

**NAME**

sixc – 6-circle geometry modes

**DESCRIPTION**

The *sixc* configuration of *spec* operates a standard six-circle diffractometer. The six-circle diffractometer combines the features of a four-circle and a z-axis diffractometer.

**CONVENTIONS**

The standard six-circle motor names and mnemonics are as follows:

Delta	del	– Detector arm rotation
Theta	th	– Rotates sample circles
Chi	chi	– Sample tilt
Phi	phi	– Sample rotation
Mu	mu	– Diffractometer rotation
Gamma	gam	– Out-of-plane detector rotation

The following optional motors select among various pseudomotor configurations (see the following section on pseudomotors).

GamTran	gamT	– Translation component of gam
GamRot	gamR	– Rotation component of gam
GamScrew	gamS	– Screw displacement for gam
MuTran	muT	– Translation component of mu
MuRot	muR	– Rotation component of mu

Two configurations of the diffractometer are available. At synchrotrons, the configuration is dictated by the layout of the experimental hutch with respect to the incident X rays. The *setconfig* macro selects the configuration. The differences between the configurations are described further down, as is the rotation sense and zero position of each circle.

**SIX-CIRCLE MODES**

The six-circle modes represent various constraints that can be made on the transformation equations from reciprocal space to diffractometer angles. Use the *setmode* macro to select a six-circle mode. The modes are as follows:

- 0 – *Omega* fixed ( $\mu = \gamma = 0$ , traditional four circle)
- 1 – *Phi* fixed ( $\mu = \gamma = 0$ , traditional four circle)
- 2 – *Zone* ( $\mu = \gamma = 0$ , traditional four circle)
- 3 – *Azimuth*,  $\mu$  and  $\gamma$  fixed
- 4 – *Alpha*,  $\mu$  and  $\gamma$  fixed
- 5 – *Beta*,  $\mu$  and  $\gamma$  fixed
- 6 – *Azimuth* and  $\gamma$  fixed,  $\mu$  varies
- 7 – *Alpha* and  $\gamma$  fixed,  $\mu$  varies
- 8 – *Beta* and  $\gamma$  fixed,  $\mu$  varies
- 9 – *Azimuth* and  $\mu$  fixed,  $\gamma$  varies
- 10 – *Alpha* and  $\mu$  fixed,  $\gamma$  varies
- 11 – *Beta* and  $\mu$  fixed,  $\gamma$  varies
- 12 – *Z-Axis* with *azimuth* fixed,  $\chi$ ,  $\phi$  set to  $-\sigma$ ,  $-\tau$
- 13 – *Z-Axis* with *alpha* fixed,  $\chi$ ,  $\phi$  set to  $-\sigma$ ,  $-\tau$
- 14 – *Z-Axis* with *beta* fixed,  $\chi$ ,  $\phi$  set to  $-\sigma$ ,  $-\tau$
- 15 – *Specular* with  $\theta = 90$ ,  $\gamma = 0$  and  $\phi$  fixed
- 16 – *Chi*,  $\phi$  and  $\mu$  fixed

In the *azimuth*-fixed modes (3, 6, 9 and 12), when the azimuthal angle is set to 90 degrees, the incident angle *alpha* will be equal to the exit angle *beta*.

In the *z-axis* modes, use the `sigtau` macro to calculate the azimuthal reference vector from the *chi* and *phi* angles used to align the sample.

### FROZEN ANGLES

Each of the six-circle modes has associated an angle (or angles) that are fixed in that mode. As a convenience, there is a `freeze` macro that turns on `spec`'s *frozen* mode. When in frozen mode, all moves that specify reciprocal space coordinates will use the frozen value(s) of the current mode's fixed angle(s). Thus you can move motors around for lining up, etc., but have the the fixed angle(s) automatically restored to the frozen value when doing reciprocal space scans, `br` moves, etc. The `unfreeze` macro turns frozen mode off.

The frozen values are stored in variables (actually macros) named `F_OMEGA` (omega-fixed mode), `F_CHI_Z` and `F_PHI_Z` (zone mode), `F_PHI` (*phi*-fixed modes), `F_CHI` (*chi*-fixed modes), `F_AZIMUTH` (*azimuth*-fixed modes), `F_ALPHA` (*alpha*-fixed modes), `F_BETA` (*beta*-fixed modes), `F_MU` (*mu*-fixed modes) and `F_GAMMA` (*gamma*-fixed modes). You can assign values to these variables by hand, or you can pass the values as arguments to the `freeze` macro. For example, if you are in mode 4 (*azimuth*, *mu* and *gamma* fixed), typing

```
freeze 90 0 10
```

will set `F_AZIMUTH` to 90 degrees `F_MU` to zero and `F_GAMMA` to 10 degrees. Without arguments, the `freeze` macro freezes the corresponding angles at their current values.

### SIX-CIRCLE SECTORS

(The six-circle sector code is still in development.)

### CUT POINTS

Cut points affect the direction the diffractometer circles turn to get from one position to the next. For example, if a cut point is at zero, the corresponding circle will only move through angles of 0 to 360 degrees. Thus, to get from 355 (=−5) to 5 degrees, the circle will turn 350 degrees. If a cut point is at −180, the circle will move through angles from −180 to 180. Thus the same motion from −5 to 5 will require only 10 degrees of movement. Use the `cuts` macro to select cut points for *theta*, *chi* and *phi* (and the *azimuth*). The *delta*, *mu* and *gamma* motors always use −180 as the cut point. (Only the sign of the *azimuth* cut point is used, and it determines the sign of the *azimuth* angle.)

### LATTICE PARAMETERS

The `setlat` and `setrlat` macros let you set the lattice parameters in direct space or reciprocal space, respectively.

### ORIENTATION MATRIX

With known lattice parameters, you need to find two (nonparallel) reflections in order to determine the orientation matrix. The macros `or0 H K L` and `or1 H K L` are used to associate the current diffractometer angles with the primary and secondary reflections. Macros called `setor0` and `setor1` also let you set the orientation reflections, but don't require you to move the diffractometer to the correct position. Instead, you are prompted for each of the angles and values for H, K and L. The `or_swap` macro is available to exchange the primary and secondary reflections.

You can also fit an orientation matrix if you have three or more known reflections (not all in the same plane), but unknown lattice parameters. Use `reflex_beg` to initialize a reflections data file. Use `reflex H K L` to add the current reflection to the file. Use `reflex_end` to complete the file. You then run the reflections file as a `spec` command file in order to fit the reflections to obtain a new orientation matrix. To calculate lattice parameters from the new orientation matrix, type `calcL`. You can then display the lattice parameters with the `pa` macro. You can edit the reflections file to delete or modify any of the reflection data there.

## PSEUDOMOTORS

spec supports configurations where the *gamma* detector rotation and/or the *mu* diffractometer rotation involves multiple motors or is done indirectly. In one *gamma* configuration, the detector is mounted on a small rotation stage that tracks the desired gamma angle. Another motor translates that stage to keep the detector oriented towards the sample. That configuration is selected by configuring two motors having mnemonics `gamR` and `gamT` in the *config* file.

In another *gamma* configuration, the rotation is created by translating the detector along a curved track using a linear screw pivoted at one end. The screw motor must have the mnemonic `gamS`. In both configurations, a motor with mnemonic `gam` must also be configured, but with the controller set to `NONE`.

In another configuration, the *mu* rotation is made up of a rotation and a translation, just as described for *gamma*, above. This configuration is selected when there are motors having mnemonics `muR` and `muT` in the *config* file. The motor configured as `mu` must have the controller type set to `NONE`.

In normal operation, a “tracking” mode is on such that commands to move the pseudomotors `gam` or `mu` are automatically converted to motions of the associated real motors, and the positions of the pseudomotors are calculated based on the positions of the real motors. With “tracking” mode off, the real motors will be unconnected from their associated pseudomotor. Commands to move the pseudomotors won’t affect the real motors, and the real motor positions won’t change the value of the pseudomotors. “Tracking” mode is stored in `g_track`. The `ontrack`, `offtrack` and `settrack` macros can be used to set `g_track`. The “tracking” macros will apply to both *gamma* and *mu* pseudomotors together.

(The *mu* pseudomotor was added in spec release 5.10.01-6.)

## GLOBALS AND MACROS

`Q[ ]` – a built-in array which holds the following six-circle parameters:

```
def H 'Q[0]' – Reciprocal space coordinate.
def K 'Q[1]' – Reciprocal space coordinate.
def L 'Q[2]' – Reciprocal space coordinate.
def LAMBDA 'Q[3]' – Wavelength of X rays.
def ALPHA 'Q[4]' – Incident angle, useful for surface diffraction.
def BETA 'Q[5]' – Exit angle, useful for surface diffraction.
def OMEGA 'Q[6]' – The theta - two-theta / 2 parameter.
def TTH 'Q[7]' – The Bragg angle.
def AZIMUTH 'Q[8]' – Rotation angle of a reference vector about the scattering
vector, useful for surface diffraction.
def SIGMA 'Q[9]' – With TAU, an alternative description of the azimuthal refer-
ence vector.
def TAU 'Q[10]' – With SIGMA, an alternative description of the azimuthal refer-
ence vector.
def F_ALPHA 'Q[11]' – Frozen value of ALPHA for alpha-fixed mode.
def F_BETA 'Q[12]' – Frozen value of BETA for beta-fixed mode.
def F_OMEGA 'Q[13]' – Frozen value of OMEGA for omega-fixed mode.
def F_AZIMUTH 'Q[14]' – Frozen value of AZIMUTH for azimuth-fixed mode.
def F_PHI 'Q[15]' – Frozen value of A[phi] for phi-fixed modes.
def F_CHI_Z 'Q[16]' – Frozen value of A[chi] for zone mode.
def F_PHI_Z 'Q[17]' – Frozen value of A[phi] for zone mode.
def F_MU 'Q[18]' – Frozen value of A[mu] for mu-fixed modes.
def F_GAMMA 'Q[19]' – Frozen value of A[gam] for gamma-fixed modes.
def CUT_AZI 'Q[20]' – The azimuth cut point flag.
```

```

def CUT_DEL 'Q[21]' - The delta cut point.
def CUT_TH 'Q[22]' - The theta cut point.
def CUT_CHI 'Q[23]' - The chi cut point.
def CUT_PHI 'Q[24]' - The phi cut point.
def CUT_MU 'Q[25]' - The mu cut point.
def CUT_GAM 'Q[26]' - The gamma cut point.
def F_CHI 'Q[16]' - Frozen value of A[chi] for chi-fixed, phi-fixed, mu-fixed
mode.

```

G[] – a built-in array which holds the following six-circle parameters:

```

def g_mode 'G[0]' - Holds current six-circle mode.
def g_sect 'G[1]' - Holds current six-circle sector.
def g_frz 'G[2]' - Nonzero when frozen mode is on.
def g_haz 'G[3]' - H of azimuthal reference vector.
def g_kaz 'G[4]' - K of azimuthal reference vector.
def g_laz 'G[5]' - L of azimuthal reference vector.
def g_zh0 'G[6]' - H of first zone-mode vector.
def g_zk0 'G[7]' - K of first zone-mode vector.
def g_zl0 'G[8]' - L of first zone-mode vector.
def g_zh1 'G[9]' - H of second zone-mode vector.
def g_zk1 'G[10]' - K of second zone-mode vector.
def g_zl1 'G[11]' - L of second zone-mode vector.
def g_len 'G[12]' - When using either of the gamma-pseudomotor configura-
tions, the distance from the sample to the gamma stage when gamma = 0.
def g_config 'G[13]' - If zero, the default configuration is selected. Otherwise,
the alternate configuration is used.
def g_track 'G[14]' - If nonzero, the gamT and gamR motors track the pseudomo-
tor gam. Must be set to zero to move gamT and gamR directly.
def g_sigtau 'G[15]' - If nonzero, SIGMA and TAU are considered fundamental
and the azimuthal reference vector is calculated from their values. Otherwise
SIGMA and TAU are calculated from the azimuthal reference vector. Use the
setconfig macro to change the value of g_sigtau.
def g_len2 'G[16]' - When using the screw gamma-pseudomotor configuration,
the distance from the screw pivot point to the gamma stage when gamma = 0.

```

calCHKL – calculates H, K, L, ALPHA, BETA, AZIMUTH, TTH, OMEGA, SIGMA and TAU from the six-circle angles in the motor array A[].

calcA – calculates A[], OMEGA, ALPHA, BETA and AZIMUTH from H, K and L.

calcZ – calculates the *phi* and *chi* to put the two vectors specified by the six elements of the built-in Z[] array in the scattering plane.

cz H0 K0 L0 H1 K1 L1 – “calculate zone”, prints the *chi* and *phi* necessary to put the two vectors specified as arguments in the scattering plane.

mz H0 K0 L0 H1 K1 L1 – “move zone”, moves to the *chi* and *phi* values that put the two vectors specified as arguments into the scattering plane. Also sets zone mode and turns on frozen mode using the calculated *chi* and *phi* values.

sz H0 K0 L0 H1 K1 L1 – “set zone”, calculates the *chi* and *phi* values that put the two vectors specified as arguments into the scattering plane. Also sets zone mode and turns on frozen mode using the calculated *chi* and *phi* values.

setaz [ H K L ] – Sets the azimuthal reference vector.

sigtau [ sigma tau ] – Sets the azimuthal reference vector according to the angles given as arguments. If no arguments are given, the macro will prompt for *sigma* and *tau*,

using the negative of the current values of *chi* and *phi*, respectively, as the defaults.

pa – Displays the current geometry parameters.

startgeo – Prompts for the various geometry parameters.

setlen [ *length* ] – Prompts for the *delta*-arm length needed for the *gamma*-pseudomotor calculations.

setconfig [ *which* ] – Prompts for the configuration. Zero selects the *z*-outward configuration. Nonzero selects the *z*-inward configuration.

settrack [ 1 | 0 ] – Sets the value of *g\_track* to the argument, if any, otherwise prompts for a value.

ontrack – equivalent to settrack 1.

offtrack – equivalent to settrack 0.

aziscan *start finish intervals time* – Does a scan of the azimuthal angle.

### ZEROS AND ROTATION SENSE

For the two supported diffractometer configurations, the *x* axis points upward and the *y* axis points along the direction of the incident beam. The two configurations are mirror images of each other in the *x-y* plane, but both use a right-handed coordinate system. When all circles are at zero, the default configuration (*g\_config* = 0) has the *z* axis pointing outward from a sample mounted on the *phi* circle. In the alternate configuration (*g\_config* != 0), the *z* axis points inward. The choice of configuration is generally dictated by the dimensions and layout of a synchrotron hutch.

In order for the calculations in the six-circle code to work correctly, you must set the rotation sense and zeros of the circles according to the conventions built into the code. The rotation sense is defined with respect to the positive unit vectors of the laboratory coordinate system described above. For a right-handed rotation, if your thumb points along the unit vector that is in line with the rotation axis, your fingers will point in the positive rotation direction.

The rotation sense of an axis can be changed by changing the sign of the *steps-per-degree* parameter or the *sign-of-user-\*-dial* parameter in the *config* file. You should set the sign of the first parameter so that the *spec* dial positions agree with the dial indicators of each circle. Set the sign of the second parameter as necessary to make the *spec* user angles agree with the rotation sense described below.

In the default (*z*-outward) configuration, the *mu* circle is in the *y-z* plane, with the rotation axis along the *x* axis with a right-handed rotation convention. The zero of *mu* is such that the *delta* and *theta* circles are on the negative side of the *x-y* plane with their rotation axes along the *z* axis.

The *delta*, *theta* and *phi* circles all have a left-handed rotation sense. The *delta* circle is at zero when the detector arm is along the negative *y* direction. *Theta* is zero when the *chi* circle is in the *x-z* plane, with the *chi* axis along the *y* axis. The *chi* rotation is right-handed. The zero of *chi* puts the *phi* rotation axis along the *z* direction with the “surface normal” of the *phi* table pointed in the positive *z* direction. The zero of the *phi* circle is arbitrary.

Finally, with *mu* and *delta* at zero, the *gamma* circle is located in the *y-z* plane with the zero of *gamma* chosen so that the detector is pointed in the negative *y* direction towards the X-ray source. The *gamma* rotation is right handed.

In the alternate, mirror image, *z*-inward configuration, all rotations are left handed.

**SEE ALSO**

*geo\_sixc.c*, C source file for the six-circle geometry calculations.

*sixc.src*, macro source file for the six-circle macros.

*hkl.mac*, macro source file for reciprocal space macros.

*ub.mac*, macro source file for orientation matrix macros.

The *Four-Circle Reference* in the spec manual.

**REFERENCES**

Angle calculations and operating modes for a six-circle diffractometer were presented by M. Lohmeier and E. Vlieg in *J. Appl. Cryst.*, **26**, 706 (1993). Extensions, including explicit formulas for the setting of diffractometer motors based on the surface normal direction, were given by D. Abernathy (PhD thesis, MIT 1993), who also assisted in developing the C code used in spec.