit and the string fits, JCL text is added after shifting the changed JCL line left and right.

3 If both of the above expansions fail, the JCL line to be changed is split at the previous comma and continued on another line and the added line is expanded as needed. The new JCL line is indented to match the split JCL line.

4 If the above expansion fails because there is not a previous parameter, StarBat scans for the next comma to the right and splits the JCL statement after that parameter. Again, the new JCL line is indented to match the split JCL line.

StarBat cannot split a JCL line for UPDATE operations. In this case, an expansion error is issued and the return code for the job is set to eight.

Restrictions:

- A JCL statement can contain up to 50 JCL lines.
- OPTIONS=JCL and MOVE from input to output is not supported.
- The JCL is assumed to be syntactically correct; however, StarBat does not issue syntax messages for incorrect JCL.
- To be considered JCL, the statements must reside in a PDS member and the data set must have a fixed record format with 80 character records.
- Only data between columns 1 through 71 are considered for determining control statement boundaries and continuations.

Appendix A

Examples and Sample Execution

This section contains the following:

Function Coding Examples

The examples in this section illustrate various uses of StarBat functions and parameters.

Example 1: This example copies all records and tests every third record for a PBCE in column 3. If a match is found, PBCE is replaced with 4 zeroes.

```
DD01 COPYALL SELECT=3,OVERYLAY=(3,EQ,C'PBCE',C'0000')
```

Example 2: This example replaces VOL with 3 blanks if DUMP is found anywhere in columns 1 through 6 AND if VOL is found 6 relative positions from the location of the character D in DUMP. The record is then copied to the output data set.

```
DD01 COPYREC IF=(1,6,C'DUMP'),IF=(+6,EQ,C'VOL'),OVERLAY=(+0,3C' ')
```

Example 3: This example builds new records by moving data with MOVE parameters. The first 5 records are printed in hexadecimal format.

```
DD01 COPYREC MOVE=(1,10,5),MOVE=(11,10,20),PRINTHEX=5
```

Example 4: The first DD01 control statement copies records until VOL001 is found. The second DD01 control statement begins with the VOL001 record and stops copying when VOL300 is found.

```
DD01 COPYREC STOPIF=(1,EQ,C'VOL001')
DD01 COPYREC STOPIF=(1,EQ,C'VOL300')
```

Example 5: This example adds one year to the date with format YYMMDD in the specified position if a record contains APP1 or APP3 in the first 9 positions. Only these selected records are written to the output data set.

```
DD01 COPYREC RDW=2,MAXDATERR=100,PIVOTYR=70,
   IF=(1,9,C'APP1'),
       OR=(1,9,C'APP3'),
   WARP=(10,C,YYMMDD,ADD=1Y)
```

Example 6: This example copies records containing an 8 or 9 and warps each record with YYDDD or YYMMDD data format. If a record has both formats, only the YYDDD format is warped.

```
DD01 COPYREC IF=(12,EQ,C'8'),
       WARP=(16,C,YYDDD,ADD=5Y),
       IF=(22,EQ,C'9'),
       WARP=(26,C,YYMMDD,ADD=5Y)
```

Example 7: This example copies all records and warps all 8 and 9 dates.

```
DD01 COPYALL IF=(12,EQ,C'8'),
       WARP=(15,C,YYDDD,ADD=5Y),
       IF=(22,EQ,C'9'),
       WARP=(26,C,YYMMDD,ADD=5Y)
```

Example 8: This example copies records containing an 8 or 9 and warps all dates on these records.

```
DD01 COPYSOME IF=(12,EQ,C'8'),
       WARP=(16,C,YYDDD,ADD=5Y),
       IF=(22,EQ,C'9'),
       WARP=(26,C,YYMMDD,ADD=5Y)
```

Example 9: Any record containing a 111 anywhere in the record is replaced with ZZZ. Any record containing a DDD in column 1 is replaced with a '---'. Any record containing ZZZZ in column 1 is not written to the output file.

```
DD01 EXCLUDEREC OVERALL=(1,0,C'111', C'ZZZ'),
         OVERLAY=(1,EQ,C'DDD',C'---'),
         IF=(1,EQ,C'ZZZZ'),MAXRECOUT=0
```

Example 10: This example selects every fifth record, and if it contains a 241104 or a 181022 in position 25, it outputs the record. A maximum of 5 records can be written to the output data set identified by DD5.

```
DD01 MULTICOPY
    SELECT=5,IF=(25,EQ,C'241104,181022'),WRITE=DD5,MAXRECOUT=5
```

Example 11: This example writes all records having 011 in column 9 and 01 in column 1 to the data set identified by DD OUT1. It writes all records having 011 in column 9 and not having a U or a V in column 1 to the data set identified by DD OUT2. It writes all records not having a 011 in column 9 to OUT2.

```
DD01 MULTICOPY IF=(9,EQ,C'011'),IF=(1,EQ,C'01'),WRITE=OUT1,
            IF=(9,EQ,C'011'),IF=(1,NE,X'E4,E7'),WRITE=OUT2,
            IF=(9,NE,C'011'),WRITE=OUT2
```

Example 12: This example inserts a JOBLIB DD statement after the jobcard in all members prefixed with PRS. These members are written to a new data set identified in the JCL by DD NEWJCL.

```
DD01 MULTICOPY MEMBERS=PRS,OPTIONS=JCL,WRITE=NEWJCL,
        IF=(3,0,C' JOB '),MOVE=(1,80C' '),
        MOVE=(1,C'//JOBLIB DD DISP=SHR'),
        MOVE=(+0,C',DSN=JOBLIB.DSN'),WRITE=NEWJCL
```

Example 13: This example processes three input data sets and writes the results to a single output file identified by DD DD02O. Control statement DD02 selects a member called C123.

```
DD01 MULTICOPY WRITE=DD02O
DD02 COPYMBR MEMBER=C123
DD03 MULTICOPY WRITE=DD02O
```

Example 14: If two blanks are found in column 1, they are changed to '**'. Changed records are written to the output file identified by OUTPUTA in the JCL. Processing stops when END is found in column 1.

```
DD01 MULTICOPY CHANGE=(1,EQ,C'
',C'**'),STOPIF=(1,EQ,C'END'),WRITE=OUTPUTA
```

Example 15: This example writes CATALOG NAME= to the output buffer followed by the 15 bytes 11 relative positions after the 'I' in IN-CAT when IN-CAT is found anywhere in the record. It writes the 15 bytes to the current relative position in the output buffer. Then, it writes the record to the output data set specified by CATREPT.

```
DD01 MULTICOPY PADCHAR=C' ',
           IF=(1,0,C'IN-CAT'),
         MOVE=(1,C'CATALOG NAME='),
         MOVE=(+0,15,+11),
         WRITE=CATREPT,MAXRECOUT=0
```

Example 16: This example replaces the first occurrence of text string CYCLE with blanks and prints the record if text string ** AAAA is found anywhere in a record.

```
DD01 PRINTCHR IF=(1,0,T'** AAAA'),
          OVERLAY=(1,0,T'cycle',5C' '),MAXRECOUT=0
```

Example 17: This example lists the contents of all members in the partitioned data set that match the member mask specified. For example, members that match the member mask include TEST111 and TESTAB1.

```
DD01 PRINTCHRMBR MEMBERS=TEST--1
```

Example 18: This example processes the input data set twice, once for each PRINTHEX control statement. Records containing EDIT or A99 in position 19 are printed in hexadecimal format.

```
DD01 PRINTHEX OPTIONS=MULTI,IF=(19,EQ,C'EDIT')
DD01 PRINTHEX IF=(19,EQ,C'A99')
```

Example 19: This example changes the SYS001 to a SYSTST for members containing a SYS001 anywhere in the first 20 positions of the record.

```
DD01 UPDATEMBR CHANGE=(1,20,'SYS001',C'SYSTST')
```

Example 20: This example changes all occurrences of hex 98365C to hex 99365C if a hex 98365C is found anywhere in the record. The second IF parameter statement is logically ORed with the first. In the second IF parameter, if a hex 98000C is found anywhere in the record, all occurrences of hex 98000C are changed to hex 99000C.

```
DD01 UPDATEREC IF=(1,0,X'98365C'),OVERALL=(1,0,X'98365C',X'98365C')
             IF=(1,0,X'98000C'),OVERALL=(1,0,X'98000C',X'99000C')
```

Example 21: This example updates the data set in place and replaces all occurrences of a '!' or a ':' or a '?' with a blank.

```
DD01 UPDATEREC REPLALL=(1,0,C'!,:,?',C' ')
```

Example 22: This example skips records 1 though 10, outputs records 11 through 20, skip records 21 through 25, then outputs records 26 through 45.

```
DD01 SKIP MAXRECIN=10
DD01 COPYREC MAXRECIN=10,MAXRECOUT=10
DD01 SKIP MAXRECIN=5
DD01 COPYREC MAXRECIN=20,MAXRECOUT=20
```

Example 23: This example skips the last two records beginning at the end of a sequential data set. Any remaining records that have 01SER20 in column 1 are printed in reverse order.

```
DD01 SKIPREV MAXRECIN=2
DD01 PRINTREV IF=(1,EQ,C'01SER20')
```

Sample JCL and Control Facilities

```
//SAMPLEA  JOB ...
//PRINT1 EXEC PGM=StarBat,TIME=1,REGION=2M
//SYSPRINT DD SYSOUT=(,)
//SYSLIST  DD SYSOUT=(,)
//SYSTOTAL DD SYSOUT=(,)
//DD01      DD DSN=SAMPLE.LIB.PDSE(StarBatT),DISP=SHR
//DD01O     DD DSN=SAMPLE.LIB.ASM(StarBatT),DISP=SHR
//SYSIN     DD *
DD01 COPYSOME  IF=(1,EQ,C'1800'),
            WARP=(6,C,YYDDD,ADD=5Y),PRINTHEX=1,
            SUM=(26,4,C,C'The 1800 totals'),
            IF=(12,EQ,C'1900'),
            WARP=(17,C,YY,ADD=2Y),PRINTHEX=1,
            SUM=(42,3,C)
//
```

Sample SYSPRINT Output

```
DD01 COPYSOME  IF=(1,EQ,C'1800'),
            WARP=(6,C,YYDDD,ADD=5Y),PRINTHEX=1,
            SUM=(26,4,C,C'The 1800 totals'),
            IF=(12,EQ,C'1900'),
            WARP=(17,C,YY,ADD=2Y),PRINTHEX=1,
            SUM=(42,3,C)
 *** End of control statement

PDS220I //DD01      DD  DSN=SAMPLE.LIB.PDSE,DISP=SHR,UNIT=3380,
PDS220I //  DCB=(RECFM=FB,LRECL=80,BLKSIZE=11440,DSORG=PX),VOL=SER=SER003,
PDS220I //  SPACE=(CYL,(420,20))                /*FREE TRK=1029*/

STRB01I BSAM input is in use
STRB05I DDNAME=SYS00001 DSN=SAMPLE.LIB.ASM opened for BSAM output
STRB05I DCB=(RECFM=FB,LRECL=80,BLKSIZE=11440),VOL=SER=SER003
--- STRB75E WARP= Invalid character number at column 0006; hex=X'F6F9F1C1F3';
```

```
                                                  char=C'691A3'; at record 7
Text:             WARP=(6,C,YYDDD,ADD=5Y)

-- STRB80E SUM= invalid numeric character; char=C'00X0'; at record 7
TEXT:             SUM=(26,4,C,C'The 1800 totals')


Actions taken for:       IF=(1,EQ,C'1800')               --------7 --------4
Actions taken for:  WARP=(6,C,YYDDD,ADD=5Y)              --------4 --------3
Actions taken for:  SUM=(26,4,C,C'The 1800 totals')      --------4 --------3
Actions taken for:  IF=(12,EQ,C'1900')                   --------6 --------3
Actions taken for:  WARP=(17,C,YY,ADD=2Y)                --------3 --------3
Actions taken for:  SUM=(42,3,C)                         --------3 --------3
```

Sample SYSLIST Output

```
StarBat Version 7.4.0;  Jan  5, 2001;  DD01=SAMPLE.LIB.PDSE(StarBatT)
     VOL=SER003  PAGE   1

RECORD      2 DATA    80   CHAR 1800+74123-1900-98-2----+0034+----+----4-123+---
     -5...7----+----8
   ***C H A N G E D***     ZONE
     FFFF4FFFFF6FFFF6FF6F66664FFFF4666646666F6FFF46666F666F666646666F
                                NUMR
     1800E7412301900098020000E0034E0000E000040123E0000500070000E00008
                                ----+----1----+----2----+----3----+----4----+---
     -5...7----+----8

RECORD      2 DATA    80   CHAR 1800+74123-1900-00-2----+0034+----+----4-123+---
     -5...7----+----8
   ***C H A N G E D***     ZONE
     FFFF4FFFFF6FFFF6FF6F66664FFFF4666646666F6FFF46666F666F666646666F
                                NUMR
     1800E7412301900000020000E0034E0000E000040123E0000500070000E00008
                                ----+----1----+----2----+----3----+----4----+---
     -5...7----+----8

RECORD      7 DATA    80   CHAR 1800+691A3-1900-A8-2----+00X0+-*ERROR--4-00X+---
     -5...7----+----8
   *****E R R O R*****     ZONE
     FFFF4FFFCF6FFFF6CF6F66664FFEF465CDDDD66F6FFE46666F666F666646666F
                                NUMR
     1800E6911301900018020000E0070E0C599690040007E0000500070000E00008
                                ----+----1----+----2----+----3----+----4----+---
     -5...7----+----8

RECORD      7 DATA    80   CHAR 1800+691A3-1900-A8-2----+00X0+-*ERROR--4-00X+---
     -5...7----+----8
   *****E R R O R*****     ZONE
     FFFF4FFFCF6FFFF6CF6F66664FFEF465CDDDD66F6FFE46666F666F666646666F
                                NUMR
     1800E6911301900018020000E0070E0C599690040007E0000500070000E00008
                                ----+----1----+----2----+----3----+----4----+---
     -5...7----+----8
```

Sample SYSTOTAL Output

```
SYSTOTAL Report

                                Following totals developed from
                                SAMPLE.LIB.PDSE  VOL=SER003

'The 1800 totals                                 ----------           155
            SUM=(42,3,C)                         ----------           458
```

Appendix B

Return Code Processing in StarBat

StarBat return code processing may be customized. This section discusses default return code processing as well as return code customization.

Job Step Return Codes

Final Return Code

Return code processing in the StarBat Option occurs whenever an error condition is encountered, and again at the end of each job step in the job stream. A final return code for the job step is issued after evaluating all return codes encountering during job step execution. In general, the final return code will equal the highest interim return code encountered.

For certain error conditions, StarBat performs additional checking to assess the severity of the error. The return code issued for a given error condition can vary based on the outcome of this checking. If a return code value is overridden in this way, both values are returned, but only the overriding return code value is considered when computing the final return code for the job step.

Empty Input Data Set Checking

Not infrequently, StarBat may return no output from a function that is normally expected to generate output. If no input data was supplied to that function, the no-output condition is probably routine. However, if input data does exist, the no-output condition may or may not be serious, depending on the function being performed and whether or not all data sets in the input concatenation were read.

By default, StarBat checks for empty input data sets whenever a no-output error condition occurs. Empty input data set checking may revise a previously issued return code value. The final return code for the job step will be evaluated accordingly.

Empty input data set checking may optionally be disabled by customizing the StarBat return code table, `SZFRCTAB`.

StarBat Return Code Table

StarBat return code processing is controlled by the StarBat return code table. This table may be customized.

Global Return Code Processing

The return code table determines global StarBat behavior for the following:

- The return code value at which end-of-job is forced.
- The return code issued when no input is found.
- The return code issued for a syntax error.
- The return code issued for an I/O error.

The following parameters in the return code table set these global values.

Parameter	Description	Default Value
EMPTYRC	Return code issued when no input found. **NOTE:** To turn off empty input data set checking, set this parameter to -1.	04
FCEOJRC	Return code value that forces end-of-job	12
IOERRRC	Return code issued for I/O error	12
SYNERRC	Return code issued for syntax error	08

Return Codes by Function

Return code values can be customized on a function-by-function basis for certain reportable conditions. These include:

- The return code issued at normal termination.
- The return code issued when input data is read, but no output is returned (for example, certain output selection criteria are not satisfied).
- The return code issued when input data is read, but one or more concatenated data sets is empty.

Function-specific return codes concerning no-input and no-output conditions are used in conjunction with the global no-input parameter EMPTYRC during empty input data set checking.

Default return code values for these conditions are shown for each StarBat function in the following table.

StarBat Function	Normal Termination	Input Data Set Read, No Output	Concatenated Input Data Set Empty
COPYALL	00	08	04
COPYMBR	00	08	04
COPYREC	00	08	04
COPYREV	00	08	04
COPYSOME	00	08	04
EXCLUDEREC	00	00	00
FPRINT	00	00	00

StarBat Function	Normal Termination	Input Data Set Read, No Output	Concatenated Input Data Set Empty
FPRINTALL	00	00	00
FPRINTMEM	00	00	00
MULTICOPY	00	08	04
PRINT	00	00	04
PRINTALL	00	00	04
PRINTCHR	00	00	04
PRINTCHRALL	00	00	04
PRINTCHRMBR	00	00	04
PRINTCHRREV	00	00	04
PRINTHEX	00	00	04
PRINTHEXALL	00	00	04
PRINTHEXMBR	00	00	04
PRINTHEXREV	00	00	04
PRINTMBR	00	00	04
PRINTREV	00	00	04
REFORMAT	00	00	00
SKIP	00	00	00
SKIPREV	00	08	04
TOTAL	00	00	00
UPDATEALL	00	00	04
UPDATEMBR	00	00	04
UPDATEREC	00	00	04

Customizing the Return Code Table

The StarBat return code table is coded in assembly language member SZFRCTAB. The SZFRCTAB source code member is located in library *somnode*.PDSE*vrm*.ASSEMBLE.

SZFRCTAB calls the assembly language macro SZFRCTG. The source member for this macro is located in library *somnode*.PDSE*vrm*.COPY. The macro is needed for assembly of SZFRCTAB if you customize the values in the return code table.

Table Customization Steps

To customize the StarBat return code table, perform the following steps:

1 Make an editable copy of SZFRCTAB in a separate library where you keep your customized StarTool FDM source members.

2. Edit the parameter values in the control table as desired, following the instructions contained in the member.

3. Assemble and link the modified SZFRCTAB member with macro SZFRCTG and place the resulting load module in the separate library where you keep your customized StarTool FDM load members.

4. Add your custom load library to the JOBLIB or STEPLIB concatenation used to run StarBat jobs.

At runtime, if StarBat finds an SZFRCTAB load module in the run-time load library concatenation, it uses it. If not, StarBat uses a return code table with default return code values that was pre-linked at the factory into the StarBat load module.

Printing the Return Code Table

The return code table in use at runtime can be printed in the SYSPRINT file. To do this, add the PARM=PRINTRCT parameter to the StarBat EXEC JCL statement. For example:

```
//stepname EXEC PGM=STARBAT,PARM=PRINTRCT
```

Index

H

I

J

L

M

N

O

P

R

S

T

U

V

W

X