

Model S561 Batch Tools Support

Reference Manual

9234203A V1.3



Copyright 2002, Canberra Industries, Inc. All rights reserved.

The material in this document, including all information, pictures, graphics and text, is the property of Canberra Industries, Inc. and is protected by U.S. copyright laws and international copyright conventions. No material in this document may be reproduced, published, translated, distributed or displayed by any means without written permission from Canberra Industries.

Canberra Industries, 800 Research Parkway, Meriden, CT 06450
Tel: 203-238-2351 FAX: 203-235-1347 <http://www.canberra.com>

The information in this manual describes the product as accurately as possible, but is subject to change without notice.

Printed in the United States of America.

Table of Contents

Preface	vi
1. Batch Procedure Tutorial	1
Additional Documentation	1
Batch Procedure Commands	1
Command Help	2
Batch Procedure Environment	2
Enhancing with Graphical Batch Tools	7
Starting the Batch Procedure.	11
2. Batch Procedure Reference	13
Command Index	13
Introduction to the Batch Procedure Environment	20
The Editor	21
Command Format	21
Environment Variables	22
Datasource Naming Conventions	23
Control Job Commands	23
Analysis Job Commands.	25
Peak Locate	25
Peak Area Analysis	33
Calibration	37
AEFFCAL ^{RX}	47
Area Correction	50
ARALPHA ^{RX}	53
Reagent Correction	53
ARREAGNT ^{RX}	53
Efficiency Correction	54
Nuclide Identification	57

Detection Limits	60
Post NID Processing.	61
General Job Commands	63
Reporting	63
Editing	65
Options	66
Hardware Control	68
Application Interaction	81
Miscellaneous Commands.	85
Application Option Commands	97
QA Batch Procedure Commands	97
IPFIT	102
General Job Functions	103
REXX Function Equivalents.	104
MakeVDMConnection.	106
MakeDirectConnection.	106
CloseConnection	107
GETAREA	108
GETPARAM.	109
GETDATA	110
PUTDATA	111
GETINT	112
EVALENEQ	112
EVALSHEQ	113
EVALEFEQ	114
COUNTREC	115
COUNTENT	115
MDAPRESET	116
ICBCOMM.	118
DTCONVERT	119
GETRECS	120
PUTRECS	121
GETENTS	122

PUTENTS	122
PUTPARAM	123
INSERTREC	124
DELETEREC	125
COPYREC	126
GETTEXT	127
Serial I/O REXX Functions	128
Batch Procedure Example	131
Routine Gamma Spectrum Analysis	132

3. Graphical Batch Tools 133

Command Index	133
Overview	134
Environment Variables.	135
Password Security	136
Function Calling Conventions	137
Function Return Values	137
Function Error Returns.	138
Description of Commands and Functions	139
Show Argument	140
Graphical Batch Tool Command.	140
GBT_INIT	140
Fixed Format Functions	142
GBT_GETMENU	142
GBT_NOTE	143
GBT_REPORT.	143
GBT_CLEAR	144
GBT_TERM	145
GBT_MESSAGE	145
GBT_GETXY	146
GBT_FINDDS	147
GBT_COUNTDS	148
GBT_VIEW	150

GBT_LOGON	151
GBT_LOGOFF.	152
GBT_FINDMID	152
FDS-Driven Functions	154
Form Design Specification (FDS) Files	154
FDS Keywords	155
GBT_DEFMENU	166
GBT_PARS	166
GBT_ENUM.	169
GBT_SELECT	171
GBT_QUERY	173
GBT_WAIT	174

4. A Graphical Batch Example 176

What You Need to Know.	177
The Startup Code (Part A)	178
The Logon Procedure (Part B)	180
The Routine Count Menu (Part C)	183
Analyze a File (Part D).	186
Count a Sample.	188
Getting the Name of the MCA to be Used (Part E)	188
Opening an MCA Display (Part F).	190
Getting the Geometry File (Part G)	190
Requesting and Setting the Preset Time (Part H)	191
Getting the Analysis Parameters (Part I).	191
Requesting the Output Filename (Part J).	194
Waiting for the Count to Finish (Part K).	195
Store or Analyze (Part L)	196
Analysis.	196
Analyzing the Spectrum File (Part M).	196
Displaying the Results (Part N)	198
The End of the Program (Part O).	199

A. Batch Procedure Errors	201
Batch Processing Errors	201
Graphical Batch Error Codes	207
System Level Errors	207
FDS File Parsing Errors	209
Index	211

Preface

One of the key requirements of laboratories involved in routine sample counting is the ability to use common count procedures. Generally speaking, the definition of count procedures must be very flexible in order to attain a very controlled end result for the count room technician.

Procedures must be developed which guide an operator through a specific set of operations. At each step, the procedures must provide clear instructions to the operator, provide facilities for validating input and provide meaningful operator feedback.

Just as important, the procedure must be as simple as possible for the operator. It must not require operator interaction except for obtaining specific information or guiding the operator in performing some operation. In short, the operational requirements - not the design of the counting system - should determine what operator interaction will be.

Genie-2000 facilitates this approach to batch procedures with a robust, highly flexible environment. You decide the appropriate level of operator interaction. You can prompt the operator for only the specific information you want the operator to furnish, and default the rest.

You don't need to provide the operator with any information that is not appropriate to the specific operation being performed. You can deny access to sensitive parameters (for example, analysis library names, start/stop channels) where an erroneous entry could destroy an analysis. You can check operator input for "reasonability" (for example, input within certain limits) and prompt for re-entry if an error has been made.

About This Manual

Chapter 1, Batch Procedure Tutorial, takes you on a quick tour of Genie-2000's batch procedure feature.

Chapter 2, Batch Procedure Reference, describes all of the batch parameters in detail, including the command's syntax and qualifiers and an example or two of how the command might be used.

Chapter 3, Graphical Batch Tools, covers the graphical commands needed for developing custom procedures which use a graphical user interface.

Chapter 4, A Graphical Batch Example, illustrates how the batch procedures and graphical tools can be used to develop a real-life routine.

Appendix A, Batch Procedure Errors, lists and describes the error codes which can be returned by the software environment while running batch commands.

1. Batch Procedure Tutorial

This chapter is designed to quickly acquaint you with the facilities of the batch procedure environment so you can get on line quickly. It provides an introduction to the overall concepts of the Genie-2000 batch procedure environment and includes a brief example to demonstrate the basic technique of procedure programming.

Chapter 2, *Batch Procedures Reference*, describes all of the procedure commands in detail and their operation; all command line switches are presented. Chapter 2 also includes a batch procedure example of standard gamma spectroscopy operations.

Many facilities will be able to use these examples just as they are written. Many others will be able to use them with very simple modifications for their own unique requirements. Those users with very specialized requirements will want to use these examples to familiarize themselves with the procedure programming techniques in detail.

Complementing the Batch Environment are the Graphical Batch Tools, which are a set of functions callable under REXX (a scripting language shipped as part of the Model S561 Genie-2000 Batch Programming Support). They allow you to develop procedures that use a Graphical User Interface (GUI). This provides not only a more friendly user interface, but also facilitates error checking and user security protection.

The Graphical Batch Tools are described in Chapter 3, *Graphical Batch Tools* and a comprehensive example of a sample counting batch procedure, using the Graphical Batch Tools, is presented and discussed in detail in Chapter 4, *A Graphical Batch Tools Example*.

Additional Documentation

Most procedure operations will use the capabilities of REXX. In this manual we have described features and capabilities of REXX that are useful for typical spectroscopy applications. Additional information can be found in the on-line help furnished with the Model S561 Batch Programming Support.

Batch Procedure Commands

The heart of the Genie-2000 batch procedure capability is the set of procedure commands that are used to execute various functions in the system. These are command line executable programs that call the underlying program modules which, in turn, perform spectroscopy related functions.

For example, individual commands exist to start acquisition on an MCA, run a peak locate algorithm, generate a report and save data from a detector to a disk file. Other commands allow for entry of parameter data or display of an application window.

Commands can be run either from the command line or, more typically from a command file. Most commands accept arguments or command line switches that give the command more detail on what to do. For example, the PEAK DIF command that performs a second difference peak search accepts arguments for search sensitivity and channel range.

Command Help

All commands accept a [/help] switch. When this switch is selected, the command will present a help text file on the screen, describing the complete use of the command. Keep this in mind as you begin to develop command procedures – this quick help reference will save you time in working on your procedures.

To see how this works, perform the following steps:

1. Launch an MS-DOS Command Window (from the Start Menu).
2. Type `STARTMCA /Help`. This will give you help text on the STARTMCA command. Note the list of arguments (for example, datasource specification or presets) associated with this command.
3. Type `EXIT<F255>` to close the DOS Command Window session and return to the desktop.

Batch Procedure Environment

As we mentioned, most of the time you will be running commands from a procedure command file rather than from the command line. This requires some familiarity with how Windows 95/NT handles procedure operations.

Command files are created with any standard ASCII text editor, such as the Notepad application supplied with Windows 95/NT.

There is one basic procedure environment associated with the Windows operating system: the native DOS batch environment. You'll recognize the constraints of this environment, particularly in terms of operator dialog, arithmetic capability, logical testing, branching, and so forth.

Batch Procedure Environment

For these operations, the capabilities of the REXX command environment are required. REXX basically serves as a “command filter”, passing the basic commands that can be handled by Windows directly to the operating system command processor. REXX commands are processed by REXX into commands that can be handled by Windows and passes them on. This is shown graphically in Figures 1 and 2.

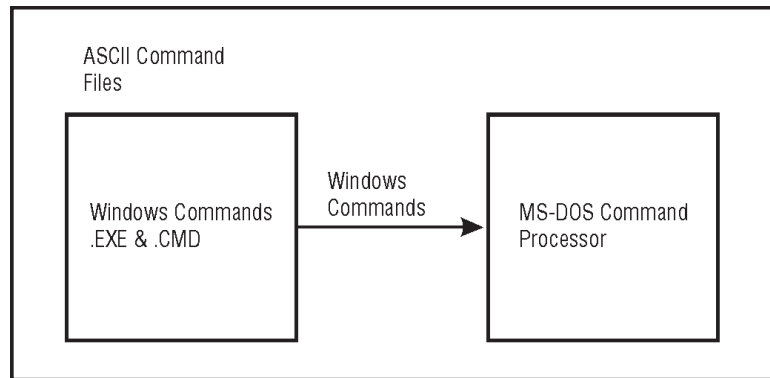


Figure 1 Native Windows Batch Environment

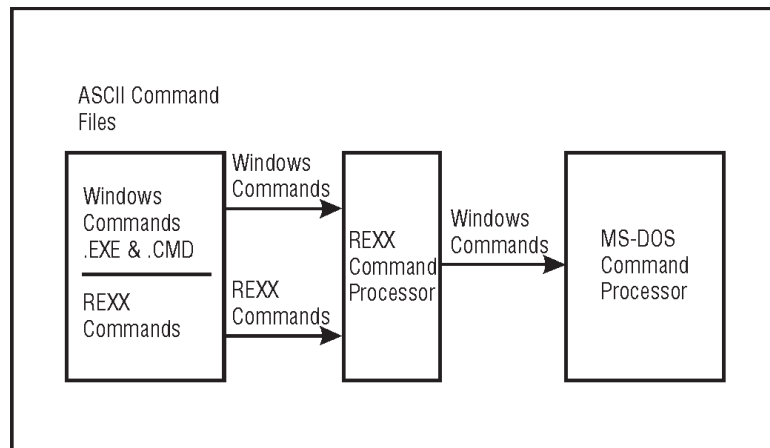


Figure 2 REXX Command Processing Environment

We can reinforce this concept with a simple example:

```
STARTMCA det:MY_HPGE /livepreset=1000
```

starts acquisition on the detector named MY_HPGE with a live time preset of 1000 seconds. However, in many applications, the preset may be requested from the operator rather than defined in the procedure. To accomplish this, we need REXX commands in addition to the STARTMCA command.

The REXX Tutorial File

To write the REXX batch tutorial file shown in the following pages, from the full window open the Notepad application. As you complete each section of the tutorial file, save it as TUTORIAL.REX in the GENIE2K\EXEFILES directory. This file can be executed from this directory opening an MS-DOS Command Window and typing:

REXX tutorial

First, we need to alert the REXX command processor that this procedure will contain REXX commands.

The REXX commands which are used for operator dialog are SAY and PULL. The SAY command sends a text string, such as a prompt, to the screen. The PULL command accepts keyboard input and sets it equal to a variable that can be used elsewhere in the procedure.

We'll add a few more lines to our simple procedure:

```
/*Sample Counting Procedure for Genie-2000*/

SAY 'Please Enter Live Preset Time'

PULL livprst

"STARTMCA det:MY_HPGE /livepreset="livprst

EXIT
```

The SAY command sends the string "Please Enter Live Preset Time" to the screen. The PULL command takes the operator response and associates it with the variable "livprst". Now whatever is entered by the operator is set equal to that variable, so when the STARTMCA command is executed, the stored value is substituted for the variable name referenced in the /livepreset argument. The EXIT command marks the end of the procedure.

Note that the STARTMCA command is enclosed in quotes. This is a signal to the REXX processor to pass this directly to the MS-DOS command processor because it is an .EXE file that is recognized directly by Windows. The quotes end just before the reference to the REXX variable "livprst" because the variable is processed by REXX, not by Windows. This technique must be used on any line that combines REXX and Windows commands.

Let's add a few enhancements to our simple procedure. In many cases, you'll want an MCA window to display the spectrum in real time while it is being acquired. Also, we will want to allow acquisition to end before removing the MCA window and exiting.

Adding the required Genie-2000 commands PUTVIEW, WAIT and ENDMETHOD makes our procedure look like this:

```
/*Sample Counting Procedure for Genie-2000*/  
  
SAY 'Please Enter Live Preset Time'  
  
PULL livprst  
  
PUTVIEW det:MY_HPGE /disp_only /xy=1,1 /cxcy=635,235  
  
"STARTMCA det:MY_HPGE /livepreset="livprst  
  
"WAIT det:MY_HPGE /acq"  
  
"ENDVIEW"  
  
EXIT
```

The PUTVIEW command puts up a MCA window in a "display only" mode (no operator acquisition or control), starting at pixel 1,1 (lower left corner) and ending at pixel 635,235. This gives us an MCA window that covers approximately the bottom half of the screen.

The WAIT command with the /acq switch indicates that we will pause on this line until input's acquisition is complete. The ENDMETHOD terminates the MCA window created with PUTVIEW before exiting.

Now let's add a few more steps to the procedure. We will add statements to prompt the operator for Sample ID and Sample Size, then we will add an automatic analysis sequence to the procedure.

Again, we use SAY and PULL to query the operator and accept input. We use the PARS command to write the sample parameters into the CAM files associated with the input. We also use the Genie-2000 ANALYZE command to run an automatic analysis sequence (these sequences are discussed in “Analysis Sequence” in the Gamma (or Alpha) Acquisition and Analysis chapter in the *Genie-2000 Operations Manual*).

Note that the Analyze command requires that a set of calibrations exist for the detector datasource. See “Calibrate Menu” in the Gamma (or Alpha) Acquisition and Analysis chapter in the *Genie-2000 Operations Manual*.

The revised procedure looks like this:

```
/*Sample Counting Procedure for Genie-2000*/

SAY 'Please Enter Live Preset Time'

PULL livprst

SAY 'Please Enter Sample ID'

PULL smpid

SAY 'Please Enter Sample Size in grams'

PULL smpsz

"PARS det:MY_HPGE /SIDENT="smpid"
  /SQUANT="smpsz" /SUNITS=grams"

"PUTVIEW det:MY_HPGE /disp_only /xy=1,1 /cxcy=635,235"

"STARTMCA det:MY_HPGE /livepreset="livprst

"WAIT det:MY_HPGE /acq"

"ANALYZE det:MY_HPGE /SEQ=c:\geniepc\ctlfiles\NID_SHO.ASF"

"ENDVIEW"

EXIT
```

Note the use of CAM variable names in the PARS command. Also, note the use of quotes to again delineate between REXX commands and native Windows commands.

Enhancing with Graphical Batch Tools

In most cases, you'll want to present the operator with a Graphical User Interface (GUI) type presentation rather than the question and answer dialog of the native REXX environment. The Graphical Batch Tools furnished with Genie-2000 will accomplish this.

These tools allow you to create custom menus, enumeration lists, parameter entry screens, message boxes, and so forth, in a GUI style that can be operated either with a mouse or the keyboard. Details are discussed in Chapter 3, *Graphical Batch Tools*.

To demonstrate the use of the tools, we will enhance the simple example procedure in the last section, replacing the REXX "SAY/PULL" dialog with a Graphical Batch Tool implementation of parameter entry using the GBT_PARS tool. For simplicity we will also include the acquisition preset time and prompt the operator only for the sample ID and size.

We will modify our example by replacing the second through eighth lines (three SAY/PULL pairs and the PARS command). First, we must register all Genie-2000 REXX functions for use by the REXX environment. To do this we simply add this command to the beginning of the procedure:

```
"REXXFCTS"
```

Now we must call an external REXX procedure (supplied with the S561 package) named G2KENVMT; This routine sets up the Genie-2000 environment variables based on setting found in the registry.

Next, we define an environment variable to tell the system in which directory to find the dialog box description files, called FDS files, which are used to define the format and contents of the various screens and dialog boxes. This is the command:

```
"SET GBT_FDSFILES=c:\geniepc\exefiles"
```

Now we are ready to initiate an application window dialog which serves two functions in this example. It serves as a "backdrop" window for subsequent GBT menus and dialogs. This will hide a portion of the desktop, reducing the visual clutter we would see while the procedure is executing. Additionally, this window functions internally as a window manager, controlling the presentation of other GBT calls that may be executed.

The command used is:

```
"GBT_INIT /title="Example Batch Program"
  /xy=-0,-50 /cxcy=-100,-100 /nosize"
```

This command initiates an application window dialog in the top half of the screen with the title "Example Batch Program". The "/nosize" switch means that the window cannot be resized.

Now we are ready to begin defining the parameter entry screen. We first specify a title and location for the screen.

```
Show.Title="Sample Information Parameters"
```

```
Show.Position="ICC"
```

This tells the system that the parameter screen will have the title "Sample Information Parameters" and will appear inside (I) the application window, centered horizontally and vertically (CC).

Our next statement actually puts up the parameter entry dialog box, described in the SAMPLE.FDS file:

```
Ans=GBT_PARS("SAMPLE.FDS", "Det:MY_HPGE" , , "Show" )
```

This command brings up the screen, associates it with the detector datasource MY_HPGE and specifies the REXX variable "Show" which defines the position and title information.

Now, we add a command telling the system what to do if the operator presses the **CANCEL** button on the entry screen (if **OK** is pressed, the program will proceed to the next line). For simplicity, we will tell the program to EXIT if CANCEL is pressed:

```
If Ans=">2" THEN SIGNAL Close
```

"Close" is a branch label that directs the program flow past the acquisition control statements and to the end of the procedure. At the end, we also need to add the statement:

```
call GBT_TERM
```

which closes the application window prior to exit.

Finally, we edit the previous STARTMCA command to include the preset:

Enhancing with Graphical Batch Tools

```
"STARTMCA det:MY_HPGE /livepreset="1000
```

Now our procedure looks like this:

```
/*Sample Counting Procedure for Genie-2000*/
```

```
"REXXFCTS"
```

```
CALL G2KENVMT
```

```
"SET GBT_FDSFILES=c:\geniepc\exefiles"
```

```
'GBT_INIT /title="Example Batch Program"  
  /xy=-0,-50 /cxcy=-100,-50 /nosize'
```

```
Show.Title="Sample Information Parameters"
```

```
Show.Position="ICC"
```

```
Ans=GBT_PARS("SAMPLE.FDS","det:MY_HPGE",,"Show")
```

```
IF Ans=">2" THEN SIGNAL CLOSE
```

```
PUTVIEW det:MY_HPGE /disp_only /xy=1,1 /cxcy=635,235"
```

```
"STARTMCA det:MY_HPGE /livepreset=1000"
```

```
"WAIT det:MY_HPGE /acq"
```

```
"ANALYZE det:MY_HPGE /SEQ=c:\geniepc\ctlfiles\NID_SHO.ASF"
```

```
"ENDVIEW"
```

```
close:
```

```
call GBT_TERM
```

```
EXIT
```

Now all we need to do is define the `SAMPLE.FDS` file, which can be created with any suitable text editor and saved as an ASCII file.

This file defines two parameter entry boxes, one of which prompts for Sample Identification and the other for Sample Size in grams. For sample size, we will define low and high limits of 1 and 10 grams. We also need to define the buttons **OK** and **CANCEL**, where OK continues the procedure and CANCEL aborts it.

The `SAMPLE.FDS` file looks like this:

```
COMMON

{ SIDENT ; ; Sample Identification ; 6; 4; 27; 25}

{ SQUANT ; ; Sample Size (Grams) ; 8; 4; 27; 25; 1; 10}

BUTTON

{ 1; ; O~k ; 12; 13; 14}

{ 2; ; ~Cancel ; 12; 29; 14}
```

The parameters in the file indicate that CAM variable name, the text prompt, the row and column for the prompt, the column and width for the entry field, and the lowest and highest values which will be accepted for sample size.

The limit testing is done internally to the `GBT_PARS` program and will prevent **OK** until entries are correct and within range. You do not need to write any error checking code to do this.

Starting the Batch Procedure

When run, the procedure will present the screen seen in Figure 3. Note the position of the view screen and the contents and position of the parameter screen as defined by SAMPLE.FDS.

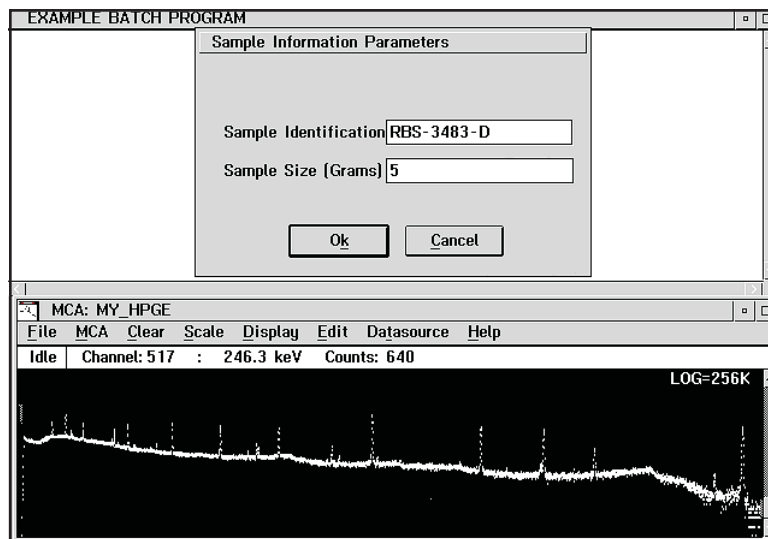


Figure 3 Completed Example Dialog

That's how easy it is to develop a quick batch procedure to do a simple job. A more comprehensive example, using the full suite of graphical batch tools as well as more powerful REXX functions is described in Chapter 4, *A Graphical Batch Example*.

Starting the Batch Procedure

Now it's time to look at how we would execute the sample batch procedures constructed in the two previous sections.

To start a REXX procedure, open an MS-DOS Command Window and type:

```
REXX <procedure_name>
```

Another way to launch the procedure is to create a desktop folder or Start Menu program shortcut. The command windows that is used to launch the batch procedure can be configured to start Minimized if desired. Refer to Windows' on-line help for a information on creating a program shortcut.

Note that because of the multitasking nature of Genie-2000, we could actually prompt for (SAY, PULL) and store the parameters (PARS) after acquisition has begun and before the “wait”. This allows us to make productive use of the time during which acquisition is occurring.

You can see that while the capabilities of the environment are quite comprehensive, the implementation is rather simple. For a more extensive treatment of the batch environment’s capabilities, refer to “Batch Procedure Example” on page 131.

2. Batch Procedure Reference

This chapter describes Genie-2000's Batch Procedures Environment and some examples of batch programming. The environment consists of the Editor, the Job Commands and the Job Functions.

In addition to the basic Model S500 Genie-2000 commands, this chapter includes the batch commands for these analysis options: Model S501 Gamma Spectroscopy Analysis, Model S505 Quality Assurance, S506 Interactive Peak Fit and Model S509 Alpha Acquisition and Analysis.

Several command procedures, which are examples of standard gamma spectroscopy operations, are copied to your \GENIE2K\EXEFILES directory during Genie-2000 software installation. These sample .REX files are described in "Batch Procedure Example" on page 131.

All commands in this manual which have been marked with a superscripted "RX" have REXX function equivalents, described in "General Job Functions" on page 103.

Command Index

Command	Description	Page
ACTLVL	Performs Action Level calculations on identified and unidentified nuclides.	62
ADVANCE	Issues an advance signal to a Sample Changer.	68
AECAL	Performs an energy and shape calibration for the specified datasource.	45
AEFFCAL	Performs an efficiency calibration on the specified datasource, which must contain an Energy Calibration.	47
AEFCOR	Performs the efficiency correction phase of analysis on the specified datasource.	56

ANALYZE	Performs an analysis on the specified datasource.	85
APZERO	Initiates an auto pole/zero operation on a computer-controlled amplifier device.	76
AREACOR	Performs an area correction on the specified datasource.	50
AREA_LIB	Performs a library-driven peak area analysis on the specified datasource.	35
AREA_NL1	Improved algorithm for performing a non-linear least squares fit (for multiplets) and either a fit or a region summation (for singlets) peak area analysis on the specified datasource.	33
ARECAL	Performs an energy recalibration on the specified datasource; FWHM and tail are not recalculated.	46
ARREAGENT	The ARREAGNT command is used to perform the Reagent Correction phase of analysis on the specified datasource.	53
ASE	Displays an Analysis Sequence Editor window for creating or changing analysis sequences.	96
CALPLOT	Plots or shows an Energy or Efficiency calibration curve as a graph.	43
CloseConnection	Closes (and optionally saves) the specified datasource.	107
COUNTENT	Returns the number of entries in the tabular CAM parameter specified.	115
COUNTREC	Returns the number of records in the CAM class of the specified parameter.	115

Command Index

COPYREC	Copies a CAM parameter from one datasource to another.	126
DATAPLOT	Plots Spectral Data.	91
DELETEREC	Deletes records from a CAM file.	125
DIGSTAB	Initiates control functions on a computer-controlled digital stabilizer.	76
DTCONVERT	Converts date and time values to and from various formats.	119
ECAL	Calibrates the specified datasource.	37
EFFCAL	Efficiency calibrates the specified datasource.	41
EFFCOR	Performs an efficiency correction on the specified datasource.	55
ENDVIEW	Terminates the Acquisition and Analysis window started with a PUTVIEW command.	85
EVALEFEQ	Evaluates an efficiency calibration equation for a specified energy, returning the percent efficiency and absolute error.	114
EVALENEQ	Evaluates the energy calibration equation for a specified channel number, returning the energy in keV.	112
EVALSHEQ	Evaluates the specified shape calibration equation for a specified channel number, returning the shape in keV.	113
EXTMCA	Initiates control of an external MCA device.	71
FILECNVT	Converts a foreign format file to a CAM format file.	86

GETAREA	Fetches the net area of a specified data region and stores it in a REXX variable.	108
GETDATA	Fetches raw spectral data from a datasource and stores it as REXX variables.	110
GETENTS	Fetches multiple entries of CAM parameters and stores them as REXX variables.	122
GETINT	Fetches the integral of a specified data region and stores it in a REXX variable.	112
GETPARAM	Fetches parameter(s) from a datasource and stores them as REXX variable(s).	109
GETRECS	Fetches multiple records of CAM parameters and stores them in REXX variables.	120
GETTEXT	Fetches text from a specified file and stores it in a REXX variable.	127
HDWCHECK	Verifies all hardware devices associated with the specified datasource.	74
HVCNTL	Controls a computer-controlled high-voltage power supply.	73
ICBCOMM	Reads/writes a set of registers at a specified ICB address.	118
INSERTREC	Inserts specified CAM parameters into a datasource.	124
IPFIT	Displays the IPF window with the selected datasource.	102
LOADMID	Loads an MID configuration file into the Runtime Database.	93

Command Index

MakeDirect-Connection	Opens the specified datasource, bypassing the VDM and going directly to the PCAM subsystem.	106
MakeVDM-Connection	Creates a connection to the VDM, opens the specified datasource and leaves it open.	106
MDA	Performs a (Currie) MDA analysis on the specified datasource.	60
MDA_KTA	Performs a (German) MDA_KTA analysis on the specified datasource.	61
MDAPRESET	Computes the preset live time (in seconds) required to reach a given MDA.	116
MOVEDATA	Transfers raw spectral data and related information between two datasources.	89
NID_INTF	Performs a nuclide identification and interference activity correction on the specified datasource.	59
NID_STD	Performs a nuclide identification on the specified datasource.	58
NORMAL	Shifts spectral data in the specified input datasource.	66
PARS	Modifies any CAM parameter.	65
PEAK_DIF	Performs an unidentified second difference peak locate on the specified datasource.	25
PEAK_LIB	Performs a library-driven peak locate on the specified datasource.	27
PEAK_ROI	Calculates a peak centroid for each ROI in the source file.	30

PEAK_SLB	Performs a “simple library” peak locate on the specified datasource.	29
PEAK_STD	Performs an Unidentified Second Difference peak locate followed by a pure Gaussian Peak Fit on the specified datasource.	31
PTCAL	Generates a Peak-to-Total efficiency ratio (P/T) calibration from multiple spectra.	49
PUTDATA	Fetches spectral data from a REXX variable and stores it in a datasource.	111
PUTENTS	Fetches multiple entries of CAM parameters and writes them to a datasource.	122
PUTPARAM	Modifies specified CAM parameters.	123
PUTRECS	FetchES multiple records of CAM parameters and writes them to a datasource.	121
PUTVIEW	Specifies the Acquisition and Analysis window with which the Job will interact.	81
PVACCESS	Permits or denies write access to datasources from within the Acquisition and Analysis window.	82
PVCLOSE	Closes a datasource in an existing Acquisition and Analysis window.	84
PVOPEN	Opens a datasource in an existing Acquisition and Analysis window.	83
PVSELECT	Selects an already open datasource in an existing Acquisition and Analysis window.	84

Command Index

PVSHOALL	Shows all open datasources in an existing Acquisition and Analysis window.	85
PWRMGMT	Controls the power manager's operating mode.	75
PWRRESET	Power resets computer-controlled front-end devices.	75
QAANALYZ	Analyze results for out-of-range test(s) and report.	98
QAMANUAL	Enter results manually.	98
QAPLOT	Plot results for an out-of-range test.	100
QAXFER	Transfer results from an analyzed CAM File or Detector.	97
RECAL	Energy recalibrates the specified data source.	40
REPORT	Produces an analysis report for a specified datasource.	63
REFCOR	Uses a reference peak in a spectrum to normalize all of the other peaks in the spectrum.	51
REXXFCTS	Registers a set of Canberra-supplied or user-supplied REXX functions.	95
SMOOTH	Smooths the specified datasource.	67
STARTMCA	Starts acquisition on an MCA hardware input.	69
STARTMCS	Performs MCS acquisition on up to eight datasources.	77
STOPMCA	Stops acquisition on an MCA hardware input.	70

STOPMCS	Stops MCS acquisition on up the current datasource.	80
STRIP	Strips one datasource from another.	67
TAKEOVER	Takes ownership of an MCA device associated with the specified datasource.	74
TENTTIVE	Performs a tentative nuclide identification on the specified datasource.	57
ULOADMID	Unloads an MID configuration file from the Runtime Database.	94
WAIT	Commands the Job to wait for a given condition.	92
WAITMCS	Commands the Job to wait for the current datasource to complete its MCS acquisition.	80
Routine Gamma Spectrum Analysis	Provides options to count and analyze a sample, count a sample and save the results, or analyze previously saved results.	132
Sample Changer Command Procedure	Counts and analyzes a set of samples.	176
Sample Counting and Filter Analysis	Provides an example of sample counting with minimal user input.	176

Introduction to the Batch Procedure Environment

This chapter assumes that you are familiar with the topics covered in this section: the editor, the command format, the batch procedure's environment variables, datasource naming conventions and control job commands.

The foundation of the Batch Procedures environment is a set of Analysis Job Commands, General Job Commands and Job Functions which are the executable components under the MS-DOS batch environment, the REXX environment, or both.

The Editor

The Editor creates and modifies Jobs (.REX files) and stores them in a file that may be processed by the MS-DOS Batch Environment. Any ASCII editor can be used to modify the .REX files on a line-by-line basis. It is assumed that if you need sophisticated structures such as complex “if” handling, looping, and so forth, you understand how to use advanced batch environments, such as REXX, to provide the needed functions. Canberra’s Training Department offers a course on using REXX in writing customized batch procedures. In addition, there are many books available which provide information on the REXX language.

Command Format

Each command description follows a basic format:

1. Command description
2. Command format
3. Command qualifiers and their descriptions¹
4. Examples of the command

Please note that many of the job commands have REXX function equivalents; these commands will be noted with an **RX** superscript (for example, PEAK_DIF^{RX}). Refer to “Registering and Using REXX Function Equivalents” on page 104 for a complete description of how to register and call these functions.

Optional qualifiers are enclosed in [brackets]. Note that complete pathnames have been omitted from the examples for clarity in presentation.

Some commands may use an environment variable instead of the qualifier (see “Environment Variables” on page 22).

1. All batch commands have at least one qualifier, [/HELP], which displays help text for the command.

White space (space, tab, and so forth) is allowed within a Command line and will be ignored by the command line parser. Each Command will return a value of zero [exit (0)] if the Command was successfully completed; a non-zero return signifies that an error occurred. It will be up to you to recognize the error and decide what to do with it (PAUSEONERROR or IF RC n command for instance).

Environment Variables

A set of Environment Variables within the Job Environment holds default information for various Commands (note that this set of Environment Variables does not apply to Functions). When several Commands within a Job need the same file (such as a datasource file), the file name can be specified in an Environment Variable rather than be repeated for each Command.

The precedence for determining where a particular parameter will be taken from is as follows:

1. A Qualifier, if present.
2. An Environment Variable, if defined.
3. The parameter within the CAM file.

The following Environment Variables can be defined once within the Job Environment and used repeatedly in successive Job Commands, as specified here.

JobInpSrc

All commands that use the <input datasource name> parameter will default to the datasource named in the JobInpSrc Environment Variable unless a datasource file name is explicitly included as part of the command.

JobOutSrc

All commands that use the <output datasource name> parameter will default to the datasource named in the JobOutSrc Environment Variable unless a datasource file name is explicitly included as part of the command.

JobOpName

The JobOpName Environment Variable can supply the Operator's name to each of the Analysis Job commands. In addition, it is used by the STARTMCA command (page 69) to store the Operator's name in a CAM parameter. If the JobOpName environment variable does not exist, a null string will be stored as the operator name.

BYPASS_VDM

If defined, the `BYPASS_VDM` environment variable instructs all subsequent job commands to open all CAM files directly (i.e. will bypass the VDM). Any detector input that is being opened will ignore this setting and continue to use the VDM. The intent of this environment variable is to improve performance of the job commands and does come with one restriction: any file opened for read/write *cannot* be shared with any other application.

Datasource Naming Conventions

The following naming convention is used when specifying datasources:

`[\\nodename\][device:]datasource name`

with the functions defined as follows:

Nodename

Optional parameter specifying name of computer node. Note that this name must be present in the Network Configuration File and must have the appropriate type (“M” or “B” for hardware). The leading `\\` and trailing `\` are required as part of the nodename syntax. If not specified, the local node is assumed.

Device

Optional parameter specifying name of device on which the datasource exists. The device name can either be a disk specifier (A: through Z:) or a special device name indicating hardware (DET: for detector inputs). The colon (:) is required as part of the device name syntax.

Datasource Name

Name of datasource. The standard operating system file naming conventions apply when referring to files; the hardware input/counter naming conventions of the VDM apply when referring to hardware (including memory groups).

Note that a fully specified pathname is required when accessing datasources since the application and the VDM have different definitions for “the current working directory”.

Control Job Commands

The Control Job Commands are the MS-DOS and REXX commands necessary for batch procedure control functions such as looping. Though all of the MS-DOS control commands are supported, you should be aware that a more comprehensive set of batch operations (complex IF testing, looping, mathematical computations) is available in the REXX language environment.

The following is a partial list of MS-DOS and REXX batch control commands which will be useful in the Job Environment:

CALL [MS-DOS, REXX]

Nests a batch file within a batch file.

CLS [MS-DOS]

Clears the display screen.

DO_END [REXX]

Tells the batch system that a group of instructions should be run *n* times.

ECHO [MS-DOS]

Displays a message and allows or prevents the display of MS-DOS commands while a batch file is running.

ENDLOCAL [MS-DOS]

Restores the drive, directory, and environment variables that were in effect before a Setlocal command was issued.

FOR [MS-DOS]

Allows repetitive processing of commands within a batch file.

GOTO [MS-DOS]

Transfers batch processing to a specified label.

IF [MS-DOS, REXX]

Allows conditional processing of commands within a batch file.

PAUSE [MS-DOS]

Suspends processing of the batch file.

PULL [REXX]

Prompts with a question mark (?). Puts an answer in memory after converting it to upper-case.

REM [MS-DOS]

The Rem command denotes a comment within a batch file.

SAY [REXX]

Tells the batch system to display words or a computed value on the screen.

SETLOCAL [MS-DOS]

Sets the drive, directory and environment variables that are local to the current batch file.

SIGNAL [REXX]

Transfers batch processing to a specified label.

Analysis Job Commands

The Analysis Job Commands are the commands which perform the spectroscopy analysis operations.

Standard Analysis Commands

Every Genie-2000 includes Peak Locate commands, which are used to locate spectral peaks, and Peak Area Analysis commands, which perform peak fitting and area calculations on the located peaks. There may be other commands available to you, depending on the options installed on your computer.

Optional Analysis Commands

The optional analysis commands, available only when an analysis option has been installed on your computer, are divided into these functional groups:

- Calibration
- Area Correction
- Efficiency Correction
- Nuclide Identification
- Detection Limits

Supporting the analysis commands are the Reporting, Editing, Options, Hardware Control, Application Interaction, and Miscellaneous Commands. These are covered in “General Job Commands,” starting on page 63.

Peak Locate

The Peak Locate group of commands locates peaks in a spectrum and builds a peak table.

PEAK_DIF^{RX}

(Applies to S500, S502, S504) The standard PEAK_DIF command performs an Unidentified Second Difference peak locate phase of analysis on the specified datasource.

Command Format

PEAK_DIF [<input datasource name>] [/Qualifier(s)]

Qualifiers

[/CHANNELS=<start channel, end channel>]

Specifies the spectrum channel range for the peak locate. If not specified, the analysis will be performed on the entire spectrum.

[/MARKERS]

Specifies that peak locate will be performed only on the spectrum between the Acquisition and Analysis window's Left and Right Markers.

[/SIGNIF=<significance value>]

Specifies the Significance factor used in the peak locate algorithm.

[/APPEND]

Merges the results with the input datasource's peak results table. If not specified, the existing results table will be replaced with the new results. This qualifier will use the datasource's mode (ETOL or FTOL) and tolerance value unless either the /ETOL or /FTOL qualifier is specified. Peaks within the tolerance value of a peak already in the table will not be merged.

[/ETOL=<value>]

If the /APPEND qualifier is specified, this qualifier causes the search to use this fixed energy tolerance, keV, to determine a match between a calculated peak and a peak already in the results table. Note that /ETOL and /FTOL are mutually exclusive.

[/FTOL=<value>]

If the /APPEND qualifier is specified, this qualifier causes the search to use a variable energy tolerance (in multiples of FWHM) to determine whether there is a match between a calculated peak and a peak already in the results table. Note that /FTOL and /ETOL are mutually exclusive.

Examples

```
PEAK_DIF MY_EXPER.CNF /CHANNELS=120,1000 /SIGNIF=4.1
```

Perform an Unidentified Second Difference peak locate on datasource MY_EXPER.CNF starting at channel 120 and ending at channel 1000. Use a Significance factor of 4.1 for the search.

PEAK_DIF MY_EXPER.CNF /MARKERS

Perform an Unidentified Second Difference peak locate on datasource MY_EXPER.CNF starting and ending at the channels specified by markers in the interactive Acquisition and Analysis window. Use the Significance factor in the datasource.

PEAK_LIB^{RX}

(Applies to S500, S502, S504) The standard PEAK_LIB command performs a library-driven peak locate on the specified datasource.

Command Format

PEAK_LIB [<input datasource name>] [/Qualifier(s)]

Qualifiers

/LIBRARY=<nuclide library file name>

Specifies the name of the Nuclide Library file (including the pathname as required) that will be used in the analysis. If you omit this qualifier, the library name currently in the datasource will be used. If this qualifier is set to a null string (/LIBRARY=""), no library will be used for the peak locate; only the unknown peaks will be found.

[/CHANNELS=<start channel, stop channel>]

Specifies the spectrum channel range which will be searched for peaks. If the range exceeds the range of energies in the library, all peaks from the library will be used. If not specified, the current value in the datasource will be used, or if the datasource has not been initialized, the analysis will be performed on the entire spectrum.

[/GAINSHIFT]

Specifies that gain shift correction will be performed. Sets and uses the flag parameter PRGAINSHFT.

[/ETOL=<value>]

Specifies the energy tolerance, as a FWHM multiplier, which determines whether a peak found during the unidentified peak locate is one of the peaks already assigned during the library locate. Peaks that are farther away from existing library locations than this tolerance are treated as separate peaks. Peaks that are closer to the existing library locations than this tolerance are assumed to be the library peaks already assigned. If not specified, the current value in the datasource will be used, or if the datasource is not initialized, will default to 0.5 for channels ≤ 1024 , otherwise to 1.5. Sets and uses the parameter UNIDETOL. Note that this tolerance

parameter is *not* the same as the ETOL parameters used during calibration or other analysis algorithms.

[/AREAW=<area window width>]

Specifies the FWHM multiplier to establish the left and right ROI limits around the peaks for peak locate fits (to determine their presence). If not specified, the current value in the datasource will be used, or if the datasource is not initialized, will default to 1.0 for channels ≤ 1024 , otherwise to 2.5. Sets and uses the parameter AREAWIND1.

[/INTFW=<interference window width>]

Specifies the FWHM multiplier to establish the limit beyond which two adjacent peaks are no longer considered to interfere with each other for the purpose of determining their presence. If not specified, the current value in the datasource will be used, or if the datasource is not initialized, will default to 1.5 for channels ≤ 1024 , otherwise to 3.0. Sets and uses the parameter INTFWIND1.

[/OVERLW=<overlap window width>]

Specifies the FWHM multiplier to establish the limit below which no attempt is made to establish the interference between two adjacent peaks for the purpose of determining their presence. If not specified, the current value in the datasource will be used, or if the datasource is not initialized, will default to 0.3. Sets and uses the parameter OVERWIND1.

[/LIBONLY]

Specifies that only the library derived peaks are to be analyzed. No unknown locate will be performed. If not specified, both the library peaks and the peaks found by the unknown locate will be analyzed. Resets and uses the flag parameter DOUNKSRCH.

Examples

```
PEAK_LIB MY_EXPER.CNF /LIBRARY=STDLIB.NLB /LIBONLY
```

Perform a library peak locate on datasource MY_EXPER.CNF using library STDLIB.NLB without the unknown locate.

```
PEAK_LIB MY_EXPER.CNF /LIBRARY=STDLIB.NLB /CHANNELS=1000,2000
```

Perform a library peak locate, including the unknown locate, on datasource MY_EXPER.CNF using library STDLIB.NLB between channels 1000 and 2000.

PEAK_SLB^{RX}

(Applies to S500, S502, S504) The standard PEAK_SLB command performs a “simple library” peak locate on the specified datasource. No complex gainshift correction or other techniques are used.

Command Format

PEAK_SLB [<input datasource name>] [/Qualifier(s)]

Qualifiers

[/LIBRARY=<nuclide library file name>]

Specifies the name of the .NLB file that will be used in the analysis. If omitted, the library name in the input datasource will be used.

[/CHANNELS=<start channel, stop channel>]

Specifies the spectrum channel range to be searched for peaks. If the range exceeds the range of energies in the library, all peaks from the library will be used. If not specified, the values in the input datasource will be used.

[/APPEND]

Merges the results with the input datasource’s peak results table. If not specified, the existing results table will be replaced with the new results. This qualifier will use the datasource’s mode (ETOL or FTOL) and tolerance value unless either the /ETOL or /FTOL qualifier is specified. Peaks within the tolerance value of a peak already in the table will not be merged.

[/ETOL=<value>]

If the /APPEND qualifier is specified, this qualifier causes the search to use this fixed energy tolerance, keV, to determine a match between a calculated peak and a peak already in the results table. Note that /ETOL and /FTOL are mutually exclusive.

[/FTOL=<value>]

If the /APPEND qualifier is specified, this qualifier causes the search to use a variable energy tolerance (in multiples of FWHM) to determine whether there is a match between a calculated peak and a peak already in the results table. Note that /FTOL and /ETOL are mutually exclusive.

Example

```
PEAK_SLB MY_EXPER.CNF /LIBRARY=STDLIB.NLB  
/CHANNELS=1000,2048 /APPEND
```

Perform a simple library peak locate on MY_EXPER.CNF using library STDLIB.NLB between channels 1000 and 2048 and merge the results with the peak

results table in MY_EXPER.CNF. Use the datasource's tolerance settings to determine whether a calculated peak might overwrite an existing entry in the datasource's peak results table.

PEAK_ROI^{RX}

(Applies to S500, S502, S504) The standard PEAK_ROI command calculates a peak centroid for each ROI in the source file, which is either the specified datasource or the specified ROI file. These centroids make up a list which can either replace or be appended to the datasource's peak results table.

Command Format

PEAK_ROI [<input datasource name>] [/Qualifier(s)]

Qualifiers

[/CHANNELS=<start channel, stop channel>]

Specifies the spectrum channel range for calculating peak centroids. If this range exceeds the range of energies in the source file, all of the ROIs in the source file will be used. If not specified, the current range in the datasource will be used.

[/ROIFILE=<ROI file name>]

Specifies the name of the file to read the ROI definitions from. If not specified, the ROIs are read from the input datasource.

[/APPEND]

Merges the calculated results with the input datasource's peak results table. If not specified, the existing results table will be replaced with the new results. This qualifier will use the datasource's mode (ETOL or FTOL) and tolerance value unless either the /ETOL or /FTOL qualifier is specified. Peaks within the tolerance value of a peak already in the table will not be merged.

[/ETOL=<value>]

If the /APPEND qualifier is specified, this qualifier causes the search to use this fixed energy tolerance, in keV, to determine a match between a calculated peak and a peak already in the results table. Note that /ETOL and /FTOL are mutually exclusive.

[/FTOL=<value>]

If the /APPEND qualifier is specified, this qualifier causes the search to use a variable energy tolerance (in multiples of FWHM) to determine whether there is a match between a calculated peak and a peak already in the results table. Note that /FTOL and /ETOL are mutually exclusive.

Example

```
PEAK_ROI MY_EXPER.CNF /CHANNELS=120,1000 /APPEND /ETOL=1.00
```

Calculate the peak centroids in datasource MY_EXPER.CNF starting at channel 120 and ending at channel 1000. Merge calculated centroids which are not within 1 keV of a peak in the datasource's peak results table. Peaks within 1 keV of an existing centroid will not be merged.

PEAK_STD^{RX}

(Applies to S500, S502, S504) The PEAK_STD command runs the VMS Standard Peak Search. This performs an Unidentified Second Difference peak locate followed by a pure Gaussian Peak Fit phase of analysis on the specified datasource. The VMS Standard Peak Search is taken from the Genie-VMS application as an optional Peak Locate/Analysis engine.

Command Format

```
PEAK_STD [<input datasource name> [/Qualifier(s)]
```

Qualifiers

```
[/CHANNELS=<start channel, end channel>]
```

Specifies the spectrum channel range for the peak search. If not specified, the peak search will be performed using the start and end channels stored in the datasource. This qualifier is mutually exclusive with the /MARKERS qualifier. As a batch command, the /CHANNELS qualifier overrides the /MARKERS qualifier. As a REXX equivalent function, both qualifiers are ignored and the peak search will be performed using the start and end channels stored in the datasource.

```
[/MARKERS]
```

Specifies that the peak search will be performed only on the spectrum between the Left and Right Markers as last stored through the Acquisition and Analysis window. This qualifier is mutually exclusive with the /CHANNELS qualifier. As a batch command, the /CHANNELS qualifier overrides the /MARKERS qualifier. As a REXX equivalent function, both qualifiers are ignored and the peak search will be performed using the start and end channels stored in the datasource.

```
[/SENSITVITY=<value>]
```

Number greater than 0 which is used to "eliminate" insignificant peaks (i.e. peaks whose significance value is less than threshold are deemed insignificant). Typical values for this parameter range from 3 - 5. Reasonable values for solid state detectors are 3 - 15.

[/GSENSITVTY=<value>]

Allows specification of the peak search Gaussian sensitivity parameter. Smaller values mean the search will demand more cleanly shaped peaks. Reasonable values are between 5 and 30. Value must be greater than 0. Scintillation detector spectra will generally require larger values for this parameter.

[/ITERATIONS=<value>]

The maximum number of iterations peak search will perform when fitting a gaussian function to a multiplet. The minimum value is 0; a reasonable value is 10.

[/FWHMREJ]

This flag indicates whether or not the standard peak search function will reject singlet peaks whose (measured FWHM) / (expected FWHM) ratio is less than a user selectable threshold. This option is very useful if you typically analyze spectra at low sensitivities (e.g. less than 3).

[REJRATIO=<value>]

This value specifies the minimum valid (measured FWHM) / (expected FWHM) ratio that will be accepted for a singlet peak. If the ratio for a peak is less than this value the peak will be rejected. This parameter is used only if “FWHM Based Rejection” flag (/FWHMREJ) is enabled. Minimum value is 0; maximum value is 1.0.

[/FITSING]

Used by Peak Search to determine how to analyze singlet peaks. If OFF (or not included as a qualifier), singlet peak characteristics are calculated by the total peak area method. If ON (or included as a qualifier), the peaks are fitted to a single peak gaussian function.

[/CRITLEVEL]

This parameter answers the question, “Should a critical level test be performed ?” If this parameter is enabled, any corrected peak that has a net area less than k times the uncertainty of the background, where k is the Confidence Level parameter, will be discarded.

[/FIXFWHM]

This flag indicates whether or not the FWHM of peaks in a region will be held fixed to their calibration value or will be allowed to freely vary during the fitting process (Fit Singlets enabled or analyzing multiplets).

[/DISPLAY_ROIS]

This flag determines if ROIs calculated during the area analysis are copied to the “display ROIs” section of the datasource, thus causing the ROIs to appear in an MCA View/Control window when the datasource is viewed.

The ROI color conventions are as follows:

- Color 1: Singlet peaks
- Color 2: Multiplet peaks
- Color 3: Not used
- Color 4: Not used

Any previously displayed ROIs are cleared before generating the new ROI list.

Example

PEAK_STD MY_EXPER.CNF /CHANNELS=1024,2048 /ITERATIONS=10

Perform the standard peak locate in channels 1024 to 2048 of datasource MY_EXPER.CNF. Perform up to 10 iterations when fitting a gaussian function to a multiplet.

Peak Area Analysis

The Peak Area Analysis group performs peak fitting and area calculations on the peaks in the datasource’s peak table.

AREA_NL1^{RX}

(Applies to S500, S502, S504) This performs a Non-Linear Least Squares Fit for multiplets and either a region Summation or a Fit for singlets.

Command Format

AREA_NL1 [<input datasource name>] [/Qualifier(s)]

Qualifiers

[/CONT=<channels>]

Specifies the number of continuum channels that will be used on each side of the peak. This is the same continuum that is used by other analysis functions (see “ECAL” on page 37). Note that /CONT and /FCONT are mutually exclusive; the last one encountered on the command line will be honored. If neither is specified, the value in the datasource will be used.

[/FCONT=<value>]

Specifies that variable continuum methodology will be used with the FWHM multiplier provided by <value>. Note that /FCONT and /CONT are mutually exclusive; the last one encountered on the command line will be honored. If neither is specified, the value in the datasource will be used.

[/CRITLEVEL]

Specifies that Critical Level checking should be performed. The default is no critical level checking.

[/MARKERS]

Specifies that the spectrum channel where the peak analysis starts will be the Left Marker and that the end will be the Right Marker in the interactive Acquisition and Analysis window.

[/CHANNELS=<start channel, end channel>]

Specifies the spectrum channel range where the peak analysis will be performed. If not specified, the analysis will be command performed on the entire spectrum.

[/DISPLAY_ROIS]

Specifies that the ROIs resulting from the peak area analysis be copied to the Display ROIs block of the datasource. These ROIs can then be viewed in an Acquisition and Analysis window looking at this datasource.

[/ETOL=<value>]

Specifies the fixed energy tolerance, in keV, used to determine a match between a peak in the datasource and the 511 keV annihilation peak. If not specified, the current value in the datasource will be used. Note that /ETOL and /FTOL are mutually exclusive.

[/FTOL=<value>]

Specifies the variable energy tolerance, in multiples of expected FWHM, used to determine a match between a peak in the datasource and the 511 keV annihilation peak. If not specified, the current value in the datasource will be used. Note that /FTOL and /ETOL are mutually exclusive.

[/FIXTAIL]

Specifies that the algorithm will use the tail parameter values from the calibration for all multiplet peaks instead of allowing the value to vary for best fit. If not specified, the tail parameter will be allowed to vary.

[/FIXFWHM]

Specifies that the algorithm will use the FWHM parameter values from the calibration for all multiplet peaks instead of allowing the value to vary for best fit. If not specified, the FWHM parameter will be allowed to vary.

[/FIT]

Specifies that singlet peaks will be fitted, rather than calculated using the summation method. If not specified, only multiplets will be fitted.

[/RES_SEARCH]

Specifies that the residual search will be performed, using the threshold value specified by /RES_THRESHOLD and the minimum peak separation specified by /RES_SEPARATION.

[/RES_THRESHOLD]

Specifies the threshold value to be used by the /RES_SEARCH qualifier.

[/RES_SEPARATION]

Specifies the minimum peak separation value (in FWHM) to be used by the /RES_SEARCH qualifier.

Examples

```
AREA_NL1 MY_EXPER.CNF /CHANNELS=120,1000
```

Perform a Non-Linear Least Squares Fit peak analysis of datasource MY_EXPER.CNF starting at channel 120 and ending at channel 1000. Use the Continuum that is specified in the datasource. Don't perform critical level checking.

```
AREA_NL1 MY_EXPER.CNF /CHANNELS=200,500 /CONT=4 /CRITLEVEL
```

Perform a Non-Linear Least Squares Fit peak analysis of datasource MY_EXPER.CNF starting at channel 200 and ending at channel 500 using 4 continuum channels. Perform critical level checking.

AREA_LIB^{RX}

(Applies to S500, S502, S504) The AREA_LIB command performs a library-driven peak area analysis on the specified datasource.

Command Format

```
AREA_LIB [<input datasource name>] [/Qualifiers]
```

Qualifiers

[/GAINSHIFT]

Specifies that gain shift correction will be performed. If not specified, the gain shift correction will not be performed. Sets and uses the flag parameter PRGAINSHIFT.

[/DISPLAY_ROIS]

Specifies that the ROIs resulting from the analysis are to be copied to the Display ROIs block in the datasource. These ROIs can be viewed in an Acquisition and Analysis window looking at this datasource. If not specified, the ROIs will not be copied to the Display ROIs block. Sets and uses the flag parameter SHOWROIS.

[/MAXGAINPASS=<integer value>]

Specifies the maximum number of iterations allowed for the fits with the automatic gain correction. If not specified, the current value specified in the datasource will be used or if not initialized, will default to 10. Sets and uses the parameter MAXGAINPASS.

[/GAINREJ=<value>]

Specifies the convergence criterion. When consecutive iterations produce a change in the chi-square less than this value, a convergence has been achieved. If not specified, the current value specified in the datasource will be used or if not initialized, will default to 0.005 for channels ≤ 1024 , otherwise to 0.0001. Sets and uses the parameter GAINREJ.

[/AREAW=<area window width>]

Specifies the FWHM multiplier to establish the left and right ROI limits around the peaks for peak locate fits (to determine their areas). If not specified, the current value in the datasource will be used or if not initialized, will default to 1.5 for channels ≤ 1024 , otherwise to 3.0. Sets and uses the parameter AREAWIND2.

[/INTFW=<interference window width>]

Specifies the FWHM multiplier to establish the limit beyond which two adjacent peaks are no longer considered to interfere with each other for the purpose of area calculations. If not specified, the current value in the datasource will be used or if not initialized, will default to 2.5 for channels ≤ 1024 , otherwise to 4.0. Sets and uses the parameter INTFWIND2.

[/OVERLW=<overlap window width>]

Specifies the FWHM multiplier to establish the limit below which no attempt is made to establish the interference between two adjacent peaks for the purpose of

area calculations. If not specified, the current value in the datasource will be used or if not initialized, will default to 0.3. Sets and uses the parameter OVERWIND2.

Examples

```
AREA_LIB MY_EXPER.CNF /GAINSHIFT
```

Perform a library area calculation of datasource MY_EXPER.CNF with the gain shift correction enabled.

```
AREA_LIB MY_EXPER.CNF /DISPLAY_ROIS /GAINSHIFT /GAINREJ=0.01
```

Perform a library area calculation of datasource MY_EXPER.CNF with the gain shift correction and displaying the ROIs in the application window as well as using a non-default convergence criterion for the gain shift.

Calibration

The Calibration group of commands performs the various calibrations for the hardware chain.

ECAL^{RX}

(Applies to S500, S502, S504) The ECAL command is used to perform an energy, FWHM and tailing calibration for the specified datasource. An energy/channel pair list will be matched against a Certificate file. For the matching peaks, the pair's centroid channel and the energy from the Certificate file will be entered into the Energy Calibration table. The calculated FWHM and the tailing for the peak will also be entered.

Initial Calibration

If an INIT qualifier is specified, an Initial Calibration will be performed. In this mode, the system asks for the centroid channel for each energy line in the specified certificate file. The centroid can be found by cursor location, by marker locations, by entered value, or by table lookup, depending on which INIT qualifier is specified. Note that performing an Initial Calibration will replace any existing calibration for the datasource.

By Cursor

(INIT_C) The centroid is entered by positioning the cursor in the Acquisition and Analysis window, then pressing ENTER to accept it or typing N to skip that line. Accepting the line will cause the exact centroid, FWHM and, optionally, the tail parameters to be calculated and displayed. Press ENTER to store the calculated values or N to discard them. When all lines have been processed, the calibration coefficients will be calculated and displayed for approval.

By Markers

(INIT_M) The centroid is entered by positioning the markers around a peak in the Acquisition and Analysis window, then pressing ENTER to accept the centroid or typing N to skip that line. Accepting the line will cause the exact centroid, FWHM and, optionally, the tail parameters to be calculated and displayed. Press ENTER to store the calculated values or N to discard them. When all lines have been processed, the calibration coefficients will be calculated and displayed for approval.

By Value

(INIT_V) If the centroid line is not known, press ENTER to skip that line. If a centroid value is entered, press ENTER to accept the centroid or N to skip that line. Accepting the line will cause the exact centroid, FWHM and, optionally, the tail parameters to be calculated and displayed. Press ENTER to store the calculated values or N to discard them. When all lines have been processed, the calibration coefficients will be calculated and displayed for approval.

By Datasource

(INIT_D) This command uses existing energy/channel or energy/ROI pairs already in the datasource's lookup table to calculate the calibration coefficients. Note that the results of the calculation are not displayed.

Update Calibration

If no INIT qualifier is specified, an initial calibration is assumed to have already been done and an update calibration is performed. For all lines in the certificate file, the update calibration attempts to find and fit a peak to each location; if a peak cannot be fitted (for example, if an isotope in the calibration source has decayed too far), that line is omitted from the calibration.

Command Format

ECAL [<input datasource name>] /Qualifier(s)

Qualifiers

/CERT=<certificate file name>

Required unless /INIT_D is specified. Specifies the name of the Certificate file which will be used in the calibration.

[/NOTAIL]

Specifies that a tailing calibration should not be performed.

[/CONT=<continuum channels>]

Specifies the number of continuum channels that will be used on each side of the peak. This is the same continuum that is used by other analysis functions (see "AREA_NL1" on page 33). Note that /CONT and /FCONT are mutually

exclusive; the last one encountered on the command line will be honored. If neither is specified, the value in the datasource will be used.

[/FCONT]

Specifies the number of continuum channels that will be used on each side of the peak as a function of FWHM instead of channels. Note that /FCONT and /CONT are mutually exclusive; the last one encountered on the command line will be honored. If neither is specified, the value in the datasource will be used.

[/INIT_C], [/INIT_M], [/INIT_V], [/INIT_D]

Specify that an Initial Calibration is to be performed, finding the centroids by: cursor location (INIT_C), by marker locations (INIT_M), by entered value (INIT_V), or by datasource table lookup (INIT_D). If no INIT qualifier is specified, an Update Calibration is performed.

[/ORDER=<order of energy calibration equation>]

Specifies the order (degree – maximum is third) of the generated energy calibration equation. If not specified and is 0 in the specified datasource, it defaults to first order (linear). Note that at least two peaks are required for first order, three peaks for second order and four peaks for third order.

[/ETOL=<value>]

Specifies the fixed energy tolerance, in keV, used to determine a match between certificate energies and the energies in the datasource's calibration list. If not specified, the current value in the datasource will be used. Note that /ETOL and /FTOL are mutually exclusive.

[/FTOL=<value>]

Specifies the variable energy tolerance (in multiples of expected FWHM) used to determine a match between certificate energies and the energies in the datasource's calibration list. If not specified, the current value in the datasource will be used. Note that /FTOL and /ETOL are mutually exclusive.

Examples

```
ECAL MY_EXPER.CNF /CERT=NBS_CERT.CTF
```

Perform an Update calibration using Certificate file NBS_CERT.CTF and datasource MY_EXPER.CNF. Use the Tolerance and Continuum from the datasource.

ECAL /INIT_V /CERT=NBS_CERT.CTF

Perform an Initial calibration using Certificate file NBS_CERT.CTF and the datasource specified in the JobInpSrc Environment Variable. Use the Tolerance and Continuum listed in the datasource. Centroids are entered by value.

RECAL^{RX}

(Applies to S500, S502, S504) The RECAL command is used to perform an energy (FWHM and tailing are not recalculated) recalibration for the specified datasource. Recalibration is allowed only on datasources which have been previously calibrated.

If a Peak Locate has not been performed, one will be automatically performed. If a Peak Locate has been performed, its results will be used. Either a Certificate file or the current energy/channel pair list is used to select peaks from the specified spectrum to be used as the basis for the recalibration.

Command Format

RECAL [<input datasource name>] [/Qualifier(s)]

Qualifiers

[/CERT=<certificate file name>]

Specifies the name of the Certificate file which will be used for the Recalibration. If not specified, the energy/channel pair list contained within the input datasource will be used for the Recalibration.

[/PLENGINE=<generic engine name>]

Specifies the peak locate algorithm to invoke during energy calibration. The algorithm name must exist in the currently defined Analysis Engine File (the one pointed to by the environment variable GAMMAAEF). If not specified, the default algorithm will be the first one found in the Analysis Engine File for the Peak Locate phase. For more information on algorithm names, refer to the section on Analysis Engine Files in the “Genie-2000 Configuration” chapter of the *Genie-2000 Operations Manual*.

[/ETOL=<value>]

Specifies the fixed energy tolerance, in keV, used to determine a match between certificate energies and the energies in the datasource’s calibration list. If not specified, the current value in the datasource will be used. Note that /ETOL and /FTOL are mutually exclusive.

[/FTOL=<value>]

Specifies the variable energy tolerance, in multiples of expected FWHM used to determine a match between certificate energies and the energies in the

datasource's calibration list. If not specified, the current value in the datasource will be used. Note that /FTOL and /ETOL are mutually exclusive.

Examples

```
RECAL AIR_SMP.CNF /CERT=NBS_CERT.CTF
```

Perform an Energy Recalibration using Certificate file NBS_CERT.CTF, datasource AIR_SMP.CTF and the peaks of the Certificate file. Use the Tolerance in the datasource.

```
RECAL
```

Perform an Energy Recalibration using the datasource specified in the JobInpSrc Environment Variable. Use the Tolerance and the peaks of the energy/channel pair list in the datasource.

EFFCAL^{RX}

(Applies to S500, S502, S504) The EFFCAL command performs an efficiency calibration on the specified datasource, which must contain an Energy Calibration. Note that all supported equation types (polynomial, linear, empirical) will be generated.

Like interactive Efficiency Calibration, EFFCAL will use either the algorithms stored in the datasource, or if none are defined, the first Peak Locate and Peak Area algorithms it finds in the spectroscopy application's .AEF file, unless you specify either or both of the /PAENGINE and /PLENGINE qualifiers. Unlike interactive Efficiency Calibration, this command stores the results of the calculations in the datasource.

This command generates a table of energy, efficiency and percent error values for each peak in the datasource that matches (within the specified tolerance) a peak in the Certificate file. This table is used to generate the various polynomial equations (linear, dual, and so forth) that define the efficiency of the detector and electronics. The order of the various efficiency equations is automatically determined by the number of points specified, as follows:

Low curve (dual polynomial with crossover):

Order = 2 (points > 2)

High curve (dual polynomial w/crossover), linear curve:

Order = 2 ($3 \leq \text{points} \leq 5$)

Order = 3 ($6 \leq \text{points} \leq 7$)

Order = 4 ($8 \leq \text{points} \leq 9$)

Order = 5 (points ≥ 10)

Command Format

EFFCAL [<input datasource name>] /CERT=<certificate file name> [/Qualifier(s)]

Required Qualifier

/CERT=<certificate file name>

Specifies the name of the Certificate file which will be used in the calibration.

Optional Qualifiers

[/CROSSOVER=<crossover energy>]

Specifies the crossover energy to be used when calculating the dual polynomial equation. If the crossover point is not present in the specified certificate file, the closest higher energy point is used. If not specified, a single polynomial equation will be generated.

[/APPEND]

Specifies that these table entries will be merged with any table entries from a previous efficiency calibration. If not specified, any existing table entries will be cleared before doing this efficiency calibration.

[/PLENGINE=<generic engine name>]

[/PAENGINE=<generic engine name>]

Specifies the particular algorithm to invoke during efficiency calibration for peak locate, peak analysis, or both. The algorithm name must exist in the currently defined Analysis Engine File (the one pointed to by the environment variable GAMMAAEF). If not specified, the default algorithms will be the first ones found in the Analysis Engine File for the given analysis phase. For more details on algorithm names, refer to “Analysis Engine Files” in the *Genie-2000 Configuration* chapter in Volume 1, *Genie-2000 Operations*.

[/ETOL=<value>]

Specifies the fixed energy tolerance, in keV, used to determine a match between a peak in the datasource and a peak in the certificate file. If not specified, the current value in the datasource will be used. Note that /ETOL and /FTOL are mutually exclusive.

[/FTOL=<value>]

Specifies the variable energy tolerance, in multiples of expected FWHM, used to determine a match between a peak in the datasource and a peak in the certificate file. If not specified, the current value in the datasource will be used. Note that /FTOL and /ETOL are mutually exclusive.

[/USEPKRESULTS]

Specifies that the existing peak locate and peak area results be used in the Efficiency Calibration performed by this batch command.

[/INIT_D]

Specifies that the EFFCAL command use the existing energy/efficiency/error triplets within the specified input datasource and proceed immediately to the curve(s) generation stage of processing.

Examples

```
EFFCAL NBS_SPEC.CNF /ETOL=1.0 /CROSSOVER=391 /CERT=ABC.CTF
```

Perform an efficiency calibration on datasource NBS_SPEC.CNF. The matching Tolerance is 1.0 keV, the Crossover between the high and low equations is 391 keV. Use the Certificate file ABC.CTF. Use the Energy Calibration that is present in the specified datasource.

```
EFFCAL /CERT=NBS_CERT.CTF
```

Perform an efficiency calibration on the datasource specified in Environment Variable JobInpSrc. Use the matching Tolerance and Energy Calibration in the datasource. Use the Certificate file NBS_CERT.CTF. Crossover of 0 will force the Dual solution to be a single polynomial.

CALPLOT

(Applies to S500, S502, S504) The CALPLOT command is used to either print or show an Energy or Efficiency calibration curve as a graph. Unless the /PRINT (send the curve(s) to the hardcopy device) qualifier is specified, the CALPLOT command will specify a PM-based Show window with which the Job will interact. The type of calibration to be printed or shown is specified with the /CAL= qualifier. The /EDIT qualifier enables the Order of the Polynomial and Drop Peaks functions on the screen. For a detailed description of these two controls, refer to “Energy Show”, for Energy Calibration curves, or “Efficiency Show”, for Efficiency Calibration curves, in the *Gamma Acquisition and Analysis* chapter in Volume 1, *Genie-2000 Operations*.

Command Format

```
CALPLOT [<input datasource name>] [/Qualifier(s)]
```

Qualifiers

[/CAL=ENERGY]

Specifies that the Energy Calibration curve will be shown or printed; this is the default if no other CAL qualifier is specified.

[/CAL=SHAPE]

Specifies that the Shape Calibration curves will be shown or printed.

[/CAL=DUAL]

Specifies that the Dual Efficiency Calibration curve will be shown or printed.

[/CAL=LINEAR]

Specifies that the Linear Efficiency Calibration curve will be shown or printed.

[/CAL=EMPIRICAL]

Specifies that the Empirical Efficiency Calibration curve will be shown or printed.

[/CAL=ALPHA]

Specifies that the Alpha Average Efficiency Calibration curve will be shown or printed. This qualifier is useful only for efficiency calibrations performed with the Model S509 Alpha Acquisition and Analysis software.

[/CAL=PTCAL]

Specifies that the Peak/Total calibration curve will be shown or printed.

[/LOG]

Formats the plot in Log/Log scale, rather than the default Linear/Linear scale.

[/EDIT]

Enables the Order of the Polynomial and Drop Peak screen functions to allow updates to the equation. If the Edit and Print qualifiers are both specified, only the Print qualifier will be executed.

[/PRINT]

Specifies that the Calibration curve(s) selected will be sent directly to the printer or plotter (the PRN: device) with no PM-based window for interaction. If the Edit and Print qualifiers are both specified, only the Print qualifier will be executed.

Examples

```
CALPLOT SPECT3.CNF /CAL=SHAPE /EDIT
```

Put a Shape Calibration window on the screen for the SPECT3 datasource. All of the edit controls are enabled for interaction.

CALPLOT /CAL=DUAL /PRINT

Send a plot of the Dual Efficiency Calibration curve for the datasource specified in the Environment Variable JobInpSrc to the default print device. No PM window will be displayed and no editing will be allowed.

AECAL^{RX}

(Applies to S509) The AECAL command is used to perform an energy and shape calibration for the specified datasource. The usual calibration method loads a list of energies from a Certificate file. For each energy in the list which matches a peak in the spectrum, an entry is made in the Energy Calibration table, consisting of a centroid channel, the certificate energy, and the calculated FWHM and tail. This table is then used to generate the calibration equations.

Command Format

AECAL <input datasource name> [/Qualifiers]

Qualifiers

/CERT=<certificate file name>

Required unless /INIT_D is specified. Specifies the name of the Certificate file which will be used in the calibration.

[/NOTAIL]

Specifies that a tail calibration should not be performed.

[/INIT_C], [/INIT_M], [/INIT_V], [/INIT_D]

Specify that an Initial Calibration is to be performed, finding the centroids by: cursor location (INIT_C), by marker locations (INIT_M), by entered value (INIT_V), or by datasource table lookup (INIT_D). If no INIT qualifier is specified, an Update Calibration is performed.

[/ORDER=<order of energy calibration equation>]

Specifies the order (degree – maximum is third) of the generated energy calibration equation. If not specified, it defaults to first order (linear). Note that a minimum of two peaks are required for first order, three peaks for second order and four peaks for third order.

[/ETOL=<value>]

Specifies the fixed energy tolerance, in keV, used to determine a match between a peak in the input datasource and a peak in the certificate file. If not specified, the current value in the datasource will be used. Note that /ETOL and /FTOL are mutually exclusive.

[/FTOL=<value>]

Specifies the variable energy tolerance, in multiples of expected FWHM, used to determine a match between a peak in the input datasource and a peak in the certificate file. If not specified, the current value in the datasource will be used. Note that /FTOL and /ETOL are mutually exclusive.

Example

```
AECAL MY_EXPER.CNF /ETOL=20
```

Perform an energy and FWHM calibration on datasource MY_EXPER.CNF using an energy tolerance of 20 keV to determine whether a calculated peak might overwrite an existing entry in the datasource's peak results table.

ARECAL^{RX}

(Applies to S509) The ARECAL command is used to perform an energy recalibration on the specified datasource; FWHM and tail are not recalculated. Recalibration is allowed only on datasources which have been previously calibrated.

If a Peak Locate has not been performed, one will be automatically performed. If a Peak Locate has been performed, its results will be used. Either a Certificate file or the current energy/channel pair list is used to select peaks from the specified spectrum to be used as the basis for the recalibration.

Command Format

```
ARECAL [<input datasource name>] [/Qualifiers]
```

Qualifiers

[/CERT=<certificate file name>]

Specifies the name of the Certificate file which will be used for the Recalibration. If not specified, the energy/channel pair list contained within the input datasource will be used for the Recalibration.

[/PLENGINE=<generic engine name>]

Specifies the peak locate algorithm to invoke during energy calibration. The algorithm name must exist in the currently defined Analysis Engine File (the one pointed to by the environment variable ALPHAAEF). If not specified, the default algorithm will be the first one found in the Analysis Engine File for the Peak Locate phase.

[/ETOL=<value>]

Specifies the fixed energy tolerance, in keV, used to determine a match between a peak in the input datasource and a peak in the certificate file. If not specified, the

current value in the datasource will be used. Note that ETOL and FTOL are mutually exclusive.

[/FTOL=<value>]

Specifies the variable energy tolerance, in multiples of expected FWHM, used to determine a match between a peak in the input datasource and a peak in the certificate file. If not specified, the current value in the datasource will be used. Note that FTOL and ETOL are mutually exclusive.

Example

```
ARECAL MY_EXPER.CNF /CERTFILE=MY_CERT.CTF /FTOL=2.00
```

Perform an energy recalibration on datasource MY_EXPER.CNF with certificate file MY_CERT.CTF using a tolerance of 2.00 FWHM to determine whether a calculated peak might overwrite an existing entry in the datasource's peak results table.

AEFFCAL^{RX}

(Applies to S509) The AEFFCAL command is used to perform an efficiency calibration on the specified datasource, which must contain an Energy Calibration.

Like interactive Efficiency Calibration, AEFFCAL will use either the algorithms stored in the datasource, or if none are defined, the first Peak Locate and Peak Area algorithms it finds in the spectroscopy application's .AEF file, unless you specify either or both of the /PAENGINE and /PLENGINE qualifiers. Unlike interactive Efficiency Calibration, this command stores the results of the calculations in the datasource.

This command generates a table of energy, efficiency and percent error values for each peak in the specified datasource that matches (within the specified tolerance) a peak in the Certificate file. This table is used to generate the average efficiency calibration that defines the efficiency of the detector and electronics.

Command Format

```
AEFFCAL [<input datasource name>] [/Qualifiers]
```

Required Qualifier

```
/CERT=<certificate file name>
```

Specifies the name of the Certificate file which will be used in the calibration.

Optional Qualifiers

[/APPEND]

Specifies that table entries will be merged with any table entries from a previous efficiency calibration. If not specified, any existing table entries will be cleared before doing this efficiency calibration.

[/PLENGINE=<generic engine name>]

[/PAENGINE=<generic engine name>]

Specifies the particular algorithm to invoke during efficiency calibration for peak locate, peak analysis, or both. The algorithm name must exist in the currently defined Analysis Engine File (the one pointed to by the environment variable ALPHA_AEF). If not specified, the default algorithms will be the first ones found in the Analysis Engine File for the given analysis phase.

[/ETOL=<value>]

Specifies the fixed energy tolerance, in keV, used to determine a match between a peak in the input datasource and a peak in the certificate file. If not specified, the current value in the datasource will be used. Note that ETOL and FTOL are mutually exclusive.

[/FTOL=<value>]

Specifies the variable energy tolerance, in multiples of expected FWHM, used to determine a match between a peak in the input datasource and a peak in the certificate file. If not specified, the current value in the datasource will be used. Note that FTOL and ETOL are mutually exclusive.

[/INIT_D]

Specifies that the AEFFCAL command use the existing energy/efficiency/error triplets within the specified input datasource and proceed immediately to the curve(s) generation stage of processing.

Example

```
AEFFCAL MY_EXPER.CNF /CERTFILE=MY_CERT.CTF /APPEND
```

Perform an energy recalibration on datasource MY_EXPER.CNF using certificate file MY_CERT.CTF and merge the calculated results with the datasource's peak results table using the datasource's tolerance value settings to avoid replacing an existing entry in the datasource's peak results table with a calculated peak.

PTCAL

Coincidence Summing Correction Software is a set of algorithms that will allow the user to generate a Peak-to-Total efficiency ratio (P/T) calibration; i.e., calculate energy and Peak/Total efficiency ratio, and its uncertainty values, from multiple spectra, each measured with single line nuclides. Using this information, the software will calculate a Peak/Total efficiency curve using the same functional representation that is used in the DUAL peak efficiency curve, and permit correcting the nuclide identification results for coincidence summing.

Command Format

PTCAL <input datasource name> /filelist = <filename> [/Qualifiers]

Required Qualifiers

[/DATASOURCE]

The name of the primary file (or detector input) to be used for the Peak/Total calculations.

[/FILELIST]

Defines the name of the file that includes the names of all the files that include the spectral data. This qualifier is mandatory in the command line mode.

Optional Qualifiers

[/BACKGR]

Specifies the name of a background file when one is used.

[/CROSSOVER]

Specifies the crossover energy for the Peak/Total curve. If this qualifier is omitted a crossover energy of zero (single curve) will be used.

[/APPEND]

Specifies that the current set of spectral results are to be appended to the Peak/Total results already stored in the datasource.

[/ EXCHANNELS]

Specifies the number of channels to extrapolate down to channel zero

[/WINDOW]

Specifies the average window width.

[/NL]

Specifies the order of the low energy Peak/Total calibration curve. If the qualifier is not given, the order will be selected by an automatic selection mechanism that is dependent on the number of calibration energies.

[/NH]

Specifies the order of the high energy Peak/Total calibration curve. If the qualifier is not given, the order will be selected by an automatic selection mechanism that is dependent on the number of calibration energies.

[/INTR]

Specifies the number of iterations. If omitted a default will be used.

[/ETOL]

Specifies the energy tolerance in keV. Note that /ETOL and /FTOL are mutually exclusive.

[/FTOL]

Specifies the energy tolerance in FWHM t. Note that /FTOL and /ETOL are mutually exclusive.

Example

```
PTCAL MY_EXPER.CNF /DATASOURCE=MY_CERT.CTF /FILELIST=MY_LIST.TXT
```

Generates a Peak-to-Total calibration from the spectral data in the files listed in MY_LIST.TXT, using the MY_CERT.CTF datasource.

Area Correction

The Area Correction group of commands perform the corrections to the calculated area of peaks in the peak table.

AREACOR^{RX}

(Applies to S501) The AREACOR command is used to perform the Area Correction analysis on the specified datasource. Reference Peak Correction, Background Subtraction, or both, can be performed.

Command Format

```
AREACOR [<input datasource name>] [/Qualifier(s)]
```

Qualifiers

[/BKGND=<background file name>]

Specifies the name of the Background Spectrum file which will be used during Background Subtract area correction.

[/ETOL=<value>]

Specifies the fixed energy tolerance, in keV, used in determining a match between a peak in the datasource and a peak in the background file.

Example

```
AREACOR /BKGND=LAB_BKGD.CNF /ETOL=1.0
```

Perform a Background Subtract Area Correction on the datasource specified in Environment Variable JobInpSrc, using background spectrum file LAB_BKGD.CNF and matching peaks to a tolerance of 1.0 keV.

REFCOR^{RX}

(Applies to S501) The REFCOR command is used to normalize the areas of all peaks in the spectrum using a reference peak of a known count rate. The reference source can be either an electronic pulser or an external stationary source.

Command Format

```
REFCOR <input datasource name> /Qualifiers
```

The <input datasource name> is mandatory, the other qualifiers are optional.

Qualifiers

[/ENERGY=<energy>]

Specifies the energy of the reference peak (in keV). If missing, and the current datasource has no entry in the REFENG parameter, an error code will be returned indicating that the reference energy value is missing. If missing and the current datasource has a value in the REFENG parameter, the current value of REFENG will be used.

[/RATE=<reference rate>]

Specifies the reference count rate (in counts per second). If missing, and the current datasource has no entry in the REFRATE parameter, an error code will be returned indicating that the reference rate value is missing. If missing and the current datasource has a value in the REFRATE parameter, the current value of REFRATE will be used.

[/ERROR=<reference rate error>]

Specifies the reference count rate error (in counts per second). If missing, a zero error will be assumed.

[/DATE=<date>]

Specifies the reference rate measurement date. If the /DATE qualifier is not present and the current datasource has no value in the REFDATE parameter, the rate given with the /RATE qualifier (or already present in the datasource) is assumed to be valid for the start of acquisition date/time (no decay). If the /DATE qualifier is not present and the current datasource has a value in the REFDATE parameter, the value of the REFDATE will be used to establish the decay.

If the /DATE qualifier is present (or the date is already present in the current datasource) and the /HLFLIFE is not present, the following rules apply: If the current datasource has a value in the REFHLF parameter, it will be used. If the current datasource does not have a value in the REFHLF parameter, an error code will be returned indicating which parameter value is missing.

[/HLFLIFE=<half-life>]

Specifies the reference nuclide half-life in days unless used in conjunction with the /UNITS qualifier. If missing, the current value in the datasource will be used. If there is no current value in the datasource and the /DATE qualifier is used, an error code will be returned indicating which parameter value is missing.

[/UNITS=<half-life units>]

Specifies the units of the /HLFLIFE qualifier. If missing and the /HLFLIFE qualifier is used, the half-life is assumed to be given in days.

[/ETOL=<value>]

Specifies the energy tolerance, in keV, used in determining a match between a peak in the datasource and a peak in the background file. Note that /ETOL and /FTOL are mutually exclusive.

[/FTOL=<value>]

Specifies the variable energy tolerance, in multiples of expected FWHM, used to determine a match between a peak in the datasource and a peak in the background file. If not specified, the current value in the datasource will be used. Note that /FTOL and /ETOL are mutually exclusive.

[/BKGND=<background file name>]

Specifies the name of the Background Spectrum file which will be used during Background Subtract area correction.

Example

```
REFCOR NBSSTD.CNF /ENERGY=1505.5
```

Perform a reference peak correction on datasource NBSSTD.CNF using the reference peak at 1505.5 keV. The reference count rate and reference rate measurement date are fetched from the CAM parameters REFRATE and REFDATE, respectively, that are stored in NBSSTD.CNF.

ARALPHA^{RX}

(Applies to S509) The ARALPHA command is used to perform background corrections on the calculated peak areas in the input datasource's peak table. The ROIs in the datasource are used to define the regions in the specified background file, which are then subtracted from the datasource's peak areas.

Command Format

```
ARALPHA [<input datasource name>] [/Qualifiers]
```

Qualifiers

```
[/BKGND=<background file name>]
```

Specifies the name of the background file to be used for background subtract.

```
[/CRITLEVEL]
```

Specifies that Critical Level checking is to be performed.

Example

```
ARALPHA MY_EXPER.CNF /BKGND=LAB_BKGD.CNF /CRITLEVEL
```

Subtract the peaks in the file LAB_BKGD.CNF from datasource MY_EXPER.CNF. Perform critical level checking.

Reagent Correction

The Reagent Correction command subtracts tracer impurities from the other peaks in the specified datasource. If both a reagent blank correction and an environmental background subtract are to be performed, the environmental background subtract correction must always be performed first.

ARREAGNT^{RX}

(Applies to S509) The ARREAGNT command is used to perform the Reagent Correction phase of analysis on the specified datasource.

Command Format

ARREAGNT [<input datasource name>] [/Qualifiers]

Qualifiers

[/BKGND=<background file name>]

Specifies the name of the reagent blank background file to be used for reagent blank subtract.

[/TRACER=<tracer certificate file>]

Specifies the tracer certificate file to be used for the energy of the tracer peak. If not specified, the file name in the input datasource will be used.

[/ETOL=<value>]

Specifies the fixed energy tolerance, in keV, used to determine a match between a peak in the input datasource and the tracer energy. If not specified, the current value in the datasource will be used. Note that /ETOL and /FTOL are mutually exclusive.

[/FTOL=<value>]

Specifies the variable energy tolerance, in multiples of expected FWHM, used to determine a match between a peak in the input datasource and the tracer energy. If not specified, the current value in the datasource will be used. Note that /FTOL and /ETOL are mutually exclusive.

[/CRITLEVEL]

Specifies that Critical Level checking is to be performed.

Example

ARREAGNT MY_EXPER.CNF /BKGND=BKGD1.CNF

Subtract the tracer peak's activity in reagent blank file BKGD1.CNF from the peaks in datasource MY_EXPER.CNF without critical level checking, using the tolerance values in the datasource.

Efficiency Correction

The Efficiency Correction group of commands corrects the activity of peaks in the peak table.

EFFCOR^{RX}

(Applies to S501) The EFFCOR command is used to perform the Efficiency Correction phase of analysis on the specified datasource.

Command Format

EFFCOR [<input datasource name>] [/Qualifier(s)]

Qualifiers

[/EMP]

Use the empirical equation to perform the efficiency correction.

[/DUAL]

Use the dual polynomial equation(s) to perform the efficiency correction.

[/LINEAR]

Use the linear equation to perform the efficiency correction.

[/INTERP]

Uses a straight-line interpolation (in the ln-ln domain) between the two points on either side of the requested energy to establish the energy. If the energy is outside the range of calibration points, uses the nearest point.

Note that the above four qualifiers are mutually exclusive; the last one encountered on the command line will be honored.

[/SPLINE]

This method is currently not supported.

Examples

EFFCOR SPECT6.CNF

Perform an Efficiency Correction on the datasource SPECT6.CNF. Use the Efficiency Calibration equation type in the datasource.

EFFCOR

Perform an Efficiency Correction on the datasource specified in Environment Variable JobInpSrc. Use the Efficiency Calibration equation type in the datasource.

AEFCOR^{RX}

(Applies to S509) The AEFCOR command is used to perform the efficiency correction phase of analysis on the specified datasource.

Command Format

AEFCOR [<input datasource name>] [/Qualifiers]

Qualifiers

[/USETRACER]

Specifies that the tracer in the spectrum will be used to determine the effective efficiency. If not specified, simple (average) efficiency correction will be performed. If average efficiency is available from the calibration and the tracer peak information is provided, the chemical recovery factor will be calculated as well.

[/TRACER=<tracer certificate file>]

If /USETRACER is specified, this qualifier specifies the tracer certificate file to be used for the tracer info (energy, emission rate, date, and so forth). If not specified and the /USETRACER qualifier is specified, the file name in the input datasource will be used.

[/QTY=<tracer quantity (mL)>]

If /USETRACER is specified, this qualifier specifies quantity of the certified tracer added to the sample (in mL).

[/ETOL=<value>]

If /USETRACER is specified, this qualifier specifies the fixed energy tolerance, in keV, used to determine a match between a peak in the input datasource and the tracer energy. If not specified, the current value in the datasource will be used. Note that /ETOL and /FTOL are mutually exclusive.

[/FTOL=<value>]

If /USETRACER is specified, this qualifier specifies the variable energy tolerance, in multiples of expected FWHM, used to determine a match between a peak in the input datasource and the tracer energy. If not specified, the current value in the datasource will be used. Note that /FTOL and /ETOL are mutually exclusive.

Example

```
AEFCOR MY_EXPER.CNF /USETRACER /QTY=5 /ETOL=20  
/TRACER=MYTRACER.CTF
```

Perform an efficiency correction on MY_EXPER.CNF using a tracer peak to calculate the effective efficiency to be used. The tracer peak is specified as 5 mL of the tracer in the certificate file MYTRACER.CTF and matched to a peak in the spectrum with an energy tolerance of 20 keV.

Nuclide Identification

The Nuclide Identification commands attempt to correlate the peaks in the peak table with the nuclide(s) they might belong to.

TENTTIVE

(Applies to S500, S502, S504). The TENTTIVE command performs a “tentative” nuclide identification on the specified datasource. That datasource must contain an energy calibration. TENTTIVE looks at each of the peaks established by the Peak Area step of the analysis, and attempts to find a match in the specified nuclide library. It uses the specified tolerance to perform the search; all nuclides that satisfy that tolerance are recorded for reporting. The nearest match will be reported first, followed by any ‘more distant’ matches in order.

Command Format

```
TENTTIVE [<input datasource name>] [/Qualifier(s)]
```

Qualifiers

```
[/LIBRARY=<nuclide library file name>]
```

Specifies the name of the Nuclide Library file which will be used in the analysis.

```
[/ETOL=<value>]
```

Specifies the energy tolerance, in keV, used to determine a match between a peak in the datasource and a peak in the nuclide library file. Note that ETOL and FTOL are mutually exclusive.

```
[/FTOL=<value>]
```

Specifies the energy tolerance in number of FWHM used to determine a match between a peak in the datasource and a peak in the nuclide library. Note that ETOL and FTOL are mutually exclusive.

Example

```
TENTTIVE MY_EXP.CNF /LIBRARY=NBS_LIB.LIB /ETOL=4
```

Performs a tentative nuclide identification on the datasource MY_EXP.CNF. Matches peaks in the datasource with the nuclides defined in NBS_LIB.LIB. Uses an energy tolerance of 4 keV.

NID_STD^{RX}

(Applies to S501, S509) The NID_STD command performs a Nuclide Identification phase of analysis on the specified datasource. NID_STD compares the peak locate results with the information in the Nuclide Library and tentatively associates peaks with a nuclide. It then calculates nuclide activities based on each energy line which was associated with each nuclide.

Command Format

```
NID_STD [<input datasource name>] [/Qualifier(s)]
```

Qualifiers

```
[/LIBRARY=<nuclide library file name>]
```

Specifies the name of the Nuclide Library file which will be used in the analysis.

```
[/ETOL=<value>]
```

Specifies the energy tolerance, in keV, used to determine a match between a peak in the datasource and a peak in the nuclide library file. This is the same energy tolerance that is used by other analysis functions (such as efficiency calibration and correction).

```
[/CONFID=<confidence index threshold>]
```

Specifies the Confidence index threshold which will be used in the analysis.

```
[/MDA_TEST]
```

Specifies that the MDA test be performed.

```
[/NOACQDECAY]
```

Specifies that decay correction not be performed during spectrum acquisition.

Examples

```
NID_STD /ETOL=1.1
```

Perform a Nuclide Identification on the datasource specified in Environment Variable JobInpSrc. Match peaks to the Nuclide Library with a Tolerance of 1.1

keV. Use the Nuclide Library specified in the datasource. Use the Confidence index threshold specified in the datasource.

NID_STD SPECT2.CNF /LIBRARY=NBS_LIB.NLB

Perform a Nuclide Identification on the datasource SPECT2.CNF. Use Peak Locate energies for calculating Efficiencies. Match peaks to the Nuclide Library with the Tolerance specified in the datasource. Use the Nuclide Library NBS_LIB.CTF. Use the Confidence index threshold specified in the datasource.

NID_INTF^{RX}

(Applies to S501, S509) The NID_INTF command is used to perform the Nuclide Identification phase of analysis together with Interference Activity Correction with weighted mean on the specified datasource.

Command Format

NID_INTF [<input datasource name>] [/Qualifier(s)]

Qualifiers

[LIBRARY=<nuclide library file name>]

Specifies the name of the Nuclide Library file which will be used in the analysis.

[ETOL=<value>]

Specifies the energy tolerance, in keV, used to determine a match between a peak in the datasource and a peak in the nuclide library. Note that this is the same energy tolerance that is used by other analysis functions (such as efficiency calibration and correction).

[CONFID=<confidence index threshold>]

Specifies the Confidence index threshold which will be used in the analysis.

[MDA_TEST]

Specifies that the MDA test be performed.

[NOACQDECAY]

Specifies that decay correction not be performed during spectrum acquisition.

Examples

NID_INTF SPECT6.CNF

Perform an Interference Activity Correction on the datasource SPECT6.CNF. Use the MDA Confidence, Nuclide Library, energy tolerance in the datasource.

NID_INTF

Perform an Interference Activity Correction on the datasource specified in Environment Variable JobInpSrc. Use the MDA Confidence, Nuclide Library, energy tolerance in the datasource.

Detection Limits

The Detection Limits group of commands performs minimum detectable activity calculations on peaks in the peak table.

MDA^{RX}

(Applies to S501, S509) The MDA command performs a Minimum Detectable Activity (Currie) analysis on the specified datasource on which a Nuclide Identification (NID) analysis has already been performed. MDA calculates the Minimum Detectable Activity for all nuclides in the Nuclide Library. For nuclides that have been identified by the NID analysis, this command calculates the MDA using the calculated continuum under each peak that has been found and using the counts in the spectrum for each peak that has not been found. For nuclides that have not been identified by the NID analysis, the command calculates the Minimum Detectable Activity by summing up the counts in the spectrum over an interval centered on the centroid of each of the nuclides' lines.

Command Format

MDA [<input datasource name>] [/Qualifier(s)]

Qualifiers

[/CONFID=<confidence level>]

Specifies the % Confidence level for the MDA calculations.

[/WIDTH=<value>]

Specifies the ROI width value to use in the MDA calculations. If not specified, the value in the datasource will be used. If the datasource is not initialized, the default value in the VPWIDTH CAM parameter will be used. If <value> is less than 0.8, the routine will use 0.8.

Examples

MDA SPECT2.CNF

Perform a Minimum Detectable Activity analysis on the datasource SPECT2.CNF. Use the MDA Confidence in the datasource.

MDA

Perform a Minimum Detectable Activity analysis on the datasource specified in Environment Variable JobInpSrc. Use the MDA Confidence in the datasource.

MDA_KTA^{RX}

(Applies to S501, S509) The MDA_KTA command performs a Minimum Detectable Activity (German) analysis on the specified datasource on which a Nuclide Identification (NID) analysis has already been performed. MDA_KTA calculates the Minimum Detectable Activity for all nuclides in the Nuclide Library. For nuclides that have been identified by the NID analysis, this command calculates the MDA using the calculated continuum under each peak that has been found and using the counts in the spectrum for each peak that has not been found. For nuclides that have not been identified by the NID analysis, the command calculates the Minimum Detectable Activity by summing up the counts in the spectrum over an interval centered on the centroid of each of the nuclides' lines.

Command Format

MDA_KTA [<input datasource name>] [/Qualifier]

Qualifiers

[/CONFID=<confidence level>]

Specifies the % Confidence level for the MDA calculations.

[/WIDTH=<value>]

Specifies the ROI width value to use in the MDA_KTA calculations. If not specified, the value in the datasource will be used. If the datasource is not initialized, the default value in the VPWIDTH CAM parameter will be used. If <value> is less than 0.8, the routine will use 0.8.

Examples

MDA_KTA SPECT2.CNF

Perform a Minimum Detectable Activity analysis on the datasource SPECT2.CNF. Use the MDA Confidence in the datasource.

MDA_KTA

Perform a Minimum Detectable Activity analysis on the datasource specified in Environment Variable JobInpSrc. Use the MDA Confidence in the datasource.

Post NID Processing

The Post NID Processing command performs Action Level calculations.

ACTLVL^{RX}

(Applies to S501, S509) The Post NID Processing command performs Action Level calculations on identified and unidentified nuclides and totals the number of identified nuclides and unidentified peaks.

Command Format

ACTLVL [<input datasource name>] [/Qualifier(s)]

Qualifiers

[/ACTLVL1_ALARM1=<value>]

Specifies the Alarm 1 limit value for the sum of Action Level 1 values for all nuclides.

[/ACTLVL1_ALARM2=<value>]

Specifies the Alarm 2 limit value for the sum of Action Level 1 values for all nuclides.

[/ACTLVL2_ALARM1=<value>]

Specifies the Alarm 1 limit value for the sum of Action Level 2 values for all nuclides.

[/ACTLVL2_ALARM2=<value>]

Specifies the Alarm 2 limit value for the sum of Action Level 2 values for all nuclides.

[/USEMDA]

Specifies that the action level calculations will include MDA values for those nuclides that were not identified, and thus had no activity values when calculated. If this qualifier is not specified, the action level calculations will not include those nuclides for which no activity was found.

[/USEUPPER]

Specifies that the activity plus its error or the MDA value plus its error will be used as the upper bound value for the action level calculation. If not specified, action level calculations will simply use activity or MDA values.

Example

```
ACTLVL MY_EXPER.CNF  
/ACTLVL1_ALARM1=50 /ACTLVL1_ALARM2=75  
/ACTLVL2_ALARM1=50 /ACTLVL2_ALARM2=75
```

Perform action level calculations on datasource MY_EXPER.CNF using the specified alarm limits for action levels 1 and 2.

General Job Commands

The General Job Commands are the commands which support the Analysis Job Commands. These Commands are divided into six groups:

- Reporting
- Editing
- Options
- Hardware Control
- Application Interaction
- Miscellaneous Commands

In addition to these job commands, there is a group of Canberra-written job functions which are extensions to the REXX language environment which let you extract specific data, such as area or integral, from a datasource. These functions are covered in detail starting on page 103.

Reporting

The Reporting group of commands generates reports for the various analysis phases.

REPORT^{RX}

(Applies to S500, S502, S504) The REPORT command produces an analysis report for a specified datasource. The sections and format to be included in the report are defined by a report template file. Note that this command will always create a report disk file; its name will be the file name portion of the input datasource name with .RPT added as the extension. This file will be created at the pathname defined by the environment variable REPPFILES.

Command Format

```
REPORT [<input datasource name>] [/Qualifier(s)]
```

Qualifiers

[/NAME_n=<datasource name>]

Where n = 2 through 36, specifying up to 35 names in addition to <input datasource name>, which is datasource number 1, the default datasource.

[/TEMPLATE=<template file name>]

Specifies the name of the Report Template file which will control the contents and format of the report.

[/SECTION=<section name>]

Include the specified Template file Section in the report. If the section name is specified as a null string (“”), all template sections will be included.

[/EM=<report uncertainty factor>]

Specifies the error multiplier (Report Uncertainty Factor) to be used when generating the report.

[/NEWFILE]

Specifies that the report output should be started as a new report disk file (that is, not appended to an existing report disk file). If not specified, the output will be appended to an existing report disk file, if any.

[/FIRSTPG]

Specifies that the report output should start at page 1. If not specified, the report output will begin on the page number stored in the specified datasource.

[/NEWPG]

Specifies that the report output should start on a new page. If not specified, no new page is started.

[/SCREEN]

Specifies that the report output should also be directed to the screen.

[/PRINT]

Specifies that the report output should also be directed to the printer (that is, to the PRN: device).

[/OUTFILE=<report disk file name>]

Changes the standard output file path and name to a different file path and name.

[/UPDATE]

Specifies that the datasource be cleared of page and line parameters from a previous report.

Examples

```
REPORT SPECT3.CNF /TEMPLATE=ANALYSIS.TPL /SCREEN  
/NEWFILE /SECTION=HEADER
```

Generates a report disk file named SPECT3.RPT; using datasource SPECT3.CNF. The Template file ANALYSIS.TPL is used; the report output is also sent to the screen. The Report will include the HEADER section of the Template.

```
REPORT /TEMPLATE=ANALYSIS.TPL /PRINT /FIRSTPG /SECTION=NID_INTF
```

Appends to a report disk file named Filename.RPT; using the datasource specified in Environment Variable JobInpSrc. The Template file ANALYSIS.TPL is used; the report output is also sent to the printer. The Report will include the NID_INTF section of the Template. This report section will start on page 1.

Editing

The Editing command allows analysis parameters to be entered and edited.

PARS

(Applies to S500, S502, S504) The PARS command is used to allow modification of any CAM parameter (these include those used by the various analysis functions, hardware and a Acquisition and Analysis window). The qualifiers will be a CAM parameter name followed by = and a value for the parameter. Up to twenty (20) CAM parameters will be allowed on one PARS command line.

For absolute date/time CAM parameters, the value can be in either of two forms: “date/time” string in the format defined by the Windows country settings *or* a floating number. The number form is primarily intended to be used in conjunction with the GETPARAM function which returns absolute date/time CAM parameter values as floating point.

Command Format

```
PARS [<input datasource name>] [/Qualifier(s)]
```

Qualifiers

```
[/RECORD=<record number>]
```

Specifies the record number to be used when accessing all the specified CAM parameters. If not specified, record number 1 is assumed.

[/ENTRY=<tabular entry number>]

Specifies the tabular entry number to be used when accessing all the specified CAM parameters. If not specified, tabular entry number 1 is assumed.

[/CAM parameter name=<value>]

Specifies that the CAM parameter given by name should be set to value. Refer to the *CAM Files* chapter of the *Genie 2000 Customization Tools Manual* for a description of CAM parameter names.

Examples

```
PARS DEF_ANA.CNF /SIDENT="Lab 5 air filter" /SQUANT=5 /SUNITS="grams"
```

Set the Sample Identification parameter equal to the string Lab 5 air filter and Sample Quantity equal to 5 grams (Sample Units) in datasource DEF_ANA.CNF.

Note that a string containing spaces must be surrounded by quotes.

```
PARS /CURSORCH=58 /LMARKCH=123 /RMARKCH=134
```

The Acquisition and Analysis window for the datasource specified in Environment Variable JobInpSrc will display its Cursor at channel 58 and Markers at channels 123 and 134.

Options

The Options group of commands performs various optional analyses of the spectral data.

NORMAL^{RX}

(Applies to S500, S502, S504) The NORMAL command is used to shift the spectral data in the specified datasource using the energy calibration factors from a specified calibration datasource.

This command only uses up to a second order energy calibration equation; that is, a third order term will not be used.

Command Format

```
NORMAL <input datasource name> [/Qualifiers]
```

Qualifiers

```
/CAL=<calibration datasource name>
```

The name of the datasource that contains the calibration factors to be applied to the input datasource.

[/OUT=<output datasource name>]

The name of the output datasource. If specified, this file will be a copy of the input datasource except for the resultant shifted spectrum and calibration factors. If not specified, the shifted spectrum and calibration factors will be written back to the input datasource.

Example

```
NORMAL NBSSTD.CNF /CAL=NEWCAL.CAL /OUT=NEWNBS.CNF
```

Perform a spectrum normalization on datasource NBSSTD.CNF by using the calibration factors stored in the datasource NEWCAL.CAL. The results of this spectrum normalization will be stored in NEWNBS.CNF.

SMOOTH^{RX}

(Applies to S500, S502, S504) The SMOOTH command is used to perform an n-point Smooth on the specified datasource.

Command Format

```
SMOOTH [<input datasource name>] [/Qualifier]
```

Qualifier

[/POINTS=<number of points>]

Specifies the number of channels that will be used for each smoothing operation. 3-, 5-, 7-, 9-, 11- and 13-point smooth are supported. A 3-point smooth is the default.

Examples

```
SMOOTH SAMPLE37.CNF /POINTS=5
```

Perform a 5-point Smooth of datasource SAMPLE37.CNF from channel 1 to the last channel in the datasource.

```
SMOOTH SAMPLE37.CNF
```

Perform a 3-point (default) Smooth of datasource SAMPLE37.CNF from channel 1 to the last channel in the datasource.

STRIP^{RX}

(Applies to S500, S502, S504) The STRIP command is used to strip one datasource from another.

$$\text{Input}' = \text{Input} - \text{Output} \times K$$

where Input is the input datasource after the Strip, Input is the original input datasource, and Output is the datasource being stripped from the input datasource and K is the Strip Factor.

Command Format

STRIP [<input datasource name>] [<output datasource name>] [/Qualifier(s)]

Qualifiers

[/FACTOR=<strip factor>]

Specifies the multiplication factor that will be used for each channel of the output datasource before the subtract. /FACTOR and /TIME_RATIO are mutually exclusive. The default is 1.0 if neither /FACTOR nor /TIME_RATIO is present.

[/TIME_RATIO]

Specifies that the multiplication factor that will be used is the ratio of the Live times from the input and output datasources. /FACTOR and /TIME_RATIO are mutually exclusive.

Examples

STRIP SPECT3.CNF SPECT5.CNF /FACTOR=5.0

Perform a Strip of datasource SPECT5.CTF times a Strip Factor of 5.0 from datasource SPECT3.CNF.

STRIP /FACTOR=-2.5

Perform a Strip of the datasource specified in Environment Variable JobOutSrc times a Strip Factor of -2.5 from the datasource specified in Environment Variable JobInpSrc.

Hardware Control

The Hardware Control group of commands controls various functions and parameters related to MCA and front end hardware.

ADVANCE

(Applies to S500, S502, S504) The ADVANCE command is used to issue an advance signal to a Sample Changer associated with a hardware input. The WAIT command (see page 92) can be used to wait until the Sample Changer has finished its advance to the next sample. The input datasource must be a hardware input.

Command Format

ADVANCE [<input datasource name>]

Examples

```
ADVANCE DET:Lab_3
```

Send an Advance signal to the Sample Changer associated with the hardware input named Lab_3.

```
ADVANCE
```

Send an Advance signal to the Sample Changer associated with the hardware input whose datasource name is contained in the Environment Variable JobInpSrc.

STARTMCA

(Applies to S500, S502, S504) The STARTMCA command is used to Start acquisition on an MCA hardware input/counter. The input datasource must be a hardware input. In this command, the JobOpName Environment Variable is used to store the operator name in the "SCOLLNAME" CAM Sample Parameter. If the JobOpName environment variable does not exist, a null string will be stored as the operator name.

Command Format

```
STARTMCA [<input datasource name>] [/Qualifier(s)]
```

Qualifiers

```
[/NOCLEAR]
```

Instructs the command not to clear live time, real time and spectrum data before starting the acquisition. The default is Clear.

```
[/LIVEPRESET=<preset value>]
```

```
[/REALPRESET=<preset value>]
```

Specifies a Time Preset (live or real) of specified value in seconds. These two qualifiers are mutually exclusive; if more than one is used, only the last one encountered on the command line will be honored.

```
[/INTPRESET=<value>,<start ch.>,<stop ch.>]
```

```
[/AREAPRESET=<value>,<start ch.>,<stop ch.>]
```

```
[/CNTSPRESET=<value>,<start ch.>,<stop ch.>]
```

Specifies a Computational Preset (integral, area, or counts) and the preset value. Also specifies the starting channel and ending channel that will be used for the Computational Preset. These three qualifiers are mutually exclusive; if more than one is used, only the last one encountered on the command line will be honored.

```
[/SWEEPS=<value>]
```

Specifies a sweeps preset for an acquisition in MCS mode.

[/PHA-], [/PHA+], [/MCS-], [/MCS+]

Specifies which Acquisition Mode will be performed. Note that if there is more than one qualifier on the command line, only the last one will be honored.

[/MULTIPLE=<det1,...,det7>]

Specified to start acquisition on up to seven additional hardware inputs. Only the input name is specified, the node is taken from the input datasource name specification. DET: should not be specified.

[/EXT_START]

[/EXT_STOP]

Specifies that acquisition will start/stop based on an external event (as supported by the specified detector input).

Examples

```
STARTMCA \\CHEMLAB\DET:MY_EXPER /LIVEPRESET=1000 /PHA+
```

Start an acquisition cycle on hardware datasource MY_EXPER on node CHEMLAB. Acquire in PHA+ Mode with a Live time preset of 1000 seconds and no computational preset.

```
STARTMCA /REALPRESET=2000 /PHA+ /AREAPRESET=25000,130,147
/NOCLEAR
```

Without clearing time and data, start an acquisition cycle on the hardware datasource specified in Environment Variable JobInpSrc. Acquire in PHA+ Mode with a Real time preset of 2000 seconds and an Area computational preset of 25 000 counts between channels 130 and 147.

STOPMCA

(Applies to S500, S502, S504) The STOPMCA command is used to stop acquisition on an MCA hardware input/counter. The input datasource must be a hardware input.

Command Format

```
STOPMCA [<input datasource name>] [/Qualifiers]
```

Qualifiers

[/Abort]

Indicates that acquisition is to stop immediately for an MCS detector input (default is to stop at end-of-sweep).

Examples

STOPMCA DET:SAM

Stop the acquisition on hardware datasource SAM.

STOPMCA

Stop the acquisition on the hardware datasource specified in Environment Variable JobInpSrc.

EXTMCA

(Applies to S500, S502, S504) The EXTMCA command is used to initiate control to an external MCA device. This command will use the xx75 (Personal Computer Interface) protocol when communicating with the external MCAs.

A control (ASCII text) file is specified which contains one or more of the following commands, which are interpreted by the program and transmitted to the external MCA.

Control File Commands

START=<memory group>

Start collect for specified memory group.

STOP=<memory group>

Stop collect for specified memory group.

CLEAR=<memory group>

Clear data/time for specified memory group.

READ=<memory group>

Read data from specified memory group and store in the output datasource.

LOAD=<memory group>

Write data into specified memory group from the output datasource.

WAIT=<memory group>

Wait for collect to complete for specified memory group.

PRESET=<memory group>,<preset value>

Set preset for specified memory group.

INIT

Initialize MCA.

BEGIN

Start communications with MCA.

END

Stop communications with MCA.

Note that the <memory group> specification conforms to the Ps notation documented in the Personal Computer Interface Model 3575/3576 Operator's manual.

It is important to realize that both the external MCA and the PC's COM port used to communicate with the MCA must be set up in advance. Use the MODE command to set the appropriate COM port parameters to match those of the external MCA. For example:

```
MODE COM1:9600,E,7,1
```

Switch the external MCA to REMOTE to ensure proper operation.

Command Format

```
EXTMCA <control file name> [<output datasource name>] [/Qualifier]
```

Qualifier

```
[/TERM=<com port>]
```

Specifies the COM port on which communications to the external MCA will be performed. If not specified, the default is 1.

Example

```
EXTMCA S35_CTL.DAT SAMPLE.CNF /TERM=2
```

where the control file S35_CTL.DAT contains:

```
BEGIN
INIT
PRESET=1,1000
START=1
WAIT=1
READ=1
END
```


Control and communicate with an external MCA device on COM port 2 using SAMPLE.CNF as storage for data. The control file (S35_CTL.DAT) initializes the MCA, sets the live time preset to 1000 seconds for memory group 1 (FULL), starts collect, waits for completion, and then transfers the data into SAMPLE.CNF (along with time information).

HVCNTL

(Applies to S500, S502, S504) The HVCNTL command is used to initiate control functions on a computer-controlled high-voltage power supply (on, off, inhibit/overload reset). The input datasource must be a hardware input.

Command Format

HVCNTL [<input datasource name>] [/Qualifier(s)]

Qualifiers

[/ON]

Specifies that the high voltage should be turned on.

[/ARMED]

Specifies that the high voltage should be turned on at the start of acquisition for an Inspector MCA.

[/OFF]

Specifies that the high voltage should be turned off.

[/RESET]

Specifies that the inhibit latch, overload latch, or both, should be reset.

Examples

HVCNTL DET:SAM /ON

Turn on the high voltage for hardware datasource SAM.

HVCNTL /RESET

Reset the inhibit/overload latches on the high voltage for the hardware datasource specified in Environment Variable JobInpSrc.

HDWCHECK

(Applies to S500, S502, S504) The HDWCHECK command is used to perform a verification procedure for all the hardware devices associated with the specified datasource. The input datasource must be a hardware input. This command should ALWAYS be performed once before using a given hardware input.

Command Format

HDWCHECK [<input datasource name>]

Examples

HDWCHECK DET:SAM

Perform a verification of all hardware devices associated with the hardware datasource SAM.

HDWCHECK

Perform a verification of all hardware devices associated with the hardware datasource specified in Environment Variable JobInpSrc.

TAKEOVER

(Applies to S500, S502, S504) The TAKEOVER command is used to takeover ownership of an MCA device associated with the specified datasource. The input datasource must be a hardware input.

Command Format

TAKEOVER [<input datasource name>] [/Qualifier(s)]

Qualifiers

[/CONTINUE]

Specifies that ownership is assumed with the MCA device remaining in its current state. This is the default if the CONTROL qualifier is not specified.

[/CONTROL]

Specifies that ownership is assumed with the MCA device being reset to an initialized state (acquisition will be stopped if active).

Examples

TAKEOVER DET:SAM

Assume ownership of the MCA device associated with the hardware datasource SAM (leaving in current state).

TAKEOVER /CONTROL

Assume ownership of the MCA device associated with the hardware datasource specified in Environment Variable JobInpSrc and initialize.

PWRMGMT

(Applies to S500, S502, S504) The PWRMGMT command is used to control the power manager's operating mode. The input datasource must be a hardware input.

Command Format

PWRMGMT [<input datasource name>] [/Qualifier]

Qualifier

[/MODE=SAVE | FULL | AC]

Specifies the power management mode as Power Save, Full Power or AC Power.

Example

```
PWRMGMT DET:AIR_SAMP /MODE=SAVE
```

Set the power management mode of hardware datasource AIR_SAMP to power save.

PWRRESET

(Applies to S500, S502, S504) The PWRRESET command is used to initiate a power reset on computer-controlled front-end devices. The input datasource must be a hardware input.

Command Format

PWRRESET [<input datasource name>] [/Qualifier(s)]

Qualifiers

[/AMP]

Specifies that the power reset is directed to the amplifier.

[/ADC]

Specifies that the power reset is directed to the ADC.

[/HVPS]

Specifies that the power reset is directed to the high voltage power supply.

[/DSS]

Specifies that the power reset is directed to the digital stabilizer.

Example

```
PWRRESET \\CHEMLAB\DET:SAM /ADC
```

Perform a power reset on the ADC device for hardware datasource SAM on node CHEMLAB.

APZERO

(Applies to S500, S502, S504) The APZERO command initiates an auto pole/zero operation on a computer-controlled amplifier device. The input datasource must be a hardware input.

Command Format

```
APZERO [<input datasource name>]
```

Examples

```
APZERO DET:SAM
```

Perform an auto pole zero on the “programmable” amplifier associated with the hardware datasource SAM.

```
APZERO
```

Perform an auto pole zero on the “programmable” amplifier associated with the hardware datasource specified in Environment Variable JobInpSrc.

DIGSTAB

(Applies to S500, S502, S504) The DIGSTAB command is used to initiate control functions on a computer-controlled digital stabilizer (on, off, hold, overrange reset). The input datasource must be a hardware input.

Command Format

```
DIGSTAB [<input datasource name>] [/Qualifier(s)]
```

Qualifiers

```
[/GAIN=<OFF | ON | HOL>]
```

Specifies that gain stabilization will be disabled (OFF), enabled (ON) or held at current correction (HOLD).

[/ZERO=<OFF | ON | HOL>]

Specifies that zero stabilization will be disabled (OFF), enabled (ON) or held at current correction (HOLD).

[/RESET]

Specifies that the correction overrange status flag will be reset.

Example

```
DIGSTAB DET:INSP_1 /ZERO=OFF /GAIN=HOLD
```

Turn off zero stabilization and hold the current gain stabilization for hardware datasource INSP_1.

STARTMCS

(Applies to S500, S502, S504) The STARTMCS batch command initiates an independent, asynchronous process which performs MCS acquisition on up to eight datasources, each of which may have up to 10 Regions of Interest (ROIs). The data for each datasource is stored as a set of files named `detnameN.mcs` in the directory defined by the `VDM$STEMPS` environment variable, where `detname` is the detector file name and `N` has the value 0 through 9, allowing up to ten ROIs for that detector. For detector file names of six or less characters, an underscore will be added before the `N.mcs`. To avoid data loss, be sure that the first seven characters of each detector name are unique.

Command Format

```
STARTMCS [<input datasource name>] [/Qualifier(s)]
```

Required Qualifier

[/MCSTABLE=<filename>]

Specifies all the required MCS input parameters. The filename specified as part of this qualifier contains all the required MCS input parameters, which are divided into the values affecting the entire spectrum and those affecting each ROI. The name for each of the CAM parameters used to write these values to the table file is included in each of the following descriptions.

Values for the entire spectrum

Dwell Time: `CAM_L_MCSCDWELL`

Dwell time (in seconds; 0 means the dwell time is to be automatically calculated by dividing the preset time by the number of channels).

Dwell Source: `CAM_T_MCSCDSRC`

Can be one of: Live Time, Real Time, or an External signal, which are written to the MCS Table as REAL, LIVE and EXTN, respectively.

MCS Channels: CAM_L_MCSCCHANS

Number of MCS data channels.

MCS Cache Size: CAM_L_MCSCCACHESIZ

If this value is non-zero, the MCS data will not be written to the MCS file until this number of data points has been accumulated.

Values for each ROI

Region Description: CAM_T_MCSCRGNDESC

A 32-character text string describing the ROI. This string will be copied to the STITLE parameter when the MCS data file is created for the region.

Start Energy: CAM_F_MCSCRSTART

Start energy region limit in keV.

End Energy: CAM_F_MCSCREND

Stop energy region limit in keV.

Gross/Net Flag: CAM_L_MCSCNET

A flag that indicates whether the MCS data represents gross (integral) counts or net (area) counts (1=net).

Alarm Level Value: CAM_F_MCSCCPSLEV

A value that represents an alarm level (in cps) for a given region. A value of 0 disables any alarm checking for the region.

MCS Gain and Offset: CAM_F_MCSCGAIN and CAM_F_MCSCOFFSET

These two values are the coefficients for an $A+Bx$ equation which define the X scale for the MCS data. These parameters are copied into the appropriate ECAL

parameters (with the number of terms of 2, etc.) in the MCS datasource so they can be viewed in the Acquisition and Analysis application. This value will be seen in the application window only if Energy Units in the window's Display Preferences is set to "Other".

MCS Units: CAM_T_MCSCUNITS

Unit text string for the gain/offset calibration equation. This parameter is stored in the ECALUNITS parameter in the MCS datasource (with an ECALCNV value of 1.0). This value will be seen in the application window only if Energy Units in the window's Display Preferences is set to "Other".

Optional Qualifiers

[/PRESET=<value>]

Specifies the number of seconds of MCS data acquisition. If the dwell source is EXTN or if both dwell time and channels are non-zero, the preset value is not needed.

[/MULTIPLE=<det2,≤,det8>]

Specified to start acquisition on up to seven additional datasources. Only the input name is specified, the node is taken from the input datasource name specification. DET: should not be specified.

[/NOCLEAR]

Specifies that data in any pre-existing .MCS file will not be cleared before acquisition starts; newly acquired data will be added to existing data.

[/ONE_EVENT]

When multiple source detectors are specified, this qualifier specifies that all MCS inputs will be triggered to start their next dwell period from one event, such as the elapsed live time, the elapsed real time, or the external signal. Only the primary detector input datasource is monitored for the occurrence of this event.

Notes:

If any two of the three preset values (channels, preset and dwell-time) are non-zero, the third value will be calculated from the other two. If all three are non-zero, the channels value will be ignored and a value calculated from preset and dwell-time will be used. When the dwell source is EXTN, a non-zero value for channels is required; the preset and dwell-time values are not used.

An external event (used when the dwell source is EXTN) occurs when a value is written to the CAM_L_MCSEXTADV parameter of the detector.

There are two output parameter flags that indicate that some type of error condition has occurred. One flag (CAM_L_MCSCCPSALRM) indicates that an alarm level (in terms of cps) has been exceeded for one of the ROIs in a given input. Another flag (CAM_L_MCSCPTIMEX) indicates that the processing time needed to perform MCS on an input is greater than the specified dwell time. These flags are written to the MCS file.

Example

```
STARTMCS DET:MY_MCS /MCSTABLE=DEFINE.MCS /NOCLEAR
```

Using the MCS Table File DEFINE.MCS and without clearing existing data, start an MCS acquisition cycle with datasource MY_MCS.

STOPMCS

The STOPMCS command is used to terminate the SOFT_MCS process initiated by the STARTMCS command. This will be needed only if normal operation fails.

Command Format

```
STOPMCS
```

Example

```
STOPMCS
```

The SOFT_MCS process will be terminated.

WAITMCS

(Applies to S500, S502, S504) The WAITMCS command is used to make the job wait until the current datasource completes its MCS acquisition or until one of the warning conditions (alarm flags) occurs.

Command Format

```
WAITMCS [<input datasource name>] [/Qualifier]
```

Qualifier

```
[/TIMEOUT=<value>]
```

Specifies the time, in seconds, that the command will wait until the current datasource signals that it is ready. The batch procedure will resume when the signal is received or when the timeout period has elapsed, whichever comes first.

Example

```
WAITMCS DET:MY_MCS /TIMEOUT=120
```

Instructs the batch procedure to wait up to 120 seconds for datasource MY_MCS to be ready before going on to the next command in the batch procedure.

Application Interaction

The Application Interaction group of commands allows the Job Environment to interact (with restrictions) with an Acquisition and Analysis window.

PUTVIEW

(Applies to S500, S502, S504) The PUTVIEW command is used to specify an Acquisition and Analysis window with which the Job will interact. Note that only one application window can be opened in any given batch procedure.

Command Format

```
PUTVIEW [<input datasource name>] [/Qualifier(s)]
```

Qualifiers

```
[/DISP_ONLY]
```

Specifies that various hardware control menu items in the Acquisition and Analysis application will be disabled. The datasource is opened for read/write access.

```
[/READ_ONLY]
```

This qualifier instructs the Acquisition and Analysis application to open the specified datasource in read-only mode.

```
[/XY=<starting X value,starting Y value>]
```

Specifies the starting X and Y screen coordinates (in pixels) of the lower-left corner of the Acquisition and Analysis window. The default is <1,1>, which is the lower left corner of the screen. Note that if the XY values are preceded by a '-', the numbers are interpreted as a percentage of screen size (for example, -50,-50 puts the window in the center of the screen).

```
[/CXCX=<delta X value,delta Y value>]
```

Specifies the X and Y screen size (in pixels) of the Acquisition and Analysis window. The default is <150,100>. Note that if the CXCX values are preceded by a '-', the numbers are interpreted as a percentage of screen size (for example, -100,-50 creates a window which is the full width of the screen and half of the screen height).

[/NO_DATASRC]

Opens the Acquisition and Analysis application window without a datasource.

[/ALPHA]

This qualifier causes the Alpha Acquisition and Analysis application to be run instead of Gamma Acquisition and Analysis.

[/NOMENU]

This qualifier hides the application window's menu and toolbar.

Examples

```
PUTVIEW /READ_ONLY /XY=5,20 /CXCX=200,100
```

Put an Acquisition and Analysis window on the screen for the datasource specified in the Environment Variable JobInpSrc. The window will be positioned at screen coordinates 5, 20 and will have a size of 200, 100. The specified datasource will be opened in read-only mode.

```
PUTVIEW /XY=1,-50 /CXCX=-50,-50 /NO_DATASRC
```

Put an Acquisition and Analysis window on the screen with no open datasources. The window will be positioned in the upper left hand corner of the screen and take up exactly one quarter of the screen (whether run on an XGA, a VGA, or an SVGA monitor).

PVACCESS

(Applies to S500, S502, S504) The PVACCESS command is used to permit or deny write access to datasources from within an Acquisition and Analysis window. Issuing this command without the qualifier will permit write access.

Command Format

```
PVACCESS [<input datasource name>] [/Qualifier]
```

Qualifier

[/NOMENU]

This qualifier hides the application window's menu and toolbar, thus making the datasources read only.

PVOPEN

(Applies to S500, S502, S504) The PVOPEN command is used to open a datasource in an existing Acquisition and Analysis window (created by a PUTVIEW command, described on page 81). This datasource becomes the active datasource in the Acquisition and Analysis window. This command will not open the specified datasource if the Acquisition and Analysis window is in Show All mode. If necessary, PVSELECT (page 84) can be used to ensure that the Acquisition and Analysis is *not* in Show All mode.

Command Format

PVOPEN [<input datasource name>] [/Qualifier(s)]

Qualifiers

[/DISP_ONLY]

Specifies that various hardware control menu items in the Acquisition and Analysis application will be disabled. The datasource is opened for read/write access.

[/READ_ONLY]

This qualifier instructs the Acquisition and Analysis window to open the specified datasource in read-only mode.

[/EXPAND]

This qualifier instructs the Acquisition and Analysis application to open the specified datasource in expand mode. The absence of this qualifier will take the Acquisition and Analysis window out of expand mode if presently in expand.

[/AUTOCLOSE]

This qualifier instructs the Acquisition and Analysis application to close the specified datasource after acquisition is stopped. The specified datasource will be closed without checking to see if any changes have been saved.

Examples

PVOPEN NBSSTD.CNF /EXPAND

Open the NBSSTD datasource in the existing Acquisition and Analysis application window. The datasource will be opened for read/write and in expand mode.

PVOPEN /READ_ONLY

Open the datasource specified in the Environment Variable JobInpSrc. The specified datasource will be opened in read-only mode.

PVCLOSE

(Applies to S500, S502, S504) The PVCLOSE command is used to close a datasource in an existing Acquisition and Analysis application window (created by a PUTVIEW command, described on page 81). The specified datasource will be closed without checking to see if any changes have been saved. Note that this command will not close the specified datasource if the Acquisition and Analysis window is in Show All mode. If necessary, PVSELECT (page 84) can be used to ensure that the Acquisition and Analysis window is not in Show All mode.

Command format:

PVCLOSE [<input datasource name>]

Examples

PVCLOSE NBSSTD.CNF

Close the NBSSTD datasource in the existing Acquisition and Analysis window.

PVCLOSE

Close the datasource specified in the Environment Variable JobInpSrc.

PVSELECT

(Applies to S500, S502, S504) The PVSELECT command is used to select an already open datasource in an existing Acquisition and Analysis application window (created by a PUTVIEW command, described on page 81). This datasource becomes the active datasource in the Acquisition and Analysis window. This command will take the Acquisition and Analysis window out of Show All mode if presently in that mode.

Command format:

PVSELECT [<input datasource name>] [/Qualifier]

Qualifier

[/EXPAND]

This qualifier instructs the Acquisition and Analysis application to go into expand mode. The absence of this qualifier will take the Acquisition and Analysis window out of expand mode if presently in expand.

Examples

PVSELECT NBSSTD.CNF /EXPAND

Select the NBSSTD datasource in the existing Acquisition and Analysis application window as the active datasource. The datasource will be shown in expand mode.

PVSELECT

Select the datasource specified in the Environment Variable JobInpSrc as the active datasource.

PVSHOALL

(Applies to S500, S502, S504) The PVSHOALL command is used to show all open datasources in an existing Acquisition and Analysis application window (created by a PUTVIEW command, described on page 81).

Command format

PVSHOALL

Example

PVSHOALL

Show all of the datasource in the existing Acquisition and Analysis application window.

ENDVIEW

(Applies to S500, S502, S504) The ENDVIEW command is used to terminate the PM-based Acquisition and Analysis application window which was previously started with a PUTVIEW command.

Command Format

ENDVIEW

Example

ENDVIEW

Terminate the Acquisition and Analysis application window currently associated with this batch procedure.

Miscellaneous Commands

The Miscellaneous group of commands includes those commands that aren't in any other of the defined groups.

ANALYZE^{RX}

(Applies to S500, S502, S504) The ANALYZE command performs an analysis on the specified datasource. The analysis sequence to be performed will be defined through the specification of an analysis sequence file.

The analysis sequence file specifies the various analysis phases to be performed (such as peak locate and peak analysis), the algorithm associated with each phase (such as unidentified second difference for peak locate), and any parameters associated with the analysis phase (such as the significance threshold for a second difference peak locate). This sequence file may be created within the Gamma Analysis Application window, the Alpha Analysis Application window, or the Analysis Sequence Editor (using the ASE batch command).

Command Format

ANALYZE [<input datasource name>] [/Qualifier(s)]

Qualifiers

[/SEQ=<sequence file name>]

Specifies the analysis sequence file that will control the analysis. If not specified, the analysis sequence in the input datasource will be used.

[/REPORTONLY]

Specifies that only the report steps in the specified Analysis Sequence file will be executed by this command.

[/ALPHA]

This qualifier must be used for Alpha Spectroscopy sequences. It instructs ANALYZE to find the algorithms in the ALPHA.AEF analysis engine file rather than the GAMMA32.AEF analysis engine file.

[/NOREPORT]

This qualifier disables execution of report steps while processing an ASF file. This is useful when doing preliminary analysis, for count-to-MDA, for example.

Example

```
ANALYZE AIR_FLTR.CNF /SEQ=MY_ANALY.ASF
```

Analyze datasource AIR_FLTR.CNF with the Analysis Sequence and parameters setup file MY_ANALY.ASF.

FILECNVT

(Applies to S500, S502, S504) The FILECNVT command is used to convert spectral data and some header information from a foreign format file to a native CAM format file. For files that contain enough information, some analysis will be attempted automatically (that is, energy calibration coefficients based on channel energy pairs from the file). The input datasource must be a file.

Command Format

FILECNVT [<input datasource name>] [<output datasource name>] [/Qualifier(s)]

Qualifiers

[/EASYSPEC]

Specifies that the input datasource file to be converted is an EasySpec file type.

[/GAMMA]

Specifies that the input datasource file to be converted is a Gamma-AT file type.

[/IEC1455]

Specifies that the input datasource to be converted conforms to the IEC 1455 standard. (This is also referred to as an RMS file.)

[/INTERTECH]

Specifies that the input datasource file to be converted is an Intertechnique file type.

[/INTERWIN]

Specifies that the input datasource file to be converted is an InterWinner (V4.1 unpacked) file.

[/ND6S]

Specifies that the input datasource file to be converted is a Nuclear Data Family-of-Sixes file type.

[/NEWORTEC]

Specifies that the input datasource to be converted is a new-format (.SPC) Ortec file type.

[/NUCLEUS]

Specifies that the input datasource file to be converted is a Nucleus file type.

[/ORTEC]

Specifies that the input datasource file to be converted is an older-format (.chn) Ortec file type.

[/OXFORD]

Specifies that the input datasource to be converted is an Oxford file type.

[/PCMCA]

Specifies that the input datasource to be converted is an Aptec PCMCA (V6.31-7.02) file.

[/S100]

Specifies that the input datasource file to be converted is a System 100 file type.

[/SAMPO]

Specifies that the input datasource file to be converted is a MicroSampo or Sampo 90 file type.

[/SILENA]

Specifies that the input datasource to be converted is a Silena file type.

[/SPECTRAN]

Specifies that the input datasource file to be converted is a Spectran AT file type.

[/TOOLKIT]

Specifies that the input datasource file to be converted is a Series 35 Plus Toolkit ASCII file type.

Note that the above qualifiers are mutually exclusive; the last one encountered on the command line will be honored.

[/DETTYPE={GE, NAI, ALPHA}]

This qualifier can be used to override the detector type-code that is usually decided based on the number of data channels found in the file. That decision sets the type to “NaI” for channels ≤ 512 ; otherwise it’s set to “Ge”. The type-code is used for parameter initialization as described in the Initial Parameters section of Chapter 2, “CAM Files” in the *S500 Customization Tools Manual*.

If this qualifier is not present, the type-code is forced to “NaI” for the EasySpec file type and is read from the file for the Silena file type.

Examples

```
FILECNVT SD0002.SPC STD_CAM.CNF /SPECTRAN
```

Convert the Spectran spectrum data file SD0002.SPC to a native CAM datasource file STD_CAM.CNF.


```
FILECNVT SPECT1.MCA STD_CAM.CNF /S100
```

Convert the System 100 spectrum data file SPECT1.MCA to a native CAM datasource file STD_CAM.CNF.

MOVEDATA^{RX}

(Applies to S500, S502, S504) The MOVEDATA command is used to transfer raw spectral data and related information between two datasources. A datasource may be either a CAM file or an MCA hardware input. New default parameters (calibration, sample and efficiency parameters) can be inserted in the output datasource during the transfer. If no parameter qualifiers are specified, *all* parameters will be moved by default. This procedure is *not* recommended when the output datasource is hardware.

If the source file includes a detector type, the type is used to initialize certain parameters for the target file (see Initial Parameters section of Chapter 2, “CAM Files” in the *S500 Customization Tools Manual* for information on these parameters). If the detector type is not included in the source file and the spectrum is greater than 512 channels, the parameters are initialized for a Ge detector; if the spectrum is equal to or less than 512 channels, the parameters are initialized for an NaI detector.

Command Format

```
MOVEDATA [<input datasource name>] [<output datasource name>] [/Qualifier(s)]
```

Qualifiers

[/OVERWRITE]

Specifies that if the output datasource already exists, that the moved parameters should overwrite the ones already in the datasource. If not specified, output datasource is created; if it already exists an error will occur. This switch is required when the output datasource is hardware.

[/ACQ]

Specifies that the acquisition parameters should be moved. CAM classes moved include: ACQP. This switch is *not* recommended when the output datasource is hardware.

[/ECAL]

Specifies that the Energy, FWHM and Tailing calibration parameters should be moved. CAM classes moved include: CALRESULTS, SHAPECALRES. Note that the energy, shape, and tail equations are also moved (these are part of the ACQP CAM class).

[/EFFCAL]

Specifies that the efficiency calibration and detector parameters should be moved. CAM classes moved include: GEOM.

[/DATA]

Specifies that the raw spectrum data should be moved. CAM classes moved include: PHA.

[/PROCESSING]

Specifies that all analysis processing parameters should be moved. These parameters include peak locate sensitivity, start and end channels, and nuclide identification energy tolerance. CAM classes moved include: PROC, ANALCNTL.

[/PTCAL]

Specifies that the Peak to Total calibration parameters should be moved. CAM classes moved include: PTCALIB.

[/SAMPLE]

Specifies that the sample parameters should be moved. CAM classes moved include: SAMP.

[/ANALYSIS]

Specifies that the analysis results parameters should be moved. CAM classes moved include: PEAK, NUCL, NLINES, INTFER, INTRES.

[/DISPLAY]

Specifies that the display parameters should be moved. CAM classes moved include: DISP.

[/CERTIFICATE]

Specifies that the certificate parameters should be moved. CAM classes moved include: CERTIF.

[/CLASS=<CAM class name>]

Allows a named CAM class to be moved.

Examples

```
MOVEDATA DET:LAB_3 SPECT4.CNF
```

By default, move the spectral data and all parameters from hardware datasource DET:LAB_3 to a new datasource SPECT4.CNF.

```
MOVEDATA /SAMPLE /ECAL /OVERWRITE
```

Move the Sample and Energy Calibration parameters from datasource specified by the Environment Variable JobInpSrc to the datasource specified by the Environment Variable JobOutSrc and overwrite the parameters if it exists, otherwise create it.

DATAPLOT

(Applies to S500, S502, S504) The DATAPLOT command is used to plot Spectral Data. The plot is always sent directly to the printer or plotter (the PRN: device) with no PM-based window for interaction. The type of scaling and the VFS can be set. A pseudo expand can be done using the CHANNELS and VFS qualifiers.

Command Format

```
DATAPLOT [<input datasource name>] [/Qualifier(s)]
```

Qualifiers

```
[/CHANNELS=<start channel,end channel>]
```

Specifies the spectrum channel range to be plotted. Up to 1024 channels may be selected with this qualifier. If not specified, the entire spectrum will be plotted.

```
[/SCALE=LINEAR]
```

Specifies that the vertical scaling will be linear (this is the default if no other SCALE qualifier is specified).

```
[/SCALE=LOG]
```

Specifies that the vertical scaling will be log base 10.

```
[/VFS=AUTO]
```

Specifies that the vertical full scale will be automatically calculated to show the highest data count in the spectrum (this is the default if no other VFS qualifier is specified).

```
[/VFS=<manual VFS value>]
```

Specifies that the vertical full scale will be set to this value (rounded up to the next VFS increment) regardless of the highest data count in the spectrum.

[/COMPARE=<Compare spectrum datasource name>]

Specifies the datasource name of the compare spectrum. The compare spectral data will be plotted with an Auto VFS and a fixed vertical offset.

[/FULL]

Specifies that every data point be plotted.

Examples

```
DATAPLOT SPECT3.CNF /SCALE=LOG /CHANNELS=1100,1900
```

Plot the spectral data from channel 1100 to 1900 of the SPECT3 datasource. Use log base 10 vertical scaling and auto VFS.

```
DATAPLOT /COMPARE=SPECT4.CNF
```

Plot the spectral data of the datasource specified in the Environment Variable JobInpSrc compared with the spectral data of datasource SPECT4 to the default print device. Both spectra will be plotted with linear scaling and auto VFS at fixed offset.

Note: If you want to plot an expand region with a VFS offset as well as a VFS maximum or a compare spectrum with a set VFS (instead of auto) or a set offset (instead of fixed) use the PUTVIEW and ENDVIEW commands (on pages 81 and 85) to set the conditions you want to plot. Then select **Data Plot** (described in “Data Plot” of the *Gamma Acquisition and Analysis* chapter in Volume 1, *Genie-2000 Operations*) from the File Menu to do the plot instead of the DATAPLOT command.

WAIT

(Applies to S500, S502, S504) The WAIT command is used for commanding the Job to wait for a given condition. The conditions to be supported include ADC timeout (acquisition done, preset reached), sample changer ready, elapsed time and time of day. Note that, with the exception of Timeout, if there is more than one qualifier on the command line, only the last one will be honored.

Command Format

```
WAIT [<input datasource name>] [/Qualifier(s)]
```

Qualifiers

[/ACQ]

Specifies that the batch procedure wait for acquisition to complete (that is, Done or Stopped) on the specified input datasource before going on to the next Command. The specified input datasource must be a hardware input.

[/SAMPLE_CHANGER]

Specifies that the batch procedure wait for Sample Changer ready on the specified input datasource before going on to the next Command. The specified input datasource must be a hardware input.

[/TIMEOUT=<time in seconds>]

Specifies the maximum time to Wait, in seconds, for another qualifier to be satisfied (see the first example, below).

[/ELAPSED=<time in seconds>]

Specifies the time in seconds that the batch job should Wait before going on to the next Command.

[/READONLY]

Specifies that the input datasource be opened in read only mode while doing the wait.

Examples

```
WAIT DET:MY_EXPER /SAMPLE_CHANGER /TIMEOUT=120
```

Instructs the batch procedure to wait until the Sample Changer associated with hardware datasource MY_EXPER becomes ready or a timeout of 120 seconds, whichever happens first, before going on to the next command in the batch procedure.

```
WAIT /ELAPSED=60
```

Instructs the batch procedure to wait until 60 seconds has expired before going on to the next command in the batch procedure.

LOADMID

(Applies to S500, S502, S504) The LOADMID command is used to load an MID configuration file into the Runtime Database. The datasource(s) defined by the MID file will then be available for opening and use. This is analogous to loading an MID file in the MCA Input Definition Editor (see “Using MCA Definition Tables” in the *MCA Input Definition Editor* chapter of Volume 1, *Genie-2000 Operations*) with the exception that duplicate definitions can automatically be overridden by use of the OVERRIDE qualifier.

The MID filename must be the base filename of an MID file. The command adds a .MID extension and uses the environment variable MCA\$DEFINES to find the file.

Command format:

```
LOADMID <MID filename> [/Qualifier(s)]
```

Qualifiers

[/NODE=<remote node name>]

Specifies the name of the remote node to connect to. The MID file will be loaded into the Runtime Database at this node.

[/OVERRIDE]

Specifies that if a MID file currently loaded in the Runtime Database has any input name(s) or hardware address(es) (for example, S100 board 1 or AIM address 01FA) that are duplicated in the new MID file being loaded, the currently loaded MID file will be unloaded. This will be repeated until there are no duplication conflicts. The new MID file will then be loaded into the Runtime Database, overriding the previous definition.

Examples

```
LOADMID S100_1-3 /NODE=COUNT_RM /OVERRIDE
```

Load MID file S100_1-3 into the Runtime Database at remote node COUNT_RM. If there are any duplicate input names or hardware addresses, Unload the existing MID and Load this one in its place.

```
LOADMID MY-AIM
```

Load MID file MY-AIM into the Runtime Database at the Local node. If there are any duplicate input names or hardware addresses, report the appropriate error message.

ULOADMID

(Applies to S500, S502, S504) The ULOADMID command is used to unload an MID configuration file from the Runtime Database. The datasource(s) defined by the MID file will then be unavailable for opening and use. This is analogous to unloading an MID file in the MCA Input Definition Editor (see “Using MCA Definition Tables” in the *MCA Input Definition Editor* chapter of Volume 1, *Genie-2000 Operations*).

The MID filename must be the base filename of an MID file.

Command format:

```
ULOADMID <MID filename> [/Qualifier]
```

Qualifier

[/NODE=<remote node name>]

Specifies the name of the remote node to connect to. The MID file will be unloaded from the Runtime Database at this node.

Examples

```
ULOADMID S100_1-3 /NODE=COUNT_RM
```

Unload MID file S100_1-3 from the Runtime Database at remote node COUNT_RM.

```
ULOADMID MY_AIM
```

Unload MID file MY_AIM from the Runtime Database at the Local node.

REXXFCTS

(Applies to S500, S502, S504) The REXXFCTS command is used for registering a set of functions (either Canberra-supplied as described in “Job Function Descriptions” on page 103, or user-supplied) which extend the REXX language environment for spectroscopy applications use. The REXXFCTS command uses the Acquisition and Analysis software environment variable OS2FCTSFILE to determine the location of an ASCII file that describes what functions are to be registered with the REXX environment. The format of this file is as follows:

```
<function-name>!<DLL-name>!<entry-point-name>
```

where:

function-name: name of function as recognized by the REXX interpreter.

DLL-name: name of DLL file containing the specified REXX function.

entry-point-name: name of entry point within the specified DLL file which is called for the specified REXX function.

An example of this file is as follows:

```
GETPARAM!REXXFCTS!GETPARAM  
GETDATA!REXXFCTS!GETDATA
```

Command Format

```
REXXFCTS
```

Example

```
REXXFCTS
```

Registers functions described in the Genie-2000 Job Functions File specified by the environment variable OS2FCTSFILE with the REXX language environment.

ASE

(Applies to S500, S502, S504) The ASE command displays an Analysis Sequence Editor window for creating or changing analysis sequences. If no datasource is specified, a new untitled sequence file will be created. Several of this command's qualifiers initialize a new file with appropriate detector parameters.² These parameters are documented in Initial Parameters section of Chapter 2, "CAM Files" in the *S500 Customization Tools Manual*.

Command Format

ASE [<datasource name>] [/Qualifier(s)]

Qualifiers

[/AEF_ALPHA]

Specifies that the editor use alpha spectroscopy analysis algorithms. If this qualifier is not specified, the editor will use gamma spectroscopy algorithms.

[/ALPHA]

Initializes new files using Alpha detector parameter settings; this is the default if AEF_ALPHA is specified. Mutually exclusive with /GE and /NAI.

[/GE]

Initializes new files using Ge detector parameter settings; this is the default if AEF_ALPHA is *not* specified. Mutually exclusive with /ALPHA and /NAI.

[/NAI]

Initializes new files using NaI detector parameter settings. Mutually with /ALPHA and /GE.

[/REVIEWONLY]

Specifies that the editor can be used to review, but not change, the specified datasource.

2. A new file may be one created with the editor's File | New menu command or may be one created when the editor is opened without specifying a datasource.

Examples

ASE /AEF_ALPHA

Open the Analysis Sequence Editor and create a new untitled analysis sequence file using alpha spectroscopy analysis algorithms.

ASE AIR_FLTR.ASF /REVIEWONLY

Open the Analysis Sequence Editor and load the AIR_FLTR.ASF analysis sequence file for review only.

BEGINNING OF QA FILE INSERT

Application Option Commands

This section covers those batch commands which are specific to the Genie-2000 application options.

QA Batch Procedure Commands

In addition to the interactive QA Editor application, all of the day-to-day QA functions can be performed through a series of These commands assume that you have a QA file with parameter definitions already defined for monitoring. There are commands for adding measurement results and for reporting and plotting the results, but not for editing the definitions or results values.

QAXFER

(Applies to S505) The QAXFER command will transfer the parameter values from a single CAM datasource to a single QA file. In order for analysis-related parameters to be transferred, the appropriate analysis algorithms must have been previously executed on the CAM datasource.

Command Format

QAXFER [<Input datasource>] [QA filename] [/Qualifier]

Qualifier

[/HELP]

Displays Help text on the screen about this command.

Example

```
QAXFER MY_EXPER.CNF QA_DET01.QAF
```

Transfer the values from the analyzed CAM File MY_EXPER.CNF into the QA file for detector #1, QA_DET01.QAF.

QAMANUAL

(Applies to S505) The QAMANUAL command will allow a user to enter all of the parameters defined in the specified QA file as manually-entered. The user is prompted for each parameter, one at a time.

Command Format

```
QAMANUAL [QA filename] [/Qualifier]
```

Qualifier

```
[/HELP]
```

Displays Help text on the screen about this command.

Example

```
QAMANUAL QA_DET01.QAF
```

Prompt for the values for each parameter with a Parameter Type of MANUAL defined in the detector #1 QA file.

QAANALYZ

(Applies to S505) The QAANALYZ command performs out-of-range tests for one or more parameters in a QA file and generates a report (to the screen and/or printer). If the PARAMETER qualifier is not specified, then all the parameters within the specified QA file are analyzed and reported on. If none of the test qualifiers are specified, then the test(s) which are set up as the defaults for the parameter are performed.

Command Format

```
QAANALYZ [QA filename] [/Qualifiers]
```

Qualifiers

```
[/REPORT = <list>]
```

Specifies which reports are to be generated. The options are

LAST

Reports on the last measurement for all parameters

Application Option Commands

FULL

Reports on a series of measurements for a specified parameter

ALL

Generate both reports

[/PARAMETER = <parameter description>]

For a Full Report, specifies the parameter. If not specified, all parameters in the QA file are analyzed and reported on.

[/START = <start date>]

For a Full Report, specifies the start date for the results to be reported. If not specified, the report starts with first results record.

[/END = <end date>]

For a Full Report, specifies the end date for the results to be reported. If not specified, the report ends with last results record.

[/BOUNDARY]

Specifies the Boundary Test to be performed.

[/SAMPLE]

Specifies the Sample Driven N-Sigma Test to be performed.

[/USER]

Specifies the User Driven N-Sigma Test to be performed.

[/BIAS]

Specifies the Bias Test to be performed.

[/SCREEN]

Specifies that the report output should also be directed to the screen.

[/TREND]

Specifies the Trend Test to be performed.

[/PRINT]

Specifies that the report output should also be directed to the printer.

[/HELP]

Displays Help text on the screen about this command.

Examples

```
QAANALYZ QA_DET01.QAF /REPORT=LAST /BOUNDARY /SCREEN
```

Generates a Last Report on the QA file for detector #1 (QA_DET01.QAF). The boundary test is performed on the last measurement for each defined parameter and the output is sent to QA_DET01.RPT and to the screen.

```
QAANALYZ QA_DET01.QAF /REPORT=FULL
/PARAMETER="Pk Centroid K-40" /PRINT
```

Generates a Full Report on the QA file for detector #1 (QA_DET01.QAF). The test(s) set up as the defaults for the "Pk Centroid K-40" parameter are performed on all measurements in the file for that parameter. The output is sent to QA_DET01.RPT and to the default printer.

QAPLOT

(Applies to S505) The QAPLOT command will generate a control chart for a single monitored parameter. A control chart is a plot of parameter values as a function of time. The control chart goes to the screen by default; it may be directed to the printer via the PRINT qualifier.

Command Format

```
QAPLOT [QA filename] [/Qualifiers]
```

Qualifiers

[/PARAMETER = <parameter description>]

Specifies the parameter to plot. This qualifier is required.

[/PARAMETER2_X]

Specifies the x-axis parameter to plot for parameter versus parameter plotting. Required only for parameter versus parameter plotting.

[/PARAMETER2_Y]

Specifies the second y-axis parameter to plot for dual parameter versus time plotting. Required only for dual parameter versus time plotting.

Application Option Commands

[/START = <start date>]

Specifies the start date for the results to be plotted. If not specified, the plot starts with first results record.

[/END = <end date>]

Specifies the end date for the results to be plotted. If not specified, the plot ends with last results record.

[/BOUNDARY]

[/SAMPLE]

[/USER]

[/BIAS]

Specifies the Boundary Test, Sample Driven N-Sigma Test, User Driven N-Sigma Test or Bias Test to be plotted. Note that these four qualifiers are mutually exclusive; the last one encountered on the command line will be honored.

[/CONNECT]

Specifies that the data points be connected.

[/REJECTS]

Specifies that rejected data points be included in the plot.

[/PRINT]

Specifies that the plot output should be directed to the printer.

[/HELP]

Displays Help text on the screen about this command.

Examples

```
QAPLOT QA_DET01.QAF /PARAMETER="Pk Centroid K-40"  
/BOUNDARY /REJECTS
```

Generates a plot of all measurements for the "Pk Centroid K-40" parameter in the detector #1 QA file. Grid lines indicate the lower and upper bounds and any rejected measurements are also plotted.

```
QAPLOT QA_DET01.QAF /PARAMETER="Pk Centroid K-40"  
/SAMPLE /PRINT /START="1/1/92 000" /END="2/1/92 000"
```

Generates a plot of January 1992 measurements for the "Pk Centroid K-40" parameter in the detector #1 QA file. Grid lines indicate the Investigate and Action out-of-range limits for the Sample Driven N-Sigma test. The plot is sent only to the printer.

IPFIT

(Applies to S506) The IPFIT command is used to display the IPF window with the selected datasource. If filter qualifiers are specified, the first peak region meeting the filter settings will be shown first. Once the IPF window is displayed, the filter settings can be changed.

Command Format

IPFIT <input datasource name>³ [/Qualifier(s)]

The input datasource name is required.

Qualifiers

If the datasource includes filter settings, they will be used unless qualifiers are specified on the command line. If the datasource does *not* include filter settings *and* no qualifiers are specified on the command line, the /NOFILTERS qualifier will automatically be used.

[/SCALE=<scale>]

Specifies which plot scale is to be used. Valid qualifier values are LINEAR, SQRT, and LOG.

[/REVIEWONLY]

Specifies that IPF can be used to look at, but not change, all peak regions that satisfy the filter criteria.

[/NOFILTERS]

Specifies that all peak regions in the datasource are to be presented for examination. This is the default filter setting.

[/ENERGY=<energy>]

Specifies which energy to look for. Any positive number between the spectrum start and stop energies is a valid entry.

[/NUCLIDE=<nuclide name>]

Specifies the nuclide whose peaks are to be looked for. Since the search library cannot be specified with a qualifier, IPF will search the library included in the

3. The 'JobInpSrc' environment variable may be used in place of the <input datasource name>

General Job Functions

input datasource. If the datasource does not include a library, IPF will use the first nuclide library file found in Genie-2000's nuclide library directory.

[/CHISQ=<value>]

Specifies the chi-square limit. All peak regions with a chi-squared value higher than the one given will be presented for examination. The value must be a positive number.

[/FWRATIO=<value>]

Specifies the FWHM ratio limit. All peak regions whose measured/expected FWHM ratio is larger than the value given will be presented for examination. The value must be a positive number.

[/MULTIP]

Specifies that only multiplet peak regions are to be selected. If this qualifier is omitted, all peak regions that satisfy the other qualifiers, multiplets or not, will be presented for examination.

General Job Functions

Job Functions are Canberra-supplied extensions to the REXX language environment which provide an extended set of functions necessary to perform the actual spectroscopy related operations; the key difference is that these functions operate only in the REXX language environment.

This section describes in detail the operation of the various job functions which are part of the Job Environment. Each description follows a basic format:

1. Function description
2. Function format, with parameters
3. Function parameter descriptions
4. Examples of the command

Each job function returns its exit status through the REXX variable `JOBERROR`; a value of zero will be returned if the job function was successfully completed, a non-zero return signifies an error occurred. It will be up to you to recognize the error and decide what to do with it (`IF JOBERROR > 0`). Refer to Appendix 1, *Error Codes and Messages*, for more information on error codes.

REXX Function Equivalents

A set of “REXX function equivalents” exists for many of the Genie-2000 job commands. The following table lists these functions under the name of the DLL that contains them. The naming convention for these functions is the name of the batch command prefixed with RX (for example, the REXX function equivalent for the PEAK_DIF job command is RXPEAK_DIF).

Setting up the Environment

Before using any Genie-2000 REX functions (or accessing any Genie-2000 environment variables), a REXX procedure must call an external procedure supplied with S561 package named G2KENVMT. This procedure will set up the Genie-2000 environment variables based on settings found in the registry. The procedure is called as follows:

```
CALL G2KENVMT
```

Registering the Functions

To use these functions, they must first be registered as shown in the following examples:

```
CALL RXFUNCADD 'RXECAL', 'RX_EQUIV', 'RXECAL'
```

```
CALL RXFUNCADD 'RXAECAL', 'RX_ALPHA', 'RXAECAL'
```

The calling convention for these functions will be in one of two forms (depending on whether the job command equivalent specifies an output datasource on the command line):

```
CALL REXXFunction InputDS [,Arguments] [,OpName]
```

```
CALL REXXFunction InputDS, OutputDS [,Arguments] [,OpName]
```

where:

InputDS	Input datasource name; can be a DSC value returned by one of the Make...Connection calls.
OutputDS	Output datasource name; can be a DSC value returned by one of the Make...Connection calls.
Arguments	REXX stem variable whose members are the names of the existing batch command line qualifiers (note that no support is provided for interactive qualifiers like HELP, INIT_M, INIT_C, etc.).
OpName	

Operator name (as typically defined for batch commands via the JobOpName environment variable). This argument will be treated as the 3rd (4th) argument for the function calls thus requiring the appropriate no. of preceding commas if used.

These functions will return error codes via the JOBERROR and ERRCAUSE REXX variables (similar to the GBT functions).

REXX Equivalents for Gamma Spectroscopy

The Gamma spectroscopy function REXX equivalents, included in the RX_EQUIV.DLL file supplied with Models S500, S502 and S504, are:

RXECAL	RXRECAL	RXEFFCAL
RXEFFCOR	RXANALYZE	RXSTRIP
RXSMOOTH	RXNORMAL	RXMOVEDATA
RXREPORT	RXAREACOR	RXREFCOR
RXAREA_LIB	RXAREA_NL1	RXPEAK_DIF
RXPEAK_ROI	RXPEAK_SLB	RXPEAK_LIB
RXACTLVL	RXMDA	RXMDA_KTA
RXNID_INTF	RXNID_STD	

REXX Equivalents for Alpha Spectroscopy

The Alpha spectroscopy function REXX equivalents, included in the RX_ALPHA.DLL file supplied with Model S509 are:

RXAECAL	RXARECAL	RXAEFFCAL
RXAEFCOR	RXARREAGNT	RXARALPHA

MakeVDMConnection

This function creates a connection to the VDM, opens the specified datasource and leaves it open, allowing it to be used multiple times by subsequent REXX functions. If successful, a DSC is generated as a return value and can be used as a datasource argument in subsequent REXX function calls. If the REXX variable JOBERROR is not zero after the call, an error has occurred.

Function Format

DSC = MakeVDMConnection(dsname[, options])

Calling Arguments

dsname

Fully specified pathname of the datasource to be opened; can be either a file or detector input.

options

An optional stem variable with the following members:

READONLY	Specifies the readonly attribute when opening the datasource.
SYSWRITE	Specifies the System Write attribute when opening the datasource.
CREATEIF	Specifies that a file should be created if it does not already exist; the value specified indicates the size (rounded up to the nearest multiple of 256) of the PHA data block.

Example

```
Args.Readonly=1 DSC=MakeVDMConnection (“\LAB_1\DET:ASM01”, “Args”)
```

Opens detector ASM01 on node LAB_1 in read only mode, leaving the connection open.

MakeDirectConnection

This function opens the specified datasource, bypassing the VDM and going directly to the PCAM subsystem. It leaves the datasource open, allowing it to be used multiple times by subsequent REXX functions. If successful, a DSC is generated as a return value and can be used as a datasource argument in subsequent REXX function calls. If the REXX variable JOBERROR is not zero after the call, an error has occurred.

Function Format

DSC = MakeDirectConnection(dsname[, options])

Calling Arguments

dsname

Fully specified pathname of the datasource to be opened; must be file input.

options

An optional stem variable with the following members:

READONLY	Specifies the readonly attribute when opening the datasource.
SYSWRITE	Specifies the System Write attribute when opening the datasource.
CREATEIF	Specifies that a file should be created if it does not already exist; the value specified indicates the size (rounded up to the nearest multiple of 256) of the PHA data block.

Example

Stem.CREATEIF=2048 "DSC=MakeDirectConnect
(c:\genie2k\camfiles\nextfile.cnf", "Stem")

Creates and opens a 2K datasource named
C:\GENIE2K\CAMFILES\NEXTFILE.CNF, bypassing the VDM and going
directly to the PCAM subsystem. The datasource file is left open.

CloseConnection

This function closes (and optionally saves) the specified datasource. Every call to MakeVDMConnection or MakeDirectConnection must have a corresponding CloseConnection call in order to properly clean up open datasources / VDM datasource connections.

Function Format

Result = CloseConnection(DSC[, options])

Calling Arguments

DSC

The DSC specification of the datasource to be closed.

options

An optional stem variable with the following members:

NOFLUSH	Specifies that the datasource is <i>not</i> to be flushed (i.e. changes saved to disk).
NOCLOSE	Specifies that the datasource is <i>not</i> to be closed.

If this optional stem variable is not specified, then the specified datasource is automatically saved and closed.

Example

```
Flags.NOFLUSH=1 CALL CloseConnectionDSC,"Flags"
```

Closes a datasource previously opened with a Make...Connect call, discarding any changes.

GETAREA

The GETAREA function is used to fetch the net area of a specified data region and store it in a REXX variable named AREA. Net area is calculated only when the region defined by <start channel> and <stop channel> meets the following criteria:

Number of spectral data channels	Size of ROI
256	≤ 256
512 - 2048	≤ 512
> 2048	≤ 1024

Function Format

```
CALL GETAREA <datasource name>,<start channel>,<stop channel>
```

Parameter Descriptions

<datasource name>

Specifies the name of the datasource from which the raw spectral data will be fetched.

<start channel>

Specifies the starting channel number to be used to define the start of the region.

<stop channel>

Specifies the stop channel number to be used to define the end of the region.

Example

```
CALL GETAREA 'DET:S100',100,200
```

Instructs the batch procedure to fetch the net area from channels 100-200 from the hardware datasource S100. The value will be returned in the REXX variable AREA.

GETPARAM

The GETPARAM function is used to fetch a parameter from a datasource and store it as a REXX variable (the name of the REXX variable is the name of the CAM parameter being fetched).

Function Format

```
CALL GETPARAM <datasource name>,<record number>,  
<entry number>,<CAM parameter 1>,...,<CAM parameter 10>
```

Parameter Descriptions

<datasource name>

Specifies the name of the datasource from which the requested CAM parameter(s) will be fetched.

<record number>

Specifies the record number to be used when fetching the requested CAM parameter(s).

<entry number>

Specifies the tabular entry to be used when fetching the requested CAM parameter(s).

<CAM parameter name>

Specifies the name(s) of up to 10 CAM parameters to be fetched with this GETPARAM call.

Examples

```
CALL GETPARAM 'DET:S100',1,1,'ELIVE'
```

Instructs the REXX function to fetch elapsed live time (CAM parameter ELIVE) from the hardware datasource S100. The value will be returned in the REXX variable ELIVE.

```
CALL GETPARAM 'DET:S100',3,1,'RGNSTART', 'RGNEND'
```

Instructs the batch procedure to fetch the start and end channels of region of interest number three, single parameter.

GETDATA

The GETDATA function is used to fetch raw spectral data from a datasource and store it as a REXX variable. By default, the data is stored in REXX variable DT1 (start channel's data) through DTnnn (stop channel's data). Up to 100 spectral data channels can be fetched with one GETDATA call.

With the optional CH parameter, GETDATA can fetch up to 4096 spectral data channels from the specified datasource and store it in REXX compound variable CH.1 (start channel's data) through CH.nnnn (stop channel's data).

Function Format

```
CALL GETDATA <datasource name>,<start channel>, <stop channel>[,CH]
```

Parameter Descriptions

<datasource name>

Specifies the name of the datasource from which the raw spectral data will be fetched.

<start channel>

Specifies the starting channel number to be used when fetching the raw spectral data.

<stop channel>

Specifies the ending channel number to be used when fetching the raw spectral data.

[CH]

Specifies that raw spectral data be returned in REXX compound variable CH.1 up to CH.4096.

Examples

```
CALL GETDATA 'DET:S100',401,500
```

Instructs the batch procedure to fetch raw spectral data from channels 401 - 500 from the hardware datasource S100. The values will be returned in the REXX variables DT1 through DT100.

```
CALL GETDATA 'DET:S100',1025,4096,'CH'
```

Instructs the batch procedure to fetch raw spectral data from channels 1025 to 4096 from the hardware datasource S100. The data will be returned in the REXX variables CH.1 through CH.3072.

PUTDATA

The PUTDATA function is used to fetch spectral data from a REXX variable and store it in a datasource. By default, the data is fetched from REXX variable DT1 (start channel's data) through DTnnn (stop channel's data). With the default DTnnn variable up to 100 spectral data channels can be stored with one PUTDATA call.

With the optional CH parameter, one PUTDATA call can fetch up to 4096 spectral data channels from REXX compound variable CH.1 (start channel's data) through CH.nnnn (stop channel's data) and store it in the specified datasource.

Function Format

```
CALL PUTDATA <datasource name>,<start channel>, <stop channel>[,CH]
```

Parameter Descriptions

<datasource name>

Specifies the name of the datasource to which the spectral data will be stored.

<start channel>

Specifies the starting channel number to be used when storing the spectral data.

<stop channel>

Specifies the ending channel number to be used when storing the spectral data.

[CH]

Specifies that the spectral data be fetched from REXX compound variable CH.1 up to CH.4096.

Examples

```
CALL PUTDATA 'MY_EXPER.CNF',401,500
```

Instructs the batch procedure to fetch spectral data from REXX variables DT1 through DT100 and store it in channels 401–500 of datasource MY_EXPER.CNF.

```
CALL PUTDATA 'MY_EXPER.CNF',1025,4096,'CH'
```

Instructs the batch procedure to fetch spectral data from REXX compound variables CH.1 through CH.3072 and store it in channels 1025–4096 of datasource 'MY_EXPER.CNF'.

GETINT

The GETINT function is used to fetch the total counts (integral) of a specified data region and store it in a REXX variable named INTEGRAL.

Function Format

```
CALL GETINT <datasource name>,<start channel>,<stop channel>
```

Parameter Descriptions

<datasource name>

Specifies the name of the datasource from which the raw spectral data will be fetched.

<start channel>

Specifies the starting channel number to be used to define the start of the region.

<stop channel>

Specifies the ending channel number to be used to define the end of the region.

Example

```
CALL GETINT 'DET:S100',100,500
```

Instructs the batch procedure to compute the integral of channels 100-500 from the hardware datasource S100. The value will be returned in the REXX variable INTEGRAL.

EVALENEQ

The EVALENEQ function is used to evaluate the energy calibration equation for a specified channel number. The function will return the energy in keV.

Function Format

```
<REXX variable> = EVALENEQ (<datasource name>,<channel>)
```


Parameter Descriptions

<datasource name>

Specifies the name of the datasource to fetch the energy calibration equation from.

<channel>

Specifies the channel number to be converted to energy.

Example

```
ENERGY = EVALENEQ('MY_EXPER.CNF',400)
```

Instructs the REXX function to fetch the energy calibration from datasource MY_EXPER.CNF and use it to calculate the energy in keV of channel number 400. The resulting energy will be stored in REXX variable ENERGY.

EVALSHEQ

The EVALSHEQ function is used to evaluate the specified shape calibration equation for a specified channel number. The function will return the shape in keV. Full Width Half Maximum (FWHM) or Low Tailing (LOTAIL) shape must be specified.

Function Format

```
<REXX variable> = EVALSHEQ (<datasource name>,<channel>,<shape type>)
```

Parameter Descriptions

<datasource name>

Specifies the name of the datasource to fetch the shape calibration equation from.

<channel>

Specifies the channel number to be converted to FWHM energy or low tail energy.

<shape type>

Specifies the type of shape equation to evaluate. Either FWHM or LOTAIL must be specified.

Examples

```
FWHM400 = EVALSHEQ('MY_EXPER.CNF',400,'FWHM')
```

Instructs the batch procedure to fetch the FWHM shape calibration from datasource MY_EXPER.CNF and use it to calculate the FWHM in keV of channel number 400. The resulting FWHM calibration energy will be stored in REXX variable FWHM400.

```
XYZ = EVALSHEQ('MY_EXPER.CNF',1400,'LOTAIL')
```

Instructs the batch procedure to fetch the low tailing shape calibration from datasource MY_EXPER.CNF and use it to calculate the low tail in keV of channel number 1400. The resulting low tail calibration energy will be stored in REXX variable XYZ.

EVALEFEQ

The EVALEFEQ function is used to evaluate an efficiency calibration equation for a specified energy. The function will return the percent efficiency and absolute error.

Function Format

```
<REXX variable> = EVALEFEQ (<datasource name>,<energy>[,<calibration type>])
```

Parameter Descriptions

<datasource name>

Specifies the name of the datasource to fetch the efficiency calibration equation from.

<energy>

Specifies the energy in keV whose efficiency and efficiency error are calculated.

<calibration type>

Specifies the efficiency equation type to be used. Dual, Linear, Empirical and Average are the types that may be specified. If this parameter is not specified, the equation type (CAM_T_EFFTYPE) specified in the datasource is used, if present, otherwise Dual is used as a default.

Examples

```
TEMP = EVALEFEQ('MY_EXPER.CNF',1100.5) parse var TEMP EFFIC ERROR
```

Instructs the REXX function to fetch the efficiency calibration type and equation from datasource MY_EXPER.CNF and use it to calculate the efficiency at energy 1100.5 keV. The resulting efficiency and error will be stored in REXX variable TEMP. The parse will split the return into REXX variables EFFIC and ERROR.

```
TEMP = EVALEFEQ('MY_EXPER.CNF',511,Linear)parse var TEMP EFFIC ERROR
```

Instructs the REXX function to fetch the Linear efficiency calibration equation from datasource MY_EXPER.CNF and use it to calculate the efficiency at energy 511 keV. The resulting efficiency and error will be stored in REXX variable TEMP. The parse will split the return into REXX variables EFFIC and ERROR.

COUNTREC

The COUNTREC function is used to return the number of records in the CAM class of the specified parameter.

Function Format

<REXX variable> = COUNTREC (<datasource name>,<CAM parameter>)

Parameter Descriptions

<datasource name>

Specifies the name of the datasource in which to find the record count.

<CAM parameter>

Specifies a CAM parameter within the class whose record count is to be returned.

Example

```
RECORDS = COUNTREC('STD.CNF', 'NCLTITLE')
```

Instructs the REXX function to return the number of records in the data class (NUCL) in datasource STD.CNF that CAM parameter NCLTITLE belongs to. The record count will be stored in REXX variable RECORDS.

COUNTENT

The COUNTENT function is used to return the number of entries in the tabular CAM parameter specified.

Function Format

<REXX variable> = COUNTENT (<datasource name>,
<CAM parameter>[,<record number>])

Parameter Descriptions

<datasource name>

Specifies the name of the datasource in which to find the entry count.

<CAM parameter>

Specifies a CAM parameter within the class whose record count is to be returned.

<record number>

Specifies the record number of interest for a record-tabular CAM parameter.

Examples

```
ENTRIES = COUNTENT('STD.CNF', 'EREAL')
```

Instructs the REXX function to return the number of tabular entries for CAM parameter EREAL in datasource STD.CNF. The entry count will be stored in REXX variable ENTRIES.

```
ENTRIES = COUNTENT('STD.CNF', 'NCLLINE', 4)
```

Instructs the REXX function to return the number of record-tabular entries for CAM parameter NCLLINE record 4 in datasource STD.CNF. The entry count will be stored in REXX variable ENTRIES.

MDAPRESET

The MDAPRESET function is used to compute the preset live time (in seconds) required to reach a given MDA for one or more nuclides. The computed preset live time is returned in a REXX variable named PRESET.

It is assumed that a complete analysis (i.e. peak locate/area, efficiency correction, NID and MDA) has been performed on the specified input datasource before calling this function. If the required MDA values are present in the nuclide library used for analysis, only the input datasource is required to be specified as part of MDAPRESET's argument list. MDAPRESET will compare the computed MDA with the required MDA for each nuclide stored in the input datasource and compute the largest preset live time necessary to meet each nuclide's required MDA.

For a given nuclide, the REXX variable PRESET will be left unchanged if any of the following conditions are met:

1. The nuclide has been identified.
2. The nuclide has no computed value.
3. Both the stored required MDA (in the nuclide library) and the required MDA passed through the argument list are zero.
4. The computed MDA is less than or equal to the required MDA.

If all required MDA values have been met, then the REXX variable PRESET will contain zero. A default maximum preset live time may be stored in the input datasource (CAM parameter CAM_X_MAXPLIVE); if this parameter is zero, then a default maximum preset live time of 43 200 seconds (12 hours) will be used.

Function Format

CALL MDAPRESET <input datasource name> [,<nuclides>,<MDAs>]

Parameter Descriptions

<datasource name>

Specifies the name of the input datasource. It is assumed that the datasource contains analysis results through MDA.

[<nuclides>]

Specifies the list (via a REXX stem variable) of nuclides whose MDAs are to be compared with the required MDAs.

[<MDAs>]

Specifies the list (via a REXX stem variable) of required MDAs for each nuclide.

If the stem variable contains a member named *.ALL*, a given nuclide (with a non-zero value for the required MDA) will be used as part of the time-to-MDA calculation whether or not it has already been identified.

If the <nuclides> and <MDAs> arguments are specified, then the required MDA values stored in the specified datasource's nuclide library are not used.

Example

```
"STARTMCA DET:S100 /LIVEPRESET=30"  
"WAIT DET:S100 /ACQ"  
:  
/* Perform preliminary analysis through MDA on input S100  
*/  
:  
Nuclide.1 = "MN-54"  
Nuclide.2 = "CS-137"  
ReqMDA.1 = 0.015  
ReqMDA.2 = 0.0023  
CALL MDAPRESET 'DET:S100', 'Nuclide', 'ReqMDA'  
IF JOBERROR <> 0 THEN DO  
IF PRESET <> 0 THEN DO  
"STARTMCA DET:S100 /LIVEPRESET="PRESET
```

```

"WAIT DET:S100 /ACQ"
      :
/* Perform final analysis through MDA on input S100 */
      :
END

```

This example demonstrates the typical usage of this function. A short acquisition (30 seconds) is performed on detector S100. After acquisition, a preliminary analysis is performed through MDA prior to calling the MDAPRESET function. Two nuclides (^{54}Mn , ^{137}Cs) and their required MDAs (0.015, 0.0023)⁴ are used when calculating the required preset live time to achieve the specified MDAs. The acquisition is restarted using the newly computed preset live time stored in the REXX variable PRESET. Once this acquisition is completed, a final analysis is performed.

ICBCOMM

The ICBCOMM function is used to read/write a set of registers at a specified ICB address. Note that register read/write operations are processed left-to-right. This allows a single function call to perform a sequence of ICB register operations in proper order.

Function Format

```
CALL ICBCOMM <input datasource name>,<ICB Address>,<Arguments>
```

Parameter Descriptions

<datasource name>

Specifies the name of the detector input which defines the Ethernet address to which the ICB commands will be directed.

<ICB Address>

Specifies the ICB address to which all register read/write operations will be directed.

<Arguments>

Specifies the register read/write operations to be performed. Four argument types are supported:

4. These values must be entered as $\mu\text{Ci/unit}$, where unit is as defined in this datasource.

REGnn: Read register nn.
REGnn=<value>: Write <value> to register nn.
Model: Read model number.
Serial: Read serial number.

Values that are read are returned in REXX variables of the same name. For instance: a value read from REG4 will be returned in REXX variable REG4; a serial number that is read will be returned in the REXX variable SERIAL.

Note that which registers are readable, which are writable, and what meaning is given to particular values, depends entirely on the design of the device at that ICB address.

Example

```
CALL ICBCOMM 'DET:AIM1',2,'MODEL','SERIAL','REG4','REG5'=11
```

Instructs the batch procedure to read model (returned in REXX variable MODEL) and serial number (returned in REXX variable SERIAL) from ICB address 2 off the Ethernet address specified by detector input AIM1. Register number 4 is then read (and returned in REXX variable REG4) followed by a write of 11 to register 5.

DTCONVERT

The CAM Date/Time REXX function converts a file's date and time to a CAM date/time value, from a CAM date/time value to an ASCII string, or from the system date/time to a CAM date/time value.

Function Format

```
RetVal = DTCONVERT( Argument[,C] )
```

Parameter Descriptions

<Argument>

The argument can take one of three forms:

1. CAM date/time value. The value returned will be an ASCII date/time string in the format specified via the Country Settings options within the MS-DOS environment. The year is expressed as a two-digit value unless the second parameter is C. The C causes the year to be expressed as a four-digit value.
2. No argument specified. The value returned will be a CAM date/time value representing current system time.
3. File timestamp as returned by the REXX function SysFileTree using the "T" option. The value returned will be a CAM date/time value.

Examples

```
CurTime = DTConvert()
```

Returns the current system time as a CAM date/time value and stores it in REXX variable CurTime.

```
CALL GETPARAM 'C:\GENIE2K\NBSSTD.CNF',1,1,'STIME'
```

```
TimeString = DTConvert(STIME)
```

Fetches the sample date/time from NBSSTD.CNF and converts it to an ASCII date/time string stored in the REXX variable TimeString.

GETRECS

The GETRECS function is used to fetch multiple records of CAM parameters and store them in REXX variables.

Function Format

```
<REXX variable>=GETRECS ( <datasource name>,
    <record>,<entry>,<rec_count>,<par1>,...,<par10> )
```

Parameter Descriptions

<datasource name>

Specifies the name of the datasource file from which the records are to be fetched.

<record>

Specifies the starting record number of the records to be fetched.

<entry>

Specifies the entry number.

<rec_count>

Specifies the maximum number of records to fetch.

<par1 - par10>

Specifies the names of the CAM parameters; up to 10 sets of parameter values can be read with one call.

Example

```
<nr>=GETRECS ('C:\TEST.CNF',3,0,99,'PSCENTRD','PSENERGY')
```

If successful, up to 99 records will be read from C:\TEST.CNF, starting with record number 3. The variable 'nr' will contain the number of records actually read, and the values of the REXX variables PSCENTRD.3 and PSENERGY.3 will be defined; they will contain the values of the specified parameters read from record 3. Likewise, PSCENTRD.4 and PSENERGY.4 will have the values read from record 4, etc.

PUTRECS

The PUTRECS function is used to write multiple records of CAM parameter values to a datasource.

Function Format

```
<REXX variable>=PUTRECS ( <datasource name>,  
    <record>,<entry>,<rec_count>,<par1>,...,<par10> )
```

Parameter Descriptions

<datasource name>

Specifies the name of the datasource file to which the records are to be written.

<record>

Specifies the starting record number of the records to be written.

<entry>

Specifies the entry number.

<rec_count>

Specifies the number of records to write.

<par1 - par10>

Specifies the names of the CAM parameters to be written; up to 10 sets of parameter values can be written with one call.

Example

```
nr=PUTRECS ('C:\TEST.CNF',3,0,12,'PSCENTRD','PSENERGY')
```

If successful, up to 12 records, starting with record number 3, will be written to datasource C:\TEST.CNF. The values of REXX variables PSCENTRD.3 through PSCENTRD.14 and PSENERGY.3 through PSENERGY.14 must be defined and

will be used to update those records. The variable 'nr' will contain the number of records actually written.

GETENTS

The GETENTS function is used to fetch multiple entries of CAM parameters and store them as REXX variables.

Function Format

```
<REXX variable>=GETENTS ( <datasource name>,
    <record>,<entry>,<ent_count>,<par1>,...,<par10> )
```

Parameter Descriptions

<datasource name>

Specifies the name of the datasource file from which the entries are to be fetched.

<record>

Specifies the number of the record to be fetched.

<entry>

Specifies the starting entry number.

<ent_count>

Specifies the maximum number of entries to fetch.

<par1 - par10>

Specifies the names of the CAM parameters to fetch the entries from; up to 10 sets of parameter values can be read with one call.

Example

```
ne=GETENTS ('C:\TEST.CNF',6,1,9,'NCLLINE')
```

If successful, up to 9 entries starting with entry number 1, will be read from the sixth record (nuclide) in C:\TEST.CNF. The variable 'ne' will contain the number of entries actually read, and the values of the REXX variables NCLLINE.1 through NCLLINE.9 will be defined; they will contain the values of the specified parameter.

PUTENTS

The PUTENTS function is used to write multiple entries of CAM parameters to a datasource.

Function Format

```
<REXX variable>=PUTENTS ( <datasource name>,  
    <record>,<entry>,<ent_count>,<par1>,...,<par10> )
```

Parameter Descriptions

<datasource name>

Specifies the name of the datasource file to which the entries are to be written.

<record>

Specifies the number of the record to be updated.

<entry>

Specifies the starting entry number.

<ent_count>

Specifies the number of entries to write.

<par1 - par10>

Specifies the names of the CAM parameters to be written; up to 10 sets of parameter values can be written with one call.

Example

```
ne=PUTENTS ('C:\TEST.CNF',7,1,9,'NCLLINE')
```

If successful, up to 9 entries, starting with entry number 1, will be read from the REXX variables NCLLINE.1 through NCLLINE.9 and written to record 7 of the NUCL class in datasource C:\TEST.CNF. The variable 'ne' will contain the number of entries actually written.

PUTPARAM

The PUTPARAM command is used to modify specified CAM parameters by fetching each CAM parameter from a REXX variable and writing them to a datasource (the name of the REXX variable is the name of the CAM parameter being written).

Function Format

```
CALL PUTPARAM <datasource name>,<record number>,  
    <entry number>,<CAM parameter 1>,...,<CAM parameter 10>)
```

Parameter Descriptions

<datasource name>

Specifies the name of the datasource to which the requested CAM parameter(s) will be written.

<record number>

Specifies the record number to be used when writing the requested CAM parameter(s).

<entry number>

Specifies the tabular entry to be used when writing the requested CAM parameter(s).

<CAM parameter name>

Specifies the name(s) of up to 10 CAM parameters to be written with this PUTPARAM call.

Examples

```
CALL PUTPARAM 'DET:S100',3,1,'RGNSTART','RGNEND'
```

Instructs the batch procedure to write the start and end channels of region of interest number three to datasource S100. The current values of the two REXX variables will be used to do the update.

INSERTREC

The INSERTREC function is used to insert records or tabular entries into a datasource.

Function Format

```
CALL INSERTREC <datasource name>,<cam-param>,<rec>,<tab>
```

Parameter Descriptions

<datasource name>

Specifies the name of the datasource file into which the record is to be inserted.

<cam-param>

Specifies the name of any CAM parameter of the class in which a record is to be inserted.

<rec>

This input parameter specifies the record number to be used as follows:

- If record number is zero, then the insert is for a common tabular entry (see the <tab> parameter).
- If record number is non-zero and the <tab> parameter is zero, then the insert creates an empty record in front of the record number specified.

<tab>

This input parameter specifies the entry number to be used as follows:

- If entry number is zero, then the insert is to create a new record (see the <rec> parameter).

If entry number is non-zero and the <rec> parameter is zero, then the insert creates an empty common tabular entry in front of the entry number specified. If the <rec> parameter is non-zero, then the insert creates a record tabular entry in front of the entry number specified on the record specified.

Example

```
CALL INSERTREC 'C:\TEST.CNF','PSCENTRD',9,0
```

If successful, a record of the PEAK class will be inserted into datasource c:\test.cnf in front of the current record 9.

DELETEREC

The DELETEREC function is used to delete records or tabular entries from a CAM file.

Function Format

```
CALL DELETEREC <datasource name>,<cam-param>,<rec>,<tab>
```

Parameter Descriptions

<datasource name>

Specifies the name of the datasource file from which the record is to be deleted.

<cam-param>

Specifies the name of any CAM parameter of the class from which a record is to be deleted.

<rec>

This input parameter specifies the record number to be used as follows:

- If record number is zero, then the delete is for a common tabular entry (see the <tab> parameter).

If record number is non-zero and the <tab> parameter is zero, then the specified record is deleted and all subsequent records are moved up (i.e. if record number 4 is deleted, then record no. 5 becomes number 4, etc.).

<tab>

This input parameter specifies the entry number to be used as follows:

- If entry number is zero, then the delete is for a record (see the <rec> parameter).

If entry number is non-zero and the <rec> parameter is zero, then the specified common tabular entry is deleted and all subsequent entries are moved up (i.e. if entry number 4 is deleted, then entry number 5 becomes number 4, etc.). If the <rec> parameter is non-zero, the specified record tabular entry is deleted and all remaining entries in that record are moved up.

Example

```
CALL DELETEREC 'C:\TEST.CNF', 'PSCENTRD', 9, 0
```

If successful, the ninth record of the Peak Search Results (PEAK) class will be deleted from datasource c:\test.cnf.

COPYREC

The COPYREC function is used to copy an entire record from one datasource to another.

Function Format

```
CALL COPYREC <src-ds>, <cam-param>, <src-rec>,
             <dst-ds, <dst-rec>[, <flags>]
```

Parameter Descriptions

<src-ds>

Specifies the name of the datasource from which the record is to be copied.

<cam-param>

Specifies the name of any CAM parameter of the class from which a record is to be copied.

<src-rec>

Specifies the number of the record in the source datasource.

General Job Functions

<dst-ds>

Specifies the name of the datasource to which the record is to be copied.

<dst-rec>

Specifies the number of the record in the destination datasource.

<flags>

Specifies that the COPYREC function is to be used as a “moverec” function. A value of ‘D’ (for delete) will delete the specified ‘src-rec’ from the source datasource after it has been copied.

Notes

The action on the <dst-ds> is always an *insert*; all records with numbers <dst-rec> and higher will be moved.

Specifying <dst-ds> as a null string (“”) means that the record is to be *moved* from one location to another within <src-ds>.

Example

```
CALL COPYREC 'C:\TEST.CNF', 'PSCENTRD', 6, 'C:\MYSOURCE.CNF', 9
```

If successful, the CAM parameter PSCENTRD and all other values in its record will be copied from record 6 in datasource ‘c:\test.cnf’ to record 9 in datasource ‘c:\mysource.cnf’. Record 6 will not be deleted from ‘c:\test.cnf’ after the copy has been made.

GETTEXT

The GETTEXT function is used to fetch text from a specified file and store it in a REXX stem variable. The file can be thought of as a “translation table.”

This function returns the following exit status code via the REXX variable JOBERROR:

- 2 Error opening specified text file
- 4 Error reading from specified text file
- 6 Bad keyword found in specified text file
- 8 No keyword found in specified text file

-10 Error interpreting contents of specified text file

Function Format

```
CALL GETTEXT <text-file>,<sect1>,<var1>,...,<sect8>,<var8>
```

Parameter Descriptions

<text-file>

Specifies the path/filename from which the text is to be fetched.

<sect*n*>

Specifies the text section name. Strings in the file are divided into sections. All strings belonging to the specified section will be fetched.

<var*n*>

Specifies the name of the REXX stem variable where the strings for that section are to be stored. The members of that variable will be the names of the section members; up to eight section names may be included. There must be a <var*n*> for each <sect*n*> specified.

Example

```
CALL GETTEXT 'C:\MSG.S.TXT','common','cmntxt'
```

If the c:\msgs.txt file contains:

```
!
!Example GETTEXT file
!
common! This is a section name
{Title ; This is a title }
{Error! ; Failure at start }
```

then the values of the cmntxt.Title and cmntxt.Error! REXX variables will be defined and will contain the strings that are paired with each of those two members.

Serial I/O REXX Functions

The RXCOMPO.DLL routine has four entry points:

```
COM_OPEN
COM_SEND
```


General Job Functions

COM_GET
COM_CLOSE

COM_OPEN (Port, PortHandle[, Baud[, Parity]])

Where:

Port is the COM port to open. [COM1, COM2, etc.]

PortHandle is the returned handle to the COM port opened by this call.

Baud (optional, default 9600 baud) is the COM port baud rate. [2400, 4800, 9600, 19200, etc.]

Parity (optional, default no parity) is the COM port parity setting. [0 - NoParity, 1 - OddParity, 2 - EvenParity]

Return: 0 - Success
 1 - Error opening COM port
 2 - Error getting Comm State
 3 - Error setting Comm State
 4 - Error getting Comm Timeouts
 5 - Error setting Comm Timeouts
 6 - Error setting Comm Setup

COM_SEND (PortHandle, Message)

Where:

PortHandle is the handle returned by the Open.

Message is the data to be sent to the serial device attached to the COM port.

Return: 0 - Success
 7 - Error writing to COM port

COM_GET (PortHandle, InputBuffer)

Where:

PortHandle is the handle returned by the Open.

InputBuffer is the REXX buffer to receive the data sent by the serial device to the COM port.

Return: 0 - Success
 7 - Error reading from COM port
 9 - No data read from COM port

COM_CLOSE (PortHandle)

Where:

PortHandle is the handle returned by the Open.

Return: 0 - Success
 9 - Error closing COM port

Sample Code

-
-
-

```
CALLRXFUNCADD 'COM_OPEN', 'RXCOMPO', 'COM_OPEN'
```

```
CALLRXFUNCADD 'COM_SEND', 'RXCOMPO', 'COM_SEND'
```

```
CALLRXFUNCADD 'COM_GET', 'RXCOMPO', 'COM_GET'
```

```
CALLRXFUNCADD 'COM_CLOSE', 'RXCOMPO', 'COM_CLOSE'
```

```
NoParity = 0
OddParity = 1
EvenParity = 2
```

```
Port = 'COM1'
Baud = 19200
Parity = EvenParity.
```

-
-
-

```
/* Open COM port 1 at 19200 baud and even parity */
rc = COM_OPEN(Port, ComPortHandle, Baud, Parity)
```

Batch Procedure Example

```
/* Send a message to the serial device */  
Msg = 'This is a test of Serial Communications' || '0D0A'x /* cr lf */  
rc = COM_SEND(ComPortHandle, Msg)  
  
/* Get a response from the serial device into buffer */  
rc = COM_GET(ComPortHandle, Buffer)  
  
/* Close the COM port when done */  
rc = COM_CLOSE(ComPortHandle)
```

Batch Procedure Example

This section describes the command procedure copied to your hard disk during Genie-2000 software installation. This procedure is an example of standard gamma spectroscopy operations which demonstrates the following features:

Variable definition

Prompts for user input

Use of user input

Text messages

Use of sub-procedures

Error checking

Transfer to different parts of the procedure

Conditional operations

Performance optimizations

The ROUTINE.REX example procedure is an ASCII file which you can read, modify, or print out with Notepad or any word processor which can save files in ASCII format. The file will normally be found in the default directory, C:\GENIE2K\EXEFILES, unless you chose to put it in another directory during installation.

The procedure is fully commented, clearly setting forth the procedure's definition and operation.

Routine Gamma Spectrum Analysis

The ROUTINE.REX procedure provides options to count and analyze a sample, count a sample and save the results, or analyze previously saved results. You'll be prompted for the option you want to use.

If a sample is to be counted, you'll be prompted for sample ID, description, and so forth, and count time.

If analysis is of a spectrum on file, you'll be asked for the file name.

The analysis is based on a sample analysis sequence file NID_SHO.ASF.

3. Graphical Batch Tools

Genie-2000's Graphical Batch Tools is a family of programming extensions to the Genie-2000 job and REXX command language environments. These tools allow development of custom Genie-2000 batch procedures which have a Graphical User Interface similar to the interactive Genie-2000 environment.

Further information will be found in Chapter 2, *Batch Procedure Reference*, which covers all of the Genie-2000 batch job commands and their operation, and in Chapter 4, *Graphical Batch Tools Example*, which describes the steps in creating an example Graphics Batch Tools application.

Command Index

<u>Command</u>	<u>Description</u>	<u>Page</u>
GBT_CLEAR	Clears the application dialog window's report area.	144
GBT_COUNTDS	Returns the count of datasources found by a GBT_ENUM (enumeration).	148
GBT_DEFMENU	Defines the set of pulldown menus available in the application dialog window.	166
GBT_ENUM	Brings up an enumeration dialog screen based on a datasource "filter".	169
GBT_FINDDS	Brings up a dialog for selecting a Genie-2000 datasource (similar to the Open dialog in the Genie-2000 interactive environment).	147
GBT_FINDMID	Lists all loaded MID files that have inputs available.	152
GBT_GETMENU	Enables the pulldown menu in the application dialog window, allowing the operator to select one of the menu options.	142

GBT_GETXY	Returns the size of the screen, in rows and columns.	146
GBT_INIT	Initiates an application dialog window which is used by all of the Graphical Batch Tool functions (this is a command; that is, an .EXE file).	140
GBT_LOGOFF	Performs a logoff (sets the security level and all access bits to 0).	152
GBT_LOGON	Puts up a dialog which asks the user for a username and password.	151
GBT_MESSAGE	Puts up a “modal” or a “modeless” message dialog.	145
GBT_NOTE	Puts a one-line message in the application dialog window’s report area.	143
GBT_PARS	Puts up a parameter editing screen for viewing or updating a Genie-2000 datasource.	166
GBT_QUERY	Puts up a dialog that asks the operator for a single response.	173
GBT_REPORT	Puts the contents of an ASCII file in the application dialog window’s report area.	143
GBT_SELECT	Puts up a selection dialog which allows the user to pick from a list of pre-defined choices.	171
GBT_TERM	Terminates the application dialog window.	145
GBT_VIEW	Displays the contents of an ASCII file in a scrollable, multi-line dialog.	150
GBT_WAIT	Causes program to wait until acquisition on the specified datasource is complete.	174

Overview

The Graphical Batch Tools include one job command (similar to the set of Genie-2000 job commands described in Chapter 2, *Batch Procedure Reference*), and a set of REXX functions (similar to the Genie-2000 REXX functions described in “Job Function Descriptions” on page 103).

A batch procedure that uses the Graphical Batch Tools will always have the following structure:

/* The first line must be a comment line for REXX */

REXXFCTS /* Register the Genie-2000 REXX functions */

GBT_INIT /* Bring up the GBT application window */

-
-
-

Invoke Genie-2000 job commands, Graphical Batch Tool functions and MS-DOS commands.

-
-
-

CALL GBT_TERM /* Terminate the GBT application window */

EXIT

The GBT_INIT program launches a PM-based application dialog window. All other Graphical Batch Tool functions perform their operation by directing graphical operation requests to this dialog window for execution.

Some of the Graphical Batch Tool functions make use of a Form Design Specification (FDS) file which defines the “look-and-feel” of the graphical dialog presented on the screen. These FDS files are ASCII files that are constructed by the user using a standard text editor; refer to “Form Design Specification Files” on page 154 for more details on FDS files, Parameter Definition Records (PDR), and their syntax.

Environment Variables

The Graphical Batch Tools use environment variables which define the location of files processed or used by the tools. These environment variables must be defined *before* invoking the GBT_INIT command (page 140).

GBT_FDSFILES: Fully specified pathname of the directory where all .FDS files are located.

GBT_PASSWORDS: Fully specified pathname of the password file used by the Graphical Batch Tools for user logon and security checking.

GBT_FDSHELPS: Fully specified pathname of the directory where the files accessed by the HELP button are located.

Password Security

The Graphical Batch Tools support both the CAM Security class of data and an ASCII Password File which stores username, password, and security level information. The security function is established through a logon process which asks for a username and password combination. The username/password entry is then compared to the entries in the Graphical Batch Tools password file pointed to by the environment variable GBT_PASSWORDS.

If the GBT_PASSWORDS environment variable specifies the name of a CAM file, the function will use the CAM Security class to verify the username-password combination being entered. If the environment variable does not specify the name of a CAM file, the function will assume that the variable is specifying the name of an ASCII Password File.

ASCII Password File

If the GBT_PASSWORDS environment variable is an ASCII Password File, the security level for that logon becomes the level used for all subsequent Graphical Batch Tool functions. Dialog items (like selection choices, pushbuttons, edit fields, etc.) are enabled if the logged-on security level is greater than or equal to the security level defined for the dialog item.

The ASCII password file is formatted like this:

```
<UserName> <Security Level> <Password>
```

There is a limit of 16 characters for both the UserName and Password. The security level must be a number between 0 and 65535. The password is optional; that is, usernames can be defined that do not require a password.

CAM File Security

If the GBT_PASSWORDS environment variable points to a CAM security file, that file is used as follows:

The entered user-name and password are compared with the parameters SECENTNAME and SECPASSWORD. When the match is made, the values of the SECLEVEL and SECMASK parameters from the matching record are saved; they are used to verify access to the dialog controls.

As with the ASCII file, the password is optional; that is, usernames can be defined that do not require a password.

The value of *level* will be used just as it is for the ASCII file. However, if the security value for the dialog item is expressed as *:nn*, then that *bit* must be set in the value of SECMASK in order to have that item be enabled. For example, if the item has *:3* in the security field, it will be enabled only if SECMASK has the 8-bit set.

FDS File Security Fields

The security fields within an FDS file allow either a security level or a CAM parameter access bit number in the form *:nn*. Note that a field that has 0 (or no value) will always be enabled for all users.

If a bit number *:nn* is specified, the password file must be a CAM file; access to that item will be limited to those users who have that bit set in their security record.

Function Calling Conventions

The REXX functions can be invoked in one of two ways from within a REXX command procedure:

Case 1: `CALL GBT_GETXY`

Case 2: `Response = GBT_GETXY()`

Case 1's return value is always available in the REXX variable 'Result'; it is usually used when the program is not interested in the return value. Case 2 is used when the return value from the function is to be stored in the REXX variable 'Response'.

With one exception, all of the functions described here require GBT_INIT to have been invoked before the function is called. That is, they require the GBT application window to be present in order to perform their function.

Function Return Values

Values returned by the Graphical Batch Tool functions fall into three classes:

1. The null (empty) string.

This value is typically returned when an error is encountered by a function. Refer to "Error Codes" on page 175 for more details on error returns.

2. A button identifier.

Those functions that bring up a dialog that contains pushbuttons (for example, GBT_MESSAGE) return this identifier indicating which pushbutton was pressed. The button identifier is returned in the form:

>nn

where *nn* is the ID number of the pushbutton.

3. A return value.

Functions that are defined to request information from the user return an actual value. In this case, the OK pushbutton (always defined as identifier number 1) will cause that value to be returned (instead of “>1”).

Function Error Returns

There are three classes of errors that can be encountered and returned during a call to one of the Graphical Batch Tool functions:

1. Application Level Errors

This error return, which is assigned to the REXX variable JOBERROR, is identical to the value returned by the standard Genie-2000 REXX functions described in “Job Function Descriptions” on page 103), with the following additions:

94: Failure to communicate with application window

96: Graphical Batch Tool error encountered.

2. System Level Errors

This error return, which is assigned to the REXX variable ERRCAUSE, is defined whenever JOBERROR has the value 96.

3. FDS File Parsing Errors

These error returns are defined whenever JOBERROR has the value 96 and ERRCAUSE has the value 0x600322 (refer to “Error Codes” on page 175 for more details). These values indicate that errors were detected while parsing an FDS file; the errors will be written to an error log file whose filename is the base name of the FDS file with the .LOG extension. This error log file will be found in the path defined by the environment variable GBT_FDSFILES.

“Error Codes” on page 175 provides a complete listing and description of all the error codes returned by the Graphical Batch Tool functions. Refer to Appendix 1, *Error Codes and Messages*, for more information on error codes.

Description of Commands and Functions

This section describes in detail the operation of the various commands and functions which comprise the Graphical Batch Tools: Batch Tool Commands (page 140), “Fixed Format” functions (page 142), and “FDS Driven” functions (page 154). Each description follows a basic format:

Description

Command/function format

Qualifier/parameter descriptions

Examples

Unless specifically stated otherwise, all commands and functions follow the same operational rules described in Chapter 2, *Batch Procedure Reference*.

The functions will typically be presented with parentheses, as if they are to be used in the “variable=function(arguments)” form.

However, if no results are expected, or if the value of the response has no use in this situation, the command-file can use the form:

```
CALL function arg1, arg2, arg3
```

An argument named <FDS-file> represents the name of an FDS file as described in “Form Design Specification Files” on page 154.

Optional Arguments

Optional arguments are shown with square brackets. If the second argument is optional and unneeded, but you need to use the third argument, you must provide the comma as a place holder for the second argument. For example, CALL function arg1,,arg3. Note that when REXX stem variables are specified as arguments, individual members of that stem variable are considered optional; that is, not all stem variable members need to be specified.

Show Argument

The final SHOW argument (where defined) always has the same meaning; it is the name of a REXX 'stem' variable whose members can be any combination of the following:

.TITLE	The text that will show in the dialog's title-bar.
.MINROWS	The least number of rows that the dialog will occupy.
.MINCOLS	The least number of columns that the dialog will occupy.
.POSITION	This is a three-character string. The first character is O if you want the dialog located Outside of the application dialog window, or I for Inside. The second character positions the dialog vertically. It can be T, B or C, to place the dialog box relative to the Top, the Bottom or the (vertical) Center of the application dialog window. The third character positions the dialog horizontally. It can be L, R or C, to place the dialog box relative to the Left edge or the Right edge, or to Center it horizontally on the application dialog window.

For example, ICC means, place the dialog in the center of the application dialog window. The default is OCC, which means centered on the screen.

Graphical Batch Tool Command

This section describes GBT_INIT, a command used by all Graphical Batch Tool functions, with its format and qualifiers, and an example of its use.

GBT_INIT

This command (an .EXE file) initiates an application dialog window which is used by all subsequent Graphical Batch Tool functions. Note that the environment variables described on page 135 must be defined *before* the GBT_INIT command is invoked. To terminate this application dialog window, call GBT_TERM (page 145).

Command format

GBT_INIT [/Qualifiers]

Qualifiers

[/HIDDEN]

Will cause the application window dialog to be invisible. If this qualifier is used, all other qualifiers (with the exception of /SECURITY) have no effect.

[/TITLE=<text-string>]

Specifies the title bar text for the application dialog window.

[/SECURITY=<value>]

Assigns an initial security-level, as if a user with that level had just logged on. If not specified, the security level value defaults to 65535.

[/XY=<starting X value,starting Y value>]

Specifies the starting X and Y screen coordinates (in pixels) of the lower-left corner of the application dialog window. The default is <1,1>, which is the lower left corner of the screen. Note that if the XY values are preceded by a '-', the numbers are interpreted as a percentage of screen size (for example, -50,-50 puts the window in the center of the screen).

[/CXCX=<delta X value,delta Y value>]

Specifies the X and Y screen size (in pixels) of the application dialog window. The default is <150,100>. Note that if the CXCX values are preceded by a '-', the numbers are interpreted as a percentage of screen size (for example, -100,-50 creates a window which is the full width of the screen and half of the screen height).

[/ICONIZE]

Will force the application dialog window to come up in a minimized (or iconized) state.

[/NOMAXIM]

Will prevent the user from maximizing the application dialog window.

[/NOMINIM]

Will prevent the user from minimizing the application dialog window.

[/NOSIZE]

Will prevent the user from changing the size of the application dialog window by grabbing and dragging the window edges.

[/DEVELOP]

Enables the output of internal diagnostic messages to the application dialog window report area.

Examples

```
GBT_INIT /XY=1,1 /CXCX=-100,-50 /NOMAXIM
/NOSIZE /NOMINIM /TITLE="Batch Procedure No. 1"
```

This command will bring up a Graphical Batch Tool application dialog window on the bottom half of the screen; the user will not be able to resize, maximize, or minimize the window. The window will be titled "Batch Procedure No. 1".

Fixed Format Functions

This section describes in detail the operation of those Graphical Batch Tool functions that do not make use of an FDS file; that is, "fixed format" functions.

GBT_GETMENU

This function is used to ask the operator for a selection from the menu-bar. When this is called, the menu-bar of the application window dialog, and any pull-down menu items that pass the security-level check, will be enabled, and the function will wait for a response.

The response will ordinarily be a two (or three) character string. The first character of that string will be the character in the name of the active menu-bar-item which is marked in the FDS file with a tilde; the second character will be the character with a tilde in the selected menu-item (note that the tilde indicates which keyboard key can be used to make that selection). If the FDS file uses the Cascade keyword, the response can have a third character representing a selection from that level.

In the special case where there are no menu items named in the FDS file for a particular pull-down menu, the response can be a single character. When that menu-bar item is selected, the response will immediately be returned to the program.

See "GBT_DEFMENU" on page 166 for more information.

Function Format

Response = GBT_GETMENU (<default-menu>)

Parameter Descriptions:

<default-menu>

This optional argument is the menu-item to be selected as default (that is, the corresponding pull-down menu-item will be selected, ready for the ENTER key).

Example

Response = GBT_GETMENU("FO")

Assuming that the menu-bar of the application window dialog contains a File menu, with two menu items Open and Close, this example will pull down the File menu with the Open menu item highlighted. If the operator simply hits the ENTER key, 'Response' will be set to "FO".

GBT_NOTE

This function will simply output the specified line of text to the next available line in the report area of the application dialog window.

Function Format

CALL GBT_NOTE <text-string>

Parameter Descriptions

<text-string>

The text string that will appear in the report area of the application dialog window.

Example

CALL GBT_NOTE "This is a message"

This call will output the message text "This is a message" in the report area of the application dialog window.

GBT_REPORT

This function will simply output the contents of the specified ASCII file starting at the next available line in the report area of the application dialog window.

Function Format

```
CALL GBT_REPORT <ASCII-file>
```

Parameter Descriptions

<ASCII-file>

A fully-specified pathname defining the file whose contents will appear in the report area of the application dialog window.

Example

```
CALL GBT_REPORT "C:\GENIEPC
\REPPFILES\NBSSTD.RPT"
```

This call will output the contents of the report file for NBSSTD.CNF in the report area of the application dialog window.

GBT_CLEAR

This function will clear the application dialog window's report area. It can also be used to dismiss a "modeless" dialog by calling GBT_CLEAR<any-arg>.

Function Format

```
CALL GBT_CLEAR [<Any-arg>]
```

Parameter Descriptions

[<Any-arg>]

Any argument value; when present, this instructs the GBT_CLEAR function to dismiss a "modeless" message box. If the value is a number (n=1-5), a GBT_PARS dialog that was brought up with .MODELESS=n will be dismissed. The contents of the application dialog window's report area are not changed.

Examples

```
CALL GBT_CLEAR
```

This call will clear the application dialog window's report area.

```
CALL GBT_CLEAR 2
```

This call will dismiss modeless GBT_PARS dialog number 2 from the screen without clearing the report area.

GBT_TERM

This function terminates the application dialog window being used by the Graphical Batch Tools environment. This should be the last function called before exiting a Genie-2000 batch procedure that uses these functions.

Function Format

```
CALL GBT_TERM
```

Parameter Descriptions

None

Example

```
CALL GBT_TERM
```

Terminate the application dialog window in preparation for exiting the batch procedure.

GBT_MESSAGE

This function takes the specified text string and displays it in a message dialog window on the screen. The message dialog can be either modal (remains until the user acknowledges by pressing one of two possible pushbuttons) or modeless (remains until the next call to a Graphical Batch Tool function) depending on the presence of a “button parameter” (refer to Parameter Descriptions below).

In the modal case, the return value of this call will be the button identifier (“>1” for the first button, “>2” for the second button); in the modeless case, an empty string is returned.

Function Format

```
Response = GBT_MESSAGE(<text-string>,  
[<options>], [<show>])
```

Parameter Descriptions

<text-string>

The text string which is displayed in the message dialog box.

[<options>]

A REXX stem variable defined as follows:

.BUTTON1

Can be an empty string if only an “OK” button is needed. Anything else (~Yes) will be used as the name of that button.

.BUTTON2

Can be an empty string if a “Cancel” button (along with an “OK”) is needed. Anything else (~No) will be used as the name of a second button.

.ICON

Can be “?”, “!” or “*”; any other character is the “stop” icon. If undefined, no icon will be displayed.

.BEEP

If this member exists, the call generates a beep when the message is displayed.

[<show>]

The show argument is described on page 140.

Example

```
Options.Button1 = “~Yes”
Options.Button2 = “~No”
Options.Icon = “?”
Response = GBT_MESSAGE( “Are you sure?”, “Options”)
```

This example will display a modal message dialog box which asks the user “Are you Sure?”. The “query” icon will be displayed to emphasize the intent. A “>1” will be returned if the Yes button is pressed; a “>2” will be returned if the No button is pressed.

GBT_GETXY

This is the one Graphical Batch Tools function that does not require GBT_INIT to have been invoked prior to its use. This function returns a string in the form:

```
<columns> <rows>
```

The first number is the width of your screen in columns, and the other is the height in rows.

If you need to use the .MINCOLS and .MINROWS values in the “show” argument (for example, in order to size a dialog to be a particular fraction of the screen), this function will prove useful.

Note that using REXX to separate the two values that are returned is straightforward. After calling the function with the return value stored in the REXX variable 'Response', the REXX statement to retrieve the two values would be:

```
PARSE VAR Response Columns Rows
```

The REXX variable Columns now contains the number of columns and the variable Rows contains the number of rows on the screen.

Function Format

```
Response = GBT_GETXY()
```

Parameter Descriptions

None

Example

```
Response = GBT_GETXY()  
PARSE VAR Response C R
```

This example returns the number of columns and rows of the screen and stores the values in the REXX variables C and R.

GBT_FINDDS

This function brings up a dialog screen which looks like a standard Open Datasource screen in Genie-PC. This screen allows the user to change disk drives and directories while looking for the desired datasource.

The value returned by this function is either the fully specified pathname of the selected datasource or >2 if canceled.

Function Format

```
Response = GBT_FINDDS(<filter>, [<flag>], [<show>])
```

Parameter Descriptions

<filter>

A pathname filter which is used to have a specified starting point for the Open dialog (for instance, C:\GENIEPC\CAMFILES*.CNF).

[<flag>]

The name of a REXX stem variable defined as follows:

.FILES

Used as a flag. If this member exists, the Open dialog will be restricted to file datasources only.

.INFO

Causes the dialog to have an “Info” button, which will bring up a dialog displaying the values of the `_SAMPLEID`, `_STITLE`, `_SDESC1` and `_ASTIME` parameters from the currently selected datasource. This parameter should only be used when the filter specifies detectors or CAM files.

[<show>]

The show argument is described on page 140.

Example

```
Flag.Files = 1
```

```
Response = GBT_FINDDDS("C:\GENIEPC\CAMFILES\*.CNF", "Flag")
```

An open datasource box will be displayed. The dialog will come up with a default disk of C:, a default directory of GENIEPC\CAMFILES, and will show all files that fit the filter *.CNF. Only file datasources will be available.

GBT_COUNTDS

This function supplements `GBT_ENUM`, in that given the same filter (and the same limiting options) as a call to `GBT_ENUM`, it returns a count of the datasources that the operator would be shown by that enumerate. Optionally, this function can return the name of a datasource (refer to the `.FETCH` parameter).

Function Format

```
Response = GBT_COUNTDS(<Filter>, [<Arg>])
```

Parameter Descriptions

<Filter>

A datasource filter used to point to (and restrict) the set of datasources that will be presented as part of an enumeration. It can represent files or detectors (for example, file: `C:*.CNF`; detector: `DET:*` or `\\node1\DET:*`).

[<Arg>]

A REXX stem variable defined as follows:

Fixed Format Functions

.AVAIL	Used as a flag. If a member with this name exists, the list of datasources will be restricted to those which can be opened for write-access.
.IDLE	Used as a flag. If a member with this name exists, the list of (detector) datasources will be restricted to those which are not currently acquiring.
.ONLINE	Used as a flag. If a member with this name exists, the list of (detector) datasources will be restricted to those detectors which are <i>not</i> off-line (that is, CAM parameter DETNOTAVAIL is 0).
.FETCH	Index of the datasource (in the list that GBT_ENUM would generate, given the same filter and flags) whose name is to be returned by this call (that is, if .FETCH = 3, the third member of the list will be returned). If this member does not exist, the function will return the count of datasources that would be enumerated.
.DEVICE	Restricts the enumerated datasources to those containing the specified device. The devices and their codes are listed under DEVICE on page 152.
.DEVICETYPE	Restricts the enumerated datasources to those with this type of the specified device (chosen in .DEVICE). Valid device types are listed under DEVICETYPE on page 153.

Note

A value of 0 for the .FETCH member has been given special meaning. Its purpose is to retrieve the name of the one datasource that meets the specified criteria, if there is *just one*. That is, it can be used to replace the sequence:

```
IF GBT_COUNTDS("DET:*","Arg") = 1 THEN DO
  Arg.Fetch = 1
  OnlyDS = GBT_COUNTDS("DET:*","Arg")
END
```

When .FETCH=0, there are three possible return values: 0 means there are no datasources that meet the selection criteria; 2 means there is more than one; otherwise the result will be the name of the one datasource.

Example

```
Arg.Avail = 1
Arg.Idle = 1
Arg.Online = 1
Arg.Fetch = 3
Response = GBT_COUNTDS("DET:*", "Arg")
```

This example would return the fully specified name of the third detector in the list that is: available, not busy, and on line.

GBT_VIEW

This function will display a text file in a multi-line edit-box dialog. The edit-box will be scrollable, both left/right and up/down, so that the entire contents of the file can be viewed.

The return value of this call will be the button identifier (“>1” for the first button, “>2” for the second button, if present).

Function Format

```
Response = GBT_VIEW(<filename>, [<Arg>], [<show>])
```

Parameter Descriptions

<filename>

Fully specified pathname of an ASCII text file whose contents will be displayed within the edit box.

[<Arg>]

A REXX stem variable defined as follows:

.BUTTON1

Can be the empty string, if only an “OK” button is needed. Anything else (~Yes) will be used as the name of that button.

.BUTTON2

Can be the empty string if a “Cancel” button (along with an “OK”) is needed. Anything else (~No) will be used as the name of a second button.

[<show>]

The show argument is described on page 140.

Example

```
Arg.Button1 = "~OK"
```

```
Response = GBT_VIEW("C:\GENIEPC\CAMFILES\NBSSTD.RPT", "Arg")
```

This example will display the contents of the report file for NBSSTD.CNF in a scrollable, multi-line edit box.

GBT_LOGON

This function brings up a dialog that asks the user to enter a username and a password. If the username/password entry is not found in the password file, the security level and access bits will be set to 0 and an error will be returned. If the operator exits the dialog via the Cancel pushbutton, the security level remains unchanged, and the response will be ">2".

Note that the same user can have more than one password (to be able to log on with different security levels, for instance). Also, the username itself can be sufficient (that is, a password is not required). See "Password Security" on page 136 for a description of the password file.

If the username/password entry is found, the security level assigned to the application dialog window will be changed to the one from the password file for the matching entry. If the password file is a CAM file, the access bits will also be read from that entry. Any following commands (until a GBT_LOGOFF or another GBT_LOGON) will use that level and/or bits when making decisions as to what buttons, fields, etc., are enabled.

The return value from a successful logon will depend on the password file. If it is a CAM file, the return value is the record number of the matching entry; use GETPARAM with that record number to access any value of interest. If it is an ASCII file, the value of the response will be a string which is the username, a space, and the security level.

Function Format

```
Response = GBT_LOGON([<show>])
```

Parameter Descriptions

[<show>]

The show argument is described on page 140.

Example

```
Response = GBT_LOGON( )
```

```
PARSE VAR Response Username Level
```

This example will display a user logon dialog on the screen and parse the return value into username and security strings stored in the REXX variables Username and Level (assuming OK was pressed and an entry was found in the ASCII password file).

GBT_LOGOFF

This function performs a user logoff, which sets the security level and access bits assigned to the application dialog window to 0.

Function Format

```
CALL GBT_LOGOFF
```

Parameter Descriptions

None

Example

```
CALL GBT_LOGOFF
```

GBT_FINDMID

This function brings up a dialog screen which lists all loaded MID files having inputs available, allowing the user to select one. No dialog is displayed if either .FETCH or .COUNT is specified. The value returned by this function is either the name of the MID file or > 2 if canceled. If used with .COUNT, the value returned is the number of items in the list of loaded MID files.

Function Format

```
Response = GBT_FINDMID([<flag>],[<show>])
```

Parameter Descriptions

[<flag>]

The name of a REXX stem variable defined as follows:

.LASTITEM

Names the last enumerated MID file. It is used in Alpha Analyst to add the option "None" to the list of enumerated MID files.

.COUNT

Returns the number of items in the MID file listing. The item specified by .LASTITEM is not included in the count.

.DEVICE

Restricts the enumerated MID files to those containing the specified device. The devices and their codes are:

MCA

0

Fixed Format Functions

ADC	1
Mixer Router	2
Digital Stabilizer	3
Sample Changer	4
Amplifier	5
High Voltage	6
Power Management	11
Vacuum	12

.DEVICETYPE

Restricts the enumerated MID files to those containing at least one input of this type of the specified device (chosen in .DEVICE). Valid device types (in hexadecimal) are:

S100	0100
AccuSpec/A	0300
AccuSpec/B	0400
AccuSpec/NaI	0500
AccuSpec/NaI+	0600
AccuSpec/MC	0700
AIM	0900
InSpector	0A00
AlphaAnalyst	0B00
Manual	1000
Internal	2000
Parallel	3000
ICB	4000
RPI ICB	5000
EtherPlus	6000

In some cases to completely define a device type, values must be OR'ed together.

.FETCH

Index of the file (in the list that GBT_FINDMID would generate, given the same flags) whose name is to be returned by this call (that is, if .FETCH = 3, the third member of the list will be returned). If .FETCH = 0, it has the special meaning described in the Note on page 149.

.DESCRIPTION

Concatenates the description of the MID file with the returned value.

[<show>]

The show argument is described on page 140.

Example

```
Arg.Device=0
Arg.Devicetype=0A00
MID=GBT_FINDMID("Arg")
```

This example will display a dialog that enumerates all MID files containing InSpector MCA detector inputs. The selected MID file becomes the value of the REXX variable MID.

FDS-Driven Functions

The following section describes in detail the operation of those Graphical Batch Tool functions that make use of an FDS file to define the format of the dialog screens presented to the user.

Form Design Specification (FDS) Files

A Form Design Specification (FDS) file is an ASCII file defining the “look-and-feel” of the graphical dialog presented on the screen. Several FDS file examples are presented in Chapter 4, *A Graphical Batch Tools Example*.

The lines of an FDS file are either Comment Lines, Keyword Lines or Parameter Definition Records.

Comment lines

A comment line is any line whose first non-space character is an exclamation point (!). Any text following the ! will be ignored.

Keyword lines

A keyword line is one that begins with one of the following keywords (these are described in more detail in “FDS Keywords” on page 155):

PULLDOWN	LABEL
MENUITEM	BOX
CASCADE	LISTBOX
SELECTION	LISTITEM
QUERY	COMMON
BUTTON	RECORD

Each keyword must be followed by one or more Parameter Definition Records.

Parameter Definition Records

A Parameter Definition Record (PDR) starts with an open brace (`{`), continues with one or more fields separated by semicolons (`;`), and is terminated by a closing brace (`}`). Any number of spaces may be used between fields for clarity; the fields in a PDR may be separated by comment lines or blank lines.

FDS Keywords

The following is a list of the supported keywords with a description of the supported fields to be specified by subsequent PDRs. The notes on page 158 provide additional information concerning the specification and use of the listed fields.

PULLDOWN, MENUITEM, CASCADE

These three keywords are used only by `GBT_DEFMENU`.
The Fields are:

1. Name (see Note 2)
2. Security field (optional, see “Password Security on page 136)

Note that an empty name field (the PDR is simply `{ }`) will define a separator bar.

SELECTION

This keyword is used only by `GBT_SELECT`.
The Fields are:

1. Prefix: 1-9, A-Z, F1-F9 (see note 15)
2. Security field (optional, see “Password Security on page 136)

3. Name
4. Description (optional, see Note 4)

QUERY

Query is used only by GBT_QUERY.
The Fields are:

1. Label
2. Row
3. Starting column position of label
4. Starting column position of edit box
5. Width of edit box (optional)

LISTBOX (see Note 5)

The Fields are:

1. Row
2. Column (see Note 1)
3. Width
4. Depth or number of lines (optional)

BUTTON

The Fields are:

1. ID (see Note 3)
2. Security field (optional, see “Password Security on page 136)
3. Name (see Note 2)
4. Row
5. Column (see Note 1)
6. Width (optional)
7. Filename (see Note 16)
8. Position of the More screen (see Note 19)
9. Record number for the More screen (see Note 19)
10. File selection subsets (see Note 17)

LABEL

The Fields are:

1. Text string
2. Row
3. Column
4. Width (optional)
5. ID (optional, see Note 4)

BOX

The Fields are:

1. Text string
2. Row
3. Column
4. Width
5. Depth

LISTITEM (see Note 5)

The Fields are:

1. CAM parameter (see Note 6)
2. Column
3. Flags (optional, see Note 18)
4. Format (optional, see Note 10)

COMMON, RECORD

The Fields are:

1. CAM parameter (see Note 6)
2. Security field (optional, see “Password Security” on page 136)
3. Label
4. Row
5. Starting column position of label (see Note 9)

6. Starting column position of edit-box
7. Width (optional)
8. Lower-limit (optional, see Note 7)
9. Upper-limit (optional, see Note 7)
10. Flags (optional, see Note 8)
11. Format (optional, see Note 10)
12. Drop down list (see Note 11)
13. Auxiliary CAM parameter (see Note 12)
14. Default value (see Note 14)
15. Refresh or File/Datasource selection subsets (see Note 17)

Notes:

1. The Column field can be empty (that is, 0); this means to center the item horizontally (useful for BUTTON or LISTBOX).
2. The Name field for buttons and menu items should include a tilde (~); the character following the tilde is used as the accelerator key. For menu items, the character following the tilde will be part of the return value from GBT_GETMENU to indicate which item was selected.
3. The button ID field is used as follows:

There are pre-defined IDs that have specific functions (most dialogs use only 1 and 2):

- 7 File selection (see Note 5)
- 6 Datasource selection (see Note 8)
- 5 <reserved>
- 4 More3
- 3 More2
- 2 More1
- 1 Help
- 1 OK
- 2 Cancel

- 3 SaveAs
- 4 Refresh (see Note 8)
- 5 Change record
- 6 Insert record
- 7 Delete record
- 8 Add record
- 9 Clear the record entry fields

For all functions, any button whose IDs have no meaning specific to that function will be handled just like the Cancel button and that ID will be returned (for example, ">12").

For all functions, if the button is going to return an ID (in the form ">1" or ">2"), if that ID is declared in the PDR as "1>nn", for example, then the actual return value will be ">nn", where nn = any number in the range 10 to 999.

For GBT_PARS, the action of each of the pre-defined buttons is as follows:

- **OK** saves all the edited values to the specified datasource, exits the dialog, and returns ">1".
- **Cancel** closes the specified datasource without saving ANY changes, exits the dialog, and returns ">2".
- **SaveAs** saves all the edited values to the specified datasource, exits the dialog, and returns ">3".
- **Refresh** updates those fields that have been flagged with R (see Note 8) and does *not* exit the dialog.
- **Change** reads the values from the RECORD edit boxes, writes them to the specified datasource on the record selected in the list box, and updates the selected line in the list box without exiting the dialog. This button's action will be modified by the K flag, if present (see Note 8).
- **Insert** reads the values from the RECORD edit boxes and uses them to create a new record on the specified datasource; the new record is inserted ahead of the record selected in the list box. The dialog is *not* exited. This button's action will be modified by the K flag, if present (see Note 8).
- **Add** creates a new record at the end of the specified datasource by using values from the RECORD edit boxes; the dialog is *not* exited. This button's action will be modified by the K flag, if present (see Note 8).

- **Delete** removes the currently selected record from the specified datasource and does *not* exit the dialog.
- **Clear** removes the values from all RECORD parameter fields (that is, makes them blank).
- **Help** takes an additional field: the name of an ASCII file (located in the path defined by environment variable GBT_FDSHELPS) containing the help text. If that field is empty, the name of the help file will default to the base name of the current FDS file with the extension .HLP.
- **More1, More2 and More3** take an additional field: the name of the .FDS file containing the specified More screen. If this field is empty or is not the name of an existing file, the name of the FDS file to be used for the More dialog must be supplied by the REXX caller of GBT_PARS. See also Note 19.
- **File Selection** and **Datasource Selection** display a dialog for selecting datasources. These buttons will update those fields flagged with O (see Note 8) with the selected datasource name and does *not* exit the dialog. Field 7 is used to specify an initial filter specification (either hardcoded or *1, *2, etc.) for the selection dialog. Field 10 is used to specify which edit boxes will be updated with the selection value. This field may have a digit 1 through 9; in that case, (see Note 17) only those edit fields specifying this number will be updated.

For GBT_ENUM, the action of the following buttons is:

- **OK** exits the dialog and returns the name of the selected datasource.
- **Refresh** rewrites the list box with the current values from the enumerated datasources.

4. The label ID is used as follows:

- In GBT_SELECT, if there is a label with ID=13, that label location is used to echo the “description” field for a given selection when it is selected (highlighted). Also, if there is a label with ID=11 and the .PROMPT parameter is specified, the value of the .PROMPT parameter will replace that label.
- In GBT_ENUM, if there is a label with ID=11 and the .PROMPT parameter is specified, the value of the .PROMPT parameter will replace that label.
- In GBT_QUERY, if there is a label with ID=11 and the .PROMPT parameter is specified, the value of the .PROMPT parameter will replace that label. Likewise, if there is a label with ID=12, the value of the .LABEL parameter will replace that label.

5. There are a few restrictions on the use of the keywords LISTBOX and LISTITEM:
 - The LISTITEM keyword must be the first keyword encountered after the PDR for LISTBOX.
 - The PDRs for LISTITEM must appear in a “block” (that is, they can’t be separated by another keyword).
 - The PDRs for LISTITEM must appear in left to right order (that is, column numbers must increase).
6. The CAM parameter field is specified as the name of the CAM parameter (for example, ASTIME, GEOMETRY). Additionally, there are some pre-defined parameters that can be specified:
 - For a LISTITEM field used by GBT_ENUM, the following pre-defined parameters can also be specified:
 1. *DS* Base name of the datasource.
 2. *AV* Will show either the string “Available”, if the datasource is available for use, or the string “Unavailable”.
 3. *ID* Will show either the string “Busy” or the string “Idle”, depending on the acquisition state of the datasource.
 4. *ST* Indicates the overall status of the datasource. Will show one of the following strings:
 - Offline** – The datasource has been placed offline.
 - Inaccessible** – The datasource is “owned” by another user or cannot be accessed.
 - In Use** – The datasource is being analyzed.
 - Busy** – The datasource is currently acquiring.
 - Available** – The datasource is accessible and can be used.

For *AV*, *ID*, and *ST*, if there are less than 3 columns provided for this LISTITEM field, only the first character will be shown.

- For a LISTITEM field used by GBT_PARS, this field can have the predefined parameter value *R#*, which indicates that the record number will be displayed. The default format for this field is “%u”.

Note that if sorting is to be done (see the K flag in Note 8), all CAM parameters in a set of RECORD PDRs must be from the same CAM class. In addition, if the record parameters are not from the same class, the Add, Insert and Delete buttons will be disabled.

7. The upper and lower limit fields are used to ensure that no updates will be made to the specified datasource that violate the specified limits. Their use depends on the type of parameter:
 - If a string CAM parameter, the limits represent the minimum and maximum lengths of the string. Note that by default, the upper limit for a string is the maximum defined for that parameter by the CAM file system (refer to the “CAM Files” appendix in the *Genie 2000 Customization Tools Manual* for more information).
 - If a date/time CAM parameter, the limits represent the number of days before (lower) and after (upper) current date/time that will be considered valid.
 - All other CAM parameters use the limits as numeric value boundaries.
8. The flags fields for the COMMON and RECORD keywords are defined as a set of characters as follows:
 - A: The parameter is treated as a logical value, meaning that a radio button is used to display and change its value. This PDR will be one of a set of consecutive PDRs, each representing a different ‘flag’. Each flag is mutually exclusive with the other flags in the set.
 - B: The parameter is treated as a logical value, meaning that a check box will be used to change its value.
 - C: If this parameter has a date-time value, the date will be displayed with a four-digit year.
 - D: Display-only; the edit box or check box can not be edited.
 - F: The initial focus will appear in this edit box/check box. The default is to put the focus on a list box if defined else put the focus on the first defined parameter field.

- I: The value in the edit box will be forced to have an integer value before being written to the specified CAM parameter; it will be displayed as integer also.
- K: If this flag is specified in a Record PDR, the list box will be sorted according to the values of that parameter. Now when that button is pressed: the ADD and INSERT buttons will insert a new record in the proper place; the CHANGE button will maintain the order of the list.
- N: If this parameter has a date-time value, only the date will be shown.
- O: A File/Datasource Selection button may cause this field to be updated (see Note 17).
- R: A Refresh button may cause this field to be updated (see Note 17).
- X: Means that the drop down list field (field 12 under COMMON, RECORD) is to be used as a fully specified pathname; all records from the CAM class defined by the drop down list CAM parameter field (field 13) will be read from that datasource in order to build the drop down list.
9. In the case of a check box, the label appears immediately to the right of the box; the “starting column position of the edit box” field is not used. Note that the label can contain a tilde (~) to provide an accelerator key for toggling the check box.
10. The Format field will be used as a C-language format descriptor for the display of the value in a list box or edit box. The default format is %g for delta date/time and floating point CAM parameters and %lu for integer CAM parameters.
11. The drop down list field can be used in one of three ways:
- It can be a list of choices separated by the “|” character.
 - It can be a “wildcard filter”, indicating that a set of datasources that meet the filter criteria will be used to define the list of choices. This will be interpreted differently if the ‘X’ flag has been used (refer to X in Note 8 for more information).
 - It can be “*1”, “*2”, ..., “*9”, which indicates that the value (filter or list) will be supplied via the function call to GBT_PARS.

12. When the drop down list field (Note 11) is a filter, the auxiliary CAM parameter field can be:
- Empty, meaning that the datasource names established via the filter will themselves form the list of choices.
 - The name of a CAM parameter, whose values will be read from the set of datasources and used to construct the list of choices.

When field 8 contains the R flag (refreshable edit field), this field contains the name of the CAM parameter whose value will be used to update the edit box when the Refresh button is pressed (this CAM parameter will typically be different than the CAM parameter specified for the edit box).

13. If a drop down list is present, the value has to be selected from the list (that is, the lower and upper limits will be ignored).
14. If any value is present in the default value field for any COMMON PDR, it will be used (instead of the value on the specified datasource) as the initial value of the edit box. If the default value field contains one double quote (“”), the default value for that parameter will be an empty string rather than the value from the file.
15. The selection prefix field can have the form “B>second” or “F3>third”. The “>xyz” part will be ignored except when the .GENERIC option is enabled in the call to GBT_SELECT (see the “GENERIC” variable on page 172).
16. The filename field contains a value used by one of the following buttons:
- For a Help button, this is the name of the .HLP file containing the help text. If the filename field is empty, the name of the help file will default to the base name of the current FDS file with the extension .HLP.
 - For a More button, this is the name of the .FDS file containing the specified More screen. If this field is empty or is not the name of an existing file, the name of the FDS file to be used for the More dialog must be supplied by the REXX caller of GBT_PARS.
 - For a Refresh button, this is an optional subset specification (see Note 17).
 - If any File/Datasource Selection button uses field 10 and that field has a digit (1 through 9), each such button can update just a subset of the updatable edit boxes.
17. If any Refresh button uses field 7 and that field has a digit (1 through 9), each such button can refresh just a subset of the refreshable edit boxes. (This is a

specialized use; usually there is only one Refresh button, with an empty field 7. Its function is to refresh *all* flagged fields.)

In this situation, some of the COMMON PDRs which have the R flag set will have a string of digits in field 15; when a given Refresh button is pressed, only those fields that have that button's field 7 digit somewhere in their field 15 string will be refreshed.

For a File/Datasource Selection button, this is the specification of a selection filter. If this field is empty, the filter defaults to "C:*. *".

In this situation, some of the COMMON or RECORD PDRs which have the O flag set will have a string of digits in field 15; when a given button is pressed, only those fields that have that button's field 10 digit somewhere in their field 15 string will be updated.

18. The flags field for the LISTITEM keyword is defined as a set of characters as follows:
 - B: The parameter is treated as a logical value, meaning that it will show either an '*' for True or a space for False.
 - K: The list box will be sorted on this parameter. Note that this will be ignored if any Record PDR has a K flag.
19. Two optional fields are provided for the More buttons. Field 8 is used to locate the dialog in (or outside of) the window of the calling dialog, as described under the .POSITION member of the Show Argument (page 140).

Field 9 can be used to specify the record number to be used for establishing the COMMON values on the More screen. The record number is determined by:

- *R The record selected in the calling screen is used.
- *C The record number is defined by the caller of GBT_PARS using the .RECORD parameter.
- nnn Any number is used as the record number.

If field 9 is empty, *C is assumed.

GBT_DEFMENU

This function instructs the application dialog window to create (or replace) its menu-bar with the items described in the specified FDS file. If successful, the menu-bar will be replaced, but all items on it will be disabled. A call to GBT_GETMENU will be needed in order to enable the menu-bar and allow the user to select an item.

Function Format

```
CALL GBT_DEFMENU <FDS-file>
```

Parameter Descriptions

<FDS-File>

The name of the FDS file that defines the menu-bar items. Note that no path is required, only the base filename.

Example

```
CALL GBT_DEFMENU "menu.fds"
```

Assuming the file "menu.fds" contains:

```
PullDown { ~File}
MenuItem { ~Open}
          { ~Close}
```

the menu-bar of the application window dialog will have a single name File. Clicking on that name (or pressing ALT+F) will bring down a menu with two items, Open and Close.

GBT_PARS

This function provides the means of viewing and editing parameter values stored in CAM datasources. Validity checking of parameter entries as well as complete editing of the specified datasource can be defined by the specified FDS file (refer to "Form Design Specification Files" on page 154).

The value of the response will be the button identifier that exited the dialog (for example, ">1"). Note that if the FDS file has a SaveAs button (identifier ">3"), it is the responsibility of the batch procedure to provide the means of dealing with that (that is, saving the data to another datasource whose name is requested from the user).

If the function was exited by an OK button but no changes were made to the file, the file's date/time stamp will not be changed.

Function Format

Response = GBT_PARS(<FDS-File>, <datasource>,
[<extra>], [<show>])

Parameter Descriptions

<FDS-File>

The name of the FDS file that defines the parameter editing screen. Note that no path is required, only the base filename.

<datasource>

The fully specified pathname of the CAM datasource to be viewed or edited.

[<extra>]

A REXX stem variable defined as follows:

.RECORD	The record number used to read/write values to the COMMON keyword parameters (refer to “FDS Keywords” on page 155 for details on this keyword).
.FILTER1	A datasource filter used for any dynamic drop-down where the value in the FDS file is simply a “*1”.
.FILTERn	Used just like .FILTER1, except that it replaces “*n”, where n can be 2 through 9.
.UPDATE	Used as a flag. If the member exists, the datasource is required to exist, and its (common) parameters are required to be on file. Without this flag, this command will create a <i>new</i> file if the specified datasource is not found.
.VIEW	Used as a flag. If the member exists, the datasource is opened read-only; no changes can be made to the file.
.ASTERISK	Used as a flag. If the member exists, the base name of the datasource will be appended to the title in the title-bar; if any edits have been made, an asterisk will be shown with that name.
.RATE	Takes a value which is the refresh rate of the PARS screen in seconds. Using this option automatically turns on the .VIEW and .UPDATE flags. In this mode, Add, Insert, Delete, Change and Refresh buttons are disabled.

.SELECTION Used as a flag. If this member exists and if the PARS screen has a list box, then the index of the currently selected line is returned to the caller by (any of) the OK button(s) as an append to the ID of the button - in the form ">1 3", for example.

.MULTI Used as a flag. If this member exists and if the PARS screen has a list box, then:

a) The list box is now multi-select.

b) The list of selected items will be appended to the button ID, separated by semicolons.

c) Any Add, Change or Delete buttons will be grayed. Note that this option implies **.Selection**. Note also that it is possible to have *no* items selected.

.MORE1 The name of an FDS file to be used by the More1 button (ID = -2). Note that this will override any filename specified in the PDR for that button.

.MORE2,
.MORE3 Used like More1, but by the More2 button (ID = -3) and More3 button (ID = -4), respectively.

.MODELESS This brings up the dialog as Modeless; it will be refreshed at the rate set by **.RATE**. If there is no **.RATE** argument or if **.RATE=0**, an automatic refresh will not be performed.

The value of **.MODELESS** is a number (1 through 5) that specifies which of five available memory locations will store this dialog. A second dialog brought up using the same location will overwrite the existing dialog.

Using this command automatically turns on the **.VIEW** and **.UPDATE** flags. In this mode, the More, Add, Insert, Delete, Change and Refresh buttons are disabled.

[<show>]

The show argument is described on page 140.

Example

```
Extra.View = 1  
Response = GBT_PARS("pars.fds", "c:\geniepc\camfiles\nbsstd.cnf", "Extra")
```

This example will display a parameter editing screen whose “look-and-feel” is defined by the FDS file pars.fds. The screen will come up in read-only mode.

GBT_ENUM

This function, by use of a datasource “filter”, presents a list of datasources (or parameters contained within a set of datasources) to the user for selection. The return value will be the fully specified pathname of the selected datasource.

Function Format

```
Response = GBT_ENUM(<FDS-File>, <filter>, [<arg>],  
[<show>])
```

Parameter Descriptions

<FDS-File>

The name of the FDS file that defines the list screen. Note that no path is required, only the base filename.

<filter>

A datasource filter used to point to (and restrict) the set of datasources to be listed. It can represent files or detectors (for example, file: C:*.CNF; detector: DET:* or \\node1\DET:*).

If the filter has no wildcards, it is assumed to be the name of a text file which has the following form:

- The first line is the path.
- All other lines contain the base names of files (or detectors) to be listed.

[<arg>]

A REXX stem variable defined as follows:

.PROMPT	Used to replace the FDS file’s label whose ID=11.
.PREFIX	Used to add a 3-column field at the beginning of every list-box line; that field contains a hot key. The first line starts with this character, which must be 1, A or F.

If the character is 1 and there are more than 9 selections, A will follow 9. If the character is F, the prefixes (and the hot keys) will be F1 through F9. If the member exists but its value is invalid, the value defaults to 1.

.AVAIL	Used as a flag. If a member with this name exists, the list of datasources will be restricted to those which can be opened for write-access.
.IDLE	Used as a flag. If a member with this name exists, the list of (detector) datasources will be restricted to those which are not currently acquiring.
.ONLINE	Used as a flag. If a member with this name exists, the list of (detector) datasources will be restricted to those detectors which are <i>not</i> off-line (that is, CAM parameter DETNOTAVAIL is 0).
.BUTTON	Used as a flag with the alias (m>nn) form of button ID. When set, any OK button will return its button number (as >nn) preceding the selected datasource, separated by a space. Note that the .PREFIX option conflicts with this. (The “hot key” has no button number to add.)
.RECORDS	Causes the <Filter> argument to be interpreted as the fully-defined pathname of the datasource containing the records whose values will be used to fill the list box. When using this option, the number of the selected record will be returned as the result (instead of datasource name).
.MULTI	Turns the list box into a ‘multi-selection’ list, so that a set of choices can be made by the operator. A list of the items selected will be returned, separated by semicolons. If .RECORDS is specified, a list of record numbers will be returned, otherwise it will be a list of datasource names (which will <i>not</i> be fully specified).
.DEVICE	Restricts the enumerated datasources to those containing the specified device. The devices and their codes are listed under DEVICE on page 152.
.DEVICETYPE	Restricts the enumerated datasources to those with this type of the specified device (chosen in .DEVICE). Valid device types are listed under DEVICETYPE on page 153.

`.SELECT` The value of this member will be used to select a default item from the list. If the `.RECORDS` option is used, it will be the number of the line to be highlighted. Otherwise, it will be the root portion (no path or `DET:`) of the file name or detector name to be selected.

[<show>]

The show argument is described on page 140.

Example

```
Arg.Avail = 1
Arg.Idle = 1
Arg.Online = 1
Response = GBT_ENUM("list.fds", "DET:*", "Arg")
```

This example will display an enumeration screen which contains a list of detectors for selection which are available, not busy, and are on line (refer to "Form Design Specification Files" on page 154 for details on the required contents of the FDS file to display this list).

GBT_SELECT

This function is an alternative to the `GBT_DEFMENU \ GBT_GETMENU` function calls in that it is a means of presenting a list of options (selections) and allowing one to be chosen from the list.

Unless the `.STRING` or `.GENERIC` member is present (see below), the return value is the prefix of the option selected by the user. It will be either a single character or "Fx" for a function key, depending on the specification in the FDS file.

Function Format

```
Response = GBT_SELECT(<FDS-File>, [<arg>], [<show>])
```

Parameter Descriptions

<FDS-File>

The name of the FDS file that defines the selection dialog screen. Note that no path is required, only the base filename.

[<arg>]

A REXX stem variable defined as follows:

`.PROMPT` Used to replace the FDS file's label whose ID=11.

<code>.PREFIX</code>	Optional override to the line prefixes in the FDS file. It should be either 1, A, or F (see <code>.PREFIX</code> in “GBT_ENUM” on page 169).
<code>.DEFAULT</code>	Prefix of the selection highlighted at startup.
<code>.FILE</code>	Name of a supplemental FDS. It is assumed to contain PDRs similar to those that follow the keyword Selection in an FDS file. Those PDRs will add to the list of choices present in the specified FDS.
<code>.STRING</code>	Used as a flag. If this member is present, the value returned to the batch is <i>not</i> the prefix – it is the full value of the currently selected item in the list box. In this special case, the value of <code>.PREFIX</code> can be “” (the empty string). If it is, the dialog box will show no prefixes (and thus have no list box accelerators).
<code>.GENERIC</code>	Specifies the return of “generic” prefixes. The prefix field in the FDS file should be of the form <code>A>xyz</code> , where A and xyz can be any value except a space, a closing brace, or a quote mark. This means that “> and what follows” is returned to the calling procedure. This variable is mutually exclusive with the <code>.STRING</code> variable; it will be ignored if the <code>.STRING</code> variable is specified.
<code>.BUTTON</code>	Used as a flag. When set, any OK button will return its button number (as <code>>nn</code>) preceding the usual return value, separated by a space.

[<show>]

The show argument is described on page 140.

Example

```
Arg.Prefix = "F"
Response = GBT_SELECT("list.fds", "Arg")
```

This example will display a selection box as defined by the FDS file `list.fds`; the prefixes for the selection entries will be function keys, beginning with F1. For example, if the third item were selected, the return value would be “F3”.

GBT_QUERY

This function presents the user with a query dialog that can be answered by a single response. The value returned is the response string entered (or “>2” if the Cancel pushbutton was pressed).

Function Format

Response = GBT_QUERY(<FDS-File>, [<arg>], [<show>])

Parameter Descriptions

<FDS-File>

The name of the FDS file that defines the query dialog screen. Note that no path is required, only the base filename.

[<arg>]

A REXX stem variable defined as follows:

.PROMPT	Used to replace the FDS file's label whose ID=11.
.LABEL	Used to replace the FDS file's label whose ID=12.
.WIDTH	Number of columns in the edit-box (overrides the default value or value specified in the FDS file).
.INITIAL	Default response value to be displayed in the edit box.
.HELP	The name of a help file that will override the one in the FDS file.

[<show>]

The show argument is described on page 140.

Example

```
Response = GBT_QUERY("query.fds")
```

This example will display a query dialog as defined in the FDS file query.fds and return the response in the REXX variable 'Response'.

GBT_WAIT

This function causes the program to wait until acquisition on the specified datasource is complete. The specified FDS file can specify buttons used to ‘stop’ or ‘abort’ the wait operation. GBT_WAIT treats all Ok and Save As button presses as issuing a ‘stop acquisition’ command, returning the ID so that the user can make a ‘stop’ or ‘abort’ decision. If no button is pressed, acquisition will complete and the string >0 will be returned to the caller. A cancel button will return the usual >2 string without stopping acquisition.

GBT_WAIT always uses a read-only modal screen; the FDS file can use all of the GBT_PARS options, but any Add, Insert, Delete, More... and Change buttons are disabled.

Function Format

```
Response = GBT_WAIT(<FDS-File>, <datasource>,
                   [<extra>], [<show>])
```

Parameter Descriptions

<FDS-File>

The name of the FDS file that defines the dialog screen. Note that no path is required, only the base filename.

<datasource>

The fully specified pathname of the CAM datasource, which must be a detector input in the form DET:<detname>.

[<extra>]

A REXX stem variable defined as follows:

.RECORD	The record number used to read/write values to the COMMON keyword parameters (refer to “FDS Keywords” on page 155 for details on this keyword).
.FILTER1	A datasource filter used for any dynamic drop-down where the value in the FDS file is simply a “*1”.
.FILTERn	Used just like .FILTER1, except that it replaces “*n”, where n can be 2 through 9.
.ASTERISK	Used as a flag. If the member exists, the base name of the datasource will be appended to the title in the title-bar.

FDS-Driven Functions

.RATE	Takes a value which is the refresh rate of the PARS screen in seconds.
.NOACQWAIT	This member is used as a flag. The datasource does not have to be busy (acquiring), and can be a file. In this case, the wait is for any write to the datasource.
.EXTERNAL	Takes a value which is the name of a datasource. Any write to this datasource will cause the wait to complete. This member can be used in conjunction with the .NOACQWAIT member.

[<show>]

The show argument is described on page 140.

Example

```
STARTMCA "DET:S100
```

```
Arg.Rate = 5
```

```
Response = GBT_WAIT ("Busy.FDS", "DET:S100, "Arg")
```

This will wait until collect is done, refreshing the display every five seconds.

Response will be >0 unless acquisition was stopped by the user via one of the buttons described in the FDS file.

4. A Graphical Batch Example

To illustrate how the Genie-2000 Batch Procedures and Graphical Batch Tools (GBT) are used, a sample program¹ called GBTRUTIN.REX has been developed which takes advantage of the routines of Genie-2000's Graphical Batch Tools. It is based on ROUTINE.CMD (page 132) which provides options to count and analyze a sample, count a sample and save the results, or analyze previously saved results.

Two additional procedures, GBFILTER.REX and GBCHANGE.REX, are also provided and are briefly discussed in the following paragraphs. Understanding this discussion of the program's logic and structure will provide you with the concepts you need to develop your own custom applications.

Sample Counting and Filter Analysis

The GBFILTER.REX procedure provides an example of sample counting with minimal user input, such as might be used in a counting lab where many of the same kind of filters are counted in the same geometry and on the same detector.

This procedure requires a geometry calibration file of the form detnameGxx.CNF, where "detname" is an input datasource name (such as DET01), up to six characters, and "xx" is a geometry number.

Data collection is started, then you're asked for the file name to save the data in, the user name, sample ID, type, size, and start and stop date/time. All the rest is performed, based on predetermined parameters, such as detector number, geometry, and so forth.

The analysis is based on a sample analysis sequence file MDA_SHO.ASF.

Sample Changer Command Procedure

The GBCHANGE.REX procedure counts and analyzes a set of samples, such as on an MCA with a sample changer or on an MCA where the same sample is counted multiple times. The reports are produced as each sample is counted.

You'll be prompted for the sample type, sample batch number, date and time, nuclide analysis library to be used, number of samples, then the preset time.

1. Shipped with the Model S561 Genie-2000 Batch Programming Support option.

What You Need to Know

Each sample is collected, then saved in a file whose name is changed for each sample. The sample changer advance signal is given. Then analysis is performed while the next sample is collected, and so forth.

Sample analysis is based on a sample analysis sequence file NID_SHO.ASF.

What You Need to Know

The discussion which follows assumes that you are an experienced Genie-2000 user, have a good working knowledge of Windows and the REXX command language, and are familiar with the material in Chapters 2, *Batch Procedure Reference* and 3, *Graphical Batch Tools*.

Printing the Listings

Because of their size, the listings for the various files which make up the GBTRUTIN program have not been included in this manual. To produce a printed copy to use in conjunction with this discussion, follow this procedure:

1. Launch the WordPad program, which can be found in the Start Menu's Accessories item. Do this by:

Clicking on the Start Menu icon.

Selecting Programs.

Selecting Accessories

Clicking on WordPad.

2. Using the Open command in the File menu, open the file C:\GENIE2K\EXEFILES\GBTRUTIN.REX.
3. Assuming you want to print the file with portrait orientation on standard sized paper, you need to be sure that the document is in 9 point Courier type. This you do by:

Press CTRL+HOME to move the cursor to the top of the document.

Press CTRL+SHIFT+END to select the entire document for editing.

Select the Font command from the Format menu.

Set the Font Name to Courier and the Font Size to 9, then click on OK.

4. Use the Print command in the File menu to print the file. If you're using a laser printer, you must specify the WYSIWYG mode.
5. Repeat steps 2 through 4 for these files as well:

```
C:\GENIE2K\EXEFILES\
  OPRNAME.FDS
  MAIN_MNU.FDS
  DETNAME.FDS
  PRESET.FDS
  PARAMS.FDS
  OUTNAME.FDS
```

Trying the Program

You should try the program to see how it operates before you start looking at its internal structure. To do that you'll need to have your Genie-2000 properly set up and calibrated. Once that's been done you can run the program by typing the following commands in a DOS Command Window:

```
CD \GENIE2K\EXEFILES
REXX GBTRUTIN
```

With the listings in hand you're ready to go on and take a closer look at how the Graphical Batch Tools are used. As you read through this chapter, you'll see references (Part A, Part B) which correspond to comment lines within the listings, allowing you to follow the discussions in this chapter.

As you review the listings that you printed you'll find them full of quotation marks in what may at first appear to be odd places. If you're familiar with REXX, this will come as no surprise; if you're new to the REXX language, keep in mind that they are *very* important in determining exactly how REXX will interpret a statement, and cannot be removed.

The Startup Code (Part A)

Practically speaking, all Graphical Batch Tools (GBT) programs must begin with statements similar to those shown here:

```
"ECHO OFF"
```

The Startup Code (Part A)

Turns off the screen display of the commands as they are executed.

```
"REXXFCTS"
```

Tells the REXX language processor to register for use all of the Genie-2000 REXX and Graphical Batch Tool functions.

```
CALL G2KENVMT
```

Sets up Genie-2000 environment variables based on Genie-2000 registry settings.

```
"SET GBT_FDSFILES=C:\GENIEPC\EXEFILES"
```

This sets up an environment variable specifying where the GBT dialog box Descriptions (called FDS files) can be found.

```
`GBT_INIT /XY=1,-50, ...'
```

This statement opens the application window for our GBT procedure. The arguments are interpreted as follows:

```
/XY=1,-50
```

The screen's display space is always measured from the lower left corner of the display. This argument places the lower left corner of the GBT window at the X coordinate of 1 (pixel); the Y coordinate will be half way up the screen (the 50% point), which is specified as -50 (note that 50 means 50 pixels and -50 means 50%, not negative 50 pixels).

```
/CXCX=-100,-50
```

This parameter establishes the size of the windows as full screen (-100, or 100%) in the X dimension and half of the screen (-50, or 50%) in the Y dimension. The net result of this argument together with the preceding one is to draw a GBT application window which occupies the entire top half of the display.

```
/TITLE=" 'Routine Sample Graphics  
Batch Demo' "
```

This establishes the text that will be displayed in the application window's title bar.

```
/NOSIZE
```

This tells the system to not allow general purpose window resizing by dragging the window borders.

```
IF RC<>0 THEN EXIT
```

Following every call to a GBT command is a test to confirm that all went well. If the RC (Return Code) is not 0, an error has occurred and the program is exited immediately.

The Logon Procedure (Part B)

Even though we're not using the software security system, this demonstration program does request a User Name, and stores that name in any data files that the program may generate. The code in Part B at label `getname:` performs that task using the function `GBT_QUERY`, which asks the operator to enter a single parameter.

```
Opt.Prompt="Please enter your name:"
```

This statement and the two which follow it set up variables for use in passing arguments to the `GBT_QUERY` function. The `.Prompt` variable passes the question to be answered to the dialog box.

```
Show.Title="Routine Count Batch"
```

The `.Title` variable passes the dialog box's title bar text to the `GBT_QUERY` function.

```
Show.Position="ICC"
```

The `.Position` variable determines where the dialog box will be displayed. `ICC` means Inside the application window, Centered horizontally, and Centered vertically on the display.

```
User=GBT_QUERY("OPRNAME.FDS", "Opt", "Show")
```

This statement displays the dialog box and processes the operator's response, which will be returned in the variable `User`. The "stem" variables `Opt` and `Show` are used to pass on the arguments just described. `OPRNAME.FDS` is the name of a file which contains a description of how the dialog box will look and the specific dialog elements it will contain. We'll take a closer look at it in the next section.

```
IF JOBEROR <> 0 THEN SIGNAL Routine_End
```

The Logon Procedure (Part B)

This is the logical equivalent to the IF RC <> 0 statement we saw earlier and is used for the same purpose: to verify that the function executed properly. If it didn't, SIGNAL would jump to the program's end.

```
IF User=">2" THEN SIGNAL Routine_End"
```

As we'll see in the next section, this dialog box includes a Cancel button. If Cancel is pressed, a string with the value of >2 will be returned. That tells the program that the operator does not want to log on, so the program exit is taken.

```
"set JobOpName="User
```

JobOpName is an Environment Variable that is always used to hold the name of the current Genie-2000 User, so that's where we store the name the operator entered.

OPRNAME.FDS Dialog Box Description File

This is the file that tells the GBT how the dialog box requested in the User=GBT_QUERY... statement, above, should look. Referring to Figure 4, which shows the dialog box, and your listing of OPRNAME.FDS, we'll see how the process operates.

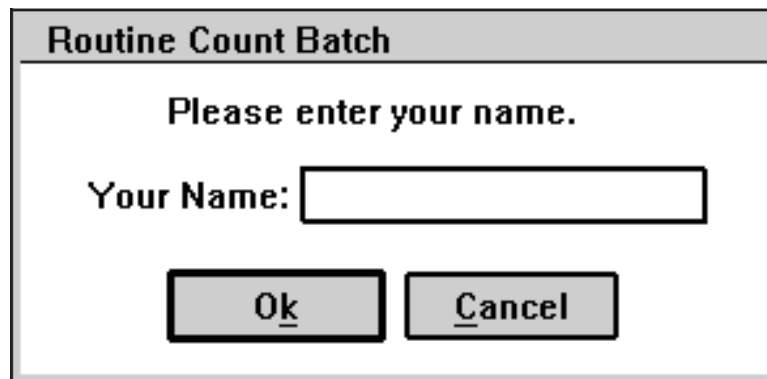


Figure 4 Dialog Generated by OPRNAME.FDS

Since all lines beginning with an exclamation mark (!) are treated as comment lines, OPRNAME.FDS really begins with the LABEL statement.

```
LABEL  
{ xxx; 3; 10; 36; 11 }
```

The first line is the keyword LABEL, which identifies the statement as the one which places a label in a dialog box. The next line, which must begin and end with braces "{...}", contains the details of how the label will be displayed.

xxx: Though these x's aren't needed to make the LABEL function operate correctly, their use is recommended. If you forget to specify a .Prompt argument in your call to the GBT_QUERY function, the string xxx will be displayed where you expected a label to appear. This can be a great aid when you're debugging a large program.

3: All locations within a dialog box are specified in terms of rows and columns of text, where row 1, column 1 is the upper left corner. This argument puts the label on row 3.

10: This argument puts the beginning of the label in the tenth character position of that row.

36: This is the width, in number of characters, of the area to be used for displaying the label. If you use a longer label, the extra characters will be truncated when the dialog box is displayed.

11: The final argument is an ID code that tells the GBT dialog box processor what type of label you are using. The 11 code means "use the Opt.Prompt text as the label".

```
QUERY
{ Your Name;;      5;      5;      19;      25 }
```

The QUERY statement defines the text edit box used to enter the operator's name. The arguments are very similar to those used with the LABEL statement.

Your Name : This is the text that appears immediately to the left of the text edit box to specify what information should be entered.

5: The text edit box is to be on row five of the dialog box.

5: The Your Name: text starts in the fifth character position of that row.

19: The text edit box itself will start in the 19th character position.

25: The text edit box will be 25 characters long.

The Routine Count Menu (Part C)

BUTTON

```
{1;           ;      O~k;           8      10      14 }
{2;           ;      ~Cancel;        8      25      14 }
```

The dialog box's buttons are specified with the `BUTTON` statement. Following the line with the keyword are the button description lines, with each button requiring its own description.

1: This defines the first button to be a button of the OK class. That is, when this button is pressed, regardless of its label, the dialog box will respond as if an OK button had been pressed. This allows you to give the button a more meaningful label (such as Accept, Yes or Continue).

;;: The field following the button class code is used for the Security Level if the application is using the GBT Security System. We aren't, so the field is left empty. Note that an empty field must still be delimited by a semicolon.

If a Security Level is specified, the button will only be enabled for users that have been assigned a Security Level equal to or greater than the level specified here.

O~k: This is the label that will appear on the button. The tilde (~) indicates that the following letter (the k) will be underscored and will be the keyboard's "hot key" which activates the button using Windows' "Alt+hot key" method.

8: The button is to be located on row eight of the dialog box.

10: The button is to begin in character position 10.

14: The button is to be 14 characters wide.

The second button specification line is interpreted similarly, except that it's for a button with an ID of 2, which is the class used for Cancel buttons.

You'll note that in all of the above we never did say how big the dialog box should be. The Graphical Batch Tools are very forgiving in that respect: unless you say otherwise they always make the dialog box big enough to hold everything you've specified to be in it.

The Routine Count Menu (Part C)

After the user has logged on to the system, the code in Part C at label menu: is used to display the main Routine Count Menu and accept the user's choice from that menu.

```
Show.Title=... and Show.Position=...
```

These are used just as in “The Logon Code” on page 151 to specify the desired title bar label and dialog box position for the menu.

```
Function=GBT_SELECT( "MAIN_MNU.FDS" , ,
  "Show" )
```

This statement displays the menu and processes the operator’s response. The file `MAIN_MNU.FDS` defines the contents of the dialog box, and `Show` calls the `Show.title` and `Show.Position` variables. The two commas in the center of the argument list indicate that we are not using the second of the three arguments, a REXX stem variable, which the function can take. Note the use of the `IF JOBERROR <> 0` immediately after the statement to test for proper operation.

```
SELECT
  WHEN Function=n THEN...
```

The operator’s response is used in a `SELECT` statement to move to the appropriate part of the program.

The `MAIN_MNU.FDS` Dialog Box Description File

This file specifies the appearance of the dialog box requested in the `Function=GBT_SELECT...` statement. Referring to 5 and your listing of `MAIN_MNU.FDS`, we’ll see how a selection menu operates.

The first non-comment statement in the file sets up the list box that will be used for displaying the menu.

```
LISTBOX { 3;    3;    54;    4; }
```

The keyword `LISTBOX` identifies the statement as one which places a list box in a dialog box. It is followed by a description of the list box, which must begin and end with braces “`{...}`”. Note that the details of the List Box are on the same line as the key word; this is just as acceptable as using a separate line, since it is the braces, not the beginning and end of a line, that delimit the description.

3: The top of the list box is to be in the third row of the dialog box.

3: The left side will start in the third column.

54: The list box will be 54 characters wide.

The Routine Count Menu (Part C)

4: The list box will be a display “window” with four lines in it; a scroll bar will be provided if more than four selections are to be offered.

```
BUTTON { 1; ; O~k; 9; 24 }
```

Only one button - the OK button - will be used; its setup is similar to the one described in “OPRNAME.FDS Dialog Box Description File” on page 181.

SELECTION

```
{1; 0; Count and analyze sample}  
{2; 0; Count Sample and store in CAMfile}  
{3; 0; Analyze previously stored CAMfile}  
{4; 0; Exit}
```

As you can see in Figure 5, the lines in the SELECTION part of the program establish the list that will be displayed in the list box defined at the beginning of MAIN_MNU.FDS. Using the first line as an example, the entries are specified as follows:

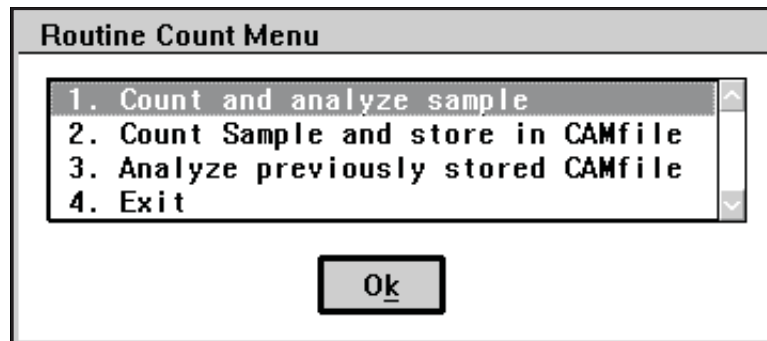


Figure 5 Dialog Generated by MAIN_MNU.FDS

1: This is the keyboard shortcut for the entry. That is, pressing this character on the keyboard is the same as double clicking on the entry.

0: This is the Security Level required for this function; 0 is the same as leaving the field empty, and means that everyone has access. Note that if the level specified here is greater than the current user’s level (e.g. 99 here, and the user was assigned 50), the selection will not be displayed.

Count and...: This is the text to be displayed on this line of the list.

Analyze a File (Part D)

This is the routine performed when menu choice 3 “Analyze previously stored CAMfile” is selected. It determines which file the operator wants to analyze, then jumps to a common analysis section to perform the operation.

Since specifying a datasource is a common Genie-2000 operation, a GBT function named GBT_FINDDDS (Find Data Source) has been designed to do that. We’ll see how it’s used here to generate the dialog box shown in Figure 6.

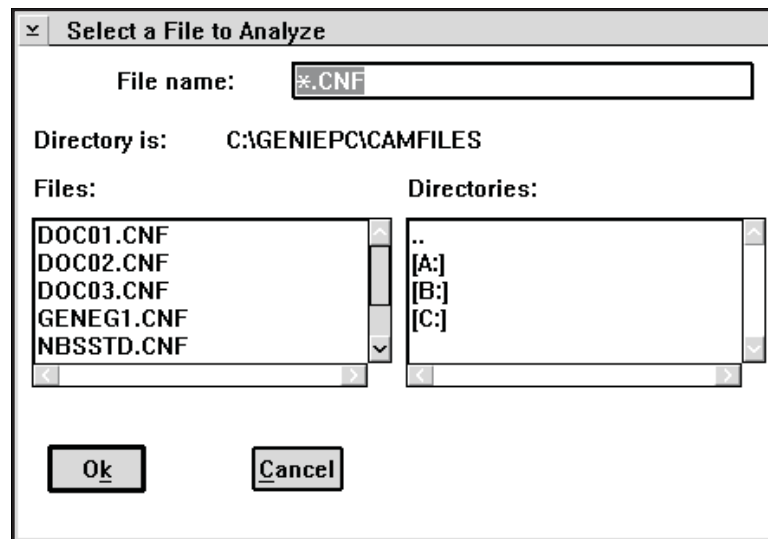


Figure 6 Find Datasource Dialog

Show.Title=... and Show.Position=...

These are used in the same way as those in the previous two examples.

Flag.Files=1

Analyze a File (Part D)

This is a stem variable used by GBT_FINDDS to specify that only CAM files, as opposed to both CAM files and MCAs, should be offered in the choice of datasources to be displayed.

```
infile=GBT_FINDDS("c:\geniepc  
\camfiles\*.cnf", "Flag", "Show")
```

This statement displays the dialog box shown in Figure 6, processes the operator's response to it, and returns that response in the variable `infile`. The arguments are as follows:

"c:\geniepc\camfiles*.cnf": This a file path filter that sets the initial path and file type that will be displayed in the dialog box.

"Flag": Use the stem variable `Flag.Files` to cause only CAM files to be displayed.

"Show": Has the same meaning as in the previous examples.

Assuming the function operated properly (`JOBERROR = 0`), the variable `infile` will contain the operator's response to the dialog box, which will be one of the following:

Exit via "Cancel". If the operator pressed Cancel rather than OK, `infile` will contain the string `>2`, which is a GBT standard response denoting that Cancel was pushed. Note the IF test for this response and the program exit which is taken if it is detected.

Exit via "OK". If the operator pressed OK to exit the dialog box, `infile` will contain the complete path to the selected file, which is then processed by:

```
"set JobInpSrc="infile  
"set JobOutSrc="infile
```

These two statements are used to save the path to the specified CAM file in the Input and Output Job Environment Variables. This is done so we can locate the file to be analyzed by the program's analysis routine (discussed in "Analysis" on page 196.)

```
filename=FILESPEC("name",infile)  
dot=POS(".",filename)  
outname=LEFT(filename,dot-1)
```

These three REXX statements are used to extract just the name portion of the CAM file from complete path contained in the variable `infile`, as follows:

1. The `FILESPEC` function extracts the filename and the file type in the form of `FILENAME.TYP` from the rest of the path and places it in the variable `filename`.
2. The `POS` function is used to locate the period (dot) which separates the filename from the file type.
3. The `LEFT` function extracts just those characters to the left of the period, so `outname` now contains just the filename of the CAM file that is going to be analyzed. This will be used in the analysis routine to label the various operations that are being performed.

Count a Sample

The next part of the program is the logic to count a sample (acquire a spectrum), which is used by both the 'Count and Analyze' and 'Count and Store' selections in the main menu.

Getting the Name of the MCA to be Used (Part E)

The first step in counting a sample is to get the name of the MCA to be used, then verify that it is available, which is done in Part E of the `GBTRUTIN.REX` listing at the label `acquire`:

```
Msg.Button=" "
Msg.Icon="!"
```

During the process of setting up and using an MCA for acquisition there are several opportunities for errors to occur. These two stem variables are used in all of the messages that are displayed when error conditions are detected.

The `.Button` variable is set to a null string. This limits the function `GBT_MESSAGE` to using an OK button as the response to the displayed message. The `.Icon` variable sets the message icon to an exclamation mark. We'll see how both of these are used a little later in the program.

```
Show.Title
Show.Position
Opt.Prompt
```

Count a Sample

These are used as described in previous sections.

```
detname=GBT_QUERY("DETNAME.FDS",  
"Opt","Show")
```

This statement displays the dialog box shown in Figure 7 and process the operator's response to that dialog. Other than the labels which are used, it is identical in operation to the OPRNAME.FDS dialog box described in "OPERNAME.FDS Dialog Box Description File" on page 181.

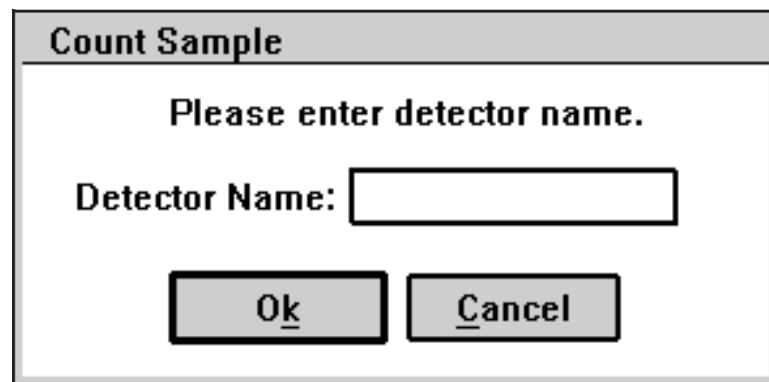


Figure 7 Dialog Generated by DETNAME.FDS

Note that the JOBERROR test for proper operation and the test for a >2 (Cancel) return are both included as they were before.

```
"HDWCHECK DET:"detname  
IF RC=0 THEN SIGNAL start
```

These two statements are used to test for the presence and availability of the specified MCA. A Return Code (RC) of zero indicates that all is well, and the program jumps to the start of the acquisition. If any other value is returned, an error condition was detected, which causes the following statements to be executed:

```
Ans=GBT_MESSAGE("Detector not  
available.", "Msg", "Show")  
IF JOBERROR...  
SIGNAL Acquire
```

The first of these three lines displays the message shown in Figure 8, using the stem variables we set up at the beginning of this section. Assuming successful operation, upon receipt of the operator's response (which is "thrown away" into the variable `Ans`), the program loops back to the beginning of the acquisition logic to allow the operator to try again.



Figure 8 Detector Not Available Message

Opening an MCA Display (Part F)

Having verified that the requested MCA is available for use, the next step is to open up an MCA display window for that MCA, which is done at the label `start:` with the `PUTVIEW` statement.

```
"PUTVIEW DET:"detname" /xy=1,1 /cxcy=-100,-50"
```

This opens standard Genie-2000 MCA display window for the detector the operator specified in "Getting the Name of the MCA to be Used (Part E)" on page 188 and, using the CRT coordinate system, places it in the bottom half of the CRT.

Getting the Geometry File (Part G)

The next step is to get the Geometry File to be used with this detector, which is done in Part G at label `get_geometry:`

Since a Geometry File is really just another CAM file, the procedure and dialog box used here are essentially identical to those discussed in "Analyze a File" on page 186.

```
"set JobInpSrc="..."
"set JobOutSrc="..."
```

Once the Geometry File name has been entered, the Environment Variables `JobInpSrc` and `JobOutSrc` are set up to transfer the geometry (Efficiency Calibration) information to the detector.

```
“MOVEDATA /EFFCAL /PROCESSING
/OVERWRITE”
IF RC>0 THEN DO
  Ans=GBT_MESSAGE(“Error
    during the Eff. Cal. move.”,
    “Msg”,“Show”
  IF JOBERROR <> 0 THEN SIGNAL
    Routine_End
  “ENDVIEW”
  SIGNAL acquire
END
```

These statements move the geometry information into the MCA, replacing any calibration that may already exist. As usual, the Return Code is tested, and a message similar to the one in Figure 8 is displayed if something went wrong. When the message is acknowledged, the system closes the MCA window and goes back to the beginning of the acquisition process.

Requesting and Setting the Preset Time (Part H)

The `GBT_QUERY` function is used again, this time to request the preset live counting time. The various stem variables that are used have been discussed in previous sections, and the `.FDS` file is a minor variation on the others we’ve already seen.

Once a valid preset value has been received (no `JOBERROR` and no Cancel), the `JobInpSrc` Environment Variable is set to the MCA and data acquisition is started with the `STARTMCA` statement.

Note the testing logic following the `STARTMCA`. It’s essentially the same as that used following the `MOVEDATA` statement described above, and is used for the same purpose. The only major difference is that the program jumps back to the main menu display after the error is acknowledged, rather than to the beginning of the acquisition routine.

Getting the Analysis Parameters (Part I)

Once the acquisition is under way the program asks the operator to enter the sample-specific information required to perform a radionuclide assay. The function `GBT_PARS`, which edits a CAM file’s parameters, is used for that purpose.

```
Show.Title=...
Show.Position=...
```

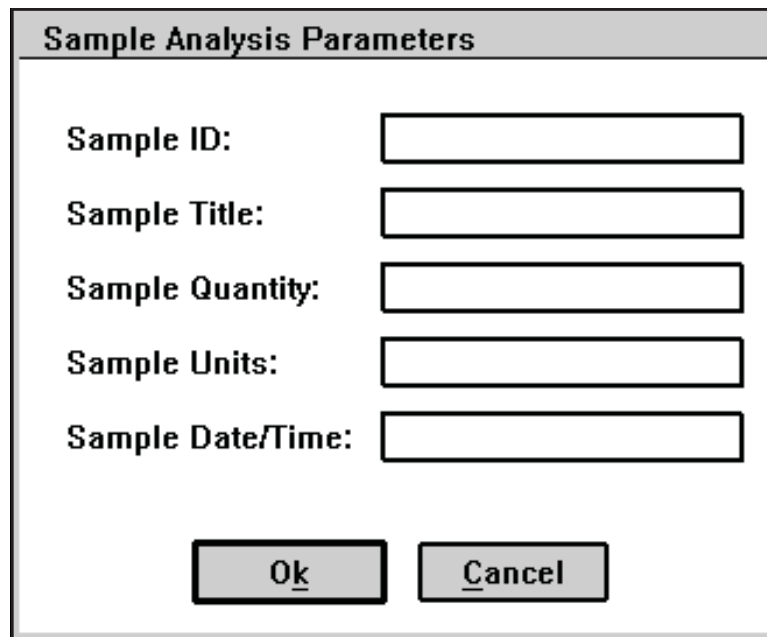
These arguments have the same meaning here as they have in all of the GBT-generated dialog boxes.

```
source="det:"detname
```

This statement concatenates the prefix `det:` with name of the detector in the variable `detname` to yield the complete name of the datasource to be updated.

```
Ans=GBT_PARS("PARAMS.FDS",source,,"Show")  
IF JOBERROR=...  
IF Ans=...
```

This GBT statement, which is followed by the standard response tests, generates the dialog box shown in Figure 9. The file `PARAMS.FDS` determines the look and contents of the dialog box and `source` specifies the datasource that is to be edited. `Show` is used here as it is in all of the other GBT dialog box functions.



The dialog box has a title bar that reads "Sample Analysis Parameters". Inside the box, there are five rows of labels followed by empty text input fields:

- Sample ID:
- Sample Title:
- Sample Quantity:
- Sample Units:
- Sample Date/Time:

At the bottom of the dialog box, there are two buttons: "Ok" and "Cancel".

Figure 9 Dialog Generated by PARAM.FDS

The PARAMS.FDS Dialog Box Description File

The first “executable” statement in the file is the `COMMON`, which tells `GBT_PARS` that you want to edit some of the variables in the “common” section of the datasource specified by the variable `source`.

COMMON

```
{ SIDENT; ; Sample ID;; 4; 4; 27; 25; 1; 16; F }  
{ STITLE ... }
```

Using the first entry as an example, the various items which make up a `COMMON` section entry are interpreted as follows:

`SIDENT`: This is the name of the CAM parameter which is to be edited.

`;`: The next field, the Security Level field, is empty; this program is not using the Security System.

`Sample ID`: This is the label that is to be displayed to the left of the `SIDENT` text edit box.

`4`: The text edit box is to be in row four of the dialog box.

`4`: The label for the text edit box is to start in column four of that row.

`27`: The text edit box itself is to start in column 27 of that row.

`25`: The text edit box is to be 25 columns wide.

`1`: The Sample ID, a mandatory parameter, must contain at least one character. If this field is blank, as in the `STITLE` entry, no value is required for the field. Note that for numeric variables, such as `SQUANT` (the Sample Quantity), this parameter specifies the lower limit of the allowable numeric range, not the minimum number of characters.

`16`: The Sample ID cannot contain more than 16 characters. Note that for numeric variables, such as `SQUANT` (the Sample Quantity), this parameter specifies the upper limit of the allowable numeric range, not the maximum number of characters

`F`: When the dialog box is first displayed, the keyboard focus defaults to this text edit box.

```
BUTTON
{ 1;...}
```

The OK and Cancel buttons are specified in the same way as in the other dialog box .FDS files.

Requesting the Output Filename (Part J)

The final step in the acquisition of the spectrum is to ask the operator for the filename that is to be used for storing it. The code for doing this can be seen at Part J of the GBTRUTIN.REX listing. The dialog box (see Figure 10) and code are generated by the standard GBT_QUERY sequence.

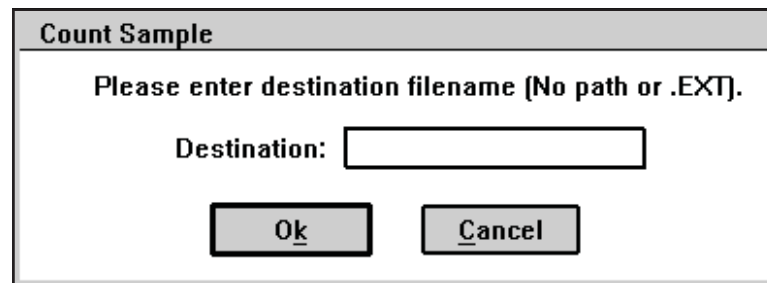


Figure 10 Dialog Generated by OUTNAME.FDS

After successfully receiving a destination filename from the operator, a test is made to see if the file already exists using:

```
"set JobOutSrc=c:\geniepc\camfiles\"outname".cnf"
"MOVEDATA /SAMPLE"
```

This stores the name of the destination file in the Job Output Environment Variable, then uses MOVEDATA to store some "dummy" data in the file.

```
IF RC>0 THEN DO
  Ans=GBT_MESSAGE (...
  IF JOBEROR...
  SIGNAL destname
```

The Return Code is then tested to see if the move was successful. If it was not, the function GBT_MESSAGE displays a message similar to the one in Figure 8 on

page 190. After the operator acknowledges the message and the JOBERROR test is passed, the program jumps back to allow the operator to enter a new filename.

Waiting for the Count to Finish (Part K)

Before the data can be saved, the program must wait for the MCA to reach its preset. While it's waiting, a message is displayed to let the operator know the system is still active. Once the acquisition is complete the data is saved in the output file named in the previous section. The code to do this is:

```
Ans=GBT_MESSAGE("Waiting for count to  
  finish." , , "Show" )  
IF JOBERROR...
```

This pair of statements is used to display and test the operation of the modeless message display shown in Figure 11. Because the middle argument has been omitted, the dialog box has no buttons in it. This means that it will stay visible until the next GBT function is executed.

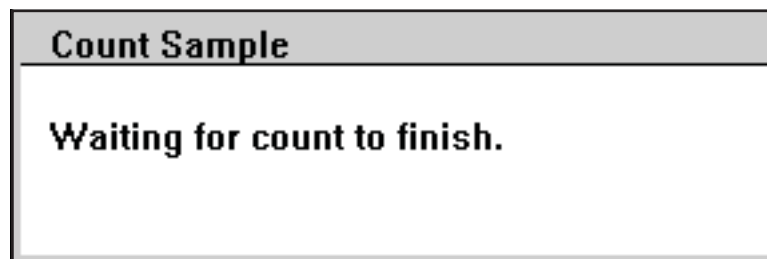


Figure 11 Modeless Message Dialog

```
"WAIT /ACQ"
```

The program will wait until the MCA reaches its preset and acquisition is terminated.

```
Ans=GBT_MESSAGE...  
IF JOBERROR...  
"MOVEDATA /OVERWRITE"
```

A new modeless message is displayed to let the operator know that the count is complete and the data is being saved. MOVEDATA is again used to write to the

destination file. The switch `/OVERWRITE` tells the function to replace the dummy data we used before with the real data from the count.

Store or Analyze (Part L)

Next the program checks to see whether the operator had selected 'Count and Analyze' or 'Count and Store' from the main menu (see "The Routine Count Menu" on page 183). If it was 'Count and Store', we're done; for 'Count and Analyze', we have to go on and do the analysis.

```
IF Function=2 THEN DO
    "ENDVIEW"
    SIGNAL menu
END
"set JobInpSrc=c:\geniepc\camfiles\outname.cnf"
```

Function will have a value of two for 'Count and Store'. If that's the case, `ENDVIEW` closes the MCA display window and the program goes back to the main menu to let the operator select the next function to be performed.

If the `Function` isn't 'Count and Store', it must be 'Count and Analyze', so the Job Input Environment Variable is set to give the analysis logic access to the data just stored.

Analysis

The next section is used for the analysis part of both the 'Analyze a previously stored file' and 'Count and Analyze' functions that were offered in the main menu.

Analyzing the Spectrum File (Part M)

The analysis section starts at the label `analysis:` by displaying a modeless message to let the operator know that the system is busy analyzing the data.

Once it has been verified that the message was displayed properly, the actual analysis is performed:

```
"ANALYZE /SEQ=c:\geniepc\ctlfiles\nid_file.asf"
```

The Environment Variable `JobInpSrc` will be used for the location of the file that is to be analyzed, and analysis sequence will come from the file

Analysis

NID_FILE.ASF. This is the standard Genie-2000 NID Analysis Sequence, with the results routed to an ASCII text file instead of the CRT.

```
Msg.Button=" "  
Msg.Icon="!"
```

These two statements set up the REXX stem variables that will be used to display messages if any problems are detected during the analysis. The first tells the GBT_MESSAGE function to use only the OK button; the second sets the message icon to an exclamation mark.

```
SELECT  
  WHEN RC=24...
```

A SELECT structure is used to test for errors generated by the ANALYZE function. As you can see by the displayed message and by the fact that the program returns to the main menu after displaying the message, RC=24 is an error that prevents any meaningful analysis from being performed.

```
  WHEN RC<>0...
```

Any other non-zero code indicates that the analysis was able to start but a problem occurred during the process. For example, if an Efficiency Calibration hadn't been made, the peak search could be performed but the activity calculations could not be made. In this case the program continues after the message acknowledgment is received.

```
  OTHERWISE NOP
```

As the comment in the listing indicates, REXX requires that an OTHERWISE be included in a SELECT statement to complete the syntax of the structure.

Displaying the Results (Part N)

After the analysis is complete, the next step is to display the generated results. Since we specified that the results should be stored in a file, the report will be in a file with the same name as the datasource that was analyzed (saved in variable `outname`), but with an extension of `.RPT` rather than the `.CNF`. The report display can be seen in Figure 12; the code which generated it follows:

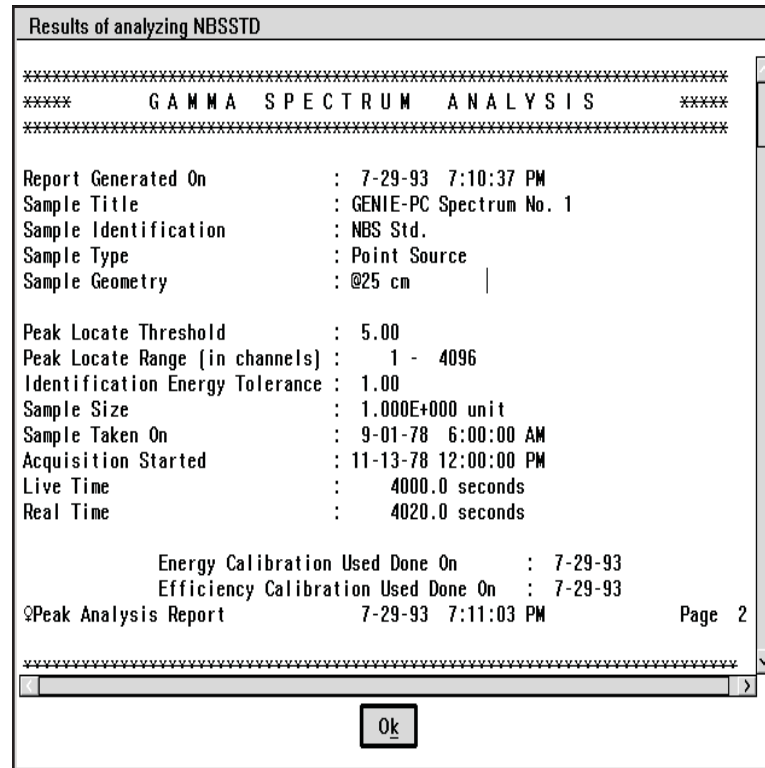


Figure 12 Report Display Window

```
MaxSize=GBT_GETXY()  
IF JOBERROR...
```

This sequence returns the size of the CRT display, in terms of number of columns and number of rows.

```
PARSE VAR MaxSize MaxCols MaxRows
```

The End of the Program (Part O)

The REXX `PARSE VAR` function is used to parse the row and column contents of the variable `MaxSize` into two individual variables, one for the number of columns and the other for the number of rows.

```
ViewShow.MinCols=MaxCols  
ViewShow.MinRows=MaxRows
```

The values are set into these two stem variables so they can be passed to the `GBT` function which will display the report.

```
ViewShow.Title=...
```

This sets up the string that will be displayed in the report window's title bar.

```
Ans=GBT_VIEW( "C:\GENIEPC\REPFILES\  
"outname".RPT" , , "ViewShow" )  
IF JOBERROR...
```

This is the `GBT` function used to display text files. The first argument is the path to the results file and the last argument is used to pass on those CRT size values we just computed, which sets the report window to fill the entire screen.

As indicated by the double commas, the second argument is not used; this means that the report window will contain only an OK button.

```
"ENDVIEW"  
SIGNAL menu
```

When the operator presses OK to close the report display, the MCA window is closed and the program returns to the main menu.

The End of the Program (Part O)

The last piece of the program is the exit code at label `Routine_End:`. If the program had been sent to the exit due to an error condition occurring while it was running, this will display the error codes:

```
IF JOBERROR<>0 THEN DO
  DROP Msg.
  DROP Show.
  Msg.Button1=""
  Show.Title="Routine Sample Batch"
  Ans=GBT_MESSAGE("Program Exit.
    Function Err: " JOBERROR
    "Detailed Err: " ERRCAUSE,"Msg","Show")
END
```

The REXX DROP statements are used to clear out any .Msg and .Show values that may have been left behind when the error was detected. Msg.Button1=" " and Show.Title= are used in the same way as in previous sections.

```
"ENDVIEW"
CALL GBT_TERM
EXIT
```

The MCA window, if it happens to be open, is closed by the ENDVIEW statement. GBT_TERM terminates the Graphical Batch Tools facility. And EXIT tells REXX that we're through.

A. Batch Procedure Errors

In the Batch Procedure Environment, all job commands return an error code, or return status, as part of their normal execution (where a return code of zero signifies successful execution). These codes are listed below with a brief description. Error messages are displayed by these commands in the following form:

Format

<Nn>: Error message [hhhhh]

where:

<Nn>: Return code from job command

Error message: Error message text (as described below)

[hhhhh]: Specific 32-bit error code as described in each section of the “Genie 2000 Error Codes” appendix of the *Genie-2000 Customization Tools Manual*.

Batch Processing Errors

- 1: Input datasource not specified
- 2: Output datasource not specified
- 3: Invalid switch was specified
- 4: Duplicate switch was specified
- 5: Required switch was not specified
- 6: Too many switches were specified
- 7: Required switch value not found
- 8: Invalid switch value specified

- 9:** Conflicting qualifiers to a REXX equivalent job function
- 10:** Unable to communicate with VDM
- 11:** System error: unable to create temporary DSC
- 12:** Specified node not defined as type 'M' in Network Configuration File
- 13:** Remote node not allowed when accessing files
- 14:** Invalid device name specified
- 15:** Unable to locate Network Configuration File
- 16:** Local node not configured to support specified datasource type
- 17:** Error while opening specified input datasource
- 18:** Invalid node name specified
- 19:** Datasource is not a hardware input
- 20:** Datasource is not a file input
- 21:** Unsupported CAM parameter storage type
- 22:** Error during CAM write operation
- 23:** Error during CAM read operation
- 24:** Error while executing internal module
- 25:** MCA window already launched from this job
- 26:** System error: unable to initiate MCA window
- 27:** System error: unable to initiate MS-DOS command session
- 28:** Previous energy calibration is required

Batch Processing Errors

- 29:** Error accessing specified certificate file
- 30:** No data present in specified certificate file
- 31:** Error while calculating energy/shape calibration equations
- 32:** Error accessing analysis module file (AEF) or module DLLs
- 33:** Not enough points available for efficiency calibration
- 34:** Error while calculating efficiency calibration equations
- 35:** Error in opening specified output datasource
- 36:** Cannot 'create' hardware datasource – must already exist
- 37:** Error while moving requested classes of data
- 38:** No. of spectral data channels unequal between datasources
- 39:** Preset type incompatible with specified acquire mode
- 40:** Computational preset stop channel less than start channel
- 41:** Error in issuing control command to hardware
- 42:** Error while querying for hardware status
- 43:** Timeout occurred while waiting for specified event
- 44:** Error while transferring analysis sequence file information
- 45:** Invalid start/end channel specification
- 46:** Error while accessing 'REXX functions' file
- 47:** Error while registering REXX function
- 48:** Specified function not supported

- 49:** Output datasource was not successfully closed
- 50:** Unable to create output datasource
- 51:** Unable to create output CAM file
- 52:** Unable to open input file
- 53:** File conversion error
- 54:** Unsupported file type
- 55:** Read/seek error on input data file
- 56:** Must supply full path for output CAM file
- 57:** Invalid channel count for spectrum file
- 58:** Warning: Data conversion problems – default values supplied
- 59:** Can't use PVOPEN or PVCLOSE while in Show All mode
- 60:** Invalid control string
- 61:** Error opening COM channel
- 62:** Error writing to COM channel
- 63:** Error reading from COM channel
- 64:** Invalid memory size parameter
- 65:** No control file specified
- 66:** No PutView acquisition and analysis window
- 67:** Could not open or clear semaphore
- 68:** Specified datasource is not open

Batch Processing Errors

- 69:** System error: unable to create DSC
- 70:** End date is less than start date
- 71:** Specified date range is not in the Q.A. file
- 72:** Warning: Insufficient data for Q.A. analysis – not all the specified tests were applied
- 73:** No data points to plot in the specified date range
- 74:** Unable to allocate memory for plotting
- 75:** System error: Unable to initiate Q/A plot window
- 76:** Warning: Not all the specified Q.A. parameters were transferred
- 77:** Warning: Points with zero error were excluded from the BIAS test
- 78:** BIAS test cannot be performed, no non-zero error values are present
- 79:** Warning: Standard deviation is 0. Results are questionable.
- 80:** Previous specified calibration is required
- 81:** System error: unable to initiate Plot window
- 82:** No response from the specified ICB address
- 83:** The stem variable does not have expected members
- 84:** No MID file specified
- 85:** Unable to open MID file
- 86:** Unable to connect to Configure topic
- 87:** Runtime Database has an input with the same name
- 88:** Runtime Database has a device with the same address

- 89:** Runtime Database has an input which is acquiring
- 90:** Unable to load MID file to Runtime Database
- 91:** Unable to unload MID file from Runtime Database
- 92:** Specified MID file is not loaded in the Runtime Database
- 93:** Reserved
- 94:** Error communicating with Application Window
- 95:** GB App window already launched from this job
- 96:** Error-return from Application Window
- 97:** Invalid /SCALE value: Must be LINEAR, LOG, or SQRT
- 98:** System error: Unable to initiate IPFIT window
- 99:** System error: Unable to allocate memory for IPFIT window
- 100:** Invalid filter value: Filter requires floating point number
- 101:** Invalid filter combination: /ALL, /ENERGY and /NUCLIDE are exclusive
- 102:** Reserved
- 103:** Reserved
- 104:** Reserved
- 105:** System error: Unable to initiate ASE window
- 106:** System error: Unable to allocate memory for ASE window
- 107:** Required environment variable not found
- 108:** Memory allocation failure

Graphical Batch Error Codes

109:	Another instance exists
110:	Required program unavailable
111:	System error
112:	Reserved
113:	Reserved
114:	Reserved
115:	MCS setup is incomplete or inconsistent
116:	MCS acquisition never started
117:	MCS acquisition aborted – error on detector
118:	MCS acquisition ended early
119:	Warning: Alarm level reached
120:	Warning: Dwell-time exceeded
121:	Detector not part of this MCS acquire

Graphical Batch Error Codes

This section describes the System Level and FDS File Parsing error codes that can be generated by the Graphical Batch Tools.

System Level Errors

As stated in “Function Error Returns” on page 138, system level errors are returned in the REXX variable ERRCAUSE, whenever JOBERROR has the value 96. That error can be any of the 32-bit errors documented in Appendix 1, *Error Codes and Messages*, but will most likely be one of the following error codes (in hexadecimal):

600301: The GBT_PASSWORDS environment variable was not found.

- 600302: No password file was found.
- 600303: The entered username or password was not found in the password file.
- 600305: Cannot open the specified datasource.
- 600306: No menu items are enabled *or* no selection items were enabled *or* no enumerated datasources.
- 600307: Unrecognized FDS keyword.
- 600308: A system error occurred while loading the dialog.
- 600309: Required field was empty.
- 60030A: A system error occurred while accessing a file.
- 60030B: Bad filter specified for GBT_FINDDS.
- 60030C: Cannot connect to a local VDM.
- 60030D: Bad filename entered during GBT_FINDDS.
- 60030E: Unrecognized CAM parameter.
- 60030F: The GBT_PARS FDS file mixed its “record-oriented” classes.
- 600310: An attempt to init a GBT_PARS field from the specified datasource failed.
- 600311: No members could be established for the drop down list.
- 600312: The .FILTER parameter was required and not found (required since ‘*’ was specified for the drop down list field).
- 600313: Incompatible CAM parameter type specified in a drop down list CAM parameter field.
- 600314: A system error occurred during a request for memory.
- 600315: The operator closed the application dialog window from the desktop.

Graphical Batch Error Codes

600320:	The specified FDS file could not be opened.
600321:	The GBT_FDSFILES environment variable was not found.
600322:	FDS parsing errors were logged to a file.
600323:	A keyword required for this command was not found.
600324:	The FDS file contains mixed-function keywords.
600325	There is an inconsistency between the record-oriented parts of the FDS file. Example: there is a RECORD parameter, but no list-box to use to select a record.
600326	There is an inconsistency with the sorting flag. Example: there is an ADD button, but the K flag is not in a RECORD PDR, so there is no correct way to add a new record.

FDS File Parsing Errors

These are the class of errors that are logged in an error log file when interpreting an FDS file (ERRCAUSE equals 600322).

1:	No closing brace.
2:	File-system returned error.
3:	Field not ended by ';' or '}'.
4:	Cannot parse PDR record.
5:	No such field defined, for the current keyword.
6:	Unknown keyword.
7:	Two list-boxes are not supported.
8:	A PDR was seen out of context (no keyword is current).
9:	An FDS line was longer than 255 characters (ignored).
10:	A numeric field could not be interpreted.

Index

A

ADVANCE batch command	68
AECAL batch command	45
AEFCOR batch command	56
AEFFCAL batch command	47
ANALYZE batch command	85
Application interaction	
Batch procedure commands	81
APZERO batch command	76
ARALPHA batch command	53
Area correction	
Batch commands	50
AREA_LIB batch command	35
AREA_NL1 batch command	33
AREACOR batch command	50
ARECAL batch command	46
ARREAGNT batch command	53
ASE batch command	96

B

Batch commands	
QAANALYZ	98
QAMANUAL	98
QAPLOT	100
QAXFER	97
Batch procedure	
Environment variables	22
Example	131
Routine gamma spectrum	132
Sample changer	176
Sample counting	176
Batch procedure commands	
Application interactions	81
Calibration	37
Efficiency Correction	54
Nuclide identification	57
Batch procedure	
Datasource names	23
Editor	21
Error codes	201
Job functions	103
Starting a	11
Batch procedure commands	
Detection limits	60

Edit	65
Hardware controls	68
Options	66
Peak analysis	33
Peak locate	25
Post NID processing	61
Reagent correction	53
Reporting	63
Batch procedure commands	
Area correction	50
Control	23
Miscellaneous	85
Bypass_VDM environ. variable	23

C

Calibration	
Batch procedure commands	37
Calling conventions, REXX	137
CALPLOT batch command	43
CAM parameter security	136
CAM security class, GBT	136
Close connection batch command	107
Coincidence summing P/T ratio	49
Comma as place holder	139
Control job commands	23
COUNTENT batch command	115
COUNTREC batch command	115

D

Data	
DATAPLOT batch command	91
DATAPLOT batch command	91
Datasource	
Batch file names	23
Detection limits	
Batch procedure commands	60
DIGSTAB batch command	76

E

ECAL batch command	37
Editing	
Batch procedure commands	65
Editor, batch procedure	21

EFFCAL batch command	41	GBT_LOGON function.	151
EFFCOR batch command	55	GBT_MESSAGE function	145
Efficiency calibration		GBT_NOTE function.	143
EFFCAL batch command.	41	GBT_PARS function	166
EVALEFEQ batch command.	114	GBT_QUERY function.	173
Efficiency correction		GBT_REPORT function	143
Batch procedure commands	54	GBT_SELECT function	171
EFFCOR batch command.	55	GBT_TERM function	145
ENDVIEW batch command	85	GBT_VIEW function.	150
Energy		GETAREA batch command	108
ECAL batch command.	37	GETDATA batch command	110
EVALENEQ batch command	112	GETINT batch command.	112
Environment variables		GETPARAM batch command	109
Bypass_V	23	Graphical batch tools	
Environment variables		FDS file	154
Batch procedure	22	Keywords	155
Graphical batch tools	135	Graphical batch tools	
JobInpSrc	22	FDS file parsing errors	209
JobOpName	22	Function error returns.	138
JobOutSrc	22	Parameter definition records.	155
Error codes		System level errors	207
Batch procedure	201	Graphical batch tools	
GBT FDS file parsing errors	209	ASCII password file	136
GBT system level errors.	207	Environment variables	135
Graphical batch tools	207	FDS keywords	155
EVALEFEQ batch command.	114	Fixed format functions	142
EVALENEQ batch command	112	Function return values	137
EVALSHEQ batch command	113	Password file format.	136
Example		Security.	136
Batch procedure	131	Show argument definition	140
EXTMCA batch command.	71	Structure of	135

F

FDS	
File, definition of	154
Keywords	155
FILECNVT batch command.	86
Files, FDS.	154
Function equivalents, REXX	104
Function error returns, GBT	138

G

GBT_CLEAR function	144
GBT_COUNTDS function	148
GBT_DEFMENU function.	166
GBT_ENUM function	169
GBT_FINDDS function	147
GBT_GETMENU function.	142
GBT_GETXY function.	146
GBT_LOGOFF function	152

H

Hardware control, batch commands	68
HDWCHECK batch command	74
HVCNTL batch command.	73

I

ICBCOMM batch command	118
Input	
Datasource name env. variable.	22

J

Job functions, description of	103
JobInpSrc environment variable	22
JobOpName environment variable.	22
JobOutSrc environment variable.	22

K	Keywords, FDS	155	Peak batch procedure command	102
			Peak locate batch procedure command	25
L	LOADMID batch command	93	Peak to total batch command	49
			PEAK_DIF batch command	25
M			PEAK_LIB batch command	27
Make direct connection	batch command	106	PEAK_ROI batch command	30
Make VDM connection	batch command	106	PEAK_SLB batch command	29
MCS batch command	STARTMCS	77	PEAK_STD batch command	31
	STOPMCS	80	Place holder, using a comma as a	139
	WAITMCS	80	Post NID processing	
MDA	MDA Currie batch command	60	Batch command	61
	MDA_KTA batch command	61	PTCAL batch command	49
MDAPRESET batch command		116	PUTDATA batch command	111
Modal message, GBT		145	PUTVIEW batch command	81
Modeless message, GBT		145	PVACCESS batch command	82
MOVEDATA batch command		89	PVCLOSE batch command	84
			PVOPEN batch command	83
			PVSELECT batch command	84
			PVSHOALL batch command	85
			PWRMGMT batch command	75
			PWRRESET batch command	75
N			Q	
NID_INTF batch command		59	QAANALYZ batch command	98
NID_STD batch command		58	QAMANUAL batch command	98
NORMAL batch command		66	QAPLOT batch command	100
Nuclide identification	Batch procedure commands	57	QAXFER batch command	97
O			R	
Omitting an optional argument		139	Reagent correction	
Operator name env. variable		22	Batch command	53
Optional	Argument omitted	139	RECAL batch command	40
	Batch procedure commands	66	REFCOR batch command	51
Output datasource name	environment variable	22	Registering REXX function	
			equivalents	104
P			Report	
P/T ratio batch command		49	Batch procedure commands	63
Parameter definition records		155	REPORT batch command	63
PARS batch command		65	Report template commands	
Password security, GBT		136	See Template commands	
PDR	See Parameter definition records		REXX function calling conventions	137
Peak area analysis batch command		33	REXX function equivalents	21,104
			REXX tutorial file	4
			REXXFACTS batch command	95
			Routine GSA batch command	132
			S	
			Sample batch procedure	131
			Sample changer	

Batch procedure command	176	T	
Sample counting batch command	176		TAKEOVER batch command 74
Security			TENTIVE batch command. 57
ASCII file for GBT.	136		Tutorial file, REXX 4
CAM parameter for GBT	136	U	
Show argument, GBT.	140		ULOADMID batch command 94
Smooth, batch command	67		Using
Starting a batch procedure	11		REXX function equivalents 104
STARTMCA batch command	69	W	
STARTMCS batch command	77		WAIT batch command. 92
STOPMCA batch command	70		WAITMCS batch command 80
STOPMCS batch command	80		
STRIP batch command.	67		

Warranty

Canberra's product warranty covers hardware and software shipped to customers within the United States. For hardware and software shipped outside the United States, a similar warranty is provided by Canberra's local representative.

DOMESTIC WARRANTY

Canberra (we, us, our) warrants to the customer (you, your) that equipment manufactured by us shall be free from defects in materials and workmanship under normal use for a period of one (1) year from the date of shipment.

We warrant proper operation of our software only when used with software and hardware supplied by us and warrant that our software media shall be free from defects for a period of 90 days from the date of shipment.

If defects are discovered within 90 days of receipt of an order, we will pay for shipping costs incurred in connection with the return of the equipment. If defects are discovered after the first 90 days, all shipping, insurance and other costs shall be borne by you.

LIMITATIONS

EXCEPT AS SET FORTH HEREIN, NO OTHER WARRANTIES, WHETHER STATUTORY, WRITTEN, ORAL, EXPRESSED, IMPLIED (INCLUDING WITHOUT LIMITATION, THE WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE) OR OTHERWISE, SHALL APPLY. IN NO EVENT SHALL CANBERRA HAVE ANY LIABILITY FOR ANY SPECIAL, INDIRECT OR CONSEQUENTIAL LOSSES OR DAMAGES OF ANY NATURE WHATSOEVER, WHETHER AS A RESULT OF BREACH OF CONTRACT, TORT LIABILITY (INCLUDING NEGLIGENCE), STRICT LIABILITY OR OTHERWISE.

EXCLUSIONS

Our warranty does not cover damage to equipment which has been altered or modified without our written permission or damage which has been caused by abuse, misuse, accident or unusual physical or electrical stress, as determined by our Service Personnel.

We are under no obligation to provide warranty service if adjustment or repair is required because of damage caused by other than ordinary use or if the equipment is serviced or repaired, or if an attempt is made to service or repair the equipment, by other than our personnel without our prior approval.

Our warranty does not cover detector damage due to neutrons or heavy charged particles. Failure of beryllium, carbon composite, or polymer windows or of windowless detectors caused by physical or chemical damage from the environment is not covered by warranty.

We are not responsible for damage sustained in transit. You should examine shipments upon receipt for evidence of damage caused in transit. If damage is found, notify us and the carrier immediately. Keep all packages, materials and documents, including the freight bill, invoice and packing list.

Software License

When purchasing our software, you have purchased a license to use the software, not the software itself. Because title to the software remains with us, you may not sell, distribute or otherwise transfer the software. This license allows you to use the software on only one computer at a time. You must get our written permission for any exception to this limited license.

BACKUP COPIES

Our software is protected by United States Copyright Law and by International Copyright Treaties. You have our express permission to make one archival copy of the software for backup protection. You may not copy our software or any part of it for any other purpose.