CHANDRA
X-ray Center 60 Garden St., Cambridge Massachusetts 02138 USA

MEMORANDUM

Date:	August 23, 2006
From:	Royce Buehler
To:	ACIS ops and ACIS MIT
Subject:	SACGS Implementation Plan: ACIS Gain Change
File:	/data/acis0/royceb/docs/notes/gains/gainChangeImpPlan_1.5.tex

1 Desired Behavior

In (http://asc.harvard.edu/acis/memos/acis_gain/web/compare.html "*Comparison of Paradigms*"), Nancy Wolk showed that very few desired events are lost when S3 gain conversions are applied to frontside chips, so long as the low energy cutoff is below 500 eV; this includes observations for which the low energy cutoff is defaulted. We have also learned that there have been very few low energy cutoffs requested above 500 eV, and that so far all of them have been at 1 keV.

Accordingly, it has been proposed that we adopt the following scheme. Use S3 gain conversions when the aimpoint is on S3, and I3 gain conversions when the aimpoint is on I3, except that I3 conversions will be used on frontside chips, and S3 on backside chips, for each chip which has a chip-wide window. "A chip-wide window" signifies a window covering all 1024 addressible columns and, for a Timed Exposure (TE) run, all 1024 addrssible rows, of the chip. Finally, it will be an error not to call for a chip-wide spatial window on a chip whenever the observer has asked for a lower energy cutoff above 500 keV on any part of that chip. (As a corollary, if the global energy cutoff is above 500 keV, all active chips must have chip-wide spatial windows.) This proposal is spelled out in fuller detail, including definitions for 'I3' and 'S3' gain conversions, in Nancy Wolk's memorandum of February 20, 2006, "Requirements for SACGS changes to the energy to PHA conversions".

Specifically, the following behavior will then be expected from SACGS.

("CCD Type conversion" in what follows refers to applying I3 conversions to an FI chip and S3 conversions to a BI chip. "Aimpoint conversion" refers to applying I3 (resp. S3) conversions depending on whether the aimpoint is at ACIS-I (resp ACIS-S).

Whichever algorithm is used, the last step in a lowerEnergyAmplitude conversion is to take the maximum of the converted value in PHA and 20, and round down the result to the nearest integral PHA.)

1. If the Obscat specifies no value for the global lower energy, treat it as if 0 keV were requested. If it specifies no value for the global energy range, treat it as if 15 keV were requested.

2. If the energy limits in a window are defaulted, treat them as if the same keV were requested as in the global energy limits.

3. When chip-wide spatial windows are requested for all active chips, and OCAT:EventFilter Lower > 0.5 keV, assign 20 and 3750 ADU to the parameter block energy limits.

4. Otherwise, apply aimpoint conversion to derive the parameter block energy limits from the global limits in the Obscat.

5. If the requested global lower energy is > 500 eV, and not all active chips have chip-wide spatial windows, make no assignment for the obsid and issue an error message.

6. When a chip-wide spatial window is requested for a chip, use CCD Type conversion on that chip.

7. If the requested lower energy for a spatial window is > 500 eV, and there is no chip-wide spatial window for that chip, make no assignment for the obsid and issue an error message.

8. If energy limits in a spatial window remain unassigned after attempting to apply each of the above rules, and there has been no error, apply aimpoint conversion on the window.

9. If any window's lowerEnergyAmplitude is less than the parameter block's lowerEnergyAmplitude, or
windetail:lowerEnergyAmplitude + windetail:energyAmplitudeRange >
pb:lowerEnergyAmplitude + pb:energyAmplitudeRange
(where 'pb' stands for 'te' or 'cc', depending on the science mode), make no assignment for the obsid and issue an error message indicating that some energy values in the window will be masked out.

10. Treat changes in PHA energy limits as calibration changes, triggering only version number changes in the pblock id, so long as the limits as requested in keV have not changed.

SACGS IMPLEMENTATION

2 Data Format Changes

The lower and upper limits on the values of these fields are the ones to be specified in the schema files. They are intended purely as a very rough sanity check; the ruleset may from time to time impose much stricter limits.

2.1 Te, Cc, Windetail

Each of the repository files te.dat, cc.dat, and windetail.dat shall be expanded by two new fields, giving the requested lower energy threshold and energy filter range in keV:
`lowerEvtAmplKeV`
`evtAmplRangeKeV`

In the schema files, these fields will be of data type DOUBLE, maximum field width eight characters, with no defaults (since defaults will be assigned, if needed, by the ruleset.) The permissible range for `lowerEvtAmplKeV` will be from 0 to 20.0, and of `evtAmplRangeKeV` from

0.0 to 20.0. The intent in permitting such high upper values is to allow, after conversion, the full range of PHA values which can be produced by the ACIS insrument (4095 ADU.)

These new keV fields will be non-calibration fields; when they change, it will trigger a change in SImode. The existing fields in these three repository files which specify energy filters in ADUs will become calibration fields; when they change, it will only trigger a change in version numbers.

The field `partialMatch` in te.dat and cc.dat is no longer used, and shall be deleted.

2.2 Wincat

Two new fields shall appear in the massaged extract file wincat:

The field `fullyCovered` is one character, value 'Y' or 'N'. The value is to be 'Y' only when some window on the same chip is chip-wide.

The field `liveChip` is one character, value 'Y' or 'N'. The value is to be 'Y' only when the window is on a chip whose video board will be powered on.

2.3 Work

Any global variables to be used by the ruleset must be specified as fields in the work.dat repository file. The naming convention for the new 'work' fields uses I3 and S3 rather than FI and SI, for consistency in case later anomalies force at least one chip-specific conversion. Four linear fits are required to handle the energy threshold conversions: one for low energy values, and one for high values, for frontside chips, and one low and one high fit for backside chips. We will treat the range conversions as additional linear fits, for which the intercept happens to be the origin.

The following 'work' fields shall be added:

Breakpoints Just one for now:
`highestLowEnergy` 1.2 keV with current algorithms.

Gain trigger The keV level above which different conversion factors are expected to be used in different chips:
`windowedGainTrigger`

Lower energy fits Slope and intercept:
```
slopeGainCnvrtLowI3
interceptGainCnvrtLowI3
slopeGainCnvrtLowS3
interceptGainCnvrtLowS3
```

Medium energy fits Slope and intercept:
```
slopeGainCnvrtMedI3
interceptGainCnvrtMedI3
slopeGainCnvrtMedS3
interceptGainCnvrtMedS3
```

Energy range fits Slope and intercept:
```
slopeGainCnvrtHighI3
```

```
interceptGainCnvrtHighI3
slopeGainCnvrtHighS3
interceptGainCnvrtHighS3
```

These fields are all of type DOUBLE. For the slopes, maximum length is 7 characters and limits are 0.0 to 2.0. For the other fields, maximum length is 8 characters. The intercepts may run from -999.0 to 20.0; highestLowEnergy and windowedGainTrigger from 0.0 to 20.0.

A field of type UINT shall track how many chips are covered by full-chip windows: `coveredWindowCount`

3 Ruleset Changes

In setting up window blocks, there has been a simple division of labor between the ruleset and the script `wincat_fmt.pl`. The script decides questions involving which windows exist. Then the ruleset decides how to fill in the parameters for those windows.

Since values of the new wincat field `fullyCovered` depend on which windows exist, deciding on those values will be the responsibility of `wincat_fmt.pl`. The mechanism will be described in section 4 (processing changes) below.

In the descriptions which follow, the order of the rules matters. Each rule which successfully sets the targeted value results in the remaining rules with the same target being skipped. The term 'dummy target' references the field `work::blank`, for which no value is ever set. These rules are therefore guaranteed never to be skipped.

The rules with target `partialMatch` are to be deleted.

3.1 Set current values of the work constants

In the file `napcat0.rul` add
R1. Dummy target.
Set

work::highestLowEnergy	1.2
work::windowedGainTrigger	0.5
work::interceptGainCnvrtLowI3	-22.84
work::slopeGainCnvrtLowI3	120.0
work::interceptGainCnvrtMedI3	-151.81
work::slopeGainCnvrtMedI3	227.0
work::interceptGainCnvrtHighI3	0.0
work::slopeGainCnvrtHighI3	250.0
work::interceptGainCnvrtLowS3	-28.09
work::slopeGainCnvrtLowS3	199.0
work::interceptGainCnvrtMedS3	-42.28
work::slopeGainCnvrtMedS3	211.0
work::interceptGainCnvrtHighS3	0.0

work::slopeGainCnvrtHighS3   250.0

In what follows, the 'work' prefix for these variables may often be left unstated.

3.2 Assign global energy filter values

In the file `napcat.rul` prior to the INCLUDE of `napcat0.rul`, add R2, R3, R6, R7. Just after the rules with target `windowCount`, add R4-R5, R8-R9.

— Fill in the global energy filter if it's defaulted in Obscat

```
R2. Target napcat:eventFilterLower.
IF
    ! KNOWN(napcat::eventFilterLower) OR
    napcat::eventFilterLower is blank
THEN
    set napcat::eventFilterLower to 0.0.

R3. Target napcat:eventFilterRange.
IF
    ! KNOWN(napcat::eventFilterRange) OR
    napcat::eventFilterRange is blank
THEN
    set napcat::eventFilterRange to 15.0.

R4. Target wincat::lower_threshold.
IF
    ! KNOWN(wincat::lower_threshold) OR
    wincat::lower_threshold is blank
THEN
    set wincat::lower_threshold to napcat::eventFilterLower.

R5. Target wincat::pha_range.
IF
    ! KNOWN(wincat::pha_range) OR
    wincat::pha_range is blank.
THEN
    set wincat::pha_range to napcat::eventFilterRange.
```

The next four rules ensure that none of these four energy filter fields in wincat and napcat are requested with too great a precision. If any of them have a precision greater than 2 decimal digits, round to the nearest .01 keV (with values ending in 5 at the third decimal place rounding up) Each of the following rules shall issue the warning message "Excess precision in {field name}; value {old value} rounded to {new value}."

```
R6. Dummy target.
IF floor( 100 * napcat::eventFilterLower) / 100.0
    ≠ napcat::eventFilterLower
THEN
    set napcat::eventFilterLower to
        floor(100.0 * napcat::eventFilterLower + 0.5) / 100.0
```

```
R7. Dummy target.
IF floor( 100 * napcat::eventFilterRange) / 100.0
    ≠ napcat::eventFilterRange
THEN
    set napcat::eventFilterRange to
        floor(100 * napcat::eventFilterRange + 0.5) / 100.
```

```
R8. Dummy target.
IF floor( 100 * wincat::lower_threshold) / 100.0
    ≠ wincat::lower_threshold
THEN
    set wincat::lower_threshold to
        floor(100 * wincat::lower_threshold + 0.5) / 100.0
```

```
R9. Dummy target.
IF floor( 100 * wincat::pha_range) / 100.0
    ≠ wincat::pha_range
THEN
    set wincat::pha_range to
        floor(100 * wincat::pha_range + 0.5) / 100.0
```

— Determine whether each active CCD has a chip-wide spatial window —

At the end of the 'FEP and CD selections' section in `napcat.rul`, add R10 and R11.
For each window, two rules (one for TE, one for CC) will increment `coveredWindowCount` if the window is chip-wide and the chip is live. (To prevent chips from being double-counted, the `wincat_fmt.pl` script will permit only a single chip-wide window on any one chip.)

```
R10. Dummy target.
IF
    napcat::exposureMode is CC AND
    wincat::start_column is 1 AND
    wincat::width is 1024 AND
    wincat::liveChip is 1
THEN
    set work::coveredWindowCount to work::coveredWindowCount + 1
```

R11. Similarly for TE, but also checking `wincat::start_row` and `wincat::height`.

The ruleset already has a 'work' field, `fepCount`, which tracks the number of active chips. Later rules can check whether all active chips have a chip-wide window by checking whether `coveredWindowCount` equals `fepCount`.

— Fill in pblock energy filter —

The rest of the global energy filter rules should appear twice, once in TE and once in CC form. When "eventFilterLower" or "eventFilterRange" appears without a prefix, the te:: or cc:: prefix should be attached accordingly. Rules R12 to R30 shall replace the present rules in `napcat.rul` with targets `lowerEventAmplitude` and `eventAmplitudeRange`, in datasets 'te' and 'cc'.

```
R12; R13. Target te/cc::lowerEventAmplitude.
IF  napcat::spwindow equals Y AND
    work::fepCount equals work::coveredWindowCount
    (that is, every active chip has a chip-wide window) AND
    napcat::eventFilterLower > work::windowedGainTrigger
THEN
    assign 20 and 3750 ADU to eventFilterLower and eventFilterRange in te:: or cc:: record.
```

```
R14; 15. Target te/cc::lowerEventAmplitude.
IF  napcat::eventFilterLower equals 0
THEN
    assign 20 to eventFilterLower.
```

```
R16. Dummy target.
IF  work::fepCount is not equal to work::coveredWindowCount AND
    napcat::eventFilterLower > work::windowedGainTrigger
THEN
    Issue error message
    "OCAT:EventFilter:Lower of {value} requires full-chip windows on each chip."
    and skip to next obsid.
```

```
R17. Dummy target.
IF  napcat::eventFilterLower > 5.0
THEN
    Issue a warning message, "OCAT::EventFilter:Lower of {value} is above 5.0 keV".
```

— pblock energy filter rules for ACIS-S aimpoint —

```
R18; R19. Target lowerEventAmplitude.
IF  napcat::si equals ACIS-S AND
    napcat::eventFilterLower ≤ work::highestLowEnergy
THEN
    set lowerEventAmplitude to floor of
    interceptGainCnvrtLowS3 + napcat::eventFilterLower * slopeGainCnvrtLowS3.
    If negative, set to zero.
```

```
R20; R21. Target lowerEventAmplitude.
IF  napcat::si equals ACIS-S AND
    napcat::eventFilterLower > work::highestLowEnergy
THEN
    set lowerEventAmplitude to floor of
    interceptGainCnvrtMedS3 + napcat::eventFilterLower * slopeGainCnvrtMedS3.
    If negative, set to zero.
```

```
R22; R23. Target eventAmplitudeRange.
IF  napcat::si equals ACIS-S
THEN
    set eventAmplitudeRange to floor of
    interceptGainCnvrtHighS3 + napcat::eventFilterRange ∗ slopeGainCnvrtHighS3.
    If negative, set to zero.
```

— pblock energy filter rules for ACIS-I aimpoint —

R24 to R29.
Repeat the ACIS-S aimpoint rules, exchanging ACIS-I for ACIS-S, and I3 for S3.

(We could make do with fewer rules on the high end, since the intercept is zero and I3 conversion has the same values as the S3 conversion. I believe it's best, though, to set up a system of rules which will make it relatively obvious when gains alter again in the future how the maintainer is to go on from here.)

— Raise ADU to 20 if needed —

`windtl::lowerEventFilter` has been assigned, so we use a dummy target to ensure the next rule gets checked.

```
R30. Dummy target.
IF  lowerEventAmplitude < 20
THEN
    Assign 20 to lowerEventFilter
    Issue warning, "Requested lower energy could not be met.
```

pblock PHA value raised to 20 ADU."

3.3 Assign window energy filter values

Rules filling in parameters within windows go into the file `make-rule2.pl`. Each of these rules is applied to each window. These rules shall replace the present ones for targets `windtl::lowerEventAmplitude` and `windtl::eventAmplitudeRange`.

— Require chip-wide window where absent

Since `wincat::fullyCovered` has been filled in by `wincat_fmt.pl`, it isn't necessary to examine the `startRow`, `startColumn`, `width` or `height` fields for the windows. Consequently, we will not need separate rules for 1d and 2d windows.

R31. Dummy target.
IF `wincat::lower_threshold` > `work::windowedGainTrigger` AND
    `wincat::fullyCovered` is 'N' AND
    `wincat::liveChip` is 'Y'
THEN
    Issue the error message,
    "Lower energy threshold of {value} on chip{wincat::chip} requires a full-chip window."
    and skip to the next obsid.

— Use CCD type conversion to fill in window energy filters, wherever chip-wide windows do exist on the same chip

R32. `Target windtl::eventFilterLower`.
IF `wincat::lower_threshold` equals 0
THEN
    assign 20 to `windtl::eventFilterLower`.

R33. Dummy target.
IF `wincat::lower_threshold` > 5.0 keV
THEN
    Issue a warning message,
    "Lower threshold of {value} in window on chip {wincat::chip} is above 5.0 keV."

— FI type window conversions —

R34. Target `windtl::lowerEventAmplitude`.
IF `wincat::fullyCovered` is 'Y' AND

```
    wincat::chip is neither 5 nor 7 (i.e., chip is FI) AND
    wincat::lower_threshold ≤ work::highestLowEnergy
THEN
    set windtl::lowerEventAmplitude to floor of
    interceptGainCnvrtLowI3 + wincat::lower_threshold * slopeGainCnvrtLowI3.
    If negative, set to zero.
```

```
R35. Target windtl::lowerEventAmplitude.
IF  wincat::fullyCovered is 'Y' AND
    wincat::chip is neither 5 nor 7 (i.e., chip is FI) AND
    wincat::lower_threshold > work::highestLowEnergy
THEN
    set windtl::lowerEventAmplitude to floor of
    interceptGainCnvrtMedI3 + wincat::lower_threshold * slopeGainCnvrtMedI3.
    If negative, set to zero.
```

```
R36. Target windtl::eventAmplitudeRange
IF  wincat::fullyCovered is 'Y' AND
    wincat::chip is neither 5 nor 7 (i.e., chip is FI)
THEN
    set windtl::eventAmplitudeRange to floor of
    interceptGainCnvrtHighI3 + wincat::pha_range * slopeGainCnvrtHighI3.
    If negative, set to zero.
```

— BI type window conversions —

R37 to R39.
These three rules are like those above, with "either 5 or 7" replacing "neither 5 nor 7", and S3 replacing I3.

— Use aimpoint conversion to fill in remaining window energy filters —

If we've reached this point, and the window energy filter is not filled in, we may be sure that all windows on this chip are partial, and this window does not have `wincat::lower_threshold` over 0.5 keV.

```
R40. Target windtl::lowerEventAmplitude.
IF  napcat::si equals ACIS-I AND
    wincat::lower_threshold ≤ work::highestLowEnergy.
THEN
    set windtl::lowerEventAmplitude to floor of
    interceptGainCnvrtLowI3 + wincat::lower_threshold ∗ slopeGainCnvrtLowI3.
    If negative. set to zero.
```

R41. Target `windtl::lowerEventAmplitude`.
IF `napcat::si` equals ACIS-I AND
    `wincat::lower_threshold` > `work::highestLowEnergy`.
THEN
    set `windtl::lowerEventAmplitude` to floor of
    `interceptGainCnvrtMedI3` + `wincat::lower_threshold` ∗ `slopeGainCnvrtMedI3`.
    If negative. set to zero.

R42. Target `windtl::eventAmplitudeRange`
IF `napcat::si` equals ACIS-I
THEN
    set `windtl::eventAmplitudeRange` to floor of
    `interceptGainCnvrtHighI3` + `wincat::pha_range` * `slopeGainCnvrtHighI3`.
    If negative. set to zero.

R43 to R45.
Three rules parallel to R40 through R42 when `napcat::si` equals ACIS-S lead to the same consequences, with S3 substituting for I3.

— Raise ADU to 20 if needed —

`windtl::lowerEventAmplitude` has been assigned, so we use a dummy target to ensure the next rule gets checked.

R46. Dummy target.
IF `windtl::lowerEventAmplitude` < 20
THEN
    assign 20 to `windtl::lowerEventAmplitude`
    issue warning, "Requested lower energy for window on chip `windtl::chip` could not be met. PHA value raised to 20 ADU."

— Check that pblock is not more constraining than any wblock —

R47 and R48. Dummy targets.
IF `windtl::lowerEventAmplitude` < [te or cc]::`lowerEventAmplitude`.
THEN
    Issue error message
    "OCAT::EventFilter:Lower {value} masks OCAT:Window:LowerEnergy
        {windtl::lowerEventAmplitude} on ccd {windtl::ccdId}."
    and skip to next obsid.

R49 and R50.
IF `windtl::lowerEventAmplitude` + `windtl::eventAmplitudeRange` >
        [te or cc]::`lowerEventAmplitude` + [te or cc]::`eventAmplitudeRange`

THEN

Issue error message

"Upper end of OCAT:EventFilter range masks high energy values for ccd {windtl::ccdId}"

and skip to next obsid.

4 Procedural changes

4.1 `wincat_fmt.pl`

This script massages the window parameter requests extracted from the Obscat, putting it into `wincat.sch` format.

A new routine shall be added which reads the `napcat`, and builds an internal live chip table, with an entry for each observation with spatial windows, indicating which video boards are powered on for that observation.

The script already sorts the extracted records by rank within chip within obsid. It shall be modified to store window information until a change of chip, setting a flag F when any window is chip-wide. If it encounters a second chip-wide window in the same chip, it shall issue a warning message, without storing the information for that window. Upon a change of chip, it shall write out the `wincat.dat` records for the old obsid and chip, with `fullyCovered` taking the value 'Y' in all records if the flag F is set, and the value 'N' otherwise.

It shall be further modified to assign values to the `liveChip` field, based on an obsid lookup in the live chip table.

The script shall fail whenever it encounters two or more full-chip windows on the same chip. This includes the case of an implicit second full-chip window, in which both a full-chip window and a window with an 'Include' flag are requested for the same chip.

4.2 `apcat2apr`

In the source files `apcat2apr, TeRec.C, CcRec.C`, and `WinDtlRec.C`, the te and cc fields `lowerEventAmplitude` and `eventAmplitude Range` shall be added to the list of calibration fields. In the source file `wincat2apr.C`, the distinction between calibration and noncalibration fields shall be introduced, and the lowerEvent Amplitude and EventAmplitude Range fields shall comprise the list of calibration fields.

In `apcat2apr.C`, simplify the logic in the `update()` routine by eliminating references to the obsolete "partialMatch" field, and the conditional brances depedent on them. (Since launch, "partialMatch" has always had the value 'Y'.)

Add to the rule language:

1. A function KNOWN, taking a rule language variable (such as `wincat::liveChip`) as an argument, returning the boolean TRUE when the inference engine has assigned a value to the variable, FALSE when no value is yet assigned.

2. Two new infix operators, SUP and INF, expecting numeric arguments. "exp1 SUP exp2" returns the maximum of the two expressions, "exp1 INF exp2" returns the minimum.

Finally, in `apcat2apr`, the tt and cc field `windowBlockId` will now get special treatment. If the `windowBlockId` changes, but only in the eighth and ninth hex digits (that is, the window block has undergone only a calibration change), it will be treated as a calibration field. Any change to the other hex digits will force a change in SImode.

4.3 auxiliary scripts

`modelist_update.pl`, which produces the file used by the SImode finder web page, should be modified to track energy filters accurately by requested keV. It will presumably have to reference `te.dat` and `cc.dat`, of which it is presently ignorant, in order to do this.

5 Implementation sequence

The conversion of SACGS will involve the following steps:

A) Prepare new .sch file formats and default values

B) Prepare new ruleset from `make-rule*.pl` and `napcat*.rul` primitives. The new script `build-rules.pl` can carry out the preparation.

C) Make coding changes to `wincat2apr.C`

D) Make coding changes to `apcat2apr.C`

E) Make coding changes to `wincat_fmt.pl`

F) Convert `te.dat`, `cc.dat`, `windetail.dat` to new format.
– It should be possible to do this with simple nawk scripts.

G) Prepare a test `napcat.dat` and `wincat.dat` (Section 6.1 spells out the required cases.)

H) Test the new system on these cases. (See Section 6.1, below.)

I) Prepare modified version of the rules, identical to the new one except that all intercepts are zero and all slopes are 250.

J) Run modified new version and old version on AO07 obsids, and confirm that all differences in resulting `napc2par.log`, `obsreqs.cmd`, session logs, and repository files are understood.

– *Once test results have been reviewed and accepted by the ACIS team:*

K) `te.dat`, `cc.dat` and `windetail.dat` may have changed since the conversion in step (F). If so, redo the conversion to the new format.

L) Do three batch runs of the new SACGS, one on each of the last three AO periods. These will update almost all SImodes which need new versions. Any stragglers can have their new versions acquired and tested as they come up in future operations.

M) Place the new sch files, repository files, ruleset, ACIS Tables, and executables into production.

6 Testing

6.1 Test Cases for Ruleset Coverage

This subsection describes the test cases to be created and used in implementation steps G and H. The cases listed here suffice to ensure that all new rules will be triggered. In addition, they sample a fairly wide range of combinations of the relevant parameters.

For these test cases, we will prepare handmade versions of the extract files: `gainCases` (the napcat file) and `gainCases_winparams` (the precursor to the `wincat` file). All the non-specified fields for `gainCases` can be based on an extract of obsid 7092, which was assigned TE_002A2. In those cases which explicitly call for fewer than six chips, simply change the corresponding chip selections to 'N', and the standard flag to 'N'.

Assign dummy obsids in a 62000 series. These dummy obsids will never appear among existing `acispar.dat` records, since they are assigned by the FOT outside of SACGS.

The numbering will, however, make it easy to locate the test case obsids within the output `acispar.dat`.

A new script shall be written (`sg_insert_trace.pl`), which will take a ruleset as input, and insert display lines in rules, using the ruleset language's "DEBUG" action, indicating when a rule has been triggered. Depending on the preceding comment, the script will insert a display for each rule with a comment beginning "# Rn", where n is one or more decimal digits; or for each rule with a comment beginning "# TRn" where n is one or more decimal digits.

The "# Rn" comments are inserted automatically for each rule by the `build-rules.pl` script, enabling a complete trace. A less cluttered trace of selected rules can be invoked using "# TRn" lines which have been inserted by hand.

The script `obsparams` shall be modified to accept a -T switch, which will trigger display of "DEBUG" lines by `apcat2apr`.

Our test cases will refer to the following conditions.

Global lower energy limit `napcat::eventFilterLower`:
GL1- defaulted
GL2- below trigger (0.4 keV)
GL3- at trigger boundary (0.5 keV)
GL4- first linear fit (1.0 keV)
GL5- above linear fit boundary (2.0 keV)
GL6- above legal lower boundary (6.0 keV)
GL7- below threshold (0.1 keV)

Global range `napcat::eventFilterRange`:
GR1- defaulted
GR2- specified at a nondefault value (We will use 11.0 keV)

For windows:
the same values for `wincat::lower_threshold` and `wincat::pha_range` will be designated as WLn and WRn.

Unless otherwise specified, all windows will have sample value 1. Partial windows will start at column 400 (row 256 for TE), with width 100 (height 150 for TE). Except where noted, each case discussed below actually represents two cases, one in TE mode and one in CC mode.

In the cases described in tables, we use these abbreviations:

pw:	partial window, column 400 (row 256 for TE), with width 100 (height 150 for TE)
fw:	full window
L:	eventFilterLower
R:	eventFilterRange
D:	value was defaulted in OCAT
-:	window not present.

Many of the new rules will be triggered by multiple cases. For more sparsely covered rules, the case descriptions note that the rule is exercised.

Case 1 **Defaults all around**. (TE only) GL1, GR1, Full windows on S2 and S3 with WL1, WR1. Should set all energy filters to 20, 3750 ADU. (Triggers R2, R3, R4, R5, R14, R32.)

Case 2 **Windows but not global values defaulted**. (TE only) GL2, GR2, Full window on S3 with WL1, WR1. Should set all energy filters to 0.4 keV, 11.0 keV. (R4, R5, R32)

Case 3 **Below threshold**. S2 and S3, no windows, GL7 and GR1. Should issue a warning that lower energy is being set up to threshold.

Case 4 **Window below threshold**. (TE only) GL1 on S2 and S3, partial window on S2 asks for WL7; instrument ACIS-S. Should issue warning that lower energy is being set up to threshold. (R32)

Case 5 **Pblock masks low window energy**. ACIS-S, I3. GL2, GR1, full window on I3 with WL1, WR1. (R30) Should issue error message from Rule 47/48.

Case 6 **Pblock masks upper energy limit**. ACIS-S, chip S3 on. The `napcat` shall have `eventFilterLower` 1.4, `eventFilterRange` 13.0, while the partial window on S3 shall have `lower_threshold` 4.0, `pha_range` 13.0. The full window on S3 defaults its energy filter values. This should produce an error message from rules 49 and 50.

Case 7 **Windowless I3** (Six subcases). ACIS-I, no windows, GL2 to GL7. Should set pblock energy filter using FI formulas.

Case 8 **Windowless S3** (Six subcases). ACIS-S, no windows, GL2 to GL7. Should set pblock energy filter using BI formulas.

Case 9 **ACIS-S, overprecise window lower energy cutoff**. (TE only)

	I2	I3	S1	S2	S3	S4
pwL	-	-	0.401	-	-	0.4
pwR	-	-	12.6	-	-	12.6
fwL	-	-	-	-	-	2.0
fwR	-	-	-	-	-	13.0

Should issue error message. (R8)

Case 10 **ACIS-S, overprecise window range**. (TE only)
Same as preceding case, but set pwL on S1 to 0.4, pwR to 14.995. Should issue error message. (R9)

Case 11 **Overprecise global energy cutoff**. (TE only) ACIS-S, no windows, `napcat::eventFilterLower` set to 0.401. Should issue error message. (R6)

Case 12 **Overprecise global energy range**. (TE only) ACIS-S, no windows, `napcat::eventFilterRange` set to 14.995. Should issue error message. (R7)

Case 13 **ACIS-S, 0.3 keV in `napcat::eventFilterLower`**.

Among other things, this case will exercise rules 12 and 13.

	I2	I3	S1	S2	S3	S4
pwL	0.5	1.0	D	D	0.4	0.45
pwR	14.5	13.0	D	D	13.0	D
fwL	-	0.45	0.4	-	-	2.0
fwR	-	D	13.0	-	-	13.0

Case 14 **ACIS-S, 0.3 keV in `napcat::eventFilterLower`**.

	I2	I3	S1	S2	S3	S4
pwL	1.0	0.4	0.5	0.45	2.0	1.0
pwR	13.0	14.5	13.0	D	13.0	13.0
fwL	0.45	-	-	0.45	2.0	0.45
fwR	13.0	-	-	D	13.0	D

Case 15 **ACIS-I, 0.3 keV in `napcat::eventFilterLower`**

	I2	I3	S1	S2	S3	S4
pwL	0.5	1.0	D	D	0.4	0.4
pwR	14.5	13.0	14.0	D	13.0	14.0
fwL	-	-	0.4	-	D	2.0
fwR	-	-	13.0	-	D	13.0

Should issue error message that full window is needed on I3.

Case 16 **ACIS-I, 0.3 keV in `napcat::eventFilterLower`**.

	I2	I3	S1	S2	S3	S4
pwL	1.0	0.4	0.5	D	2.0	1.0
pwR	13.0	14.5	13.0	D	13.0	13.0
fwL	D	-	-	0.4	2.0	0.4
fwR	13.0	-	-	14.5	13.0	14.6

Case 17 **ACIS-S, 1.0 keV in `napcat::eventFilterLower`**

	I3	S1	S2	S3	S4
pwL	1.5	D	1.5	1.0	1.5
pwR	13.5	D	13.5	13.0	13.5
fwL	1.5	1.0	1.5	D	2.0
fwR	13.5	13.0	13.5	D	13.0

Case 18 **ACIS-I, 1.0 keV in `napcat::eventFilterLower`** .

	I2	I3	S1	S2	S3	S4
pwL	1.0	1.0	D	D	0.7	1.0
pwR	D	13.0	13.0	D	13.0	D
fwL	D	D	0.7	D	D	2.0
fwR	D	D	13.0	D	14.0	13.0

Case 19 **Double dip full chip**. Repeat case 17, but drop the full window on I3, and add a second full window on S1. The second window will have `sample` = 0, and `lowerEventAmplitude` = 1.0. Should result in an error message that full windows are required on all chips. (R16)

Case 20 **One full chip window needed**. Repeat Case 14, but drop the full window on I2. Should result in an error message that a full chip window is required on chip 2. (R31)

Case 21 **Full window on inactive chip**. Repeat case 19, but move the second full window on S1 to the inactive chip I0. Should result in an error message that full chip windows are required on all chips. (R31)

Case 22 **ACIS-I, 0.3 keV in `napcat::eventFilterLower`**. As in case 15, but with 0.4 pwL for I3, and (to avoid masking by the pblock) 13.0 fwR for S3..

	I2	I3	S1	S2	S3	S4
pwL	0.5	0.4	D	D	0.4	0.4
pwR	14.5	13.0	14.0	D	13.0	14.0
fwL	-	-	0.4	-	D	2.0
fwR	-	-	13.0	-	13.0	13.0

6.2 Comparison test on AO07 data

This is the test indicated by implementation steps (I) and (J). The previous test was aimed at showing that the effect of the new rules and processing on energy filter PHA levels is as expected. This test is aimed at showing that none of these modifications introduce changes in other parameters.

To this end, an “old gains” copy will be made of the new ruleset, and in this copy “R1” will be altered to set all intercepts to 0 and all slopes to 250. Also, ’windowedGainTrigger’ will be set to 5.0, so that there will be no forced creation of full chip windows.

AO07 will be re-extracted from the ODB, using the root file name `gainsAO07`. Since obsparams skips over observations unless they are in a reasonable processing state, edit the extract file to change all instances of ’scheduled’ and ’observed’ to ’unobserved’. To simulate batch operations at the beginning of AO07, prepare a set of repository files in which te.dat is terminated before 0x0071c014, cc.dat before 0x000a8014, acispar.dat before obsid 6299 and win.dat, windetail.dat before 0x00092014.

Set up soft pointers from $SGDAT/napcat.dat and $SGDAT/wincat.dat to the `gainsAO07` extract files. Execute ‘make clean’ and ‘make start’. Then copy the “old gains” ruleset to $SGDAT/sch/ap2apr.rul, and the prepared repository files to $SGDAT. Execute `obsparams`. Save the resulting repository files, `napc2par.log`, and `obsreqs.cmd` in another directory, where ‘make clean’ can’t wipe them out.

Repeat the process in the previous paragraph, but now using the old (pre-gain change) ruleset. The resulting repository files, `napc2par.log`, and `obsreqs.cmd` should diff cleanly against the ones saved in the previous run, except perhaps for differing garbage characters at the ends of the `napc2par.log` lines.

7 Impacts on other teams

7.1 Impacts on proposers and MP

Proposers may wish to maintain the same actual PHA values for a series of observations, before and after the changeover. This can be done by applying the reversion formulas embodied in the script `convert_energy.pl` User interface contacts will want to check with those proposers who have continuing series, for which specific energy limits had been requested, to learn whether they want to make such an adjustment.

No other impacts on proposers, MP, or USINT are anticipated.

7.2 Impacts on FOT

There will be no format changes to the ACIS Tables. This conversion should be transparent for the FOT.