

NAME

changes – highlights of modifications for spec release 2.15+

DESCRIPTION

These notes summarize the incremental patches made to spec release 2.15 (released February 21, 1992), as of April 19, 1993.

INCOMPATIBILITIES**String Value Of Uninitialized Variables**

A long standing inconsistency (bug) dealing with the string value of an uninitialized variable has been corrected. The string value of an uninitialized variable is now the null string, "". Previously, the string value would have been "0". This change makes the values of uninitialized variables consistent with `awk`. The standard macros have been modified in several places to take account of this new behavior. If you are currently using expressions such as

```
if (some_name == "0") ...
```

to test whether a variable is uninitialized, you should change such expressions to

```
if (some_name == "") ...
```

to work with this new version of `spec`.

NEW FEATURES**Two High-Res Plot Windows**

The `plot_cntl()` function can now be used to create and select between two high resolution plot windows. See the `funcs` help file for more information.

Multiple Curves On A Single Plot

The `data_plot()` function can now be used to plot multiple curves with one call. See the `data` help file for more information.

Including Local Help Files

Local help files can be included in this list of topics in the basic help file by including the local help file names in a file named `SPECD/help/.local`. The basic help file is only updated during a full installation of `spec`.

Macro To Configure Counters

A new macro named `counters` displays the current counter configuration and then prompts for counter channel numbers for the global variables `MON` and `DET`. Those variables hold the channels to be treated as monitor and detector in the counting and scan macros. The monitor and detector channel numbers can also be given as arguments to `counters`.

Reusing Macro Names As Local Variables

`spec`'s syntax has been changed to allow macro names to be reused as local symbols. Previously, input such as the following would not be allowed:

```
def d 'print date()'
{ local d ; d = 2 ; print d }
```

In the current version, reuse of the symbol `d` is accepted.

Can Use `.log` Extension For Log Files

Previously, `log` files (files that recorded all of `spec`'s output) were any files with names that began with the three characters `log`. Now files ending with the four characters `.log`, are also used as log files.

Debugging Log Files

A new type of *log* file is available. If an output file name begins with the characters *dlog* or ends with the characters *.dlog*, all of spec's output will be sent to that file, as with normal *log* files. In addition, if any debugging output is being generated (by assigning a value to the built-in `DEBUG` variable), the debugging output will only be sent to the *dlog* file and won't be printed on the screen or sent to any other files.

New Function To Prompt For User Input

A new function named `getval()` is available for reading values from the keyboard. See the *funcs* help file for usage information. Macros that use the slightly awkward `getvar` macro can be rewritten with this new function.

New Random Number Function

A new function named `rand()` returns random numbers. See the *funcs* help file for usage information.

New \a Attention Character

A new special character sequence, `\a` (for attention or audible alert), has been added. This sequence sounds the terminal bell (and appears to be part of the ANSI C grammar).

MCA No Longer Automatically Cleared

spec will no longer clear the LeCroy 2301, 3512 and 3521 MCA modules when counting using `tcount()` or `mcount()`. You will have to use the `mca_par("clear")` function before counting to clear the MCA channels. This change is made to accommodate the needs of some users.

Multiple Processes Can Share GPIB Controller

On System V UNIX operating systems with IPC (inter-process communication), it is now possible to share one National Instruments GPIB controller among multiple versions of spec running on the same computer. In addition, it is also possible to share a single KS 3988 GPIB-to-CAMAC controller among multiple processes. The IPC version of the GPIB controller is selected by the name `PC_GPIBPC_L` in the *config* file.

Multiple Versions Of 386/486 NI GPIB Supported

Both the National Instruments PCII and AT-GPIB cards and drivers are now supported within the same binary for 386 systems. The former is chosen using the `PC_GPIBPC` (or `PC_GPIBPC_L`) and the latter using `PC_GPIBPC2` (or `PC_GPIBPC2_L`) in the *config* file.

Warning: Selecting the wrong controller type may crash your computer or damage your file system.

Multiple Versions of Sun NI GPIB Supported

Three versions of the National Instruments GPIB drivers are now available for Sun platforms. In the *config* file, the keywords `PC_GPIBPC` (or `PC_GPIBPC_L`) select the SB-GPIB Version 1.3 driver, the keywords `PC_GPIBPC2` (or `PC_GPIBPC2_L`) select the GPIB 1014-1S driver and the keywords `PC_GPIBPC3` (or `PC_GPIBPC3_L`) select the SB-GPIB Version 2.1 driver. **Warning:** Selecting the wrong controller type may crash your system.

Support For ACS MCB-4 Motor Controller

Preliminary hardware support is available for the Advanced Control System MCB-4 stepping motor controllers over RS-232 and GPIB interfaces.

Support For NSLS MMC32 Motor Controller

Preliminary hardware support is available for the MMC32 stepping motor controller (produced locally at NSLS) over a GPIB interface.

Support For Oriel Encoder Mike 18011 Motor Controller

Preliminary hardware support is available for the Oriel Encoder Mike 18011 controller over an RS-232 interface.

Support For Klinger Motion Master 2000

Preliminary hardware support is available for the Klinger Motion Master 2000 DC and stepper motor controller over a GPIB interface. See the *kl2000* help file for more details.

Support For the *kappa* Geometry

Preliminary support for the *kappa* diffractometer geometry is included.

Support For ESRF

Support for the distributed computing, VME-based hardware environment of the European Synchrotron Radiation Facility (ESRF) is now included. A general hardware control function `esrf_io()` has been added to `spec` (see the *esrf* help file) and support for the ESRF motor controllers is available.

Allowing For Slop In E500 Motor Positions

An experimental new feature is available to deal with small discrepancies between the E500 controller and `spec`. The function `motor_par(motor, "slop", steps)` sets a threshold value for the selected motor for which `spec` will automatically use the E500 value as the correct position. A message is printed if such a correction occurs. The values set are saved in the state file.

CHANGED FEATURES**New Definition For the `scan_plot` Macro**

The `scan_plot` macro is now defined in the standard macros as `_plot`, which is a newly introduced name. The `setplot` macro will change the definition of `_plot`, rather than `scan_plot`. Local sites that wish to redefine `scan_plot` no longer need to redefine the `setplot` macro.

New `get_angles` Macro

Occurrences of the built-in command `getangles` in the standard macros have been replaced with the macro `get_angles`. This new macro contains the `getangles` command, along with invocation of a new macro named `user_getangles`. The latter is initially defined as a null macro.

New `get_counts` Macro

The built-in command `getcounts` in the standard macros has been replaced with the macro `get_counts`, which calls `getcounts` and a new macro named `user_getcounts`, which is initially defined as a null macro. The purpose of this change is to allow the user to insert code, if necessary, after the call to `getcounts` either to postprocess the counter values, or to implement user-level counters. For one particular configuration, the distance of the detector from the sample varies with the cosine of an angle, so `user_getcounts` is defined to multiply `S[DET]` by the appropriate factor.

New `count_em` Macro

A new `count_em` macro replaces all the code and immediate logic surrounding calls to `mcount()` and `tcount()`. Included in `count_em` are calls to new macros named `user_precount` and `user_postcount`. By default, these macros are defined as null macros. They can be given definitions by users for various purposes, including implementation of user-level counters. See the *count.mac* source file for details.

Orientation Matrix Code Accommodates Six Angles

The maximum number of angles used by the orientation-matrix code to describe each reflection has been increased from five to six, to accommodate the new six-circle geometry.

Increase In the Number Of Data Groups

The number of data groups has been increased from eight to 256. A limit has been placed on the total number of points allowed for all groups. That limit is currently 65,536 points, but can be increased by CSS if requested.

Data Groups Can Be Deleted

Calling `data_grp()` with the number-of-elements argument set to zero will now delete the group.

Renaming Of Background-Flag Variable

The flag for background subtraction in the macros, defined in the source file *plot.mac*, has been renamed `BG`. Previously the name of the flag was `PG`.

New Measure Macros

The `measuretemp` macro, called after counting in the `_loop` macro, has been replaced with two new macros called `measure1` and `measure2`. Initially, `measure1` is defined as `measuretemp` to maintain compatibility with prior usage. Also introduced in the `_loop` macro is a new `measure0` macro that is called before counting. Both `measure0` and `measure2` are initially defined as null macros.

Macros Work Without A Monitor Counter

The standard scan and counting macros have been modified so that they can still be used if there is no monitor counter configured. Set the global variable `MON` to `-1` if you don't have a monitor counter.

IMPROVED FEATURES**No More Need For So Many Underscores**

The initial underscores in the names of many of the local symbols in the standard macros have been removed in order to improve the readability of the macros. The underscores no longer serve a purpose due to the improvements in the functionality of the `local` keyword.

Improvements To Output File Handling

A number of changes have been made to improve the handling of output files.

- 1) `spec` now keeps track of the current working directory when each file is opened. If you exit `spec` and then restart it from a different directory, the existing output files will be reopened in their correct directories.
- 2) If you restart `spec` using an existing state file and there is an output file present in the state file that has disappeared from the file system, `spec` now prints a warning message. If possible, `spec` will create a new file using the previous name and directory, though.
- 3) The `on()` and `open()` functions, when used without arguments, now also print the full path name of the files, along with the name by which each was opened.
- 4) When the `close()` function is used with an unopened file, `spec` no longer prints an error message.
- 5) Previously, if you removed a `spec` output file while `spec` still had it open, all subsequent output to the file would be lost until the file was closed, either through the `close()` function, by exiting and restarting `spec` or through the `fixstate` command. `spec` will now detect when you attempt to write to an open file that has been removed from the file system. `spec` will print a warning message, close the old file descriptor, create a new version of the file and open that file for writing. As a general rule, it is a bad idea to use any other UNIX utility to change a file currently opened for writing by any program, including `spec`.
- 6) The previous versions of the `newfile` macro opened the data file in a directory named `./data` if it existed. A new global variable named `DATA_DIR` is now used to hold the name of the optional data directory. `DATA_DIR` is initially set to `./data`, so the default behavior is as before.

- 7) If the file you enter in the `newfile` macro has the same name as the current data file, but if you have changed directory, the previous file will be closed and a new one opened in the current directory.
- 8) If the `newfile` macro can't open the file asked for, the `DATAFILE` variable is set to `"/dev/null"`.
- 9) The `newfile` macro now prints warning messages if the file already exists, if the file exists and doesn't have the `#F` characters at the start of the file or if the file name given is the same as the current `DATAFILE` but doesn't exist. This feature uses a program called `chk_file`, which is installed in the `SPECD` directory.

Monochromator Support Revamped

The monochromator support has been revamped. All of the functionality is now in macro code. The `geo_mono.c` module has been eliminated. All of the monochromator macros are in the macro source file `macros/energy.mac`. One-, three- and four-motor monochromators are supported. The macros automatically detect which type of monochromator is being used based on the motor mnemonics present in the `config` file. The `get_E`, `set_E` and `move_E` macros have been renamed `getE`, `setE` and `moveE`, respectively. See the new `mono` help file for details.

CAMAC Info In `install_data` File

The `-C` flag to the `Install` program has been replaced with a line in the `install_data` file that keeps track of whether or not to include CAMAC support.

Non-Super User Installation

The `-S` option to the `Install` program that lets you install `spec` without being super user as long as you have write permission on all the relevant files has been fixed to work on most, if not all, systems. Previously, the call of the `chown` UNIX command caused problems on some systems where it was only available to the super user. Now, the `-S` option disables all calls to `chown`.

Driver Files Can Be Installed Without Kernel Rebuild

The driver install scripts for the 386 systems now accepts a `-n` option to indicate not to rebuild the kernel after installing the driver files.

Improvements to `surf` Geometry Code

The code in `geo_surf.c` that determines which version of the available surface diffractometer configurations to use based on the configured motor mnemonics has been improved to check that all the necessary mnemonics are present, rather than just looking for one or two of them. Also, the code now does better error checking for mathematical singularities. In addition, the calculations between motor positions and reciprocal space now work correctly in all the available modes.

Improvements To `fourc` and `fivec` Code

The `fourc` and `fivec` geometry code has a few more checks for unobtainable reflections in the **azimuthal** modes.

Changes To `twoc` Internals

The numbering of the `Q[]` parameters for the `twoc` geometry has been changed to be consistent with the other geometries. The indices of the `LAMBDA`, `ALPHA` and `BETA` parameters have all been increased by one. `Q[2]`, formerly `LAMBDA`, is now unused.

Improvements to `bug` Macro

The `bug` macro now uses the value of the new global variable `MAIL` for the name of the mail program to run. The default value is the string `mail`.

Faster, More Robust High-Res Graphics

The interface code to the high-resolution graphics filters has been rewritten for improved speed and robustness.

X11 Graphics Automatically Brought To Front

The *x11filt* program will now automatically raise the window to the front each time its contents are changed. This feature can be disabled in your *.Xdefaults* file by setting the *spec.AutoRaise* parameter to off.

X11 Graphics Checks For Backing Store

The *x11filt* code has been fixed to automatically detect whether “backing store” is available. If backing store is requested when it is not available, the code switches automatically to retained pixmap mode.

Am9513 Counting Improved

The code for the Am9513 based counting boards for the PC has been improved. The maximum count has been increased from from 10.9 minutes to 71.5 minutes. When counting to time, the time base resolution (in seconds) is now set according to the following table:

0.00001	for $t < 0.6$ sec
0.0001	for $t < 6$ sec
0.001	for $t < 60$ sec
0.01	for $t < 655.35$ sec (10.9 min)
0.0655	for $t < 71.5$ min

When counting to monitor counts, the 0.01 second time base is used, and the value returned for the time channel will be corrected to account for the rollovers that occur every 655.36 seconds. Also, the powder-mode counting mode used by the *setpowder* macro is now implemented.

MCU Code Improvements

The MCU code now only sends values for the base rate, steady-state rate and acceleration parameters when *spec* starts, when the *reconfig* command is executed or when you change the steady-state rate with the *motor_par()* function. Previously, these values were sent before each move.

Compumotor 4000 Code Updated

The code for the Compumotor 4000 has been updated to work with the revised PROMS that Compumotor is now shipping.

BUG FIXES**X11 Color Names Updated**

The color definitions have been updated to agree with the X11R4 values. Of the two (out of 175) color names used within the *spec* X11 filter that are no longer defined in X11R4, one, unfortunately, was the default background color for the *splot* and *rplot* screens, resulting in a black text on a black background.

Bugs in Assigning Values to Built-in Variables Fixed

It is no longer legal to assign string values to built-in number variables such as *COLS* or *ROWS* or number values to built-in string variables such as *TERM* or *GTERM*. Attempts to do so will generate a warning message, but will not cause *spec* to reset to command level.

Bugs Fixed in *edconf*

A bug in the *edconf* program, where deleting the highest numbered motor resulted in an incorrect display, has been fixed.

Bug Fixed in `ca_fna()`

A bug in the `ca_fna()` function, where it required an unnecessary fourth argument for command-type F codes (those with bit 3 set), has been fixed.

Bug in Producing Lp-Style Plots Fixed

A bug in release 2.15, where the lp-style plots would not be produced if high-resolution graphics were enabled, has been fixed.

Bug in `begin_mac` Fixed

A bug in the implementation of the `begin_mac` feature has been fixed. Previously, `begin_mac` wouldn't execute unless a startup command file, such as *site.mac* or *spec.mac*, was read. Also, the message (running `begin_mac ...`) is now printed on the screen when `spec` starts up if the `begin_mac` macro is defined.

Bug in `data_plot()` Fixed

A bug in the `data_plot()` function, where if the number-of-points parameter was not zero, the plots weren't made, has been fixed.

Bugs in `move_poll()` Implementation Fixed

The way in which the `move_poll` macro was added to the motor moving macros in release 2.15 broke the updated display of motor positions in the updated-moving macros. The macros that use `move_poll` have been rewritten so that the updated display will now work. The change involves reorganizing all the motor moving macros along the lines of

```
def mv  '_mv $*; move_poll'  
def umv '_mv $*; _update1 $1'
```

where the new `_mv` macro contains what used to be in `mv`.

Bugs in CAMAC MCA Code Fixed

Bugs in the code for LeCroy 2301 and 3588 MCAs that handled large numbers incorrectly have been fixed.

Bugs in `scans.4` Fixed

A bug, introduced in the last version of *scans.4.c*, where the error bars would be calculated incorrectly for a point with zero detector counts, has been fixed.

Another bug in *scans.4* (introduced in the January 1992 revision), where when #C comment lines contained numbers in columns that matched the data, the function included those numbers in the data that was returned, has been fixed.

Bug in E500 Acceleration Parameter Interpretation Fixed

An inconsistency concerning the interpretation of the units of the CAMAC E500 stepping motor controller acceleration parameter has been clarified. The E500 has an eight-bit register for each motor that contains the acceleration time in tens of milliseconds. Valid values for that register are from zero to 255, which translate to valid acceleration times from 10 milliseconds to 2.55 seconds. Since the very first versions of `spec`, the values given in the *config* file have been copied directly to the E500 registers. However, `spec`'s documentation has always stated that the value in the *config* file was in milliseconds. Thus, the default value of 125 translated to 1.25 seconds when the number was copied directly to the E500 register. With this version, `spec` now properly treats the values in the *config* file as milliseconds. `spec` also prints a warning message if the values in the *config* are outside the legal range. A negative value causes a zero to be written to the acceleration register, which is a valid acceleration time on the E500. `spec` does not allow zero to be entered directly, though. You may want to check all the acceleration parameters for your motors after you install this version of `spec`. In particular, if you have values between 1 and 255 that seemed suitable for your motors, you should change those values by multiplying by ten.

Bug in MCU Acceleration Parameter Interpretation Fixed

An error concerning the acceleration (ramp) parameter for the ACS MCU motor controller has been fixed. The values sent to the MCU were previously ten times greater than the values requested in the *config* file. Since the MCU often has communication problems during the ramp times at the start and end of a move, the long ramp times increased the chances of communication errors.

Bug in DSP and OMS Driver Installation Fixed

The *install* scripts for DSP CAMAC driver and OMS motor driver have been fixed to correctly set the IO-port end address in the kernel configuration files on AT&T System V/386 installations. Previously, if you changed the base address from the default using a command line option, the end address was not updated correctly.

Bug in Macro Installation Fixed

A bug in the installation procedure that put the orientation matrix macros in *standard.mac* rather than the appropriate configuration macro files (*four.mac*, etc.) has been fixed. If the last macro package installed didn't use the orientation matrix macros, the orientation matrix macros would not be put in *standard.mac* and thus would not be available to a previously installed geometry configuration that required them.

Bugs in Ortec Counting Fixed

Many changes have been made for the Ortec 974/994/995/997 modules, (Apparently the previous code wasn't working as well as CSS was led to believe.)

Microvax GPIB Bug Fixed

A possible bug with the code for the National Instruments GPIB controller for the DEC Microvax Q-bus dealing with non-terminated reads has been fixed.

Simulate-Mode Bug Fixed

A bug in simulate mode that occurred when the change to simulate mode took place before any motors had been moved has been fixed. The manifestation of the bug was the appearance of strange values for motor positions. The bug only occurred on some computer platforms and only affected the behavior in simulate mode.

dofile() Bug Fixed

A bug of longstanding, where *spec* would hang if the file used with *dofile()* or *qdofile()* was empty, has been fixed.

Bugs in *edconf* Wizard Mode Fixed

The main hole in the motor security features available with the *wizard* option of the *edconf* program (dealing with reading in backup files that describe a less secure configuration) has been plugged up.