SUPER HELP

Robert Fleming, ATT Bell Labs

Abstract

The super program was written by Robert Fleming from ATT Bell Labs. This document was derived from the HTML page conversion of the man pages that he distributes with his program. Enclosed is an excellent introduction as well as a section on basics.

Contents

1 INTRODUCTION

PROGRAM DESCRIPTION

SUPER is an interactive program designed to operate an x-ray diffractometer. The program is designed to be the opposite of "menu-driven", i.e. it is designed to perform a series of arbitrarily defined and sequenced tasks with minimum operator intervention and with a minimum number of questions The program is modular with little "intelligence" or decision making capability. In most cases, commands perform simple tasks with the provision for the user to create "alias" commands to perform more elaborate customized tasks. The program is expected to avoid disasters by checking motor limits before movement, checking scan lists before execution, and not allowing nonsense variable assignments that might cause an arithmetic overflow (e.g. lattice parameter = 0).

SUPER performs tasks by executing commands that can have several arguments. For example, "br 0 0 2" will move the diffractometer to the (002) reciprocal lattice point. SUPER also recognizes certain variables. For example, "as = 1.64" changes the reciprocal lattice parameter a-star to 1.64. The current value of the variable is displayed if the value is omitted. Commands and variable assignments can be listed in a file and executed as a block. In addition the user can create new commands by defining an "alias" that associates a user-defined name with a collection of commands.

HISTORY

SUPER was first written in FORTRAN in 1978 by Robert Fleming to operate a two-circle diffractometer from a PDP 11/03. It was subsequently ported to C in 1982 by Robert Fleming and Rick Lerner and now runs in a variety of Unix environments (V7, BSD, SYS V, Venix, Sun SPARC). Most recently SUPER has been ported to a PC running MS/DOS (with a Microsoft C-complier) by Mark Filipski. The operating environment (type of Unix, DOS, etc) can be specified when the progam is compiled.

The original version of SUPER operated a two-circle diffractometer The C version added multiple calculation modes and a four-circle capability. Surface diffraction features were first added by Ian Robinson in 1987 (the "fixed-azimuth" mode) and refined in 1989 by Elias Vlieg (the "5-circle" mode). Current changes in progress include features to allow the program to be used to collect crystallographic data.

HARDWARE

SUPER presently requires an interface to a Camac crate for motors and scalers. Future versions may incorporate a PC-based motor/scaler interface.

1. Computer with Camac interface. Operating system can be either Unix or MS/DOS. (DOS interface is DSP model 6001/6002 - PC004)
2. Camac crate.
3. DSP E-500 stepping motor controller (camac).
4. DSP QS-450 or Kinetic Systems 3610 scaler (camac).
5. DSP RTC-018 real time clock (camac).
6. Camac clock generator (Joerger model CG or equivalent).

INVOKING SUPER

To run the program you must have a CAMAC interface and if a printer file is designated in the file optins you must have that interface also.

To run SUPER from the Unix shell, first change directories to a directory that will contain the data output. To run the program with the default I/O files, type "super" after the shell prompt. To end the program, type "end".

SUPER saves the program parameters when the program ends. By convention, the program parameter file is called "filename˙vars" where filename is a user-supplied identifier. To create a local copy of the program parameters, either copy and existing "vars" file or type "touch filename˙vars". (This creates an empty file by that name). Now run the program with the command "super filename˙vars." When super ends or when the "save" command is executed, the program will save a local copy of the program parameters. See the help file SUPER˙IO(SUMMARY) for more information.

2 BASIC INFORMATION

SUPER is an interactive program originally designed to operate an x-ray diffractometer, but is easily modified for general experimental control. SUPER expects commands which can be followed on the same line by arguments. Commands and arguments are separated by spaces or commas. Some commands will prompt the user for the required arguments if the correct number of arguments is not present.

SUPER also recognizes variable names. To change the value of a variable, type the name followed by "=" and the new value. To check the value of the variable, omit the argument, e.g. "as =" will print the current value of the variable as. For integer variables that are flags, e.g. iflag, you can substitute "on" or "off" or "yes" or "no" for the integer argument. This allows one to use syntax like iflag = on to set flags.

Several commands may be placed on one line if they are separated by a semi-colon. For example "mv 10 5 0 0; ct" moves the motors to a given setting and then counts.

The number of letters necessary to specify a command in SUPER is given by the variable PRECISION (currently=3) located in options.h. Characters in excess of the value given in PRECISION are ignored.

LIMITS

There are both hardware and software motor limits. A hardware limit causes the affected motor to instantly stop. This will usually result in lost pulses and the motor will have to be reinitialized. Software limits are set in the program and are designed to keep a motor from hitting a hardware limit. The delete key causes the motors to ramp down and stop in the usual way without losing pulses.

To print the software motor limits, use the lm command. To set new limits for a motor type "lm *mnum high low*" where *mnum* is the motor name or number and *high/low* are the values of the limits.

HELP

To list all allowed commands and variables, type "help". To obtain information about a specific command or variable, include the name as an argument to help, e.g. "help ct" will print information about the command ct.

BASIC COMMANDS

mv - move motors (Four arguments)

ct - count (One optional argument = preset)

br - go to position defined by H, K and L (Three arguments).

wh - print current H, K and L and values of the motor angles (No args).

ca - calculate motor angles from H, K and L (Three arguments). 208 ci - calculate H, K and L from motor angles (Four arguments).

asc - angle scan: Scan four diffractometer motors.
msc - mscan scan: Scan any motor (up to three at once).
isc - integer scan - Scan from initial HKL in units of delta HKL.
rsc - rocking scan: A symmetric scan centered about a given HKL. Scan a single Miller index, H, K, L or a single motor.
lup - line-up scan: A symmetric scan centered about the present position, one motor only.
fpk line-up scan: Like lup but the command does not ask for a heading and does not store the data on disk.

BASIC VARIABLES

as - Lattice parameter a*
bs - Lattice parameter b*
cs - Lattice parameter c*
al - Interfacial angle alfa*
be - Interfacial angle beta*
ga - Interfacial angle gamma*
h1, k1, l1, h2, k2, ... HKL's of orientation reflections
w1, p1, c1, w2, p2, ... Omega, Phi, Chi of orientation reflections
wv - incident wave-vector (2*PI/lambda or 1/lambda)

CALCULATION MODES

The direction of the scattering vector is given by three angles OMEGA, PHI, and CHI. Since only two angles are required to specify a direction, the direction of q is over-determined. This degeneracy may be lifted by holding one of the three angles constant. SUPER is designed to calculate the direction of the scattering vector with either OMEGA, PHI, or CHI constant. The command frz can be used to specify the calculation mode as well as the variable "fl" (freeze lattice) which indicates whether the lattice parameters are known or not.

Calculation modes:
OMEGA or PHI = constant
CHI = constant = 90 degrees
2-circle (PHI and CHI = constant)
5-circle (Surface diffraction)

ABORT

To abort any activity (moving motors, counting entering scans, etc), press the key which will give an interrupt signal. This may be "rubout", "delete" or "cntrl-c", depending on the system. An interrupt will return control to the monitor as denoted by the prompt. A "scan", e.g. iscan, rscan, mscan, vscan, pkup, lup or any command command list started by the sc command, may be restarted by using the co (continue) command.

END

To terminate the program and return to the UNIX shell, type end.

ALIAS COMMANDS

The program SUPER is designed to have simple, modular commands that can be combined to make more complex commands by the user. Alias commands can be defined with the commands ea (edit alias) or da (define alias). For example, you might define a command called *mount* to move the diffractometer to a special position, say TTH=TH=40. An alias can be defined so that when the user types mount the program responds as if "mv 40 40" had been typed.

CAMAC

The motor and scalar parameters are displayed with the mpar and the spar commands. The parameters can be changed with the mset and the sset commands (normally placed in the file /usr/super/lib/super˙init). An easier way to change motor and scalar parameters, however, is with the ec command (edit camac), a command that runs the unix editor on the file /usr/super/lib/super˙init. The ec command can also be used to make any variable assume a default value at run-time by placing in the "init" file. For example you would force the program to make the preset counter (pc) have a CAMAC slot number of 5 by putting "pc=5" in /usr/super/lib/super˙init.

SEE ALSO

FILES(SUMMARY), MOVE(SUMMARY), SUPER˙IO(SUMMARY)

3 MORE GENERAL INFO

HELP ON HELP

"Help" gives information about specific commands and variables, or more general information about subjects. Type "help" to invoke the routine. Then enter one of the following:

1. "all" to get a list of all material available.
2. "cmd", "var" or "subj" to obtain a list of the available commands, variables or subjects respectively.
3. Command or variable name name (lower case) or the subject name (UPPER CASE) to obtain information about a particular command or subject.

SUMMARIES

Summaries of specific subjects are denoted by a name that is ALL˙CAPS.

COMMANDS

A command usually performs some action such as moving motors or setting the temperature. A command may have variables which follow it on the same line separated by spaces.

VARIABLES

A variable is a parameter used by the program and set with a "name=value" syntax. Variables are saved by the program so that their value is retained from one run to the next.

4 MOVING

BASIC MOTOR MOVEMENT COMMANDS

mv Move four diffractometer motors to an absolute position - This command expects four arguments: Two Theta, Theta, Phi, Chi

gmv Move all non-inhibited motors - The motors moved will depend on the settings of the inhibit flags.

an Angle - This command moves the Two Theta and Theta motors only. It expects two arguments: Two Theta, Theta.

pl Plane - This command sets the diffraction plane by moving only the Phi and Chi motors. It expects two args: Phi, Chi.

tt Theta/Two Theta - This command sends the two-theta motor to the given angle and the theta to half the given angle plus the omega offset given by the frz command. There is one argument, the two-theta target. motor Motor - This command moves a single motor to a given angle. It expects two arguments, motor number and target angle. Motor numbering convention: 1-Two Theta, 2-Theta, 3-Phi, 4-Chi. A third optional argument may be used to specify a motor speed.

dmotor Delta-motor - Same as motor command, but the position is a relative position measured from the present position. E.g. *dmot tth -0.1* means move the tth motor -0.1 degrees from the present position.

br Bragg - This command moves to the angles defined by H, K and L. The routine expects three arguments (H, K, and L) with an optional fourth argument specifying the preset value (same format as TI). If the preset is given, the routine counts after moving.

MOTOR POSITION READOUT

The wh command will print out motor positions that have been designated by the format command. The look command prints out the positions of all the motors. All of the "move" commands will print out continuously updated motor positions until the target is reached. To suppress this repetitive print-out on a line printer type "video = 0".

MOTOR INHIBIT FLAGS

The movement of a motor can be prevented by either making the variable nmot less than the motor number (this effectively reduces the number of motors that the program will control) or by setting the "inhibit" flag with the minh command. E.g. to prevent the CHI and PHI motors from being moved type "minh phi chi on".

SEE ALSO

BASICS(SUMMARY)

5 ENERGY AND ENERGY RELATED VARIABLES

efix Energy Fixed (variable) - The program will run with fixed incident energy (efix=1) or variable energy calculated from the theta-angle of the monochromator (efix=0).

mds Monochromator d-star (variable) - 2 PI / d-spacing of monochromator.

cme Calculate Monochromator Energy - (command) Gives current monochromator energy (no argument) or calculates energy using the argument as the monochromator theta. Use "p" as a final argument to force output to line-printer.

cma Calculate Monochromator Angle - (command) Gives monochromator theta calculated using the argument as the energy in keV. Use "p" as a final argument to force output to line-printer.

mme Move Monochromator Energy - (command) Expects the energy in keV as an argument. Moves the monochromator theta.

6 FILES

To list all files currently used by SUPER use the files command.

SUPER LIBRARY FILES

SUPER keeps a number of files in a special directory defined options.h. This manual assumes that directory is called */usr/super/lib*.

PROGRAM FILES

Two data files are necessary to run SUPER, "super˙vars" and "super˙init", both located in */usr/super/lib*. The file "super˙vars" contains a list of all variable assignments used by SUPER. This file is automatically updated when SUPER terminates. The user should not modify "super˙vars" as it is re-written by the program. For new systems the user may create an empty super˙vars file. The user can specify a "super˙vars" file on the command line when SUPER is executed or via the restart command from within super.

The read-only file "super˙init" may contain variables or commands. Because "super˙init" is read after "super˙vars", any variable assignments which are in "super˙init" will take precedence. In contrast to the "super˙vars" file, "super˙init" is never re-written by the program. These variable assignments will not be updated when SUPER terminates. Both files must contain "end" as the last command. The file "super˙init" contains commands which are usually not changed from experiment to experiment such as "init", "gs", motor speed settings and Camac slot assignments.

SUPER may be run with files other than the default ones. To execute SUPER with other filenames type: super [*filename˙var filename˙init*]

The "vars" filename may be specified without the "init" filename. This is a good way to keep lattice parameters and scans separate for separate experiments.

Normally, if there is an error message during program start-up (one of the commands was incorrect), you will never see it. A more perverse problem is that one of the commands will cause SUPER to hang. The user can add a third argument (anything will work), and the program will print each command and the response as it is executed. Type: super [*filename˙vars filename˙init* [*verbose˙flag*]

SCAN/ALIAS FILES

The command list defined with the ds or the es command is stored in a file with the default name "super˙scans". A different file may be read from SUPER with the gs (get scans) command. Similarly, the alias definitions are stored in the file "super˙alias". A different file may be read from SUPER with the ga (get alias) command.

The data output from SUPER are stored in the files specified by the fi command. SUPER always appends data to a file if it already exists. Three data output files are allowed, DATA (disk file), LINE-PRINTER (device file), and TTY (stdout). Any output file may be set to "null" if no output is desired. To list the current output files type fi with no argument.

7 FLAGS

FLAG SYNTAX

A "flag" is a special variable that controls options by being either zero or one. The flag is "on" when equal to one and "off" when equal to zero. As with a regular variable able, the status of a flag can be interrogated by typing the flag name followed by "=".

The following are all examples of legal syntax:
flag =
flag = 0
flag = 1
flag = on
flag = off

FLAGS in SUPER

a˙scale	auto-scale flag for plot
afs	flag to control automatic file numbering
aplot	automatic plotting flag
box	flag to put a box around the points on the plot
cryst	flag for crystallographic options
efix	Constant/Variable energy flag
enq	enquire flag for scan heading
iflag	flag for "intg" (integrate) mode
t˙files	output flag for cryostat temperature
video	flag to denote a crt terminal

8 HARDWARE

CAMAC MODULES

1. Camac crate controller and software interface.
2. DSP (Standard Engineering) E-500 stepping motor control module (at least one).
3. DSP (Standard Engineering) RTC-018 real-time clock / preset counter.
4. DSP (Standard Engineering) QS-450 quad scaler or equivalent (ttl outputs, at least one)
5. Clock module - Joerger model CG (clock pulses for preset counter)

COMPUTER

SUPER has been ported to a PDP-11 (V7 or BSD Unix), a Sun SPARC station and a MS-DOS PC. Set the type of computer in options.h.

9 INTEGRATION PROTOCOL

Two modes of counting exist, the normal mode and the "intg" mode. In the normal mode (iflag = 0) the program counts for a time given by the monitor. In the "intg" mode (iflag = 1) the program counts while moving a designated motor. Background counts before and after moving the motor may also be obtained (if the monitor time is non-zero).

Several commands and variables are used to set up and control the "intg" mode:

iflag (variable) - Set iflag=1 (or "on") for "intg", 0 (or "off") for normal operation. Set iflag=2 for "flip-flop" mode where the motor alternates direction.

intg (command) - Set up parameters to designate motor and speed

iset (command) - Set up names of intg headings

ipar (command) - Print all heading names and parameters (as defined by iset).

ifor (command) - Set up format for data output

intg (INTEGRATION PARAMETERS):

Set up "intg" parameters. INTG: args = mot˙num start stop time iflag The mot˙num may be given as a name (e.g. phi) or a number. The start and stop are given in degrees from the present position. The time is the time in seconds for the motor movement (while counting). The last argument, iflag, is optional.

iset i1set i2set (INTG Scaler Set):

These work like sset, but only the scalar headings used in the INTG mode are set. The argument list, which may be obtained by typing "iset" is: ISET: Args = S#, LABEL. (The other parameters, CHAN, and SCALE are set by the "sset" command). Three sets of scaler parameters are defined by iset, the scaler channels used for "bkg1", the scaler channels used for "bkg2" and the scaler channels used for integration ("count-while-slewing"). See intg.cmd and iflag.var.

10 ORIENTATION

Sample orientation is done in one of two ways depending on whether the lattice parameters are known or unknown.

LATTICE PARAMETERS KNOWN

The six lattice parameters are entered manually using the name=value syntax. The names of the six reciprocal lattice constants are: as, bs, cs, al, be and ga. Two orientation reflections are required with em hkl, em th, *phi* and *chi* specified for each reflection. These may be entered manually using the variable names: *h_ik_il_i*, where i is an integer in the range 1-nobs.

The orientation reflections may also be entered using the or (orient) command. To enter the parameters with the or command type or *r#¡cr¿* where *r#* is the number of the reflection (in this case 1 or 2). One may also manually enter all variables on one line by typing or r# h k l tth th phi chi¡cr¿.

The orientation matrix has nine degrees of freedom. Six degrees of freedom are supplied by the lattice parameters, two more come from the first orientation reflection (the two polar angles) and the final degree of freedom comes from the second orientation reflection (the second reflection specifies the azimuthal rotation about the interfacial angle). The angles of the first orientation reflection, called the primary reflection, will always be reproduced exactly. The angles of the second reflection may not necessarily be consistent with the lattice parameters, and consequently the program may not reproduce the angles of the secondary reflection exactly.

LATTICE PARAMETERS UNKNOWN

The six lattice parameters are determined from the four angles specified for each of the orientation reflections (two-theta, omega, phi, chi). In addition to the variable names listed above, two-theta must be entered for each reflection (variable names = t1, t2, t3, ...).

Lattice parameters which are consistent with the three observed reflections will be calculated. In general, the lattice parameters will correspond to a triclinic cell with interfacial angles nearly equal to those expected from the symmetry. The orientation parameters may be entered manually or by using the or command.

INITIALIZING THE ORIENTATION MATRIX

In both modes an orientation matrix is calculated with the init command. (The routine init is called automatically each time a lattice parameter or an orientation parameter is changed.)

The command om (orientation mode switch) switches between the lattice parameters known mode and the lattice parameters unknown mode.

The command gor (get orientation reflection) drives the motors to the position recorded for the given orientation reflection.

The orientation matrix uses the pairs of variables, tthR / q and thR / omegaR interchangably. The orietation matrix always prints "tth" and "th" but the user can enter q or omega instead and the program will convert. If the incident wave-vector is changed, the two-thetas in the orientation matrix will also changed, i.e. q is preserved.

SEE ALSO

t1(var)

REFERENCE

W.R. Busing and H.R. Levy, Acta. Crys. 22, 457 (1967).

11 PLOTTING

GRAPHICS FEATURES OF SUPER (Unix only)

Simple plotting of the current scan on a 4014 compatible terminal can be done. XMIN is set to zero and XMAX is set to the number of points in the scan. YMIN is zero and YMAX may either be set from the keyboard using the "YMAX = value" syntax, or else the program will calculate a convenient value for YMAX.

More extensive plotting as well as curve fitting may be done with the program PLOT. PLOT may be executing from SUPER by using the shell escape command ("!").

PLOT COMMANDS (Unix only)

erase Erase plot

rep Re-plot

lin Plot the data on a linear scale

log Plot the data on a log scale

lsc Enter the ymin and ymax for the log scale (two arguments(.

PLOT VARIABLES (Unix only)

a˙plot Auto-Plot Flag (variable): Type "a˙plot = 1" to implement plotting of the current scan on terminal screen.

a˙scale Auto-Scale Flag (variable): Type "a˙scale = 1" to implement automatic scaling of Y-axis for plot.

box "Box" flag (variable): Type "box = 1" to plot boxes around the data points. Type "box = 0" to plot lines only (this is faster).

ymax Maximum value of Y-axis (variable) - Set "ymax" manually if "a˙scale" = 0.

xbot Variable to define the terminal plotting area (in plotter units). The maximum size of the plotting area is 1024 x 780. xbot is the minimum x-coordinate.

xtop Same as above but the maximum x-coordinate of the terminal plotting area.

ybot Same as above but the minimum y-coordinate of the terminal plotting area.

ytop Same as above but the maximum y-coordinate of the terminal plotting area.

The default size for the terminal plotting area is xbot=250, xtop=1000, ybot=50, xtop=1000, ytop=750.

12 SCANS and SCAN TYPES

ascan angle scan - Scan four diffractometer motors TTH, TH, PHI, CHI

mscan motor scan - Scan up to three arbitrary motors

iscan index scan - Input three starting HKL's and three delta HKL's.

rscan rocking scan - Input HKL value, motor number (or "h", "k", "l") and delta value. The scan moves either one motor or one hkl component and centers the scan at the given HKL value.

jscan centered index scan - Like iscan except scan is centered on the given starting hkl value.

lup line-up scan - Input motor number and delta angle. Similar to a rocking scan, but the scan is centered at the current position of the motors. At the end of the scan, the motors move to the peak position.

fpk find peak scan - Like lup except fewer parameters are required and the data are not saved on disk. After the scan the motors moved to the peak calculated from the first moment.

vscan variable scan - Input HKL, variable name, starting value. The scan increments the value of the variable and moves the motors to the given HKL.

vlup centered vscan - Input HKL, variable name, delta, npts. The scan increments the value of the variable and moves the motors to the given HKL. The value of the scan is centered on the current value of the variable.

pkup peakup reflection - The scan does a series of *lup* scans centered at each orientation reflection At the end of the series, the angles of the orientation reflection are updated to reflect the new peak position.

A scan may be executed by typing the command name followed by a list of arguments. Any scan can be continued if interrupted during execution by typing co. A series of commands (including scans) can be executed by using either the sc command or the ex command.

To use the sc command, one first has to enter a list of commands into the scan table. The list can contain almost any command executable from the command line. A scan list can be created in three ways. The most direct method is to use the ds (define scans) command. Alternatively one can use the es (edit scans) command to edit the scan table directly. The command gs filename (get scans) will read a new scan table file. Once the scan table has been entered, a list of commands or scans can be executed with the sc command which accepts a series of numbers in arbitrary order corresponding to the position of the command in the scan table. Before execution begins the program does a "dry-run" to check for limit problems. If the execution is halted, the command will resume execution. The program is limited to 50 commands in the table and 100 commands in the execution list.

A series of commands or scans can also be executed with the ex command. This command diverts program input to a file (default name .EXEC) and executes the commands in order. There is no limit to the number of commands in the table. In contrast to sc, the ex command does not do a "dry-run" before execution begins. If a problem is encountered during execution, the program skips to the next command in the list. If execution halted, there is no continuation provision sion, instead a new command file must be created.

The data from all scans are stored on a file specified by the fi command.

13 SUPER INPUT/OUTPUT

INVOKING SUPER

To start the program type:

super [filename˙vars filename˙init verbose˙vlag]

filename˙Vars is the name of the file containing the program parameters saved. The default name is /usr/super/lib/super˙vars.

filename˙init is the name of the file containing the motor parameters and other information that is not saved each time the program halts. The default name is /usr/super/lib/super˙init. Any valid SUPER command can be placed in super˙init. The user can edit commands in super˙init before the progam is executed or with the ec command from within SUPER.

verbose˙flag is de-bugging aid that will cause the commands to be printed to the terminal during the start-up phase. Any character as the third argument will cause the output to appear.

DIVERTING CONTROL TO A REMOTE LOCATION

Provision has been made to divert the output of SUPER to a remote terminal device. In this procedure it is assumed that the program has been started and is running on a local terminal. The "divert" provision is useful to the user who wants to call in and use the program from a remote location and have the program continue to take data after the remote conversation is complete. The "divert" process works in the following way:

(1) When executed, SUPER puts its process i.d(PID) in a file:
(/usr/super/lib/super˙pid).

(2) When you want to divert SUPER to a remote terminal, execute the program divert on the remote device. Divert reads the PID of SUPER and saves its own PID and the remote device name in another file: (/usr/super/lib/super˙devname) . The divert program then sends the signal SIG˙TERM to SUPER.

(3) Upon receipt of the signal SUPER executes a "halt˙interrupt" and then changes its I/O to the remote device.

(4) To return to the original device, execute the command div from SUPER. This will cause SUPER to restore the I/O and send the signal SIG˙HANGUP to divert running on the Unix shell.

To restore SUPER to the original device while a scan is in progress, first halt the scan (it must be a scan executed via the sc command), then execute the command divert;co. This restores SUPER and then continues the scan.

SEE ALSO

divert(cmd)

14 TEMPERATURE CONTROL AND CONTROLLER COMMANDS

The software for these commands is not part of super, instead the program calls a separate temperature control program using the unix "system()" call.

The following is a summary of the temperature control commands. mands. See the individual entries for more detailed information.

st *temp* Set Temperature - One argument, temperature in degrees.

rt Read Temperature - Reads current temperature in both volts and degrees.

sv Set Volts - Set temperature controller to a temperature set point given in volts.

dt Delta temp - One argument, change the temperature controller set point by delta degrees.

t˙files A variable controlling the default output of the temperature controller commands. Use t˙files=1 for output to terminal only, t˙files=3 for output to terminal, line-printer and data file.

tchk A variable flag to cause the program to read the cryostat temperature before each scan (if tchk=1).

15 ZONES AND ZONE LEVELING

When operating in the two-circle mode, the diffraction zone is defined by the polar angles phi and chi. One can change the zone by simply moving the motors, e.g. the pl command or the mv command. One can also calculate the angles phi and chi such that the plane defined by two vectors will be the diffraction plane. The command cz calculates the zone defined by two vectors, mz moves to the phi and chi defined by the vectors. In the two-circle mode, the diffractometer will not move if the reciprocal lattice point is not the current diffraction zone, e.g. a br command will not attempt to move to a point not in the current diffraction plane.

If automatic zone changes are invoked (by using the dv command), the phi and chi motors will move to a point out of the diffraction plane by the use of a "default vector" (given by three arguments to the dv command). In the autozone mode, a br command will send phi and chi to the values calculated by cz using the given point and the vector given in the dv command. If auto-zone is set by invoking the dv command, iscan will do the scan in a zone defined by the starting and the ending point.

The auto-zone mode may be toggled on and off by repeating the dv command with no arguments.

BUGS

This procedure is ancient and dates from the earliest FORTRAN version of SUPER that ran on a two-circle diffractometer. It is seldom used now and consequently, it is not as well de-bugged as the rest of the program.

16 Alias

ALIAS COMMANDS

Both SUPER and PLOT are designed to implement user-defined alias commands. For example, in PLOT an alias command "zero" will set all plot limits to zero and force the plot to re-scale. In super the command "phi 100" will send the phi motor to 100.

MAKING AN ALIAS USING da

The "da" command uses a line-by-line character editor. The command responds with: "Enter Command Number ! " You enter the number of the alias you want to change. After giving a number, a cr can be used to move to the next alias, while a "-" will move to the previous alias. The format of an alias is: alias-name-¿ command

For example, you might want the alias "c60" to read scan number 2351. The alias, in this case alias #3 would read: 3!c60-¿ read 2351 tth

To change an alias, use the "¿" character to move the cursor under the characters to be changed. Type new charcters under the old ones. Enter a ¡cr¿ to enter the changes

MAKING AN ALIAS USING ea

The "ea" uses a screen editor such as "vi" to edit the alias file. In this case an alias name is entered on one line, and the command list is entered on the next line. The alias list is updated after you exit the editor. Remember that the odd-numbed lines of the file contain command names and the even numbers contain the action to be executed when the name is entered.

USING ARGUMENTS TO ALIAS COMMANDS

An alias command can use a user-defined argument. Simply use the name "arg1", "arg2" ... to refer to arguments. For example, to make an alias that uses the command "phi" to send the phi motor to a target angle, use the alias: phi-¿ motor phi arg1

SEE ALSO

ea(cmd), da(cmd)

17 Super Commands

an - angle: move two-theta and theta motors

SYNOPSIS

an tth th

DESCRIPTION

This command moves the TWO-THETA and THETA motors only.

SEE ALSO

MOVE(SUMMARY), mv(cmd), pl(cmd), motor(cmd), dmotor(cmd). To scan motor angles see ascan(cmd), mscan(cmd), and lup(cmd).

ascan - angle Scan of four diffractometer motors

SYNOPSIS

ascan tth th phi chi dtt dth

DESCRIPTION

This routine scans four diffractometer motors in units of delta degrees. (See "mscan" for scanning an arbitrary motor.) There are ten arguments: four initial positions (Two Theta, Theta, Phi, Chi), four delta angles, number of points and monitor preset.

If the variable fbkg is non-zero and less than one, the spacing between the data points will be a factor of three larger in the wings of the scan. For example, if fbkg = 0.5, the scan will have a spacing of 3 * delta for the first 25% of the scan, a spacing of delta for the second 50% of the scan, and a spacing of 3 * delta for the last 25%.

SEE ALSO

SCANS(SUMMARY), mscan(cmd), iscan(cmd), rscan(cmd), lup(cmd)

att - read or set attenuator

SYNOPSIS

att [new˙posit]

DESCRIPTION

This command sets attenuator postions numbered 1 - AMAX, where AMAX is a variable set by the user. If no argument is given, the current attenuator position is read. The attenuators have scale factors that are given by the variables af1, af2, ... Setting an attenuator causes the scale factor to be entered into the scalar parameters. The curent scalar parameters can be printed with the "spar" command.

SEE ALSO

amax(var), spar(cmd)

br - bragg: move to angles calculated from hkl

SYNOPSIS

br *h, k, l* [*mon*]
  pbr *h, k, l* [*mon*]

DESCRIPTION

This command moves to the angles defined by *h, k, l.* The routine expects three arguments *h, k l,* with an optional fourth argument specifying the monitor value. If the last argument is given, the routine counts after moving.

The command pbr moves only the *theta* and the *two-theta* motors. Use this command with powders or while aligning crystals.

SEE ALSO

MOVE(SUMMARY), mv(cmd), pl(cmd), motor(cmd), dmotor(cmd), br(cmd), wh(cmd). To scan motor angles see ascan(cmd), mscan(cmd), and lup(cmd).

BUGS

br does not automatically count like it used to in earlier versions of SUPER. This was changed to allow more flexibility in building commands with the "alias" option. To make a command that functions almost like the old br, make the alias "brg -¿ br arg1 arg2 arg3; ct." Now the command brg will function like the old br command.

ca - calculate angles from H, K, and L

SYNOPSIS

ca *h k l* [*phi chi*] [*p*]

DESCRIPTION

This routine calculates the angles *two-theta, theta, phi* and *chi* from *h, k,* and *l.* Three arguments are required (*h, k,* and *l*). The results are displayed as follows:

CA: h, k, l, two-theta, theta, phi, chi

The optional arguments, *phi* and *chi*, are used only in the two-circle mode. They specify the diffraction zone used in the calculation. If *phi* and *chi* are not supplied, the current values are used.

If the letter "*p*" is included as a final argument, the results will be printed on the line-printer.

SEE ALSO

ci(cmd)

cd - change working directory

SYNOPSIS

cd *dirname* [*filename*]

DESCRIPTION

The cd command changes the name of the working directory for data file output. (Similar to the unix shell command "cd"). The routine assumes that the current directory is the one given by the file command. If a second argument is given, it is used as the file name in the new directory. If the second argument is ".", the filename is unchanged. Otherwise the routine will prompt for a new file name.

The command pwd prints the name of the current working directory.

These are pseudo-unix commands, they do not invoke the unix shell.

SEE ALSO

file(cmd), unix(cmd)

BUGS

If "DOS" is defined in*options.h* before compiling, the program uses a different file assignment scheme. In the DOS case, each scan is written to a separate file and one uses the syntax "file dirname p" to change the working directory.

ci - calculate indices

SYNOPSIS

ci *tth th* [*phi chi*] [*p*]

DESCRIPTION

This routine calculates *h, k* and *l* from the angles *two-theta, theta, phi* and *chi.* Two arguments are required (*two-theta* and *theta*), and two are optional (*phi* and *chi*). If *phi* and *chi* are not supplied, the current values are used in the calculation. The results are displayed in the following format:

CI: I h, k, l, two-theta, theta, phi, chi

If an optional final argument is the letter *p*, e.g. "ci p", the results will be printed on the line-printer.

SEE ALSO

ca(cmd)

cm - Write a comment to the data file

SYNOPSIS

cm

DESCRIPTION

Use this command to place comments on the data file and the line printer. Comments are recorded line-by-line until a line beginning with "." is encountered or an interrupt is generated (usually the del or rub key).

cma - calculate the monochromator theta from the energy

SYNOPSIS

cma *energy* [*p*]

DESCRIPTION

Gives monochromator theta calculated using the argument as the energy in keV. Use the letter "p" as an optional final argument to force output to line-printer and data file.

The variable mds must be set to the monochromator d-spacing.

SEE ALSO

ENERGY(SUMMARY), cme(cmd), efix(var), mds(var), mme(cmd)

cme - calculate incident energy from monochromator theta

SYNOPSIS

cme [*energy*] [*p*]

DESCRIPTION

Gives current monochromator energy in keV (no argument) or calculates energy using the argument as the monochromator theta. Use the letter "p" as a final argument to force output to line-printer and the data file.

The variable mds must be set to the monochromator d-spacing.

SEE ALSO

ENERGY(SUMMARY), cma(cmd), efix(var), mds(var), mme(cmd)

co - continue an aborted scan

SYNOPSIS

co

DESCRIPTION

Continue an aborted scan or command list. A single scan entered from the keyboard or a list of commands executed with the sc command may be continued with the co command. A simple command entered from the keyboard (e.g. br). may **not** be continued, however a scan entered from the keyboard may be continued.

BUGS

The co command does not work in conjunction with the ex command.

cp - unix "cp" command

See the unix section.

cr - count read

See the cs command.

cs - count start

SYNOPSIS

cs cr

DESCRIPTION

cs clears the scalers and starts acquiring counts. cr stops acquiring counts and prints the results.

These commands can be used to integrate counts while moving motors. For example the sequence "cs;mot 3 45 500;cr" will acquire counts while the motor 3 (phi) is moved to 45 degrees at 500 pps.

SEE ALSO

cr(cmd)

cread - maintenance command to read CAMAC module

SYNOPSIS

cread *f n a*

DESCRIPTION

Maintenance command used to test the CAMAC interface. Three arguments are required,*f*, *n*, and *a*. This command can be deleted at compile time by editing options.h.

SEE ALSO

cset(cmd), cwrite(cmd)

cset - maintenance cmd to execute CAMAC "cset"

SYNOPSIS

cset *f n a*

DESCRIPTION

Maintenance command used to test the CAMAC interface. Three arguments are required, *f*, *n*, *a*. This command can be deleted at compile time by editing options.h.

SEE ALSO

cread(cmd), cwrite(cmd)

ct - count

SYNOPSIS

ct [*mon*] [*output˙flag*]

DESCRIPTION

This command will count until the preset value given by the argument is reached. If *mon* is not given, the value of the variable ti will be used. For example, "ct ¡return¿" will count to the preset value stored in ti. "ct 1.5e6 ¡return¿" will count for 1.5e+06 monitor.

If the flag iflag is set, counting is done in the "intg" (integrate) mode. In the intg mode the counts are acquired while moving a specified motor. Background counts may also be acquired at the beginning and the end of the integration.

If the *output˙flag* = 0, the output will be local, i.e. it will appear on the tty only. For any other third argument, *output˙flag* = "no˙output" the column headings will be suppressed so that a "scan" can be created by the user. (The command phd will manually print a heading for such a scan.)

If desired, an alias command "lct" can be created to make a "local count" command: "lct -¿ ct arg1 0."

SEE ALSO

INTG(SUMMARY), phd(cmd), ti(var)

cwrite - maintenance command to write to a CAMAC module

SYNOPSIS

cwrite *f n a value*

DESCRIPTION

Maintenance command used to test the CAMAC interface. Four arguments are required, *f, n, a,*, and the value to be writ- ten. This command can be deleted at compile time by editing options.h.

SEE ALSO

cread(cmd), cset(cmd)

cz - calculate zone

SYNOPSIS

cz H1 K1 L1 H2 K2 L2

DESCRIPTION

This command calculates a diffraction zone defined defined by two vectors. The six arguments give the H, K and L of the two vectors used in the calculation.

The variable fv determines which hemisphere the calculated value of chi (or phi) falls in. (Two-circle mode only).
fv ¿= 0 —¿ -90 ¡ chi ¡ 90
fv ¡ 0 —¿ 90 ¡ chi ¡ 270
Chi = 90 is a special case where there is a +/- 180 degree degeneracy in phi.
fv ¿= 0 —¿ -90 ¡ phi ¡ 90 (chi = 90)
fv ¡ 0 —¿ 90 ¡ phi ¡ 270 (chi = 90)

SEE ALSO

ZONE(SUMMARY), mz(cmd), fv(var)

da - define alias

SYNOPSIS

da

DESCRIPTION

This routine is used for defining "alias" commands An "alias" is a list of commands which will be executed each time you type the alias name. Any command which is recognized by SUPER may used in the list of commands. The alias list currently in memory is contained in a file given by the ga command (default = /usr/super/lib/super˙alias).

The command da initially responds with "Enter Alias Number ¿". The user then enters a command number and the routine lists the current alias/command list stored with that number. The user can either (1) type in a new alias/command list (you must separate the alias name and each command in the list by ";"), or (2) edit the current alias/command list, or (3) view the next alias/command list (enter a carriage return), or (4) view the preceding alias/command list (enter a "-").

To edit a alias/command list, type "¿" until the cursor is under the character to be changed, then enter the correct character. Be careful, if you are adding characters to the command, you may over-write spaces.

To avoid this you may want to add extra spaces between arguments when you first enter a command. The command ea can also be used to edit the alias list. When this command is typed the text editor is invoked.

Arguments may be included in the command list. "arg1" means the first argument, "arg2" means the second, etc. For example suppose you wanted to define a command to collect the integrated intensity while slewing motor 3 from one angle to another. In da you would type

intphi;mot phi arg1;cs;mot phi arg2 500;cr

(where the 500 refers to the motor speed). The program would respond with

intphi-¿mot phi arg1;cs;mot phi arg2 500;cr

To collect the integrated intensity between 1.5 an 2.5 degrees you would type "intg 1.5 2.5"

Several commands can be listed by entering "p start stop" after "Enter Alias Number ¿" For example "p 10 15" will list command 10 through command 15.

Return to SUPER with the interrupt key.

The maximum number of alias commands is 50.

SAMPLE ALIAS COMMANDS

Move diffractometer motors to special position (in this case 40 40 0 0)

mount; mv 40 40

Set the temperature to value given by argument and wait ten minutes before proceeding

templ; st arg1; wait 10

Turn the line printer on and off

lpon; file stdl

lpoff; file null l

Turn "intg" mode on and off

ion; iflag=on

ioff; iflag=off

Set the "standard" data output format

stdf; format h k l cts tim tth th phi chi

Set a special data output format

enfor; format hkl e cts tim mcr

Set the "standard" limits

nlm; lm tth 1.5 100;lm th -5 48;lm rel -26 37; lm

Move to an h, k, l and count. (This looks like the old version of br. brg; br arg1 arg2 arg3; ct

Make SUPER's help command look like the Unix man command.

man; help arg1

Move the various motors to a given position at a given speed. (Omit the speed to get the default).

tth; mot 1 arg1 arg2

the; mot 2 arg1 arg2

phi; mot phi arg1 arg2

chi; mot chi arg1 arg2

SEE ALSO

ga(cmd), ea(cmd), ds(cmd), es(cmd), gs(cmd)

BUGS

When using da strange things may happen if you try to extend the number of characters with the "¿" syntax described above. Since the editing capabilities of da are limited, users experienced in using a Unix text editor may prefer ea to enter new alias definitions.

divert - Restore diverted output

SYNOPSIS

divert
    divert; co
    divert; ex

DESCRIPTION

Provision has been made to divert the output of SUPER to a remote terminal device. This is useful for interrupting SUPER from a remote location. This process works in the following way:

(1) When executed, SUPER puts its process i.d. (PID) in a file called PIDFILE[] in the library directory /usr/super/lib.

(2) When you want to divert SUPER to a remote terminal, execute the program DIVERT on the remote device. DIVERT reads the PID of SUPER and saves its own PID and the remote device name in the file DEVFILE[] in /usr/super/lib. DIVERT then sends the signal SIG˙TERM to SUPER.

(3) Upon receipt of the signal SUPER executes a "halt˙interrupt" and then changes its I/O to the device found in DEVFILE.

(4) To return to the original device, execute the command divert from SUPER. This will cause SUPER to restore the I/O and send the signal SIG˙HANGUP to DIVERT

To restore SUPER to the original device while a scan is in progress, first halt the scan (it must be a scan executed via the sc command), then execute the command "divert;co". This restores SUPER and then continues the scan.

If you want to use ex to execute the scan list, do not start execution from the remote terminal. Instead, first execute the command "enq=0" to turn off the automatic heading enquiry. (Failure to to this will cause the program to hang while waiting for a response to "New heading?". Second, execute the command sequence "divert;ex". This restores SUPER and executes the command ex.

SEE ALSO

co(cmd), sc(cmd), ex(cmd)

BUGS

This command runs under the Unix operating system only (not DOS).

dmotor - incrementally move a motor

SYNOPSIS

dmotor mnum degrees [speed]

DESCRIPTION

This command moves a single motor to a given angle that is calculated as the present position plus the argument *degrees.* It expects two arguments, *mnum*, the motor number and incremental angle in degrees. *Mnum* may be expressed as either the motor number or the motor name. A third optional argument may be used to specify a motor speed. E.g. dmot tth .1 means move the TTH motor +.1 degrees from the present position.

SEE ALSO

MOVE(SUMMARY), mv(cmd), gmv(cmd), an(cmd), pl(cmd), motor(cmd)

ds - define scans

SYNOPSIS

ds

DESCRIPTION

This routine is used for entering a command list for subsequent execution with the sc command. Any command which is recognized by SUPER may be placed in the command list. The scans which are currently in memory are listed in a file given in the gs command.

The command ds initially responds with "Enter Command Number ¿". The user then enters a command number (between 0 and 50) and the routine lists the current command stored with that number. The user can either (1) type in a new command, or (2) edit the current command, or (3) view the next commmand (enter a carriage return), or (4) view the preceding command (enter a "-").

To edit a command, type "¿" until the cursor is under the character to be changed, then enter the correct character. Be careful, if you are adding characters to the command, you may over-write spaces. To avoid this you may want to add extra spaces between arguments when you first enter a command.

Several commands can be listed by entering "p start stop" after "Enter Command Number ¿" For example "p 10 15" will list command 10 through command 15.

Return to SUPER with the interrupt key.

SEE ALSO

es(cmd), gs(cmd), sc(cmd)

BUGS

When using ds strange things may happen if you try to extend the number of characters in the scan definition using the "¿" syntax described above. Since the editing capabilities of ds are limited, users experienced in using a Unix text editor may prefer es to enter new scans.

dt - change cryostat temp. by incremental amount

SYNOPSIS

dt *temp* [*p*]

DESCRIPTION

Change the temperature to a value calculated from the current temperature plus the argument *temp*. An external program is called with the appropriate arguments. The output will appear on the terminal only if the variable t˙files = 1 and on the terminal, line-printer and data file if t˙files = 3. The letter "p" as a final argument will force the output to appear on the line printer and the data file (current call only).

SEE ALSO

TEMP(SUMMARY), st(cmd), rt(cmd), t˙files(var), tchk(var)

BUGS

Check to see that you can execute dt from the shell level before you try it from SUPER.

dv - set the "default vector" used in zone leveling

SYNOPSIS

dv [*h k l*]

DESCRIPTION

Set the "default vector" used the the auto-zone mode. (TWO-CIRCLE MODE ONLY). Three optional arguments specify *h, k, l* of the default vector. This command toggles the "auto-zone" mode on and off.

SEE ALSO

ZONE(SUMMARY), cz(cmd), mz(cmd), frz(cmd)

BUGS

This and other two-circle mode commands are relics from the original FORTRAN version of super and have not been extensively de-bugged in the C-version.

ea - edit alias list

SYNOPSIS

ea

DESCRIPTION

This command is designed for editing the alias list and is an alternative to the da command. The routine automatically calls the ga (get alias) routine when the editing is complete.

Use of this command requires knowledge of one of the Unix text editors. The name of the text editor can be changed with the ed (editor name) command.

SEE ALSO

ga(cmd), da(cmd), es(cmd), ed(cmd)

ec - edit camac slot assignments

SYNOPSIS

ec

DESCRIPTION

Edit the "init" file used at start-up. This is the preferredn command to enter the mset and sset commands used to set motor and scaler slot numbers, channel numbers, etc. The routine calls slots when the editing is finished.

Use of this command requires knowledge of one of the Unix text editors. The name of the text editor can be changed with the ed (editor name) command.

FILES

Default is "super˙init" in the SUPER library directory (see /usr/super/lib or options.h).

SEE ALSO

ea(cmd), es(cmd), ed(cmd), slots(cmd)

ed - editor name

SYNOPSIS

ed [editor˙name]

DESCRIPTION

With no arg, this command lists the name of the unix text editor used with the es, ea and ec commands, e.g. "vi" or "emacs" or "ed".

The editor name provided must be recognized by the Unix shell.

SEE ALSO

ea(cmd), ec(cmd), es(cmd)

end - end the program

SYNOPSIS

end

DESCRIPTION

This command ends program SUPER. It closes all files and flushes all buffers. The command asks the user to enter a carriage return to verify that you really want to end the program.

SEE ALSO

save(cmd)

erase - erase plot

SYNOPSIS

erase

DESCRIPTION

Erase the graphics from the terminal. This command also works from from the Unix shell. The command rep will erase and re-plot the graphics display.

SEE ALSO

rep(cmd)

es - edit scan list

SYNOPSIS

es

DESCRIPTION

This command is designed for editing the scan list and is an alternative to the ds command. The routine automatically calls the gs (get scans) routine when the editing is complete.

Use of this command requires knowledge of one of the Unix text editors. The name of the text editor can be changed with the ed (editor name) command.

SEE ALSO

gs(cmd), ds(cmd), ea(cmd), ed(cmd)

ex - execute sequential command list

SYNOPSIS

ex [filename]

DESCRIPTION

This command executes a series of commands that are listed in order in an input file. The name of the input file may be supplied as an argument, or if no argument is given, a default name, .EXEC, (defined in options.h) is used.

The ex command can be used as an alternative to the sc command to execute a command list. In contrast to the sc command, the ex command does not do a "dry˙run" to check for limit conditions, and the commands must be listed in order in the input file. If an error is encountered, e.g. a limit condition occurs during a scan or incorrect argument list is given, the program goes on to the next command in the list.

An important difference between ex and sc commands is the use of the continuation command, co. Unlike sc or a keyboard initiated scan, the execution of a list given by ex **cannot** be continued. The ex command always starts with the first command in the input file.

The alias command "eex-¿ ! vi .EXEC" is useful for editing the file .EXEC.

SEE ALSO

sc(cmd), divert(cmd)

BUGS

This is still under development and may have bugs. It does not work under DOS. You cannot include ex commands in the scan list used with the sc command and vice versa.

file - open file or device for data output

SYNOPSIS

file
  file filename
  file std/null l/t

DESCRIPTION

Data are output to three "files", the DATA file (saved on disk), the LINE-PRINTER device file and the TTY device file. Any file may be set to "null" if no output is desired.

With no argument the command lists all files.

With one argument the command opens a new file for data output

With three arguments the command expects the words std or null as the second argument and the letters l or t as the third argument. Std refers to the standard output device as defined in options.h. The letters l or t refer to the lineprinter device or the terminal device respectively.

Examples:

file xray2 -

Open a data file named xray2 for data output. If the flag afs is set (see below), a "file sequence number" (variable name fsn) is appended to the file name. If the current fsn = 4, the new file name would be "xray2.05".

file null l - Do not write data on the line printer.

file std t - Resume sending data to the terminal.

AUTOMATIC FILE NUMBERING

If the flag afs is set (automatic file sequence), the program will automatically open a new data file if the size gets above a limit set at compile-time in the file options.h. Currently this limit is 80000 blocks. file name as an argument to fi. The program will append a number to the end of the name. The number is the value of the variable fsn + 1. E.g. if fsn is currently 6, the command line fi xray would open a data file named "xray.07".

FILES

options.h: Location of default file and device names.

SEE ALSO

t˙files(var), afs(var), fsn(var)

BUGS

The program uses an irregular subroutine gprintf() to send the same data to three output files.

If "DOS" is defined in options.h before compiling, the program ses a different file assignment scheme. In the DOS case, each scan is written to a separate file.

format - set the format of data output

SYNOPSIS

format

format [col1˙name col2˙name col3˙name ...]

DESCRIPTION

This command is used to set up the format of the columns in a scan. The order of the columns is fixed, but you may choose to print or not print a column.

The names of the available columns are given by the hpar command, the spar command, and the mpar command. To change the format, use an argument made up of the column heading names. For example the command "format e tim cts tth mcr" will print E TIM CTS TTH MCR as columns for the scan output.

To add or subtract a column from the format use "+" or "-" as the last argument. For example, to add the chi motor to the existing format use "format chi +"

The command with no argument will print the current column format.

The number of digits in the output for each column is set with the hset, sset, and mset commands.

In the "integrate" mode, additional formating of the scaler channels is done via the iformat command.

SEE ALSO

iformat(cmd)

fpk - find peak

SYNOPSIS

fpk mname delta [npts mon]

DESCRIPTION

This command is designed to quickly center a motor on a peak position. The routine scans a single motor (or the th/tth motors) in increments of delta degrees. The scan is centered about the current position of the motors. There are two required arguments, motor number (or motor name) and delta (degrees), and two optional arguments, the number of points and the monitor counts. If mname=0, a theta/two-theta scan is performed with a two-theta increment given by delta. The argument mname may be specified as a number or a name, e.g. "tth". At the end of the scan, the motors are moved to the peak position calculated from the first moment. If npts is omitted, the value contained in the variable nfpk is used. If mon is omitted the monitor is set to the value given by the variable ti.

The fpk command is functionally the same as the lup command, but fewer arguments are required and no data are saved on disk.

SEE ALSO

lup(cmd), pkup(cmd), nfpk(var)

frz - freeze motor

SYNOPSIS

frz

frz frz˙angle [frz˙value] [frz˙value˙2]

DESCRIPTION

This command is used to set the angle calculation modes. A four-circle diffractometer over determines the angles required to reach a point in reciprocal space, so a motor is generally "frozen" to allow calculations to be performed.

Only the first letter of the string frz˙angle is significant. Allowed values of frz˙angle are "omega", "phi", "chi", "2" (2-circle mode), "5" (five-circle mode), or "a" (fixed azimuth mode). The second argument and third arguments are numerical values. If no argument is given, the current mode is listed. A summary of the modes is given below.

omega Fix the angle omega to the value given by frz˙value. A bisecting geometry is selected: frz omega 0. Note that if omega is non-zero, not all of reciprocal space can be accessed.

phi Fix the angle phi to the value given by frz˙value.

chi Fix the angle chi to either +90 or -90. The two choices are assigned by using the sign of the argument frz˙value. It is also possible to specify the hemisphere that the calculated value of phi. To restrict chi to -90 ¡ chi ¡ 90 use —frz˙value— ¿ 89. To restrict chi to 90 ¡ chi ¡ -90 use —frz˙vale— ¡ 89.

2-cir Hold both phi and chi constant. In this mode phi and chi may be set by the pl command. The calculated value of chi falls either in the range -90 ¡ chi ¡ 90 (if frz˙value ¿= 0), or 90 ¡ chi ¡ 270 (if frz˙value ¡ 0). The calculated diffraction plane has a +/- 180 degree degeneracy in phi if (and only if) chi = 90.

5-cir Five-circle mode (for surface diffraction).

az Fixed azimuth mode. If both frz˙angle and frz˙angle˙2 are given, assign new values to fphi and fchi, polar angles that give the direction of the surface normal.

SEE ALSO

ORIENT(SUMMARY), om(cmd).

BUGS

Both "2-circle" and "azimuth" are obsolete modes that have been left in the program. Any calculation mode can be deleted at compile-time (see options.h).

ga - get alias list

SYNOPSIS

ga [filename]

DESCRIPTION

Read scan parameters from the current file (no argument supplied) or from the given file (argument on command line).

FILES

/usr/super/lib/super˙alias (default)

SEE ALSO

da(cmd), gs(cmd), ds(cmd)

gmv - global move: move all motors

SYNOPSIS

gmv motor1 motor2 ...

DESCRIPTION

This command moves all non-inhibited motors, (not just the diffractometer motors). The order of the motors is the same as that given by the mparm command. Any motors positions missing from the end of the list are assumed to be zero.

SEE ALSO

MOVE(SUMMARY), mv(cmd), motor(cmd), dmotor(cmd), mparm(cmd), look(cmd).

gor - get orientation reflection

SYNOPSIS

gor refl#

DESCRIPTION

This command moves the motors to the position defined by the given orientation reflection. The argument refl# must be in the range 1-MAXOBS (currently 20).

SEE ALSO

ORIENT(SUMMARY), or(cmd), sor(cmd), nobs(var)

gs - get scan list

SYNOPSIS

gs filename sscan filename

DESCRIPTION

Read or save scan parameters from/to the current file (no argument supplied) or from the given file (argument on command line).

The command ex will execute a list of commands contained in a file. This list is not stored as a scan list, and can be any length.

FILES

/usr/super/lib/super˙scans (default)

SEE ALSO

ds(cmd), ga(cmd), da(cmd), sc(cmd)

help - help command

SYNOPSIS

help [-k] [command˙name]

DESCRIPTION

This manual is available on-line via the help command. With an argument, the command will print information about one command, variable or summary on the tty. With no argument the routine will list all commands and variables. A prompt then asks the user to enter a carriage return ¡cr¿ to run the HELP program. With -k as the first argument, the index will be searched for the keyword given by the second argument e.g. help -k count.

The HELP program is a shell script that is separate from the SUPER program. HELP contains a menu driven help facility. The user may ask to see an index of commands, variables or summaries. Individual commands, variables or summaries can also be requested. The user can ask to have the HELP output printed on the line printer.

The file names are listed at the top of each manual page. The complete command name need not be given. For example if one enters help while in the HELP program, the computer will print will print this manual page (stored in the file help.cmd) on the terminal. An input of a single letter, e.g. f will print information about all files beginning with f. (This will sometimes result in more information than you care to see.)

The HELP program can also be executed from the shell (external to the SUPER program) by typing /usr/super/bin/help/help˙super.

FILES

/usr/super/bin/help˙super HELP program shell script
/usr/super/bin/help˙cmd HELP program shell script
/usr/super/help/SUPER Directory containing help files
/usr/super/help/SUPER/cmdindex Index of the command files
/usr/super/help/super/varindex Index of the variable files

hpar - hkl print parameters

SYNOPSIS

hpar

DESCRIPTION

This routine prints the "hkl parameters". This includes the Miller indices plus a collection of other parameters such as the d-spacing. The output format argument (width.decimal format, e.g 8.3) can be set with hset.

The HKL list is divided into a "read-only" section and a user-defined section. The names of the parameters in the "read-only" section of the HKL list cannot be changed. Only the data output format may be varied. The user can add any valid SUPER variable to the user-defined section of the HKL list with the hset command. The variable nhkl controls the number of HKL parameters in the list. Any of the HKL parameters can be included in the data output using the format command.

SEE ALSO

mpar(cmd), spar(cmd), ipar(cmd), hset(cmd), nhkl(var)

hset - set hkl parameters

SYNOPSIS

hset # label format print˙flag

DESCRIPTION

The "HKL" list includes a number of parameters such as H, K, L, d, d*, etc. These parameters can be included in the data output with the format command. The user can add any valid SUPER variable to the HKL list with the hset command. The number, "#", is the next higher number in the HKL list. The maximum number in the list, nhkl, is automatically incremented with each hset. The "label" parameter must be the name of a valid SUPER variable. The "format" parameter is in a "width.decimal" format, e.g. 8.3. The print flag is 0 or 1.

Use hpar to display the parameters. Once a variable has been added to the HKL list, it can be included in the data output with the format command.

The print flag can also be changed with the "format" command.

SEE ALSO

hpar(cmd), mset(cmd), sset(cmd), iset(cmd), nhkl(var)

iformat - set data output format in "intg" mode

SYNOPSIS

iformat
    iformat [col1˙name col2˙name col2˙name]

DESCRIPTION

This works similar to the format command but only the headings of the scalers used in the "intg" (integrate counts) mode are checked. (To view the complete list of scaler headings in the intg mode use the command ipar).

NOTE: The column format will be printed out as given by the format command unless the variable iflag is set.

The command with no argument will print the current column format of an integrated scan including the motors and H, K, L, etc.

The number of digits in the output for each column is set with the iset, i1set, and i2set commands.

EXAMPLE

If a previous call to format had set up the columns
H L TIM CTS TTH,
an input of
iformat cts.b1 cts.b2 cts.i
would result in a "intg" column format of
H L CTS.B1 CTS.B2 CTS.I TTH TH.

SEE ALSO

format(cmd), iflag(var), intg(cmd), ipar(cmd), iset(cmd), i1set(cmd), i2set(cmd)

init - initialize orientation matrix

SYNOPSIS

init [flag]

DESCRIPTION

This command calculates the orientation matrix from the lattice parameters and the orientation reflections. If the flag is "1", the orientation matrix is printed on the terminal. Normally init is called from the file "super˙init" which is read when SUPER is executed. init is called automatically each time a lattice parameter or an orientation reflection is changed.

init has two different modes. If the variable FRZ˙LAT = 1, the orientation matrix is calculated from H, K, L, omega, PHI, CHI of the two orientation reflections. If FRZ˙LAT = 0, the lattice parameters are calculated by the program to be consistent with three or more orientation reflections which specify two-theta in addition to the above variables.

The pertinent variable names are as, bs, cs, al, be, ga (the six lattice parameters) and *h_i*, *k_i*, *l_i*, *t_i*, *w_i*, *p_i*, *c_i* where i is equal to 1, 2 or 3 ... MAXOBS. (MAXOBS is set at compile time in define.h)

SEE ALSO

ORIENT(SUMMARY), om(cmd)

intg - set up integration parameters

SYNOPSIS

intg mnum start stop time [iflag]

DESCRIPTION

Set up "intg" parameters. The argument mnum may be given as a name (e.g. phi) or a number. The arguments start and stop are given in degrees from the present position. The time is the time in seconds for the motor movement (while counting). The last argument, iflag, is optional, default = "off".

If mnum = 0, the integration is done over theta/two-theta with start, stop and time referring to the two-theta motor (the theta motor moves half as far half as fast).

Note that iflag=2 is a special "flip-flop" mode where the start and ending motor positions alternate to eliminate the "flyback" between successive points of a scan.

SEE ALSO

INTG(SUMMARY), iflag(var), iset(cmd), ipar(cmd), iformat(cmd)

BUGS

There are some serious problems with integration related to hardware. The computer issues "start acquiring counts", "move motors" and "stop acquiring counts" as three separate commands. There is no provision for preventing a computer time-out between two of the commands causing the scalers to acquire counts while the motors are not moving (this will be at one of the limits of motor motion). A solution will be to build a ttl line to gate the counters only when the motors are actually moving. This is the reverse of the normal the scalers usually count when the motors are stopped.

A second problem is that the integration is done in a time mode. No provision is made to adjust the motor speed if the monitor time changes. One can, however, store the accumulated counts in the monitor channel and take a ratio.

inv - invert direct/reciprocal lattice

SYNOPSIS

inv

DESCRIPTION

Invert the direct and reciprocal and direct space lattice parameters. This allows the user to first type in the direct space lattice parmaters using "as= ...", etc. The command inv will then invert the lattice and swap the direct and reciprocal lattice parameters.

SEE ALSO

rl(cmd)

ipar - intg scaler parameters

SYNOPSIS

ipar

DESCRIPTION

This routine prints the scaler parameters for used in the "intg" mode, background(1), slew, and background(2). The intg headings are are set with the iset command. The scale factor is set by the sset command. The print flag is set by the iformat command.

The intg scalar parameters are easily changed by use of the ec (edit camac) command that edits the file /usr/super/lib/super˙init

SEE ALSO

mpar(cmd), hpar(cmd), spar(cmd), i1set(cmd), i2set(cmd), iset(cmd), ec(cmd)

iscan - integer (hkl) scan

SYNOPSIS

iscan start˙h start˙k start˙l del˙h del˙k del˙l npts mon

DESCRIPTION

This routine scans from an initial HKL in units of delta HKL.There are eight arguments: three initial HKL values, three delta HKL values, number of points and monitor preset.

If the variable fbkg is non-zero the spacing between the data points will be a factor of three larger in the wings of the scan. For example, if fbkg = 0.5, the scan will have a spacing of 3 * delta for the first 25% of the scan, a spacing of delta for the second 50% of the scan, and a spacing of 3 * delta for the last 25%.

SEE ALSO

SCANS(SUMMARY), ascan(cmd), rscan(cmd), lup(cmd), jscan(cmd), fbkg(var)

iset - set intg scaler parameters

SYNOPSIS

iset s# lab slot chan scale [dec]
i1set s# lab slot chan scale [dec]
i2set s# lab slot chan scale [dec]

DESCRIPTION

In the "intg" mode of counting (integration), each "count" consists of three parts: a "first background" where counts are acquired with stationary motors, a "slew portion" where counts are acquired while a motor moves to a second position and a "second background" where counts are acquired at the second position. The scaler parameter used in each portion are kept separately from those used in normal counting. The three iset commands are used to set the "intg" scaler parameters and the command ipar reads them.

iset This command sets the scaler parameters used in the "slew" portion of data acquisition.

i1set This command sets the scaler parameters used in the first background portion of data acquisition.
i2set This command sets the scaler parameters used in the second background portion of data acquisition.
sset command the arguments are:
s# scaler number in the range 1-nscl.
lab scaler name
slt camac slot number
chn channel number in the range 1-4
scale scale factor ¿ 0 (output is actual number divided by the scale factor)
dec number of digits in the output. (optional, if omitted the current value remains)

SEE ALSO

INTG(SUMMARY), sset(cmd), ipar(cmd), nscl(var)

BUGS

Don't use a zero scale factor. I don't remember what happens.

i1set - set intg scaler parameters

See command iset.

i2set - set intg scaler parameters

See command iset

jscan - centered hkl scan

SYNOPSIS

jscan start(h) start(k) start(l) del(h) del(k) del(l) numpts mon

DESCRIPTION

This routine works like iscan except that the scan is centered about the given hkl There are eight arguments: three starting hkl values, three delta jkl values, number of points and monitor preset.

If the variable fbkg is non-zero the spacing between the data points will be a factor of three larger in the wings of the scan. For example, if fbkg = 0.5, the scan will have a spacing of 3 * del for the first 25% of the scan, a spacing of del for the second 50% of the scan, and a spacing of 3 * del for the last 25%.

SEE ALSO

SCANS(SUMMARY), ascan(cmd), rscan(cmd), lup(cmd), fbkg(var)

log - plot data with a logarithmic y-axis

SYNOPSIS

log
lin
lsc ymin ymax

lsc

DESCRIPTION

These commands switch between log and linear scales for the data plot. The linear scale y-axis is between zero and the variable given by ymax. The log scale y-axis is set by the lsc command. If no arguments are given with the command lsc, the current parameters are given. PP Both modes have an auto-scale mode set by the flag a˙scale.

SEE ALSO

PLOT(SUMMARY), a˙scale(var)

lin - plot data with a linear y-axis

See the command log.

lsc - set the log scale y-axis limits

See the command log.

lm - set/read software motor limits

SYNOPSIS

```
lm
    lm mnum
    lm mnum high low
```

DESCRIPTION

With no arguments all the motor limits are printed. With one argument the limits of that motor only are printed. With three arguments the high and low limits of the given motor are changed. The argument "mnum" can be either the motor number or the motor name as it appears in the list generated by the mparm command.

The maximum number of motors in the list is controlled by the variable nmotors.

The "rel" limits control the relative position of the theta and two-theta motor and are designed to prevent collisions between the flight paths and the chi circle.

If both the high and low limits are set to zero, no limit checking is done for that motor.

SEE ALSO

LIMITS(SUMMARY), nmotors(var)

look - look at motor positions

SYNOPSIS

look [p]

DESCRIPTION

Print the current position of all motors. Unlike the wh command, look will not calculate the energy or the HKL coordinates of the point.

SEE ALSO

wh(cmd)

ls - unix "ls" command

See the unix section.

lup - line-up scan

SYNOPSIS

lup mnum˙delta npts˙mon

DESCRIPTION

This routine scans a single motor in increments of delta degrees. The scan is centered about the current position of the motors. There are four arguments: motor number, delta (degrees), number of points and monitor preset. If mnum = 0, a theta/two-theta scan is performed with a two-theta increment given by delta. The argument mnum may be specified as a number or a name, e.g. "tth". At the end of the scan, the motors are moved to the peak position calculated from the first moment.

If the variable fbkg is non-zero the spacing between the data points will be a factor of three larger in the wings of the scan. For example, if fbkg = 0.5, the scan will have a spacing of 3 * delta for the first 25% of the scan, a spacing of delta for the second 50% of the scan, and a spacing of 3 * delta for the last 25%.

The pkup command does a series of lup scans at each of the orientation reflections. The fpk command (find peak) is a quick version of lup that does not save data on disk.

SEE ALSO

SCANS(SUMMARY), rscan(cmd), pkup(cmd), fpk(cmd), ascan(cmd), fbkg(var)

minh - motor inhibit command

SYNOPSIS

minh mnum1 [mnum2, mnum3, ...] mflag

DESCRIPTION

Use this command to turn the inhibit flag on the motors on and off. (This flag is also set with the mset command). The arguments mnum1, mnum2, etc are the names or numbers of motors and mflag is either 0, 1 or the words "on" or "off". The motor name "all" can be used to set the same parameter for all the motors.

SEE ALSO

mset(cmd), mpar(cmd), mv(cmd)

mme - Move monochromator energy

SYNOPSIS

mme energy

DESCRIPTION

Expects the energy in keV as an argument. Moves the monochromator theta.

SEE ALSO

ENERGY(SUMMARY), cma(cmd), cme(cmd), efix(var), mds(var)

mod - motor modulo 360/180/off

SYNOPSIS

mod motor value

DESCRIPTION

Each motor has a flag to define the allowed motor values. For circular motion the value can be either 360 or 180. (Other values are not allowed.) In the former case the value of the printed angles are adjusted (by adding or subtracting 360) so that they fall in the range 0 ¡ angle ¡ 360 while in the latter case the values fall in the range -180 ¡ angle ¡ 180. For linear motion the flag can be turned off.

The motor parameter can be a name or a number. The name "all" can be used to set the same parameter for all the motors.

The flag can be set with the mod command or with the mset command. The flag value can be read with the mparm command.

Examples:
mod tth 180 -180 ¡ tth ¡ 180
mod xyz off Turn off angle normalization on xyz motor.
mod xyz -1 Turn off angle normalization on xyz motor.
mod all 360 Set all motors to modulo 360

SEE ALSO

mset(cmd), mpar(cmd)

motor - move a motor

SYNOPSIS

motor mnum degrees [speed]

DESCRIPTION

This command moves a single motor to a angle given by the argument degrees. The command expects two arguments, mnum, the motor number and target angle in degrees. mnum may be expressed as either the motor number or the motor name. A third optional argument may be used to specify a motor speed.

SEE ALSO

MOVE(SUMMARY), mv(cmd), gmv(cmd), an(cmd), pl(cmd), dmotor(cmd)

mpar - print motor parameters

SYNOPSIS

mparm [p]

DESCRIPTION

This routine prints the motor parameters for all motors. The motor parameters are set with the mset command.

The motor parameters are easily changed by use of the ec (edit camac) command that edits the file /usr/super/lib/super˙init.

A list of pseudo-motor drivers is obtained with the command "mpar p".

SEE ALSO

ipar(cmd), mset(cmd), spar(cmd), hpar(cmd), ec(cmd)

mpk - move to peak position as calculated from moment

SYNOPSIS

mpk

DESCRIPTION

This command will move the motors to the peak position of the last scan as calculated from the moment. This command is executed automatically following a lup scan.

SEE ALSO

ppk(cmd)

mscan - motor scan

SYNOPSIS

mscan #mot mn1 start1 del1 [mn1 start2 del2 ...] npts mon

DESCRIPTION

This routine scans one or more motors. It is similar in function to ascan, but any motor may be scanned rather than the four diffractometer motors. The argument #mot gives the number of motors to be moved (maximum number = 3). The argument mn can be given as either the number or the name of the motor. The motor number, start, delta sequence can be repeated up to the maximum.

SEE ALSO

SCANS(SUMMARY), ascan(cmd), iscan(cmd), rscan(cmd), lup(cmd)

BUGS

The maximum number of motors is given by MAX˙MOT in define.h. The real limit, however, is MAX˙ARGS which is now set to 13.

mset - set motor parameters

SYNOPSIS

mset m# lab [drvname] slt chn acc spd bac bsp bstp cor ppd inh [format mod]

DESCRIPTION

This command sets the motor parameters. A prompt of the argument list may be obtained by typing mset with no arguments. The most common use for this command will be in the super˙init file that contains initial set-up parameters. The command is also used in the super˙vars file to save motor parameters from one run to the next. The list of motor parameters was derived from E500 motors in a Camac crate. Other motor drivers may use the parameters differently. A summary of the arguments follows (slot number ¡ 0 identifies a pseudo motor):

m# motor number, sequential from 1-NMOTORS

lab name of the motor

drvname Optional name of the motor driver (1st char must be a letter). If ommitted a default motor driver is assumed.

slt camac slot number

chn channel number (in the range 1-8 for E500 motors)

acc acceleration time in seconds

spd speed in pulses per second

bac backlash acceleration time in seconds

bsp backlash speed in pulses per second

bstp number of backlash steps (positive or negative)

corr number of correction steps (max=250) (see e500 instruction manual)

ppd pulses per degree, change to negative number to reverse sense of motor rotation

inh motor inhibit flag, 0 -¿ normal motor motion, 1 -¿ inhibit motor motion

format Output format in a "width.decimal" format, e.g. "8.3" (optional), if omitted, the last value entered is retained.

mod Modulo flag, 180, 360 or "OFF"

Pseudo motors are identified by a negative "slot" number. The E500 motor parameters are available to pseudo motors, but they may have different functions.

The motor parameters for all motors may be printed with the mparm command.

The software will attempt to take care of motor backlash if the parameter bstp (backlash steps) is non zero. In this case, the software drives the motors bstps past the target and then makes a final approach with a speed and acceleration given by bsp and bacc. (The corr parameter is a separate parameter that is internal to the E500 as described in the E500 manual.)

While mset can be executed directly from the keyboard, it is easier to use the ec (edit camac parameters) command to place the command in the "init" file that is read each time the program starts.

SEE ALSO

hset(cmd), sset(cmd), iset(cmd)

mv - move: move four diffractometer motors

SYNOPSIS

mv two-theta theta phi chi

DESCRIPTION

This command moves four diffractometer motors. If less than four arguments are given, the missing angles are assumed to be zero. The movement of any motor can be inhibited with the minh command.

To move only the first two motors, use the an command. To move only the last two motors, use the pl command. To move one motor only, use the motor or the dmotor command. To move all motors, not just the diffractometer motors, use the gmv command.

SEE ALSO

MOVE(SUMMARY), mv(cmd), gmv(cmd), pl(cmd), motor(cmd), dmotor(cmd), wh(cmd), minh(cmd).

To scan motor angles see ascan(cmd), mscan(cmd), and lup(cmd).

mz - move zone

SYNOPSIS

mz H1 K1 L1 H2 K2 L2

DESCRIPTION

This command calculates a diffraction zone defined by two vectors and moves phi and chi to the calculated value. (The command cz only calculates the zone.) The six arguments give the h, k, and l of two vectors used to calculate the diffraction zone.

The variable fv determines which hemisphere the calculated value of chi (or phi) falls in. (Two-circle mode only)

fv ¿= 0 —¿ -90 ¡ chi ¡ 90

fv ¡ 0 —¿ 90 ¡ chi ¡ 270

Chi = 90 is a special case where there is a +/- 180 degree degeneracy in phi. fv ¿= 0 —¿ -90 ¡ phi ¡ 90 (chi = 90)

fv ¡ 0 —¿ 90 ¡ phi ¡ 270 (chi = 90)

SEE ALSO

ZONE(SUMMARY), cz(cmd), fv(var)

om - orientation mode switch

SYNOPSIS

om

DESCRIPTION

This command is a two position switch. Typing the command changes the switch from one state to another. The switch is used to change from the "lattice parameters known" mode to to the "lattice parameter unknown" mode. In the former case two orientation reflections with known phi, chi and omega are required. In the "parameters unknown" mode, at least three orientation reflections with known two-theta, phi, chi and omega are required.

SEE ALSO

ORIENT(SUMMARY)

or - enter orientation parameters

SYNOPSIS

or i h(i) k(i) l(i) [t(i) w(i) phi(i) chi(i)]

DESCRIPTION

This command is used to provide the parameters associated with the orientation reflections. The argument i is the number of the reflection in the range 1-MAXOBS (currently MAXOBS=20). H, K, and L for the reflection must also be provided. If not included as arguments, two-theta, omega, phi and chi are read from the current motor position.

The orientation variables t(i), w(i), etc can also be entered individually using the "name=value" syntax.

To be used in the sample orientation the reflection number i must be 1 or 2 if the lattice parameters are known and in the range 1-nobs if the lattice parameters are not known.

Examples:

or 2 0 3 1 Orientation reflection 2, (031) reflection. Use current motor positions for two-theta, omega, phi and chi.

or 3 0 3 1 25.6 Orientation reflection 2, (031) reflection. two-theta = 25.6, omega = 0, phi = 90.4, chi = 5.0

w3 = 25.5 Change omega of the third reflection to 25.5 degrees.

SEE ALSO

ORIENT(SUMMARY), sor(cmd), gor(cmd), nobs(var)

p - print lattice params and orientation reflections

SYNOPSIS

p [flag]

DESCRIPTION

Print the current values of the lattice parameters and orientation reflections. This command normally sends output to the terminal. If an argument is supplied, the command will send output to the terminal, the data file and the line-printer file. If the lattice parameters or orientation parameters are changed, a scan execution will cause "p" (with an argument) to be called automatically.

SEE ALSO

ORIENT(SUMMARY), init(cmd)

pbr - powder bragg: move to tth/th calculated from hkl

This command is a special case of the br command.

phd - manually print scan heading

SYNOPSIS

phd [mon]

DESCRIPTION

Use this command to manually print a scan heading for user created scans, e.g. "scans" using the alias feature. The value given by the optional argument mon is printed in the heading. Column headings are printed using the format specified by the format command.

User-defined "scans" can be created by using a combination of motor movement commands and the ct command (with a third argument to suppress the column heading print-out). For example, suppose that scan table contains:

1 "phd 1e5"
2 "dmot phi .01"
3 "ct 1e5 no˙header"
The command "sc 1 (2 3)*10" will then create a phi motor "scan" ten steps long.

SEE ALSO

ct(cmd)

pkup - peak-up orientation reflections

SYNOPSIS

pkup r#, -mname˙1 del˙1, [-mname˙2 del˙2, ...] NPTS MON

DESCRIPTION

This is a "peak-up" routine which converts the arguments to a series of line-up or fpk scans at specified orientation reflections to tune-up the orientation matrix. The pkup comand can specify up to five scans to be perfomed at any or all of the orientation reflections. Following each scan, the motors are moved to the peak position as calculated from the first moment of the scan.

The number of the reflection, given by r#, may be a numeric value or zero. If r#=0, the list of scans is executed at all orientation reflections. The list of pairs of motor names and increments, -mname˙1 del˙1 mname˙2 del˙2 ..., specifies the order of execution of the scans. Each motor name (which can be a motor number or name) must have a "-" preceeding it. Pseudo motors are allowed in the list, but variables are not allowed. For a TH/TTH scan, use mname = 0.

Example: peak up reflection #5 using the phi motor first then a Theta-Two Theta scan, 11 points, 1e4 monitor:

pkukp 5 -phi 0.01 -0 0.10 11 1e4

The type of scan performed is determined by the variable pkflg. If pkflg = 0, each scan is a "lup" scan (data saved to disk). If pkflg = 1, each scan is an "fpk" scan (data not saved to disk). Each scan uses the same number of points and monitor. At the end of a series of scans, the routine executes an or command to assign new motor positions to that orientation reflection.

SEE ALSO

SCANS(SUMMARY), lup(cmd), fpk(cmd) pkflg(var)

pl - plane: move phi and chi motors

SYNOPSIS

pl phi chi

DESCRIPTION

This command moves the PHI and CHI motors only.

SEE ALSO

MOVE(SUMMARY), mv(cmd), an(cmd), motor(cmd).

To scan motor angles see ascan(cmd), mscan(cmd), and lup(cmd).

ppk - print peak position as calculated from moment

SYNOPSIS

ppk

DESCRIPTION

This command will print the peak position of the last scan as calculated from the moment.

SEE ALSO

mpk(cmd)

pr - preset a motor value

SYNOPSIS

pr mnum value

DESCRIPTION

This command sets a motor position to the desired value. The routine expects two arguments, motor number and position (degrees). The argument "mnum" can be either a motor name or a motor number as defined by the list given by the mpar command.

SEE ALSO

mpar(cmd), look(cmd)

pwd - print workding directory

See the cd command.

replot - replot scan

SYNOPSIS

rep

DESCRIPTION

Clear the screen and replot the data. Use era to erase the screen only.

SEE ALSO

erase(cmd)

res - restart program with new "vars" file

SYNOPSIS

res filename

DESCRIPTION

This command restarts the super program with a new vars file. There is one argument, the name of the vars file. This is useful if you want scans to use different lattice parameters for each scan, e.g. after changing the temperature. Set up several vars files with appropriate lattice parameters. Put "res filename" into the command list along with the scan parameters. The current filename and scan # are preserved after executing res.

restart - restart program with new "vars" file

SYNOPSIS

restart filename svars filename

DESCRIPTION

This command restarts the super program with a new vars file. There is one argument, the name of the vars file. The current parameters are first saved to the old "vars" file.

To copy the current "vars" file to a new file, use the svars command.

All file operations take place in the current working directory.

rl - reciprocal lattice

SYNOPSIS

rl [p]

DESCRIPTION

Print the current reciprocal and direct space lattice parameters Any argument causes the output to printed on the line printer and the data file as well as the terminal.

SEE ALSO

inv(cmd)

rm - unix "rm" command

See the unix section.

rscan - rocking scan centered on a Bragg peak

SYNOPSIS

rscan hkl mnum delta npts

DESCRIPTION

This routine scans one motor (or one hkl component) with the scan centered about the Bragg position given by hkl.

If a motor is to be scanned, use either the motor number or the motor name for mnum. If the motor number is "0", the scan is a theta/two-theta scan (radial scan) with the given increment assumed to be the two-theta increment. (The motor numbers and names can be listed with the command mpar.) To scan an Miller index, use the index name, e.g. k, instead of mnum

The parameter delta is either the motor increment (in degrees) or the hkl increment.

If the variable fbkg is non-zero the spacing between the data points will be a factor of three larger in the wings of the scan. For example, if fbkg = 0.5, the scan will have a spacing of 3 * delta for the first 25% of the scan, a spacing of delta for the second 50% of the scan, and a spacing of 3 * delta for the last 25%.

SEE ALSO

SCANS(SUMMARY), mpar(cmd)

rt - read cryostat temperature

SYNOPSIS

rt [p]

DESCRIPTION

Read the temperature (output in degrees and volts). An external program is called with the appropriate arguments. The output will appear on the terminal only if the variable t˙files = 1 and on the terminal, line-printer and data file if t˙files = 3. The letter "p" as a final argument will force the output to appear on the line printer and the data file (current call only).

SEE ALSO

TEMP(SUMMARY), st(cmd), dt(cmd), t˙files(var), tchk(var)

BUGS

Check to see that you can execute rt from the shell level before you try it from SUPER.

save - save program parameters

SYNOPSIS

save

DESCRIPTION

Re-write all files ("vars" file, "scans" file, flush data file). A save is performed automatically each time a scan command is executed.

sc - execute scan or command list

SYNOPSIS

scan scanlist

DESCRIPTION

This command executes a series of commands (usually scans) which have been previously defined (usually with the or the command). Arguments are supplied to the routine in the form of a scanlist on the command line. The scanlist consists of a series of arguments separated by any non-number (except the characters "(", ")", "*" and "-" which are special). The list of commands can be in arbitrary order, e.g

1 3 4 7 3 1 8 10 : A list of commands

The "-" character can be used to designate a sequential list in either forward or reverse order. Non-numbers (e.g. spaces) before or after "-" will negate its special meaning.

3-7 20-15 : scans 3-7 inclusive and 20-15 inclusive

5-10 : scans 5 and 10 only

The syntax "(...)*n" can be used to repeat a scan sequence n times. Non-numbers (e.g. spaces) are not allowed between ")", "*", and the repetition number.

(3,2,5)*4 : scans 3, 2, 5 repeated 4 times

Combinations of syntax are allowed

21 ((3-9 4)*3 22)*5

Invoking the sc command with no arguments will cause the previous scan list to be re-executed.

The routine performs a "dry-run" to check those commands which involve motor movements for possible limit violations.

A list of commands can also be executed with the ex command. The ex command is different in that a fixed list of commands is contained in a file and no "dry˙run" checking of limits is done.

SEE ALSO

ex(cmd), ds(cmd), es(cmd)

BUGS

You cannot combine ex commands in a list of commands exe- cuted with sc.

slots - execute commands in the "init" file

SYNOPSIS

slots

DESCRIPTION

The "init" file contains those commands (e.g. motor/scalar parameters) that do not often change. The "init" file is called on start-up and any variable assignment or command in the file is NOT updated each time the program is run. The command slots re-executes the the "init" file without stopping the program. Use the command ec to edit the "init" file and automatically run slots.

FILES

/usr/super/lib/super˙init

SEE ALSO

ec(cmd), mset(cmd), sset(cmd), hset(cmd)

sor - swap orientation reflection

SYNOPSIS

sor first˙refl second˙refl

DESCRIPTION

This command swaps the order of the orientation reflections in the list. This is useful when you are in the FRZ˙LAT = 1 mode (lattice parameters known) and you want to try several reflections in the orientation matrix.

SEE ALSO

ORIENT(SUMMARY), or(cmd), gor(cmd), nobs(var)

spar - print scaler parameters

SYNOPSIS

sparm

DESCRIPTION

This routine prints the scaler parameters for all scaler channels. The scaler parameters are set with the sset command. These are the scaler parameters used in the "normal" count mode (not the "intg" mode). The print flag is set by the format command.

The scalar parameters are easily changed by use of the ec (edit camac) command that edits the file /usr/super/lib/super˙init

SEE ALSO

ipar(cmd), sset(cmd), mpar(cmd), hpar(cmd), ec(cmd)

sset - set scaler parameters

SYNOPSIS

sset s# LAB SLT CHN OP NCH OP SCALE [DEC]

DESCRIPTION

This works similar to mset, but the scalar parameters are set. The argument list may be obtained by typing "sset" with no arguments. The arguments are as follows:

s# scaler number in the range 1-nscl.

LAB Scaler name, e.g. "CTS"

SLT Camac slot number

CHN Channel number starting with 1 as the lowest chan- nel.

OP Mathematical operator for normalization. Valid entries are i (identity operator), +, -, *, /

NCH Normalization channel. The counts in this channel are appended to the counts using the mathematical operator.

OP Mathematical operator for the scale factor. Valid entries are i (identity operator), +, -, *, /

SCALE scale factor. The scale factor is appended to the previous result using the mathematical operator.

DEC Number of digits in the output. (optional, if omit- ted the current value remains)

The complete mathematical operation for computing the scaler output is given by output˙cts = (measured˙cts OP cts[NCH]) OP SCALE where OP is one of the following:

i Identity. The identity operator does nothing. Actually, any unrecognized symbol is interpreted as the identity operator.

+, a, A Add

-, s, S Subtract

*, m, M Multiply

/, d, D Divide, if the number ¿0.

Example: To print out the counts (slot 17, channel 3) normalized by the monitor (channel 1), one could type

sset 3 CTS 17 3 / 1 * 1e6

Since the scaler output is in an integer format, the scale factor of 1e6 is included to keep the output value from being too small.

The logical place to use the sset command is in the super˙init file. This file can be edited using the ec command. The sset command is also easily incorporated into alias commands. For example, the attenuation could be set by an alias command:

atset-¿sset 3 CTS 17 3 i 0 * arg1

and

atoff-¿sset 3 CTS 17 3 i 0 i 1

To view the current scaler parameters use spar

SEE ALSO

spar(cmd), ec(cmd), nscl(var)

BUGS

The output is a long integer format. You can get some small numbers (zero) or negative numbers if you try.

st - set temperature

SYNOPSIS

st temp [p]

DESCRIPTION

Set the temperature (in degrees). An external program is called with the appropriate arguments. The output will appear on the terminal only if the variable t˙files = 1 and on the terminal, line-printer and data file if t˙files = 3. The letter "p" as a final argument will force the output to appear on the line printer and the data file (current call only).

SEE ALSO

TEMP(SUMMARY), sv(cmd), rt(cmd), dt(cmd), t˙files(var), tchk(var)

BUGS

Check to see that you can execute st from the shell level before you try it from SUPER.

sv - set voltage of temperature control diode

SYNOPSIS

sv volts [p]

DESCRIPTION

Set the temperature (in volts). An external program is called with the appropriate arguments. The output will appear on the terminal only if the variable t˙files = 1 and on the terminal, line-printer and data file if t˙files = 3. The letter "p" as a final argument will force the output to appear on the line printer and the data file (current call only).

SEE ALSO

TEMP(SUMMARY), rt(cmd), st(cmd), dt(cmd), t˙files(var), tchk(var)

BUGS

Check to see that you can execute sv from the shell level before you try it from SUPER.

sscan - save scan list

See the gs command.

svars - copy "vars" file to a new file

See the restart command.

title - enter experiment title

SYNOPSIS

title

DESCRIPTION

Each scan is labeled for later identification. The label is divided into two parts, the "title" and the "heading". The "title" does not change from scan to scan and can be used to identify the sample. The first execution of a sc command automatically prompts for a new "title". Subsequent executions of "sc" will prompt for a new "heading" only (if the variable enq=0). The "heading" can be used for those items which change for each scan.

This command provides for changing the "title" during the course of the experiment without re-starting SUPER.

Set the flag enq to zero to inhibit the request for a new scan heading each time sc is invoked.

SEE ALSO

enq(var), sc(cmd)

trans - transform coord. system of orientation matrix

SYNOPSIS

trans [nine matrix elements by row] utrans

DESCRIPTION

Use this command to transform the coordinate system of the orientation matrix. The vector comprised of h_i, k_i, l_i for each reflection in the orientation matrix is multiplied by the transform matrix to obtain a new h_i, k_i, l_i.

Examples:
rotate 45 deg. about the c-axis:
1 -1 0 1 1 0
0 0 1
interchange the a and the b-axis:
0 1 0 -1 0 0
0 0 1
double a-axis lattice parameter: 2 0 0
0 1 0
0 0 1

To enter the arguments one can either (1) enter trans and wait for the prompt or (2) enter the nine arguments on the command line. The matrix is entered by rows, three columns to a row.

The command utrans un-do a previous transformation in the event of an error.

SEE ALSO

in(cmd)

BUGS

Do not repeat the utrans command and expect to get the ori- ginal result. The command isn't that smart.

tt - Theta/Two-Theta: move two-theta and theta motors

SYNOPSIS

tt tth

DESCRIPTION

This command moves the TWO-THETA motor to the target position tion. It also moves the THETA motor to half the target value plus any offset specified by the frz command.

SEE ALSO

MOVE(SUMMARY), an(cmd), frz(cmd).

tw - Tweak a motor

SYNOPSIS

tw mnum delta [mon]

DESCRIPTION

Use this command to tweak a motor a given amount. The tw command expects two arguments, motor number (mnum) and increment size in degrees (delta). An optional third argu- ment (mon) gives monitor counts. If the third argument is present, the program counts before each tweak.

After entering the command line, type "p" (or "k") to move the motor positive and "n" (or "j") to move the motor negative.

It is preferred to use the tw command over the E505 "control pod" because the latter method results in lost motor pulses (Standard Engineering hardware problem).

SEE ALSO

mv(cmd), motor(cmd)

BUGS

The Unix version of Super accepts the letter command for the motor direction without waiting for a carriage return, the DOS version waits for a carriage return.

utrans - un-do the transform command

See the trans command.

unix commands

NAME

cat - unix "cat" command
cp - unix "cp" command
ls - unix "ls" command
rm - unix "rm" command
vi - unix "vi" command
pwd - unix "pwd" command
! - unix shell escape

SYNOPSIS

command [args]
! [args]

DESCRIPTION

A limited number of unix commands can be run directly from super without using a shell escape. These commands operate in the directory defined with the cd command.

Optionally one can use a shell escape "!" and invoke a command in the unix shell, .e.g "! ls *vars". The unix command starts with the second argument.

Two unix-style commands "pwd" and "cd" run inside super rather than as separate shell scripts.

SEE ALSO

cd(cmd)

v - Print version number

SYNOPSIS

v

DESCRIPTION

Print the version number of SUPER. The current revision of the manual is for version C/DOS V12.1

vi - unix "vi" command

See the unix section.

vlup - variable line-up scan

SYNOPSIS

vlup h k l variable˙name delta npts

DESCRIPTION

This routine scans a single variable in increments of delta. The scan is centered about the position calculated using the current value of the variable.

This command combines the functions of the vscan command and the lup command.

SEE ALSO

SCANS(SUMMARY), lup(cmd), vscan(cmd)

vscan - Scan variable at fixed hkl

SYNOPSIS

vscan h k l variable˙name start delta

DESCRIPTION

This routine scans the value of a named variable and re- calculates the motor angles corresponding to the given h,k,l. at each point. Arguments may either be entered on the command line separated by spaces or commas, or the user will be prompted for them. If the command is executed from a scan script and the parameters are not fully given on the command line an error message is printed and the command will be ignored.

This scan is useful for peak searches when the orientation matrix is not known. For example, to scan the orientation variable w1 from 0 to 45 in 9 steps at 1e5 mon while looking for the (002) peak, type "vscan 0 0 2 w1 0 5 9 1e5".

SEE ALSO

SCANS(SUMMARY), mscan(cmd), ascan(cmd), lup(cmd)

wait - wait for a given time interval

SYNOPSIS

wait no.˙of˙minutes

DESCRIPTION

Suspend program execution for a given number of minutes. This routine expects one argument (delay time in minutes). The resolution of the timing routine is one second.

This command is useful in conjunction with the temperature control routines.

SEE ALSO

st(cmd)

BUGS

This routine counts seconds by causing the real time clock (rtc) to count for successive one second intervals. There will be an accumulated error equal to the lag time between successive calls to the real time clock.

wh - where? print h,k,l and angles

SYNOPSIS

wh [p]

DESCRIPTION

Prints current location in the format defined by the format command. With no argument, output is to tty only. With any argument, output is to lpr and datafile.

Since the data output is in the format given by the format command, the position of all motors may not be printed. Use look to print the location of all motors.

SEE ALSO

look(cmd)

yname - change name of y-axis of plot

SYNOPSIS

yname
    yname variable˙name

DESCRIPTION

With no argument, this routine prints the current name of the variable to be plotted during a scan for a normal scan (iflag=0) and a "integrated" scan (iflag=1). If an argument is given, the name of the variable is changed if possible. The allowed variable names are given by the spar command (for normal mode) and the ipar command (for the "integrate" mode). If the given name matches one of the normal variable names, the variable to be plotted for a normal mode scan is changed. Variable name changes for an "integrate" mode scan work the same way.

Note that the variable aplot must be set to 1 for plotting to occur.

SEE ALSO

aplot(var), replot(cmd), spar(cmd), ipar(cmd), iflag(var)

18 Super Variables

a˙scale - auto-scale flag for plot

SYNOPSIS

a˙scale = 1
a˙scale = on
a˙scale = 0
a˙scale = off

DESCRIPTION

This is a flag that to tell the program to automatically calculate a new ymax if the data exceed the current value. Note that plots are only done if the flag a˙plot is set.

SEE ALSO

PLOT(SUMMARY), ymax(var)

afs - flag to control automatic file numbering

SYNOPSIS

afs = 1
afs = on
afs = 0
afs = off

DESCRIPTION

This flag turns on the automatic file numbering scheme as described in the help file file.cmd.

SEE ALSO

file(cmd)

as - reciprocal lattice parameter, a-star

SYNOPSIS

as =
  as = value
  bs =
  bs = value
  cs =
  cs = value
  al =
  al = value
  be =
  be = value
  ga =
  ga = value

DESCRIPTION

The lattice parameters are only supplied by the user in the "lattice parameter known" mode (fl = 1). In this mode the orientation matrix is determined by the six lattice parameters and two orientation reflections.

In the "lattice parameter unknown" mode the lattice parameters are calculated from a minimum of three orientation reflections.

The user can toggle between the "lattice parameters known" and "lattice parameters unknown" modes with the command om.

SEE ALSO

ORIENT(SUMMARY), om(cmd)

al - reciprocal interfacial angle, alpha-star

See variable as.

alm - alpha-mode (5-circle geometry)

SYNOPSIS

alm = value
  alm =
  bem = value
  bem =

DESCRIPTION

These variables are flags in the five-circle mode. Enter the five-circle mode with the command frz 5. The flags have the following actions:

alm=1 Alpha (table rotation angle) fixed

alm=2 Alpha (table rotation angle) set to give horizon- tal surface normal (using vertical-scattering plane).

bem=1 Fixed angle of incidence
bem=2 Fixed outgoing angle
bem=3 Incidence angle = outgoing angle

SEE ALSO

frz(cmd), b(var), b2(var)

amax - maximum number of attenuators

See command att

aplot - automatic plotting flag

SYNOPSIS

apl = 1
apl = on
apl = 0
apl = off

DESCRIPTION

Use this flag to implement automatic plotting of data during a scan.

SEE ALSO

erase(var), rep(var), a˙scale(var), box(var), ymax(var), xbot(var), xtop(var), ybot(var), ytop(var)

b - incidence angle (5-circle geometry)

SYNOPSIS

b = value
b =
b2 = value
b2 =

DESCRIPTION

These variables control the incidence and outgoing angles in the five-circle mode. The angles are set upon calculating a bragg angle using the convention defined by the variable bem.
bem=1 Fixed angle of incidence
bem=2 Fixed outgoing angle
bem=3 Incidence angle = outgoing angle
Enter the five-circle mode with the command frz 5.

SEE ALSO

frz(cmd), bem(var)

b2 - outgoing angle (5-circle geometry)

See variable b.

be - reciprocal interfacial angle, beta-star

See variable as.

bem - beta-mode (5-circle geometry)

See Variable alm

box - “box” flag for plot

SYNOPSIS

```
box = 1
    box = on
    box = 0
    box = off
```

DESCRIPTION

Set this flag to plot boxes around the data points in the plot. The plotting is faster if the flag is not set.

SEE ALSO

PLOT(SUMMARY)

bs - reciprocal lattice parameter, b-star

See variable as.

c1, c2, ... - chi of orientation reflection

See variable t1.

c˙tic - number of clock ticks per second

SYNOPSIS

```
c˙tic =
    c˙tic = value
```

DESCRIPTION

Set the clock rate for the time channel of the scaler. You can do this for any arbitrary scaler channel with sset.

SEE ALSO

sset(cmd), spar(cmd)

cryst - flag for crystallographic options

SYNOPSIS

cryst = 1
 cryst = on
 cryst = 0
 cryst = off

DESCRIPTION

This is a flag that to tell the program to invoke a number of options designed to enable SUPER to collect crystallographic data. If this option is set, the program automatically sets the flag afs to automatically number and open data files.

Normally one will collect crystallographic data using the ex command that executes commands in order from a file.

SEE ALSO

ex(cmd), afs(var)

cs - reciprocal lattice parameter, c-star

See variable as

efix - Constant/Variable energy flag

SYNOPSIS

efix = 1
 efix = on
 efix = 0
 efix = off

DESCRIPTION

The program will run with fixed incident energy (efix = 1 or "on") or variable energy calculated from the theta-angle of the monochromator (efix = 0 or "off"). In the efix = "on" mode, the incident energy is determined by the user-supplied wave vector, wv.

SEE ALSO

ENERGY(SUMMARY), cma(cmd), cme(cmd), mme(var), mds(var), wv(var)

enq - enquire flag for scan heading

SYNOPSIS

enq = 0
 enq = off
 enq = 1 enq = on

DESCRIPTION

Set **enq** = 1 (on) to force the program to ask for a new heading each time a scan is executed. Set **enq** = 0 (off) to use the same heading for every scan.

SEE ALSO

title(cmd)

fbkg - fraction of scan with expanded increment

SYNOPSIS

fbkg = value (0 ¡ value ¡ 1)
  fbkg =

DESCRIPTION

If this variable is non-zero, the wings of a scan will have a spacing between points that is a factor of three larger than the value specified in the scan command. The value of fbkg gives the percentage of the scan that is done in the expanded mode. For example, if fbkg = 0.5, the scan will have a spacing of 3 * delta for the first 25% of the scan, a spacing of delta for the second 50% of the scan, and a spacing of 3 * delta for the last 25%.

SEE ALSO

SCANS(SUMMARY), iscan(cmd), rscan(cmd), jscan(cmd), lup(cmd)

fflush - data buffering flag

SYNOPSIS

fflush =
  fflush = value

DESCRIPTION

If fflush = 0, normal data buffering is enabled. Data is written to disk after every scan or after the data buffer is filled (usually a large number of points.) Setting fflush to a positive value will cause the data to be written to disk after every fflush points of a scan.

fphi - polar angle for fixed azimuth fchi - polar angle for fixed azimuth

SYNOPSIS

fphi = value
  fphi =
  fchi = value
  fchi =

DESCRIPTION

Polar angles of the surface normal used in the fixed azimuth calculation mode.

SEE ALSO

frz(cmd)

fsn - file sequence number

SYNOPSIS

fsn =
  fsn = value

DESCRIPTION

This variable contains the current file sequence number used in the automatic file numbering mode (afs=1). When the file command is called or the data file gets larger than the limit set at compile time in the file options.h the sequence number is incremented and a new file is opened with fsn appended.

SEE ALSO

file(cmd), afs(var)

ga - reciprocal interfacial angle, gamma-star

See variable as.

gpib - camac slot number of KS 3388 GPIB controller

SYNOPSIS

gpib = value

DESCRIPTION

The Camac slot number of the Kinetic Systems 3388 GPIB controller. Usually this variable is set in the file "super˙init" and is changed with the ec command.

SEE ALSO

ec(cmd)

h1,h2,... - h-coordinate of orientation reflection

SYNOPSIS

h1, h2, ... =
  h1, h2, ... = value
  k1, k2, ... =
  k1, k2, ... = value

l1, l2, ... =
l1, l2, ... = value

DESCRIPTION

These are the h,k,l values of the orientation reflections. The variable nobs determines how many of the reflections are currently in use.

SEE ALSO

ORIENT(SUMMARY), nobs(var), or(cmd), t1(var)

k1,k2,... - k-coordinate of orientation reflection

See variable h1.

l1,l2,... - l-coordinate of orientation reflection

See variable h1

iflag - flag for "intg" (integrate) mode

SYNOPSIS

iflag =
  iflag = 1
  iflag = on
  iflag = 0
  iflag = off

DESCRIPTION

Flag for turning on the "intg" (integrate motors) option.

If iflag is set to 2, the "flip-flop" integration mode is enabled. In this case the start and end postions alternate to minimize motor movement.

SEE ALSO

INTG(SUMMARY), intg(cmd)

mds - monochromator d-star

SYNOPSIS

mds = dstar
  mds =

DESCRIPTION

2 PI / d (or 1/d) of monochromator. Used only in variable energy mode (efix = 1).

SEE ALSO

ENERGY(SUMMARY), cma(cmd), cme(cmd), mme(var), efix(var)

nfpk - default number of points in fpk scan

SYNOPSIS

nfpk =
 nfpk = value

DESCRIPTION

This parameter gives the default number of points in a fpk (find peak) scan. The number cannot be less than three.

SEE ALSO

fpk(cmd)

nhkl - number of parameters in the "HKL" list

SYNOPSIS

nhkl =
 nhkl = value

DESCRIPTION

The number of parameters in the "HKL" list. Any parameter in the HKL list can be included in the data output with the format command.

SEE ALSO

hset(cmd), hpar(cmd)

nmot - number of motors

SYNOPSIS

nmot =
 nmot = num˙mot

DESCRIPTION

This variable contains the number of motors in use. The maximum number is 16. Use mpar to view motor parameters and mset to set the parameters.

FILES The maximum number of motors is set in define.h. Change at your own risk.

SEE ALSO

mset(cmd), mpar(cmd)

nobs - number of orientation reflections

SYNOPSIS

nobs =
nobs = max˙refl

DESCRIPTION

This parameter gives the number of orientation reflections to be used. It may be changed with the "name = value" syntax. If FRZ˙LAT = 1 (lattice parameters known) NOBS is fixed at 2. If FRZ˙LAT = 0, NOBS must be ¿= 3 and less or equal to a maximum value set by MAXOBS (currently = 20).

SEE ALSO

ORIENT(SUMMARY), or(cmd), sor(cmd), gor(cmd)

nscl - number of scaler channels

SYNOPSIS

nscl =
nscl = num˙chan

DESCRIPTION

This variable contains the total number of scaler channels in use. The maximum number is 12, as given in define.h. Change at your own risk.

Use spar to view scaler parameters and sset to set the parameters.

SEE ALSO

sset(cmd), spar(cmd)

- scan serial number

SYNOPSIS

=
= new˙snum

DESCRIPTION

This variable is used to record or change the scan serial number which is incremented after each scan.

p1, p2, ... - phi of orientation reflection

See varibale p1.

pc - camac slot number of preset counter

SYNOPSIS

pc = value

DESCRIPTION

The Camac slot number of the preset counter. Usually this variable is set in the file "super'init" and is changed with the ec command.

SEE ALSO

ec(cmd)

pkflg - type of scan for pkup

SYNOPSIS

pkflg =
    pkflg = value

DESCRIPTION

This parameter gives the scan type, "lup" (pkflg=0) or "fpk" (pkflg=1) to be used in a pkup (peak up orientation reflection) scan list.

SEE ALSO

pkup(cmd)

q1,q2,... - wave-vector of orientation reflection

See variable t1.

t1,t2,... - two-theta of orientation reflection

SYNOPSIS

t1, t2, ... =
    t1, t2, ... = two-theta
    q1, q2, ... =
    q1, q2, ... = wave-vector
    u1, u2, ... =
    u1, u2, ... = theta
    w1, w2, ... =
    w1, w2, ... = omega
    p1, p2, ... =
    p1, p2, ... = phi
    c1, c2, ... =
    c1, c2, ... = chi

DESCRIPTION

These are the two-theta, omega, phi and chi values of the orientation reflections. The variable nobs determines how many of the reflections are currently in use.

When you enter tth, the wave-vector is automatically computed. When you enter theta, omega is automatically computed. These operations also work in reverse, so you can enter either tth or q and either th or omega. The orientation tion matrix will always be printed in the "tth", "th" format.

SEE ALSO

ORIENT(SUMMARY), nobs(var), or(cmd)

t˙files - output flag for cryostat temperature

SYNOPSIS

t˙files =
  t˙files = 1
  t˙files = 2
  t˙files = 3

DESCRIPTION

This controls the default output for cryostat temperature commands st , rt , and dt Depending on the value of t˙files, the output will appear as follows:
  1 Terminal only
  2 Terminal and line-printer only
  3 Terminal, line-printer and data file

SEE ALSO

TEMP(SUMMARY), st(cmd), rt(cmd), dt(cmd)

BUGS

Setting t˙files to any value other than 1, 2, or 3 would be unwise.

tcheck - check cryostat temp before each scan

SYNOPSIS

tcheck = 1
  tcheck = on
  tcheck = 0
  tcheck = off

DESCRIPTION

Use this flag to implement automatic temperature check before each scan. With this flag set the program does the equivalent of an rt p command before executing each scan.

DESCRIPTION

Maximum value of y-coordinate of plot made during a scan. If the flag a˙scale is set, the program will pick a value for ymax and change it as needed.

SEE ALSO

PLOT(SUMMARY), a˙scale(var), a˙plot(var), rep(cmd), erase(cmd)

ytop - maximum y-coord. of plotting area (Unix only)

See variable xbot.

19 Revision Info

This is *Revision* : 1.1 of the HTML web page for the Super 12.7 Documentation.