MIT CSR ACIS	**ENGINEERING CHANGE ORDER**	**ECO No. 36–1022**

CENTER FOR SPACE RESEARCH
MASSACHUSETTS INSTITUTE OF TECHNOLOGY

DWG. NO.	NEW REV.	DRAWING TITLE
36-58020	C	Flight Software Optional Patch Release C

REASON FOR CHANGE:

Addition of patches *smtimedlookup*, *ctireport1*, *ctireport2*, *compressall*, and *untricklebias*; updated *eventhist* patch and IP&CL; these patches address software problem reports 131-134 and implement the new *cti1* and *cti2* timed-exposure modes.

DESCRIPTION OF CHANGE:

The standard patches—release B—are unchanged. The new set of optional—release C—patches is compiled and loaded into a common address space so that each patch can be loaded independently of the others, provided the load order defined in *PatchRelease.spec* is maintained.

Patches *eventhist*, *ctireport1*, and *ctireport2* require that *smtimedlookup* is also loaded; similarly, the engineering patches *deaeng*, *dearepl*, and *printswhouse* require the *tlmio* patch. *deaeng* and *dearepl* must not be loaded at the same time.

	SIGNATURE	DATE	REMARKS:
ORIGINATOR	Peter Ford	03/21/03	Final sign-off version
MECHANICAL			
ELECTRICAL			
SOFTWARE			See attached report by Jim Francis.
STRUCTURE			
FABRICATION			
SCIENCE			
SYSTEMS ENG.			
QUALITY			
PROJ. ENGINEER			
DEPUTY PM			
PROJ. MANAGER			

Contents

ACIS SOFTWARE PROBLEM REPORT

CENTER FOR SPACE RESEARCH
MASSACHUSETTS INSTITUTE OF TECHNOLOGY

FOR: Part Number	Rev:	Sub-Section Name:	Used on hardware: DEA Rev:	Human Interface:
36-54002.08	1.5	SW ACIS FLT 1.5	Flight	

Originator:	Phone:	Date:	RCTU Rev:	Front End HW:
P. Ford	x3–7277	10/12/99	Flight	Flight

Description of Problem: (should be sufficiently complete to be duplicated by engineering):

SPR-131: Bogus Parity Errors Reported in Te5x5 Mode

When a 3x3 event candidate is centered at (Row 2, Column 1), one of the pixels reported in 5x5 mode will be at the unphysical location (Row 0, Column -1). The corresponding bias value, also unphysical, appears to the FEP flight software as an entry with parity flags `0xf`. The column number is ANDed with `0xffe` before downlinking, so the `patchDataBiasError` packet will contain the following element:

```
biasErrors[0] = {
  row              = 0
  column           = 1022
  evenPixelFlags   = 15
  oddPixelFlags    = 0
}
```

Corrective Action:

This feature has been documented in the ACIS IP&CL (36-53206.0204), Revision N, and described in the documentation accompanying the `reportgrade1` patch.

Problem closed on: Date:	Refer to ECO #:	Refer to Patch ID:
	36-1021	`reportgrade1`

Problem ID: **M99101202**	Status: Open	Sheet: 1 of 4

MIT CSR ACIS

ACIS SOFTWARE PROBLEM REPORT

CENTER FOR SPACE RESEARCH
MASSACHUSETTS INSTITUTE OF TECHNOLOGY

FOR: Part Number	Rev:	Sub-Section Name:	Used on hardware: DEA Rev:	Human Interface:
36-54002.08	1.5	SW ACIS FLT 1.5	Flight	

Originator:	Phone:	Date:	RCTU Rev:	Front End HW:
P. Ford	x3–7277	03/09/00	Flight	Flight

Description of Problem: (should be sufficiently complete to be duplicated by engineering):

SPR-132: The BEP doesn't report the grade codes of rejected events

For purposes of CTI modelling, &c., several people have asked that ACIS be made to report a breakdown of the events that it filters. It would be best if these statistics were reported for each CCD separately and at "frequent intervals" during each science run. The chosen solution should best have minimal impact on telemetry throughput and on existing ground software.

Corrective Action:

Develop a patch to report the following entities for each FEP within software housekeeping, *i.e.*, over approx. 64 second intervals:

- events removed by pulse height
- events removed by each of 5 pre-determined grade codes
- events removed by any other grade code
- events removed by any window filter
- events telemetered

This necessitates a small in-line patch to the `SwHousekeeper` constructor in "*filesswhouse/swhousekeeper.C*" to increase the number of allowed housekeeping codes and a replacement of `PmEvent::filterEvent()` in "*filesscience/pmevent.C*" to add the housekeeping calls. Finally, "*acis_h/interface.h*" and the IP&CL release notes, "*ipcl/ipcl_notes.doc*", must be updated to include the new housekeeping codes.

Problem closed on: Date:	Refer to ECO #:	Refer to Patch ID:
	36-1021	`reportgrade1`

Problem ID: **M00030901**	Status: Open	Sheet: 2 of 4

MIT CSR ACIS

ACIS SOFTWARE PROBLEM REPORT

CENTER FOR SPACE RESEARCH
MASSACHUSETTS INSTITUTE OF TECHNOLOGY

FOR: Part Number	Rev:	Sub-Section Name:	Used on hardware: DEA Rev:	Human Interface:
36-54002.08	1.5	SW ACIS FLT 1.5		

Originator:	Phone:	Date:	RCTU Rev:	Front End HW:
P. Ford	x3-7277	06/28/00		

Description of Problem: (should be sufficiently complete to be duplicated by engineering):

SPR-133: Event Packets interleaved with Bias Packets, causing FEP T-Plane Lock-Up

Hardware thresholding ceased abruptly in two FEPs (`FEP_0` and `FEP_2`) during OBSID 371, a timed exposure run. An examination of the telemetry stream showed that event and bias packets were being interleaved although the patch to prevent this (see `biastiming` and Problem Report 117) was active at the time. The telemetry can be found in "*/nfs/min/d7/acis4a/psci/acis4.35.**".

This problem may be related to the run that preceded OBSID 371. This was Squeegee Test #3, OBSID 62015, in which a single CCD, I0, was run in raw-mode with a bias-only command.

Note added 11/17/01: the anomaly recurred twice in the space of a (high background radiation) week, during OBSID 3403 (10/29/01) and OBSID 2010 (11/04/01). The symptoms were identical to OBSID 371. The preceding runs were, respectively, a Cc3x3 faint mode (without recomputing bias) and a Te3x3 graded mode (with bias).

Corrective Action:

As a first measure, the `forcebiastrickle` procedure has been developed to be used in conjunction with the `RADMON ENABLE/DISABLE` commands and FEP power-cycling, to enable a "latched" science run to be quickly restarted during a real-time contact.

A more complete solution has been developed in the form of the `untricklebias` patch, which alters code in the `biasThief` and `scienceMode` methods so that all biasThief functions are executed within the science task, not the bias task.

Problem closed on: Date:	Refer to ECO #:	Refer to Patch ID:
	36-1024, 36-1028	`untricklebias`

Problem ID: **M00062901**	Status: Open	Sheet: 3 of 4

ACIS SOFTWARE PROBLEM REPORT

CENTER FOR SPACE RESEARCH
MASSACHUSETTS INSTITUTE OF TECHNOLOGY

FOR: Part Number	Rev:	Sub-Section Name:	Used on hardware: DEA Rev:	Human Interface:
36-54002.08	1.5	SW ACIS FLT 1.5		

Originator:	Phone:	Date:	RCTU Rev:	Front End HW:
P.Ford	x3-7277	04/09/02		

Description of Problem: (should be sufficiently complete to be duplicated by engineering):

SPR-134: Raw mode compression fails if the compressed packet cannot fit within buffer

During ground processing of telemetry from `OBSID 61267` (`CAP 785`), a calibration test with special PRAM to test charge injection, it was noticed that the CCD rows that should have contained the accumulated charge were reported as being of zero length.

The anomaly was traced to code in `PmTeRaw::digestRawRecord` that, when it came to compress a row of pixel values with high variance and found that the length of the compressed row exceeded the maximum buffer length, gave up and output a zero length record.

The same behavior is anticipated from the continuous-clocking equivalent: `PmCcRaw::digestRawRecord`, and from the equivalent bias map compression routines, although the latter are less likely to be incompressible.

Corrective Action:

The `compressall` patch has been developed to force "incompressible" rows to be formatted without compression. Their presence is indicated by setting `compressionTableSlotIndex = 255` and `compressionTableIdentifier = 0` in the header of the *dataCcRaw* or *dataTeRaw* packet. The ground software (*e.g.*, *psci*) must be upgraded to anticipate these values and act accordingly.

Problem closed on:	Date:	Refer to ECO #:	Refer to Patch ID:
		36-1027	`compressall`

Problem ID: **M02040901**	Status: Open	Sheet: 4 of 4

Existing ACIS Flight Software Patches

	Name	Rev	Size	Part	ECO	SPR
Standard Release B						
1	corruptblock	A	16	36–58030.01	994	113
2	digestbiaserror	A	64	36–58030.02	995	116
3	histogramvar	A	16	36–58030.03	999	115
4	biastiming	A	1612	36–58030.04	993	117
5	rquad	A	16	36–58030.14	1000	121
6	histogrammean	A	156	36–58030.15	996	123
7	zap1expo	A	64	36–58030.16	997	122
8	condoclk	A	472	36–58030.17	1012	127
9	fepbiasparity2	A	636	36–58030.19	1015	130
10	cornermean	A	32	36–58030.21	1017	128
Optional Release C						
1	eventhist	B	5908	36–58030.05	1025	N/A
2	cc3x3	B	4636	36–58030.06	1018	120,124,126
3	teignore	A	36	36–58030.09	1003	N/A
4	ccignore	A	36	36–58030.10	1004	N/A
5	smtimedlookup	A	3712	36–58030.24	1025	N/A
6	ctireport1	A	4548	36–58030.25	1026	N/A
7	ctireport2	A	2768	36–58030.26	1026	N/A
8	compressall	A	2368	36–58030.27	1027	134
9	untricklebias	A	1612	36–58030.28	1028	133
10	reportgrade1	A	816	36–58030.22	1021	131, 132
Under Development						
1	hybrid	03	6104	36–58030.13	1010	N/A
2	fepbiasparity1	02		36–58030.18	1014	N/A
3	squeegy	06	4412	36-58030.23	1023	N/A
4	forcebiastrickle	01	N/A	36-58030.29	1024	133
Engineering Unit Utility Patches						
1	tlmio	02	10312	36–58030.07	1010	N/A
2	printswhouse	01	7224	36–58030.08	986	N/A
3	deaeng	02	2604	36–58030.11	1010	N/A
4	dearepl	02	556	36–58030.12	1010	N/A

Status of Optional S/W Patches, revision C

Name	Part Number	Description	Typos*	RIDs†	Status
IP&CL	36–53204.0204	Software Structure Definitions	2	0	Some minor changes were made to the documentation.
reportgrade1	36–58030.22	Report filtered events in software housekeeping packets	12	1	A minor coding change was been made, and the results tested.
smtimedlookup	36–58030.24	Make timed-exposure mode selection by table lookup	0	0	Some minor changes were made to the documentation.
eventhist	36-58030.05	Implement time-exposure event histogram mode	0	1	Existing patch was updated for compatibility with *smtimedlookup.* An additional test was requested—this has been implemented and run.
ctireport1	36–58030.25	Implement precursor charge reporting, using the 16 outlier pixels in 5x5–mode event packets	4	1	The reviewers requested that substantial changes be made to the code and the documentation. This has been implemented and tested.
ctireport2	36–58030.26	Implement precursor charge reporting, using three low-order bits in 3x3–mode event packets	1	0	Some minor changes were made to the documentation.
compressall	36–58030.27	Correct a bug in raw-mode data compression	5	1	Additional tests were requested. They have been coded and run.
untricklebias	36–58030.28	Run the bias thief methods from the science task	2	1	Additional tests were requested. They have been coded and run.
S/W Review	36–58020 C	Documentation accompanying the individual patch ECOs	1	0	Some minor changes were made to the documentation.
Certification	36–58021.02	Documentation describing the multi-patch certification tests	0	1	The reviewers requested that untricklebias be added to the list of certified patches. The certification code has been updated and the tests have been run.

* typographical errors in the documentation
† review item discrepancies—requiring changes to the patch code and/or test procedures

Review Report

Prepared by: James Francis
Review Date: August 6, 2002
Report Date: August 12, 2002

Summary

This review report covers the ACIS software patches from ECO 36-58020 and ECO 36-58021.02. Based on my review of the material, and of the discussions that came up during the review meeting, it's been determined that the proposed patches meet the following objectives:

- They pose no risk to the health and safety of the instrument
- Based on the existing tests and based on code inspection, they pose little to no risk to the mission science objectives
- With the exception of “ctireport1”, they effectively implement the desired changes

Recommendations

In addition to the recommendations made by the other review team members, I recommend the following actions prior to use of the proposed patches. Except for “ctireport1”, none of these recommendations are critical to the success of the patch set:

- Concur with the decision to revise “ctireport1” to handle precursor masking of charge, and re-review it once the changes are complete
- Recommend that the tests for “smtimedlookup” be enhanced to verify the “start” and “terminate” behaviors of all of the science modes provided at launch.
- Recommend that the “DebugProbe” declarations be omitted from the patch code. Originally, they were a debugging hook during the flight software development. They're unused and unnecessary at this point.
- Consider setting software housekeeping limit check to array size N+64 instead of active limit (N+54). Will save having to worry about it in later housekeeping additions.

Details

For this review, I read the material presented with ECO 36-58020 and ECO 36-58021.02, and studied the code in detail for “smtimedlookup”, “compressall” and “untricklebias”. In general, everything looks well thought out and tested.

In “smtimedlookup”, the lookup table strategy looks sound and well implemented. Until the review, I hadn't realized the importance of needing 8-byte entries, but Peter correctly and effectively explained the need. My only real concern came up during the review, and only in testing. If the patch improperly re-orders the dispatch of pre-existing modes, there is no compile-time or automatic check to verify the right function pointer is in the correct array slot. To verify the proper dispatching of these pre-existing modes, the tests (either unit or certification) should re-test the starting and termination of the modes provided at launch. Test-by-inspection is ok, but by incorporating pre-existing mode tests would be more reliable, and easier to use during regression testing for future patches.

“compressall” also looks well thought out and implemented. I verified the overall flow and it looks sound. My only concern here is the time it takes to re-initialize the huffman tables during telemetry process (we usually setup the tables once at the start of the run). Peter correctly indicated that this is only used in raw mode, and that, in this mode, the system is always telemetry-bound, and the table unpacking won't impact the overall system throughput. In the case where the raw data is heavily reduced, such as with narrow window filters, the compressed data is very unlikely expand to overflow a telemetry buffer, causing the huffman table expansion.

“untricklebias” is an interesting patch for an interesting problem. The main trick with this patch is to ensure that all needed member functions of BiasThief can be safely called from the science task. The main issue with this patch is how to handle the BiasThief::checkMonitor function. All the other key functions (with the exception of the main task loop, which is patched to only respond to taskMonitor queries), appear to be task independent (i.e. they key off the running task, rather than the “this” task instance variable). In the case of checkMonitor(), it uses the Task:requestEvent function, which is specifically coupled to the task instance, instead of the currently running task. Peter successfully addressed this by patching the checkMonitor function to use Task::queryCurrentTask to get the currently running instance, rather than use the “this” instance variable. He also properly modified the function to handle the science task ABORT requests, and developed quite an extensive set of tests to verify the proper termination and abort scenarios needed by the science task.

The most serious issue with the patches came during the review meeting. Peter noticed that the “ctireport1” patch didn't behave as he described its operation to the group (none of the review members, myself included, had noticed the problem). Peter will fix the problem and provide updated material at a later point in time.

In conclusion, I believe that the proposed patches, with the exception of “ctireport1”, provide the desired functionality. All of the proposed patches and are of virtually no risk to the instrument, and of little risk to temporary loss of instrument science.

MIT CSR ACIS	**ENGINEERING CHANGE ORDER**	**ECO No. 36–1021**

CENTER FOR SPACE RESEARCH
MASSACHUSETTS INSTITUTE OF TECHNOLOGY

DWG. NO.	NEW REV.	DRAWING TITLE
36-53204.0204	N	IP&CL Software Structure Definitions
36-58030.22	A	*reportgrade1*: Patch to Report Filtered Events in S/W H/K

REASON FOR CHANGE:

A stated need by the ACIS SI team and CXC Calibration team for additional statistics relating to on-board event rejection via pulse-height, grade, and window filters. This was the subject of ACIS problem report M00030901 (SPR 132). Also, as reported in M99101202 (SPR 131), the bogus bias parity errors reported in timed exposure 5x5 mode must be described.

DESCRIPTION OF CHANGE:

(a) Patch `SwHousekeeper::SwHousekeeper()` to accept an additional 54 housekeeping codes, defined in an update to "*acis_h/interface.h*".

(b) Replace `PmEvent::filterEvent()` with a version that reports the acceptance or rejection of each event to the S/W housekeeper.

(c) Document bogus bias parity errors within the IP&CL release notes.

	SIGNATURE	DATE	REMARKS:
ORIGINATOR	Peter Ford	03/21/03	Final sign-off version
MECHANICAL			
ELECTRICAL			
SOFTWARE			
STRUCTURE			
FABRICATION			
SCIENCE			
SYSTEMS ENG.			
QUALITY			
PROJ. ENGINEER			
DEPUTY PM			
PROJ. MANAGER			

1. Reasons for Change

1.1. Characterizing event rejection

In order to understand the on-orbit increase in ACIS CTI and develop ways to improve it or correct for it, several groups have expressed an interest in knowing which particular grade codes are rejected by the BEP filters. At present, only the total number of events rejected for (a) pulse height, (b) grade code, or (c) spatial window filter are reported in the exposure packets for each FEP. It would be useful to break (b) into separate tallies for each grade code.

1.2. Reporting event rejection

The most scientifically useful statistic would be the breakdown into individual grades for each exposure, but this would necessarily decrease telemetry throughput and require considerable changes to the ground data system to accommodate the necessary changes in the exposure packet formats.

A second possibility would be to accumulate the statistics during the science run and report them in an expanded *scienceReport* packet. However, this would not track variations in the event background during the science run, and would also necessitate a change in ground software.

The reporting method chosen in this ECO uses software housekeeping packets which are generated once every 64 seconds, provided space is available in the telemetry queue. Since each new housekeeping channel requires 8 bytes of BEP RAM D-Cache, only a limited number of new channels (64) may be added before it becomes necessary to patch the entire software housekeeping task. We propose to use 54 new channels, 9 per active FEP.

1.3. Bogus bias parity errors

In timed-exposure 5x5 mode, the FEP executes the same code as in 3x3 mode until a local 3x3 maximum has been identified within the input image array. It then waits until the following image row has been processed by the firmware, and appends the 16 surrounding pixel and bias values to a ring buffer record, augmenting the 9 already there. In 5x5 mode, this process start with the 5x5 array centered on the third image line, and stops 2 lines before the last image line. However, in order to share the 3x3 code, it scans each row from the second pixel to one before the last on each line. Therefore, the left-most column of the 5x5 array reported from events centered in the second column is always incorrect—it is in fact the right-most column of the previous row. Similarly, for events centered in the last-but-one column (*i.e.*, column 1022), the right-most column of the 5x5 is bogus.

In practice, the bogus pixels are easily identified and discarded by ground software, but there is one instance in which a complication arises—the extreme “top right” bias entry reported at row 0 and column 1022 is actually at row -1 and column 0, *i.e.*, a FEP memory location that is not within the physical bias memory. When asked to retrieve a 16-bit number from this location, the FEP signals a parity error, but an “illogical” one since the parity code 0x15 that it assigns is not one that can be generated by a normal parity error. Unlike normal parity errors, this cannot be corrected by FEP software and is therefore likely to be reported several times during the course of a lengthy 5x5 timed-exposure run.

As luck would have it, the equivalent non-physical location at row 1024, column 0 doesn’t generate a parity error.

2. Proposed Changes

2.1. Inline patch to SwHousekeeper::SwHousekeeper.C

The following patch to the SwHousekeeper class constructor adds 64 new software housekeeping channels.

```
   .text
   .globlswhousekeeper_lst_0034_0034
   .entswhousekeeper_lst_0034_0034
swhousekeeper_lst_0034_0034:
   li $2,91+64
   .endswhousekeeper_lst_0034_0034
```

2.2. Replacement of PmEvent::filterEvent()

This method is called by all science modes to determine whether an event is to be accepted (returns `BoolTrue`) or rejected (returns `BoolFalse`). It calls virtual methods *phFilter->filterEvent*, *gradeFilter->filterEvent*, and *windowFilter->filterEvent* to determine whether to reject the event on the basis of pulse height, grade code, or window location, respectively.

```
Boolean PmEvent::filterEvent(PixelEvent& event)
{
    DebugProbe probe;
    // ---- Test pulse height range ----
    if (phFilter->filterEvent (event) == BoolFalse)
    {
      return BoolFalse;
    }
    // ---- Test event grade ----
    if (gradeFilter->filterEvent (event) == BoolFalse)
    {
      return BoolFalse;
    }
    // ---- Test processing windows ----
    if (windowFilter->filterEvent (event) == BoolFalse)
    {
      return BoolFalse;
    }
    return BoolTrue;
}
```

The replacement function, which will be compiled as a method of the `Test_PmEvent` subclass and subsequently called as *PmEvent::filterEvent()* via an address patch, retains the original functions but adds a call to *swHousekeeper.report()*.

```
class Test_PmEvent : public PmEvent
{
    public:
      ~Test_PmEvent();
      Boolean filterEvent(PixelEvent& event);
};

Test_PmEvent::~Test_PmEvent()
{
}

Boolean Test_PmEvent::filterEvent(PixelEvent& event)
{
```

```
    SwFilterId swStat = SW_FILT_NONE;
    Boolean retCode = BoolFalse;

    // ---- Test pulse height range ----
    if (phFilter->filterEvent (event) == BoolFalse)
    {
      swStat = SW_FILT_EVENT;
    }
    // ---- Test event grade ----
    else if (gradeFilter->filterEvent (event) == BoolFalse)
    {
      switch (event.getGrade()) {
      case SW_GRADE_CODE1:
          swStat = SW_FILT_GRADE1;
          break;
      case SW_GRADE_CODE2:
          swStat = SW_FILT_GRADE2;
          break;
      case SW_GRADE_CODE3:
          swStat = SW_FILT_GRADE3;
          break;
      case SW_GRADE_CODE4:
          swStat = SW_FILT_GRADE4;
          break;
      case SW_GRADE_CODE5:
          swStat = SW_FILT_GRADE5;
          break;
      default:
          swStat = SW_FILT_OTHER;
          break;
      }
    }
    // ---- Test processing windows ----
    else if (windowFilter->filterEvent (event) == BoolFalse)
    {
      swStat = SW_FILT_WIN;
    }
    else retCode = BoolTrue;

    swHousekeeper.report(SW_FILT(swStat, getFepId()),
      getExposureInfo().getExposureNumber());
    return retCode;
}
```

2.3. Updates to acis_h/interface.h

The new event filter statistics are defined by adding the following to *interface.h*:

```
/* ---- Software Housekeeping sub-codes for BEP filter statistics ---- */
enum SwFilterId {
    SW_FILT_NONE,    /* events unfiltered */
    SW_FILT_ENERGY,  /* events filtered by energy */
    SW_FILT_GRADE1,  /* events filtered by SW_GRADE_CODE1 */
    SW_FILT_GRADE2,  /* events filtered by SW_GRADE_CODE2 */
    SW_FILT_GRADE3,  /* events filtered by SW_GRADE_CODE3 */
    SW_FILT_GRADE4,  /* events filtered by SW_GRADE_CODE4 */
    SW_FILT_GRADE5,  /* events filtered by SW_GRADE_CODE5 */
    SW_FILT_OTHER,   /* events filtered by other grade */
    SW_FILT_WIN,     /* events filtered by window */
    SW_FILT_COUNT
};
```

```
/* ---- Special grade codes reported in software housekeeping ---- */
enum SwSpecialGrade {
    SW_GRADE_CODE1 = 24,
    SW_GRADE_CODE2 = 66,
    SW_GRADE_CODE3 = 107,
    SW_GRADE_CODE4 = 214,
    SW_GRADE_CODE5 = 255
};

#define SW_FILT_SIZE (FEP_COUNT*SW_FILT_COUNT)
#define SW_FILT(code,fep) \
      SwStatistic(SWSTAT_FILT_BASE+(code)+(fep)*SW_FILT_COUNT)
```

and by appending the following to the `SwStatistic` enumeration:

```
SWSTAT_FILT_FEP_0_NONE,/* FEP_0 events unfiltered [Arg: expNum] */
SWSTAT_FILT_BASE = SWSTAT_FILT_FEP_0_NONE, /* Sci Mode Event Stats */
SWSTAT_FILT_FEP_0_ENERGY,/* FEP_0 events filtered by energy */
SWSTAT_FILT_FEP_0_GRADE1,/* FEP_0 events filtered by SW_GRADE_CODE1 */
SWSTAT_FILR_FEP_0_GRADE2,/* FEP_0 events filtered by SW_GRADE_CODE2 */
SWSTAT_FILT_FEP_0_GRADE3,/* FEP_0 events filtered by SW_GRADE_CODE3 */
SWSTAT_FILT_FEP_0_GRADE4,/* FEP_0 events filtered by SW_GRADE_CODE4 */
SWSTAT_FILT_FEP_0_GRADE5,/* FEP_0 events filtered by SW_GRADE_CODE5 */
SWSTAT_FILT_FEP_0_OTHER,/* FEP_0 events filtered by other grade */
SWSTAT_FILT_FEP_0_WIN,/* FEP_0 events filtered by window */

SWSTAT_FILT_FEP_1_NONE,/* FEP_1 events unfiltered [Arg: expNum] */
SWSTAT_FILT_FEP_1_ENERGY,/* FEP_1 events filtered by energy */
SWSTAT_FILT_FEP_1_GRADE1,/* FEP_1 events filtered by SW_GRADE_CODE1 */
SWSTAT_FILT_FEP_1_GRADE2,/* FEP_1 events filtered by SW_GRADE_CODE2 */
SWSTAT_FILT_FEP_1_GRADE3,/* FEP_1 events filtered by SW_GRADE_CODE3 */
SWSTAT_FILT_FEP_1_GRADE4,/* FEP_1 events filtered by SW_GRADE_CODE4 */
SWSTAT_FILT_FEP_1_GRADE5,/* FEP_1 events filtered by SW_GRADE_CODE5 */
SWSTAT_FILT_FEP_1_OTHER,/* FEP_1 events filtered by other grade */
SWSTAT_FILT_FEP_1_WIN,/* FEP_1 events filtered by window */

SWSTAT_FILT_FEP_2_NONE,/* FEP_2 events unfiltered [Arg: expNum] */
SWSTAT_FILT_FEP_2_ENERGY,/* FEP_2 events filtered by energy */
SWSTAT_FILT_FEP_2_GRADE1,/* FEP_2 events filtered by SW_GRADE_CODE1 */
SWSTAT_FILT_FEP_2_GRADE2,/* FEP_2 events filtered by SW_GRADE_CODE2 */
SWSTAT_FILT_FEP_2_GRADE3,/* FEP_2 events filtered by SW_GRADE_CODE3 */
SWSTAT_FILT_FEP_2_GRADE4,/* FEP_2 events filtered by SW_GRADE_CODE4 */
SWSTAT_FILT_FEP_2_GRADE5,/* FEP_2 events filtered by SW_GRADE_CODE5 */
SWSTAT_FILT_FEP_2_OTHER,/* FEP_2 events filtered by other grade */
SWSTAT_FILT_FEP_2_WIN,/* FEP_2 events filtered by window */

SWSTAT_FILT_FEP_3_NONE,/* FEP_3 events unfiltered [Arg: expNum] */
SWSTAT_FILT_FEP_3_ENERGY,/* FEP_3 events filtered by energy */
SWSTAT_FILT_FEP_3_GRADE1,/* FEP_3 events filtered by SW_GRADE_CODE1 */
SWSTAT_FILT_FEP_3_GRADE2,/* FEP_3 events filtered by SW_GRADE_CODE2 */
SWSTAT_FILT_FEP_3_GRADE3,/* FEP_3 events filtered by SW_GRADE_CODE3 */
SWSTAT_FILT_FEP_3_GRADE4,/* FEP_3 events filtered by SW_GRADE_CODE4 */
SWSTAT_FILT_FEP_3_GRADE5,/* FEP_3 events filtered by SW_GRADE_CODE5 */
SWSTAT_FILT_FEP_3_OTHER,/* FEP_3 events filtered by other grade */
SWSTAT_FILT_FEP_3_WIN,/* FEP_3 events filtered by window */

SWSTAT_FILT_FEP_4_NONE,/* FEP_4 events unfiltered [Arg: expNum] */
SWSTAT_FILT_FEP_4_ENERGY,/* FEP_4 events filtered by energy */
```

```
SWSTAT_FILT_FEP_4_GRADE1,/* FEP_4 events filtered by SW_GRADE_CODE1 */
SWSTAT_FILT_FEP_4_GRADE2,/* FEP_4 events filtered by SW_GRADE_CODE2 */
SWSTAT_FILT_FEP_4_GRADE3,/* FEP_4 events filtered by SW_GRADE_CODE3 */
SWSTAT_FILT_FEP_4_GRADE4,/* FEP_4 events filtered by SW_GRADE_CODE4 */
SWSTAT_FILT_FEP_4_GRADE5,/* FEP_4 events filtered by SW_GRADE_CODE5 */
SWSTAT_FILT_FEP_4_OTHER,/* FEP_4 events filtered by other grade */
SWSTAT_FILT_FEP_4_WIN,/* FEP_4 events filtered by window */

SWSTAT_FILT_FEP_5_NONE,/* FEP_5 events unfiltered [Arg: expNum] */
SWSTAT_FILT_FEP_5_ENERGY,/* FEP_5 events filtered by energy */
SWSTAT_FILT_FEP_5_GRADE1,/* FEP_5 events filtered by SW_GRADE_CODE1 */
SWSTAT_FILT_FEP_5_GRADE2,/* FEP_5 events filtered by SW_GRADE_CODE2 */
SWSTAT_FILT_FEP_5_GRADE3,/* FEP_5 events filtered by SW_GRADE_CODE3 */
SWSTAT_FILT_FEP_5_GRADE4,/* FEP_5 events filtered by SW_GRADE_CODE4 */
SWSTAT_FILT_FEP_5_GRADE5,/* FEP_5 events filtered by SW_GRADE_CODE5 */
SWSTAT_FILT_FEP_5_OTHER,/* FEP_5 events filtered by other grade */
SWSTAT_FILT_FEP_5_WIN,/* FEP_5 events filtered by window */
```

Finally, the following new section documents the bogus bias parity errors:

A feature in the implementation of Timed Exposure 5x5 mode within the FEP can generate bogus bias parity errors. With the *digestbiaserror* patch installed, these can be readily identified within *patchDataBiasError* packets from the following unique combination of field values:

```
biasErrors = {
  row             = 0
  column          = 1022
  evenPixelFlags  = 15
  oddPixelFlags   = 0
}
```

All *patchDataBiasError* elements with this combination of values should be discarded.

3. Controlled Sources

reportgrade1	
reportgrade1.mak	Generates ReportGrade1 object files.
reportgrade1.C	BEP patch to report grade rejections in S/W housekeeping reports.
reportgrade1a.S	Inline BEP patch to increase number of S/W housekeeping codes.
reportgrade1.pkg	Build file for creating an optional ReportGrade1 patch.
eco-1021.doc	Engineering change order describing the ReportGrade1 patch.
reportgrade1/testsuite	
makebias	Generate a bias image and copy it to the image loader.
makeimage	Generate an image with events and copy it to the image loader.
reportgrade1/testsuite/smoke	
Makefile	Run tests of the ReportGrade1 patch.
runtest.tcl	Test ReportGrade1 in TE mode.
runtestcc.tcl	Test ReportGrade1 in CC mode.

4. TESTING

All tests were performed on the ACIS Engineering Unit using 6 FEPs, an image loader, and an L-RCTU interface. After setting up a *shim* process to handle I/O between UNIX and the L-RCTU, the tests were controlled by scripts written in the *expect* dialect of TCL.

4.1. TE Mode Test

An *expect* procedure, "*smoke/runtest.tcl*", performs a science run with the *standard* and *reportgrade1* patches. The following steps are performed:

1. A command pipe is spawned down which ACIS commands will be written.
2. A telemetry pipe is spawned, terminating in the "*psci -m -u*" packet monitoring function with *expect* examining the standard output.
3. ACIS is cold-booted.
4. Software housekeeping, DEA replacement, and standard flight patches are applied.
5. The *"reportgrade1.bcmd"* patch file is applied.
6. ACIS is warm-booted.
7. The appropriate FEPs are powered up.
8. A bias map containing the same value in each pixel of a given quadrant is written to the image loader.
9. A *te_3x3* parameter block is sent to ACIS. It calls for 6 FEPs to be run in faint-with-bias mode, using a window block.
10. A 2-D window block is sent to ACIS.
11. A bias-only science run is performed.
12. An image containing 48 events scattered across it is written to the image loader. The events have been chosen in coordination with the parameter and window blocks so that all possible `SWUSER_FILT_*` actions will be reported, as shown in the following table.

Name	Grade	PH	Count	Filter Type	Comment
event_0	0	500	5	SW_FILT_NONE	The window block specifies rejection of one of the grade 0 events from each FEP.
			1	SW_FILT_WIN	
event_1	248	500	6	SW_FILT_OTHER	The parameter block specifies rejection of grades 24, 66, 107, 214, 248, and 255.
event_2	24	650	6	SW_FILT_GRADE1	
event_3	66	650	6	SW_FILT_GRADE2	
event_4	107	1100	6	SW_FILT_GRADE3	
event_5	214	1100	6	SW_FILT_GRADE4	
event_6	255	950	6	SW_FILT_GRADE5	
event_7	0	2000	6	SW_FILT_EVENT	The parameter block specifies rejection of pulse heights above 1500 ADU.

13. A *teFaintBias* science run is started.

14. When *expect* first receives a "pseudo-user" packet reporting a `SWUSER_FILT_*` action, it sends a *stopScience* command to the BEP.
15. *expect* waits until a *scienceReport* packet then a *swHousekeeping* packet are received. Meanwhile, it inspects *exposureTeFaint* packets, counting the number of accepted and dropped events reported in each.
16. Finally, *expect* pipes the BEP's raw packet stream through *"psci -s -p"* and examines the software housekeeping packets, comparing the numbers accepted and dropped against the number expected from the *makeimage* script and the totals reported in the *exposureTeFaint* packets. The test PASSES when the numbers are equal and non-zero; otherwise it FAILS.

4.2. CC3x3 Mode Test

An *expect* procedure, "*smoke/runtestcc.tcl*", perform a CC 3x3 science run with the *standard*, *cc3x3* and *reportgrade1* patches. The following steps are performed:

1. A command pipe is spawned down which ACIS commands will be written.
2. A telemetry pipe is spawned, terminating in the "*psci -m -u*" packet monitoring function with *expect* examining the standard output.
3. ACIS is cold-booted.
4. Software housekeeping, DEA replacement, and standard flight patches are applied.
5. The *"cc3x3.bcmd"* and *"reportgrade1.bcmd*" patch files are applied.
6. ACIS is warm-booted.
7. The appropriate FEPs are powered up.
8. A bias map containing the same value in each pixel of a given quadrant is written to the image loader.
9. A *cc_3x3* parameter block is sent to ACIS. It calls for 6 FEPs to be run in faint mode, using a window block.
10. A 1-D window block is sent to ACIS.
11. A bias-only science run is performed.
12. An image containing a small number of events scattered across it is written to the image loader. The events have been chosen in coordination with the parameter and window blocks so that all possible `SWUSER_FILT_*` actions will be reported. The assignments are identical to those described in step 12 of section 4.1, above.
13. A *ccFaint3x3* science run is started.
14. When *expect* first receives a "pseudo-user" packet reporting a `SWUSER_FILT_*` action, it sends a *stopScience* command to the BEP.
15. *expect* waits until a *scienceReport* packet then a *swHousekeeping* packet are received. Meanwhile, it inspects *exposureCcFaint* packets, counting the number of accepted and dropped events reported in each.

16. Finally, *expect* pipes the BEP's raw packet stream through *"psci -s -p"* and examines the software housekeeping packets, comparing the numbers accepted and dropped against the number expected from the *makeimage* script and the totals reported in the *exposureTeFaint* packets. The test PASSES when the numbers are equal and non-zero; otherwise it FAILS.

MIT CSR ACIS	**ENGINEERING CHANGE ORDER**	**ECO No. 36–1025**

CENTER FOR SPACE RESEARCH
MASSACHUSETTS INSTITUTE OF TECHNOLOGY

DWG. NO.	NEW REV.	DRAWING TITLE
36-58030.24	A	*smtimedlookup*: Patch to make TE mode selection by table lookup
36-58030.05	B	*eventhist*: Patch to implement TE Event Histogram mode

REASON FOR CHANGE:

Each new timed-exposure mode must patch several methods of *smtimedexposure*. When only one new mode (*eventhist*) existed, this was simple. The addition of new modes (*ctireport1*, *ctireport2*, etc.) necessitate a change to a table-driven patchable interface.

DESCRIPTION OF CHANGE:

(a) Patch *SmTimedExposure* methods `setupProcess`, `setupFepBlock`, and `terminate` to be driven from static tables `smTimedLookupMode`, `smTimedSetupFep`, `smTimedLookup3x3`, `smTimedLookup5x5`, and `smTimedTerminate`, each of which is a 16-element array containing pointers to the methods that are to handle the current `fepMode`.

(b) In the *eventhist* patch, replace the patched `SmTimedExposure::setupProcess` by `setupEventHist`.

	SIGNATURE	DATE	REMARKS:
ORIGINATOR	Peter Ford	03/21/03	Final sign-off version
MECHANICAL			
ELECTRICAL			
SOFTWARE			
STRUCTURE			
FABRICATION			
SCIENCE			
SYSTEMS ENG.			
QUALITY			
PROJ. ENGINEER			
DEPUTY PM			
PROJ. MANAGER			

1. Reason for Change

1.1. Multiple new event modes

When new event modes are made available via software patches, the BEP must respond correctly to the new `fepMode` and `bepPackingMode` values that it finds in loadTeBlock parameter bock slots. This behavior must depend on whether the corresponding patches have been loaded.

When only a single new mode, *eventhist*, was added, it was merely necessary to patch three *SmTimedExposure* methods, `setupProcess`, `setupFepBlock`, and `terminate`, and force them to recognize the new `fepMode` value. With the addition of new modes, *e.g.*, *ctireport1*, *ctireport2*, it is no longer feasible to patch *SmTimedExposure* in the same way, since each combination of patches would need its own methods.

The solution to this problem is to create a separate patch to *SmTimedExposure* that runs each possible mode from the contents of a set of tables. Unassigned modes will be given null entries in the tables and patches that implement new modes need merely update the tables in an appropriate manner.

2. Proposed Changes

2.1. Patched SmTimedExposure methods

```
class Test_SmTimedExposure : public SmTimedExposure
{
public:
  Test_SmTimedExposure();
  ~Test_SmTimedExposure();
  Boolean setupProcess();
  void setup3x3();
  void setup5x5();
  Boolean setupFepBlock(FepId fep, FEPparmBlock& fepblock);
  void setupFepTeRaw(FepId fep, FEPparmBlock& fepblock);
  void setupFepTeHist(FepId fep, FEPparmBlock& fepblock);
  void setupFepTe3x3(FepId fep, FEPparmBlock& fepblock);
  void setupFepTe5x5(FepId fep, FEPparmBlock& fepblock);
  void terminate();
  void normalTermination();
};
```

The *smtimedlookup* patch defines a `Test_SmTimedExposure` class, a sub-class of `SmTimedExposure`, and overrides the following methods:

`setupProcess`—this method uses `fepMode` as an index into the `smTimedLookupMode` table. If the value is null, the *loadTeBlock* is rejected. Otherwise, it is interpreted as a function address, which is called to initialize the mode.

`setupFepBlock`—this method uses `fepMode` as an index into the `smTimedSetupFep` table. If the value is null, the *loadTeBlock* is rejected. Otherwise, it is interpreted as a function address, which is called to set up the mode. The `fepblock` object is then initialized.

`terminate`—this method uses `fepMode` as an index into the `smTimedTerminate` table. If the value is null, an error (`SWSTAT_TE_BAD_FEP_MODE`) is reported to software housekeeping. Otherwise, it is interpreted as a function address, which is called to terminate the mode.

2.2. New Test_SmTimedExposure methods

The following methods are called from the patched `setupProcess`:

`setup3x3`—this method uses `bepPackingMode` as an index into the `smTimedLookup3x3` table. If the value is null, the *loadTeBlock* is rejected. Otherwise, it is interpreted as a function address, which is called to further initialize 3x3 event modes.

`setup5x5`—this method uses `bepPackingMode` as an index into the `smTimedLookup5x5` table. If the value is null, the *loadTeBlock* is rejected. Otherwise, it is interpreted as a function address, which is called to further initialize 5x5 event modes.

The following methods are called because their entry-point addresses are stored in the `smTimedLookupMode`, `smTimedSetupFep`, `smTimedLookup3x3`, `smTimedLookup5x5`, and `smTimedTerminate` arrays, replacing code that was executed in the single `setupProcess` method of the unpatched `SmTimedLookup` class.

`setupFepTeRaw`—initializes the unpatched TE raw mode.

`setupFepTeHist`—initializes the unpatched TE raw histogram mode.

`setupFepTe3x3`—initializes the unpatched TE 3x3 event mode.

`setupFepTe5x5`—initializes the unpatched TE 5x5 event mode.

`setupFepTeRaw`—initializes the unpatched TE raw mode.

`normalTermination`—invokes the unpatched termination actions.

2.3. Static method tables for SmTimedExposure

The following 16-element arrays are defined:

```
typedef (Test_SmTimedExposure::* lookupPtr)();
static lookupPtr smTimedLookupMode[16];
static lookupPtr smTimedLookup3x3[16];
static lookupPtr smTimedLookup5x5[16];
static lookupPtr smTimedTerminate[16];

typedef void (Test_SmTimedExposure::* setupFepPtr)(FepId fep,
                FEPParmBlock& fepblock);
static setupFepPtr smTimedSetupFep[16];
```

They are statically initialized to perform the same functions as the unpatched `setupProcess`. In some cases, this can be achieved by existing member functions of `SmTimedExposure`. In others, it is necessary to add a new function to `Test_SmTimedExposure`. The C++ compiler will accept this mixed usage provided the functions are cast appropriately, *e.g.*,

```
static lookupPtr smTimedLookupMode[16] = {
  (lookupPtr) &SmTimedExposure::setupRaw,
  (lookupPtr) &SmTimedExposure::setupHist,
  &Test_SmTimedExposure::setup3x3,
  &Test_SmTimedExposure::setup5x5,
};
```

2.4. Updated replaceFuncBcmd.pl script

When building an ACIS patch, function references in named memory locations are updated in response to `"func"` commands supplied in `*.pkg` file of that patch. The work is performed by the *tools/bin/replaceFuncBcmd.pl* script. To use the same logic to insert func-

tion references into the method tables described in §2.3, the syntax has been extended from:

`func` *address_in_memory_map entry_point_in_patch*

to include

`func` *address_in_memory_map*`[`*element_index*`]` *entry_point_in_patch*

Since the size of the elements in the static method arrays is 8 bytes, the effect of specifying an [*element_index*] is to store the 4-byte *entry_point_in_patch* value into memory location *address_in_memory_map*+8**element_index*.

2.5. Updated eventhist patch

The only changes to this version of the *eventhist* patch are that the `Test_SmTimedExposure` class has been renamed `Test1_SmTimedExposure` so as not to clash with the test class created by the *smtimedlookup* patch, and the `setupProcess` method has been replaced by `setupEventHist` which is considerably smaller, since it no longer has to parse the `fepMode` and `bepPackingMode` values. In addition, the `func` commands within *eventhist.pkg* have been changed to update the static method arrays, as described in §2.4.

3. Controlled Sources

smtimedlookup	
smtimedlookup.mak	Generates SmTimedLookup object files.
smtimedlookup.C	BEP patch to control TE science mode via table lookup.
smtimedlookup.pkg	Build file for creating an optional SmTimedLookup patch.
eco-1025.doc	Engineering change order describing the SmTimedLookup patch.
smtimedlookup/testsuite/smoke	
runtest.tcl	Test 3x3 event mode with SmTimedLookup patch.
eventhist	
eventhist.mak	Generates EventHist object files.
eventhist.pkg	Build file for creating an optional EventHist patch.
hamming.C	Adds Hamming code to histogram value.
hanning.C	Adds Hamming code to histogram value.
interface.h	Updated BEP interface description.
ipcl_struct.tsv	Updated description of ACIS IP&CL.
ipcl_struct.xs3	Updated description of ACIS IP&CL.
pmteevhist.H	Implements event histogram mode.
pmteevhist.C	Implements event histogram mode.
eventhist/testsuite	
hist.txt	Expected *psci* output from smoke tests.
raw..mg	Expected *psci* output from smoke test
rawhist.txt	Expected *psci* output from smoke test
makebias	Generate a bias image and copy it to the image loader.
makeimage	Generate an image with events and copy it to the image loader.
standard.bcmd	Standard patch load.

eventhist/testsuite/smoke	
Makefile	Run tests of the SmTimedLookup+EventHist patches.
runtest.tcl	Test *evhist* mode with SmTimedLookup+EventHist patches.
runtest2.tcl	Test 3x3 event mode with SmTimedLookup+EventHist patches.

4. TESTING

The test is performed on the ACIS Engineering Unit using 6 FEPs, an image loader, and an L-RCTU interface. After setting up a *shim* process to handle I/O between UNIX and the L-RCTU, the test is controlled by scripts written in the *expect* dialect of TCL.

An *expect* procedure, *"smtimedlookup/smoke/runtest.tcl"*, tests *smtimedlookup* alone by performing a faint-mode science run with the *"standard"* and *"smtimedlookup"* patches. A second *expect* procedure, *"eventhist/smoke/runtest.tcl"*, tests it with an optional processing mode by performing an event-histogram science run with the *"standard"*, *"smtimedlookup"*, and *"eventhist"* patches. A third procedure, *"eventhist/smoke/runtest2.tcl"*, tests the three original modes, raw, histogram, and event, with the same patches. The following steps are performed:

4.1. Test of 3x3 faint mode with smTimedLookup patch

1. A command pipe is spawned down which ACIS commands will be written.
2. A telemetry pipe is spawned, terminating in the "*psci -m -u*" packet monitoring function with *expect* examining the standard output.
3. ACIS is cold-booted.
4. Software housekeeping, DEA replacement, and standard flight patches are applied.
5. The *"smtimedlookup"* patch file is applied.
6. ACIS is warm-booted.
7. The appropriate FEPs are powered up.
8. A bias map containing the same value in each pixel of a given quadrant is written to the image loader.
9. A *loadTeBlock* parameter block is sent to ACIS. It calls for 6 FEPs to be run in 3x3 faint mode.
10. A bias-only science run is executed.
11. An image file containing 48 events is written to the image loader.
12. A science run is started.
13. The run is stopped after the first exposure record is reported from the last FEP to be loaded. If no such record is found, the test fails.

4.2. Test of evhist mode with smTimedLookup and eventHist patches

1. A command pipe is spawned down which ACIS commands will be written.
2. A telemetry pipe is spawned, terminating in the "*psci -m -u*" packet monitoring function with *expect* examining the standard output.

3. ACIS is cold-booted.
4. Software housekeeping, DEA replacement, and standard flight patches are applied.
5. The *"smtimedlookup" and "eventhist"* patch files are applied.
6. ACIS is warm-booted.
7. The appropriate FEPs are powered up.
8. A bias map containing the same value in each pixel of a given quadrant is written to the image loader.
9. An *evhist* parameter block is sent to ACIS. It calls for 6 FEPs to be run in event histogram mode.
10. A science run is started.
11. The run is stopped after the first exposure record is reported from the last FEP to be loaded. If no such record is found, the test fails.
12. The recorded telemetry stream is passed through *psci* with options *"–a –B –l foo"* and the last FEP's histogram compared against "eventhist/testsuite/hist.txt". A match signals a pass; otherwise the test fails.

4.3. Further tests of smTimedLookup with eventHist

1. A command pipe is spawned down which ACIS commands will be written.
2. A telemetry pipe is spawned, terminating in the "*psci -m -u*" packet monitoring function with *expect* examining the standard output.
3. ACIS is cold-booted.
4. Software housekeeping, DEA replacement, and standard flight patches are applied.
5. The *"smtimedlookup" and "eventhist"* patch files are applied.
6. ACIS is warm-booted.
7. The appropriate FEPs are powered up.
8. A set of 6 science run are performed, with FEP and BEP parameters as specified in Table 1.

Table 1: Test of normal science modes with smTimedExposure and evHist patches

Test	fepMode	bepPackingMode	reload bias map?	reload image map?	expected data packet
1	FEP_TE_MODE_RAW	BEP_TE_MODE_FAINT	No	Yes	dataTeRaw
2	FEP_TE_MODE_HIST	BEP_TE_MODE_FAINT	No	No	dataTeHist
3	FEP_TE_MODE_EV3x3	BEP_TE_MODE_FAINT	Yes	Yes	dataTeFaint
4	FEP_TE_MODE_EV3x3	BEP_TE_MODE_FAINTBIAS	Yes	Yes	dataTeFaintBias
5	FEP_TE_MODE_EV3x3	BEP_TE_MODE_GRADED	Yes	Yes	dataTeGraded
6	FEP_TE_MODE_EV5x5	BEP_TE_MODE_FAINT	Yes	Yes	dataTeVeryFaint

9. A *loadTeBlock* parameter block is sent to ACIS. It calls for 6 FEPs to be run in the appropriate `fepMode` and `bepPackingMode` in Table 1. The run is stopped after the first exposure record is reported from the last FEP to be loaded. If no such record is found, the test fails.
10. If Table 1 indicates that a bias map is to be loaded, one containing the same value in each pixel of a given quadrant is written to the image loader, a bias-only run is started, and the script waits for the ending *scienceReport* packet.
11. If Table 1 indicates that an image map is to be loaded, an image file containing 48 events is written to the image loader.
12. A science run is started and a *stopScience* command is sent after the first record of the type indicated in Table 1 is encountered.
13. The script waits for the *scienceReport* packet and verifies that its `terminationCode` is `SMTERM_STOPCMD` (1).
14. If all 6 tests complete without error, the test passes; otherwise, it fails.

MIT CSR ACIS	**ENGINEERING CHANGE ORDER**	**ECO No. 36–1026**

CENTER FOR SPACE RESEARCH
MASSACHUSETTS INSTITUTE OF TECHNOLOGY

DWG. NO.	NEW REV.	DRAWING TITLE
58030.25	A	*ctireport1*: Patch #1 to Implement Precursor Charge Reporting
58030.26	A	*ctireport2*: Patch #2 to Implement Precursor Charge Reporting

REASON FOR CHANGE:
Desire to report precursor charge in each column of each reported event.

DESCRIPTION OF CHANGE:
(a) Create new `Test2_SmTimedExposure` class methods to call from *smtimedlookup* tables for mode-dependent setup and termination;

(b) create `PmTeCti1` class to update telemetry packet headers (Mode 2, only);

(c) create new FEP routines, `FEPtestCti1`, `FEPappendCti1`, `FEPtestCti2`, and `FEPappendCti2`, to report precursor charge in existing data structures;

(d) create inline FEP patches: `ctireport1fep2` and `ctireport2fep2` to cause FEP code to branch to the routines.

	SIGNATURE	DATE	REMARKS:
ORIGINATOR	Peter Ford	03/21/03	Final sign-off version
MECHANICAL			
ELECTRICAL			
SOFTWARE			
STRUCTURE			
FABRICATION			
SCIENCE			
SYSTEMS ENG.			
QUALITY			
PROJ. ENGINEER			
DEPUTY PM			
PROJ. MANAGER			

1. Reasons for Change

Damage to the ACIS front-illuminated CCDs has resulted in an increase of charge-transfer inefficiency (CTI), leading to a loss of energy resolution. If the effects of CTI can be fully characterized, it may be possible to correct for the charge loss, provided an account can be kept of the charge in each CCD column that contributes charge to an event.

The two optional patches, *ctiReport1* and *ctiReport2*, are intended as tests of the ability of the ACIS front end processors (FEPs) to report the precursor charge, *i.e.*, the charge that was clocked out of the contributing columns. Both patches run as new timed-exposure modes.

In a manner similar to the *eventHist* patch, which is built on raw histogram mode, *ctiReport1* is based on 5x5 event mode, and *ctiReport2* on 3x3 event mode. Neither of the new patches uses new output packet formats. *ctiReport1* produces *exposureTeFaint* and *dataTeVeryFaint* telemetry packets, using the "outer" 16 elements of the `pulseheights` array to report the precursor charge value and location. *ctiReport2* produces *exposureTeFaint* and *dataTeFaint* packets, using 3 low-order bits of the corner pixels in the `pulseheights` array.

2. Proposed Changes

2.1. Changes to the `SmTimedExposure` Class

New `Test2_SmTimedExposure` and `Test3_SmTimedExposure` classes are defined. Their *setupCti1*, *setupCti1Fep*, *setupCti2Fep*, and *terminate* methods will be patched as replacement elements in static arrays of the `SmTimedExposure` class.

```
class Test2_SmTimedExposure: public SmTimedExposure
{
public:
  ~Test2_SmTimedExposure();
  void setupCti1();
  void setupCti1Fep(FepId fep, FEPparmBlock& fepblock);
  void terminate();
};
class Test3_SmTimedExposure : public SmTimedExposure
{
public:
  Test3_SmTimedExposure();
  ~Test3_SmTimedExposure();
  void setupCti2Fep(FepId fep, FEPparmBlock& fepblock);
  void terminate();
};
```

The methods select mode-dependent functions by means of the `FepMode` and `BepPackingMode` fields of the `teBlock`. These fields are used as indices into arrays of pointers to member functions:

```
static lookupPtr smTimedLookupMode[16];
static setupFepPtr smTimedSetupFep[16];
static lookupPtr smTimedLookup3x3[16];
static lookupPtr smTimedLookup5x5[16];
static lookupPtr smTimedTerminate[16];
```

2.2. New PmTeCti1 Class

This is a subclass of `PmTeFaint5x5`, and is used to change the values of the `formatId` keywords of exposure and event packet headers to `TTAG_SCI_TE_REC_CTI1` and `TTAG_SCI_TE_DAT_CTI1`, respectively:

```
class PmTeCti1 : public PmTeFaint5x5
{
public:
    ~PmTeCti1();
    void    initialize();
    Boolean finishExposure();
};
```

A companion `PmTeCti2` class is not needed to implement the *ctireport2* mode since the latter does not define new telemetry packet formats.

Class and Method	Description
`Test2_SmTimedExposure`	
`~Test2_SmTimedExposure()`	Destructor—unused.
`setupCti1()`	Initializes `pmTeCti1` objects for each FEP.
`setupCti1Fep()`	Initializes the FEP control block and calls the *fepManager* to load default FEP code, inline FEP patches, and the new FEP routines.
`terminate()`	Stops the science run, resets the FEPs, and writes a science report.
`PmTeCti1`	
`~PmTeCti1()`	Destructor—unused.
`initialize()`	Called by Test2_SmTimedExposure::setupCti1 to initialize the event packet headers.
`finishExposure()`	Called to post each exposure packet with updated header format.
`Test3_SmTimedExposure`	
`setupCti2Fep()`	Initializes the FEP control block and calls the fepManager to load default FEP code, inline FEP patches, and the new FEP routines.
`terminate()`	Stops the science run, resets the FEPs, and writes a science report.

2.3. Inline FEP patches

Files *ctireport1fep2.S* and *ctireport2fep2.S* contain in-line assembler instructions that increase the size of D-cache stack used by `fepSciTimed()` and cause its code to branch to the routines described in §2.4. and §2.5. Their contents are described in these sections.

2.4. New FEP Routines for ctireport1

The *ctireport1* patch defines `FEPtestCti1()` and `FEPappendCti1()`. `FEPtestCti1` is called from `FEPsciTimedEvent`, in place of `FEPtestEvenPixel` and `FEPtestOddPixel`. Its function is as follows:

1. Determine whether a new exposure frame has begun and, if so, clear the D-cache arrays that contain precursor-charge information.

2. Call `FEPtestEvenPixel` or `FEPtestOddPixel`, according to whether the current pixel is located on an even- or odd-numbered row, respectively.
3. Save information about the current pixel in the D-cache arrays.

The new D-cache structure, whose address is stored in `fp->phist`, consists of an array of stacks, one per CCD column:

```
#define NSTK     24  /* maximum val[] entries per column */
#define NCOLS  1024  /* number of columns */

struct cti1stk {
  int expnum;        /* current exposure number */
  struct _stk {
    int count;       /* number of val[] entries in use */
    struct _pix {
      short row;     /* precursor charge row number */
      short adu;     /* precursor charge value */
    } val[NSTK];
  } stk[NCOLS];      /* one stack per column */
};
```

2.4.1 ctitreport1fep2.S

The code branches to the new routines by means of in-line patches described by the assembler instructions in `ctireport1fep2.S`. These make the following changes to the FEP code:

1. Increase the stack size on entry to the `fepSciTimed` routine by 102,404 bytes to accommodate a `cti1stk` structure (see below).
2. Decrease the stack size accordingly on exit from `fepSciTimed`.
3. Store a zero at the head of the `cti1stk` structure on entry to `FEPsciTimedEvent`, indicating that a new CCD frame is being processed.
4. Replace the code in `FEPsciTimedEvent` that branches to `FEPtestEvenPixel` for odd columns, or to `FEPtestOddPixel` for even columns, when a threshold crossing is detected, with a call to `FEPtestCti1`.
5. Within `FEPtestEvenPixel` and `FEPtestOddPixel`, replace the conditional branches to `FEPappend5x5` that were only taken in 5x5 mode with unconditional branches to `FEPappendCti1`.

2.4.2 FEPtestCti1

When a candidate event is detected by `FEPsciTimedEvent`, the patched code branches unconditionally to `FEPtestCti1`. It performs the following steps:

1. If `cp->expnum` is out-of-date, this is the first threshold crossing of a new exposure, so the `stk[].count` fields are set to zero:

```
struct _stk *cp = (struct cti1stk *)(fp->phist);
if (cp->expnum < fp->ex.expnum) {
    cp->expnum = fp->ex.expnum;
    for (istk = 0; istk < NCOLS; istk++)
        cp->stk[istk].count = 0;
}
```

2. The original `FEPtestOddPixel` and `FEPtestEvenPixel` routines are then called, depending on whether the threshold crossing occurs in an even or odd numbered column:

```
if (icol & 1) {
    FEPtestOddPixel(irow, icol, fp);
    pval = PIXEL1(*pImage) & PIXEL_MASK;
    bval = PIXEL1(*pBias) & PIXEL_MASK;
} else {
    FEPtestEvenPixel(irow, icol, fp);
    pval = PIXEL0(*pImage) & PIXEL_MASK;
    bval = PIXEL0(*pBias) & PIXEL_MASK;
}
```

3. If the bias value, `bval`, does not indicate a bad pixel or parity error, it is subtracted from the pixel value, `pval`, and adjusted for any change in average overclock. The result is stored in `pval`; the row index is in `irow` and the column index in `icol`.

```
pval -= bval + fp->ex.dOclk[icol >> fp->colshft];
if (bval >= BIAS_BAD || pval < 0)
    return;
```

4. The pixel value is rescaled to `PIXBITS` bits to fit into the 12-bit telemetry field along with the similarly rescaled `irow` value. However, `irow` is <u>not</u> rescaled at this point, for reasons stated in step 1 of §2.4.3.

```
#define PIXBITS      7
#define PIXVAL(adu)  (((adu) & 0xfff) >> (12-PIXBITS))
pval = PIXVAL(pval > 4095 ? 4095 : pval);
```

5. The following code is executed to remove items from the stack with `adu` values that do not exceed `pval`. If the scaled row index of the new top-of-stack entry is the same as the scaled row index of the new threshold crossing, we don't add a new entry to the stack, merely readjust the item count.

```
#define ROWBITS      5
#define ROWVAL(row)  (((row) & 0x3ff) >> (10-ROWBITS))
struct _stk *sp = &cp->stk[icol];
for (istk = sp->count; istk > 0; istk--) {
    if (sp->val[istk-1].adu > pval)
        if (ROWVAL(sp->val[istk-1].row - irow) == 0) {
            sp->count = istk;
            return;
        } else
            break;
}
```

6. We add the current threshold crossing to the stack. If the stack is full, we remove the top entry since it refers to a row that is the most distant from the current `irow`. This necessitates moving all remaining entries:

```
if (jj >= NSTK) {
    for (istk = 0; istk < NSTK-1; istk++)
        sp->val[istk] = sp->val[istk+1];
```

7. Finally, the current scaled, bias-subtracted pixel value, `pval`, and its (unscaled) row index, `irow`, are pushed into the stack and the counter adjusted:

```
sp->val[istk].adu = pval;
sp->val[istk].row = irow;
sp->count = istk+1;
```

2.4.3 FEPappendCti1

The *ctireport1* patch causes this information to be passed to the FEP's ring buffer by replacing calls to `FEPappend5x5` in `FEPtestOddPixel` and `FEPtestEvenPixel` with calls to `FEPappendCti1`. This function performs the following steps:

1. Inspects the three columns, `ncol=0,1,2`, that comprise the event candidate. If the `cti1stk` stack contains an entry for a column, the values of the three pixels in that column are extracted and `pval`, the maximum bias-subtracted pixel value, is computed. If `pval` exceeds the event threshold, the `cti1stk` stack is examined for the highest element that is not less that `pval`. Ideally, we wish to report all elements from here to the top of the stack. If no element exceeds `pval`, we want to report all elements. The stack index that identifies the first element to report, `istk`, is as follows:

   ```
   struct _stk *sp = &cp->stk[ev->col+ncol-1];
   int doclk = fp->ex.dOclk[(ev->col+ncol-1) >> fp->colshft];
   int thresh = fp->ex.dOclk[(ev->col+ncol-1) >> fp->colshft];
   int istk = sp->count;
   int nstk = sp->count;
   for (nrow = pval = 0; nrow < 3; nrow++) {
       int bval = ev->b[nrow][ncol];
       int ival = ev->p[nrow][ncol] - bval - doclk;
       if (bval < BIAS_BAD && ival > pval)
           pval = ival;
   }
   if (pval >= fp->tp.thresh[ev->col >> fp->colshft]) {
       pval = PIXVAL(pval);
       while (--istk >= 0)
           if (sp->val[istk].row+1 >= ev->row)
               nstk--;
           else if (sp->val[istk].adu >= pval)
               break;
       if (istk < 0)
           istk = 0;
   }
   ```

   Note the test of the stack element's row index, `sp->val[istk-1].row`, against the current row index, `ev->row`, to avoid reporting stack values belonging to pixels in the current event!

2. The value of `istk` for each of the 3 columns is saved in `istks[ncol]` and the number of stack elements above this in the stack, `nstk-istk`, is saved in `npix[ncol]`. The total number of stack elements that we want to report is saved in `ntotal`. If this exceeds the 15 available in the output array, `ep->pe[]`, we reduce the three `np` values as follows:

   ```
   while (ntotal > 15) {
       int dstk = (ntotal-13)/3;
       ntotal -= dstk;
       if (npix[0] > npix[1])
           if (npix[0] > npix[2])
               npix[0] -= dstk;
   ```

```
            else
                npix[2] -= dstk;
        else if (npix[1] > npix[2])
            npix[1] -= dstk;
        else
            npix[2] -= dstk;
    }
```

In words: lower the largest `np` value by no more than a third of the number required to reduce `ntotal`, the total, to 15, making very sure that the loop terminates! This has the effect of equalizing those `npix` values that were initially greater than 5, reporting those stack elements that had the larger bias-subtracted ADU values.

3. The remaining stack elements from the three columns are copied to the `ev->ep` array. The bias-subtracted pixel value has already been truncated to `PIXBITS`; the row value is now truncated to `ROWBITS` most significant bits. The maximum `STORE_PIX()` value is 12 bits long, *i.e.*, `PIXBITS+ROWBITS=12`.

```
#define STORE_PIX(val) ((((val).adu) << ROWBITS) | ROWVAL((val).row))
for (ncol = 0, ntotal = 0; ncol < 3; ncol++) {
    struct _stk *sp = &cp->stk[ev->col+ncol-1];
    istk = istks[ncol];
    for (nstk = 0; nstk < npix[nc]; nstk++, istk++)
        ev->pe[ntotal++] = STORE_PIX(sp->val[istk]);
}
```

4. Any unused elements in `ev->ep` are zero filled and the contents of the `npix[]` array are packed into `ev->ep[0]`:

```
ev->pe[0] = (npix[0] << 8) | (npix[1] << 4) | npix[2];
```

Thus, the 12 low-order bits of `pe[0]` contain three 4-bit counters, specifying the number of elements in `pe[1..15]` that are being used to report charge in, respectively, `icol-1`, `icol`, and `icol+1`.

2.5. New FEP Routines for ctireport2

The *ctireport2* patch defines `FEPtestCti2()` and `FEPappendCti2()`. `FEPtestCti2` is called from `FEPsciTimedEvent`, in place of `FEPtestEvenPixel` and `FEPtestOddPixel`. Their function is as follows:

1. Determine whether a new exposure frame has begun and, if so, clear the D-cache arrays that contain precursor-charge information.
2. Call `FEPtestEvenPixel` or `FEPtestOddPixel`, according to whether the current pixel is located on an even- or odd-numbered row, respectively.
3. Save information about the current pixel in the D-cache arrays.

The new D-cache structure, whose address is stored in `fp->phist`, consists of an array of integers, one per column:

```
struct cti2stk {
  int expnum;           /* current exposure number */
  unsigned stk[1024][2] /* precursor charge row indices */
};
```

2.5.1 ctireport2fep2.S

The code branches to the new routines by means of in-line patches described by the assembler instructions in `ctireport2fep2.S`. These make the following changes to the FEP code:

1. Increase the stack size on entry to the `fepSciTimed` routine by 8196 bytes to accommodate a `cti2stk` structure (see below).
2. Decrease the stack size accordingly on exit from `fepSciTimed`.
3. Store a zero at the head of the `cti1stk` structure on entry to `FEPsciTimedEvent`, indicating that a new CCD frame is being processed.
4. Replace the code in `FEPsciTimedEvent` that branches to `FEPtestEvenPixel` for odd columns, or to `FEPtestOddPixel` for even columns, when a threshold crossing is detected, with a call to `FEPtestCti2`.
5. Within `FEPtestEvenPixel` and `FEPtestOddPixel`, replace the conditional branches to `FEPappend5x5` that were only taken in 5x5 mode with unconditional branches to `FEPappendCti2`.

2.5.2 FEPtestCti2

When a candidate event is detected in either `FEPsciTimedEvent`, the patched code branches unconditionally to `FEPtestCti2`. This function performs the following steps:

1. If `cp->expnum` is out-of-date, this is the first threshold crossing of a new exposure, so all elements of the `stk` array are set to zero:

```
struct _stk *cp = (struct cti1stk *)(fp->phist);
if (cp->expnum < fp->ex.expnum) {
    cp->expnum = fp->ex.expnum;
    for (ncol = 0; ncol < 1024; ncol++)
        cp->stk[ncol][0] = cp->stk[ncol][1] = 0;
}
```

2. The original `FEPtestOddPixel` or `FEPtestEvenPixel` routines are called, depending on whether the threshold crossing occurs in an even or odd numbered column:

```
if (icol & 1)
    FEPtestOddPixel(irow, icol, fp);
else
    FEPtestEvenPixel(irow, icol, fp);
```

3. Finally, the row index is stored in the stack, keeping one row “ahead” so as to avoid updating a threshold crossing that was already recorded in the first row of the current event:

```
cp->stk[icol][1] = cp->stk[icol][0];
cp->stk[icol][0] = irow+1;
```

2.5.3 FEPappendCti2

When a candidate event is detected in either `FEPtestEvenPixel` or `FEPtestOddPixel`, the patched code branches unconditionally to `FEPappendCti2`. Information is extracted

from the D-cache arrays and inserted into the event block that is then written to the FEP-to-BEP ring buffer. This information consists of just 3 bits—`'1'` if the corresponding column contained precursor charge, or `'0'` if it didn't. These flags are stored in the least significant bit of three of the 4 corner pixels in the `FEPeventRec3x3.ev[3][3]` array. Since at least one of the corner pixel values is quite likely to contribute to the x-ray energy, the three 12-bit pixels are selected on the basis of having the lowest values of their 11 most significant bits. The code goes as follows:

1. The three columns represented in the event are inspected in the `cti2stk.stk` array. If they indicate precursor charge in that column, the fact is noted in the `chg[]` array.

```
struct cti2stk *cp = (struct cti2stk *)(fp->phist);
int chg[3];
for (ncol = 0; ncol < 3; ncol++) {
  int *pp = cp->stk[ev->col+ncol-1];
  chg[ncol] = (pp[0] && pp[0] < ev->row)||(pp[1] && pp[1] < ev->row);
}
```

2. The indices (`rrow,rcol`) of the maximum valued corner pixel is found:

```
for (nrow = rrow = rcol = 0; nrow < 3; nrow += 2) {
  for (ncol = 0; ncol < 3; ncol += 2) {
    if ((ev->p[nrow][ncol] & 0xffe) > (ev->p[rrow][rcol] & 0xffe)) {
      rrow = nrow;
      rcol = ncol;
    }
  }
}
```

3. The flag bits from `chg[]` are copied into the least significant bits of the three remaining corner pixels:

```
for (nrow = ichg = 0; nrow < 3; nrow += 2) {
  for (ncol = 0; ncol < 3; ncol += 2) {
    if (nrow != rrow || ncol != rcol) {
      ev->p[nrow][ncol] = (ev->p[nrow][ncol] & 0xffe) |
                          (chg[ichg++] & 1);
    }
  }
}
```

Note the necessity of masking the low-order bit in step 2. If not, it would not be possible in all instances to determine which three corner pixels contained the flags.

3. TELEMETRY IMPACT

3.1. ctiReport1

The downlinked exposure and event data packets are identical in format to *exposureTeFaint* and *dataTeVeryFaint* except that their `formatTag` fields contain `TTAG_SCI_TE_-REC_CTI1` and `TTAG_SCI_TE_DAT_CTI1`, respectively. When a `TTAG_SCI_TE_DAT_CTI1` is received, precursor charge data will be located in the `dataTeVeryFaint.pulseHeights` array, as follows:

```
pulseHeights[0]                          - three 4-bit counters
pulseHeights[1..5,9,10,14,15,19..24]     - precursor ADU and row
```

The sub-fields of `pulseHeights[0]` determine the contents of the remaining 15 fields:

```
ncol[0] = (pulseHeights[0] >> 8) & 15  - # items from icol-1
ncol[1] = (pulseHeights[0] >> 4) & 15  - # items from icol
ncol[2] = pulseHeights & 15            - # items from icol+1
```

The fields from `icol-1`, if any, are written starting at `pulseHeights[1]`, followed by those from `icol`, and finally those from `icol+1`. The ADU values are stored in the 7 most significant bits of `pulseHeights[]` and the row indices in the least significant 5 bits, and should be extracted as follows:

```
adu = pulseHeights[i] & 0xfe0;
row = (pulseheights[i] & 0x01f) << 5;
```

Unused `pulseHeights[]` will be filled with zeroes.

3.2. ctiReport2

The downlinked exposure and event data packets are identical in format to *exposureTeFaint* and *dataTeFaint*. To process the precursor charge information, ground software must first inspect the `loadTeBlock` reported in the *dumpedTeBlock* packet that started the run. If the `fepMode` field is equal to `FEP_TE_MODE_CTI2`, subsequent *dataTeFaint* packets should be inspected. The following code fills `ee[i]` with one (zero) according to whether column `(ccdColumn+i-1)` did (did not) contain precursor charge:

```
unsigned nn, mm, ii, ee[3];
for (mm = 0, nn = 2; nn < 9; nn++) {
    if ((nn & 1) == 0 && nn != 4) {
        if ((pulseHeights[nn] & 0xffe) > (pulseHeights[mm] & 0xffe))
            mm = nn;
    }
}
for (nn = ii = 0; nn < 9; nn++) {
    if ((nn & 1) == 0 && nn != 4 && nn != mm) {
        ee[ii++] = pulseHeights[nn] & 1;
    }
}
```

4. Controlled Sources

ctireport1	
ctireport1.mak	Generates object files.
ctireport1.pkg	Build file for creating an optional ctiReport1 patch.
ctireport1.C	BEP patch to control TE mode with precursor charge reporting.
ctireport1fep2.S	FEP inline patch to implement precursor charge reporting.
ctireport1fep1.c	new FEP routines to implement precursor charge reporting.
ctireport1/testsuite	
erv5.txt	Expected *psci* output from smoke tests.
makebias	Generate a bias image and copy it to the image loader.
makeimage	Generate an image with events and copy it to the image loader.
ctireport1/testsuite/smoke	

Makefile	Run tests of the SmTimedLookup+ctiReport1 patches.
runtest.tcl	Test SmTimedLookup+ctiReport1.
ctireport2	
ctireport2.mak	Generates object files.
ctireport2.pkg	Build file for creating an optional ctiReport2 patch.
ctireport2.C	BEP patch to control TE mode with precursor charge reporting.
ctireport2fep2.S	FEP inline patch to implement precursor charge reporting.
ctireport2fep1.c	new FEP routines to implement precursor charge reporting.
eco-1026.doc	Documentation of ctiReport1 and ctiReport2 patches.
ctireport2/testsuite	
erv.txt	Expected *psci* output from smoke tests.
makebias	Generate a bias image and copy it to the image loader.
makeimage	Generate an image with events and copy it to the image loader.
ctireport2/testsuite/smoke	
Makefile	Run tests of the SmTimedLookup+ctiReport2 patches.
runtest.tcl	Test SmTimedLookup+ctiReport2.

5. TESTING

The test is performed on the ACIS Engineering Unit using 6 FEPs, an image loader, and an L-RCTU interface. After setting up a *shim* process to handle I/O between UNIX and the L-RCTU, the test is controlled by a script written in the *expect* dialect of TCL.

5.1. ctireport1

An *expect* procedure, "*smoke/runtest.tcl*", performs a science run with the *"standard"*, *"smtimedlookup"*, and *"ctireport1"* patches. The following steps are performed:

1. A command pipe is spawned down which ACIS commands will be written.
2. A telemetry pipe is spawned, terminating in the "*psci -m -u*" packet monitoring function with *expect* examining the standard output.
3. ACIS is cold-booted.
4. Software housekeeping, DEA replacement, and standard flight patches are applied.
5. The *"smtimedlookup" and "ctireport1"* patch files are applied.
6. ACIS is warm-booted.
7. FEP_0 is powered up.
8. A bias map containing the same value in each pixel of a given quadrant is written to the image loader.
9. A *cti1* parameter block is sent to ACIS. It calls for FEP_0 to be run in cti1 mode.
10. A bias-only science run is started.

11. After the run completes, a test image is sent to the image loader containing 48 events with charge in columns 254-257, 510-513, and 766-769.
12. A science run in the new *cti1* mode is started.
13. The run is stopped after an exposure record is reported from FEP_0 with *exposureNumber* greater that 10. If no such record is found, the test fails.
14. The recorded telemetry stream is passed through *psci* with options *"–a –B –l foo"* and the events compared against "*ctireport1/testsuite/erv5.txt*". A match signals a pass; otherwise the test fails.

5.2. ctireport2

An *expect* procedure, "*smoke/runtest.tcl*", performs a science run with the *"standard"*, *"smtimedlookup"*, and *"ctireport2"* patches. The following steps are performed:

1. A command pipe is spawned down which ACIS commands will be written.
2. A telemetry pipe is spawned, terminating in the "*psci -m -u*" packet monitoring function with *expect* examining the standard output.
3. ACIS is cold-booted.
4. Software housekeeping, DEA replacement, and standard flight patches are applied.
5. The *"smtimedlookup" and "ctireport2"* patch files are applied.
6. ACIS is warm-booted.
7. FEP_0 is powered up.
8. A bias map containing the same value in each pixel of a given quadrant is written to the image loader.
9. A *cti2* parameter block is sent to ACIS. It calls for FEP_0 to be run in cti1 mode.
10. A bias-only science run is started.
11. After the run completes, a test image is sent to the image loader containing 48 events with charge in columns 254-257, 510-513, and 766-769.
12. A science run in the new *cti2* mode is started.
13. The run is stopped after an exposure record is reported from FEP_0 with *exposureNumber* greater that 10. If no such record is found, the test fails.
14. The recorded telemetry stream is passed through *psci* with options *"–a –B –l foo"* and the events compared against "*ctireport2/testsuite/erv.txt*". A match signals a pass; otherwise the test fails.

MIT CSR ACIS	**ENGINEERING CHANGE ORDER**	**ECO No. 36–1027**

CENTER FOR SPACE RESEARCH
MASSACHUSETTS INSTITUTE OF TECHNOLOGY

DWG. NO.	NEW REV.	DRAWING TITLE
58030.27	A	*compressall*: Patch to correct bug in raw-mode data compression

REASON FOR CHANGE:

The current raw-mode compression algorithm fails—and generates a zero-length packet—if a single compressed row of pixels cannot fit into a data packet. This was the subject of ACIS problem report M02040901 (SPR 134).

DESCRIPTION OF CHANGE:

(a) Create a new `Test_PmTeRaw` class method to override `PmTeRaw::digestRawRecord`;

(b) create a new `Test_PmCcRaw` class method to override `PmCcRaw::digestRawRecord`;

(c) in each method, first try to append the new row to an existing output packet; if this isn't possible, write the packet, get a new one and insert the new row; if this fails (and data compression is enabled), temporarily disable data compression, insert the uncompressed row into the packet, indicate in its header that the row isn't compressed, and write it out.

	SIGNATURE	DATE	REMARKS:
ORIGINATOR	Peter Ford	03/21/03	Final sign-off version
MECHANICAL			
ELECTRICAL			
SOFTWARE			
STRUCTURE			
FABRICATION			
SCIENCE			
SYSTEMS ENG.			
QUALITY			
PROJ. ENGINEER			
DEPUTY PM			
PROJ. MANAGER			

1. Reasons for Change

During OBSID 61267, the first test of ACIS PRAM and SRAM to measure charge injection, it was noticed that the raw-mode image rows that were expected to contain the most charge were appearing in telemetry as zero-length packets, *i.e.*, *dataTeRaw* packets whose `pixelCount` field value was zero. The cause was resolved by inspecting the flight software code. In the `PmTeRaw::digestRawRecord` method, if a compressed pixel row is too long to fit into a *dataTeRaw* packet, it is discarded and `pixelCount` is set to zero.

While this behavior is acceptable for bias maps, since an "incompressible" bias map would take an unacceptably long time to report in telemetry, it is undesirable in raw mode, where all pixel values must be presumed to be valuable.

2. Proposed Changes

2.1. Changes to the `PmTeRaw` Class

A `Test_PmTeRaw` class is defined. Its `digestRawRecord` method will replace the method of the same name in the `PmTeRaw` class.

```
class Test_PmTeRaw : public PmTeRaw
{
public:
  Test_PmTeRaw();
  ~Test_PmTeRaw();
  Boolean digestRawRecord(const FEPeventRecRaw*record);
  enum { NO_TE_COMPRESSION = 255 };

};
```

The constructor and destructor are defined merely to keep the MIPS loader happy. The replacement `digestRawRecord` is as follows. The new code is highlighted:

```
Boolean Test_PmTeRaw::digestRawRecord(const FEPeventRecRaw*record)
{
    DebugProbe probe;

    static PixelRow row;       // Static to prevent large stack use

    // extract the desired pixels from the current row
    EventExposure& exp = getExposureInfo ();
    row.setup (&exp);
    row.attachData (record);
    filterRow (row);

    if (setupDataPkt (row) == BoolFalse)
    {
        return BoolFalse;
    }

    unsigned pixels = 0;

    // try to append the pixels to the current output packet
    if (packRow(row, pixels) == BoolFalse)
    {
      // if unsuccessful, post the old packet
      rawForm.put_Pixel_Count(packetPixels);
      packetPixels = 0;
      rawForm.set_Data_Written(packer.getPackedWordCnt());
```

```
        rawForm.post();

        if (setupDataPkt(row) == BoolFalse)
        {
          return BoolFalse;
        }

        // try to insert the pixels into an empty output packet
        if (packRow(row, pixels) == BoolFalse)
        {
          // if even this won't work, save the compression code
          unsigned slotindex = getCompression();

          // if we aren't compressing, this is an error
          if (slotindex >= 32)
            {
              return BoolFalse;
            }

          // reset the output buffer and stop compressing
          setCompression(NO_TE_COMPRESSION);
          unsigned* packPtr = rawForm.get_Data_Address();
          unsigned  packLen = rawForm.get_Data_Avail();
          unsigned initial = 0;
          packer.setOutputBuffer(packPtr, packLen, initial);
          Boolean retval = BoolFalse;

          // pack the row without compression
          if (packRow(row, pixels) == BoolTrue)
            {
              // complete the header and post the packet
              rawForm.put_Compression_Table_Slot_Index
                 (NO_TE_COMPRESSION);
              rawForm.put_Compression_Table_Identifier (0);
              packetPixels += pixels;
              packPixels += pixels;
              rawForm.put_Pixel_Count(packetPixels);
              packetPixels = 0;
              rawForm.set_Data_Written(packer.getPackedWordCnt());
              rawForm.post();
              retval = BoolTrue;
            }

         // reload the original Huffman table and resume compression
          setCompression(slotindex);
          return retval;
        }
      }
      packetPixels += pixels;
      packPixels += pixels;

      return BoolTrue;
  }
```

2.2. Changes to the PmCcRaw Class

A `Test_PmCcRaw` class is defined. Its `digestRawRecord` method will replace the method of the same name in the `PmTeRaw` class.

```
class Test_PmCcRaw : public PmCcRaw
{
public:
  Test_PmCcRaw();
```

```
    ~Test_PmCcRaw();
    Boolean digestRawRecord(const FEPeventRecRaw*record);
    enum { NO_CC_COMPRESSION = 255 };
};
```

The constructor and destructor are defined merely to keep the MIPS loader happy. The replacement `digestRawRecord` is practically identical to that of the `Test_PmTeRaw` class, as shown in §2.1, above, and will not be reproduced here.

3. Telemetry Impact

Ground software must examine the *compressionTableSlotIndex* and *compressionTableIdentifier* fields of all *dataCcRaw* and *dataTeRaw* packets. If their values are 255 and 0, respectively, the pixel array should not be decompressed, *i.e.*, it should be treated as an array of 12-bit values.

4. Controlled Sources

compressall	
compressall.mak	Generates object files.
ctircompressall.pkg	Build file for creating an optional CompressAll patch.
compressall.C	BEP patch to correct bug in raw-mode data compression.
eco-1027.doc	Documentation for the CompressAll patch.
compressall/testsuite	
image.fits.gz	Raw CCD pixel image with overclocks and several very “noisy” rows.
DATA/.fits.gz*	Laboratory CCD frames with examples of “incompressible” rows.
makeimage	Generate an image with events and copy it to the image loader.
compressall/testsuite/bug-hw	
Makefile	Run a test without the CompressAll patch.
runtest.tcl	Test raw-mode compression without the CompressAll patch.
compressall/testsuite/fix-hw	
Makefile	Run a test with the CompressAll patch.
runtest.tcl	Test raw-mode compression with the CompressAll patch.
testimage.pl	PERL script to compare *DATA/*.fits.gz* files against *psci* output.

5. Testing

The test is performed on the ACIS Engineering Unit using one FEP, an image loader, and an L-RCTU interface. After setting up a *shim* process to handle I/O between UNIX and the L-RCTU, the test is controlled by a script written in the *expect* dialect of TCL.

5.1. bug-hw

An *expect* procedure, "*smoke/runtest.tcl*", performs a science run with the *"standard"* patch. The following steps are performed:

1. A command pipe is spawned down which ACIS commands will be written.
2. A telemetry pipe is spawned, terminating in the "*psci -m -u*" packet monitoring function with *expect* examining the standard output.
3. ACIS is cold-booted.
4. Software housekeeping, DEA replacement, and standard flight patches are applied.
5. ACIS is warm-booted.
6. FEP_1 is powered up.
7. An image is written to the image loader. The image was created from laboratory tests of "charge injection" and contains several rows with a high pixel variance that are known to fail to be compressed under the unpatched flight code.
8. A timed-exposure raw-mode parameter block is sent to ACIS.
9. A science run is started.
10. The run is terminated after the first raw frame with `pixelCount=0` has telemetered, indicating that the compression algorithm has failed. This indicates that the test has passed, since the *CompressAll* patch was not loaded.
11. If the entire frame has been telemetered without any packet containing `pixelCount=0`, or if no *exposureTeRaw* packet is ever received, the test has failed. Otherwise, it succeeds.

5.2. fix-hw

An *expect* procedure, "*smoke/runtest.tcl*", performs a science run with the *"standard"* and *"compressall"* patches. The following steps are performed:

1. A command pipe is spawned down which ACIS commands will be written.
2. A telemetry pipe is spawned, terminating in the "*psci -m -u*" packet monitoring function with *expect* examining the standard output.
3. ACIS is cold-booted.
4. Software housekeeping, DEA replacement, and standard flight patches are applied.
5. The *"compressall"* patch is loaded.
6. ACIS is warm-booted.
7. FEP_1 is powered up.
8. An image is written to the image loader. The image was created from laboratory tests of "charge injection" and contains several rows with a high pixel variance that are known to fail to be compressed under the unpatched flight code.
9. A timed-exposure raw-mode parameter block is sent to ACIS.
10. A science run is started.

11. The run is terminated after the first raw frame has been telemetered. If a packet is encountered with `pixelCount=0`, or if no *exposureTeRaw* packet is ever received, the test has failed.
12. Finally, the recorded telemetry stream is run through *psci* with the -C flag, and the *testimage.pl* script is invoked to compare the resulting raw image frame file against the data pixels from the input frames, "*../DATA/*.fits*". If they match, the test succeeds; otherwise, it fails.

MIT CSR ACIS	**ENGINEERING CHANGE ORDER**	**ECO No. 36–1028**

CENTER FOR SPACE RESEARCH
MASSACHUSETTS INSTITUTE OF TECHNOLOGY

DWG. NO.	NEW REV.	DRAWING TITLE
58030.28	A	*untricklebias*: Patch to run bias thief methods from science task

REASON FOR CHANGE:

The ACIS BEP has occasionally run the science and bias thief tasks simultaneously, a firmware lock-up condition in one or more FEPs. This is the subject of ACIS problem report M00062901 (SPR 133). The current patch causes the BEP to execute the bias thief methods from the science task, making it physically impossible for the BEP ever to run both functions simultaneously.

DESCRIPTION OF CHANGE:

(a) Create a new `Test_BiasThief` class with *"goTaskEntry"*, *"biasReady"*, *"abort"*, and *"checkMonitor"* methods to override those of the same name in the `BiasThief` class, and with a new *"doBiasTrickle"* method to perform the previous functions of *"goTaskEntry"*;

(b) create a new `Test_ScienceMode` class method to override `ScienceMode::waitForBiasTrickle` by calling `Test_BiasThief::doBiasTrickle` directly.

	SIGNATURE	DATE	REMARKS:
ORIGINATOR	Peter Ford	03/21/03	Final sign-off version
MECHANICAL			
ELECTRICAL			
SOFTWARE			
STRUCTURE			
FABRICATION			
SCIENCE			
SYSTEMS ENG.			
QUALITY			
PROJ. ENGINEER			
DEPUTY PM			
PROJ. MANAGER			

1. Reasons for Change

On three occasions since launch, the ACIS BEP has started to process science events while the bias thief task is still copying bias maps to telemetry. This has invariably led to a lock-up condition in one or more FEPs—the thresholding firmware has entered a lock-up state, from which it cannot return without external power cycling.

Command procedures have been established (a) to power cycle all FEPs as often as possible without reducing observing time, and (b) to restart any science run as soon as a latch-up has been detected in real-time telemetry.

The current patch goes further: by merging the bias thief methods into the science task, it makes it physically impossible for the BEP ever to run both functions simultaneously.

2. Proposed Changes

2.1. Changes to the ScienceMode Class

A `Test_ScienceMode` class is defined. Its `waitForBiasTrickle` method will replace the method of the same name in the `ScienceMode` class. Its sole function is to call the new `Test_BiasThief::doBiasTrickle` method, passing the address of the static `biasThief` object:

```
class Test_ScienceMode : public ScienceMode
{
public:
  Boolean waitForBiasTrickle();
};

Boolean Test_ScienceMode::waitForBiasTrickle()
{
    return Test_BiasThief::doBiasTrickle(biasThief);
}
```

2.2. Changes to the BiasThief Class

A `Test_BiasThief` class is defined:

```
class Test_BiasThief : public BiasThief
{
private:
  enum Test_Events { EV_SM_BIAS_ABORT_RUN = 1 << 1 };
public:
  Test_BiasThief(unsigned taskid);
  static Boolean doBiasTrickle(BiasThief& thief);
  virtual void goTaskEntry();
  virtual void biasReady();
  virtual void abort();
  virtual Boolean checkMonitor();
};
```

The constructor is defined merely to keep the MIPS loader happy. In the original code, the `goTaskEntry` method controlled all bias thief functions. In the patched version, it merely sits in an infinite loop, waiting to acknowledge “keep-alive” queries from the task monitor:

```
void Test_BiasThief::goTaskEntry()
{
  DebugProbe probe;
  for (;;)
    {
      unsigned caught = waitForEvent (EV_TASKQUERY);
      if (caught & EV_TASKQUERY)
        {
          taskMonitor.respond ();
        }
    }
}
```

The two bias thief methods that were called from the science task, `biasReady` and `abort`, must be replaced since they no longer need to send a `notify` signal to the task manager to wake up the bias thief task:

```
void Test_BiasThief::biasReady()
{
  abortFlag = BoolFalse;
  busyFlag = BoolTrue;
}

void Test_BiasThief::abort()
{
  abortFlag = BoolFalse;
}
```

The code that was previously executed in `goTaskEntry` is moved to `doBiasTrickle`:

```
Boolean Test_BiasThief::doBiasTrickle(BiasThief& thief)
{
  Boolean retval = BoolTrue;

  if (thief.isBusy()) {
    for (unsigned fepid = 0; fepid < FEP_COUNT; fepid++) {
      if (thief.fepInfo[fepid].base == 0) {
        continue;
      }

      if (thief.modetype == 0) {
        if (thief.trickleTeBias (FepId(fepid)) == BoolFalse) {
          retval = BoolFalse;
          break;
        }
      } else {
        if (thief.trickleCcBias (FepId(fepid)) == BoolFalse) {
          retval = BoolFalse;
          break;
        }
      }
    }
    thief.busyFlag = BoolFalse;
  }

  return retval;
}
```

The `trickleTeBias` and `trickleCcBias` methods do not need to be patched, but they make frequent calls to `checkMonitor`, which checks to see whether the task monitor has sent a message to the task (originally the bias thief, but now the science task). Since the bit

locations in the event masks differ within the two tasks, `checkMonitor` must be replaced:

```
Boolean Test_BiasThief::checkMonitor()
{
  Boolean retval = BoolTrue;
  unsigned mask = EV_TASKQUERY | EV_SM_BIAS_ABORT_RUN;
  unsigned caught=taskManager.queryCurrentTask()->requestEvent(mask);

  if (caught & EV_TASKQUERY) {
    taskMonitor.respond ();
  }

  if (caught & EV_SM_BIAS_ABORT_RUN) {
    retval = BoolFalse;
  }

  return retval;
}
```

Note the `EV_SM_BIAS_ABORT_RUN` mask value that is defined in `Test_BiasThief`. Although several of the other `BiasThief` methods address the task manager, they inherit the `Task` object of the current task, *i.e.*, the science task, and don't need to be altered. Similarly, the bias telemetry packets become *"owned"* by the science task that posted them.

3. Controlled Sources

untricklebias	
untricklebias.mak	Generates object files.
untricklebias.pkg	Build file for creating an optional UntrickleBias patch.
untricklebias.C	BEP patch to execute bias thief from science task.
eco-1028.doc	Documentation for the UntrickleBias patch.
untricklebias/testsuite	
makebias	Generate a 100-row bias image, and copy it to the image loader.
makeimage	Generate a 100-row image with 32 events, and copy it to the image loader.
untricklebias/testsuite/fix-hw	
Makefile	Run tests with the UntrickleBias patch.
runtest.tcl	Test the UntrickleBias patch in timed-exposure modes.
runtestcc.tcl	Test the UntrickleBias patch in continuous-clocking modes.
runtestall.tcl	Test bias trickling in timed-exposure mode with all release-C optional patches.

4. Testing

4.1. Philosophy

Since the BEP code involved in this patch is multi-threaded, it is important to test all possible scenarios. A science run comprises three processing phases: bias map creation, bias map copying to telemetry, and event processing. The scenarios interrupt this process with the receipt of one or two *stopScience* commands, or a *startScience* command, or a RADMON ENABLE signal. Furthermore, these events can occur at any of the three processing

phases. This leads us to conduct 12 tests each in timed-exposure and continuous-clocking modes.

The tests are performed on the ACIS Engineering Unit using six FEPs, an image loader, and an L-RCTU interface. After setting up a *shim* process to handle I/O between UNIX and the L-RCTU, the test is controlled by a script written in the *expect* dialect of TCL.

A single *expect* procedure, "*smoke/runtest.tcl*", performs all 12 timed-exposure mode tests, with the *"standard"* and *"untricklebias"* patches. A second procedure, "*smoke/runtestall.tcl*", executes the same tests after loading *all* the release C standard and optional patches. A third procedure, "*smoke/runtestcc.tcl*", performs the 12 continuous-clocking mode tests, with the *"standard"*, *"cc3x3"*, and *"untricklebias"* patches. The BEP is not rebooted between tests so as to increase the likelihood of detecting any non-reentrant behavior. The following steps are performed:

4.2. Setup

1. A command pipe is spawned down which ACIS commands will be written.
2. A telemetry pipe is spawned, terminating in the "*psci -m -u*" packet monitoring function with *expect* examining the standard output.
3. ACIS is cold-booted.
4. Software housekeeping, DEA replacement, and standard flight patches are applied.
5. The *"untricklebias"* patch is loaded.
6. ACIS is warm-booted.
7. All 6 FEPs are powered up.
8. 3x3 faint-mode timed-exposure and continuous-clocking parameter blocks are sent to ACIS. These parameter blocks all specify `recomputeBias=1` and `trickleBias=1`, *i.e.*, that bias maps are to be created and written to telemetry. To save testing time, the timed-exposure block specifies 100-row sub-arrays and 0.5 second exposures. The continuous-clocking block specifies 512 rows. Both blocks request that bias maps be computed and telemetered.
9. FEP input is taken from the image loader. Two loads are prepared—a 100-row (TE mode) or 512-row (CC-mode) bias image containing the same value in each pixel of a given quadrant, and a companion event image containing 32 events.

4.3. Test 1: stopScience command issued while creating bias maps

1. A bias map is written to the image loader.
2. A science run is started.
3. When the `SWSTAT_FEP_STARTBIAS` message is received via software housekeeping, a *stopScience* is commanded. The test fails if no `SWSTAT_FEP_STARTBIAS` is received.
4. The script waits for a *scienceReport* packet. If none is received, or if the `terminationCode` is not `SMTERM_STOPCMD` (1), or if event packets are found to be interleaved with bias packets, the test fails.

4.4. Test 2: two stopScience commands issued while creating bias maps

1. A bias map is written to the image loader.
2. A science run is started.
3. When the SWSTAT_FEP_STARTBIAS message is received via software housekeeping, a pair of *stopScience* commands are issued, separated by a 1-second delay. The test fails if no SWSTAT_FEP_STARTBIAS is received.
4. The script waits for a *scienceReport* packet. If none is received, or if its terminationCode is not SMTERM_STOPCMD (1), or if event packets are found to be interleaved with bias packets, the test fails.

4.5. Test 3: startScience command issued while creating bias maps

1. A bias map is written to the image loader.
2. A science run is started.
3. When the SWSTAT_FEP_STARTBIAS message is received via software housekeeping, a *startScience* command is issued. The test fails if no SWSTAT_FEP_STARTBIAS is received.
4. The script waits for a *scienceReport* packet. If none is received, or if the terminationCode is not SMTERM_FEP_DATA_START (6), the test fails.
5. In the CC-mode test, the script waits for the FEP software to be reloaded.
6. When the first *data*BiasMap* is received from the new run, an event image is written to the image loader. The test fails if an *exposure*Faint* or *scienceReport* packet is received first.
7. The run is terminated with a *stopScience* command after the first *exposure*Faint* packet is received. If no such packet is received, the test fails.
8. The script waits for a *scienceReport* packet. If none is received, or if its terminationCode is not SMTERM_STOPCMD (1), or if event packets are found to be interleaved with bias packets, the test fails.

4.6. Test 4: RADMON asserted and deasserted while creating bias maps

1. A bias map is written to the image loader.
2. A science run is started.
3. When the SWSTAT_FEP_STARTBIAS message is received via software housekeeping, a "RADMON ENABLE" command is sent to the BEP. The test fails if no SWSTAT_FEP_STARTBIAS is received.
4. The script waits for a SWSTAT_SCI_INHIBIT_ON message from software housekeeping, and then a *scienceReport* packet. Unless both are received and the terminationCode is SMTERM_RADMON (3), the test fails. In CC-mode, a terminationCode of SMTERM_FEP_DATA_START (6) is also acceptible.
5. A "RADMON DISABLE" command is sent to the BEP to restart the science run. The script waits for a SWSTAT_SCI_INHIBIT_OFF message from software housekeeping, and then a *data*BiasMap* packet. Unless both are received, the test fails.

6. An event image is sent to the image loader and the script waits for an *exposure*Faint* packet.
7. The script issues a *stopScience* command and waits for a *scienceReport* packet. If this isn't received, or if its `terminationCode` is not `SMTERM_STOPCMD` (1), or if event packets are found to be interleaved with bias packets, the test fails.

4.7. Test 5: stopScience command issued while copying bias maps

1. A bias map is written to the image loader.
2. A science run is started.
3. When the first *data*BiasMap* is received, a *stopScience* command is issued. The test fails if an *exposure*Faint* or *scienceReport* packet is received first.
4. The script waits for a *scienceReport* packet. If none is received, or if its `terminationCode` is not `SMTERM_STOPCMD` (1), or if event packets are found to be interleaved with bias packets, the test fails.

4.8. Test 6: two stopScience commands issued while copying bias maps

1. A bias map is written to the image loader.
2. A science run is started.
3. When the first *data*BiasMap* is received, a pair of *stopScience* commands are issued, separated by a 1-second delay. The test fails if an *exposure*Faint* or *scienceReport* packet is received first.
4. The script waits for a *scienceReport* packet. If none is received, or if its `terminationCode` is not `SMTERM_STOPCMD` (1), or if event packets are found to be interleaved with bias packets, the test fails.

4.9. Test 7: startScience command issued while copying bias maps

1. A bias map is written to the image loader.
2. A science run is started.
3. When the first *data*BiasMap* is received, a *startScience* command is issued. The test fails if an *exposure*Faint* or *scienceReport* packet is received first.
4. The script waits for a *scienceReport* packet. If none is received, or if the `terminationCode` is not `SMTERM_CLOBBERED` (4, in TE-mode) or `SMTERM_FEP_DATA_START` (6, in CC-mode), the test fails.
5. In the CC-mode test, the script waits for the FEP software to be reloaded.
6. When the first *data*BiasMap* is received from the new run, an event image is written to the image loader. The test fails if an *exposure*Faint* or *scienceReport* packet is received first.
7. The run is terminated with a *stopScience* command after the first *exposure*Faint* packet is received. If no such packet is received, the test fails.

8. The script waits for a *scienceReport* packet. If none is received, or if its `terminationCode` is not `SMTERM_STOPCMD` (1), or if event packets are found to be interleaved with bias packets, the test fails.

4.10. Test 8: RADMON asserted and deasserted while copying bias maps

1. A bias map is written to the image loader.
2. A science run is started.
3. When the first *data*BiasMap* is received, a "`RADMON ENABLE`" command is sent to the BEP. The test fails if no `SWSTAT_FEP_STARTBIAS` is received.
4. The script waits for a `SWSTAT_SCI_INHIBIT_ON` message from software housekeeping, and then a *scienceReport* packet. Unless both are received and the `terminationCode` is `SMTERM_RADMON` (3), the test fails. In CC-mode, a `terminationCode` of `SMTERM_FEP_DATA_START` (6) is also acceptible.
5. A "`RADMON DISABLE`" command is sent to the BEP to restart the science run. The script waits for a `SWSTAT_SCI_INHIBIT_OFF` message from software housekeeping. If this is not received, the test fails.
6. When the first *data*BiasMap* is received, an event image is written to the image loader. The test fails if an *exposure*Faint* or *scienceReport* packet is received first.
7. The run is terminated with a *stopScience* command after the first *exposure*Faint* packet is received. If no such packet is received, the test fails.
8. The script waits for a *scienceReport* packet. If none is received, or if its `terminationCode` is not `SMTERM_STOPCMD` (1), or if event packets are found to be interleaved with bias packets, the test fails.

4.11. Test 9: stopScience command issued during event processing

1. A bias map is written to the image loader.
2. A science run is started.
3. When the first *data*BiasMap* is received, an event image is written to the image loader. The test fails if an *exposure*Faint* or *scienceReport* packet is received first.
4. The run is terminated with a *stopScience* command after the first *exposureTeFaint* packet is received. If no such packet is received, the test fails.
5. The script waits for a *scienceReport* packet. If none is received, or if its `terminationCode` is not `SMTERM_STOPCMD` (1), or if event packets are found to be interleaved with bias packets, the test fails.

4.12. Test 10: two stopScience commands issued during event processing

1. A bias map is written to the image loader.
2. A science run is started.
3. When the first *data*BiasMap* is received, an event image is written to the image loader. The test fails if an *exposure*Faint* or *scienceReport* packet is received first.

4. After the first *exposure*Faint* packet is received, the run is terminated with a pair of *stopScience* commands, separated by a 1-second delay. If no *exposure*Faint* packet is received, the test fails.
5. The script waits for a *scienceReport* packet. If none is received, or if its `terminationCode` is not `SMTERM_STOPCMD` (1), or if event packets are found to be interleaved with bias packets, the test fails. In CC-mode, a `terminationCode` of `SMTERM_FEP_IO_ERROR` (15) is also acceptible.

4.13. Test 11: startScience command issued during event processing

1. A bias map is written to the image loader.
2. A science run is started.
3. When the first *data*BiasMap* is received, an event image is written to the image loader. The test fails if an *exposure*Faint* or *scienceReport* packet is received first.
4. After the first *exposure*Faint* packet is received, the run is terminated with a *startScience* command. If no *exposure*Faint* packet is received, the test fails.
5. The script waits for a *scienceReport* packet. If none is received, or if its `terminationCode` is not `SMTERM_FEP_IO_ERROR` (15), the test fails.
6. In the CC-mode test, the script waits for the FEP software to be reloaded.
7. When the first *data*BiasMap* is received from the new run, an event image is written to the image loader. The test fails if an *exposure*Faint* or *scienceReport* packet is received first.
8. The run is terminated with a *stopScience* command after the first *exposure*Faint* packet is received. If no such packet is received, the test fails.
9. The script waits for a *scienceReport* packet. If none is received, or if its `terminationCode` is not `SMTERM_STOPCMD` (1), or if event packets are found to be interleaved with bias packets, the test fails.

4.14. Test 12: RADMON asserted and deasserted during event processing

1. A bias map is written to the image loader.
2. A science run is started.
3. When the first *data*BiasMap* is received, an event image is written to the image loader. The test fails if an *exposure*Faint* or *scienceReport* packet is received first.
4. When the first *exposure*Faint* packet is received, a `"RADMON ENABLE"` command is sent to the BEP. The test fails if no `SWSTAT_FEP_STARTBIAS` is received.
5. The script waits for a `SWSTAT_SCI_INHIBIT_ON` message from software housekeeping, and then a *scienceReport* packet. Unless both are received and the `terminationCode` is `SMTERM_FEP_IO_ERROR` (15), the test fails.
6. A `"RADMON DISABLE"` command is sent to the BEP to restart the science run. The script waits for a `SWSTAT_SCI_INHIBIT_OFF` message from software housekeeping. If this is not received, the test fails.

7. When the first *data*BiasMap* is received, an event image is written to the image loader. The test fails if an *exposure*Faint* or *scienceReport* packet is received first.
8. The run is terminated with a *stopScience* command after the first *exposure*Faint* packet is received. If no such packet is received, the test fails.
9. The script waits for a *scienceReport* packet. If none is received, or if its `terminationCode` is not `SMTERM_STOPCMD` (1), or if event packets are found to be interleaved with bias packets, the test fails.

CSR	MASSACHUSETTS INSTITUTE OF TECHNOLOGY CENTER FOR SPACE RESEARCH CAMBRIDGE, MASSACHUSETTS 02139	
REVISION LOG	**TITLE:** **IP&CL Software IP&CL Structure Definition Notes**	**DOC. NO.** **36-53204.0204 Rev. N**

Revision	Date (mm/dd/yy)	ECO No.	Page(s) Affected	Reason	Approval
01	9/25/95	36-359	all	Initial version	
02	12/7/95	36-441	all	review corrections and update	
A	12/7/95	36-442	all	Fixed typos in preparation for external review. Incorporated comments on Notes from internal review.	RFG 1/19/96
B	04/08/96	36-557	all	Responded to SPRs and comments. Resolved some TBDs in the notes section. Responded to review comments.	RFG 4/8/96
C	06/17/96	36-623	all	Added "Ignore Initial Frames" parameters into Load Te/Cc commands.	RFG 6/18/96
D	07/18/96	36-699	all	Corrected some descriptions. "Ignore Initial Frames" only applies to Bias runs. Add initial value and pixel count fields to compressed data packets Add additional dump commands. Add Huffman/DEA/FEP load formats. Add DEA Query Ids Add Sys. Configuration Ids Update SW House codes Add some Fatal error codes.	RFG 7/23/96
E	09/06/96	36-736	all	Added some references to fields in IP&CL definitions. All interface enum definitions now provided in acis_h/interface.h. Added science run termination codes.	RFG 9/6/96

Revision	Date (mm/dd/yy)	ECO No.	Page(s) Affected	Reason	Approval
F	11/26/96	36-804	1,13, 14, 16-18, 20, 22, 23	Updated DEA settings and housekeeping enumerations to match proposed Rev. B version of the DEA-DPA ICD. Update software housekeeping and fatal error enumerations. Added new FEP error code. Added FEP interface #defines Removed initial bias value from raw mode data packets. Incorporated review comments that do not affect the existing Beta Flight Software Release.	RFG 11/26/96
G	1/7/97	36-834	1, 14, 16, 18, 25	Removed all pending items. Added missing entry for Output Gate in DEA Housekeeping list. Added additional Sof tware Housekeeping codes. Document response to a DEA query timeout or query to an unpowered board. In structures, renamed Continouous Clocking "Ccd Row" fields to "Transfer Row" to reflect DEA->FEP transfer coordinates, rather than CCD coordinates. Software housekeeping now reports a variable length array of statistic entries, reported since the previous housekeeping packet.	RFG 1/5/97
H	2/6/97	36-852	1, 20, 25, 27	Add medmean Timed Exposure bias algorithm parameters. Added Graded Mode corner mean special code definitions. Allowed "**bit**" type definition to represent a signed value, if the field's "*srange*" parameter is negative. In structures, modified range and description of Graded Event corner mean, mentioned limitations on exposure times less than 0.3 seconds, modified science timestamp descriptions.	RFG 2/6/97
I	2/7/97	36-859	1, 7, 23	Added Timed Exposure Faint 5x5 Mode. Fixed CC event amplitude descriptions.	RFG 2/13/97
J	4/4/97	36-889	1, 5, 7, 19, 26	Added support for clipping of system configuration inputs. Added 5x5-specific exposure record tag (format is the same). Fixed some descriptions. Listed now "unused" LED codes as spares. Reserved telemetry tag 45 to avoid confusion with fill-pattern.	RFG 4/12/97

Revision	Date (mm/dd/yy)	ECO No.	Page(s) Affected	Reason	Approval
K[a]	6/23/97	36-933	1, 16, 19,	Bakeout Enable clipped to "disabled" without a patch. Added SWSTATs for Jitter failure and bias trickle abort. In structures, updated integration time limitations.	RFG 4/18/97
L	11/16/98	36-983	1, 7, 24	Added new format for fix bias error telemetry (SPR-116). Added support for Event Histogram Mode. Added support for CC 3x3 Mode. Modified BEP Startup Msg to reflect actual implementation (see SPR-114)	RFG 11/16/98
M	04/21/99	36-1005	IP&CL Format Table only	IP&CL table changes only. Cleanup histogram exposure record exposureCount descriptions. Change Event Histogram Exposure Recored's Variance High field to contain SEU error correction count.	RFG 05/10/99
N	03/09/00	36-1021	5.16, 5.25	Added new codes to software housekeeping to report BEP event filtering from *reportgrade1* patch. Added section describing bogus parity errors in TE 5x5 mode.	RFG 03/21/03
	05023/00	36-1023	IP&CL Format Table only	IP&CL table changes only. Add *squeegeeRows*, *squeegeeIdle1*, *squeegeeIdle2*, and *squeegeeFlushCount* fields to *loadTeBlock* structure.	
	03/15/01		7, 25, & IP&CL Tables.	Add codes to command and report CTI1 and CTI2 modes.	
	03/21/03		5.16	Changed names of `SWSTAT_FILT_FEP_*_EVENT` fields generated by the *reportgrade1* patch	

a.THIS IS THE FLIGHT RELEASE VERSION. All subsequent versions are either clarifications or only apply to patches.

ACIS Software IP&CL Structure Definition Notes

MIT 36-53204.0204 Rev. N

1.0 Purpose

This document describes the conventions used in defining the ACIS Software IP&CL Structures definitions, and to provide an initial list of enumerated code values used within the structures.

This version of the IP&CL software structures are stored under *patches/ipcl* CVS Tag "release-N". To obtain a copy of the source (MIT only):

```
setenv CVSROOT /nfs/acis/h3/acisfs/configcntl
cvs export -r release-N patches/ipcl
```

2.0 References

TABLE 1. Reference Documents

Part Number	Version	Title
MIT 36-01103	G	ACIS Science Instrument Software Requirements Specification
MIT 36-02104	C	DPA Hardware Specification and System Description
MIT 36-02203	A	Focal Plane to Detector Housing Interface Control Document
MIT 36-02205	C	DPA to DEA Interface Control Document

3.0 Naming Conventions

In general, all record and field names contain one or more words, separated, by a single space, with the first letter in each word capitalized, and the remaining letters of each word in lower-case. The following abbreviations have been used in the naming of the record and field names:

1d .....- 1 Dimensional (1D)

2d .....- 2 Dimensional (2D)

Arg.....- Argument

Bep.....- Back End Processor (BEP)

Cc.....- Continuous Clocking Mode

Ccd.....- Charge Coupled Device (CCD)

Config .....- Configuration

Dea.....- Detector Electronics Assembly (DEA)

Fep.....- Front End Processor (FEP)

Id .....- Identifier

Oc.....- Overclock

Pram.....- DEA Program RAM (PRAM)

Sram.....- DEA Sequencer RAM (SRAM)

Sw .....- Software

Te .....- Timed Exposure Mode

WD.....- Watchdog Timer

4.0 Packet Sizes

All command packets contain between 3 and 256 16-bit words (inclusive). All telemetry packets will contain between 2 and 1023 32-bit words (inclusive).

5.0 Constants

This section describes constant codes used with the IP&CL structures. These codes are preliminary, and will be final upon release of the instrument software and hardware (some constants are defined by the DEA and DPA hardware design). These constants are listed below in the form of “C” programming enumerated types, or pre-processor definitions. Unless otherwise specified, the value of an enumerated symbol is the value of the preceding symbol plus 1. The first symbol in the enumerated list has a value of 0.

5.1 Command Opcodes

The following is the current list of command opcodes use by the instrument software. These values are used in the “Command Opcode” field of all command packets. These codes are provided in “acis_h/interface.h”

```
enum CmdOpcode
{
    CMDOP_UNUSED,             // Unused Opcode

    // ---- Load from Uplink Commands ----
    CMDOP_START_UPLOAD,       // Start Load From Uplink
    CMDOP_CONTINUE_UPLOAD,    // Continue Load From Uplink

    // ---- Memory Read/Write/Execute Commands ----
    CMDOP_READ_BEP,           // Read BEP Memory
   // BEP Write and Execute are now near the end of the table

    CMDOP_READ_FEP,           // Read FEP Memory
    CMDOP_WRITE_FEP,          // Write FEP Memory
```

```
 CMDOP_EXEC_FEP,           // Execute FEP Memory

 CMDOP_READ_PRAM,          // Read DEA PRAM
// Write PRAM is now near the end of the table

 CMDOP_READ_SRAM,          // Read DEA SRAM

 // ---- Load Parameter Block Commands ----
 CMDOP_LOAD_TE,            // Load Timed Exposure Block
 CMDOP_LOAD_CC,            // Load Continuous Clocking Block
 CMDOP_LOAD_2D,            // Load 2D Window List
 CMDOP_LOAD_1D,            // Load 1D Window List
 CMDOP_LOAD_DEA,           // Load DEA Housekeeping Block

 // ---- Start/Stop Run Commands ----
 CMDOP_START_TE,           // Start Timed Exposure Run
 CMDOP_BIAS_TE,            // Start Timed Exposure Bias Run

 CMDOP_START_CC,           // Start Continuous Clocking Run
 CMDOP_BIAS_CC,            // Start Continuous Clocking Bias Run

 CMDOP_START_DEA,          // Start DEA Housekeeping Run

 CMDOP_STOP_SCIENCE,       // Stop Science Run
 CMDOP_STOP_DEA,           // Stop DEA Housekeeping Run

 // ---- Patch Commands ----
 CMDOP_ADD_PATCH,          // Add Patches
 CMDOP_REMOVE_PATCH,       // Remove Patches

 // ---- System Configuration/Bad Pixel List Commands ----
 CMDOP_ADD_BAD_PIXEL,      // Add Bad Pixel
 CMDOP_RESET_BAD_PIXEL,    // Reset Bad Pixel List
 CMDOP_DUMP_BAD_PIXELS,    // Dump Bad Pixel List

 CMDOP_ADD_BAD_TE_COL,     // Add Bad Timed Exposure Column
 CMDOP_RESET_BAD_TE_COL,   // Reset Bad Timed Exposure Column List
 CMDOP_DUMP_BAD_TE_COL,    // Dump Bad Timed Exposure Column List

 CMDOP_ADD_BAD_CC_COL,     // Add Bad Cont. Clocking Column
 CMDOP_RESET_BAD_CC_COL,   // Reset Bad Cont. Clocking Column List
 CMDOP_DUMP_BAD_CC_COL,    // Dump Bad Cont. Clocking Column List

 CMDOP_CHANGE_SYS_ENTRY,   // Change System Configuration Settings
 CMDOP_DUMP_SYS_CONFIG,    // Dump System Configuration Settings

CMDOP_DUMP_PATCHLIST,      // Dump PatchList
CMDOP_DUMP_HUFFMAN,        // Dump Huffman Table
CMDOP_DUMP_TE_SLOTS,       // Dump Timed Exposure Block Slots
CMDOP_DUMP_CC_SLOTS,       // Dump Continuous Clocking Block Slots
CMDOP_DUMP_2D_SLOTS,       // Dump 2D Window Parameter Block Slots
CMDOP_DUMP_1D_SLOTS,       // Dump 1D Window Parameter Block Slots
CMDOP_DUMP_DEA_SLOTS,      // Dump DEA Housekeeping Block Slots

// ---- Two-bit difference codes ----
CMDOP_WRITE_BEP = 0xc0,    // Write BEP Memory
CMDOP_EXEC_BEP = 0xc3,     // Execute BEP Memory

CMDOP_WRITE_PRAM = 0xcc,   // Write DEA PRAM
```

```
    CMDOP_WRITE_SRAM = 0xf0,   // Write DEA SRAM

    // ---- Miscellaneous Codes ----
    CMDOP_COUNT,               // Total # of potential Command Opcodes
    CMDOP_LASTID = CMDOP_COUNT - 1, // Value of last opcode

    CMDOP_INVALID = 0xffff     // Code for an invalid opcode
};
```

[see Command Header: Command Opcode]

5.2 Command Execution Result Codes

The following is the current list of command result use by the instrument software when echoing the command to telemetry. These values are used in the “Result” field of “Command Echo” telemetry packets. These codes are provided in “acis_h/interface.h”

```
enum CmdResult
{
   CMDRESULT_UNUSED,          // Unused response code

   CMDRESULT_OK,              // Command Successfully Dispatched
   CMDRESULT_NO_HANDLER,      // No handler for command opcode
   CMDRESULT_BUSY,            // Target of command is busy
   CMDRESULT_BAD_ARGUMENT,    // Command contains a bad parameter
   CMDRESULT_CORRUPT_DEFAULT, // Parameter Blk corrupt, use default
   CMDRESULT_CORRUPT_IDLE,    // Parameter Blk/Default corrupt, no run
   CMDRESULT_TABLE_FULL,      // Pixel/Column/Patch Table Full
   CMDRESULT_TABLE_EMPTY,     // Patch Table Empty
   CMDRESULT_INVALID_PKT,     // Command Packet Header corrupt
   CMDRESULT_BOARD_OFF,       // Selected Board has no power
   CMDRESULT_BOARD_RESET,     // Select Board is Reset
   CMDRESULT_STORE_ERROR,     // Error while storing parameter block

   CMDRESULT_INHIBITED,       // Run was inhibited prior to stop/start
   CMDRESULT_CLOBBERED,       // Operation aborted by new operation

   CMDRESULT_ITEM_CLIPPED,    // Input value was clipped to limit

   CMDRESULT_COUNT,           // Total number of result codes
   CMDRESULT_LASTID = CMDRESULT_COUNT - 1
};
```

[see Command Echo: Result]

5.3 Telemetry Format Tags

The following is the current list of telemetry format tags. These appear in the “Format Tag” field of the header portion of all telemetry packets. These codes are provided in “acis_h/interface.h”

```
enum TlmFormatTag
{
    TTAG_UNUSED,              // Unused format tag

    // ---- Memory Command Responses ----
    TTAG_READ_BEP,            // Read Back End Memory
    TTAG_READ_FEP,            // Read Front End Memory
    TTAG_READ_SRAM,           // Read SRAM
    TTAG_READ_PRAM,           // Read Pram
    TTAG_EXEC_BEP,            // Execute Back End Subroutine
    TTAG_EXEC_FEP,            // Execute Front End Subroutine

    // ---- Command Echoes ----
    TTAG_CMD_ECHO,            // Echoed command

    // ---- Miscellaneous Housekeeping ----
    TTAG_STARTUP,             // Startup Message
    TTAG_FATAL,               // Fatal Error Message
    TTAG_SW_HOUSE,            // Software Housekeeping
    TTAG_DEA_HOUSE,           // DEA Housekeeping

    // ---- Parameter Dumps ----
    TTAG_DUMP_TE,             // Timed Exposure Parameters + 2D Window
    TTAG_DUMP_CC,             // Cont. Clocking Parameters + 1D Windows

    // ---- Miscellaneous Science ----
    TTAG_SCI_TE_BIAS,         // Timed Exposure Bias Map
    TTAG_SCI_REPORT,          // Science Run Report

    // ---- Timed Exposure Science ----
    TTAG_SCI_TE_REC_RAW,      // Timed Exposure Raw Mode Exposure Hdr
    TTAG_SCI_TE_DAT_RAW,      // Timed Exposure Raw Mode Data
    TTAG_SCI_TE_REC_HIST,     // Timed Exposure Histogram Mode Exp. Hdr
    TTAG_SCI_TE_DAT_HIST,     // Timed Exposure Histogram Mode Data
    TTAG_SCI_TE_REC_FAINT,    // Timed Exposure Faint Mode Exp. Hdr
    TTAG_SCI_TE_DAT_FAINT,    // Timed Exposure Faint Mode Data
    TTAG_SCI_TE_REC_FAINTB,   // Timed Exposure Faint-Bias Mode Exp. Hdr
    TTAG_SCI_TE_DAT_FAINTB,   // Timed Exposure Faint-Bias Mode Data
    TTAG_SCI_TE_REC_GRADED,   // Timed Exposure Graded Mode Exp. Hdr
    TTAG_SCI_TE_DAT_GRADED,   // Timed Exposure Graded Mode Data

    // ---- Continuous Clocking Science ----
    TTAG_SCI_CC_REC_RAW,      // Continuous Clk Raw Mode Exposure Hdr
    TTAG_SCI_CC_DAT_RAW,      // Continuous Clk Raw Mode Data
    TTAG_SCI_CC_REC_FAINT,    // Continuous Clk Faint Mode Exp. Hdr
    TTAG_SCI_CC_DAT_FAINT,    // Continuous Clk Faint Mode Data
    TTAG_SCI_CC_REC_GRADED,   // Continuous Clk Graded Mode Exp. Hdr
    TTAG_SCI_CC_DAT_GRADED,   // Continuous Clk Graded Mode Data

    TTAG_SCI_CC_BIAS,         // Continuous Clocking Bias Map
    TTAG_SCI_BIAS_ERROR,      // Science Bias Error Data Packet
```

```
    TTAG_DUMP_SYS_CONFIG,     // Dump of System Configuration Table
    TTAG_DUMP_BAD_PIXEL,      // Dump of Bad Pixel Map
    TTAG_DUMP_BAD_TE_COL,     // Dump of Bad Timed Exposure Columns
    TTAG_DUMP_BAD_CC_COL,     // Dump of Bad Cont. Clocking Columns
    TTAG_DUMP_PATCHES,        // Dump of Patch List
    TTAG_DUMP_HUFFMAN,        /* Dump of Huffman Tables */
    TTAG_DUMP_TE_SLOTS,       /* Dump of Timed Exposure Slots */
    TTAG_DUMP_CC_SLOTS,       /* Dump of Cont. Clocking Slots */
    TTAG_DUMP_2D_SLOTS,       /* Dump of 2D Window Slots */
    TTAG_DUMP_1D_SLOTS,       /* Dump of 1D Window Slots */
    TTAG_DUMP_DEA_SLOTS,      /* Dump of DEA Housekeeping Slots */

    TTAG_FILL_PATTERN = 45,   /* Tag received if hw fill pattern used */

    TTAG_SCI_TE_DAT_FAINT_5x5 /* Timed Exposure Faint 5x5 Data */
    TTAG_SCI_TE_REC_FAINT_5x5,/* Timed Exposure Faint 5x5 Exp. Hdr */

    /* ---- NEW PATCHED TAGS ---- */
    TTAG_SCI_TE_DAT_EV_HIST,    /* 3x3 Event Histogram Data */
    TTAG_SCI_TE_REC_EV_HIST,    /* 3x3 Event Histogram Exposure Record */

    TTAG_SCI_PATCHED_BIAS_ERROR = 50, /* Patched Bias Error Packet */

    TTAG_SCI_CC_DAT_FAINT3x3,    /* Continuous Clocking 3x3 Mode Data */
    TTAG_SCI_CC_REC_FAINT3x3,    /* Continuous Clocking 3x3 Mode Record */

    TTAG_SCI_CC_DAT_GRADED3x3,   /* Continuous Clock Graded 3x3 Mode Data */
   TTAG_SCI_CC_REC_GRADED3x3,   /* Continuous Clock Graded 3x3 Mode Record */

   TTAG_SCI_TE_DAT_CTI1,        /* Timed Exposure with CTI correction Data */
    TTAG_SCI_TE_REC_CTI1,        /* Timed Exposure with CTI correction Hdr */

    // ---- Miscellaneous Codes ----
    TTAG_RESERVED = 0x3f,     /* Reserved for Maintenance Use */

    TTAG_COUNT,               /* Number of Format Tags */
    TTAG_LASTID = TTAG_COUNT-1
};
```

[see Telemetry Header: Format Tag]

5.4 Software Bi-level Discrete Telemetry (LED) Codes

The following is the current list of software bi-level discrete telemetry codes (LED codes) (NOTE: The remaining bi-levels used by the BEP hardware are not described in this section). These values appear within the bi-level discrete telemetry in a TBD location (by TRW) of the engineering portion of the telemetry frames. These codes are provided in “acis_h/interface.h”

```
enum LedState
{
 LED_WD_SCIENCE_A,         // WD Science Active - Blink State A
 LED_WD_SCIENCE_B,         // WD Science Active - Blink State B
 LED_WD_IDLE_A,            // WD Idle - Blink State A
 LED_WD_IDLE_B,            // WD Idle - Blink State B
 LED_RUN_SCIENCE_A,        // Science Active - Blink State A
```

```
 LED_RUN_SCIENCE_B,        // Science Active - Blink State B
 LED_RUN_IDLE_A,           // Idle - Blink State A
 LED_RUN_IDLE_B,           // Idle - Blink State B
 LED_RUN_STARTUP,          // Initializing loaded code
 LED_RUN_PATCH,            // About to patch loaded code
 LED_BOOT_UPLINK_EXECUTE,  // About to execute loaded code
 LED_BOOT_UPLINK_COPY,     // Copying packets from uplink FIFO
 LED_BOOT_UPLINK_WAIT,     // Waiting for initial load from uplink pkt.
 LED_BOOT_SPARE1,          // Spare (was About to execute copied code)
 LED_BOOT_SPARE2,          // Spare (was Copying ROM into Bep RAM)
 LED_BOOT_RESET            // Bep was just reset
};
```

5.5 CCD Identifiers

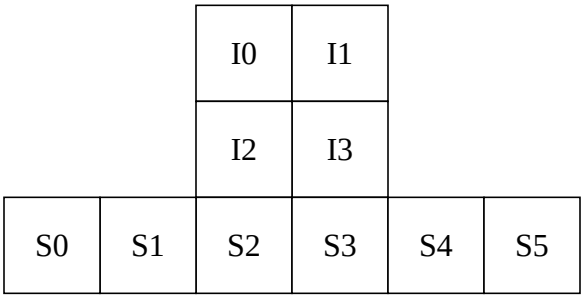
The following values are used to identify a CCD. These codes are provided in “acis_h/interface.h”

```
enum CcdId
{
    CCD_I0,         // Imaging CCD I0
    CCD_I1,         // Imaging CCD I1
    CCD_I2,         // Imaging CCD I2
    CCD_I3,         // Imaging CCD I3

    CCD_S0,         // Spectroscopy CCD S0
    CCD_S1,         // Spectroscopy CCD S1
    CCD_S2,         // Spectroscopy CCD S2
    CCD_S3,         // Spectroscopy CCD S3
    CCD_S4,         // Spectroscopy CCD S4
    CCD_S5,         // Spectroscopy CCD S5

   CCD_DESELECT,    // Code used to indicate no CCD selection
   CCD_COUNT = CCD_DESELECT,// Number of selectable CCD Codes
   CCD_LASTID = CCD_COUNT - 1
};
```

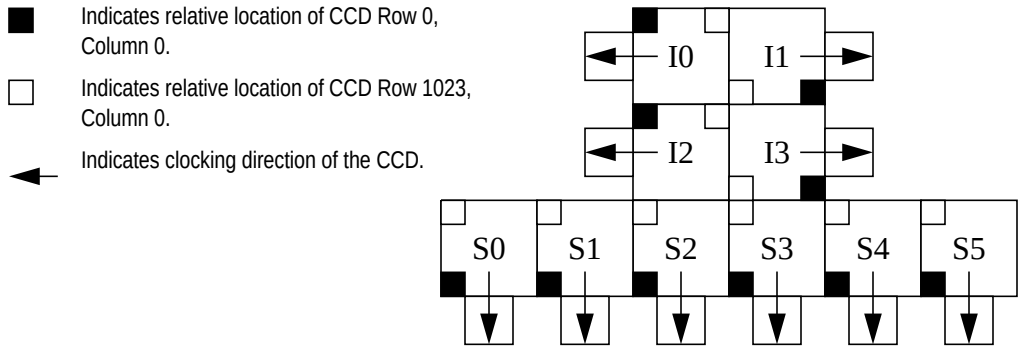
For reference purposes only, the following diagram illustrates the relative layout of the indicated CCDs in the focal plane assembly, as viewed from the High Resolution Mirror Assembly (HRMA). This information was obtained from the “Focal Plane to Detector Housing Interface Control Document”, MIT 36-02203, Rev. A.



5.6 CCD Row and Column Position Definition

The following illustration defines the CCD Row and Column positions used by the ACIS Software. Row 0 is defined to be the CCD imaging row closest to the Framestore. Column

0 is defined to be the left-most imaging column (i.e. the first to be clocked out of output node A).



5.7 Event Grade Code Definition

The following illustration defines the CCD Grade Code bit-definitions for a 3x3 event, where the row and column positions are indicated relative to the reported center of the event.

3x3 Event

row				
	1	5	6	7
	0	3		4
	-1	0	1	2
		-1	0	1

column

Grade Code Bits

msb							lsb
7	6	5	4	3	2	1	0

The following illustration defines the CCD Grade Code bit-definitions for a 1x3 event, where the column positions are indicated relative to the reported center of the event.

5.8 Huffman Compression Table Format

The on-board Huffman table array consists of a 32-word index table followed by a set of Huffman compression tables. Each word in the table is indexed by table slot id, and indicates the offset of the corresponding Huffman Compression table after the index table (i.e. a value of 0 indicates that the Huffman table immediately follows the index table, a value of 200 indicates that the table is 200 32-bit words after the index table). If an index table word is 0xffffffff, there is no Huffman table referenced by corresponding slot.

The format of each Huffman table is as follows:

32-bit Words	Item	
1	Table Identifier	
1	Low Limit	
1	Table Size	
1	Truncation Code	
1	Bad Bias Code	
1	Bad Pixel Code	
Table Size	Huffman Code Entries	
	The Truncation Code, Bad Bias Code, Bad Pixel Code and Huffman Code entries have the following format:	
	Bits	**Item**
	5	Code Bit Length
	32 - (Code Bit Length + 5)	Pad
	Code Bit Length	Huffman Code

5.9 DEA PRAM/SRAM Load Format

The DEA PRAM/SRAM load format is as follows:

16-bit Words	Item	
1	Sequence Type (always 0)	
1	Section Count	
Variable	Sections	
	where the format of each section is as follows:	
	16-bit Words	**Item**
	1	CCD Controller Select
		where:
		bit 0 selects CCD I0 (0 - no load, 1 - load)
		bit 1 selects CCD I1
		bit 2 selects CCD I2
		bit 3 selects CCD I3
		bit 4 selects CCD S0
		bit 5 selects CCD S1
		bit 6 selects CCD S2
		bit 7 selects CCD S3
		bit 8 selects CCD S4
		bit 9 selects CCD S5
		If bit 15 is 0, load the data into SRAM
		If bit 15 is 1, load the data into PRAM
	1	Index into SRAM or PRAM
	1	Word Count
	Word Count	Data to Load into SRAM or PRAM

5.10 FEP Load Format

The Front End Processor load format is as follows:

16-bit Words	Item
1	Program Type (always 0)
1	Section Count
1	Execute Address (low order 16-bits)
	Execute Address (high order 16-bits)
1	Command Mailbox Address (low order 16-bits)
1	Command Mailbox Address (high order 16-bits)
1	Ring Buffer Address (low order 16-bits)
1	Ring Buffer Address (high order 16-bits)

16-bit Words	Item
Variable	Sections

where the format of each section is as follows:

16-bit Words	Item
1	Section Type (always 0)
1	Load Address (low order 16-bits)
1	Load Address (high order 16-bits)
1	Number of 16-bit words to load (MUST BE EVEN)
Word Count	Data to Load

5.11 FEP Identifiers

The following are used to identify specific Front End Processors. These codes are provided in “acis_h/interface.h”

```
enum FepId
{
    FEP_0,          // FEP Slot 0 Identifier
    FEP_1,          // FEP Slot 1 Identifier
    FEP_2,          // FEP Slot 2 Identifier
    FEP_3,          // FEP Slot 3 Identifier
    FEP_4,          // FEP Slot 4 Identifier
    FEP_5,          // FEP Slot 5 Identifier

   FEP_COUNT,       // Number of FEP Ids
   FEP_LASTID = FEP_COUNT - 1
};
```

5.12 Video Chain Identifiers

The following codes are used to identify a video chain from a single CCD. These codes are provided in “acis_h/interface.h”

```
enum QuadId
{
    ONODE_A,        // Output Node/Video Chain A
    ONODE_B,        // Output Node/Video Chain B
    ONODE_C,        // Output Node/Video Chain C
    ONODE_D         // Output Node/Video Chain D

   ONODE_COUNT,     // Number of Output Node Ids
   ONODE_LASTID = ONODE_COUNT - 1
};
```

5.13 Output Register Clocking Modes

The following codes are used to indicate how to clock a CCD’s output register. These codes are provided in “acis_h/interface.h”

```
enum QuadMode
{
   QUAD_FULL,       // Clock charge to all four output nodes
   QUAD_DIAG,       // Clock charge away from all four output nodes
   QUAD_AC,         // Clock charge toward output nodes A and C
   QUAD_BD,         // Clock charge toward output nodes B and D

   QUAD_COUNT,      // Number of Quadrant Clocking Modes
   QUAD_LASTID = QUAD_COUNT - 1
};
```

[see Load Cc Block: Output Register Mode, Load Te Block: Output Register Mode]

5.14 DEA Query Identifiers

The following lists the currently defined DEA Housekeeping Query identifiers. These codes are provided in “acis_h/interface.h”.

```
enum DeaQueryCntlId
{
  /* ---- Interface Board Queries ---- */
  DEAHOUSE_CNTL_BASE = 0,         /* Base Query for controller board */
                                  /* Relay Positions */
  DEAHOUSE_CNTL_RELAY=DEAHOUSE_CNTL_BASE,

  DEAHOUSE_CNTL_ADC_BASE,         /* ADC Base Query */
                                   /* DPA Thermistor 1 - BEP PC Board */
  DEAHOUSE_CNTL_ADC_TMP_BEP_PCB=DEAHOUSE_CNTL_ADC_BASE,
  DEAHOUSE_CNTL_ADC_TMP_BEP_OSC,    /* DPA Thermistor 2 - BEP Oscillator */
  DEAHOUSE_CNTL_ADC_TMP_FEP0_MONG,  /* DPA Thermistor 3 - FEP 0 Mongoose */
  DEAHOUSE_CNTL_ADC_TMP_FEP0_PCB,   /* DPA Thermistor 4 - FEP 0 PC Board */
  DEAHOUSE_CNTL_ADC_TMP_FEP0_ACTEL, /* DPA Thermistor 5 - FEP 0 ACTEL */
  DEAHOUSE_CNTL_ADC_TMP_FEP0_RAM,   /* DPA Thermistor 6 - FEP 0 RAM */
  DEAHOUSE_CNTL_ADC_TMP_FEP0_FB,    /* DPA Thermistor 7 - FEP 0 Frame Buf. */
  DEAHOUSE_CNTL_ADC_TMP_FEP1_MONG,  /* DPA Thermistor 8 - FEP 1 Mongoose */
  DEAHOUSE_CNTL_ADC_TMP_FEP1_PCB,   /* DPA Thermistor 9 - FEP 1 PC Board */
  DEAHOUSE_CNTL_ADC_TMP_FEP1_ACTEL, /* DPA Thermistor 10- FEP 1 ACTEL */
  DEAHOUSE_CNTL_ADC_TMP_FEP1_RAM,   /* DPA Thermistor 11- FEP 1 RAM */
  DEAHOUSE_CNTL_ADC_TMP_FEP1_FB,    /* DPA Thermistor 12- FEP 1 Frame Buf. */
  DEAHOUSE_CNTL_ADC_SUBAHK,      /* DEA Video Board ADC */
  DEAHOUSE_CNTL_ADC_SPARE1,      /* Spare - Unused */
  DEAHOUSE_CNTL_ADC_FPTEMP_12,      /* Spare - Focal Plane Temp. Board 12 */
  DEAHOUSE_CNTL_ADC_FPTEMP_11,      /* Spare - Focal Plane Temp. Board 11 */
  DEAHOUSE_CNTL_ADC_DPAGNDREF1,     /* DPA Ground Reference 1 */
  DEAHOUSE_CNTL_ADC_DPA5VHKA,     /* DPA 5V Housekeeping A */
  DEAHOUSE_CNTL_ADC_DPAGNDREF2,     /* DPA Ground Reference 2 */
  DEAHOUSE_CNTL_ADC_DPA5VHKB,     /* DPA 5V Housekeeping B */
  DEAHOUSE_CNTL_ADC_UNUSED1,      /* Unused */
  DEAHOUSE_CNTL_ADC_UNUSED2,      /* Unused */
  DEAHOUSE_CNTL_ADC_UNUSED3,      /* Unused */
  DEAHOUSE_CNTL_ADC_UNUSED4,      /* Unused */
  DEAHOUSE_CNTL_ADC_DEA28VDCA,      /* PSMC A DEA 28V DC */
  DEAHOUSE_CNTL_ADC_DEA24VDCA,      /* PSMC A DEA 24V DC */
  DEAHOUSE_CNTL_ADC_DEAM15VDCA,     /* PSMC A DEA -15.5V */
  DEAHOUSE_CNTL_ADC_DEAP15VDCA,     /* PSMC A DEA +15.5V */
  DEAHOUSE_CNTL_ADC_DEAM6VDCA,      /* PSMC A DEA -6V DC */
  DEAHOUSE_CNTL_ADC_DEAP6VDCA,      /* PSMC A DEA +6V DC */
  DEAHOUSE_CNTL_ADC_RAD_PCB_A,      /* Relative Dose Rad. Monitor Side A */
  DEAHOUSE_CNTL_ADC_GND_1,       /* Interface Ground Reference */
  DEAHOUSE_CNTL_ADC_DEA28VDCB,      /* PSMC B DEA 28V DC */
  DEAHOUSE_CNTL_ADC_DEA24VDCB,      /* PSMC B DEA 24V DC */
  DEAHOUSE_CNTL_ADC_DEAM15VDCB,     /* PSMC B DEA -15.5V DC */
  DEAHOUSE_CNTL_ADC_DEAP15VDCB,     /* PSMC B DEA +15.5V DC */
  DEAHOUSE_CNTL_ADC_DEAM6VDCB,      /* PSMC B DEA -6V DC */
  DEAHOUSE_CNTL_ADC_DEAP6VDCB,      /* PSMC B DEA +6V DC */
  DEAHOUSE_CNTL_ADC_RAD_PCB_B,      /* Relavtive Dose Rad. Monitor Side B */
  DEAHOUSE_CNTL_ADC_GND_2,       /* Ground */

  DEAHOUSE_CNTL_ADC_END = DEAHOUSE_CNTL_ADC_GND_2,
  DEAHOUSE_CNTL_END = DEAHOUSE_CNTL_ADC_END,
};
```

```
enum DeaQueryCcdId
{
    // ---- CCD Controller Queries ----
    DEAHOUSE_CCD_BASE = 0,       // Base Query for CCD Queries
                               // Register 0 Sequencer Control
    DEAHOUSE_CCD_REG_0 = DEAHOUSE_CCD_BASE,
    DEAHOUSE_CCD_REG_1,        // Register 1 Video ADC Control
    DEAHOUSE_CCD_REG_2,        // Register 2
    DEAHOUSE_CCD_REG_3,        // Register 3

    DEAHOUSE_CCD_ADC_BASE = 0x80, // Base Index for ADC Registers
                               // Image Array Parallel +
    DEAHOUSE_CCD_PIA_P = DEAHOUSE_CCD_ADC_BASE,
    DEAHOUSE_CCD_PIA_M,        // Image Array Parallel -
    DEAHOUSE_CCD_PFS_P,        // Framestore Parallel +
    DEAHOUSE_CCD_PFS_M,        // Framestore Parallel -
    DEAHOUSE_CCD_S_P,          // Serial Register +
    DEAHOUSE_CCD_S_M,          // Serial Register -
    DEAHOUSE_CCD_R_P,          // Reset Gate +
    DEAHOUSE_CCD_R_M,          // Reset Gate -
    DEAHOUSE_CCD_OG,          // Output Gate Bias Level
    DEAHOUSE_CCD_SCP,          // Scupper
    DEAHOUSE_CCD_RD,           // Reset Diode
    DEAHOUSE_CCD_DR0,          // Drain Output Channel A
    DEAHOUSE_CCD_DR1,          // Drain Output Channel B
    DEAHOUSE_CCD_DR2,          // Drain Output Channel C
    DEAHOUSE_CCD_DR3,          // Drain Output Channel D
    DEAHOUSE_CCD_SPARE,           // Spare Housekeeping Channel
    DEAHOUSE_CCD_TEMP_BOARD,    // Board Temperature (RTD4)
    DEAHOUSE_CCD_TEMP_SRAM,     // SRAM Temperature (RTD3)
    DEAHOUSE_CCD_TEMP_ADC,      // ADC Temperature (RTD2)
    DEAHOUSE_CCD_TEMP_ACTEL,    // Gate Array Temperature (RTD1)

    DEAHOUSE_CCD_END = DEAHOUSE_CCD_TEMP_ACTEL // Last CCD Query
};
```

[see Query Dea Housekeeping: Query Id]

These identifiers are based on the Rev. C release of the DPA to DEA ICD (MIT 36-02205).

Queries which fail, either due to a time-out or because the queried board is not powered, will be indicated in the telemetered housekeeping packet as having a value of 0xffff in the queried data:

```
#define DEAHOUSE_VALUE_INVALID (0xffff)/* Result of Failed Query */
```

5.15 System Configuration Item Identifiers

The following lists the currently defined System Configuration Item Identifiers. These codes are provided in “acis_h/interface.h”

```
enum SystemSettings
{
    SETTING_DEA_POWER,           // DEA CCD-controller Power Settings
    SETTING_FEP_POWER            // FEP Power Settings
```

```
// ---- DEA Interface Board Settings ----
 SYSSET_CNTL_BASE,            // Base settings for controller board
                                // Master clock during science
 SYSSET_CNTL_MASTER_CLK = SYSSET_CNTL_BASE,
 SYSSET_CNTL_FOCAL_TEMP,       // Focal Plane Temperature
 SYSSET_CNTL_BAKE_TEMP,        // BakeOut Temperature
 SYSSET_CNTL_BAKE_ENABLE,      // BakeOut Enable[1]
 SYSSET_CNTL_LED_ENABLE,       // LED Enable
 SYSSET_CNTL_HOUSE_HOLD,       // Hold Housekeeping Address
 SYSSET_CNTL_SIGNAL_PATH,      // Signal Path Selection
 SYSSET_CNTL_CMDCLOCK_DISABLE, // Command Clock Enable Select
 SYSSET_CNTL_CMDDATA_DISABLE, // Command Data Enable Select
 SYSSET_CNTL_RELAY_SET_0,      // DEA Board Relay Selections
 SYSSET_CNTL_RELAY_SET_1,
 SYSSET_CNTL_RELAY_SET_2,
 SYSSET_CNTL_RELAY_SET_3,
 SYSSET_CNTL_RELAY_SET_4,
 SYSSET_CNTL_END = SYSSET_CNTL_RELAY_SET_4,
 SYSSET_CNTL_COUNT = SYSSET_CNTL_END - SYSSET_CNTL_BASE + 1,

 // ---- DEA CCD Controller Register Settings ----
 SYSSET_CCD_BASE = SYSSET_CNTL_END+1, // Base Settings for CCDs

                                // Sequencer Offset (for all CCDs)
 SYSSET_CCD_SEQ_OFFSET = SYSSET_CCD_BASE,
 SYSSET_CCD_ADC_OFFSET,        // Video ADC Offset
 SYSSET_CCD_VIDEO_DISABLE,     // Video Channel Disable Mask
 SYSSET_CCD_HOLD_HOUSE,        // Hold Housekeeping Address
 SYSSET_CCD_BJD,              // Back-Junction Diode Enable
 SYSSET_CCD_HIGH_SPEED_TAP,   // High-speed tap disable

// ---- DEA CCD Controller Digital-To-Analog Converter Settings ---
 SYSSET_DAC_BASE,               // Delimit Start of DAC codes
 SYSSET_DAC_PIA_P = SYSSET_DAC_BASE, // Image Array Parallel +
 SYSSET_DAC_PIA_MP,           // Image Array Parallel -+
 SYSSET_DAC_PIA_M,            // Image Array Parallel -

 SYSSET_DAC_PFS_P,            // Framestore Parallel +
 SYSSET_DAC_PFS_MP,           // Framestore Parallel -+
 SYSSET_DAC_PFS_M,            // Framestore Parallel -

 SYSSET_DAC_S_P,              // Serial Register +
 SYSSET_DAC_S_M,              // Serial Register -

 SYSSET_DAC_R_P,              // Reset Gate +
 SYSSET_DAC_R_MP,             // Reset Gate -+
 SYSSET_DAC_R_M,              // Reset Gate -

 SYSSET_DAC_SCP,              // Scupper

 SYSSET_DAC_OG_P,             // Output Gate +
 SYSSET_DAC_OG_M,             // Output Gate -

 SYSSET_DAC_RD,               // Reset Diode
```

1. Bakeout Enable is now prevented from being used without a small patch to the limit table. Attempts to set this field to any value other than zero will result in a CMDRESULT_ITEM_CLIPPED result code.

```
        SYSSET_DAC_DR0,              // Drain Output (A)
        SYSSET_DAC_DR1,              // Drain Output (B)
        SYSSET_DAC_DR2,              // Drain Output (C)
        SYSSET_DAC_DR3,              // Drain Output (D)

        SYSSET_DAC_A_OFF,            // A channel offset
        SYSSET_DAC_B_OFF,            // B channel offset
        SYSSET_DAC_C_OFF,            // C channel offset
        SYSSET_DAC_D_OFF,            // D channel offset

        SYSSET_DAC_SPARE,            // Spare DAC Channel
        SYSSET_DAC_END = SYSSET_DAC_SPARE, // Last DAC Setting
        SYSSET_DAC_COUNT = SYSSET_DAC_END - SYSSET_DAC_BASE + 1,
        SYSSET_CCD_END = SYSSET_DAC_END,   // Last CCD Controller Setting

        SYSSET_NSETTINGS = SYSSET_CCD_BASE +
                      ((SYSSET_CCD_END - SYSSET_CCD_BASE + 1)*10),
        SYSSET_COUNT
    };
    // The above settings are based on the Rev. C release
    // of the DPA to DEA ICD (MIT 36-02205)
```

[see Config Setting: Item Id]

The bit-fields of the DEA and FEP power settings correspond to the CCD and FEP identifier codes, respectively. For example, bit 0 of the SETTING_DEA_POWER field corresponds to CCD Controller I0 (CCD_I0), bit 1 corresponds to CCD Controller I1, etc. A “1” indicates that the corresponding board is to be turned on, and “0” indicates that the board should be off.

The settings from SYSSET_CCD_BASE through SYSSET_CCD_END index the settings for CCD I0. The arrays of settings for the remaining CCDs immediately follow SYSSET_CCD_END, and are indexed in CCD Id order.

5.16 Software Housekeeping Statistic Codes

The following Software Housekeeping sub-codes are used to report BEP filter statistics when the "reportgrade1" patch is active.

```
enum SwFilterId {
    SW_FILTER_NONE,         /* events unfiltered */
    SW_FILTER_EVENT,        /* events filtered by energy */
    SW_FILTER_GRADE1,       /* events filtered by SW_GRADE_CODE1 */
    SW_FILTER_GRADE2,       /* events filtered by SW_GRADE_CODE2 */
    SW_FILTER_GRADE3,       /* events filtered by SW_GRADE_CODE3 */
    SW_FILTER_GRADE4,       /* events filtered by SW_GRADE_CODE4 */
    SW_FILTER_GRADE5,       /* events filtered by SW_GRADE_CODE5 */
    SW_FILTER_OTHER,        /* events filtered by other grade */
    SW_FILTER_WIN,          /* events filtered by window */
    SW_FILTER_COUNT
};

#define SW_FILTER_SIZE      (FEP_COUNT*SW_FILTER_COUNT)
```

The following grade codes are reported in software housekeeping when the "reportgrade1" patch is active.

```
enum SwSpecialGrade {
    SW_GRADE_CODE1  = 24,
    SW_GRADE_CODE2  = 66,
    SW_GRADE_CODE3  = 107,
    SW_GRADE_CODE4  = 214,
    SW_GRADE_CODE5  = 255
};
```

The following list the currently defined Software Housekeeping codes. These codes are provided in “acis_h/interface.h”

```
enum SwStatistic
{
   SWSTAT_VERSION,           /* ACIS Software Version Number */

   SWSTAT_SWHOUSE_RANGE,     /* Housekeeping report beyond end of list */
   SWSTAT_SWHOUSE_SKIPPED,   /* Dropped Software Housekeeping Statistic */

   SWSTAT_TIMERCB_INVOKE,    /* Timer Interrupt Callback invocations */

   SWSTAT_FEPLOCK_TIMEOUT,   /* Fep Wait: timed out */
   SWSTAT_FEPLOCK_POWEROFF,  /* Fep Wait: no power */
   SWSTAT_FEPLOCK_RESET,     /* Fep Wait: is reset */
   SWSTAT_FEPLOCK_NOIO,      /* Fep Wait: No mailbox/ringbuffer */

   SWSTAT_FEPREPLY_TIMEOUT,  /* Fep Reply: timed out */
   SWSTAT_FEPREPLY_POWEROFF, /* Fep Reply: no power */
   SWSTAT_FEPREPLY_RESET,    /* Fep Reply: is reset */
   SWSTAT_FEPREPLY_NOIO,     /* Fep Reply: No mailbox/ringbuffer */

   SWSTAT_SCI_STOPRUN,       /* Science Run Stop Invoked */
   SWSTAT_SCI_STOPRUN_IDLE,  /* Stopped when already idle */
   SWSTAT_SCI_STOPRUN_RSTOP, /* Stop Request issued to mode */

   SWSTAT_SCI_STARTRUN,      /* Science Run Start Invoked */
   SWSTAT_SCI_STARTRUN_BUSY, /* Start when not idle */
   SWSTAT_SCI_STARTRUN_RUNNING,/* Start aborted previous run */
   SWSTAT_SCI_STARTRUN_RSTOP, /* Start requested stop to prv. mode */

   SWSTAT_SCI_EXPSTART_ZERO_EXPNUM,/* Exposure Number from FEP is 0 */
   SWSTAT_SCI_EXPEND_EXPNUM, /* Ending Exposure Number did not match cur */
   SWSTAT_SCI_EXPSTART_NOEND, /* Prev. Exposure missing end. */

   SWSTAT_INTR_FEPBUS,       /* FEP Bus Error Timeout [Arg: Bad Vaddr] */

   SWSTAT_TE_SHORT_DUMP_TLM, /* Timed Exposure Dump tlm pkt too small */
   SWSTAT_2D_SHORT_DUMP_TLM, /* TE 2-D Windows Dump tlm pkt too small */
   SWSTAT_TE_BAD_FEP_MODE,   /* Unrecognized Timed Exp. FEP Mode */
   SWSTAT_TE_BAD_BEP_MODE,   /* Unrecognized Timed Exp. BEP Mode */

   SWSTAT_CCD_NULL_SETTING,  /* CCD Setting Ptr NULL [Arg: setting id] */

   SWSTAT_CMDECHO_NULL,      /* Cmd Echo passed NULL Pkt ptr */
   SWSTAT_CMDECHO_MISMATCH,  /* Cmd Echo pkt != curpkt [Arg: curpkt] */
   SWSTAT_CMDECHO_BADLENGTH, /* Cmd Echo cmd too long [Arg: cmd data cnt] */
```

```
SWSTAT_CMDECHO_TRUNCATE,   /* Cmd Echo truncated [Arg: cmd data cnt] */
SWSTAT_CMDECHO_DROPPED,    /* Cmd Echo Dropped [Arg: cmd pkt id] */

SWSTAT_CMDMAN_INVALID,     /* Invalid Cmd Pkt [Arg: First word read] */
SWSTAT_CMDMAN_ERRCALLED,   /* # calls to CmdMan::handleError() */
SWSTAT_CMDMAN_ERRRETRY,    /* # retries in CmdMan::handleError() */
SWSTAT_CMDMAN_HANDLED,     /* # calls to CmdMan::handleCommand() */
SWSTAT_CMDMAN_BADLENGTH,   /* CmdMan Bad Pkt Length [Arg: Cmd Length] */

SWSTAT_DEAMAN_PRAMWRITE,   /* Bad PRAM Write Address [Arg: PRAM Index] */
SWSTAT_DEAMAN_PRAMREAD,    /* Bad PRAM Read Address [Arg: PRAM Index] */
SWSTAT_DEAMAN_SRAMWRITE,   /* Bad SRAM Write Address [Arg: SRAM Index] */
SWSTAT_DEAMAN_SRAMREAD,    /* Bad SRAM Read Address [Arg: SRAM Index] */
SWSTAT_DEAMAN_BADCNTLREG,  /* Bad DEA Cntl Reg [Arg: Reg Index] */

SWSTAT_PHHIST_BADQUAD,     /* PH Histogram Bad Quad Mode [Arg: mode] */
SWSTAT_PIX1X3_CORRUPTROW,  /* Pixel1x3 Bad Row [Arg: row] */
SWSTAT_PIX1X3_CORRUPTCOL,  /* Pixel1x3 Bad Column [Arg: col] */
SWSTAT_PMTEHIST_BADQUAD,   /* PM Te Hist Bad Quad Mode [Arg: mode] */

SWSTAT_PIX3X3_CORRUPTROW,  /* Pixel3x3 Bad Row [Arg: row] */
SWSTAT_PIX3X3_CORRUPTCOL,  /* Pixel3x3 Bad Column [Arg: col] */

SWSTAT_FEPMAN_RINGRDINDX,  /* FepMan Corrupt Rd Index [Arg: readIndex] */
SWSTAT_FEPMAN_RINGWRINDX,  /* FepMan Corrupt Wr Index [Arg: writeIndex] */

SWSTAT_PM_BADRECTYPE,      /* ProcessMode Bad Record Type [Arg: type] */

SWSTAT_DEACCD_LOADINVALID, /* CCD Cntl Start on invalid load [Arg: 0] */
SWSTAT_DEABOARD_ERROR,     /* DEA Error [Arg: (slot << 16) | errcode] */

SWSTAT_FEPCMD_MBOXSTATE,   /* FEP Mailbox not empty [Arg: mbox state] */

SWSTAT_FEP_READMEM,        /* FEP Read Memory Called [Arg: fepid] */
SWSTAT_FEP_WRITEMEM,       /* FEP Write Memory Called [Arg: fepid] */
SWSTAT_FEP_EXECMEM,        /* FEP Execute Memory Called [Arg: fepid] */
SWSTAT_FEP_STARTBIAS,      /* FEP Start Bias [Arg: none] */
SWSTAT_FEP_STOP,           /* FEP Stop Issued [Arg: abortFlag] */
SWSTAT_FEP_STARTDATA,      /* FEP Start Data Process [Arg: requestType] */
SWSTAT_FEP_QUERY,          /* FEP Query [Arg: fepid] */

SWSTAT_SMPROC_RSTOP,       /* Sci Mode Data Proc Stop Rqst [Arg: none] */
SWSTAT_SMWAITBIAS_ABORT,   /* Sci Mode Bias Wait Abort [Arg: caught] */
SWSTAT_SMWAITEVENT_CAUGHT, /* Sci Mode Event Wait Signal [Arg: caught] */
SWSTAT_SMWAITEVENT_ABORT,  /* Sci Mode Event Wait Abort [Arg: caught] */
SWSTAT_SMRABORT,           /* Sci Mode Request Abort [Arg: reason] */

SWSTAT_SCI_DUMPFAILED,     /* Sci Manager Dump Failed [Arg: none] */
SWSTAT_SCI_SETUPFAILED,    /* Sci Manager Setup Failed [Arg: none] */
SWSTAT_SCI_DEADUMPFAILED,  /* Sci Manager DEA Dump Failed [Arg: none] */
SWSTAT_SCI_DEACHECKFAILED, /* Sci Manager DEA Check Failed [Arg: none] */
SWSTAT_SCI_BIASFAILED,     /* Sci Manager Bias Failed [Arg: none] */
SWSTAT_SCI_DATACOMPLETE,   /* Sci Manager Data Run Complete [Arg: none] */
SWSTAT_SCI_BIASCOMPLETE,   /* Sci Manager Bias Run Complete [Arg: none] */

SWSTAT_SCI_INHIBIT_ON,     /* Sci Man Inhibit On [Arg: prv state] */
SWSTAT_SCI_INHIBIT_OFF,    /* Sci Man Inhibit Off [Arg: prv state] */

SWSTAT_FEPMAN_POWERON,     /* FEP Manager Power On [Arg: fepid] */
```

```
SWSTAT_FEPMAN_POWEROFF,    /* FEP Manager Power Off [Arg: fepid] */
SWSTAT_FEPMAN_STARTLOAD,   /* FEP Manager Start Prog. Load [Arg: fepid] */
SWSTAT_FEPMAN_ENDLOAD,     /* FEP Manager End Prog. Load [Arg: fepid] */
SWSTAT_DEACCD_POWERON,     /* DEA Ccd Cntl Power On [Arg: board] */
SWSTAT_DEACCD_POWEROFF,    /* DEA Ccd Cntl Power Off [Arg: board] */

SWSTAT_SCI_EXPSTART_FEPTIME,/* FEP Timestamp corrupted [Arg: fepTime] */

SWSTAT_FEPREPLY_BADTYPE,   /* FEP Reply Bad Type [Arg: fepid] */
SWSTAT_FEPREC_POWEROFF,    /* FEP Read Record No Power [Arg: fepid] */
SWSTAT_FEPREC_RESET,       /* FEP Read Record Reset [Arg: fepid] */
SWSTAT_FEPCFG_NACK,        /* FEP Config Nack [Arg: fepid] */
SWSTAT_FEPDIST_NACK,       /* FEP Distribute Cmd Nack [Arg: fepid] */

SWSTAT_SYSCFG_IN_CLIP,     /* SysCfg clipped stored item [Arg: item] */

 SWSTAT_SCI_JITTERFAILED,  /* Sci Man Jitter DAC operation failed */
SWSTAT_SMWAITTRICKLE_ABORT, /* Sci Mode Bias Trickle Abort [Arg:caught]*/

SWSTAT_FILT_FEP_0_NONE,    /* FEP_0 events unfiltered [Arg: expNum] */
SWSTAT_FILT_BASE = SWSTAT_FILT_FEP_0_NONE, /* Sci Mode Event Stats */
SWSTAT_FILT_FEP_0_ENERGY,  /* FEP_0 events filtered by energy */
SWSTAT_FILT_FEP_0_GRADE1,  /* FEP_0 events filtered by SW_GRADE_CODE1 */
SWSTAT_FILR_FEP_0_GRADE2,  /* FEP_0 events filtered by SW_GRADE_CODE2 */
SWSTAT_FILT_FEP_0_GRADE3,  /* FEP_0 events filtered by SW_GRADE_CODE3 */
SWSTAT_FILT_FEP_0_GRADE4,  /* FEP_0 events filtered by SW_GRADE_CODE4 */
SWSTAT_FILT_FEP_0_GRADE5,  /* FEP_0 events filtered by SW_GRADE_CODE5 */
SWSTAT_FILT_FEP_0_OTHER,   /* FEP_0 events filtered by other grade */
SWSTAT_FILT_FEP_0_WIN,     /* FEP_0 events filtered by window */

SWSTAT_FILT_FEP_1_NONE,    /* FEP_1 events unfiltered [Arg: expNum] */
SWSTAT_FILT_FEP_1_ENERGY,  /* FEP_1 events filtered by energy */
SWSTAT_FILT_FEP_1_GRADE1,  /* FEP_1 events filtered by SW_GRADE_CODE1 */
SWSTAT_FILT_FEP_1_GRADE2,  /* FEP_1 events filtered by SW_GRADE_CODE2 */
SWSTAT_FILT_FEP_1_GRADE3,  /* FEP_1 events filtered by SW_GRADE_CODE3 */
SWSTAT_FILT_FEP_1_GRADE4,  /* FEP_1 events filtered by SW_GRADE_CODE4 */
SWSTAT_FILT_FEP_1_GRADE5,  /* FEP_1 events filtered by SW_GRADE_CODE5 */
SWSTAT_FILT_FEP_1_OTHER,   /* FEP_1 events filtered by other grade */
SWSTAT_FILT_FEP_1_WIN,     /* FEP_1 events filtered by window */

SWSTAT_FILT_FEP_2_NONE,    /* FEP_2 events unfiltered [Arg: expNum] */
SWSTAT_FILT_FEP_2_ENERGY,  /* FEP_2 events filtered by energy */
SWSTAT_FILT_FEP_2_GRADE1,  /* FEP_2 events filtered by SW_GRADE_CODE1 */
SWSTAT_FILT_FEP_2_GRADE2,  /* FEP_2 events filtered by SW_GRADE_CODE2 */
SWSTAT_FILT_FEP_2_GRADE3,  /* FEP_2 events filtered by SW_GRADE_CODE3 */
SWSTAT_FILT_FEP_2_GRADE4,  /* FEP_2 events filtered by SW_GRADE_CODE4 */
SWSTAT_FILT_FEP_2_GRADE5,  /* FEP_2 events filtered by SW_GRADE_CODE5 */
SWSTAT_FILT_FEP_2_OTHER,   /* FEP_2 events filtered by other grade */
SWSTAT_FILT_FEP_2_WIN,     /* FEP_2 events filtered by window */

SWSTAT_FILT_FEP_3_NONE,    /* FEP_3 events unfiltered [Arg: expNum] */
SWSTAT_FILT_FEP_3_ENERGY,  /* FEP_3 events filtered by energy */
SWSTAT_FILT_FEP_3_GRADE1,  /* FEP_3 events filtered by SW_GRADE_CODE1 */
SWSTAT_FILT_FEP_3_GRADE2,  /* FEP_3 events filtered by SW_GRADE_CODE2 */
SWSTAT_FILT_FEP_3_GRADE3,  /* FEP_3 events filtered by SW_GRADE_CODE3 */
SWSTAT_FILT_FEP_3_GRADE4,  /* FEP_3 events filtered by SW_GRADE_CODE4 */
SWSTAT_FILT_FEP_3_GRADE5,  /* FEP_3 events filtered by SW_GRADE_CODE5 */
SWSTAT_FILT_FEP_3_OTHER,   /* FEP_3 events filtered by other grade */
SWSTAT_FILT_FEP_3_WIN,     /* FEP_3 events filtered by window */
```

```
    SWSTAT_FILT_FEP_4_NONE,    /* FEP_4 events unfiltered [Arg: expNum] */
    SWSTAT_FILT_FEP_4_ENERGY,  /* FEP_4 events filtered by energy */
    SWSTAT_FILT_FEP_4_GRADE1,  /* FEP_4 events filtered by SW_GRADE_CODE1 */
    SWSTAT_FILT_FEP_4_GRADE2,  /* FEP_4 events filtered by SW_GRADE_CODE2 */
    SWSTAT_FILT_FEP_4_GRADE3,  /* FEP_4 events filtered by SW_GRADE_CODE3 */
    SWSTAT_FILT_FEP_4_GRADE4,  /* FEP_4 events filtered by SW_GRADE_CODE4 */
    SWSTAT_FILT_FEP_4_GRADE5,  /* FEP_4 events filtered by SW_GRADE_CODE5 */
    SWSTAT_FILT_FEP_4_OTHER,   /* FEP_4 events filtered by other grade */
    SWSTAT_FILT_FEP_4_WIN,     /* FEP_4 events filtered by window */

    SWSTAT_FILT_FEP_5_NONE,    /* FEP_5 events unfiltered [Arg: expNum] */
    SWSTAT_FILT_FEP_5_ENERGY,  /* FEP_5 events filtered by energy */
    SWSTAT_FILT_FEP_5_GRADE1,  /* FEP_5 events filtered by SW_GRADE_CODE1 */
    SWSTAT_FILT_FEP_5_GRADE2,  /* FEP_5 events filtered by SW_GRADE_CODE2 */
    SWSTAT_FILT_FEP_5_GRADE3,  /* FEP_5 events filtered by SW_GRADE_CODE3 */
    SWSTAT_FILT_FEP_5_GRADE4,  /* FEP_5 events filtered by SW_GRADE_CODE4 */
    SWSTAT_FILT_FEP_5_GRADE5,  /* FEP_5 events filtered by SW_GRADE_CODE5 */
    SWSTAT_FILT_FEP_5_OTHER,   /* FEP_5 events filtered by other grade */
    SWSTAT_FILT_FEP_5_WIN,     /* FEP_5 events filtered by window */

    SWSTAT_COUNT,
    SWSTAT_LAST = SWSTAT_COUNT - 1 /* Last slot is sent but unused */
};
```

[see Sw Housekeeping: Statistics]

5.17 Fatal Error Codes

The following lists the currently defined fatal error codes. These codes are provided in “acis_h/interface.h”.

```
enum FatalCode
{
    FATAL_UNKNOWN = 0,        /* Unknown Fatal Error */
    FATAL_RTXERROR,           /* Nucleus RTX generated fatal error */
    FATAL_EXCEPTION,          /* Processor Exception *
    FATAL_INTERRUPT,          /* Unexpected Interrupt Cause */
    FATAL_FEPDEVICE_BAD_FEPID,/* Bad FEP Id */
    FATAL_TASK_EXIT,          /* Task returned [Arg: Task Ptr] */
    FATAL_RTX_RETURNED,       /* Nucleus RTX Returned */
    FATAL_INTR_FEP_BUS_ERROR, /* FEP Bus Error [Arg: Bad Vaddr] */

    FATAL_LAST,               /* Last slot unused */
    FATAL_COUNT
};
```

[see Fatal Message: Fatal Code, Bep Startup Message: Last Fatal Code]

5.18 Bias Algorithm Selection Codes

The following define the Bias Algorithm selection codes for Timed Exposure Mode and Continuous Clocking Mode. The determination of which algorithm is used and which parameters affect the computation is derived from the parameter values.

5.18.1 Timed Exposure

```
typedef enum {
 FEP_NO_BIAS,          /* none */
 FEP_BIAS_1,           /* algorithm #1:Whole Frame Mode */
 FEP_BIAS_2            /* algorithm #2:Strip Mode */
} fepBiasType;
```

The following table (copied from the “FEP Timed Exposure Bias Calibration” section of the Detailed Design Specification (MIT 36-53226)) indicates how the parameters from the Timed Exposure Parameter Block (see “Load Te Block” in the IP&CL Structures definitions) affect the bias computation for each mode:

TABLE 2. Timed Exposure Bias Parameter Usage

Field Name	**“Whole-Frame” Mode**	**“Strip” Mode**
Bias Algorithm Id	FEP_BIAS_1	FEP_BIAS_2
Bias Arg 0	Number of conditioning exposures (PHASE2)	Number of exposures per pixel
Bias Arg 1	Number of approximation-to-mean exposures (PHASE3), including the conditioning exposures listed in Bias Arg 0.	=0 to use ***mean*** =1 to use ***fractile*** =2 to use ***medmean***
Bias Arg 2	Rejection threshold for low-pixel elimination (immediately prior to PHASE3)	For ***mean*** and ***medmean***, specifies σ rejection criterion. For ***fractile***, index of sorted pixel array.
Bias Arg 3	Threshold for event rejection (PHASE3)	Specifies how many of the largest samples are to be removed from the pixel array before applying the ***mean***, ***medmean***, or ***fractile*** algorithm
Bias Arg 4	Rejection threshold for approximation-to-mean	Specifies how many of the smallest samples are to be removed from the pixel array before applying the ***mean***, ***medmean***, or ***fractile*** algorithm

[see Load Te Block: Bias Algorithm Id, Load Te Block: Bias Arg]

5.18.2 Continuous Clocking

The following table (copied and adjusted from the “FEP Continuous Clocking Bias Calibration” section of the Detailed Design Specification (MIT 36-53227)) indicates how the parameters from the Continuous Clocking Parameter Block (see “Load Cc Block” in the IP&CL Structures definitions) affect the bias computation

TABLE 3. Continuous Clocking Bias Parameter Usage

Field Name	Description
Bias Algorithm Id	=0 to use the Iterated Mean algorithm, ***mean*** =1 to use the Fractile algorithm, ***fractile***
Bias Rejection	For ***mean***, specifies σ rejection criterion. For ***fractile***, index of sorted pixel array.

[see Load Cc Block: Bias Algorithm Id, Load Cc Block: Bias Rejection]

5.19 FEP Science Report Error Codes

The following list the FEP error codes supplied by the Science Run report. The first set of codes are defined by the FEP/BEP interface, and are supplied by the FEP in response to a command or action initiated by the BEP. The second set are defined by the BEP to report power, reset conditions, or I/O error conditions when attempting to access a FEP. All of the definitions are provided in “acis_h/interface.h”

```
typedef enum {
 FEP_CMD_NOERR= 0,           /* no errors detected */
 FEP_CMD_ERR_NO_RUN,         /* no command currently running */
 FEP_CMD_ERR_UNK_CMD,        /* unknown command type */
 FEP_CMD_ERR_PARM_LEN,       /* parameter block too long */
 FEP_CMD_ERR_PARM_TYPE,      /* unknown parameter block type */
 FEP_CMD_ERR_QUAD_CODE,      /* unknown quadrant code */
 FEP_CMD_ERR_BIAS_TYPE,      /* unknown bias type code */
 FEP_CMD_ERR_BIAS_PARM0,     /* bad bias parm 0 */
 FEP_CMD_ERR_NROWS,          /* bad number of rows */
 FEP_CMD_ERR_NCOLS,          /* bad number of columns */
 FEP_CMD_ERR_NOCLK,          /* bad number of overclocks */
 FEP_CMD_ERR_NHIST,          /* bad histogram exposure count */
 FEP_CMD_ERR_NO_PARM,        /* no parameter block loaded */
 FEP_CMD_ERR_BAD_CMD,        /* illegal secondary command */
 FEP_CMD_ERR_NO_BIAS,        /* no bias map stored */
} fepCmdRetCode;

enum FepIoErrors
{
 FEP_ERR_LOCK_TIMEOUT   = 0x80, // Timeout on FEP lock
 FEP_ERR_NO_POWER       = 0x81, // FEP has no power
 FEP_ERR_IS_RESET       = 0x82, // FEP is reset
 FEP_ERR_NO_CMDRING     = 0x83, // FEP program has no command mailbox
 FEP_ERR_REPLY_TIMEOUT  = 0x84, // FEP reply timed-out
 FEP_ERR_BAD_REPLY_TYPE = 0x85, // FEP produced bad reply
 FEP_ERR_BAD_MBOX_STATE = 0x86  // FEP mailbox state invalid
};
```

[see Science Report: Fep Error Codes]

5.20 FEP Science Mode Codes

The following define the FEP mode codes for Timed Exposure and Continuous Clocking Mode. These definitions are provided in “acis_h/interface.h”.

5.20.1 Timed Exposure

```
enum TeFepMode
{
    FEP_TE_MODE_RAW,    // Raw Mode
    FEP_TE_MODE_HIST,   // Histogram Mode
    FEP_TE_MODE_EV3x3,  // 3x3 Event Detection Mode
    FEP_TE_MODE_EV5x5,  // 5x5 Event Detection Mode
    FEP_TE_MODE_CTI1,   // 5x5 CTI Reporting Mode
    FEP_TE_MODE_CTI2,   // 3x3 CTI Reporting Mode

    FEP_TE_MODE_COUNT,
    FEP_TE_LASTMODE = FEP_TE_MODE_COUNT - 1
};
```

[see Load Te Block: Fep Mode]

5.20.2 Continuous Clocking

```
enum CcFepMode
{
    FEP_CC_MODE_RAW,    // Raw Mode
    FEP_CC_MODE_EV1x3,  // 1x3 Event Detection Mode
    FEP_CC_MODE_EV3x3.  // 3x3 Event Detection Mode

    FEP_CC_MODE_COUNT,
    FEP_CC_LASTMODE = FEP_CC_MODE_COUNT - 1
};
```

[see Load Cc Block: Fep Mode]

5.21 BEP Packing Mode Codes

The following define the BEP Event List Packing Codes for Timed Exposure and Continuous Clocking Mode. These definitions are provided in “acis_h/interface.h”.

5.21.1 Timed Exposure

```
enum TeBepMode
{
    BEP_TE_MODE_FAINT,         // 3x3 Faint Mode Event Telemetry
    BEP_TE_MODE_FAINTBIAS,     // 3x3 Faint with Bias Event Telemetry
    BEP_TE_MODE_GRADED,        // 3x3 Graded Event Telemetry
    BEP_TE_MODE_EVHIST,        // 3x3 Event Histogram Telemetry

    BEP_TE_MODE_COUNT,
    BEP_TE_LASTMODE = BEP_TE_MODE_COUNT - 1
};
```

[see Load Te Block: Bep Packing Mode]

5.21.2 Continuous Clocking

```
enum CcBepMode
{
   BEP_CC_MODE_FAINT,         // 1x3 Faint Mode Event Telemetry
   BEP_CC_MODE_GRADED,        // 1x3 Graded Event Telemetry

   BEP_CC_MODE_COUNT,
   BEP_CC_LASTMODE = BEP_CC_MODE_COUNT - 1
};
```

[see Load Cc Block: Bep Packing Mode]

5.22 Science Mode Termination Codes

The following list the science mode termination reason codes. These codes are provided in “acis_h/interface.h”

```
enum SmTerminationCode
{
 SMTERM_UNUSED,            /* Unused */
 SMTERM_STOPCMD,           /* Commanded to Stop i.e. normal term.*/
 SMTERM_BIASDONE,          /* Bias-only Run completed */
 SMTERM_RADMON,            /* Radiation Monitor was asserted */
 SMTERM_CLOBBERED,         /* Clobbered by another start command */

 SMTERM_FEP_BIAS_START,    /* FEP Bias Processing did not start */
 SMTERM_FEP_DATA_START,    /* FEP Data Processing did not start */
 SMTERM_CCD_BIAS_START,    /* Cmd. start clock. CCDs for bias failed */
 SMTERM_CCD_DATA_START,    /* Cmd. start clock. CCDs for data failed */
 SMTERM_CCD_BIAS_STOP,     /* Cmd. stop clock. CCDs for bias failed */

 SMTERM_PROC_PARM_INVALID, /* Processing Parameter out of range */
 SMTERM_DEA_PARM_INVALID,  /* DEA Parameter out of range */
 SMTERM_FEP_PARM_INVALID,  /* FEP Parameter out of range */
 SMTERM_FEP_CONFIG_ERROR,  /* FEP Configuration Error */

 SMTERM_DEA_IO_ERROR,      /* I/O errors, or no CCD controllers on */
 SMTERM_FEP_IO_ERROR,      /* I/O errors, or no FEPs are on */

 SMTERM_UNSPECIFIED,       /* Reason is unspecified */
};
```

[see Science Report: Termination Code]

5.23 Miscellaneous FEP Constants

The following list some miscellaneous constants and limits used by the Front End Processor software:

```
#define BIAS_BAD    (0xffe)/* bias parity error value */
#define PIXEL_BAD   (0xfff)/* pixel in bad pixel map */
#define MAX_NOCLK   (30)     /* maximum overclocks per output node */
#define MAX_NOCLKR  (16)    /* max raw overclocks per output node */
#define MAX_NROWS   (1024)  /* maximum number of pixel rows */
#define MAX_NCOLS   (1024)  /* maximum number of pixel columns */
#define MAX_STRIPS  (64)   /* maximum bparm[0] in strip mode */
```

```
#define CCLK_NROWS  (512)   /* number of pixel rows in CClk mode */
#define PIXEL_MASK   (0xfff)/* valid pixel and bias bits */
#define INITSKIP     2       /* number of science exposures to skip */
```

5.24 Miscellaneous BEP Constants

The following list some miscellaneous constants and limits used by the Back End Processor software:

```
/* ------------------------------------------------------------------------
 * BEP Graded Mode Special Codes
 * ------------------------------------------------------------------------ */

enum GradedMeanCode
{
    CORNER_MEAN_LOW = -4096,  /* Graded Mode Corner Mean below -4095 */
    CORNER_MEAN_MISSING = 4095/* Graded Mode No Valid Corner Pixels */
};

[see Event Te Graded:Corner Mean]
```

5.25 Bias Parity Errors in Te5x5 Mode

A feature in the implementation of Timed Exposure 5x5 mode within the FEP can generate bogus bias parity errors. With the standard *digestbiaserror* patch installed, these can be readily identified within *patchDataBiasError* packets from the following unique combination of field values:

```
biasErrors = {
   row             = 0
   column          = 1022
   evenPixelFlags  = 15
   oddPixelFlags   = 0
}
```

All *patchDataBiasError* elements with this combination of values should be discarded.

6.0 Miscellaneous Notes

6.1 Telemetry Fill Pattern

When the ACIS Back End Processor (BEP) has no telemetry packets to send when queried by the spacecraft for data, the BEP hardware places a constant byte value into the telemetry stream. This value is **0xb7** (hexadecimal).

When the instrument is not powered, the telemetry hardware logic tends to float to a logical 1, usually producing a fill byte of **0xff** (don't count on this).

6.2 System Configuration Table Limits

Some of the numerically possible analog settings in the System Configuration Table (see Section 5.15) can cause voltages that exceed the tested limits of the CCDs to be applied to the CCDs. In order to prevent this, the instrument contains a table of maximum value settings for each System Configuration Table item. If a command is received to change a setting which exceeds its limit value, the value will be clipped to its maximum value. The following limits are applied to their corresponding settings:

TABLE 4. System Configuration Setting Limits

Item Identifier	Voltage Limit	DAC Limit	Item Description
SYSSET_DAC_PIA_P,	12.775 V	255	Image Array Parallel +
SYSSET_DAC_PIA_MP,	12.775 V	255	Image Array Parallel -+
SYSSET_DAC_PIA_M,	-7.025V	140	Image Array Parallel -
SYSSET_DAC_PFS_P,	12.775 V	255	Framestore Parallel +
SYSSET_DAC_PFS_MP,	12.775 V	255	Framestore Parallel -+
SYSSET_DAC_PFS_M,	-7.025V	140	Framestore Parallel -
SYSSET_DAC_S_P,	12.775 V	255	Serial Register +
SYSSET_DAC_S_M,	-7.025V	140	Serial Register -
SYSSET_DAC_R_P,	12.775 V	255	Reset Gate +
SYSSET_DAC_R_MP,	12.775 V	255	Reset Gate -+
SYSSET_DAC_R_M,	-7.025V	140	Reset Gate -
SYSSET_DAC_SCP,	12.775 V	255	Scupper
SYSSET_DAC_OG_P,	12.775 V	255	Output Gate +
SYSSET_DAC_OG_M,	-7.025V	140	Output Gate -
SYSSET_DAC_RD,	11.7V	233	Reset Diode
SYSSET_DAC_DR0,	20.6V (8.9V)	177	Drain Output (A)
SYSSET_DAC_DR1,	20.6V (8.9V)	177	Drain Output (B)
SYSSET_DAC_DR2,	20.6V (8.9V)	177	Drain Output (C)
SYSSET_DAC_DR3,	20.6V (8.9V)	177	Drain Output (D)

7.0 Table Description

The ACIS software structures are described using a single table. Each column in the table describes a particular attribute or property of a software record or field belonging to a record. Each row in the table describes a field of a record. By convention, columns which describe a record are included only in the first field of the record.

7.1 Bit and Byte Order

Within ACIS, all data is packed least-significant bit first. For example, two 12-bit words are bit-packed as follows, where *w0* corresponds to the first word to pack, *w1* corresponds to the second, and *b0* through *b11* indicate the bits of the corresponding word (*b0* corresponds to the least-significant bit, and *b11* corresponds to the most-significant bit of the 12-bit value):

TABLE 5. ACIS Internal Bit Order

Byte Location	Bit 7 (msb)	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0 (lsb)
0	w0	w0	w0	w0	w0	w0	w0	w0
	b7	b6	b5	b4	b3	b2	b1	b0
1	w1	w1	w1	w1	w0	w0	w0	w0
	b3	b2	b1	b0	b11	b10	b9	b8
2	w1	w1	w1	w1	w1	w1	w1	w1
	b11	b10	b9	b8	b7	b6	b5	b4

7.2 Column Definitions

The column definitions are as follows:

TABLE 6. ACIS Software IP&CL Structures Table Column Definitions

Column Number	Name	Description
A	Record Description	The first field of each record contains a description of the record in this column.
B	Record Name	This is the name of the record being described. This must be filled in for each field in the table.
C	Field Number	This is the order of the field within the record.
D	Subfield Number	Unused by ACIS
E	Owner	This field indicates the owner of the field. For ACIS, this field always contains the string “ACIS”
F	Field Name	This is the name of the field.

TABLE 6. ACIS Software IP&CL Structures Table Column Definitions

Column Number	Name	Description
G	Mnemonic	This is intended to be the 8-character mnemonic of the field. Currently, only the last two characters are meaningful. “CI” indicates that the fields belong to a command packet record, “XF” indicates that they belong to a telemetry packet record, and “IT” indicate that the fields belong to an internal structure definition, usable within both command and telemetry packet records.
H	Type	This indicates the data type of the field. This may contain the following: bit - The field is a bit-field, whose length is determined by the dimension column (see column I). If field’s Start Range (see Column O) is negative, the value of the field is represented using two’s complement, otherwise, the field is treated as unsigned. uint[8,16,32] - The field is an 8, 16, or 32 bit unsigned integer. int[8,16,32] - The field is an 8, 16, or 32 bit 2’s complement signed integer “record name” - The field is a structure whose’s type is defined by the referenced record name.
I	Dimension	This is used to define an array of elements, or to specify the number of bits in a bit-field. This may be defined as a constant number, or as a formula
J	Max. Dimension	Unused by ACIS
K	Variable	Unused by ACIS
L	Field Text	This describes the contents of the field
M	Size in Bytes	Unused by ACIS
N	Total Size	Unused by ACIS
O	Start Range	This is the lowest value that can be contained to the field.
P	End Range	This is the largest value that can be contained in the field.
Q	Units	This describes the units of the field. This is currently unspecified by ACIS.
R	Value	This specifies the value of the field. This field is unused by ACIS. The corresponding information is provided using the Start Range and End Range fields.
S	Format	Unused by ACIS
T	Question	Unused by ACIS
U	Alignment	This specifies the bit-alignment requirement of the field. For fields defined as an array or record, this specifies the alignment requirement of the first element in the array, or first field in the referenced record.
V	Command Timeout	This defines the minimum time to allow for a command record to be handled by ACIS. This item is specified for the first field of command records only.

TABLE 6. ACIS Software IP&CL Structures Table Column Definitions

Column Number	Name	Description
W	Tag	This is used by ACIS to assist in implementing the described format. It specifies the symbolic Command Opcode, or Telemetry Format Tag for command and telemetry packet records, respectively.
X	Allocator	This is used by ACIS to assist its code-generator. This specifies the telemetry packet buffer allocator to use for the corresponding telemetry packet record. It is only specified for telemetry packets.
Y	Implementation Key	This is used by ACIS to assist its code-generator.

7.3 Dimension Formula Description

The dimension field in the table structure may express a constant integer value, or a function of one or more fields in the record description. The dimension formula may use the following operators:

+ .....- Addition

- .....- Subtraction

***** .....- Multiplication

DIV .....- Integer division (truncated)

bitoffset.....- The bit-position of the field within the record

bitsize(*record name or type***)**.....- The number of bits occupied by the referenced record

Typically, the ACIS table definitions use a constant integer value for the dimension, however, some dimensions are coupled to the length of the containing command packet or telemetry packet.

Since the Command Packet Length field describes the total number of 16-bit words in a command packet, command packets which contain a variable number of elements at the end of the packet usually use the following form to describe the number of elements:

(Command Length ***** 16 **- bitoffset) DIV bitsize(***field's data type***)**

Since the Telemetry Packet Length field describes the total number of 32-bit words in the packet. Telemetry packets which contain a variable number of elements at the end, usually use the following form:

(Telemetry Length ***** 32 **- bitoffset) DIV bitsize(***field's data type***)**

In order to shorten the formula text in the simplified format descriptions, the following abbreviations are used:

Command Length.....CL

Telemetry Length .....TL

bitoffset .....OFF

bitsize .....SIZE

TITLE: ACIS Flight Software Optional Patch Component Release Notes

DOCUMENT NUMBER: 36-58020 REVISION: C

ORIGINATOR: Peter G. Ford <pgf@space.mit.edu>

LETTER	SCO NO.	DESCRIPTION	APPROVED	DATE
01	36-987	Initial numeric release	jimf	11/12/1998
A	36-1007	Bug fixes, incorporate tests	RFG	05/12/1999
B	36-1019	Add new patches, retest	RFG	12/16/1999
C	36-1022	Add smtimedlookup, cti*, compres	RFG	03/21/2003

```
======================================================================

Title: ACIS Optional Patch Release Notes for Version C

Software Change Order: 36-1022

Build Date:   Tue Feb 18 13:19:34 EST 2003
Part Number:  36-58020
Version:      C
CVS Tag:      release-B-opt-C

Std Number:   36-58010
Std Version:  B
Std Tag:      release-B
Std SCO:      36-1019

IPCL Number:  36-53204.0204
IPCL Version: N
IPCL CVS Tag: release-N

----------------------------------------------------------------------
Description:
        This is the third letter release of the optional patch set for the
        ACIS Flight Software. The purpose of this release is to incorporate
        new optional patches into the Rev. B Standard Patch release, and to
        retest all optional patches against the Rev. B Standard Patch release.

        Although the patches listed in this release have been tested in
        combination with the standard patch release, they have NOT been tested
        in various combinations with each other as part of this release. Each
        needed combination will be provided a distinct part number, and will
        be released invidually, based on the patches provided in this release.

        This release relies on the patches produced by Revision B of the
        standard patch release. Refer to MIT 36-58010, Rev. B.

        This release consists of the following optional flight patches, where
        * indicates additional or modified patches since the previous release:

                  cc3x3          - Continuous Clocking 3x3 Event Mode
                  ccignore       - Ignore Continuous Clocking data frames
                * compressall    - Fixes SPR 134
                * ctireport1     - Reports precursor charge
                * ctireport2     - Reports precursor charge
                * eventhist      - Timed Exposure Event Histogram Mode
                * reportgrade1   - Addresses SPR 132
                * smtimedlookup  - Supports eventhist and ctireport*
                  teignore       - Ignore Timed Exposure data frames
                * untricklebias  - Fixes SPR 133

        This release also contains a set of informally controlled engineering
        patches, used for ground testing, debugging and experimentation:

                  hybrid         - Prototype of a hybrid clocking mode
                  squeegy        - Prototype of a squeegee clocking mode
                  fepbiasparity1 - Prototype of the fepbiasparity2 patch
                  forcebiastrickle - Patch to set trickleBias flag
                  tlmio          - Telemetry Standard I/O Utility Routines
                  printswhouse   - Print S/W Housekeeping reports in realtime
                  deaeng         - Detect/configure for DEA Engineering video boards
                  dearepl        - Stubs for use when a DEA is not attache

----------------------------------------------------------------------
```

```
Addressed Problem Reports:
        SPR-134
        SPR-126
        SPR-132
        SPR-133
        SPR-120
        SPR-124

------------------------------------------------------------------------
Included Patches:
        cc3x3 (4636 bytes)
        ccignore (36 bytes)
        compressall (2368 bytes)
        ctireport1 (5452 bytes, depends on smtimedlookup)
        ctireport2 (2784 bytes, depends on smtimedlookup)
        deaeng (2604 bytes, depends on tlmio, conflicts with dearepl)
        dearepl (556 bytes, conflicts with deaeng)
        eventhist (5908 bytes, depends on smtimedlookup)
        printswhouse (7224 bytes, depends on tlmio)
        reportgrade1 (816 bytes)
        smtimedlookup (3712 bytes)
        teignore (36 bytes)
        tlmio (10312 bytes)
        untricklebias (1612 bytes)
```

```
======================================================================

Patch Name: reportgrade1

Part Number: 36-58030.22
Version:     A
SCO:         36-1021
Environment: flight

Conflicts:
Depends On:
Size:        816 bytes

Bcmd File:   opt_reportgrade1.bcmd
Pkts File:   opt_reportgrade1.pkts

Description:
        This patch reports per-FEP event filtering statistics via software
        housekeeping. The SwHousekeeper constructor is patched in order to
        add an extra 54 housekeeping codes, 9 per FEP, as follows:

            SW_FILT_NONE,    /* events unfiltered */
            SW_FILT_EVENT,   /* events filtered by energy */
            SW_FILT_GRADE1,  /* events filtered by SW_GRADE_CODE1 */
            SW_FILT_GRADE2,  /* events filtered by SW_GRADE_CODE2 */
            SW_FILT_GRADE3,  /* events filtered by SW_GRADE_CODE3 */
            SW_FILT_GRADE4,  /* events filtered by SW_GRADE_CODE4 */
            SW_FILT_GRADE5,  /* events filtered by SW_GRADE_CODE5 */
            SW_FILT_OTHER,   /* events filtered by other grade */
            SW_FILT_WIN,     /* events filtered by window */

        These SwStatistic codes begin at a value of SWSTAT_FILTER_BASE. They
        are defined in "acis_h/interface.h", along with the 5 special grade
        codes:

            SW_GRADE_CODE1  = 24,
            SW_GRADE_CODE2  = 66,
            SW_GRADE_CODE3  = 107,
            SW_GRADE_CODE4  = 214,
            SW_GRADE_CODE5  = 255

        Thus, the number of grade 214 events rejected by FEP_3 during the
        current housekeeping interval will be reported in swHousekeeping
        packets with a "statistics[].swStatisticId" value of
        SWSTAT_FILTER_BASE+SW_FILT_GRADE4+(9*FEP_3). The corresponding
        "statistics[].count" field will contain the number of events in this
        particular class from this particular FEP during the current ~64 sec
        housekeeping interval. As an aide to synchronizing housekeeping data
        and event packets, the "statistics[].value" field will contain the
        most recent exposure number read from this FEP during this interval.



Applicable Reports/Requests:
        SPR-132



Test Results:
        testTe --> PASS
        testCc --> PASS
```

Replaced Functions:
PmEvent::filterEvent

Command Impact:
None.

Telemetry Impact:
No reduction of telemetry throughput is anticipated. To identify the new housekeeping fields, ground software must recognize the new SwStatistic codes. Refer to the ACIS Software IP&CL Release Notes, Rev. L or later, for details

Science Impact:
None.

==========

Patch Name: untricklebias

Part Number: 36-58030.28
Version: A
SCO: 36-1028
Environment: flight

Conflicts:
Depends On:
Size: 1612 bytes

Bcmd File: opt_untricklebias.bcmd
Pkts File: opt_untricklebias.pkts

Description:
For reasons unknown, the BEP has occasionally run the science and bias thief tasks simultaneously. This causes the FEPs to start searching for x-ray events while the BEP is copying their bias maps to telemetry. If the threshold crossing freqency is sufficiently high, this can trigger an error in the FEP firmware leading to a "T-plane latchup" condition.

The untricklebias patch prevents this behavior by ensuring that the FEP bias maps are never accessed by the BiasThief task. Instead, the science task is given these functions.

The main routine of the bias thief task is repaced by Test_BiasThief::goTaskEntry, which does nothing beyond waking up whenever the task monitor tells it to, but goes back to sleep again immediately.

Where necessary, the remaining BiasThief methods that are called from the science task are replaced by methods that do not notify the bias thief task that a change has been made. The trickleTeBias and trickleCcBias do not need to be patched, but the checkMonitor method must be replaced with a version that is appropriate for being called from the science task. Note that it tests the EV_SM_BIAS_ABORT_RUN in the event mask: this is the value appropriate for a science task abort.

Applicable Reports/Requests:
SPR-133

Test Results:
patchTe --> PASS
patchAll --> PASS
patchCc --> PASS

Replaced Functions:
BiasThief::abort
ScienceMode::waitForBiasTrickle
BiasThief::goTaskEntry
BiasThief::biasReady
BiasThief::checkMonitor

```
Command Impact:
        None.


Telemetry Impact:
        None.


Science Impact:
        None.
```

```
======================================================================

Patch Name: deaeng

Part Number: 36-58030.11
Version:     02
SCO:         36-1010
Environment: engineering

Conflicts:   dearepl
Depends On:  tlmio
Size:        2604 bytes

Bcmd File:   opt_deaeng.bcmd
Pkts File:   opt_deaeng.pkts

Description:
        This patch provides the basic capability to detect
        and communicate with the engineering version of the
        DEA CCD controller boards. For historical reasons,
        these boards have a different interface than
        the flight CCD controllers.

        This patch relies on printf() being installed
        (see tlmio).



Applicable Reports/Requests:
        TOOL-PENDING


Test Results:
        No Tests Specified


Replaced Functions:
        DeaCcdController::updateRegister
        DeaCcdController::powerOn
        DeaCcdController::writeData


Command Impact:
        This patch will determine the type of video boards
        installed in the system. Due to the interface differences
        between boards, high-speed tap commands will not work
        on engineering video boards, but will continue to work
        on "flight-like" video boards.


Telemetry Impact:
        Since this patch calls printf(), it will result
        in TTAG_USER telemetry packets.


Science Impact:
        N/A
```

```
======================================================================

Patch Name: cc3x3

Part Number: 36-58030.06
Version:     B
SCO:         36-1018
Environment: flight

Conflicts:
Depends On:
Size:        4636 bytes

Bcmd File:   opt_cc3x3.bcmd
Pkts File:   opt_cc3x3.pkts

Description:
        This patch implements the Continuous Clocking 3x3
        Event Mode. In this mode, the instrument performs the
        standard continuous clocking manipulation of the CCDs,
        but rather than accept and telemetry 1x3 events, the mode
        processes 3x3 event islands, improving the spectral performance
        of the mode and reducing the problems associated with vertically
        split events.

        Because the Continuous Clocking parameter block only provides
        4 bits for defining the grade selection for the mode (in 1x3, only
        4 bits were necessary), this patch provides table which maps
        the 4-bit code into a set of pre-built 256-bit grade selection
        masks. In this release, the grade selection map is populated with
        masks provided by Fred Baganoff. Refer to grade_table.html for
        a description of the grade families. The following table summarizes
        the selections:

        Code 0  - Reject all grades
        Code 1  - Reject ASCA grades 1,2,3,4,5,6,7
        Code 2  - Reject ASCA grades 1,5,6,7
        Code 3  - Reject ASCA grades 1,5,7
        Code 4  - Undefined (currently rejects all grades)
        Code 5  - Undefined (currently rejects all grades)
        Code 6  - Undefined (currently rejects all grades)
        Code 7  - Reject ACIS flight grades 24,66,107,127,214,223,248,251,254,255
        Code 8  - Reject ACIS flight grades 24,107,127,214,223,248,251,254,255
        Code 9  - Reject ACIS flight grades 24,66,107,214,248,255
        Code 10 - Reject ACIS flight grades 24,66,107,214,255
        Code 11 - Reject ACIS flight grades 24,107,214,248,255
        Code 12 - Reject ACIS flight grades 24,107,214,255
        Code 13 - Reject ASCA grade 7
        Code 14 - Reject ACIS flight grade 255
        Code 15 - Accept all grades

        NOTE: CC3x3 Codes 0 and 15 have the same effect
        as their numerical equivalents in CC1x3, where 0
        will reject all events, and 15 will accept events
        with any grade code.



Applicable Reports/Requests:
        SPR-126
        SPR-120
        SPR-124
```

Test Results:
unit --> PASS
smoke --> PASS

Replaced Functions:
SmContClocking::setupFepBlock
SmContClocking::setupProcess
SmContClocking::terminate

Command Impact:
This version of CC3x3 uses different grade sets than the previous version. This may have an impact on the grade selection field of CC Parameter Block command packets already built built for CC3x3 observations.

This mode is invoked by using the FEP_CC_MODE_EV3x3 (2) in the fepMode field of the Continuous Clocking Parameter block, in conjunction with any of the BEP_CC event processing modes for the bepPackingMode field. This restricts the use of this mode to CC Faint and CC Graded modes. This patch does NOT support other Timed Exposure derived modes, such as Faint with Bias, 5x5, nor any of the exisiting nor patched histogram modes.

At the onset of a CC3x3 science run, the run will force two resets and reloads of the FEP software, the first to ensure that the boot-strap code is in the FEPs, and the second to load the patch code into the FEPs. This will always add up to 14 seconds per FEP to the start-up time of the run, compared to runs where the FEPs were already loaded and running.

To ensure that the patch is not present at the start of the next run, which may or may not be a CC3x3 run, a CC3x3 science run will always force the FEPs into a reset state at the end of the run. This will add another 7 seconds per FEP to the start up time of the run following a CC3x3 run, relative to the normal start up time, where the FEPs were already loaded and running.

These resets will also impact the power consumption of ACIS, where the system will draw up to 16 watts less than normal (with all 6 on and running) while the FEPs are held a reset state.

Refer to the ACIS Software IP&CL Structure Definitions, Rev. L or later for details.

Telemetry Impact:
This mode defines 4 new telemetry packet types.

When configured for FEP_CC_MODE_EV3x3 and BEP_CC_MODE_FAINT, the patch produces TTAG_SCI_CC_REC_FAINT3x3 exposure records and TAG_SCI_CC_DAT_FAINT3x3 event data packets.
When configured for FEP_CC_MODE_EV3x3 and BEP_CC_MODE_GRADED, it produces TTAG_SCI_CC_REC_GRADED3x3 exposure records and TTAG_SCI_CC_DAT_GRADED3x3 event data packets.

The size of and overhead of these packets are the same as their Timed Exposure counterparts, TTAG_SCI_TE_REC_FAINT3x3, TTAG_SCI_TE_DAT_FAINT3x3, TTAG_SCI_TE_REC_GRADED3x3 and TTAG_SCI_TE_DAT_GRADED3x3.

When used, a CC3x3 science run will produce additional Software Housekeeping counts to the FEP write and execute statistics, reflecting the additional resets and reloads of the FEPs. Runs immediately following a CC3x3 run will also produce additional FEP related counts, as they load and run the reset FEPs.

Refer to the ACIS Software IP&CL Structure Definitions, Rev. L or later for details

Science Impact:

This version of CC3x3 uses different grade sets than the previous version. The ground data analysis software may have to be aware of which version of CC3x3 is installed for a given set of CC3x3 data. Please refer to the ACIS command generation system for the set of ACIS Software Version identifiers (telemetered in the BEP Startup Message and in each Software Housekeeping telemetry packet) corresponding to the different installed CC3x3 versions.

This mode produces a new type of data product, consisting of 3x3 islands around accepted events in Continuous Clocking mode. This is intended to provide better spectral resolution and event detection performance when in Continuous Clocking mode.

This mode will not report events on row 0 and row 511, leaving a 2-row timing gap with a period of 512 rows.

As in other Continuous Clocking modes, no bias errors will be reported when in this mode, since the bias map is extremely redundant (there’s 512 copies of the bias value for any given column).

```
======================================================================

Patch Name: tlmio

Part Number: 36-58030.07
Version:     02
SCO:         36-1010
Environment: flight

Conflicts:
Depends On:
Size:        10312 bytes

Bcmd File:   opt_tlmio.bcmd
Pkts File:   opt_tlmio.pkts

Description:
        This patch provides basic standard I/O functions
        which emit TTAG_USER telemetry packets containing
        data written via calls to write().

        This patch stubs the functions open(), close() and
        read(), and implements the function write(), used
        by higher level I/O library functions, such as printf().

        The patch maintains a 1024 word telemetry buffer just
        at the end of bulk memory. write() appends data
        to this buffer until either the buffer fills, or
        until a newline is written. Once write() fills the
        buffer or a newline is encountered, the telemetry buffer
        is sent as follows:
        1. Interrupts are disabled
        2. The hardware is polled until the current packet
        is finished.
        3. The packet buffer header is filled in, and the
        first data word is set to 0 (a hook used to support
        different subtypes of TTAG_USER).
        4. Transfer the packet
        5. Wait for the transfer to complete
        6. If no transfer was in progress prior to the
        interrupt disable, clear the pending interrupt
        caused by the TTAG_USER packet transfer
        7. Reset the the buffer contents
        8. Reenable interrupts



Applicable Reports/Requests:
        TOOL-PENDING

Test Results:
        No Tests Specified


Replaced Functions:


Command Impact:
        None


Telemetry Impact:
        If this patch is used by client code (this patch itself doesn't
```

```
        initiate any messages), it will emit telemetry packets consisting
        of the tag TTAG_USER. The format of these packets consist of the
        standard telemetry header, followed by 1 32-bit word containing a zero,
        followed by the number of data words indicated by the packet length.
        If the clients of the patch issue "printf" calls, the data will consist
        of a single null-terminated ascii string.

        Word 0:          SYNC (0x736f4166)
        Word 1: [0..9]   Length (3 + "n"/4)
        Word 1: [10..31] TTAG_USER
        Word 2: 0
        Word 3..Length: Data


Science Impact:
        Since this patch "plays" with the hardware and telemetry software,
        the use of this patch may interfere with the smooth operation of
        science runs.
```

```
======================================================================

Patch Name: compressall

Part Number: 36-58030.27
Version:     A
SCO:         36-1027
Environment: flight

Conflicts:
Depends On:
Size:        2368 bytes

Bcmd File:   opt_compressall.bcmd
Pkts File:   opt_compressall.pkts

Description:
        This patch ensures that all raw mode packets are written to the
        telemetry stream without data loss. It eliminates the prior behavior
        in which, if a compressed pixel row was too long to fit into an output
        packet, the entire row was skipped and a zero-data-length was
        telemetered.

        In the new version, rows that are too long when compressed are written
        uncompressed, with the telemetry packet header fields rewritten to
        indicate that that particular packet is uncompressed.



Applicable Reports/Requests:
        SPR-134

        SER-none



Test Results:
        reproduce --> PASS
        fix --> PASS


Replaced Functions:
        PmCcRaw::digestRawRecord
        PmTeRaw::digestRawRecord

Command Impact:
        None.


Telemetry Impact:
        Ground software must examine the compressionTableSlotIndex and
        compressionTableIdentifier fields of all dataCcRaw and dataTeRaw
        packets. If their values are 255 and 0, respectively, the pixel
        array should not be decompressed.


Science Impact:
        None. Raw mode is intended for diagnostic purposes only.
```

```
======================================================================

Patch Name: ccignore

Part Number: 36-58030.10
Version:     A
SCO:         36-1004
Environment: flight

Conflicts:
Depends On:
Size:        36 bytes

Bcmd File:   opt_ccignore.bcmd
Pkts File:   opt_ccignore.pkts

Description:
        This patch causes the FEP to ignore "ignoreInitialFrames"
        frames of data at the onset of Continuous Clocking data processing.



Applicable Reports/Requests:
        SER-PENDING



Test Results:
        smoke --> PASS


Replaced Functions:


Command Impact:
        This patch will cause the start up time of a Continuous
        Clocking run to increase by "ignoreInitialFrames" times
        the frame rate configured for the run. If "ignoreInitialFrames"
        is less than 2, the 2 frames will be skipped.


Telemetry Impact:
        When "ignoreInitialFrames" is greater than 2,
        the first telemetered Continous Clocking exposure number
        will be "ignoreInitialFrames", rather than "2".


Science Impact:
        This may reduce the amount of noise in the early
        telemetered frames of the Continuous Clocking run by
        running the CCDs longer before processing and sending the data.
```

```
=====================================================================

Patch Name: eventhist

Part Number: 36-58030.05
Version:     B
SCO:         36-1025
Environment: flight

Conflicts:
Depends On:  smtimedlookup
Size:        5908 bytes

Bcmd File:   opt_eventhist.bcmd
Pkts File:   opt_eventhist.pkts

Description:
        This patch implements the Event Histogram Mode.  In this mode, the
        instrument performs the standard timed exposure clocking, and event
        detection and filtering, but rather than send the events to telemetry,
        the instrument builds CCD quadrant specific histograms of the summed
        corrected pulse heights of the accepted events.  These histograms
        contain bins 0 through 4095. Events with a pulse height above 4095 are
        counted in bin 4095 and events with a negative value are counted in
        bin 0. All histogram bin values consist of a 26-bit count, followed by
        5-bit of Hamming error detection/correction code, and 1 spare bit. The
        code is capable of detecting and correcting 1-bit errors in the count
        and hamming code bits.

        Important: This version of the eventhist patch will only run correctly
        if the smtimedlookup patch is also loaded.



Applicable Reports/Requests:


Test Results:
        smoke --> PASS
        smoke2 --> PASS


Replaced Functions:
        smTimedLookup3x3[3]
        smTimedLookup5x5[3]


Command Impact:
        As in normal Raw Histogram Mode, Event Histogram mode can only be used
        for Timed Exposure Science runs, and not in Continuous Clocking runs.

        This mode is invoked by using the FEP_TE_MODE_EV3x3 or
        FEP_TE_MODE_EV5x5 for the fepMode field of the Timed Exposure
        Parameter Block, in conjunction with the new BEP_TE_MODE_EVHIST (3)
        for the bepPackingMode field.

        Refer to the ACIS Software IP&CL Structure Definitions, Rev. M for
        details.


Telemetry Impact:
        This mode defines new telemetry formats, TTAG_SCI_TE_REC_EV_HIST for
        exposure records, and TTAG_SCI_TE_DAT_EV_HIST for histogram data
```

packets. This new mode now places the count of error corrections performed on the quadrant's histogram bins within the previously unused "Variance Overclock High" of the exposure record, TTAG_SCI_TE_REC_EV_HIST. The Rev. M version of IP&CL renames this field accordingly.

The size of these packets are the same as those for TTAG_SCI_TE_REC_HIST and TTAG_SCI_TE_DAT_HIST respectively.

This mode always requires 10 telemetry buffers for each quadrant it accumulates (9 data buffers + 1 exposure record buffer per histogram). When accumulating histograms from all 4 quadrants on all 6 CCDs, the system requires 216 data buffers, and once the histograms are complete, it requires an additional 24 exposure record buffers. ACIS is configured for 400 science telemetry buffers, and as such, has enough buffering to accumulate only 1 complete set of histograms at a time. This will cause time gaps between sets of histograms when no events are accumulated. These gaps will consist of complete exposures, so partial exposures will not be accumulated in the histograms. As the previous buffers are telemetered and released back to the telemetry pool, eventually enough buffers (to be exact, 56) will be available to hold the 2nd set of histograms. At 24Kbps (format 2), this results in a time gap on the order of half a minute to a minute, and, at 500bps (format 1), a gap on the order of a half an hour to 45 minutes.

The total transmission time for a set of histograms at 24Kbps is about 3 minutes, whereas at 500bps, it starts approaching 2 hours.

If only 5 CCDs are used, ACIS can double-buffer the histograms, eliminating this gap, assuming that the histogram count times the frame time (exposure time + overhead) is large enough to accommodate the transmission time of the histograms. The total transmission time for 5 CCDs at 24Kbps is about 2 minutes, and at 500bps, the transmission time approaches 1.5 hours.

Details of these formats are described in the ACIS Software IP&CL Structure Definitions, Rev. M.

Science Impact:
This mode produces a new type of data product, histograms of the corrected and summed pulse heights from filtered events.

```
=====================================================================

Patch Name: printswhouse

Part Number: 36-58030.08
Version:     01
SCO:         36-986
Environment: flight

Conflicts:
Depends On:  tlmio
Size:        7224 bytes

Bcmd File:   opt_printswhouse.bcmd
Pkts File:   opt_printswhouse.pkts

Description:
        This patch provides a diagnotic which prints software
        housekeeping reports to telemetry in real-time,
        using the tlmio package.



Applicable Reports/Requests:
        TOOL-PENDING


Test Results:
        No Tests Specified


Replaced Functions:
        SwHousekeeper::report


Command Impact:
        None


Telemetry Impact:
        This patch will cause the system to emit TTAG_USER
        packets containing a null terminated string, which describes
        the software housekeeping element currently being reported.
        See a description of the tlmio patch, MIT 36-58030.07.


Science Impact:
        See the tlmio patch, 36-58030.07
```

```
======================================================================

Patch Name: dearepl

Part Number: 36-58030.12
Version:     02
SCO:         36-1010
Environment: engineering

Conflicts:   deaeng
Depends On:
Size:        556 bytes

Bcmd File:   opt_dearepl.bcmd
Pkts File:   opt_dearepl.pkts

Description:
        This patch provides the basic capability to fake
        the existence of a DEA. This patch is used when
        no DEA box is available, or one wants to test
        without actually talking to the DEA.



Applicable Reports/Requests:
        TOOL-PENDING


Test Results:
        No Tests Specified


Replaced Functions:
        DeaDevice::sendCmd
        DeaManager::writeData
        DeaManager::checkLoads
        DeaDevice::isReplyReady
        DeaCcdController::updateRegister
        DeaDevice::readReply
        DeaDevice::isCmdPortReady


Command Impact:
        This "fakes" the existence of the DEAs. Commands
        which read and write PRAM, SRAM or DEA hardware
        will not crash, but won't work either.


Telemetry Impact:
        This will produce true fiction from the DEAs.


Science Impact:
        Can't do any, since the patch replaces the
        interface to the real DEAs.
```

```
======================================================================

Patch Name: teignore

Part Number: 36-58030.09
Version:     A
SCO:         36-1003
Environment: flight

Conflicts:
Depends On:
Size:        36 bytes

Bcmd File:   opt_teignore.bcmd
Pkts File:   opt_teignore.pkts

Description:
        This patch causes the FEP to ignore "ignoreInitialFrames"
        frames of data at the onset of Timed Exposure data processing.



Applicable Reports/Requests:
        SER-PENDING


Test Results:
        smoke --> PASS


Replaced Functions:


Command Impact:
        This patch will cause the start up time of a Timed Exposure
        run to increase by "ignoreInitialFrames" times the frame
        rate configured for the run. If "ignoreInitialFrames"
        is less than 2, the 2 frames will be skipped.


Telemetry Impact:
        When "ignoreInitialFrames" is greater than 2,
        the first telemetered exposure number will be
        "ignoreInitialFrames", rather than "2".


Science Impact:
        This may reduce the amount of noise in the early
        telemetered frames of the Timed Exposure run by running
        the CCDs longer before processing and sending the data.
```

==========

Patch Name: smtimedlookup

Part Number: 36-58030.24
Version: A
SCO: 36-1025
Environment: flight

Conflicts:
Depends On:
Size: 3712 bytes

Bcmd File: opt_smtimedlookup.bcmd
Pkts File: opt_smtimedlookup.pkts

Description:
This patch replaces several "switch" statements in SmTimedExposure class methods with a set of lookup tables indexed by the value of the BepMode and FepMode fields from the current TE parameter block. If a table slot is empty, the corresponding mode will be treated as unimplemented. With this patch, it is therefore possible to add more than one new TE mode via optional patches without the need to deliver a version of each patch for every possible combination of the other patches. The following methods, tables, and indices are used:

Method	lookup table	index
SmTimedExposure::setupProcess	smTimedLookupMode	FepMode
	smTimedLookup3x3	BepPackingMode
	smTimedLookup5x5	BepPackingMode
SmTimedExposure::setupFepBlock	smTimedSetupFep	FepMode
SmTimedExposure::terminate	smTimedTerminate	FepMode

These tables may be patched by an extension of the "func" directive in the *.pkg file used to describe an ACIS patch. Hence, the line

```
func smTimedLookupMode[4] Test2_SmTimedExposure::setupCti1
```

instructs the linker to insert the address of the setupCti1() method of the Test2_SmTimedExposure class into slot 4 of the smTimedLookupMode table, so that setupCti1() will be called when FepMode == 4.

Applicable Reports/Requests:

Test Results:
smoke --> PASS

Replaced Functions:
SmTimedExposure::terminate
SmTimedExposure::setupProcess
SmTimedExposure::setupFepBlock

Command Impact:

None.

Telemetry Impact:
None.

Science Impact:
None.

```
======================================================================

Patch Name: ctireport1

Part Number: 36-58030.25
Version:     A
SCO:         36-1026
Environment: flight

Conflicts:
Depends On:  smtimedlookup
Size:        5452 bytes

Bcmd File:   opt_ctireport1.bcmd
Pkts File:   opt_ctireport1.pkts

Description:
        This patch implements a variant of timed-exposure 3x3 faint event mode
        in which the presence of precursor charge in each of the three columns
        that can contribute to each event is encoded in the 16 "outlying" pixels
        of Te5x5 mode.

        FEP patches are loaded after the default code by two additional calls
        to fepManager.loadRunProgram from Test2_SmTimedExposure::setupCti1Fep.
        Once loaded, the FEPs are marked as having been reset, thereby causing
        the following run to reload their default code.

        Within the FEP, additional stack space is reserved for the cti1stk
        structure that holds the row indices and bias-subtracted pixel values
        of the most recently located precursor charge in each CCD column.

        The new FEPtestCti1 routine is called from an inline patch within
        FEPsciTimedEvent in advance of the FEPtestOddPixel or FEPtestEvenPixel
        routines. When a threshold crossing is detected, FEPtestCti1 clears
        the cti1stk array (if this is a new frame), calls FEPtestOddPixel or
        FEPtestEvenPixel, and then pushes the pixel value and row index onto
        cti1stk. If cti1stk is full, the most distant (by row) value is
        dropped.

        FEPappendCti1 is called by the patched FEP code in place of the
        original FEPappend5x5 routine. It determines the maximum bias-
        subtracted pixel value in each column, then inspects the cti1stk
        stacks for those columns, and packs up to 15 precursor charge values
        (adu and row) into elements 1 through 15 of the pe[] array:



          pe[i] = STORE_PIX(pixel - bias - delta_overclock, row_index)

        pe[0] contains three 4-bit fields, the number of successive pe[]
        precursor values corresponding to col-1, col, and col+1 of the event.



Applicable Reports/Requests:


Test Results:
        smoke --> PASS


Replaced Functions:
        smTimedLookupMode[4]
```

```
        smTimedTerminate[4]
        smTimedSetupFep[4]


Command Impact:
        This patch requires that the smtimedlookup patch must also be loaded.
        Once loaded, it is invoked by setting fepMode = FEP_TE_MODE_CTI1 in a
        loadTeBlock packet, writing that packet to a parameter block slot, and
        then starting a timed-exposure science run from that slot. The uplink
        format is defined in the ACIS IP&CL document 36-53204.0204 Rev. N.


Telemetry Impact:
        The downlinked exposure and event data packets are identical in format
        to exposureTeFaint and dataTeVeryFaint except that their formatTag
        fields contain TTAG_SCI_TE_REC_CTI1 and TTAG_SCI_TE_DAT_CTI1,
        respectively. When a TTAG_SCI_TE_DAT_CTI1 is received, precursor
        charge data will be located in the dataTeVeryFaint.pulseHeights array,
        as follows:

          pulseHeights[0]                          - three 4-bit counters
          pulseHeights[1..5,9,10,14,15,19..24]     - precursor ADU and row

        The sub-fields of pulseHeights[0] determine the contents of the
        other 15 fields:

          ncol[0] = (pulseHeights[0] >> 8) & 15  -
          ncol[1] = (pulseHeights[0] >> 4) & 15  -
          ncol[2] = pulseHeights & 15            -

        The fields from icol-1, if any, are written starting at pulseHeights[1],
        followed by those from icol, and finally those from icol+1. The ADU
        values are stored in the 7 most significant bits of pulseHeights[] and
        the row indices in the least significant 5 bits, and should be extracted
        as follows:

          adu = pulseHeights[i] & 0xfe0;
          row = (pulseheights[i] & 0x01f) << 5;

        Unused pulseHeights[] will be filled with zeroes.


Science Impact:
        This patch is intended for on-orbit diagnostic use only.
```

```
======================================================================

Patch Name: ctireport2

Part Number: 36-58030.26
Version:     A
SCO:         36-1026
Environment: flight

Conflicts:
Depends On:  smtimedlookup
Size:        2784 bytes

Bcmd File:   opt_ctireport2.bcmd
Pkts File:   opt_ctireport2.pkts

Description:
        This patch implements a variant of timed-exposure 3x3 faint event mode
        in which the presence of precursor charge in each of the three columns
        that can contribute to each event is encoded in the low-order bits of
        three of the corner pixels.

        FEP patches are loaded after the default code by two additional calls
        to fepManager.loadRunProgram from Test3_SmTimedExposure::setupCti1Fep.
        Once loaded, the FEPs are marked as having been reset, thereby causing
        the following run to reload their default code.

        Within the FEP, additional stack space is reserved for the cti2stk
        structure that holds the row indices of the most recently located
        precursor charge in each CCD column.

        The new FEPtestCti2 routine is called from an inline patch within
        FEPsciTimedEvent in advance of the FEPtestOddPixel or FEPtestEvenPixel
        routines. When a threshold crossing is detected, FEPtestCti2 clears
        the cti2stk array (if this is a new frame), calls FEPtestOddPixel or
        FEPtestEvenPixel, and then updates cti2stk to indicate that this
        column contains charge.

        FEPappendCti2 is called by the patched FEP code instead of the
        original FEPappend5x5. It finds the maximum of the 4 corner pixels
        of the event that is being reported. Then it determines whether
        any of the three contributing columns contained precursor charge.
        Finally, it encodes this information in the low order bytes of
        the three smallest corner pixels. (Since the low-order bit of
        each corner pixel may be replaced, only the 11 high-order bits
        are compared when determining the maximum value).



Applicable Reports/Requests:


Test Results:
        smoke --> PASS


Replaced Functions:
        smTimedLookupMode[5]
        smTimedTerminate[5]
        smTimedSetupFep[5]


Command Impact:
```

The uplink format is defined in the ACIS IP&CL document 36-53204.0204 Rev. N. The fepMode field in the loadTeBlock command packet must be set equal to FEP_TE_MODE_CTI2. Unless the smtimedlookup patch has also be loaded, this value will cause a subsequent startScience command that references this parameter block to fail.

Telemetry Impact:

The downlinked exposure and event data packets are identical in format to exposureTeFaint and dataTeFaint. To process the precursor charge information, ground software must first inspect the loadTeBlock reported in the dumpedTeBlock packet that started the run. If the fepMode field is equal to FEP_TE_MODE_CTI2, subsequent dataTeFaint packets should be inspected. The following code fills ee[i] with one (zero) according to whether column (ccdColumn+i-1) did (did not) contain precursor charge:

```
    unsigned nn, mm, ii, ee[3];

    for (mm = 0, nn = 2; nn < 9; nn++) {
        if ((nn & 1) == 0 && nn != 4) {
            if ((pulseHeights[nn] & 0xffe) > (pulseHeights[mm] & 0xffe))
                mm = nn;
        }
    }
    for (nn = ii = 0; nn < 9; nn++) {
        if ((nn & 1) == 0 && nn != 4 && nn != mm) {
            ee[ii++] = pulseHeights[nn] & 1;
        }
    }
```

Science Impact:

This patch is intended for on-orbit diagnostic use only.

```
/* ============================================================
 *
 * $$Source: /acis/h3/acisfs/configcntl/patches/compressall/compressall.C,v $$
 *
 * Patch Name: compressall - handle incompressible raw rows
 *
 * Description:
 *
 *  This patch ensures that all raw mode packets are written to the
 *  telemetry stream without data loss. It eliminates the prior
 *  behavior in which, if a compressed pixel row was too long to fit
 *  into an output packet, the entire row was skipped and a
 *  zero-data-length was telemetered.
 *
 *  In the new version, rows that are too long when compressed are
 *  written uncompressed, with the telemetry packet header fields
 *  rewritten to indicate that that particular packet is uncompressed.
 *
 *   The compressall.C file defines and implements the following:
 *
 *    Test_PmTeRaw            - a "friendly" subclass of PmTeRaw
 *                              which provides replacement functions for
 *                              digestRawRecord.
 *
 *      Test_PmTeRaw          - this constructor is provided to avoid
 *                              compiler/linker complaints. It is
 *                              never executed.
 *
 *      ~Test_PmTeRaw         - this constructor is provided to avoid
 *                              compiler/linker complaints. It is
 *                              never executed.
 *
 *      digestRawRecord       - this function replaces digestRawRecord
 *                              for PmTeRaw. If it cannot compress a
 *                              row into an output packet, it calls
 *                              setCompression(255) to stop compression,
 *                              constructs and posts the uncompressed
 *                              data form, and then calls setCompression
 *                              again to resume compression.
 *
 *    Test_PmCcRaw            - a "friendly" subclass of PmCcRaw
 *                              which provides replacement functions for
 *                              digestRawRecord.
 *
 *      Test_PmCcRaw          - this constructor is provided to avoid
 *                              compiler/linker complaints. It is
 *                              never executed.
 *
 *      ~Test_PmCcRaw         - this constructor is provided to avoid
 *                              compiler/linker complaints. It is
 *                              never executed.
 *
 *      digestRawRecord       - this function replaces digestRawRecord
 *                              for PmCcRaw. If it cannot compress a
 *                              row into an output packet, it calls
 *                              setCompression(255) to stop compression,
 *                              constructs and posts the uncompressed
 *                              data form, and then calls setCompression
 *                              again to resume compression.
 *
 * $$Log: compressall.C,v $
 * $Revision 1.13  2003/02/05 23:53:40  pgf
 * $Use NO_TE_COMPRESSION and NO_CC_COMPRESSION enumerations.
 * $
```

```
 * $Revision 1.12  2002/06/11 21:50:01  pgf
 * $Add comments.
 * $
 * $Revision 1.11  2002/04/09 18:48:40  pgf
 * $Fix the loader.
 * $
 * $Revision 1.10  2002/04/09 18:26:15  pgf
 * $Strip debug stuff. Add CC support.
 * $
 * $Revision 1.9  2002/04/09 16:22:15  pgf
 * $Force output packet pointer reset.
 * $
 * $Revision 1.8  2002/04/09 15:18:33  pgf
 * $Add extra call to setupDataPkt.
 * $
 * $Revision 1.7  2002/04/09 14:26:30  pgf
 * $Add more debug.
 * $
 * $Revision 1.6  2002/04/09 05:43:43  pgf
 * $Force null Huffmann table.
 * $
 * $Revision 1.5  2002/04/09 04:51:49  pgf
 * $Add more debug.
 * $
 * $Revision 1.4  2002/04/09 03:25:19  pgf
 * $Add include statement.
 * $
 * $Revision 1.3  2002/04/09 03:23:22  pgf
 * $Add calls to S/W housekeeper.
 * $
 * $Revision 1.2  2002/04/08 22:25:34  pgf
 * $Add constructor/destructor definitions.
 * $
 * $Revision 1.1  2002/03/25 19:37:44  pgf
 * $Initial test of compression fix.
 * $$
 * ========================================================== */

#include "filesscience/pmteraw.H"
#include "filesscience/pmccraw.H"

class Test_PmTeRaw : public PmTeRaw
{
public:
  Test_PmTeRaw();
  ~Test_PmTeRaw();
  Boolean digestRawRecord(const FEPeventRecRaw*record);
  enum { NO_TE_COMPRESSION = 255 };
};

Test_PmTeRaw::Test_PmTeRaw() {};
Test_PmTeRaw::~Test_PmTeRaw() {};

// -------------------------------------------------
Boolean Test_PmTeRaw::digestRawRecord(const FEPeventRecRaw*record)
// -------------------------------------------------
{
    DebugProbe probe;

    static PixelRow row;        // Static to prevent large stack use

    // extract the desired pixels from the current row
    EventExposure& exp = getExposureInfo ();
    row.setup (&exp);
```

```
row.attachData (record);
filterRow (row);

if (setupDataPkt (row) == BoolFalse)
{
    return BoolFalse;
}

unsigned pixels = 0;

// try to append the pixels to the current output packet
if (packRow(row, pixels) == BoolFalse)
{
  // if unsuccessful, post the old packet
  rawForm.put_Pixel_Count(packetPixels);
  packetPixels = 0;
  rawForm.set_Data_Written(packer.getPackedWordCnt());
  rawForm.post();

  if (setupDataPkt(row) == BoolFalse)
  {
    return BoolFalse;
  }

  // try to insert the pixels into an empty output packet
  if (packRow(row, pixels) == BoolFalse)
  {
    // if even this won't work, save the compression code
    unsigned slotindex = getCompression();

    // if we aren't compressing, this is an error
    if (slotindex >= 32)
      {
        return BoolFalse;
      }

    // reset the output buffer and stop compressing
    setCompression(NO_TE_COMPRESSION);
    unsigned* packPtr = rawForm.get_Data_Address();
    unsigned  packLen = rawForm.get_Data_Avail();
    unsigned initial = 0;
    packer.setOutputBuffer(packPtr, packLen, initial);
    Boolean retval = BoolFalse;

    // pack the row without compression
    if (packRow(row, pixels) == BoolTrue)
      {
        // complete the header and post the packet
        rawForm.put_Compression_Table_Slot_Index (NO_TE_COMPRESSION);
        rawForm.put_Compression_Table_Identifier (0);
        packetPixels += pixels;
        packPixels += pixels;
        rawForm.put_Pixel_Count(packetPixels);
        packetPixels = 0;
        rawForm.set_Data_Written(packer.getPackedWordCnt());
        rawForm.post();
        retval = BoolTrue;
      }

    // reload the original Huffman table and resume compression
    setCompression(slotindex);
    return retval;
  }
}
```

```
    packetPixels += pixels;
    packPixels += pixels;

    return BoolTrue;
}

class Test_PmCcRaw : public PmCcRaw
{
public:
  Test_PmCcRaw();
  ~Test_PmCcRaw();
  Boolean digestRawRecord(const FEPeventRecRaw*record);
  enum { NO_CC_COMPRESSION = 255 };
};

Test_PmCcRaw::Test_PmCcRaw() {};
Test_PmCcRaw::~Test_PmCcRaw() {};

// -------------------------------------------------
Boolean Test_PmCcRaw::digestRawRecord(const FEPeventRecRaw*record)
// -------------------------------------------------
{
    DebugProbe probe;

    static PixelRow row;        // Static to prevent large stack use

    // extract the desired pixels from the current row
    EventExposure& exp = getExposureInfo ();
    row.setup (&exp);
    row.attachData (record);
    filterRow (row);

    if (setupDataPkt (row) == BoolFalse)
    {
        return BoolFalse;
    }

    unsigned pixels = 0;

    // try to append the pixels to the current output packet
    if (packRow(row, pixels) == BoolFalse)
    {
      // if unsuccessful, post the old packet
      rawForm.put_Pixel_Count(packetPixels);
      packetPixels = 0;
      rawForm.set_Data_Written(packer.getPackedWordCnt());
      rawForm.post();

      if (setupDataPkt(row) == BoolFalse)
      {
        return BoolFalse;
      }

      // try to insert the pixels into an empty output packet
      if (packRow(row, pixels) == BoolFalse)
      {
        // if even this won't work, save the compression code
        unsigned slotindex = getCompression();

        // if we aren't compression, this is an error
        if (slotindex >= 32)
          {
            return BoolFalse;
          }
```

```
        // reset the output buffer and stop compressing
        setCompression(NO_CC_COMPRESSION);
        unsigned* packPtr = rawForm.get_Data_Address();
        unsigned  packLen = rawForm.get_Data_Avail();
        unsigned initial = 0;
        packer.setOutputBuffer(packPtr, packLen, initial);
        Boolean retval = BoolFalse;

        // pack the row without compression
        if (packRow(row, pixels) == BoolTrue)
          {
            // complete the header and post the packet
            rawForm.put_Compression_Table_Slot_Index (NO_CC_COMPRESSION);
            rawForm.put_Compression_Table_Identifier (0);
            packetPixels += pixels;
            packPixels += pixels;
            rawForm.put_Pixel_Count(packetPixels);
            packetPixels = 0;
            rawForm.set_Data_Written(packer.getPackedWordCnt());
            rawForm.post();
            retval = BoolTrue;
          }

        // reload the original Huffman table and resume compression
        setCompression(slotindex);
        return retval;
      }
    }

    packetPixels += pixels;
    packPixels += pixels;

    return BoolTrue;
}
```

```
#! /usr/bin/env expect
#
# $Source: /acis/h3/acisfs/configcntl/patches/compressall/testsuite/bug-hw/runtest.tcl,v $
#
# Test raw compression -- without compressall patch
#
# $Log: runtest.tcl,v $
# Revision 1.1  2002/06/11 21:49:12  pgf
# Initial version.
#

puts "Welcome to compressall/testsuite/bug-hw/runtest.tcl"

# ---- Launch the command and telemetry server processes ----
set basedir [lindex $argv 0]    ; # patch base directory
set tools [lindex $argv 1]      ; # tool directory
set patchdir [lindex $argv 2]   ; # patches under test
set test_ccd 4                  ; # CCD under test
set test_fep 1                  ; # FEP under test
set quad_mode "0 \# QUAD_ABCD"  ; # desired outputRegisterMode
set ccd_list "10 $test_ccd 10 10 10 10" ; # desired fepCcdSelect

# ---- Embed procedure library ----
source $basedir/$tools/lib/lib-exp/runtest_support.tcl

# ---- Start command pipe ----
spawn $basedir/$tools/bin/cmdclient $env(ACISSERVER)
set cmd_id $spawn_id

# ---- Start telemetry pipe ----
spawn $basedir/$tools/bin/tlmclient $env(ACISSERVER)
sleep 1

# ---- Input from image loader ----
system "make loaderselect"

# ---- Apply patches ----
cold_boot
load_patch_list "$basedir/$tools/share/opt_tlmio.bcmd\
                 $basedir/$tools/share/opt_printswhouse.bcmd\
                 $basedir/$tools/share/opt_dearepl.bcmd\
                 $basedir/$tools/share/standard.bcmd"
warm_boot

# ---- Power on FEPs and CCDs ----
power_on_boards "$ccd_list"

# ---- While powering up, load the Raw Image ----
system "make image"

# ---- Wait for FEPs to finish powering ----
expect {
    -re ".*SWSTAT_FEPMAN_ENDLOAD: $test_fep\[\r\n]" {}
    timeout { fail "Power-up Failure" }
}

# ---- Load TE Parameter Block ----
send -i $cmd_id "load 0 te 4 {
    parameterBlockId             = 0x00535014
    fepCcdSelect                 =    10     4    10    10    10    10
    fepMode                      = 0 # FEP_TE_MODE_RAW
    bepPackingMode               = 0 # BEP_TE_MODE_FAINT
    onChip2x2Summing             = 0
    ignoreBadPixelMap            = 0
```

```
    ignoreBadColumnMap          = 0
    recomputeBias               = 0
    trickleBias                 = 0
    subarrayStartRow            = 0
    subarrayRowCount            = 959
    overclockPairsPerNode       = 8
    outputRegisterMode          = 0 # QUAD_FULL
    ccdVideoResponse            =     0     0     0     0     0     0
    primaryExposure             = 33
    secondaryExposure           = 0
    dutyCycle                   = 0
    fep0EventThreshold          =     0     0     0     0
    fep1EventThreshold          =    38    38    38    38
    fep2EventThreshold          =     0     0     0     0
    fep3EventThreshold          =     0     0     0     0
    fep4EventThreshold          =     0     0     0     0
    fep5EventThreshold          =     0     0     0     0
    fep0SplitThreshold          =     0     0     0     0
    fep1SplitThreshold          =    13    13    13    13
    fep2SplitThreshold          =     0     0     0     0
    fep3SplitThreshold          =     0     0     0     0
    fep4SplitThreshold          =     0     0     0     0
    fep5SplitThreshold          =     0     0     0     0
    lowerEventAmplitude         = 0
    eventAmplitudeRange         = 3750
    gradeSelections             = 0xfeffffff 0xffffffff 0xfffffffb 0xfffff7ff
                      0xffffffff 0xffffffff 0xffbfffff 0x7fffffff
    windowSlotIndex             = 65535
    histogramCount              = 0
    biasCompressionSlotIndex    =     0     1     0     0     0     0
    rawCompressionSlotIndex     = 1
    ignoreInitialFrames         = 2
    biasAlgorithmId             =     0     1     0     0     0     0
    biasArg0                    =     0     1     0     0     0     0
    biasArg1                    =     0     1     0     0     0     0
    biasArg2                    =     0     0     0     0     0     0
    biasArg3                    =     0     0     0     0     0     0
    biasArg4                    =     0     0     0     0     0     0
    fep0VideoOffset             =     0     0     0     0
    fep1VideoOffset             =    73    75    73    83
    fep2VideoOffset             =     0     0     0     0
    fep3VideoOffset             =     0     0     0     0
    fep4VideoOffset             =     0     0     0     0
    fep5VideoOffset             =     0     0     0     0
    deaLoadOverride             = 0
    fepLoadOverride             = 0
}
"
command_echo 1 9 "load te"

# ---- Start Science Run ----
send -i $cmd_id "start 0 te 4\n"
command_echo 1 14 "start science run"

set stopsw 0
set timeout 3600

expect {
    -re "exposureTeRaw\[^\r]*\
        ccdId=(\[0-9])\[^\r]*\
        fepId=(\[0-9])\[^\r]*\
        exposureNumber=(\[0-9]+).*\[\r\n]*" {

        set ccd $expect_out(1,string)
```

```
        set fep $expect_out(2,string)
        set nexp $expect_out(3,string)

        # --- Only allow expected FEP and CCD Ids ---
        if {$fep != $test_fep} {
            fail "Bad FEP Id: $fep"
        } elseif {$ccd != $test_ccd} {
            fail "Bad CCD Id: $ccd"
        } else {
            fail "Noisy row was compressed"
        }
    }

    -re "dataTeRaw\[^\r]*\
        ccdId=(\[0-9])\[^\r]*\
        fepId=(\[0-9])\[^\r]*\
        pixelCount=(\[0-9a-fx]+)\[\r\n]*" {

        set ccd $expect_out(1,string)
        set fep $expect_out(2,string)
        set npix $expect_out(3,string)

        # --- Only allow expected FEP and CCD Ids ---
        if {$fep != $test_fep} {
            fail "Bad FEP Id: $fep"
        } elseif {$ccd != $test_ccd} {
            fail "Bad CCD Id: $ccd"
        }

        # ---- stop the run ----
        if {$stopsw == 0} {
            send -i $cmd_id "stop 0 science\n"
            incr stopsw
        }

        if {$npix != 0} {
            exp_continue
        }
    }

    -re "scienceReport\[^\r]*\
        terminationCode=(\[0-9]+)\[\r\n]*" {
        fail "BEP termination code $expect_out(1,string)"
    }

    timeout { fail "No exposure record" }
}

# ---- Wait ----
wait_stop_science

pass "Unable to compress noisy row"
```

```
#! /usr/bin/env expect
#
# $Source: /acis/h3/acisfs/configcntl/patches/compressall/testsuite/fix-hw/runtest.tcl,v $
#
# Test raw compression -- with compressall patch
#
# $Log: runtest.tcl,v $
# Revision 1.2  2003/02/05 23:56:43  pgf
# Add test of image.
#
# Revision 1.1  2002/06/11 21:49:16  pgf
# Initial version.
#

puts "Welcome to compressall/testsuite/fix-hw/runtest.tcl"

# ---- Launch the command and telemetry server processes ----
set basedir [lindex $argv 0]    ; # patch base directory
set tools [lindex $argv 1]      ; # tool directory
set patchdir [lindex $argv 2]   ; # patches under test
set test_ccd 4                  ; # CCD under test
set test_fep 1                  ; # FEP under test
set quad_mode "0 \# QUAD_ABCD"  ; # desired outputRegisterMode
set ccd_list "10 $test_ccd 10 10 10 10" ; # desired fepCcdSelect

# ---- Embed procedure library ----
source $basedir/$tools/lib/lib-exp/runtest_support.tcl

# ---- Start command pipe ----
spawn $basedir/$tools/bin/cmdclient $env(ACISSERVER)
set cmd_id $spawn_id

# ---- Start telemetry pipe ----
spawn $basedir/$tools/bin/tlmclient $env(ACISSERVER)
sleep 1

# ---- Input from image loader ----
system "make loaderselect"

# ---- Apply patches ----
cold_boot
load_patch_list "$basedir/$tools/share/opt_tlmio.bcmd\
                 $basedir/$tools/share/opt_printswhouse.bcmd\
                 $basedir/$tools/share/opt_dearepl.bcmd\
                 $basedir/$tools/share/standard.bcmd\
                 $basedir/$patchdir/opt_compressall.bcmd"
warm_boot


# ---- Power on FEPs and CCDs ----
power_on_boards "$ccd_list"

# ---- While powering up, load the Raw Image ----
system "make image"

# ---- Wait for FEPs to finish powering ----
expect {
    -re ".*SWSTAT_FEPMAN_ENDLOAD: $test_fep\[\r\n]" {}
    timeout { fail "Power-up Failure" }
}

# ---- Load TE Parameter Block ----
send -i $cmd_id "load 0 te 4 {
    parameterBlockId             = 0x00535014
```

```
    fepCcdSelect                =    10     4    10    10    10    10
    fepMode                     = 0 # FEP_TE_MODE_RAW
    bepPackingMode              = 0 # BEP_TE_MODE_FAINT
    onChip2x2Summing            = 0
    ignoreBadPixelMap           = 0
    ignoreBadColumnMap          = 0
    recomputeBias               = 0
    trickleBias                 = 0
    subarrayStartRow            = 0
    subarrayRowCount            = 959
    overclockPairsPerNode       = 8
    outputRegisterMode          = 0 # QUAD_FULL
    ccdVideoResponse            =     0     0     0     0     0     0
    primaryExposure             = 33
    secondaryExposure           = 0
    dutyCycle                   = 0
    fep0EventThreshold          =     0     0     0     0
    fep1EventThreshold          =    38    38    38    38
    fep2EventThreshold          =     0     0     0     0
    fep3EventThreshold          =     0     0     0     0
    fep4EventThreshold          =     0     0     0     0
    fep5EventThreshold          =     0     0     0     0
    fep0SplitThreshold          =     0     0     0     0
    fep1SplitThreshold          =    13    13    13    13
    fep2SplitThreshold          =     0     0     0     0
    fep3SplitThreshold          =     0     0     0     0
    fep4SplitThreshold          =     0     0     0     0
    fep5SplitThreshold          =     0     0     0     0
    lowerEventAmplitude         = 0
    eventAmplitudeRange         = 3750
    gradeSelections             = 0xfeffffff 0xffffffff 0xfffffffb 0xfffff7ff
                  0xffffffff 0xffffffff 0xffbfffff 0x7fffffff
    windowSlotIndex             = 65535
    histogramCount              = 0
    biasCompressionSlotIndex    =     0     1     0     0     0     0
    rawCompressionSlotIndex     = 1
    ignoreInitialFrames         = 2
    biasAlgorithmId             =     0     1     0     0     0     0
    biasArg0                    =     0     1     0     0     0     0
    biasArg1                    =     0     1     0     0     0     0
    biasArg2                    =     0     0     0     0     0     0
    biasArg3                    =     0     0     0     0     0     0
    biasArg4                    =     0     0     0     0     0     0
    fep0VideoOffset             =     0     0     0     0
    fep1VideoOffset             =    73    75    73    83
    fep2VideoOffset             =     0     0     0     0
    fep3VideoOffset             =     0     0     0     0
    fep4VideoOffset             =     0     0     0     0
    fep5VideoOffset             =     0     0     0     0
    deaLoadOverride             = 0
    fepLoadOverride             = 0
}
"
command_echo 1 9 "load te"

# ---- Start Science Run ----
send -i $cmd_id "start 0 te 4\n"
command_echo 1 14 "start science run"

set timeout 3600
set stopsw 0
set npix0 0

expect {
```

```
    -re "exposureTeRaw\[^\r]*\
        ccdId=(\[0-9])\[^\r]*\
        fepId=(\[0-9])\[^\r]*\
        exposureNumber=(\[0-9]+)\[^\r]*\
        pixelCount=\[0-9a-fx]+\[\r\n]*" {

        set ccd $expect_out(1,string)
        set fep $expect_out(2,string)
        set nexp $expect_out(3,string)

        # --- Only allow expected FEP and CCD Ids ---
        if {$fep != $test_fep} {
            fail "Bad FEP Id: $fep"
        } elseif {$ccd != $test_ccd} {
            fail "Bad CCD Id: $ccd"
        } elseif {$stopsw == 0} {
            send -i $cmd_id "stop 0 science\n"
        }
    }

    -re "dataTeRaw\[^\r]*\
        ccdId=(\[0-9])\[^\r]*\
        fepId=(\[0-9])\[^\r]*\
        pixelCount=(\[0-9a-fx]+)\[\r\n]*" {

        set ccd $expect_out(1,string)
        set fep $expect_out(2,string)
        set npix $expect_out(3,string)

        # --- Only allow expected FEP and CCD Ids ---
        if {$fep != $test_fep} {
            fail "Bad FEP Id: $fep"
        } elseif {$ccd != $test_ccd} {
            fail "Bad CCD Id: $ccd"
        } elseif {$npix == 0} {
            incr npix0
        }

        # --- Stop the run ---
        if {$stopsw == 0} {
            send -i $cmd_id "stop 0 science\n"
            incr stopsw
        }

        exp_continue
    }

    -re "scienceReport\[^\r]*\
        terminationCode=(\[0-9]+)\[\r\n]*" {
        fail "BEP termination code $expect_out(1,string)"
    }

    timeout { fail "No exposure record" }
}

# ---- Wait ----
wait_stop_science

if {$npix0 == 0} {
    system "psci -C -l foo -h /nfs/acis/h2/pgf/acis/psci/huff.dat pkts.raw"
    system "testimage.pl foo.1.1.2.raw.fits ../DATA/test-12.00??.fits.gz"
    pass "Compressed noisy rows"
} else {
    fail "Did not compress noisy rows"
```

```
}
```

```
#! /bin/env perl
#
# $Source: /acis/h3/acisfs/configcntl/patches/compressall/testsuite/fix-hw/testimage.pl,v $
#
# Function: testimage.pl: compare raw output image against input images
#
# Date:     Feb  5, 17:33 2003 <pgf@space.mit.edu>
#
# Syntax:   uncoo.pl output-image input-images
#
# $Log: testimage.pl,v $
# Revision 1.1  2003/02/05 23:55:58  pgf
# Initial release.
#
#-------------------------------------------------------------------------

$out = shift(@ARGV);
$off = 0;
$data = &read_image($out);

for $in (@ARGV) {
    local($comp) = &read_image($in);
    if ($comp ne substr($data, $off, length($comp))) {
        0 while substr($comp, $i++, 1) eq substr($data, $off++, 1);
        die "Mismatch: $out:$off vs. $in:$i\n"
    }
    $off += length($comp);
}

exit(0);

sub read_image {
    local ($file) = shift;
    local ($hdr, $sw, $x, $y, $nx, $ny, $off, $mit, $line, $pix, $nn, %col1, %col2);
    open(IN, "gzcat -f $file|") || die "$file: $!\n";
    @col1{'A','B','C','D'} = ( 0, 256, 512, 768 );
    @col2{'A','B','C','D'} = ( 255, 511, 767, 1023 );
    for ($sw = 0; $sw == 0; ) {
        read(IN, $hdr, 2880) == 2880 || die "$in: unexpected EOF\n";
        for ($off = 0; $sw == 0 && $off < 2880; $off += 80) {
            $_ = substr($hdr, $off, 80);
            $nx = $1 if /^NAXIS1\s*=\s*(\d+)/;
            $ny = $1 if /^NAXIS2\s*=\s*(\d+)/;
            $col1{$1} = $2-1 if /^I(.)MINCOL\s*=\s*(\d+)/;
            $col2{$1} = $2-1 if /^I(.)MAXCOL\s*=\s*(\d+)/;
            $mit = 1 if /^ORIGIN\s*=\s*\'MIT\s*\'/;
            $sw = 1 if /^END\s+$/;
        }
    }
    for ($y = 0; $y < $ny; $y++) {
        read(IN, $line, 2*$nx) == 2*$nx || die "$file: unexpected EOF\n";
        for $nn ('A', 'B', 'C', 'D') {
            $_ = substr($line, 2*$col1{$nn}, 2*($col2{$nn}-$col1{$nn}+1));
            if ($mit && ($nn eq 'B' || $nn eq 'D')) {
                $pix .= pack('S*', reverse(unpack('S*', $_)));
            } else {
                $pix .= $_;
            }
        }
    }
    close(IN);
    $file =~ s/.*\/([^\/]+)$/\1/;
    printf "%s: %dx%d pixels %d bytes\n", $file, $nx, $ny, length($pix);
    return $pix;
```

```
}
```

```
/* ============================================================
 *
 * $$Source: /acis/h3/acisfs/configcntl/patches/ctireport1/ctireport1.C,v $$
 *
 * Patch Name: ctireport1
 *
 * Description:
 *
 *   This patch implements a variant of timed-exposure 3x3 faint event mode
 *   in which the presence of precursor charge in each of the three columns
 *   that can contribute to each event is encoded in the 16 "outlying"
 *   pixels of Te5x5 mode.
 *
 *   FEP patches are loaded after the default code by two additional
 *   calls to fepManager.loadRunProgram from
 *   Test3_SmTimedExposure::setupCti2Fep. Once loaded, the FEPs are
 *   marked as having been reset, thereby causing the following run to
 *   reload their default code.
 *
 *   The new mode uses that same telemetry formats as teVeryFaint mode but
 *   with new format codes TTAG_SCI_TE_DAT_CTI1 and TTAG_SCI_TE_REC_CTI1.
 *   The FEP patches are loaded after the default code by two extra calls
 *   to fepManager.loadRunProgram from Test2_SmTimedExposure::setupCti1Fep.
 *
 *   Once loaded, the FEPs are marked as having been reset, thereby causing
 *   the following run to reload their default code.
 *
 *   The ctireport1.C file defines and implements the following:
 *
 *     Test2_SmTimedExposure    - a subclass of SmTimedExposure which
 *                                provides replacement functions for the
 *                                terminate method and new setupCti1 and
 *                                setupCti1Fep methods that are patched
 *                                into the smTimedSetup and smTimedSetupFep
 *                                arrays
 *
 *       Test2_SmTimedExposure  - this constructor is provided to avoid
 *                                compiler/linker complaints. Never invoked.
 *
 *       ~Test2_SmTimedExposure - this destructor is provided to avoid
 *                                compiler/linker complaints. Never invoked.
 *
 *       setupCti1              - this function calls pmTeCti1[].initialize
 *                                to set the correct formatTag for each
 *                                event packet.
 *
 *       setupCti1Fep           - this function initializes the event
 *                                packet headers and then makes 3 calls
 *                                to fepManager.loadRunProgram to load the
 *                                default FEP code, the new subroutines,
 *                                and finally, the inline patches.
 *
 *       terminate              - this function is called at the end of
 *                                the science run. After performing the
 *                                default actions, it calls
 *                                fepManager.resetFeps to ensure that
 *                                the default FEP code is reloaded before
 *                                the FEPs are used again.
 *
 *     PmTeCti1                 - a subclass of PmTeFaint5x5 that handles
 *                                the CTI1 event packets.
 *
 *       ~PmTeCti1              - this destructor is provided to avoid
 *                                compiler/linker complaints. Never invoked.
```

```
 *
 *       initialize             - this function inserts the correct formatTag
 *                                into the event packets
 *
 *       finishExposure         - this fuction inserts the correct formatTag
 *                                into the exposure packets, and then
 *                                releases them to the telemetry queue
 *
 * Note:
 *  This patch requires that the smtimedlookup patch must also be loaded.
 *
 * References:
 *   Refer to the 1.5 release of filesscience/smtimedexposure.C and to
 *   the method of FEP patching used in the cc3x3 patch.
 *
 * $$Log: ctireport1.C,v $
 * $Revision 1.13  2002/06/11 22:25:02  pgf
 * $Add comments.
 * $
 * $Revision 1.12  2001/04/12 17:44:17  pgf
 * $Add destructors to keep loader happy.
 * $
 * $Revision 1.11  2001/04/12 17:22:23  pgf
 * $Force call to teBlock.get_Bep_Packing_Mode() for squeegee initialization.
 * $
 * $Revision 1.10  2001/04/12 05:44:48  pgf
 * $Bring static declaration within method.
 * $
 * $Revision 1.9  2001/04/06 21:03:44  pgf
 * $Define the constructor.
 * $
 * $Revision 1.8  2001/04/06 20:46:08  pgf
 * $Add dummy constructor.
 * $
 * $Revision 1.7  2001/04/05 15:37:11  pgf
 * $Use smtimedlookup patch.
 * $
 * $Revision 1.6  2001/03/19 20:45:07  pgf
 * $Rearrange hangar pointers.
 * $
 * $Revision 1.5  2001/03/19 18:00:40  pgf
 * $Create correct FEP map.
 * $
 * $Revision 1.4  2001/03/16 16:42:21  pgf
 * $Use "fepobject" when building replacement FEP routines.
 * $
 * $Revision 1.3  2001/03/15 20:47:03  pgf
 * $Load the FEP updates in two sections.
 * $
 * $Revision 1.2  2001/03/15 00:07:49  pgf
 * $Change Test_* to Test2_* to avoid name conflicts.
 * $
 * $Revision 1.1  2001/03/14 22:30:10  pgf
 * $First very crude test version.
 * $$
 * ========================================================== */

#ifndef private
#define private public
#endif

#ifndef protected
#define protected public
#endif
```

```
#include "ipcl/interface.h"
#include "filesscience/smtimedexposure.H"
#include "filesswhouse/swhousekeeper.H"

extern SmTimedExposure smTimedExposure;

// ###################################################
// ###################################################

// --- Needed to override format tags ---
class PmTeCti1 : public PmTeFaint5x5
{
public:
    ~PmTeCti1();
    void    initialize();
    Boolean finishExposure();
};

// -------------------------------------------------
PmTeCti1::~PmTeCti1()
// -------------------------------------------------
{
}

// -------------------------------------------------
void PmTeCti1::initialize()
// -------------------------------------------------
{
    PmTeFaint5x5* tmp = new (this) PmTeCti1();

    // cast needed to remove "const" attribute and allow assignment
    *((TlmFormatTag*)&dataForm.formatTag) = TTAG_SCI_TE_DAT_CTI1;
}

// -------------------------------------------------
Boolean PmTeCti1::finishExposure()
// -------------------------------------------------
{
    packetNum = 0;

    if (dataForm.hasBuffer ())
    {
        dataForm.post ();
    }

    Tf_Exposure_Te_Faint form(TTAG_SCI_TE_REC_CTI1);

    if (waitForPkt (form) == BoolFalse)
    {
        return BoolFalse;
    }

    setupExposureRecord (form);
    form.post ();

    return BoolTrue;
}

// ###################################################
// ###################################################

class Test2_SmTimedExposure : public SmTimedExposure
{
```

```
public:
    ~Test2_SmTimedExposure();
    void    setupCti1();
    void    setupCti1Fep(FepId fep, FEPparmBlock&  fepblock);
    void    terminate();
};

extern const unsigned short ctireport1fep1[];
FepProgram* ctireport1fep1Hanger = (FepProgram*) ctireport1fep1;
extern const FepProgram* ctireport1fep2Hanger;
extern const FepProgram* defaultFepLoadHanger;

// ------------------------------------------------
Test2_SmTimedExposure::~Test2_SmTimedExposure()
// ------------------------------------------------
{
}

// ------------------------------------------------
void Test2_SmTimedExposure::setupCti1()
// ------------------------------------------------
{
    static PmTeCti1  pmTeCti1[6];

    // --- Select type of BEP event packing ---
    TeBepMode tmp = TeBepMode(teBlock.get_Bep_Packing_Mode());

    for (FepId fep = FEP_0; fep < FEP_COUNT; fep = FepId(fep+1))
      {
        CcdId ccd = fepCcd[fep];

        if (ccd < CCD_DESELECT)
          {
            pmTeCti1[fep].initialize(); // Override tag
            setupEventProcess(fep, pmTeCti1[fep]);
          }
      }
}

// ------------------------------------------------
void Test2_SmTimedExposure::setupCti1Fep(FepId fep, FEPparmBlock&  fepblock)
// ------------------------------------------------
{
    // ---- Supply fixed output packet header fields ----
    fepblock.type = FEP_TIMED_PARM_5x5;
    fepblock.nhist = 0;

    // ---- Load default FEP code and inline and subroutine patches ----
    fepManager.loadRunProgram (fep, defaultFepLoadHanger);
    fepManager.loadRunProgram (fep, ctireport1fep1Hanger);
    fepManager.loadRunProgram (fep, ctireport1fep2Hanger);
}

// ------------------------------------------------
void Test2_SmTimedExposure::terminate()
// ------------------------------------------------
{
    // ---- Flush all exposure records ----
    flushProcesses();

    // ---- Form and issue report ----
    Tf_Science_Report report;

    if (waitForPkt(report) == BoolFalse)
```

```
    {
      fepManager.resetFeps();
      return;
    }

    setupScienceReport(report);
    report.post();

    // ---- Reset the FEPs to force reload on next run ----
    fepManager.resetFeps();
}
```

```
/* ============================================================
*
* $$Source: /acis/h3/acisfs/configcntl/patches/ctireport1/ctireport1fep1.c,v $$
*
* Patch Name: ctireport1
*
* Description:
*
*   This file supplies additional FEP routines, FEPtestCti1 and
*   FEPappendCti1, that together cause the FEP to report the
*   presence of precursor charge in any of the 3 CCD columns that
*   span each candidate event.
*
*   FEPtestCti1 is called from an inline patch (ctireport2fep2.S)
*   in place of the default FEPtestOddPixel or FEPtestEvenPixel
*   routines. When a threshold crossing is detected, it clears the
*   cti1stk stack (if this is a new frame), calls FEPtestOddPixel
*   or FEPtestEvenPixel, and pushes row and amplitude information
*   onto cti1stk to indicate that this column contains charge.
*
*   FEPappendCti1 is called by the patched FEP code instead of the
*   original FEPappend5x5. It examines the three columns of the
*   cti1stk stack and retrieves information on <= 15 most significant
*   precursor charges. It packs this information into the "outlier"
*   pixel array, FEPeventRec5x5.pe[1..15], while the low 12 bits of
*   pe[0] contain 3 4-bit fields: the number of succcessive pe[]
*   elements assigned to each of the three event columns. The (bias-
*   subtracted) pixel value and row are truncated to PIXBITS and
*   ROWBITS, respectively, and packed into 12 bits by the STORE_PIX
*   macro.
*
* Note:
*  This patch requires that the smtimedlookup patch must also be loaded.
*
* References:
*   Refer to the 1.5 release of filesscience/smtimedexposure.C and to
*   the method of FEP patching used in the cc3x3 patch.
*
* $$Log: ctireport1fep1.c,v $
* $Revision 1.9  2003/03/22 00:12:08  pgf
* $Signed off for Optional release C
* $
* $Revision 1.8  2003/02/18 08:50:46  pgf
* $Fix indexing bug.
* $
* $Revision 1.7  2003/02/18 07:24:17  pgf
* $Fix bug in stack algorithm.
* $
* $Revision 1.6  2003/02/18 07:04:28  pgf
* $Refine the reporting algorithm.
* $
* $Revision 1.5  2003/02/17 01:55:47  pgf
* $Update the charge stacking and selection algorithm.
* $
* $Revision 1.4  2002/06/11 22:25:05  pgf
* $Add comments.
* $$
* ============================================================ */
#include "fepCtl.h"

#define PIXBITS 7
#define ROWBITS 5

#define PIXVAL(adu)             (((adu) & 0xfff) >> (12-PIXBITS))
```

```
#define ROWVAL(row)             (((row) & 0x3ff) >> (10-ROWBITS))
#define STORE_PIX(adu,row)      (((adu) << ROWBITS) | ROWVAL(row))

extern void FEPtestEvenPixel(unsigned irow, unsigned icol, FEPparm *fp);
extern void FEPtestOddPixel(unsigned irow, unsigned icol, FEPparm *fp);

#define NSTK 24         /* maximum val[] entries per column */
#define NCOLS 1024      /* number of columns */

struct cti1stk {
    int expnum;         /* exposure number -- must be first element */
    struct _stk {
        int     count;  /* number of val[] entries currently used */
        struct _pix {
            short row;  /* precursor charge row number */
            short adu;  /* precursor charge value */
        } val[NSTK];
    } stk[NCOLS];               /* one stack per column */
};

void FEPtestCti1(unsigned irow, unsigned icol, FEPparm *fp)
{
    unsigned *pImage = fp->image + (icol/2);
    unsigned *pBias = pImage + BIAS_OFFSET;
    struct cti1stk *cp = (struct cti1stk *)(fp->phist);
    struct _stk *sp = &cp->stk[icol];
    int pval, bval, rval, istk;

    /* on new frame, clear stack counters */
    if (cp->expnum < fp->ex.expnum) {
        cp->expnum = fp->ex.expnum;
        for (istk = 0; istk < NCOLS; istk++)
            cp->stk[istk].count = 0;
    }

    /* find and report any event */
    if (icol & 1) {
        FEPtestOddPixel(irow, icol, fp);
        pval = PIXEL1(*pImage) & PIXEL_MASK;
        bval = PIXEL1(*pBias) & PIXEL_MASK;
    } else {
        FEPtestEvenPixel(irow, icol, fp);
        pval = PIXEL0(*pImage) & PIXEL_MASK;
        bval = PIXEL0(*pBias) & PIXEL_MASK;
    }

    /* test for bad pixel or parity error */
    pval -= bval + fp->ex.dOclk[icol >> fp->colshft];
    if (bval >= BIAS_BAD || pval < 0)
        return;

    /* reduce significance of ADU and row values */
    pval = PIXVAL(pval > 4095 ? 4095 : pval);

    /* locate slot in stack (ADU values decrease with stack index) */
    for (istk = sp->count; istk > 0 ; istk--) {
        if (sp->val[istk-1].adu > pval)
            /* don't stack new ADU value if top row index unchanged */
            if (ROWVAL(sp->val[istk-1].row - irow) == 0) {
                sp->count = istk;
                return;
            } else
                break;
    }
```

```
    /* if the stack is full, shift off the bottom element */
    if (istk >= NSTK)
        for (istk = 0; istk < NSTK-1; istk++)
            sp->val[istk] = sp->val[istk+1];

    /* push the new pixel and row index onto the stack */
    sp->val[istk].adu = pval;
    sp->val[istk].row = irow;
    sp->count = istk+1;
}

void FEPappendCti1(FEPeventRec5x5 *ev, unsigned *pImage, FEPparm *fp)
{
    struct cti1stk *cp = (struct cti1stk *)(fp->phist);
    struct _stk *sp = &cp->stk[ev->col-1];
    int ncol, nrow, istk, nstk, ntotal, istks[3], npix[3];

    /* find the number of significant precursors in each column */
    for (ncol = ntotal = 0; ncol < 3; ncol++, sp++) {
        istk = nstk = sp->count;

        if (istk > 0) {
            int doclk = fp->ex.dOclk[(ev->col+ncol-1) >> fp->colshft];
            int thresh = fp->tp.thresh[(ev->col+ncol-1) >> fp->colshft];
            int pval = -4096;

            /* find maximum scaled ADU in the column */
            for (nrow = 0; nrow < 3; nrow++) {
                int bval = ev->b[nrow][ncol];
                int ival = ev->p[nrow][ncol] - bval - doclk;
                if (bval < BIAS_BAD && ival > pval)
                    pval = ival;
            }

            /* ignore the column if below event threshold */
            if (pval >= thresh) {
                pval = PIXVAL(pval);
                /* find nearest higher precursor */
                while (--istk >= 0)
                    if (sp->val[istk].row+1 >= ev->row)
                        nstk--;
                    else if (sp->val[istk].adu >= pval)
                        break;
                if (istk < 0)
                    istk = 0;
            }
        }
        /* istks[ncol] is the index of the first stack element to report */
        istks[ncol] = istk;
        npix[ncol] = nstk - istk;
        ntotal += npix[ncol];
    }

    /* if total precursors > 15, reapportion them */
    while (ntotal > 15) {
        int dstk = (ntotal-13)/3;
        ntotal -= dstk;
        /* reduce largest column count by istk */
        if (npix[0] > npix[1])
            if (npix[0] > npix[2])
                npix[0] -= dstk;
            else
                npix[2] -= dstk;
```

```
        else if (npix[1] > npix[2])
            npix[1] -= dstk;
        else
            npix[2] -= dstk;
    }

    /* insert up to 15 ADU and row values into ev->pe[] */
    sp = &cp->stk[ev->col-1];
    for (ncol = ntotal = 0; ncol < 3; ncol++, sp++) {
        istk = istks[ncol];
        for (nstk = 0; nstk < npix[ncol]; nstk++, istk++)
            ev->pe[ntotal++] = STORE_PIX(sp->val[istk].adu,
                                         sp->val[istk].row);
    }

    /* clear remaining output elements */
    while (ntotal < 16)
        ev->pe[ntotal++] = 0;

    /* report the column counts */
    ev->pe[0] = (npix[0] << 8) | (npix[1] << 4) | npix[2];
}
```

```
/*============================================================================
//
//   $Source: /acis/h3/acisfs/configcntl/patches/ctireport1/ctireport1fep2.S,v $
//
//   MODULE NAME: ctireport1fep2.C
//
//   PURPOSE:
//     This code patches the static version of the fepSciTimed routines.
//     The purpose of the patches is as follows:
//
//      1. Increase D-cache stack length on entry into fepSciTimed to
//         locate the cti1stk structure.
//
//      2. Decrease D-cache stack length on exit from fepSciTimed.
//
//      3. Initialize cti1stk by patching over mode-dependent code in
//         FEPsciTimedInit.
//
//      4. Call FEPtestCti1, not FEPtestEvenPixel from FEPsciTimedEvent.
//
//      5. Call FEPtestCti1, not FEPtestOddPixel from FEPsciTimedEvent.
//
//      6. Force FEPtestEvenPixel to execute Te5x5 code.
//
//      7. Call FEPappendCti1, not FEPappend5x5, from FEPtestOddPixel.
//
//      8. Force FEPtestEvenPixel to execute Te5x5 code.
//
//      9. Call FEPappendCti1, not FEPappend5x5, from FEPtestOddPixel.
//
//   ASSUMPTIONS: NONE
//
//   PART NUMBER: 36-
//
//   REFERENCES:
//
//   $Log: ctireport1fep2.S,v $
//   Revision 1.3  2002/06/11 22:25:05  pgf
//   Add comments.
//
//   Revision 1.2  2001/03/20 00:11:19  pgf
//   Change stack size calculation.
//   Set first word in phist to zero.
//
//   Revision 1.1  2001/03/15 20:45:58  pgf
//   Initial skeleton test version.
//
//   Revision 1.1  2001/03/14 22:30:13  pgf
//   First very crude test version.
//
//   COPYRIGHT: Massachusetts Institute of Technology 2001
//
============================================================================*/
        .set    noreorder
        .set    nomacro
        .set    noat
####################
#
# Compute new stack size to accommodate expanded phist array.
#
####################
        .equ    stack,(64+4+1024*(4+4*24))
        .equ    stack_hi,(stack & 0xffff0000)
        .equ    stack_lo,(stack & 0x0000ffff)
```

```
        .text
####################
#
# Patch 1. fepSciTimed_lst_0000_0004 - increase stack length
#
####################

        .globl  fepSciTimed_lst_0000_0004
        .ent    fepSciTimed_lst_0000_0004

fepSciTimed_lst_0000_0004:
        li      $8,stack_hi
        ori     $8,$8,stack_lo
        .end    fepSciTimed_lst_0000_0004

####################
#
# Patch 2. fepSciTimed_lst_017c_0180 - decrease stack length
#
####################

        .globl  fepSciTimed_lst_017c_0180
        .ent    fepSciTimed_lst_017c_0180

fepSciTimed_lst_017c_0180:
        li      $8,stack_hi
        ori     $8,$8,stack_lo
        .end    fepSciTimed_lst_017c_0180

####################
#
# Patch 3. fepSciTimed_lst_02dc_02ec - initialize the precursor charge stack
#
####################

        .globl  fepSciTimed_lst_02dc_02ec
        .ent    fepSciTimed_lst_02dc_02ec

fepSciTimed_lst_02dc_02ec:
        lw      $3,120($16)             # load fp->phist
        li      $2,2                    # R2 = 2
        sw      $0,0($3)                # store zero in phist
        b       fepSciTimed_lst_02dc_02ec+0x0354-0x02dc
        move    $3,$0                   # R3 = 0
        .end    fepSciTimed_lst_02dc_02ec

####################
#
# Patch 4. fepSciTimed_lst_0604_0604 - call FEPtestCti1 not FEPtestEvenPixel
#
####################

        .globl  fepSciTimed_lst_0604_0604
        .ent    fepSciTimed_lst_0604_0604

fepSciTimed_lst_0604_0604:
        jal     FEPtestCti1
        .end    fepSciTimed_lst_0604_0604

####################
#
# Patch 5. fepSciTimed_lst_0620_0620 - call FEPtestCti1 not FEPtestOddPixel
#
####################
```

```
        .globl  fepSciTimed_lst_0620_0620
        .ent    fepSciTimed_lst_0620_0620

fepSciTimed_lst_0620_0620:
        jal     FEPtestCti1
        .end    fepSciTimed_lst_0620_0620

######################
#
# Patch 6. fepSciTimed_lst_0c0c_0c0c - force FEP_TIMED_PARM_5x5 behavior
#
######################

        .globl  fepSciTimed_lst_0c0c_0c0c
        .ent    fepSciTimed_lst_0c0c_0c0c

fepSciTimed_lst_0c0c_0c0c:
        nop
        .end    fepSciTimed_lst_0c0c_0c0c

######################
#
# Patch 7. fepSciTimed_lst_0c1c_0c1c - call FEPappendCti1 not FEPappend5x5
#
######################

        .globl  fepSciTimed_lst_0c1c_0c1c
        .ent    fepSciTimed_lst_0c1c_0c1c

fepSciTimed_lst_0c1c_0c1c:
        jal     FEPappendCti1
        .end    fepSciTimed_lst_0c1c_0c1c


######################
#
# Patch 8. fepSciTimed_lst_10ec_10ec - force FEP_TIMED_PARM_5x5 behavior
#
######################

        .globl  fepSciTimed_lst_10ec_10ec
        .ent    fepSciTimed_lst_10ec_10ec

fepSciTimed_lst_10ec_10ec:
        nop
        .end    fepSciTimed_lst_10ec_10ec

######################
#
# Patch 9. fepSciTimed_lst_10fc_10fc - call FEPappendCti1 not FEPappend5x5
#
######################

        .globl  fepSciTimed_lst_10fc_10fc
        .ent    fepSciTimed_lst_10fc_10fc

fepSciTimed_lst_10fc_10fc:
        jal     FEPappendCti1
        .end    fepSciTimed_lst_10fc_10fc
```

```
#! /usr/bin/env expect
#
# $Source: /acis/h3/acisfs/configcntl/patches/ctireport1/testsuite/smoke/runtest.tcl,v $
#
# Test of Event in Right Edge of Quadrant -- with ctireport1 patch
#

puts "Welcome to ctireport1/testsuite/smoke/runtest.tcl"

# ---- Launch the command and telemetry server processes ----
set basedir [lindex $argv 0]    ; # patch base directory
set tools [lindex $argv 1]      ; # tool directory
set patchdir [lindex $argv 2]   ; # patches under test
set quad_mode "0 \# QUAD_ABCD"  ; # desired outputRegisterMode
#set ccd_list "7 0 1 2 3 6"     ; # desired fepCcdSelect
#set last_fep 5                 ; # last FEP read out
set ccd_list "0 10 10 10 10 10" ; # desired fepCcdSelect
set last_fep 0                  ; # last FEP read out

# ---- Embed procedure library ----
source $basedir/$tools/lib/lib-exp/runtest_support.tcl

# ---- Start command pipe ----
spawn $basedir/$tools/bin/cmdclient $env(ACISSERVER)
set cmd_id $spawn_id

# ---- Start telemetry pipe ----
spawn $basedir/$tools/bin/tlmclient $env(ACISSERVER)
sleep 1

# ---- Apply patches ----
cold_boot
load_patch_list "$basedir/$tools/share/opt_tlmio.bcmd\
                 $basedir/$tools/share/opt_printswhouse.bcmd\
                 $basedir/$tools/share/opt_dearepl.bcmd\
                 $basedir/$tools/share/standard.bcmd\
                 $basedir/$tools/share/opt_smtimedlookup.bcmd\
                 $basedir/$patchdir/opt_ctireport1.bcmd"
warm_boot

# ---- Power on FEPs and CCDs ----
power_on_boards "$ccd_list"

# ---- While powering up, load the Bias Image ----
system "make bias"

# ---- Wait for FEPs to finish powering ----
expect {
    -re ".*SWSTAT_FEPMAN_ENDLOAD: $last_fep\[\r\n]*" {}
    timeout { fail "No ENDLOAD: $last_fep message" }
}

# ---- Load TE 3x3 CTI1 Mode -----
send -i $cmd_id "load 0 te 4 {
    parameterBlockId             = 0xffffffff
    fepCcdSelect                 = $ccd_list
    fepMode                      = 4 # FEP_TE_MODE_CTI1
    bepPackingMode               = 0 # BEP_TE_MODE_FAINT
    onChip2x2Summing             = 0
    ignoreBadPixelMap            = 1
    ignoreBadColumnMap           = 1
    recomputeBias                = 0
    trickleBias                  = 0
    subarrayStartRow             = 0
```

```
    subarrayRowCount             = 1023
    overclockPairsPerNode        = 8
    outputRegisterMode           = $quad_mode
    ccdVideoResponse             =     0     0     0     0     0     0
    primaryExposure              = 33
    secondaryExposure            = 0
    dutyCycle                    = 0
    fep0EventThreshold           =   100   100   100   100
    fep1EventThreshold           =   100   100   100   100
    fep2EventThreshold           =   100   100   100   100
    fep3EventThreshold           =   100   100   100   100
    fep4EventThreshold           =   100   100   100   100
    fep5EventThreshold           =   100   100   100   100
    fep0SplitThreshold           =    50    50    50    50
    fep1SplitThreshold           =    50    50    50    50
    fep2SplitThreshold           =    50    50    50    50
    fep3SplitThreshold           =    50    50    50    50
    fep5SplitThreshold           =    50    50    50    50
    fep4SplitThreshold           =    50    50    50    50
    fep5SplitThreshold           =    50    50    50    50
    lowerEventAmplitude          = 0
    eventAmplitudeRange          = 65535
    gradeSelections              = 0xffffffff 0xffffffff 0xffffffff 0xffffffff\
                                   0xffffffff 0xffffffff 0xffffffff 0xffffffff
    windowSlotIndex              = 65535
    histogramCount               = 0
    biasCompressionSlotIndex     =     3     3     1     1     1     1
    rawCompressionSlotIndex      = 1
    ignoreInitialFrames          = 2
    biasAlgorithmId              =     1     1     1     1     1     1
    biasArg0                     =     1     1     1     1     1     1
    biasArg1                     =     0     0     0     0     0     0
    biasArg2                     =     0     0     0     0     0     0
    biasArg3                     =    26    26    50    50    50    50
    biasArg4                     =    20    20    20    20    20    20
    fep0VideoOffset              =    65    65    65    65
    fep1VideoOffset              =    65    65    65    65
    fep2VideoOffset              =    65    65    65    65
    fep3VideoOffset              =    65    65    65    65
    fep4VideoOffset              =    65    65    65    65
    fep5VideoOffset              =    65    65    65    65
    deaLoadOverride              = 0
    fepLoadOverride              = 0
}
"
command_echo 1 9 "load te"

# ---- Start the Bias Run ----
send -i $cmd_id "start 0 te bias 4\n"
command_echo 1 15 "start bias run"
set timeout 3600
wait_stop_science

# ---- Load the Event Image ----
system "make image"

# ---- Start the Science Run ----
send -i $cmd_id "start 0 te 4\n"
command_echo 1 14 "start science run"

# ---- Wait for Exposure Records ----
set events 0

expect {
```

```
    -re "exposureTe\[^\r]*\
        fepId=(\[0-9])\[^\r]*\
        exposureNumber=(\[0-9]+)\[^\r]*\
        eventsSent=(\[0-9]+)\[^\r]*\
        discardGrade=\[0-9]+\[\r\n]*" {

        set fep $expect_out(1,string)
        set expo $expect_out(2,string)
        set events $expect_out(3,string)

        if {$fep == $last_fep && $expo > 10 && $events > 0} {
            send -i $cmd_id "stop 0 science\n"
        } else {
            exp_continue
        }
    }

    -re "scienceReport.*\[\r\n]*" {
        fail "Immediate Science Report"
    }

    timeout {
        fail "No exposure record"
    }
}

wait_stop_science

# ---- Unpack telemetry into event lists ----
system psci -a -B -l foo -h /nfs/acis/h2/pgf/acis/psci/huff.dat pkts.raw
system cut -c21- foo.2.$last_fep.erv*.txt | diff ../erv5.txt -
eval exec rm [glob foo.*]

# ---- Report fate ----
pass "$events events received from FEP_$fep"
```

```
/* ============================================================
 *
 * $$Source: /acis/h3/acisfs/configcntl/patches/ctireport2/ctireport2.C,v $$
 *
 * Patch Name: ctireport2
 *
 * Description:
 *
 *   This patch implements a variant of timed-exposure 3x3 faint event
 *   mode in which the presence of precursor charge in each of the
 *   three columns that can contribute to each event is encoded in the
 *   low-order bits of three of the corner pixels.
 *
 *   FEP patches are loaded after the default code by two additional
 *   calls to fepManager.loadRunProgram from
 *   Test3_SmTimedExposure::setupCti2Fep. Once loaded, the FEPs are
 *   marked as having been reset, thereby causing the following run to
 *   reload their default code.
 *
 *   Once loaded, the FEPs are marked as having been reset, thereby
 *   causing the following run to reload their default code.
 *
 *   The ctireport2.C file defines and implements the following:
 *
 *     Test3_SmTimedExposure    - a subclass of SmTimedExposure which
 *                                provides replacement functions for the
 *                                terminate method and a new setupCti2Fep
 *                                method that is patched into the
 *                                smTimedSetupFep array
 *
 *       Test3_SmTimedExposure  - this constructor is provided to avoid
 *                                compiler/linker complaints. Never invoked.
 *
 *       ~Test3_SmTimedExposure - this destructor is provided to avoid
 *                                compiler/linker complaints. Never invoked.
 *
 *       setupCti2Fep           - this function initializes the event
 *                                packet headers and then makes 3 calls
 *                                to fepManager.loadRunProgram to load the
 *                                default FEP code, the new subroutines,
 *                                and finally, the inline patches.
 *
 *       terminate              - this function is called at the end of
 *                                the science run. After performing the
 *                                default actions, it calls
 *                                fepManager.resetFeps to ensure that
 *                                the default FEP code is reloaded before
 *                                the FEPs are used again.
 *
 * Note:
 *   This patch requires that the smtimedlookup patch must also be loaded.
 *
 * References:
 *    Refer to the 1.5 release of filesscience/smtimedexposure.C and to
 *    the method of FEP patching used in the cc3x3 patch.
 *
 * $$Log: ctireport2.C,v $
 * $Revision 1.5  2002/06/12 03:56:39  pgf
 * $Add comments and clean up the code.
 * $
 * $Revision 1.4  2001/04/06 21:04:22  pgf
 * $Define the constructor.
 * $
 * $Revision 1.3  2001/04/06 20:47:10  pgf
```

```
 * $Add dummy constructor.
 * $
 * $Revision 1.2  2001/04/06 03:11:32  pgf
 * $Fix bug in global names.
 * $
 * $Revision 1.1  2001/04/05 22:43:43  pgf
 * $Initial test release.
 * $$
 * ========================================================== */

#ifndef private
#define private public
#endif

#ifndef protected
#define protected public
#endif

#include "ipcl/interface.h"
#include "filesscience/smtimedexposure.H"
#include "filesswhouse/swhousekeeper.H"

extern SmTimedExposure smTimedExposure;

// ###################################################
// ###################################################

class Test3_SmTimedExposure : public SmTimedExposure
{
public:
    Test3_SmTimedExposure();
    ~Test3_SmTimedExposure();
    void    setupCti2Fep(FepId fep, FEPparmBlock&  fepblock);
    void    terminate();
};

extern const unsigned short ctireport2fep1[];
FepProgram* ctireport2fep1Hanger = (FepProgram*) ctireport2fep1;
extern const FepProgram* ctireport2fep2Hanger;
extern const FepProgram* defaultFepLoadHanger;

// ---------------------------------------------------
Test3_SmTimedExposure::Test3_SmTimedExposure()
// ---------------------------------------------------
{
};

// ---------------------------------------------------
Test3_SmTimedExposure::~Test3_SmTimedExposure()
// ---------------------------------------------------
{
};

// ---------------------------------------------------
void Test3_SmTimedExposure::setupCti2Fep(FepId fep, FEPparmBlock&  fepblock)
// ---------------------------------------------------
{
    // ---- Supply fixed output packet header fields ----
    fepblock.type = FEP_TIMED_PARM_3x3;
    fepblock.nhist = 0;

    // ---- Load default FEP code and inline and subroutine patches ----
    fepManager.loadRunProgram (fep, defaultFepLoadHanger);
    fepManager.loadRunProgram (fep, ctireport2fep1Hanger);
```

```
    fepManager.loadRunProgram (fep, ctireport2fep2Hanger);
}

// ------------------------------------------------
void Test3_SmTimedExposure::terminate()
// ------------------------------------------------
{
    // ---- Flush all exposure records ----
    flushProcesses();

    // ---- Form and issue report ----
    Tf_Science_Report report;

    if (waitForPkt(report) == BoolFalse)
    {
        fepManager.resetFeps();
        return;
    }

    setupScienceReport(report);
    report.post();

    // ---- Reset the FEPs to force reload on next run ----
    fepManager.resetFeps();
}
```

```
/* ============================================================
 *
 * $$Source: /acis/h3/acisfs/configcntl/patches/ctireport2/ctireport2fep1.c,v $$
 *
 * Patch Name: ctireport2
 *
 * Description:
 *
 *   This file supplies additional FEP routines, FEPtestCti2 and
 *   FEPappendCti2, that together cause the FEP to report the
 *   presence of precursor charge in any of the 3 CCD columns that
 *   span each candidate event.
 *
 *   FEPtestCti2 is called from an inline patch (ctireport2fep2.S)
 *   in place of the default FEPtestOddPixel or FEPtestEvenPixel
 *   routines. When a threshold crossing is detected, it clears the
 *   cti2stk array (if this is a new frame), calls FEPtestOddPixel
 *   or FEPtestEvenPixel, and marks cti2stk to indicate that this
 *   column contains charge.
 *
 *   FEPappendCti2 is called by the patched FEP code instead of the
 *   original FEPappend5x5. It finds the maximum of the 4 corner pixels
 *   of the event that is being reported. Then it determines whether
 *   any of the three contrinuting columns contained precursor charge.
 *   Finally, it encodes this information in the low order bytes of
 *   the three smallest corner pixels. (Since the low-order bit of
 *   each corner pixel may be replaced, only the 11 high-order bits
 *   are compared when determining the maximum value).
 *
 * Note:
 *  This patch requires that the smtimedlookup patch must also be loaded.
 *
 * References:
 *   Refer to the 1.5 release of filesscience/smtimedexposure.C and to
 *   the method of FEP patching used in the cc3x3 patch.
 *
 * $$Log: ctireport2fep1.c,v $
 * $Revision 1.3  2003/02/17 02:00:30  pgf
 * $Change variable names to improve readability.
 * $
 * $Revision 1.2  2002/06/12 03:56:41  pgf
 * $Add comments and clean up the code.
 * $$
 * ============================================================ */
#include "fepCtl.h"

extern void FEPtestEvenPixel(unsigned irow, unsigned icol, FEPparm *fp);
extern void FEPtestOddPixel(unsigned irow, unsigned icol, FEPparm *fp);

struct cti2stk {
  int   expnum;                 /* exposure number -- must be first element */
  unsigned stk[1024][2];        /* two per column */
};

void FEPtestCti2(unsigned irow, unsigned icol, FEPparm *fp)
{
  struct cti2stk *cp = (struct cti2stk *)(fp->phist);
  int ncol;

  /* on new frame, clear stack counters */
  if (cp->expnum < fp->ex.expnum) {
    cp->expnum = fp->ex.expnum;
    for (ncol = 0; ncol < 1024; ncol++)
      cp->stk[ncol][0] = cp->stk[ncol][1] = 0;
```

```
  }

  /* find and report any event */
  if (icol & 1) {
    FEPtestOddPixel(irow, icol, fp);
  } else {
    FEPtestEvenPixel(irow, icol, fp);
  }

  /* mark the column */
  cp->stk[icol][1] = cp->stk[icol][0];
  cp->stk[icol][0] = irow+1;
}

void FEPappendCti2(FEPeventRec5x5 *ev, unsigned *pImage, FEPparm *fp)
{
  struct cti2stk *cp = (struct cti2stk *)(fp->phist);
  int nrow, ncol, rrow, rcol, chg[3], ichg;

  /* determine whether each column has precursor charge */
  for (ncol = 0; ncol < 3; ncol++) {
    int *pp = cp->stk[ev->col+ncol-1];
    chg[ncol] = (pp[0] && pp[0] < ev->row) || (pp[1] && pp[1] < ev->row);
  }

  /* find maximum corner pixel */
  for (nrow = rrow = rcol = 0; nrow < 3; nrow += 2) {
    for (ncol = 0; ncol < 3; ncol += 2) {
      if ((ev->p[nrow][ncol] & 0xffe) > (ev->p[rrow][rcol] & 0xffe)) {
        rrow = nrow;
        rcol = ncol;
      }
    }
  }

  /* insert precursor charge info into corner pixels */
  for (nrow = ichg = 0; nrow < 3; nrow += 2) {
    for (ncol = 0; ncol < 3; ncol += 2) {
      if (nrow != rrow || ncol != rcol) {
        ev->p[nrow][ncol] = (ev->p[nrow][ncol] & 0xffe) | (chg[ichg++] & 1);
      }
    }
  }
}
```

```
/*=========================================================================
//
//   $Source: /acis/h3/acisfs/configcntl/patches/ctireport2/ctireport2fep2.S,v $
//
//   MODULE NAME: ctireport2fep2
//
//   PURPOSE:
//     This code patches the static version of the fepSciTimed routines.
//     The purpose of the patches is as follows:
//
//      1. Increase D-cache stack length on entry into fepSciTimed to
//         locate the cti2stk structure.
//
//      2. Decrease D-cache stack length on exit from fepSciTimed.
//
//      3. Initialize cti2stk by patching over mode-dependent code in
//         FEPsciTimedInit.
//
//      4. Call FEPtestCti2, not FEPtestEvenPixel from FEPsciTimedEvent.
//
//      5. Call FEPtestCti2, not FEPtestOddPixel from FEPsciTimedEvent.
//
//      6. Call FEPappendCti2, not FEPappend5x5, from FEPtestEvenPixel.
//
//      7. Call FEPappendCti2, not FEPappend5x5, from FEPtestOddPixel.
//
//   ASSUMPTIONS: NONE
//
//   PART NUMBER: 36-
//
//   REFERENCES:
//
//   $Log: ctireport2fep2.S,v $
//   Revision 1.2  2002/06/12 03:56:41  pgf
//   Add comments and clean up the code.
//
//   Revision 1.1  2001/04/05 22:43:47  pgf
//   Initial test release.
//
//   COPYRIGHT: Massachusetts Institute of Technology 2001
//
=========================================================================*/
        .set    noreorder
        .set    nomacro
        .set    noat
#####################
#
# Compute new stack size to accommodate expanded phist array.
#
#####################
        .equ    stack,(64+4+1024*2*4)
        .equ    stack_hi,(stack & 0xffff0000)
        .equ    stack_lo,(stack & 0x0000ffff)
        .text
#####################
#
# Patch 1. fepSciTimed_lst_0000_0004 - increase stack length
#
#####################

        .globl  fepSciTimed_lst_0000_0004
        .ent    fepSciTimed_lst_0000_0004

fepSciTimed_lst_0000_0004:
```

```
        li      $8,stack_hi
        ori     $8,$8,stack_lo
        .end    fepSciTimed_lst_0000_0004

####################
#
# Patch 2. fepSciTimed_lst_017c_0180 - decrease stack length
#
####################

        .globl  fepSciTimed_lst_017c_0180
        .ent    fepSciTimed_lst_017c_0180

fepSciTimed_lst_017c_0180:
        li      $8,stack_hi
        ori     $8,$8,stack_lo
        .end    fepSciTimed_lst_017c_0180

####################
#
# Patch 3. fepSciTimed_lst_02dc_02ec - initialize the precursor charge stack
#
####################

        .globl  fepSciTimed_lst_02dc_02ec
        .ent    fepSciTimed_lst_02dc_02ec

fepSciTimed_lst_02dc_02ec:
        lw      $3,120($16)             # load fp->phist
        li      $2,2                    # R2 = 2
        sw      $0,0($3)                # store zero in phist
        b       fepSciTimed_lst_02dc_02ec+0x0354-0x02dc
        move    $3,$0                   # R3 = 0
        .end    fepSciTimed_lst_02dc_02ec

####################
#
# Patch 4. fepSciTimed_lst_0604_0604 - call FEPtestCti2 not FEPtestEvenPixel
#
####################

        .globl  fepSciTimed_lst_0604_0604
        .ent    fepSciTimed_lst_0604_0604

fepSciTimed_lst_0604_0604:
        jal     FEPtestCti2
        .end    fepSciTimed_lst_0604_0604

####################
#
# Patch 5. fepSciTimed_lst_0620_0620 - call FEPtestCti2 not FEPtestOddPixel
#
####################

        .globl  fepSciTimed_lst_0620_0620
        .ent    fepSciTimed_lst_0620_0620

fepSciTimed_lst_0620_0620:
        jal     FEPtestCti2
        .end    fepSciTimed_lst_0620_0620

####################
#
# Patch 6. fepSciTimed_lst_0c04_0c18 - call FEPappendCti2
```

```
#
#####################

        .globl  fepSciTimed_lst_0c04_0c18
        .ent    fepSciTimed_lst_0c04_0c18

fepSciTimed_lst_0c04_0c18:
        addu    $4,$sp,16
        move    $5,$22
        jal     FEPappendCti2
        move    $6,$17
        b       fepSciTimed_lst_0c04_0c18+0x0c38-0x0c04
        li      $2,2
        .end    fepSciTimed_lst_0c04_0c18

#####################
#
# Patch fepSciTimed_lst_10e4_10f8 - call FEPappendCti2
#
#####################

        .globl  fepSciTimed_lst_10e4_10f8
        .ent    fepSciTimed_lst_10e4_10f8

fepSciTimed_lst_10e4_10f8:
        addu    $4,$sp,16
        move    $5,$21
        jal     FEPappendCti2
        move    $6,$17
        b       fepSciTimed_lst_10e4_10f8+0x1118-0x10e4
        li      $2,2
        .end    fepSciTimed_lst_10e4_10f8
```

```
#! /usr/bin/env expect
#
# $Source: /acis/h3/acisfs/configcntl/patches/ctireport2/testsuite/smoke/runtest.tcl,v $
#
# Test of Event in Right Edge of Quadrant -- with ctireport2 patch
#

puts "Welcome to ctireport2/testsuite/smoke/runtest.tcl"

# ---- Launch the command and telemetry server processes ----
set basedir [lindex $argv 0]    ; # patch base directory
set tools [lindex $argv 1]      ; # tool directory
set patchdir [lindex $argv 2]   ; # patches under test
set quad_mode "0 \# QUAD_ABCD"  ; # desired outputRegisterMode
set ccd_list "7 0 1 2 3 6"      ; # desired fepCcdSelect
set last_fep 5                  ; # last FEP read out

# ---- Embed procedure library ----
source $basedir/$tools/lib/lib-exp/runtest_support.tcl

# ---- Start command pipe ----
spawn $basedir/$tools/bin/cmdclient $env(ACISSERVER)
set cmd_id $spawn_id

# ---- Start telemetry pipe ----
spawn $basedir/$tools/bin/tlmclient $env(ACISSERVER)
sleep 1

# ---- Apply patches ----
cold_boot
load_patch_list "$basedir/$tools/share/opt_tlmio.bcmd\
                 $basedir/$tools/share/opt_printswhouse.bcmd\
                 $basedir/$tools/share/opt_dearepl.bcmd\
                 $basedir/$tools/share/standard.bcmd\
                 $basedir/$tools/share/opt_smtimedlookup.bcmd\
                 $basedir/$patchdir/opt_ctireport2.bcmd"
warm_boot

# ---- Power on FEPs and CCDs ----
power_on_boards "$ccd_list"

# ---- While powering up, load the Bias Image ----
system "make bias"

# ---- Wait for FEPs to finish powering ----
expect {
    -re ".*SWSTAT_FEP_EXECMEM: $last_fep\[\r\n]" { sleep 10 }
    timeout {}
}

# ---- Load TE 3x3 CTI1 Mode -----
send -i $cmd_id "load 0 te 4 {
    parameterBlockId             = 0xffffffff
    fepCcdSelect                 = $ccd_list
    fepMode                      = 2 # FEP_TE_MODE_EV3x3
    bepPackingMode               = 0 # BEP_TE_MODE_FAINT
    onChip2x2Summing             = 0
    ignoreBadPixelMap            = 1
    ignoreBadColumnMap           = 1
    recomputeBias                = 0
    trickleBias                  = 0
    subarrayStartRow             = 0
    subarrayRowCount             = 1023
    overclockPairsPerNode        = 8
```

```
    outputRegisterMode          = $quad_mode
    ccdVideoResponse            =     0     0     0     0     0     0
    primaryExposure             = 33
    secondaryExposure           = 0
    dutyCycle                   = 0
    fep0EventThreshold          =   100   100   100   100
    fep1EventThreshold          =   100   100   100   100
    fep2EventThreshold          =   100   100   100   100
    fep3EventThreshold          =   100   100   100   100
    fep4EventThreshold          =   100   100   100   100
    fep5EventThreshold          =   100   100   100   100
    fep0SplitThreshold          =    50    50    50    50
    fep1SplitThreshold          =    50    50    50    50
    fep2SplitThreshold          =    50    50    50    50
    fep3SplitThreshold          =    50    50    50    50
    fep5SplitThreshold          =    50    50    50    50
    fep4SplitThreshold          =    50    50    50    50
    fep5SplitThreshold          =    50    50    50    50
    lowerEventAmplitude         = 0
    eventAmplitudeRange         = 65535
    gradeSelections             = 0xffffffff 0xffffffff 0xffffffff 0xffffffff\
                                  0xffffffff 0xffffffff 0xffffffff 0xffffffff
    windowSlotIndex             = 65535
    histogramCount              = 0
    biasCompressionSlotIndex    =     3     3     1     1     1     1
    rawCompressionSlotIndex     = 0
    ignoreInitialFrames         = 2
    biasAlgorithmId             =     1     1     1     1     1     1
    biasArg0                    =     1     1     1     1     1     1
    biasArg1                    =     0     0     0     0     0     0
    biasArg2                    =     0     0     0     0     0     0
    biasArg3                    =    26    26    50    50    50    50
    biasArg4                    =    20    20    20    20    20    20
    fep0VideoOffset             =    65    65    65    65
    fep1VideoOffset             =    65    65    65    65
    fep2VideoOffset             =    65    65    65    65
    fep3VideoOffset             =    65    65    65    65
    fep4VideoOffset             =    65    65    65    65
    fep5VideoOffset             =    65    65    65    65
    deaLoadOverride             = 0
    fepLoadOverride             = 0
}
"
command_echo 1 9 "load te"

# ---- Start the Bias Run ----
send -i $cmd_id "start 0 te bias 4\n"
command_echo 1 15 "start bias run"
set timeout 360
wait_stop_science

# ---- Load the Event Image ----
system "make image"

# ---- Start the Science Run ----
send -i $cmd_id "start 0 te 4\n"
command_echo 1 14 "start science run"

# ---- Wait for Exposure Records ----
set events 0

expect {
    -re "exposureTe\[^\r]*\
        fepId=(\[0-9])\[^\r]*\
```

```
        exposureNumber=(\[0-9]+)\[^\r]*\
        eventsSent=(\[0-9]+)\[^\r]*\
        discardGrade=\[0-9]+\[\r\n]*" {

        set fep $expect_out(1,string)
        set expo $expect_out(2,string)
        set events $expect_out(3,string)

        if {$fep == $last_fep && $events > 0} {
            send -i $cmd_id "stop 0 science\n"
        } else {
            exp_continue
        }
    }

    timeout {
        fail "No exposure record"
    }
}

wait_stop_science

# ---- Unpack telemetry into event lists ----
system psci -a -B -l foo pkts.raw
system cut -c21- foo.2.$last_fep.erv.txt | sed 47q | diff ../erv.txt -
eval exec rm [glob foo.*]

# ---- Report fate ----
pass "$events events received from FEP_$fep"
```

```
/* ============================================================
 *
 * $$Source: /acis/h3/acisfs/configcntl/patches/eventhist/hamming.C,v $$
 *
 * Patch Name: Hamming Class Implementation File
 *
 * Description:
 *  This implements the Hamming error detection/correction
 *  class used for Event Histogram processing mode.
 *
 * References:
 *
 * $$Log: hamming.C,v $
 * $Revision 1.4  1999/04/20 12:42:41  jimf
 * $Add correction counter argument
 * $
 * Revision 1.3  1998/11/10  20:37:20  jimf
 * Assign partnumbers and small documentation cleanup.
 *$
 *
 * ============================================================ */

#include "hamming.H"

// Hamming parity mask for 26 data bits, 5 parity bits
static const int hammingArray[HAM_P] = {
  0x06aaad5b, 0x0b33366d, 0x13c3c78e, 0x23fc07f0, 0x43fff800
};

// map syndrome values to bit error offsets
static const char syndromeBitOffset[1<<HAM_P] = {
  0, 0, 0, 1, 0, 2, 3, 4, 0, 5, 6, 7, 8, 9, 10, 11, 0, 12,
  13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26
};

// compute generalized parity of cell
int Hamming::syndrome(int cell)
{
  int syn = 0;

  for (int ii = 0; ii < HAM_P; ii++) {
    int jj = hammingArray[ii] & cell;
    jj ^= jj >> 16;
    jj ^= jj >> 8;
    jj ^= jj >> 4;
    jj ^= jj >> 2;
    jj ^= jj >> 1;
    syn |= (jj & 1) << ii;
  }

  return syn;
}

// correct errors, increment the argument, add parity bits
int Hamming::increment(int cell, unsigned& error_count)
{
  int syn = syndrome(cell);

  if (syn && syndromeBitOffset[syn])
  {
    cell ^= 1 << (syndromeBitOffset[syn]-1);
    error_count++;
  }
```

```
  cell = (cell+1) & HAM_DX;
  cell |= syndrome(cell) << HAM_D;

  return cell;
}
```

```
/* ============================================================
 *
 * $$Source: /acis/h3/acisfs/configcntl/patches/eventhist/hamming.H,v $$
 *
 * Patch Name: Hamming Class Definition File
 *
 * Description:
 *  This defines the Hamming error detection/correction
 *  class used for Event Histogram processing mode.
 *
 * References:
 *
 * $$Log: hamming.H,v $
 * $Revision 1.3  1999/04/20 12:42:40  jimf
 * $Add correction counter argument
 * $
 * Revision 1.2  1998/11/10  20:37:22  jimf
 * Assign partnumbers and small documentation cleanup.
 *$
 *
 * ============================================================ */

#ifndef HAM_D
#define HAM_D   26              // number of data bits
#define HAM_P   5               // number of parity bits
#define HAM_DX  ((1<<HAM_D)-1)  // data bit mask

class Hamming
{
  int syndrome(int cell);       // return syndrome of cell
  public:
    int increment(int cell, unsigned& error_count); // return incremented cell value
};
#endif HAM_D
```

```
/*==========================================================================
 *
 *   $Source: /acis/h3/acisfs/configcntl/patches/eventhist/Attic/interface.h,v $
 *
 *   MODULE NAME: ACIS Interface Include File
 *
 *   PURPOSE: Provides all enuerations and defines needed by the ground
 *            to control ACIS, and to interpret its data. This file
 *            is used in conjunction with the ACIS IP&CL Software Structres
 *            inputs to the AXAF IP&CL.
 *
 *   ASSUMPTIONS:
 *
 *   PART NUMBER:
 *
 *   REFERENCES:
 *            "ACIS IP&CL Software Structure Definitions" 36-53202.0204 Rev. F+
 *
 *   $Log: interface.h,v $
 *   Revision 1.1  1998/01/02 18:25:02  jimf
 *   Use distinct telemetry tags for Event Histogram Mode
 *
 * Revision 1.24  1997/06/20  17:59:14  jimf
 * Add software housekeepign for jitter failure and aborted bias trickle
 *
 * Revision 1.23  1997/04/05  03:05:06  jimf
 * Reserve tlm tag 45. Unused LED codes are now listed as spare
 *
 * Revision 1.22  1997/03/31  20:57:30  jimf
 * Added 5x5-specific exposure record tag
 *
 * Revision 1.21  1997/03/31  15:18:46  jimf
 * CCD Safety: Limit Check DEA System Configuration Inputs
 *
 * Revision 1.20  1997/02/07  14:06:43  jimf
 * ECO 36-855: Added 5x5 Mode
 *
 * Revision 1.19  1997/01/30  20:20:54  jimf
 * Make Graded Mean codes an enum
 *
 * Revision 1.18  1997/01/30  18:43:43  jimf
 * ECO: 36-846 Define special codes for corner pulse height mean in graded mode
 *
 * Revision 1.17  1997/01/07  20:24:00  jimf
 * Rev. G IP&CL - Added TTAG_RESERVED, DEAHOUSE_VALUE_INVALID, and argument to SWSTAT_SCI_EX
PSTART_FEPTIME
 *
 * Revision 1.16  1996/12/19  22:31:14  jimf
 * Added some housekeeping. Added margin definition for housekeeping buffers
 *
 * Revision 1.15  1996/12/03  14:45:08  jimf
 * Address SPR 47: Add statistic to report FEP Timestamp out-of-bounds
 *
 * Revision 1.14  1996/11/04  15:44:47  jimf
 * Updated definitions to reflect preliminary DEA/DPA ICD Rev. B
 *
 * Revision 1.13  1996/11/04  14:50:00  jimf
 * Added FEP_ERR_BAD_MBOX_STATE and move defines from fep/fepBep.h and fep/fepCtl.h
 *
 * Revision 1.12  1996/10/23  12:56:57  jimf
 * Added more SwHousekeeping
 *
 * Revision 1.11  1996/10/16  12:44:01  jimf
 * Added DEA board error report
```

```
 *
 * Revision 1.10  1996/10/03  23:22:29  jimf
 * Added loadinvalid housekeeping report
 *
 * Revision 1.9  1996/09/26  17:23:01  jimf
 * Add statistic for unrecognized FEP record type
 *
 * Revision 1.8  1996/09/25  18:32:53  danh
 * Added codes for 3x3 row/column corruption and ring buffer index corruptions
 *
 * Revision 1.7  1996/09/18  13:00:39  jimf
 * Added FEP Bus error statistic and fatal error codes
 *
 * Revision 1.6  1996/09/11  13:36:03  jimf
 * Incorporate ECO 36-736 IP&CL Rev. E 1.19. Replaced assert() with ASSERT. Added some softw
are housekeeping and fatal errors.
 *
 * Revision 1.5  1996/09/05  00:59:31  jimf
 * Re. DPA/DEA ICD - Rev. B ECO 36-735: Added Interface Housekeeping ADC Queries. Removed sp
ecial Focal Temp. Query.
 *
 * Revision 1.4  1996/08/26  21:10:17  jimf
 * Added call to set reason when bias-only run completes
 *
 * Revision 1.3  1996/08/16  19:57:02  jimf
 * Added TE Dump and Bad Mode software housekeeping codes
 *
 * Revision 1.2  1996/08/13  21:58:56  jimf
 * Moved all interface definitions into acis_h/interface.h
 *
 * Revision 1.1  1996/08/13  20:07:40  jimf
 * Master interface definition file
 *
 *
 *   COPYRIGHT: Massachusetts Institute of Technology 1996
 *
 ========================================================================*/

#ifndef INTERFACE_H
#define INTERFACE_H 1
```

```
/* ------------------------------------------------------------------
 * Command Opcode Definitions
 * ------------------------------------------------------------------ */

/* ---- Define command opcodes ---- */
enum CmdOpcode
{
    CMDOP_UNUSED,               /* Unused Opcode */

    /* ---- Load from Uplink Commands ---- */
    CMDOP_START_UPLOAD,         /* Start Load From Uplink */
    CMDOP_CONTINUE_UPLOAD,      /* Continue Load From Uplink */

    /* ---- Memory Read/Write/Execute Commands ---- */
    CMDOP_READ_BEP,             /* Read BEP Memory */
    /* WRITE/EXEC BEP moved near end of table */

    CMDOP_READ_FEP,             /* Read FEP Memory */
    CMDOP_WRITE_FEP,            /* Write FEP Memory */
    CMDOP_EXEC_FEP,             /* Execute FEP Memory */

    CMDOP_READ_PRAM,            /* Read DEA PRAM */
    /* WRITE PRAM moved near end of table */

    CMDOP_READ_SRAM,            /* Read DEA SRAM */
    /* WRITE SRAM moved near end of table */

    /* ---- Load Parameter Block Commands ---- */
    CMDOP_LOAD_TE,              /* Load Timed Exposure Block */
    CMDOP_LOAD_CC,              /* Load Continuous Clocking Block */
    CMDOP_LOAD_2D,              /* Load 2D Window List */
    CMDOP_LOAD_1D,              /* Load 1D Window List */
    CMDOP_LOAD_DEA,             /* Load DEA Housekeeping Block */

    /* ---- Start/Stop Run Commands ---- */
    CMDOP_START_TE,             /* Start Timed Exposure Run */
    CMDOP_BIAS_TE,              /* Start Timed Exposure Bias Run */

    CMDOP_START_CC,             /* Start Continuous Clocking Run */
    CMDOP_BIAS_CC,              /* Start Continuous Clocking Bias Run */

    CMDOP_START_DEA,            /* Start DEA Housekeeping Run */

    CMDOP_STOP_SCIENCE,         /* Stop Science Run */
    CMDOP_STOP_DEA,             /* Stop DEA Housekeeping Run */

    /* ---- Patch Commands ---- */
    CMDOP_ADD_PATCH,            /* Add Patches */
    CMDOP_REMOVE_PATCH,         /* Remove Patches */

    /* ---- System Configuration/Bad Pixel List Commands ---- */
    CMDOP_ADD_BAD_PIXEL,        /* Add Bad Pixel */
    CMDOP_RESET_BAD_PIXEL,      /* Reset Bad Pixel List */
    CMDOP_DUMP_BAD_PIXELS,      /* Dump Bad Pixel List */

    CMDOP_ADD_BAD_TE_COL,       /* Add Bad Timed Exposure Column  */
    CMDOP_RESET_BAD_TE_COL,     /* Reset Bad Timed Exposure Column List */
    CMDOP_DUMP_BAD_TE_COL,      /* Dump Bad Timed Exposure Column List */

    CMDOP_ADD_BAD_CC_COL,       /* Add Bad Cont. Clocking Column */
    CMDOP_RESET_BAD_CC_COL,     /* Reset Bad Cont. Clocking Column List */
    CMDOP_DUMP_BAD_CC_COL,      /* Dump Bad Cont. Clocking Column List */
```

```
    CMDOP_CHANGE_SYS_ENTRY,     /* Change System Configuration Settings */
    CMDOP_DUMP_SYS_CONFIG,      /* Dump System Configuration Settings */

    CMDOP_DUMP_PATCHLIST,       /* Dump PatchList */
    CMDOP_DUMP_HUFFMAN,         /* Dump Huffman Table */
    CMDOP_DUMP_TE_SLOTS,        /* Dump Timed Exposure Parameter Block Slots */
    CMDOP_DUMP_CC_SLOTS,        /* Dump Continuous Clocking Block Slots */
    CMDOP_DUMP_2D_SLOTS,        /* Dump 2D Window Parameter Block Slots */
    CMDOP_DUMP_1D_SLOTS,        /* Dump 1D Window Parameter Block Slots */
    CMDOP_DUMP_DEA_SLOTS,       /* Dump DEA Housekeeping Parameter Block Slots */

    /* ---- Two-bit difference codes ---- */
    CMDOP_WRITE_BEP = 0xc0,     /* Write BEP Memory */
    CMDOP_EXEC_BEP = 0xc3,      /* Execute BEP Memory */

    CMDOP_WRITE_PRAM = 0xcc,    /* Write DEA PRAM */
    CMDOP_WRITE_SRAM = 0xf0,    /* Write DEA SRAM */

    /* ---- Miscellenenous Codes ----- */
    CMDOP_COUNT,                /* Total # of Command Opcodes */
    CMDOP_LASTID = CMDOP_COUNT - 1, /* Value of last opcode */

    CMDOP_INVALID = ~0          /* Code for an invalid opcode */
};
```

```
/* ---------------------------------------------------------------------
 * Command Execution Result Code Definitions
 * --------------------------------------------------------------------- */

/* ---- Define command result codes ---- */
enum CmdResult
{
   CMDRESULT_UNUSED,            /* Unused response code */

   CMDRESULT_OK,                /* Command Successfully Dispatched */
   CMDRESULT_NO_HANDLER,        /* No handler for command opcode */
   CMDRESULT_BUSY,              /* Target of command is busy */
   CMDRESULT_BAD_ARGUMENT,      /* Command contains a bad parameter */
   CMDRESULT_CORRUPT_DEFAULT,   /* Selected Parameter Blk corrupt, use default */
   CMDRESULT_CORRUPT_IDLE,      /* Parameter Blk and Default corrupt, no run */
   CMDRESULT_TABLE_FULL,        /* Pixel/Column/Patch Table Full */
   CMDRESULT_TABLE_EMPTY,       /* Patch Table Empty */
   CMDRESULT_INVALID_PKT,       /* Command Packet Header corrupt */
   CMDRESULT_BOARD_OFF,         /* Selected Board has no power */
   CMDRESULT_BOARD_RESET,       /* Select Board is Reset */
   CMDRESULT_STORE_ERROR,       /* Error while storing parameter block */

   CMDRESULT_INHIBITED,         /* Run was inhibited when stop/start invoked */
   CMDRESULT_CLOBBERED,         /* Operation was aborted by start of new op */

   CMDRESULT_ITEM_CLIPPED,      /* Accepted input value was clipped to limit */

   CMDRESULT_COUNT,             /* Total number of result codes */
   CMDRESULT_LASTID  = CMDRESULT_COUNT - 1
};
```

```
/* ---------------------------------------------------------------------
 * Telemetry Synch Code and Format Tag Definitions
 * --------------------------------------------------------------------- */

/* ---- Define Telemetry Packet Synch Code ---- */
enum TpktSynch
{
    SYNCH_PATTERN = 0x736f4166
};

/* ---- Define Telemetry format tag codes ---- */
enum TlmFormatTag
{
    TTAG_UNUSED,                /* Unused format tag */

    /* ---- Memory Command Responses ---- */
    TTAG_READ_BEP,              /* Read Back End Memory */
    TTAG_READ_FEP,              /* Read Front End Memory */
    TTAG_READ_SRAM,             /* Read SRAM */
    TTAG_READ_PRAM,             /* Read Pram */
    TTAG_EXEC_BEP,              /* Execute Back End Subroutine */
    TTAG_EXEC_FEP,              /* Execute Front End Subroutine */

    /* ---- Command Echoes ---- */
    TTAG_CMD_ECHO,              /* Echoed command */

    /* ---- Miscellenious Housekeeping ---- */
    TTAG_STARTUP,               /* Startup Message */
    TTAG_FATAL,                 /* Fatal Error Message */
    TTAG_SW_HOUSE,              /* Software Housekeeping */
    TTAG_DEA_HOUSE,             /* DEA Housekeeping */

    /* ---- Parameter Dumps ---- */
    TTAG_DUMP_TE,               /* Timed Exposure Parameters + 2D Window */
    TTAG_DUMP_CC,               /* Cont. Clocking Parameters + 1D Windows */

    /* ---- Miscelleneous Science ---- */
    TTAG_SCI_TE_BIAS,           /* Timed Exposure Bias Map */
    TTAG_SCI_REPORT,            /* Science Run Report */

    /* ---- Timed Exposure Science ---- */
    TTAG_SCI_TE_REC_RAW,        /* Timed Exposure Raw Mode Exposure Hdr */
    TTAG_SCI_TE_DAT_RAW,        /* Timed Exposure Raw Mode Data */
    TTAG_SCI_TE_REC_HIST,       /* Timed Exposure Histogram Mode Exp. Hdr */
    TTAG_SCI_TE_DAT_HIST,       /* Timed Exposure Histogram Mode Data */
    TTAG_SCI_TE_REC_FAINT,      /* Timed Exposure Faint Mode Exp. Hdr */
    TTAG_SCI_TE_DAT_FAINT,      /* Timed Exposure Faint Mode Data */
    TTAG_SCI_TE_REC_FAINTB,     /* Timed Exposure Faint-Bias Mode Exp. Hdr */
    TTAG_SCI_TE_DAT_FAINTB,     /* Timed Exposure Faint-Bias Mode Data */
    TTAG_SCI_TE_REC_GRADED,     /* Timed Exposure Graded Mode Exp. Hdr */
    TTAG_SCI_TE_DAT_GRADED,     /* Timed Exposure Graded Mode Data */

    /* ---- Continuous Clocking Science ---- */
    TTAG_SCI_CC_REC_RAW,        /* Continuous Clk Raw Mode Exposure Hdr */
    TTAG_SCI_CC_DAT_RAW,        /* Continuous Clk Raw Mode Data */
    TTAG_SCI_CC_REC_FAINT,      /* Continuous Clk Faint Mode Exp. Hdr */
    TTAG_SCI_CC_DAT_FAINT,      /* Continuous Clk Faint Mode Data */
    TTAG_SCI_CC_REC_GRADED,     /* Continuous Clk Graded Mode Exp. Hdr */
    TTAG_SCI_CC_DAT_GRADED,     /* Continuous Clk Graded Mode Data */

    TTAG_SCI_CC_BIAS,           /* Continuous Clocking Bias Map */
    TTAG_SCI_BIAS_ERROR,        /* Science Bias Error Data Packet */
```

```
    TTAG_DUMP_SYS_CONFIG,       /* Dump of System Configuration Table */
    TTAG_DUMP_BAD_PIXEL,        /* Dump of Bad Pixel Map */
    TTAG_DUMP_BAD_TE_COL,       /* Dump of Bad Timed Exposure Columns */
    TTAG_DUMP_BAD_CC_COL,       /* Dump of Bad Cont. Clocking Columns */
    TTAG_DUMP_PATCHES,          /* Dump of Patch List */
    TTAG_DUMP_HUFFMAN,          /* Dump of Huffman Tables */
    TTAG_DUMP_TE_SLOTS,         /* Dump of Timed Exposure Slots */
    TTAG_DUMP_CC_SLOTS,         /* Dump of Cont. Clocking Slots */
    TTAG_DUMP_2D_SLOTS,         /* Dump of 2D Window Slots */
    TTAG_DUMP_1D_SLOTS,         /* Dump of 1D Window Slots */
    TTAG_DUMP_DEA_SLOTS,        /* Dump of DEA Housekeeping Slots */

    TTAG_FILL_PATTERN = 45,     /* Tag received if hw fill pattern used */

    TTAG_SCI_TE_DAT_FAINT_5x5,  /* Timed Exposure Faint 5x5 Data */
    TTAG_SCI_TE_REC_FAINT_5x5,  /* Timed Exposure Faint 5x5 Exp. Hdr */

    TTAG_SCI_TE_DAT_EV_HIST,    /* 3x3 Event Histogram Data */
    TTAG_SCI_TE_REC_EV_HIST,    /* 3x3 Event Histogram Exposure Record */

    /* ---- Miscelleneous Codes ---- */
    TTAG_RESERVED = 0x3f,       /* Reserved for Maintenance Use */

    TTAG_COUNT,                 /* Number of Format Tags */
    TTAG_LASTID = TTAG_COUNT-1
};
```

```
/* ---------------------------------------------------------------------
 * Software Bi-level Discrete Telemetry (LED) Code Definitions
 * --------------------------------------------------------------------- */

/* ---- LED Value States ---- */
enum LedState
{
  LED_WD_SCIENCE_A,             /* WD Science Active - Blink State A */
  LED_WD_SCIENCE_B,             /* WD Science Active - Blink State B */
  LED_WD_IDLE_A,                /* WD Idle - Blink State A */
  LED_WD_IDLE_B,                /* WD Idle - Blink State B */
  LED_RUN_SCIENCE_A,            /* Science Active - Blink State A */
  LED_RUN_SCIENCE_B,            /* Science Active - Blink State B */
  LED_RUN_IDLE_A,               /* Idle - Blink State A */
  LED_RUN_IDLE_B,               /* Idle - Blink State B */
  LED_RUN_STARTUP,              /* Initializing loaded code */
  LED_RUN_PATCH,                /* About to patch loaded code */
  LED_BOOT_UPLINK_EXECUTE,      /* About to execute loaded code */
  LED_BOOT_UPLINK_COPY,         /* Copying packets from uplink FIFO */
  LED_BOOT_UPLINK_WAIT,         /* Waiting for initial load from uplink pkt */
  LED_BOOT_SPARE1,              /* Spare (was About to execute copied code) */
  LED_BOOT_SPARE2,              /* Spare (was Copying ROM into Bep RAM) */
  LED_BOOT_RESET                /* Bep was just reset */
};
```

```
/* ---------------------------------------------------------------------
 * CCD Identifier Code Definitions
 * --------------------------------------------------------------------- */

/* ---- CCD Id Codes ---- */
enum CcdId
{
    CCD_I0 = 0,                 /* Imaging CCD I0 */
    CCD_I1 = 1,                 /* Imaging CCD I1 */
    CCD_I2 = 2,                 /* Imaging CCD I2 */
    CCD_I3 = 3,                 /* Imaging CCD I3 */

    CCD_S0 = 4,                 /* Spectroscopy CCD S0 */
    CCD_S1 = 5,                 /* Spectroscopy CCD S1 */
    CCD_S2 = 6,                 /* Spectroscopy CCD S2 */
    CCD_S3 = 7,                 /* Spectroscopy CCD S3 */
    CCD_S4 = 8,                 /* Spectroscopy CCD S4 */
    CCD_S5 = 9,                 /* Spectroscopy CCD S5 */

    CCD_DESELECT,               /* Code used to indicate no CCD selection */
    CCD_COUNT = CCD_DESELECT,   /* Number of selectable CCD Codes */
    CCD_LASTID = CCD_COUNT - 1
};
```

```
/* ---------------------------------------------------------------------
 * SRAM/PRAM Load Format Structures
 * --------------------------------------------------------------------- */

/* ---- Define DEA Load Structures ---- */
struct DeaSection
{
    unsigned short select;      /* DEA CCD Controller Set and PRAM/SRAM Set */
                                /* Bit 0 - Sequencer 0 */
                                /* Bit 1 - Sequencer 1 */
                                /* Bit 2 - Sequencer 2 */
                                /* Bit 3 - Sequencer 3 */
                                /* Bit 4 - Sequencer 4 */
                                /* Bit 5 - Sequencer 5 */
                                /* Bit 6 - Sequencer 6 */
                                /* Bit 7 - Sequencer 7 */
                                /* Bit 8 - Sequencer 8 */
                                /* Bit 9 - Sequencer 9 */
                                /* Bits 10..14 unused */
                                /* Bit 15 - 0 Load SRAM, 1 Load PRAM */
    unsigned short index;       /* Starting RAM Index */
    unsigned short count;       /* # words in block */

    /* ---- Remainder of section contains words to load ---- */
};

struct DeaSequenceLoad
{
    unsigned short sequence;    /* Sequence Type (TBD) */
    unsigned short sectcnt;     /* Sequencer Section Count */

    /* -- The remainder of the structure contains "sectcnt" sectinos -- */
};
```

```
/* ---------------------------------------------------------------------
 * FEP Program Load Format Structures
 * --------------------------------------------------------------------- */

/* ---- Define FEP Program Load Structures ---- */
struct FepSection
{
    unsigned short type;        /* Section Type (TBD) */
    unsigned short addrl;       /* Low word of address to load */
    unsigned short addrh;       /* High word of address to load */
    unsigned short count;       /* # words in section */
    /* -- The remainder of the structure contains the data to load -- */
};

struct FepProgram
{
    unsigned short program;     /* Program Type (TBD) */
    unsigned short sectcnt;     /* Section Count */

    unsigned short execl;       /* Low word of Execution address */
    unsigned short exech;       /* High word of Execution address */

    unsigned short cmdboxl;     /* Low word of Command Mailbox address */
    unsigned short cmdboxh;     /* High word of Command Mailbox address */

    unsigned short ringl;       /* Low word of Ring Buffer address */
    unsigned short ringh;       /* High word of Ring Buffer address */

    /* -- The remainder of the structure contains "sectcnt" sections -- */
};
```

```
/* ---------------------------------------------------------------------
 * FEP Identification Codes
 * --------------------------------------------------------------------- */

/* ---- Front End Processor Id Codes ---- */
enum FepId
{
    FEP_0 = 0,                  /* FEP Slot 0 Identifier */
    FEP_1 = 1,                  /* FEP Slot 1 Identifier */
    FEP_2 = 2,                  /* FEP Slot 2 Identifier */
    FEP_3 = 3,                  /* FEP Slot 3 Identifier */
    FEP_4 = 4,                  /* FEP Slot 4 Identifier */
    FEP_5 = 5,                  /* FEP Slot 5 Identifier */

    FEP_COUNT,                  /* Number of FEP Ids */
    FEP_LASTID = FEP_COUNT - 1
};
```

```
/* ---------------------------------------------------------------------
 * CCD Video Chain Identification Codes
 * --------------------------------------------------------------------- */

/* ---- CCD Output Node (Quadrant) Id Codes ---- */
enum QuadId
{
    ONODE_A = 0,                /* Output Node A */
    ONODE_B = 1,                /* Output Node B */
    ONODE_C = 2,                /* Output Node C */
    ONODE_D = 3,                /* Output Node D */

    ONODE_COUNT,                /* Number of Output Node Ids */
    ONODE_LASTID = ONODE_COUNT - 1
};
```

```
/* ---------------------------------------------------------------------
 * CCD Output Register Clocking Mode Codes
 * --------------------------------------------------------------------- */

/* ---- CCD Quadrant Clocking Mode Codes ---- */
enum QuadMode
{
    QUAD_FULL = 0,              /* Full Mode */
    QUAD_DIAG = 1,              /* Reverse Direction Mode */
    QUAD_AC   = 2,              /* AC Mode */
    QUAD_BD   = 3,              /* BD Mode */

    QUAD_COUNT,                 /* Number of Quadrant Clocking Modes */
    QUAD_LASTID = QUAD_COUNT - 1
};
```

```
/* ---------------------------------------------------------------------
 * DEA Housekeeping Query Identification Codes
 * --------------------------------------------------------------------- */

/* ---- DEA Interface Board Query Identifiers ---- */
enum DeaQueryCntlId
{
    /* ---- Interface Board Queries ---- */
    DEAHOUSE_CNTL_BASE = 0,           /* Base Query for controller board */
                                      /* Relay Positions */
    DEAHOUSE_CNTL_RELAY=DEAHOUSE_CNTL_BASE,

    DEAHOUSE_CNTL_ADC_BASE,           /* ADC Base Query */
                                      /* DPA Thermistor 1 - BEP PC Board */
    DEAHOUSE_CNTL_ADC_TMP_BEP_PCB=DEAHOUSE_CNTL_ADC_BASE,
    DEAHOUSE_CNTL_ADC_TMP_BEP_OSC,    /* DPA Thermistor 2 - BEP Oscillator */
    DEAHOUSE_CNTL_ADC_TMP_FEP0_MONG,  /* DPA Thermistor 3 - FEP 0 Mongoose */
    DEAHOUSE_CNTL_ADC_TMP_FEP0_PCB,   /* DPA Thermistor 4 - FEP 0 PC Board */
    DEAHOUSE_CNTL_ADC_TMP_FEP0_ACTEL, /* DPA Thermistor 5 - FEP 0 ACTEL */
    DEAHOUSE_CNTL_ADC_TMP_FEP0_RAM,   /* DPA Thermistor 6 - FEP 0 RAM */
    DEAHOUSE_CNTL_ADC_TMP_FEP0_FB,    /* DPA Thermistor 7 - FEP 0 Frame Buf. */
    DEAHOUSE_CNTL_ADC_TMP_FEP1_MONG,  /* DPA Thermistor 8 - FEP 1 Mongoose */
    DEAHOUSE_CNTL_ADC_TMP_FEP1_PCB,   /* DPA Thermistor 9 - FEP 1 PC Board */
    DEAHOUSE_CNTL_ADC_TMP_FEP1_ACTEL, /* DPA Thermistor 10- FEP 1 ACTEL */
    DEAHOUSE_CNTL_ADC_TMP_FEP1_RAM,   /* DPA Thermistor 11- FEP 1 RAM */
    DEAHOUSE_CNTL_ADC_TMP_FEP1_FB,    /* DPA Thermistor 12- FEP 1 Frame Buf. */
    DEAHOUSE_CNTL_ADC_SUBAHK,         /* DEA Video Board ADC */
    DEAHOUSE_CNTL_ADC_SPARE1,         /* Spare - Unused */
    DEAHOUSE_CNTL_ADC_FPTEMP_12,      /* Spare - Focal Plane Temp. Board 12 */
    DEAHOUSE_CNTL_ADC_FPTEMP_11,      /* Spare - Focal Plane Temp. Board 11 */
    DEAHOUSE_CNTL_ADC_DPAGNDREF1,     /* DPA Ground Reference 1 */
    DEAHOUSE_CNTL_ADC_DPA5VHKA,       /* DPA 5V Housekeeping A */
    DEAHOUSE_CNTL_ADC_DPAGNDREF2,     /* DPA Ground Reference 2 */
    DEAHOUSE_CNTL_ADC_DPA5VHKB,       /* DPA 5V Housekeeping B */
    DEAHOUSE_CNTL_ADC_UNUSED1,        /* Unused */
    DEAHOUSE_CNTL_ADC_UNUSED2,        /* Unused */
    DEAHOUSE_CNTL_ADC_UNUSED3,        /* Unused */
    DEAHOUSE_CNTL_ADC_UNUSED4,        /* Unused */
    DEAHOUSE_CNTL_ADC_DEA28VDCA,      /* Primary Raw DEA 28V DC */
    DEAHOUSE_CNTL_ADC_DEA24VDCA,      /* Primary Raw DEA 24V DC */
    DEAHOUSE_CNTL_ADC_DEAM15VDCA,     /* Primary Raw DEA -15.5V */
    DEAHOUSE_CNTL_ADC_DEAP15VDCA,     /* Primary Raw DEA +15.5V */
    DEAHOUSE_CNTL_ADC_DEAM6VDCA,      /* Primary Raw DEA -6V DC */
    DEAHOUSE_CNTL_ADC_DEAP6VDCA,      /* Primary Raw DEA +6V DC */
    DEAHOUSE_CNTL_ADC_RAD_PCB_A,      /* Relative Dose Rad. Monitor Side A */
    DEAHOUSE_CNTL_ADC_GND_1,          /* Interface Ground Reference */
    DEAHOUSE_CNTL_ADC_DEA28VDCB,      /* Backup Raw DEA 28V DC */
    DEAHOUSE_CNTL_ADC_DEA24VDCB,      /* Backup DEA 24V DC */
    DEAHOUSE_CNTL_ADC_DEAM15VDCB,     /* Backup DEA -15.5V DC */
    DEAHOUSE_CNTL_ADC_DEAP15VDCB,     /* Backup DEA +15.5V DC */
    DEAHOUSE_CNTL_ADC_DEAM6VDCB,      /* Backup DEA -6V DC */
    DEAHOUSE_CNTL_ADC_DEAP6VDCB,      /* Backup DEA +6V DC */
    DEAHOUSE_CNTL_ADC_RAD_PCB_B,      /* Relavtive Dose Rad. Monitor Side B */
    DEAHOUSE_CNTL_ADC_GND_2,          /* Ground */

    DEAHOUSE_CNTL_ADC_END = DEAHOUSE_CNTL_ADC_GND_2,
    DEAHOUSE_CNTL_END = DEAHOUSE_CNTL_ADC_END,
};

/* ---- CCD Controller Query Identifiers ---- */
enum DeaQueryCcdId
{
    /* ---- CCD Controller Queries ---- */
```

```
    DEAHOUSE_CCD_BASE = 0,                    /* Base Query for CCD Queries */
    DEAHOUSE_CCD_REG_0 = DEAHOUSE_CCD_BASE,   /* Register 0 Sequencer Control */
    DEAHOUSE_CCD_REG_1,                       /* Register 1 Video ADC Control */
    DEAHOUSE_CCD_REG_2,                       /* Register 2 Test Aid */
    DEAHOUSE_CCD_REG_3,                       /* Register 3 Misc. */

    DEAHOUSE_CCD_ADC_BASE = 0x80,             /* Base Index for ADC Regs. */
                                              /* Image Array Parallel + */
    DEAHOUSE_CCD_PIA_P = DEAHOUSE_CCD_ADC_BASE,
    DEAHOUSE_CCD_PIA_M,                       /* Image Array Parallel - */
    DEAHOUSE_CCD_PFS_P,                       /* Framestore Parallel + */
    DEAHOUSE_CCD_PFS_M,                       /* Framestore Parallel - */
    DEAHOUSE_CCD_S_P,                         /* Serial Register + */
    DEAHOUSE_CCD_S_M,                         /* Serial Register - */
    DEAHOUSE_CCD_R_P,                         /* Reset Gate + */
    DEAHOUSE_CCD_R_M,                         /* Reset Gate - */
    DEAHOUSE_CCD_OG,                          /* Output Gate Bias Level */
    DEAHOUSE_CCD_SCP,                         /* Scupper */
    DEAHOUSE_CCD_RD,                          /* Reset Diode */
    DEAHOUSE_CCD_DR0,                         /* Drain Output Channel A */
    DEAHOUSE_CCD_DR1,                         /* Drain Output Channel B */
    DEAHOUSE_CCD_DR2,                         /* Drain Output Channel C */
    DEAHOUSE_CCD_DR3,                         /* Drain Output Channel D */
    DEAHOUSE_SPARE,                           /* Spare Housekeeping */
    DEAHOUSE_CCD_TEMP_BOARD,                  /* Board Temperature (RTD4) */
    DEAHOUSE_CCD_TEMP_SRAM,                   /* SRAM Temperature (RTD3) */
    DEAHOUSE_CCD_TEMP_ADC,                    /* ADC Temperature (RTD2) */
    DEAHOUSE_CCD_TEMP_ACTEL,                  /* Gate Array Temp. (RTD1) */

    DEAHOUSE_CCD_END = DEAHOUSE_CCD_TEMP_ACTEL /* Last CCD Query */
};

#define DEAHOUSE_VALUE_INVALID (0xffff)       /* Result of Failed Query */
```

```
/* ---------------------------------------------------------------------
 * System Configuration Identification Codes
 * --------------------------------------------------------------------- */

/* ---- System Configuration Settings ---- */
enum SystemSettings
{
    SYSSET_BASE,

    /* ---- Power Control ---- */
    SYSSET_DEA_POWER = SYSSET_BASE,    /* DEA Power Setting Selections */
    SYSSET_FEP_POWER,                  /* FEP Power Setting Selections */

    /* ---- DEA Interface Board Settings ---- */
    SYSSET_CNTL_BASE,           /* Base settings for controller board */
    SYSSET_CNTL_MASTER_CLK = SYSSET_CNTL_BASE, /* Master clock during science */
    SYSSET_CNTL_FOCAL_TEMP,            /* Focal Plane Temperature */
    SYSSET_CNTL_BAKE_TEMP,             /* BakeOut Temperature */
    SYSSET_CNTL_BAKE_ENABLE,           /* BakeOut Enable */
    SYSSET_CNTL_LED_ENABLE,            /* LED Enable */
    SYSSET_CNTL_HOUSE_HOLD,            /* Hold Housekeeping Address */
    SYSSET_CNTL_SIGNAL_PATH,           /* Signal Path Selection */
    SYSSET_CNTL_CMDCLOCK_DISABLE,      /* Command Clock Enable Select */
    SYSSET_CNTL_CMDDATA_DISABLE,       /* Command Data Enable Select */
    SYSSET_CNTL_RELAY_SET_0,           /* DEA Board Relay Selections */
    SYSSET_CNTL_RELAY_SET_1,
    SYSSET_CNTL_RELAY_SET_2,
    SYSSET_CNTL_RELAY_SET_3,
    SYSSET_CNTL_RELAY_SET_4,
    SYSSET_CNTL_END = SYSSET_CNTL_RELAY_SET_4,
    SYSSET_CNTL_COUNT = SYSSET_CNTL_END - SYSSET_CNTL_BASE + 1,

    /* ---- DEA CCD Controller Register Settings ---- */
    SYSSET_CCD_BASE = SYSSET_CNTL_END+1, /* Base Settings for CCD Controllers */

    SYSSET_CCD_SEQ_OFFSET = SYSSET_CCD_BASE, /* Sequencer Offset */
    SYSSET_CCD_ADC_OFFSET,             /* Video ADC Offset */
    SYSSET_CCD_VIDEO_ENABLE,           /* Video Channel Enable Mask */
    SYSSET_CCD_HOLD_HOUSE,             /* Hold Housekeeping Address */
    SYSSET_CCD_BJD,                    /* Back-Junction Diode Enable */
    SYSSET_CCD_HIGH_SPEED_TAP,         /* High-speed tap enable */

    /* ---- DEA CCD Controller Digital-To-Analog Converter Settings ---- */
    SYSSET_DAC_BASE,                    /* Delimit Start of DAC codes */
    SYSSET_DAC_PIA_P = SYSSET_DAC_BASE, /* Image Array Parallel + */
    SYSSET_DAC_PIA_MP,                  /* Image Array Parallel -+ */
    SYSSET_DAC_PIA_M,                   /* Image Array Parallel - */

    SYSSET_DAC_PFS_P,                  /* Framestore Parallel + */
    SYSSET_DAC_PFS_MP,                 /* Framestore Parallel -+ */
    SYSSET_DAC_PFS_M,                  /* Framestore Parallel - */

    SYSSET_DAC_S_P,                    /* Serial Register + */
    SYSSET_DAC_S_M,                    /* Serial Register - */

    SYSSET_DAC_R_P,                    /* Reset Gate + */
    SYSSET_DAC_R_MP,                   /* Reset Gate -+ */
    SYSSET_DAC_R_M,                    /* Reset Gate - */

    SYSSET_DAC_SCP,                    /* Scupper */

    SYSSET_DAC_OG_P,                   /* Output Gate + */
    SYSSET_DAC_OG_M,                   /* Output Gate - */
```

```
    SYSSET_DAC_RD,                      /* Reset Diode */

    SYSSET_DAC_DR0,                     /* Drain Output (A) */
    SYSSET_DAC_DR1,                     /* Drain Output (B) */
    SYSSET_DAC_DR2,                     /* Drain Output (C) */
    SYSSET_DAC_DR3,                     /* Drain Output (D) */

    SYSSET_DAC_A_OFF,                   /* A channel offset */
    SYSSET_DAC_B_OFF,                   /* B channel offset */
    SYSSET_DAC_C_OFF,                   /* C channel offset */
    SYSSET_DAC_D_OFF,                   /* D channel offset */

    SYSSET_DAC_SPARE,                   /* Spare DAC Channel */
    SYSSET_DAC_END = SYSSET_DAC_SPARE, /* Last DAC Setting */
    SYSSET_DAC_COUNT = SYSSET_DAC_END - SYSSET_DAC_BASE + 1,
    SYSSET_CCD_END = SYSSET_DAC_END,   /* Last CCD Controller Setting */

    SYSSET_NSETTINGS = SYSSET_CCD_BASE +
                       ((SYSSET_CCD_END - SYSSET_CCD_BASE + 1)*10),
    SYSSET_COUNT
};
```

```
/* ------------------------------------------------------------------
 * Software Statistics Definitions
 * ------------------------------------------------------------------ */

/* ---- Software Housekeeping Information Structures ---- */
enum SwStatistic
{
    SWSTAT_VERSION,             /* ACIS Software Version Number */

    SWSTAT_SWHOUSE_RANGE,       /* Housekeeping report beyond end of list */
    SWSTAT_SWHOUSE_SKIPPED,     /* Dropped Software Housekeeping Statistic */

    SWSTAT_TIMERCB_INVOKE,      /* Timer Interrupt Callback invocations */

    SWSTAT_FEPLOCK_TIMEOUT,     /* Fep Wait: timed out */
    SWSTAT_FEPLOCK_POWEROFF,    /* Fep Wait: no power */
    SWSTAT_FEPLOCK_RESET,       /* Fep Wait: is reset */
    SWSTAT_FEPLOCK_NOIO,        /* Fep Wait: No mailbox/ringbuffer */

    SWSTAT_FEPREPLY_TIMEOUT,    /* Fep Wait: timed out */
    SWSTAT_FEPREPLY_POWEROFF,   /* Fep Wait: no power */
    SWSTAT_FEPREPLY_RESET,      /* Fep Wait: is reset */
    SWSTAT_FEPREPLY_NOIO,       /* Fep Wait: No mailbox/ringbuffer */

    SWSTAT_SCI_STOPRUN,         /* Science Run Stop Invoked */
    SWSTAT_SCI_STOPRUN_IDLE,    /* Stopped when already idle */
    SWSTAT_SCI_STOPRUN_RSTOP,   /* Stop Request issued to mode */

    SWSTAT_SCI_STARTRUN,         /* Science Run Start Invoked */
    SWSTAT_SCI_STARTRUN_BUSY,    /* Start when not idle */
    SWSTAT_SCI_STARTRUN_RUNNING, /* Start aborted previous run */
    SWSTAT_SCI_STARTRUN_RSTOP,   /* Start requested stop to prv. mode */

    SWSTAT_SCI_EXPSTART_ZERO_EXPNUM, /* Exposure Number from FEP is 0 */
    SWSTAT_SCI_EXPEND_EXPNUM,        /* Ending Exposure Number did not match cur */
    SWSTAT_SCI_EXPSTART_NOEND,       /* Prev. Exposure missing end. */

    SWSTAT_INTR_FEPBUS,         /* FEP Bus Error Timeout [Arg: Bad Vaddr] */

    SWSTAT_TE_SHORT_DUMP_TLM,   /* Timed Exposure Dump tlm pkt too small */
    SWSTAT_2D_SHORT_DUMP_TLM,   /* TE 2-D Windows Dump tlm pkt too small */
    SWSTAT_TE_BAD_FEP_MODE,     /* Unrecognized Timed Exp. FEP Mode */
    SWSTAT_TE_BAD_BEP_MODE,     /* Unrecognized Timed Exp. BEP Mode */

    SWSTAT_CCD_NULL_SETTING,    /* CCD Setting Ptr NULL [Arg: setting id] */

    SWSTAT_CMDECHO_NULL,        /* Cmd Echo passed NULL Pkt ptr */
    SWSTAT_CMDECHO_MISMATCH,    /* Cmd Echo pkt != curpkt [Arg: curpkt] */
    SWSTAT_CMDECHO_BADLENGTH,   /* Cmd Echo cmd too long [Arg: cmd data cnt] */
    SWSTAT_CMDECHO_TRUNCATE,    /* Cmd Echo truncated [Arg: cmd data cnt] */
    SWSTAT_CMDECHO_DROPPED,     /* Cmd Echo Dropped [Arg: cmd pkt id] */

    SWSTAT_CMDMAN_INVALID,      /* Invalid Cmd Pkt [Arg: First word read] */
    SWSTAT_CMDMAN_ERRCALLED,    /* # calls to CmdMan::handleError() */
    SWSTAT_CMDMAN_ERRRETRY,     /* # retries in CmdMan::handleError() */
    SWSTAT_CMDMAN_HANDLED,      /* # calls to CmdMan::handleCommand() */
    SWSTAT_CMDMAN_BADLENGTH,    /* CmdMan Bad Pkt Length [Arg: Cmd Length] */

    SWSTAT_DEAMAN_PRAMWRITE,    /* DeaMan Bad PRAM Address [Arg: PRAM Index] */
    SWSTAT_DEAMAN_PRAMREAD,     /* DeaMan Bad PRAM Address [Arg: PRAM Index] */
    SWSTAT_DEAMAN_SRAMWRITE,    /* DeaMan Bad SRAM Address [Arg: SRAM Index] */
    SWSTAT_DEAMAN_SRAMREAD,     /* DeaMan Bad SRAM Address [Arg: SRAM Index] */
    SWSTAT_DEAMAN_BADCNTLREG,   /* DeaMan Bad Cntl Reg [Arg: Reg Index] */
```

```
    SWSTAT_PHHIST_BADQUAD,      /* PH Histogram Bad Quad Mode [Arg: mode] */
    SWSTAT_PIX1X3_CORRUPTROW,   /* Pixel1x3 Bad Row [Arg: row] */
    SWSTAT_PIX1X3_CORRUPTCOL,   /* Pixel1x3 Bad Column [Arg: col] */
    SWSTAT_PMTEHIST_BADQUAD,    /* PH Te Hist Bad Quad Mode [Arg: mode] */

    SWSTAT_PIX3X3_CORRUPTROW,   /* Pixel3x3 Bad Row [Arg: row] */
    SWSTAT_PIX3X3_CORRUPTCOL,   /* Pixel3x3 Bad Column [Arg: col] */

    SWSTAT_FEPMAN_RINGRDINDX,   /* FepMan Corrupt Rd Index [Arg: readIndex] */
    SWSTAT_FEPMAN_RINGWRINDX,   /* FepMan Corrupt Wr Index [Arg: writeIndex] */

    SWSTAT_PM_BADRECTYPE,       /* ProcessMode Bad Record Type [Arg: type] */

    SWSTAT_DEACCD_LOADINVALID,  /* CCD Cntl Start on invalid load [Arg: 0] */
    SWSTAT_DEABOARD_ERROR,      /* DEA Error [Arg: (slot << 16) | errcode] */

    SWSTAT_FEPCMD_MBOXSTATE,    /* FEP Mailbox not empty [Arg: mbox state] */

    SWSTAT_FEP_READMEM,         /* FEP Read Memory Called [Arg: fepid] */
    SWSTAT_FEP_WRITEMEM,        /* FEP Write Memory Called [Arg: fepid] */
    SWSTAT_FEP_EXECMEM,         /* FEP Execute Memory Called [Arg: fepid] */
    SWSTAT_FEP_STARTBIAS,       /* FEP Start Bias [Arg: none] */
    SWSTAT_FEP_STOP,            /* FEP Stop Issued [Arg: abortFlag] */
    SWSTAT_FEP_STARTDATA,       /* FEP Start Data Processing [Arg: requestType] */
    SWSTAT_FEP_QUERY,           /* FEP Query [Arg: fepid] */

    SWSTAT_SMPROC_RSTOP,        /* Sci Mode Data Proc Stop Rqst [Arg: none] */
    SWSTAT_SMWAITBIAS_ABORT,    /* Sci Mode Bias Wait Abort [Arg: caught] */
    SWSTAT_SMWAITEVENT_CAUGHT,  /* Sci Mode Event Wait Signal [Arg: caught] */
    SWSTAT_SMWAITEVENT_ABORT,   /* Sci Mode Event Wait Abort [Arg: caught] */
    SWSTAT_SMRABORT,            /* Sci Mode Request Abort [Arg: reason] */

    SWSTAT_SCI_DUMPFAILED,      /* Sci Manager Dump Failed [Arg: none] */
    SWSTAT_SCI_SETUPFAILED,     /* Sci Manager Setup Failed [Arg: none] */
    SWSTAT_SCI_DEADUMPFAILED,   /* Sci Manager DEA Dump Failed [Arg: none] */
    SWSTAT_SCI_DEACHECKFAILED,  /* Sci Manager DEA Check Failed [Arg: none] */
    SWSTAT_SCI_BIASFAILED,      /* Sci Manager Bias Failed [Arg: none] */
    SWSTAT_SCI_DATACOMPLETE,    /* Sci Manager Data Run Complete [Arg: none] */
    SWSTAT_SCI_BIASCOMPLETE,    /* Sci Manager Bias Run Complete [Arg: none] */

    SWSTAT_SCI_INHIBIT_ON,      /* Sci Man Inhibit On [Arg: prv state] */
    SWSTAT_SCI_INHIBIT_OFF,     /* Sci Man Inhibit Off [Arg: prv state] */

    SWSTAT_FEPMAN_POWERON,      /* FEP Manager Power On [Arg: fepid] */
    SWSTAT_FEPMAN_POWEROFF,     /* FEP Manager Power Off [Arg: fepid] */
    SWSTAT_FEPMAN_STARTLOAD,    /* FEP Manager Start Prog. Load [Arg: fepid] */
    SWSTAT_FEPMAN_ENDLOAD,      /* FEP Manager End Prog. Load [Arg: fepid] */
    SWSTAT_DEACCD_POWERON,      /* DEA Ccd Cntl Power On [Arg: board] */
    SWSTAT_DEACCD_POWEROFF,     /* DEA Ccd Cntl Power Off [Arg: board] */

    SWSTAT_SCI_EXPSTART_FEPTIME, /* FEP Timestamp corrupted [Arg: fepTime] */

    SWSTAT_FEPREPLY_BADTYPE,    /* FEP Reply Bad Type [Arg: fepid] */
    SWSTAT_FEPREC_POWEROFF,     /* FEP Read Record No Power [Arg: fepid] */
    SWSTAT_FEPREC_RESET,        /* FEP Read Record Reset [Arg: fepid] */
    SWSTAT_FEPCFG_NACK,         /* FEP Config Nack [Arg: fepid] */
    SWSTAT_FEPDIST_NACK,        /* FEP Distribute Cmd Nack [Arg: fepid] */

    SWSTAT_SYSCFG_IN_CLIP,      /* SysCfg clipped stored item [Arg: item] */

    SWSTAT_SCI_JITTERFAILED,    /* Sci Man Jitter DAC operation failed */
    SWSTAT_SMWAITTRICKLE_ABORT, /* Sci Mode Bias Trickle Abort [Arg: caught] */
```

```
    SWSTAT_COUNT,
    SWSTAT_LAST = SWSTAT_COUNT - 1 /* Last slot is sent but unused */
};

#define SWSTAT_MARGIN  (64)     /* Reserved space for patched in codes */
```

```
/* ---------------------------------------------------------------------
 * Fatal Error Code Definitions
 * --------------------------------------------------------------------- */

/* ---- Fatal Error Telemetry Packet Information ---- */
enum FatalCode
{
    FATAL_UNKNOWN = 0,          /* Unknown Fatal Error */
    FATAL_RTXERROR,             /* Nucleus RTX generated fatal error */
    FATAL_EXCEPTION,            /* Processor Exception */
    FATAL_INTERRUPT,            /* Unexpected Interrupt Cause */
    FATAL_FEPDEVICE_BAD_FEPID,  /* Bad FEP Id */
    FATAL_TASK_EXIT,            /* Task returned [Arg: Task Ptr] */
    FATAL_RTX_RETURNED,         /* Nucleus RTX Returned */
    FATAL_INTR_FEP_BUS_ERROR,   /* FEP Bus Error [Arg: Bad Vaddr] */

    FATAL_LAST,                 /* Last slot unused */
    FATAL_COUNT
};
```

```
/* ---------------------------------------------------------------
 * Bias Algorithm Selection Codes
 * --------------------------------------------------------------- */

/* ---- FEP Bias Algorithm Type ---- */
typedef enum {                  /* what sort of bias algorithm? */
  FEP_NO_BIAS,                  /* none */
  FEP_BIAS_1,                   /* algorithm #1 */
  FEP_BIAS_2                    /* algorithm #2 */
} fepBiasType;
```

```
/* ---------------------------------------------------------------------
 * FEP Science Report Error Codes
 * --------------------------------------------------------------------- */

/* ---- FEP to BEP Error Codes ---- */
typedef enum {
  FEP_CMD_NOERR = 0,            /* no errors detected */
  FEP_CMD_ERR_NO_RUN,           /* no command currently running */
  FEP_CMD_ERR_UNK_CMD,          /* unknown command type */
  FEP_CMD_ERR_PARM_LEN,         /* parameter block too long */
  FEP_CMD_ERR_PARM_TYPE,        /* unknown parameter block type */
  FEP_CMD_ERR_QUAD_CODE,        /* unknown quadrant code */
  FEP_CMD_ERR_BIAS_TYPE,        /* unknown bias type code */
  FEP_CMD_ERR_BIAS_PARM0,       /* bad bias parm 0 */
  FEP_CMD_ERR_NROWS,            /* bad number of rows */
  FEP_CMD_ERR_NCOLS,            /* bad number of columns */
  FEP_CMD_ERR_NOCLK,            /* bad number of overclocks */
  FEP_CMD_ERR_NHIST,            /* bad histogram exposure count */
  FEP_CMD_ERR_NO_PARM,          /* no parameter block loaded */
  FEP_CMD_ERR_BAD_CMD,          /* illegal secondary command */
  FEP_CMD_ERR_NO_BIAS,          /* no bias map stored */
} fepCmdRetCode;

/* ---- BEP Generated FEP I/O Errors ---- */
enum FepIoErrors
{
  FEP_ERR_LOCK_TIMEOUT   = 0x80, /* Timeout on FEP exclusive access lock */
  FEP_ERR_NO_POWER       = 0x81, /* FEP has no power */
  FEP_ERR_IS_RESET       = 0x82, /* FEP is reset */
  FEP_ERR_NO_CMDRING     = 0x83, /* FEP has no command mailbox or ringbuffer */
  FEP_ERR_REPLY_TIMEOUT  = 0x84, /* FEP reply timed-out */
  FEP_ERR_BAD_REPLY_TYPE = 0x85, /* FEP produced invalid reply to BEP action */
  FEP_ERR_BAD_MBOX_STATE = 0x86  /* FEP state invalid after successful lock */
};
```

```
/* ---------------------------------------------------------------------
 * FEP Science Mode Selection Codes
 * --------------------------------------------------------------------- */

/* ---- Timed Exposure FEP Modes ---- */
enum TeFepMode
{
  FEP_TE_MODE_RAW,              /* Raw Mode */
  FEP_TE_MODE_HIST,             /* Histogram Mode */
  FEP_TE_MODE_EV3x3,            /* 3x3 Event Detection Mode */
  FEP_TE_MODE_EV5x5,            /* 5x5 Event Detection Mode */

  FEP_TE_MODE_COUNT,
  FEP_TE_LASTMODE = FEP_TE_MODE_COUNT - 1
};

/* ---- Continuous Clocking FEP Modes ---- */
enum CcFepMode
{
  FEP_CC_MODE_RAW,              /* Raw Mode */
  FEP_CC_MODE_EV1x3,            /* 1x3 Event Detection Mode */

  FEP_CC_MODE_COUNT,
  FEP_CC_LASTMODE = FEP_CC_MODE_COUNT - 1
};
```

```
/* ---------------------------------------------------------------------
 * BEP Packing Mode Selection Codes
 * --------------------------------------------------------------------- */

/* ---- Timed Exposure BEP Packing Modes ---- */
enum TeBepMode
{
  BEP_TE_MODE_FAINT,            /* 3x3 Faint Mode Event Telemetry */
  BEP_TE_MODE_FAINTBIAS,        /* 3x3 Faint with Bias Event Telemetry */
  BEP_TE_MODE_GRADED,           /* 3x3 Graded Telemetry */
  BEP_TE_MODE_EVHIST,           /* 3x3 Event Histogram Telemetry */

  BEP_TE_MODE_COUNT,
  BEP_TE_LASTMODE = BEP_TE_MODE_COUNT - 1
};

/* ---- Continuous Clocking BEP Packing Modes ---- */
enum CcBepMode
{
  BEP_CC_MODE_FAINT,            /* 1x3 Faint Mode Event Telemetry */
  BEP_CC_MODE_GRADED,           /* 1x3 Graded Event Telemetry */

  BEP_CC_MODE_COUNT,
  BEP_CC_LASTMODE = BEP_CC_MODE_COUNT - 1
};
```

```
/* ------------------------------------------------------------------
 * Science Run Termination Reason Codes
 * ------------------------------------------------------------------ */

/* ---- Science Run Termination Reason Codes ---- */
enum SmTerminationCode
{
  SMTERM_UNUSED,                /* Unused */
  SMTERM_STOPCMD,               /* Commanded to Stop i.e. normal termination */
  SMTERM_BIASDONE,              /* Bias-only Run completed */
  SMTERM_RADMON,                /* Radiation Monitor was asserted */
  SMTERM_CLOBBERED,             /* Clobbered by another start command */

  SMTERM_FEP_BIAS_START,        /* FEP Bias Processing did not start */
  SMTERM_FEP_DATA_START,        /* FEP Data Processing did not start */
  SMTERM_CCD_BIAS_START,        /* Command start clocking CCDs for bias failed */
  SMTERM_CCD_DATA_START,        /* Command start clocking CCDs for data failed */
  SMTERM_CCD_BIAS_STOP,         /* Command stop clocking CCDs for bias failed */

  SMTERM_PROC_PARM_INVALID,     /* Processing Parameter out of range */
  SMTERM_DEA_PARM_INVALID,      /* DEA Parameter out of range */
  SMTERM_FEP_PARM_INVALID,      /* FEP Parameter out of range */
  SMTERM_FEP_CONFIG_ERROR,      /* FEP Configuration Error */

  SMTERM_DEA_IO_ERROR,          /* Either I/O errors, or no CCD controllers on */
  SMTERM_FEP_IO_ERROR,          /* Either I/O errors, or no FEPs are on */

  SMTERM_UNSPECIFIED,           /* Reason is unspecified */
};
```

```
/* ---------------------------------------------------------------------
 * Miscelleneous System Definitions
 * --------------------------------------------------------------------- */
enum MiscValues
{
  PKTDELAY_SECONDS = 1,         /* Command Packet Recovery Time Delay */
};
```

```
/* ---------------------------------------------------------------------
 * FEP Implementation Constants
 * --------------------------------------------------------------------- */
                                /* Originally from fep/fepBep.h */
#define BIAS_BAD    (0xffe)     /* bias parity error value */
#define PIXEL_BAD   (0xfff)     /* pixel in bad pixel map */
#define MAX_NOCLK   (30)        /* maximum overclocks per output node */
#define MAX_NOCLKR  (16)        /* max raw overclocks per output node */
#define MAX_NROWS   (1024)      /* maximum number of pixel rows */
#define MAX_NCOLS   (1024)      /* maximum number of pixel columns */
#define MAX_STRIPS  (64)        /* maximum bparm[0] in strip mode */
#define CCLK_NROWS  (512)       /* number of pixel rows in CClk mode */
#define MAX_FID_PIX (128)       /* maximum fiducial pairs per FEP */

                                /* Originally from fep/fepCtl.h */
#define PIXEL_MASK   (0xfff)    /* valid pixel and bias bits */
#define INITSKIP     2          /* number of science exposures to skip */
```

```
/* ------------------------------------------------------------------
 * BEP Graded Mode Special Codes
 * ------------------------------------------------------------------ */

enum GradedMeanCode
{
    CORNER_MEAN_LOW = -4096,    /* Graded Mode Corner Mean below -4095 */
    CORNER_MEAN_MISSING = 4095  /* Graded Mode No Valid Corner Pixels */
};

#endif /* INTERFACE_H */
```

```
/* ============================================================
 *
 * $$Source: /acis/h3/acisfs/configcntl/patches/eventhist/pmteevhist.C,v $$
 *
 * Patch Name: Process Mode Event Histogram Source File
 *
 * Description:
 *  This implements the classes needed to implement the Event
 *  Histogram Processing mode. This Timed Exposure mode detects and
 *  filters events, as in a normal event-finding mode, but then
 *  "bins" the estimated event energies into CCD quadrant-specific
 *  histograms. Once these histograms have accumulated events from
 *  the number exposures specified in the histogramCount field of the
 *  parameter block, the mode transfers the telemetry buffers to
 *  the spacecraft, and waits for enough buffers to be transferred
 *  to start a fresh set of histograms.
 *
 *  Event Histogram mode was originally not implemented in the main
 *  flight software because the team did not think there was enough
 *  distinct memory in the system to accumulate the histograms. This
 *  was incorrect. There is enough memory if the histograms are
 *  accumulated directly into the science telemetry buffers.
 *
 *  This version of pmteevhist.C has been changed to run with the new
 *  smtimedexposure patch. The principal changes are that the old
 *  Test_SmTimedExposure class has been renamed Test1_SmTimedExposure
 *  and the setupProcess method has been replaced by the considerably
 *  shorter setupEventHist.
 *
 *  This patch consists of the following files:
 *      hamming.H    - Interface definition for the Hamming class.
 *      hamming.C    - Implementation of the Hamming class.
 *      pmteevhist.H - Interface definitions of new event histogramming
 *                     classes.
 *      pmteevhist.C - Defintition and implementation of some utility
 *                     "accessor" classes, replacement functions to
 *                     existing flight software classes, and implementation
 *                     of the new event histogramming classes.
 *
 *  The pmteevhist.H/C files define and implement the following:
 *      Tf_Data_Te_Ev_Hist - Subclass of Tf_Data_Te_Hist, whose constructor
 *                           overrides the telemetry format tags, and which
 *                           adds a "releaseBuffer" function to place
 *                           the packet's data buffer back into its pool.
 *                           Each CCD quadrant histogram (see EventHistgram)
 *                           contains enough instances of this class to hold
 *                           a single 4096-bin histogram.
 *         Tf_Data_Te_Ev_Hist - The constructor invokes the parent
 *                              constructor, and then overwrites the telemetry
 *                              format tag. It uses the utility class
 *                              Test_TlmForm to gain access to the tag instance
 *                              variable.
 *         releaseBuffer      - This member function release the packet's
 *                              buffer to its pool. It uses the utility class
 *                              Test_TlmForm to gain access to the packet
 *                              buffer pointer.
 *
 *      EventHistogram     - This class is responsible for binning events
 *                           from a single CCD quadrant, and managing
 *                           the buffers for this quadrant. Each PmTeEvHist
 *                           instance has instance of this class for each
 *                           quadrant in a CCD (i.e. 4).
 *         EventHistogram     - The constructors the instance variables
 *                              to default values.
```

```
*         ~EventHistogram    - This class's destructor has no explicit
*                              code, but the destructors for the
*                              contained Tf_Data_Te_Ev_Hist packets will
*                              release all buffers associated with the
*                              histogram.
*         waitForBuffers     - If the histogram has no telemetry buffers,
*                              This function blocks until there are
*                              enough packet buffers to hold the entire
*                              histogram, allocating the buffers as they
*                              become available. Once it has allocated
*                              enough buffers, it logs the passed exposure
*                              number and FEP timestamp and clears
*                              the exposure accumulation counter.
*                              If the histogram already has unposted
*                              buffers (i.e. it's in the proces of
*                              accumulating a multi-exposure histogram),
*                              the function increments the exposure counter
*                              and returns without modifying the current
*                              exposure number nor FEP timestamp.
*         postBuffers        - This function posts each histogram data
*                              packet for transfer to the spacecraft, storing
*                              accounting information into the packet as it
*                              goes. Once it has posted the data packets,
*                              it forms and posts an Event Histogram record
*                              packet.
*         releaseBuffers     - This function releases the data buffer for
*                              each data packet in the histogram.
*         incrementBin       - This function increments an energy bin
*                              in the histogram, computing which packet
*                              word contains the bin. It uses the Hamming
*                              class to perform the read-increment-store.
*                              The Hamming class will correct any single-bit
*                              errors in the bin prior to incrementing the
*                              bin, and will re-compute and store the new
*                              error correction code with the incremented
*                              value.
*         This class provides a variety of functions which store the FEP
*         identifier, CCD identifier, quadrant identifier, run start-time
*         and parameter block identifier. The class uses this information
*         when it builds and posts its data and exposure record telemetry
*         packets.
*
*       PmTeEvHist           - This is a subclass of the generic event
*                              processing mode, PmEvent. It is responsible
*                              for the overall processing in this mode.
*                              The patched science mode (Test_SmTimedExposure)
*                              defines one static instance of this class for
*                              each FEP in the system (i.e. 6).
*         PmTeEvHist         - The constructor initializes each Event Histogram
*                              instance, assigning the instance the mode
*                              (back to the PmTeEvHist instance), and quadrant
*                              assignment.
*         ~PmTeEvHist        - The destructor will invoke the destructors
*                              for its Event Histogram instances, which in turn
*                              will invoke the destructors for their telemetry
*                              packets, resulting in the packet buffers being
*                              returned to their pool. NOTE: buffers posted
*                              to telemetry will be returned to the pool
*                              by the telemetry code once their contents have
*                              been transferred to the spacecraft.
*         setExposureCount   - This function stores the exposure count needed
*                              to complete a histogram for a given science run.
*         processRecord      - This function processes a ring-buffer record
*                              sent by the instance's FEP. If the record
```

```
*                               indicates the start of an exposure, this
*                               function first invokes the parent's
*                               processRecord, and then initializes or logs
*                               the new exposure by calling setupBuffers().
*                               If the record is a 3x3 or 5x5 event, it
*                               filters the event, and calls binEvent() if
*                               the event passes muster. For any other type
*                               of record, the function calls the parent
*                               processRecord() to deal with it. If an abort
*                               to the science run is indicated by any of the
*                               calls, the routine calls cleanupBuffers() to
*                               ensure that all telemetry packet buffers are
*                               returned to their pool.
*          endRun               - This function starts by calling the parent
*                               endRun(). If any exposures were accumulated
*                               (the exposure count is not zero), the routine
*                               posts the partial histogram to telemetry.
*                               If not, it then calls cleanupBuffers() to
*                               release the data buffers back to their pools
*                               (NOTE: In the code, cleanupBuffers() is called
*                               unconditionally. It will have no ill effect if
*                               the data buffers were posted).
*          waitForPkt           - This function provides an accessor to the
*                               EventHistogram class when it needs to wait
*                               for a telemetry buffer.
*          cacheQuadMode        - This routine obtains and caches a local copy
*                               of the quadrant clocking mode (FULL/DIAG/AC/BD)
*                               in an instance variable.
*          getQuadMode          - This inline routine just returns the cached
*                               quadrant clocking mode.
*          getQuadrant          - This routine maps the passed CCD column position
*                               into the correct CCD quadrant identifier, based
*                               on the quadrant clocking mode.
*          binEvent             - This routine obtains the quadrant identifer
*                               associated with the event's column position,
*                               obtains the event's pulse height, and
*                               invoke's the appropriate histogram's
*                               incrementBin().
*          finishExposure       - This routine is called at the end of
*                               each exposure. It increments the exposure
*                               counter, and if the count reaches the configured
*                               limit for the histograms, sends the accumulated
*                               histograms and resets the counter.
*          setupBuffers         - This routine sets the FEP, CCD, run start time,
*                               and parameter block id for each quadrant
*                               histogram for its FEP, and instructs each
*                               instance to obtain telemetry packet buffers.
*          sendHistograms       - This routine instructs the quadrant
*                               histograms to post their respective telemetry
*                               buffers for transfer to the spacecraft.
*          cleanupBuffers       - This routine instructs the quadrant histograms
*                               to release any held telemetry packet buffers
*                               back to their pool.
*
*       Test1_SmTimedExposure - This is a "friendly" subclass of
*                               SmTimedExposure, and is responsible for
*                               providing replacement functions which dispatch
*                               and configure the new Event Histogram mode.
*                               There are no real instances of this class.
*                               The class is overlayed (there are no additional
*                               instance variables for this class) on top
*                               of the main flight software SmTimedExposure
*                               instance.
*          Test1_SmTimedExposure - This constructor is provided to avoid
```

```
 *                              compiler/linker complaints. It is never invoked.
 *         ~Test1_SmTimedExposure - This destructor is provided to avoid
 *                              compiler/linker errors. It is never invoked.
 *         setupEventHist   - This function configures the system to
 *                            produce Event Histogram data. For each FEP
 *                            configured in the run (i.e. whose CCD is not
 *                            CCD_DESELECT), the function re-initializes
 *                            a corresponding PmTeEvHist instance, configures
 *                            the exposure accumulation limit, and caches
 *                            the quadrant clocking mode.
 *      Tf_Exposure_Te_Ev_Histogram - This subclass of Tf_Exposure_Te_Histogram
 *                            is responsible for formatting the exposure
 *                            record telemetry packets for Event Histogram
 *                            mode.
 *         Tf_Exposure_Te_Ev_Hist - The constructor invokes the parent
 *                              constructor, and then overwrites the telemetry
 *                              format tag. It uses the utility class
 *                              Test_TlmForm to gain access to the tag instance
 *                              variable.
 *     Test_TlmForm         - This "friendly" subclas of TlmForm provides
 *                            inline accessor functions used to modify
 *                            protected/private instance variables of the
 *                            TlmForm class.
 *         setTag           - This function is used to overwrite the telemetry
 *                            format tag of a constructed telemetry formatter
 *                            instance.
 *         releaseBuffer    - This function is used to release the telemetry
 *                            buffer of a packet formatter back to the buffer's
 *                            pool.
 *
 * References:
 *   Refer to the 1.5 release of filesscience/processmode.C,
 *   filesscience/pmevent.C, ipclgen/output/Tf_Data_Te_Hist.C
 *
 * $$Log: pmteevhist.C,v $
 * $Revision 1.18  2003/02/17 02:05:06  pgf
 * $CHange name of test class to differentiate it from other new modes.
 * $
 * $Revision 1.17  2002/06/12 05:25:32  pgf
 * $Update for use with the smtimedlookup patch.
 * $
 * $Revision 1.16  2001/04/05 15:39:34  pgf
 * $Use with the smtimelookup patch.
 * $
 * $Revision 1.15  1999/04/20 12:55:43  jimf
 * $Continue Rev. A comments - Moved Tf_*Ev_Histogram include to *.H
 * $
 * $Revision 1.14  1999/04/20 12:44:44  jimf
 * $Comments from Rev. A - Fix descriptions, add support for correction counter,
 * $moved Tf_Exposure_Te_Ev_Histogram class declaration to header file.
 * $
 * $Revision 1.13  1999/03/08 20:32:32  jimf
 * $Added some documentation
 * $
 * Revision 1.12  1998/11/10  20:37:23  jimf
 * Assign partnumbers and small documentation cleanup.
 *$
 *
 * ============================================================ */

#ifndef private
#define private public
#endif
```

```
#ifndef protected
#define protected public
#endif

#include "ipcl/interface.h"
#include "pmteevhist.H"
#include "filesscience/smtimedexposure.H"
#include "filesswhouse/swhousekeeper.H"
#include "filesfatal/fatalerror.H"

class Test1_SmTimedExposure : public SmTimedExposure
{
public:
    Test1_SmTimedExposure();
    ~Test1_SmTimedExposure();

    void setupEventHist();
};

#ifdef ASSERT
#undef ASSERT
#endif /* ASSERT */

#ifndef NDEBUG
#define ASSERT(expr) if(expr) ; \
                     else \
                     { FatalError fatal; fatal.report(FATAL_UNKNOWN,\
                                                      __LINE__); }
#else
#define ASSERT(expr)
#endif /* if !NDEBUG else */

class Test_TlmForm
{
public:
    inline void setTag(TlmForm& form, TlmFormatTag tag)
    {
        *((TlmFormatTag*)&form.formatTag) = tag;
    };

    inline void releaseBuffer(TlmForm& form)
    {
        if (form.pktPtr != 0)
        {
            form.pktPtr->release();
            form.pktPtr = 0;
        }
    };
};

extern TlmAllocator scienceAllocator;

Tf_Exposure_Te_Ev_Histogram::Tf_Exposure_Te_Ev_Histogram()
    : Tf_Exposure_Te_Histogram()
{
    Test_TlmForm test;
    test.setTag(*this, TTAG_SCI_TE_REC_EV_HIST);
};

void Tf_Exposure_Te_Ev_Histogram::put_Error_Count(unsigned error_count)
{
    Tf_Exposure_Te_Histogram::put_Variance_Overclock_High(error_count);
};
```

```
Tf_Data_Te_Ev_Hist::Tf_Data_Te_Ev_Hist()
    : Tf_Data_Te_Hist()
{
    Test_TlmForm test;
    test.setTag(*this, TTAG_SCI_TE_DAT_EV_HIST);
};

void Tf_Data_Te_Ev_Hist::releaseBuffer()
{
    Test_TlmForm test;
    test.releaseBuffer(*this);
};

// ------------------------------------------------
Test1_SmTimedExposure::Test1_SmTimedExposure()
// ------------------------------------------------
{
};

// ------------------------------------------------
Test1_SmTimedExposure::~Test1_SmTimedExposure()
// ------------------------------------------------
{
};

// ------------------------------------------------
void Test1_SmTimedExposure::setupEventHist()
// ------------------------------------------------
{
    static PmTeEvHist pmTeEvHist[FEP_COUNT];
    unsigned expcount = teBlock.get_Histogram_Count();

    // ---- For each FEP ----
    for (FepId fep = FEP_0; fep < FEP_COUNT; fep = FepId(fep+1))
    {
        // --- If FEP configured to be used ---
        CcdId ccd = fepCcd[fep];

        if (ccd < CCD_DESELECT)
        {
            // --- Ensure initialization from patch ---
            PmTeEvHist* hist = new (&pmTeEvHist[fep]) PmTeEvHist();

            // --- Setup exposure count and event parameters ---
            pmTeEvHist[fep].setExposureCount(expcount);
            setupEventProcess (fep, pmTeEvHist[fep]);

            pmTeEvHist[fep].cacheQuadMode();
        }
    } // END FOR each FEP
};

// ------------------------------------------------
inline
void EventHistogram::setMode(PmTeEvHist& mode)
// ------------------------------------------------
{
    modeptr = &mode;
};

// ------------------------------------------------
inline
void EventHistogram::setFepId(FepId fep)
// ------------------------------------------------
```

```
{
    fepId = fep;
};

// ------------------------------------------------
inline
void EventHistogram::setCcdId(CcdId ccd)
// ------------------------------------------------
{
    ccdId = ccd;
};

// ------------------------------------------------
inline
void EventHistogram::setQuadrant(QuadId quadrant)
// ------------------------------------------------
{
    quadCode = quadrant;
};

// ------------------------------------------------
inline
void EventHistogram::setStartTime(unsigned time)
// ------------------------------------------------
{
  starttime = time;
};

// ------------------------------------------------
inline
void EventHistogram::setPblockId(unsigned blockid)
// ------------------------------------------------
{
  pblockid = blockid;
};

// ------------------------------------------------
Boolean PmTeEvHist::waitForPkt(TlmForm& form)
// ------------------------------------------------
{
    return PmEvent::waitForPkt(form);
};


// ------------------------------------------------
EventHistogram::EventHistogram()
// ------------------------------------------------
    : hasBuffers(BoolFalse),
      modeptr(0),
      fepId(FEP_COUNT),
      ccdId(CCD_COUNT),
      quadCode(ONODE_COUNT),
      error_count(0)
{
};

// ------------------------------------------------
EventHistogram::~EventHistogram()
// ------------------------------------------------
{
    // Packet destructors will release the buffers
};

// ------------------------------------------------
```

```
void EventHistogram::releaseBuffers()
// -------------------------------------------------
{
    // ---- Post all telemetry buffers ----
    for (unsigned ii = 0;
         ii < PACKETS_PER_QUADRANT;
         ii++)
    {
        if (tlmpkt[ii].hasBuffer() == BoolTrue)
        {
            tlmpkt[ii].releaseBuffer();
        }
    }
    hasBuffers = BoolFalse;
};

// -------------------------------------------------
Boolean EventHistogram::postBuffers(unsigned expnum)
// -------------------------------------------------
{
    endexp = expnum;

    // ---- Post all telemetry buffers ----
    {
        unsigned binnumber = 0;
        unsigned binsleft = BINS_PER_QUADRANT;

        for (unsigned ii = 0;
             ii < PACKETS_PER_QUADRANT;
             ii++)
        {
            unsigned bins_in_packet = BINS_PER_PACKET;
            if (bins_in_packet > binsleft)
            {
                bins_in_packet = binsleft;
            }

            if (tlmpkt[ii].hasBuffer() == BoolTrue)
            {
                // --- Fill in header info ---
                tlmpkt[ii].put_Ccd_Id(ccdId);
                tlmpkt[ii].put_Fep_Id(fepId);
                tlmpkt[ii].put_Data_Packet_Number(ii);
                tlmpkt[ii].put_Output_Node_Id(quadCode);
                tlmpkt[ii].put_Starting_Bin(binnumber);
                tlmpkt[ii].set_Bin_Values_Written(bins_in_packet);

                tlmpkt[ii].post();
            }

            binsleft -= bins_in_packet;
            binnumber += bins_in_packet;
        }
        hasBuffers = BoolFalse;
    }

    // ---- Form and emit a Histogram Exposure Record ----
    {
        Tf_Exposure_Te_Ev_Histogram form;

        if (modeptr->waitForPkt (form) == BoolFalse)
        {
            return BoolFalse;
        }
```

```
        form.put_Run_Start_Time (starttime);
        form.put_Parameter_Block_Id (pblockid);
        form.put_Start_Exposure_Number (startexp);
        form.put_End_Exposure_Number (endexp);
        form.put_Exposure_Count (expcnt);
        form.put_Output_Node_Id (quadCode);
        form.put_Ccd_Id (ccdId);
        form.put_Fep_Id (fepId);
        form.put_Minimum_Overclock (0);
        form.put_Fep_Timestamp (fepTime);
        form.put_Maximum_Overclock (0);
        form.put_Mean_Overclock (0);
        form.put_Variance_Overclock_Low (0);

        form.put_Error_Count(error_count);

        form.post();
    }
    return BoolTrue;
};

// ------------------------------------------------
Boolean EventHistogram::incrementBin(BinNumber binnumber)
// ------------------------------------------------
{
    ASSERT(hasBuffers == BoolTrue);

    if (binnumber >= BINS_PER_QUADRANT)
    {
        binnumber = BINS_PER_QUADRANT - 1;
    }

    // ---- Get packet number and address of histogram area ----
    unsigned packetnumber = binnumber / BINS_PER_PACKET;
    ASSERT(packetnumber < PACKETS_PER_QUADRANT);
    ASSERT(tlmpkt[packetnumber].hasData());

    unsigned* binaddress = tlmpkt[packetnumber].get_Bin_Values_Address();
    ASSERT(binaddress != 0);

    // ---- Get bin number ----
    unsigned packetbin = binnumber % BINS_PER_PACKET;
    ASSERT(packetbin < tlmpkt[packetnumber].get_Bin_Values_Avail());

    // ---- Bump bin count ----
    binaddress[packetbin] = hamming.increment(binaddress[packetbin],
                                              error_count);

    // ---- Ok ----
    return BoolTrue;
};

// ------------------------------------------------
Boolean EventHistogram::waitForBuffers(unsigned expnum, unsigned feptime)
// ------------------------------------------------
{
    ASSERT(modeptr != 0);

    if (hasBuffers == BoolFalse)
    {
        // ---- Save the FEP timestamp for the 1st exposure ----
        fepTime = feptime;
```

```
        // ---- Pre-allocate telemetry buffers ----
        unsigned binnumber = 0;
        unsigned binsleft = BINS_PER_QUADRANT;

        for (unsigned ii = 0;
             ii < PACKETS_PER_QUADRANT;
             ii++)
        {
            ASSERT(binsleft != 0);

            // --- Get a buffer ---
            if (modeptr->waitForPkt(tlmpkt[ii]) == BoolFalse)
            {
                return BoolFalse;
            }

            ASSERT(tlmpkt[ii].get_Bin_Values_Avail() >= BINS_PER_PACKET);

            // --- Zero histogram and set # bins in packet ---
            unsigned bins_in_packet = BINS_PER_PACKET;
            if (bins_in_packet > binsleft)
            {
                bins_in_packet = binsleft;
            }

            unsigned* binptr = tlmpkt[ii].get_Bin_Values_Address();

            for (unsigned bin = 0; bin < bins_in_packet; bin++)
            {
                binptr[bin] = 0;
            }

            binsleft -= bins_in_packet;
            binnumber += bins_in_packet;
        }

        // ---- We have buffers ----
        error_count = 0;
        hasBuffers = BoolTrue;
        startexp = expnum;
        expcnt = 0;
    }
    else
    {
        expcnt++;
    }

    return BoolTrue;
};


// ------------------------------------------------
PmTeEvHist::PmTeEvHist()
// ------------------------------------------------
    : exposureLimit(0),
      exposureCount(0)
{
    for (QuadId quadrant = ONODE_A;
         quadrant <= ONODE_D;
         quadrant = QuadId(quadrant+1))
    {
        histogram[quadrant].setMode(*this);
        histogram[quadrant].setQuadrant(quadrant);
    }
```

```
};

// ------------------------------------------------
PmTeEvHist::~PmTeEvHist()
// ------------------------------------------------
{
};

// ------------------------------------------------
Boolean PmTeEvHist::processRecord(const RINGREC& record)
// ------------------------------------------------
{
    Boolean retval = BoolTrue;

    switch (fepRingType(record.data[0]))
    {
    case FEP_EXPOSURE_REC:
        if (PmEvent::processRecord(record) == BoolTrue)
        {
            EventExposure& exp = getExposureInfo();
            retval = setupBuffers(exp.getExposureNumber());
        }
        break;

    case FEP_EVENT_REC_3x3:
    case FEP_EVENT_REC_5x5:
        {
            EventExposure& exp = getExposureInfo();

            Pixel3x3 event;
            event.attachData ((const FEPeventRec3x3*) record.data, &exp);

            if (filterEvent (event) == BoolTrue)
            {
                retval = binEvent(event);
            }
        }
        break;

    default:
        retval = PmEvent::processRecord (record);
        break;
    }

    // ---- If run aborted, cleanup the buffers ----
    if (retval == BoolFalse)
    {
        cleanupBuffers();
    }

    // ---- Return ok or aborted/error ----
    return retval;
};

// ------------------------------------------------
void PmTeEvHist::setExposureCount(unsigned expcount)
// ------------------------------------------------
{
    exposureLimit = expcount;
    exposureCount = 0;
};

// ------------------------------------------------
inline QuadMode PmTeEvHist::getQuadMode()
```

```
// ------------------------------------------------
{
    return cachedQuadMode;
}

// ------------------------------------------------
QuadId PmTeEvHist::getQuadrant(unsigned ccdcolumn)
// ------------------------------------------------
{
    QuadId quadrant;
    unsigned index = ccdcolumn / 256;

    switch(getQuadMode())
    {
    case QUAD_FULL:
    case QUAD_DIAG:
        quadrant = QuadId(index);
        ASSERT((quadrant >= ONODE_A) || (quadrant <= ONODE_D));
        break;

    case QUAD_AC:
        quadrant = QuadId(index & 0x2);
        ASSERT((quadrant == ONODE_A) || (quadrant == ONODE_C));
        break;

    case QUAD_BD:
        quadrant = QuadId(ONODE_B + (index & 0x2));
        ASSERT((quadrant == ONODE_B) || (quadrant == ONODE_D));
        break;

    default:
        quadrant = QuadId(index);
        ASSERT((quadrant >= ONODE_A) || (quadrant <= ONODE_D));
        break;
    }
    return quadrant;
};

// ------------------------------------------------
Boolean PmTeEvHist::setupBuffers(unsigned expnum)
// ------------------------------------------------
{
    Boolean retval = BoolTrue;

    FepId fep = getFepId();
    CcdId ccd = getCcdId();
    unsigned datatime = getTimeData();

    unsigned pblock;
    unsigned dummy;
    getRunIdInfo(dummy, pblock, dummy, dummy, dummy);

    EventExposure& exp = getExposureInfo();
    unsigned feptime = exp.getFepTimestamp();

    for (QuadId ii = ONODE_A;
         ii <= ONODE_D;
         ii = QuadId(ii+1))
    {
        histogram[ii].setFepId(fep);
        histogram[ii].setCcdId(ccd);
        histogram[ii].setStartTime(datatime);
        histogram[ii].setPblockId(pblock);
```

```
        retval = histogram[ii].waitForBuffers(expnum, feptime);
        if (retval == BoolFalse)
            break;
    }
    return retval;
};

// ------------------------------------------------
Boolean PmTeEvHist::binEvent(Pixel3x3& event)
// ------------------------------------------------
{
    // ---- Get bin number ----
    BinNumber bin = event.getPulseHeight();

    // ---- Get quadrant ----
    unsigned row;
    unsigned col;
    event.getCcdPosition(row, col);

    QuadId quadrant = getQuadrant(col);

    // ---- Bin the event into the proper quadrant ----
    Boolean retval = histogram[quadrant].incrementBin(bin);

    return retval;
};

// ------------------------------------------------
Boolean PmTeEvHist::finishExposure()
// ------------------------------------------------
{
    Boolean retval = BoolTrue;

    // ---- Bump exposure counter ----
    exposureCount++;

    // ---- If desired # exposures integrated, flush'em ----
    if (exposureCount >= exposureLimit)
    {
        // ---- Send the histograms to telemetry ----
        EventExposure& exp = getExposureInfo();
        retval = sendHistograms(exp.getExposureNumber());

        // ---- Reset exposure counter ----
        exposureCount = 0;
    }

    return BoolTrue;
};

// ------------------------------------------------
Boolean PmTeEvHist::sendHistograms(unsigned expnum)
// ------------------------------------------------
{
    switch(getQuadMode())
    {
    case QUAD_FULL:
    case QUAD_DIAG:
        histogram[ONODE_A].postBuffers(expnum);
        histogram[ONODE_B].postBuffers(expnum);
        histogram[ONODE_C].postBuffers(expnum);
        histogram[ONODE_D].postBuffers(expnum);
        break;
```

```
    case QUAD_AC:
        histogram[ONODE_A].postBuffers(expnum);
        histogram[ONODE_C].postBuffers(expnum);
        break;

    case QUAD_BD:
        histogram[ONODE_B].postBuffers(expnum);
        histogram[ONODE_D].postBuffers(expnum);
        break;

    default:
        histogram[ONODE_A].postBuffers(expnum);
        histogram[ONODE_B].postBuffers(expnum);
        histogram[ONODE_C].postBuffers(expnum);
        histogram[ONODE_D].postBuffers(expnum);
        break;
    }
    return BoolTrue;
};

// ------------------------------------------------
void PmTeEvHist::cleanupBuffers()
// ------------------------------------------------
{
    for (QuadId ii = ONODE_A;
         ii <= ONODE_D;
         ii = QuadId(ii+1))
    {
        histogram[ii].releaseBuffers();
    }
};

// ------------------------------------------------
void PmTeEvHist::endRun()
// ------------------------------------------------
{
    // ---- Close out parent's state variables ----
    PmEvent::endRun();

    // ---- If got partial histograms, flush them ----
    if (exposureCount != 0)
    {
        // ---- Send the histograms to telemetry ----
        EventExposure& exp = getExposureInfo();
        sendHistograms(exp.getExposureNumber());

        // ---- Reset exposure counter ----
        exposureCount = 0;
    }

    // ---- Ensure all buffers are released ----
    cleanupBuffers();
}

// ------------------------------------------------
void PmTeEvHist::cacheQuadMode()
// ------------------------------------------------
{
    EventExposure& info = getExposureInfo();
    unsigned rowscale = 0;
    unsigned colscale = 0;
    unsigned rowoffset = 0;
    info.getGeometry(rowscale, colscale, rowoffset, cachedQuadMode);
}
```

```
/* ============================================================
 *
 * $$Source: /acis/h3/acisfs/configcntl/patches/eventhist/pmteevhist.H,v $$
 *
 * Patch Name: Process Mode Event Histogram Include File
 *
 * Description:
 *  This defines the classes needed to implement the Event
 *  Histogram Processing mode.
 *
 * References:
 *   Refer to the 1.5 release of filesscience/processmode.H,
 *   filesscience/pmevent.H, ipclgen/output/Tf_Data_Te_Hist.H
 *
 * $$Log: pmteevhist.H,v $
 * $Revision 1.10  1999/04/20 12:55:18  jimf
 * $Continue Rev. A comments - Moved Tf_*Ev_Histogram include over from .C
 * $
 * $Revision 1.9  1999/04/20 12:43:30  jimf
 * $Comments from Rev. A - Add support for correction counter
 * $
 * Revision 1.8  1998/11/10  20:37:24  jimf
 * Assign partnumbers and small documentation cleanup.
 *$
 *
 * ============================================================ */

#ifndef pmteevhist_H
#define pmteevhist_H 1

#include "acis.h"

#include "filesscience/pmevent.H"
#include "filesscience/pixel3x3.H"
#include "ipclgen/output/Tf_Exposure_Te_Histogram.H"
#include "ipclgen/output/Tf_Data_Te_Hist.H"
#include "hamming.H"

class PmTeEvHist;

typedef unsigned BinNumber;

class Tf_Exposure_Te_Ev_Histogram : public Tf_Exposure_Te_Histogram
{
public:
    Tf_Exposure_Te_Ev_Histogram();
    void put_Error_Count(unsigned error_count);
};


class Tf_Data_Te_Ev_Hist : public Tf_Data_Te_Hist
{
public:
    Tf_Data_Te_Ev_Hist();
    void releaseBuffer();
};

class EventHistogram
{
public:
    EventHistogram();
    ~EventHistogram();

    Boolean waitForBuffers(unsigned firstexp, unsigned feptime);
```

```
    Boolean postBuffers(unsigned lastexp);
    void    releaseBuffers();

    Boolean incrementBin(BinNumber binnumber);

    void setMode(PmTeEvHist& mode);
    void setFepId(FepId fep);
    void setCcdId(CcdId ccd);
    void setQuadrant(QuadId quadrant);
    void setStartTime(unsigned time);
    void setPblockId(unsigned blockid);

protected:

private:
    // ---- Define assumed constants - must re-patch if they change!!! ----
    // All units are in terms of 32-bit words
    enum Constants
    {
        PACKET_SIZE             = 512,  // Current Packet Size (words)
        PACKET_HEADER_SIZE      = 4,    // Current Packet Header Size (words)
        BINS_PER_QUADRANT       = 4096, // Supports ADU dynamic range

        BINS_PER_PACKET         =       // Derive # bins/packet
        (PACKET_SIZE - PACKET_HEADER_SIZE),

        PACKETS_PER_QUADRANT    =       // Derive # packets/quadrant
        (BINS_PER_QUADRANT / BINS_PER_PACKET) + 1,
    };

    // ---- Define array of telemetry format pointers ----
    Boolean            hasBuffers;
    PmTeEvHist*        modeptr;
    FepId              fepId;
    CcdId              ccdId;
    QuadId             quadCode;
    unsigned           startexp;
    unsigned           endexp;
    unsigned           expcnt;
    Hamming            hamming;
    unsigned           starttime;
    unsigned           pblockid;
    unsigned           fepTime;
    unsigned           error_count; // # of bit errors during accumulation
    Tf_Data_Te_Ev_Hist tlmpkt[PACKETS_PER_QUADRANT];
};

class PmTeEvHist : public PmEvent
{
public:
    PmTeEvHist();
    virtual ~PmTeEvHist();

    void     setExposureCount(unsigned expcount);
    Boolean  processRecord(const RINGREC& record);

    void     endRun();

    Boolean  waitForPkt(TlmForm& form);

    void     cacheQuadMode();

protected:
    QuadMode getQuadMode();
```

```
    QuadId   getQuadrant(unsigned rawcolumn);

    Boolean  binEvent(Pixel3x3& event);
    Boolean  finishExposure();
    Boolean  setupBuffers(unsigned firstexp);
    Boolean  sendHistograms(unsigned lastexp);
    void     cleanupBuffers();

private:
    unsigned       exposureLimit;
    unsigned       exposureCount;
    QuadMode       cachedQuadMode;
    EventHistogram histogram[ONODE_COUNT];
};


#endif /* pmteevhist_H */
```

```
#! /bin/env expect
#
# $Source: /acis/h3/acisfs/configcntl/patches/eventhist/testsuite/smoke/runtest.tcl,v $
#
# Test of Event in Right Edge of Quadrant -- with eventhist patch
#

puts "Welcome to eventhist/testsuite/smoke/runtest.tcl"

# ---- Launch the command and telemetry server processes ----
set basedir [lindex $argv 0]    ; # patch base directory
set tools [lindex $argv 1]      ; # tool directory
set patchdir [lindex $argv 2]   ; # patches under test
set quad_mode "0 \# QUAD_ABCD"  ; # desired outputRegisterMode
set ccd_list "7 0 1 2 3 6"      ; # desired fepCcdSelect
set last_fep 5                  ; # last FEP read out

# ---- Embed procedure library ----
source $basedir/$tools/lib/lib-exp/runtest_support.tcl

# ---- Start command pipe ----
spawn $basedir/$tools/bin/cmdclient $env(ACISSERVER)
set cmd_id $spawn_id

# ---- Start telemetry pipe ----
spawn $basedir/$tools/bin/tlmclient $env(ACISSERVER)
sleep 1

# ---- Apply patches ----
cold_boot
load_patch_list "$basedir/$tools/share/opt_tlmio.bcmd\
                 $basedir/$tools/share/opt_printswhouse.bcmd\
                 $basedir/$tools/share/opt_dearepl.bcmd\
                 $basedir/$tools/share/standard.bcmd\
                 $basedir/$tools/share/opt_smtimedlookup.bcmd\
                 $basedir/$patchdir/opt_eventhist.bcmd"
warm_boot

# ---- Power on FEPs and CCDs ----
power_on_boards "$ccd_list"

# ---- While powering up, load the Bias Image ----
system "make bias"

# ---- Wait for FEPs to finish powering ----
expect {
    -re ".*SWSTAT_FEP_EXECMEM: $last_fep\[\r\n]" {}
    timeout {}
}

send -i $cmd_id "load 0 te 4 {
    parameterBlockId             = 0x00057014
    fepCcdSelect                 = $ccd_list
    fepMode                      = 2 # FEP_TE_MODE_EV3x3
    bepPackingMode               = 3 # BEP_TE_MODE_EVHIST
    onChip2x2Summing             = 0
    ignoreBadPixelMap            = 1
    ignoreBadColumnMap           = 1
    recomputeBias                = 0
    trickleBias                  = 0
    subarrayStartRow             = 0
    subarrayRowCount             = 1023
    overclockPairsPerNode        = 8
    outputRegisterMode           = $quad_mode
```

```
    ccdVideoResponse              =     0     0     0     0     0     0
    primaryExposure               = 33
    secondaryExposure             = 0
    dutyCycle                     = 0
    fep0EventThreshold            =   100   100   100   100
    fep1EventThreshold            =   100   100   100   100
    fep2EventThreshold            =   100   100   100   100
    fep3EventThreshold            =   100   100   100   100
    fep4EventThreshold            =   100   100   100   100
    fep5EventThreshold            =   100   100   100   100
    fep0SplitThreshold            =    50    50    50    50
    fep1SplitThreshold            =    50    50    50    50
    fep2SplitThreshold            =    50    50    50    50
    fep3SplitThreshold            =    50    50    50    50
    fep5SplitThreshold            =    50    50    50    50
    fep4SplitThreshold            =    50    50    50    50
    fep5SplitThreshold            =    50    50    50    50
    lowerEventAmplitude           = 0
    eventAmplitudeRange           = 65535
    gradeSelections               = 0xffffffff 0xffffffff 0xffffffff 0xffffffff\
                                    0xffffffff 0xffffffff 0xffffffff 0xffffffff
    windowSlotIndex               = 65535
    histogramCount                = 40
    biasCompressionSlotIndex      =     3     3     1     1     1     1
    rawCompressionSlotIndex       = 0
    ignoreInitialFrames           = 2
    biasAlgorithmId               =     1     1     1     1     1     1
    biasArg0                      =     1     1     1     1     1     1
    biasArg1                      =     0     0     0     0     0     0
    biasArg2                      =     0     0     0     0     0     0
    biasArg3                      =    26    26    50    50    50    50
    biasArg4                      =    20    20    20    20    20    20
    fep0VideoOffset               =    65    65    65    65
    fep1VideoOffset               =    65    65    65    65
    fep2VideoOffset               =    65    65    65    65
    fep3VideoOffset               =    65    65    65    65
    fep4VideoOffset               =    65    65    65    65
    fep5VideoOffset               =    65    65    65    65
    deaLoadOverride               = 0
    fepLoadOverride               = 0
}
"
command_echo 1 9 "load te"

# ---- Start the Bias Run ----
send -i $cmd_id "start 0 te bias 4\n"
command_echo 1 15 "start bias run"
set timeout 360
wait_stop_science

# ---- Load the Event Image ----
system "make image"

# ---- Start the Science Run ----
send -i $cmd_id "start 0 te 4\n"
command_echo 1 14 "start science run"

# ---- Wait for Exposure Records ----
set expo1 -1
set events 0

expect {
    -re "exposureTeEvHistogram\[^\r]*\
         startExposureNumber=(\[0-9]+)\[^\r]*\
```

```
         fepId=(\[0-5])\[^\r]*\[\r\n]*" {

        set expo $expect_out(1,string)
        set fep $expect_out(2,string)

        if {$expo1 < 0 && $fep == $last_fep} {
            send -i $cmd_id "stop 0 science\n"
            set expo1 $expo
        } else {
                if {$fep != $last_fep || $expo <= $expo1} {
                    exp_continue
                }
        }
    }

    timeout {
        fail "No exposure record"
    }
}

command_echo 1 19 "stop science"

set timeout 600
science_report 1 "science report"

# ---- Unpack telemetry into event lists ----
system psci -a -B -l foo pkts.raw
system test-hist.pl foo.2.$last_fep.?-*.hist.txt | diff ../hist.txt -
eval exec rm [glob foo.*]

# ---- Report fate ----
pass "histograms were received from FEP_$fep"
```

```
/* ============================================================
 *
 * $$Source: /acis/h3/acisfs/configcntl/patches/reportgrade1/reportgrade1.C,v $$
 *
 * Patch Name: reportgrade1
 *
 * Description:
 *   This patch replaces the filterEvent method of the PmEvent class,
 *   adding calls to swHousekeeper.report to accumulate counts of each
 *   event processed during the housekeeping period, detailing whether
 *   it was filtered out (by event amplitude, grade filter, or window)
 *   or telemetered. Separate counts are kept of 5 particular grades.
 *   Otherwise, the function of PmEvent::filterEvent is inchanged.
 *
 * References:
 *   1.5 release of filesscience/pmevent.C
 *   ACIS IP&CL 36-53204.0204 Revision N
 *   ACIS Engineering Change Order 36-1021.
 *
 * $$Log: reportgrade1.C,v $
 * $Revision 1.3  2003/02/17 02:08:42  pgf
 * $Remove a probe definition.
 * $
 * $Revision 1.2  2000/03/12 19:09:00  pgf
 * $Change SwStatistic enumerations.
 * $Add documentation.
 * $
 * $Revision 1.1  2000/03/09 23:09:32  pgf
 * $Initial release.
 * $$
 *
 * ============================================================ */

#include "ipcl/interface.h"
#include "filesscience/pmevent.H"
#include "filesswhouse/swhousekeeper.H"

class Test_PmEvent : public PmEvent
{
    public:
        ~Test_PmEvent();
        Boolean filterEvent(PixelEvent& event);
};

Test_PmEvent::~Test_PmEvent()
{
}

Boolean Test_PmEvent::filterEvent(PixelEvent& event)
{
    SwFilterId swStat = SW_FILT_NONE;
    Boolean retCode = BoolFalse;

    // ---- Test pulse height range ----
    if (phFilter->filterEvent (event) == BoolFalse)
    {
        swStat = SW_FILT_EVENT;
    }

    // ---- Test event grade ----
    else if (gradeFilter->filterEvent (event) == BoolFalse)
    {
        switch (event.getGrade()) {
        case SW_GRADE_CODE1:
```

```
            swStat = SW_FILT_GRADE1;
            break;
        case SW_GRADE_CODE2:
            swStat = SW_FILT_GRADE2;
            break;
        case SW_GRADE_CODE3:
            swStat = SW_FILT_GRADE3;
            break;
        case SW_GRADE_CODE4:
            swStat = SW_FILT_GRADE4;
            break;
        case SW_GRADE_CODE5:
            swStat = SW_FILT_GRADE5;
            break;
        default:
            swStat = SW_FILT_OTHER;
            break;
        }
    }

    // ---- Test processing windows ----
    else if (windowFilter->filterEvent (event) == BoolFalse)
    {
        swStat = SW_FILT_WIN;
    }

    else retCode = BoolTrue;

    swHousekeeper.report(SW_FILT(swStat, getFepId()),
        getExposureInfo().getExposureNumber());
    return retCode;
}
```

```
/* ============================================================
 *
 * $$Source: /acis/h3/acisfs/configcntl/patches/reportgrade1/reportgrade1a.S,v $$
 *
 * Patch Name: reportgrade1
 *
 * Description:
 *   Add 54 additional slots to the software housekeeping tables to
 *   hold 9 statistics for each of 6 FEPs, as reported by the
 *   reportgrade1 patch.
 *
 * References:
 *   ACIS IP&CL 36-53204.0204 Revision N
 *   ACIS Engineering Change Order 36-1021.
 *
 * $$Log: reportgrade1a.S,v $
 * $Revision 1.3  2003/02/17 02:10:24  pgf
 * $Increase housekeeping index ceiling by 64.
 * $
 * $Revision 1.2  2000/03/12 19:09:00  pgf
 * $Change SwStatistic enumerations.
 * $Add documentation.
 * $
 * $Revision 1.1  2000/03/09 23:09:34  pgf
 * $Initial release.
 * $$
 *
 * ============================================================ */

        .text
        .globl  swhousekeeper_lst_0034_0034
        .ent    swhousekeeper_lst_0034_0034

swhousekeeper_lst_0034_0034:
        li      $2,91+64
        .end    swhousekeeper_lst_0034_0034
```

```
#! /usr/bin/env expect
#
# $Source: /acis/h3/acisfs/configcntl/patches/reportgrade1/testsuite/smoke/runtest.tcl,v $
#
# Test of TE Event Statistic Reportage -- with reportgrade1 patch
#

puts "Welcome to reportgrade1/testsuite/smoke/runtest.tcl"

# ---- Launch the command and telemetry server processes ----
set basedir [lindex $argv 0]    ; # patch base directory
set tools [lindex $argv 1]      ; # tool directory
set patchdir [lindex $argv 2]   ; # patches under test
set quad_mode "0 \# QUAD_ABCD"  ; # desired outputRegisterMode
set ccd_list "7 0 1 2 3 6"      ; # desired fepCcdSelect
set last_fep 5                  ; # last FEP read out

# ---- Embed procedure library ----
source $basedir/$tools/lib/lib-exp/runtest_support.tcl

# ---- Start command pipe ----
spawn $basedir/$tools/bin/cmdclient $env(ACISSERVER)
set cmd_id $spawn_id

# ---- Start telemetry pipe ----
spawn $basedir/$tools/bin/tlmclient $env(ACISSERVER)
sleep 1

# ---- Apply patches ----
cold_boot
load_patch_list "$basedir/$tools/share/opt_tlmio.bcmd\
                 $basedir/$tools/share/opt_printswhouse.bcmd\
                 $basedir/$tools/share/opt_dearepl.bcmd\
                 $basedir/$tools/share/standard.bcmd\
                 $basedir/$patchdir/opt_reportgrade1.bcmd"
warm_boot

# ---- Power on FEPs and CCDs ----
power_on_boards "$ccd_list"

# ---- While powering up, load the Bias Image ----
system "make bias"

# ---- Wait for FEPs to finish powering ----
expect {
    -re ".*SWSTAT_FEP_EXECMEM: $last_fep\[\r\n]" {}
    timeout {}
}

# ---- Load TE 3x3 Mode -----
send -i $cmd_id "load 0 te 4 {
    parameterBlockId             = 0xffffffff
    fepCcdSelect                 = $ccd_list
    fepMode                      = 2 # FEP_TE_MODE_EV3x3
    bepPackingMode               = 1 # BEP_TE_MODE_FAINTBIAS
    onChip2x2Summing             = 0
    ignoreBadPixelMap            = 1
    ignoreBadColumnMap           = 1
    recomputeBias                = 0
    trickleBias                  = 0
    subarrayStartRow             = 0
    subarrayRowCount             = 1023
    overclockPairsPerNode        = 8
    outputRegisterMode           = $quad_mode
```

```
    ccdVideoResponse              =     0     0     0     0     0     0
    primaryExposure               = 33
    secondaryExposure             = 0
    dutyCycle                     = 0
    fep0EventThreshold            =   100   100   100   100
    fep1EventThreshold            =   100   100   100   100
    fep2EventThreshold            =   100   100   100   100
    fep3EventThreshold            =   100   100   100   100
    fep4EventThreshold            =   100   100   100   100
    fep5EventThreshold            =   100   100   100   100
    fep0SplitThreshold            =    50    50    50    50
    fep1SplitThreshold            =    50    50    50    50
    fep2SplitThreshold            =    50    50    50    50
    fep3SplitThreshold            =    50    50    50    50
    fep5SplitThreshold            =    50    50    50    50
    fep4SplitThreshold            =    50    50    50    50
    fep5SplitThreshold            =    50    50    50    50
    lowerEventAmplitude           = 0
    eventAmplitudeRange           = 1500
    gradeSelections               = 0xfeffffff 0xffffffff 0xfffffffb 0xfffff7ff\
                                    0xffffffff 0xffffffff 0xffbfffff 0x7effffff
    windowSlotIndex               = 4
    histogramCount                = 0
    biasCompressionSlotIndex      =     3     3     1     1     1     1
    rawCompressionSlotIndex       = 0
    ignoreInitialFrames           = 2
    biasAlgorithmId               =     1     1     1     1     1     1
    biasArg0                      =     1     1     1     1     1     1
    biasArg1                      =     0     0     0     0     0     0
    biasArg2                      =     0     0     0     0     0     0
    biasArg3                      =    26    26    50    50    50    50
    biasArg4                      =    20    20    20    20    20    20
    fep0VideoOffset               =    65    65    65    65
    fep1VideoOffset               =    65    65    65    65
    fep2VideoOffset               =    65    65    65    65
    fep3VideoOffset               =    65    65    65    65
    fep4VideoOffset               =    65    65    65    65
    fep5VideoOffset               =    65    65    65    65
    deaLoadOverride               = 0
    fepLoadOverride               = 0
}
"
command_echo 1 9 "load te"

# ---- Load TE Window -----
send -i $cmd_id "load 0 window2d 4 {
    windowBlockId               = 0xffffffff
    windows = {
        ccdId                     = 0
        ccdRow                    = 335
        ccdColumn                 = 0
        width                     = 1023
        height                    = 10
        sampleCycle               = 0
        lowerEventAmplitude       = 0
        eventAmplitudeRange       = 65535
    }
    windows = {
        ccdId                     = 1
        ccdRow                    = 335
        ccdColumn                 = 0
        width                     = 1023
        height                    = 10
        sampleCycle               = 0
```

```
    lowerEventAmplitude      = 0
    eventAmplitudeRange      = 65535
}
windows = {
    ccdId                    = 2
    ccdRow                   = 335
    ccdColumn                = 0
    width                    = 1023
    height                   = 10
    sampleCycle              = 0
    lowerEventAmplitude      = 0
    eventAmplitudeRange      = 65535
}
windows = {
    ccdId                    = 3
    ccdRow                   = 335
    ccdColumn                = 0
    width                    = 1023
    height                   = 10
    sampleCycle              = 0
    lowerEventAmplitude      = 0
    eventAmplitudeRange      = 65535
}
windows = {
    ccdId                    = 4
    ccdRow                   = 335
    ccdColumn                = 0
    width                    = 1023
    height                   = 10
    sampleCycle              = 0
    lowerEventAmplitude      = 0
    eventAmplitudeRange      = 65535
}
windows = {
    ccdId                    = 5
    ccdRow                   = 335
    ccdColumn                = 0
    width                    = 1023
    height                   = 10
    sampleCycle              = 0
    lowerEventAmplitude      = 0
    eventAmplitudeRange      = 65535
}
windows = {
    ccdId                    = 6
    ccdRow                   = 335
    ccdColumn                = 0
    width                    = 1023
    height                   = 10
    sampleCycle              = 0
    lowerEventAmplitude      = 0
    eventAmplitudeRange      = 65535
}
windows = {
    ccdId                    = 7
    ccdRow                   = 335
    ccdColumn                = 0
    width                    = 1023
    height                   = 10
    sampleCycle              = 0
    lowerEventAmplitude      = 0
    eventAmplitudeRange      = 65535
}
windows = {
```

```
        ccdId                   = 8
        ccdRow                  = 335
        ccdColumn               = 0
        width                   = 1023
        height                  = 10
        sampleCycle             = 0
        lowerEventAmplitude     = 0
        eventAmplitudeRange     = 65535
    }
    windows = {
        ccdId                   = 9
        ccdRow                  = 335
        ccdColumn               = 0
        width                   = 1023
        height                  = 10
        sampleCycle             = 0
        lowerEventAmplitude     = 0
        eventAmplitudeRange     = 65535
    }
}
"
command_echo 1 11 "load window2d"

# ---- Start the Bias Run ----
send -i $cmd_id "start 0 te bias 4\n"
command_echo 1 15 "start bias run"
set timeout 360
wait_stop_science

# ---- Load the Event Image ----
system "make image"

# ---- Start the Science Run ----
send -i $cmd_id "start 0 te 4\n"
command_echo 1 14 "start science run"

# ---- Wait for Exposure Records ----
set expo -1
set expo1 -1
set events 0
set stats 0
set ended 0

expect {
    -re "exposureTe\[^\r]*\
        exposureNumber=(\[0-9]+)\[^\r]*\
        eventsSent=(\[0-9]+)\[^\r]*\
        discardEventAmplitude=(\[0-9]+)\[^\r]*\
        discardWindow=(\[0-9]+)\[^\r]*\
        discardGrade=(\[0-9]+)\[\r\n]*" {

        set expo $expect_out(1,string)
        set nevt $expect_out(2,string)
        set namp $expect_out(3,string)
        set ngrd $expect_out(4,string)
        set nwin $expect_out(5,string)

        set events [expr $events + $nevt + $namp + $ngrd + $nwin]
        exp_continue
    }

    -re ".*SWSTAT_FILT_FEP_\[^\r\n]*" {
        if {$stats == 0} {
            send -i $cmd_id "stop 0 science\n"
```

```
            incr stats
        }
        exp_continue
    }

    -re "scienceReport.*terminationCode=\[0-9]+\[\r\n]*" {
        set expo1 $expo
        exp_continue
    }

    -re "swHousekeeping\[^\r\n]*" {
        if {$expo1 < 0} {
            exp_continue
        }
        if {$ended == 0} {
            incr ended
            exp_continue
        }
    }

    timeout {
        fail "No exposure record"
    }
}

# inspect packets, accumulate the SWSTAT_FILT_* counts
system "psci -s -p pkts.raw > foo.log"
set fid [open foo.log r]
set stats 0
for {set id 91} {$id<145} {incr id} {set stat($id) 0 ; set title($id) ""}
while {[gets $fid line] >= 0} {
    if {[regexp {^ *swStatisticId *= ([0-9]*) \# (SWSTAT_FILT_.*)$} \
         $line all id name]} {
        if {[gets $fid line] >= 0} {
            if {[regexp {^ *count *\= ([0-9]*).*$} $line all count]} {
                set stat($id) [expr $stat($id) + $count]
                set title($id) $name
                set stats [expr $stats + $count]
            }
        }
    }
}
system "rm -f foo.log"

# examine the individual statistics
foreach id [lsort -integer [array names stat]] {
    set ii [expr ($id - 91) % 9]
    if {($ii == 0 && $stat($id) != 5) ||
        ($ii == 8 && $stat($id) != 1) ||
        ($ii != 0 && $ii != 8 && $stat($id) != 6)} {
        fail "incorrect number of $title($id) messages"
    }
}

# ---- Report fate ----
if {$events > 0 && $events == $stats} {
    pass "$events events through exposure $expo1, $stats correct grade reports"
} else {
    fail "$events events through exposure $expo1, $stats correct grade reports"
}
```

```
#! /usr/bin/env expect
#
# $Source: /acis/h3/acisfs/configcntl/patches/reportgrade1/testsuite/smoke/runtestcc.tcl,v $
#
# Test of CC Event Statistic Reportage -- with reportgrade1 patch
#

puts "Welcome to reportgrade1/testsuite/smoke/runtestcc.tcl"

# ---- Launch the command and telemetry server processes ----
set basedir [lindex $argv 0]    ; # patch base directory
set tools [lindex $argv 1]      ; # tool directory
set patchdir [lindex $argv 2]   ; # patches under test
set quad_mode "0 \# QUAD_ABCD"  ; # desired outputRegisterMode
set ccd_list "7 0 1 2 3 6"      ; # desired fepCcdSelect
set last_fep 5                  ; # last FEP read out

# ---- Embed procedure library ----
source $basedir/$tools/lib/lib-exp/runtest_support.tcl

# ---- Start command pipe ----
spawn $basedir/$tools/bin/cmdclient $env(ACISSERVER)
set cmd_id $spawn_id

# ---- Start telemetry pipe ----
spawn $basedir/$tools/bin/tlmclient $env(ACISSERVER)
sleep 1

# ---- Apply patches ----
cold_boot
load_patch_list "$basedir/$tools/share/opt_tlmio.bcmd\
                 $basedir/$tools/share/opt_printswhouse.bcmd\
                 $basedir/$tools/share/opt_dearepl.bcmd\
                 $basedir/$tools/share/standard.bcmd\
                 $basedir/$tools/share/opt_cc3x3.bcmd\
                 $basedir/$patchdir/opt_reportgrade1.bcmd"
warm_boot

# ---- Power on FEPs and CCDs ----
power_on_boards "$ccd_list"

# ---- While powering up, load the Bias Image ----
system "make bias"

# ---- Wait for FEPs to finish powering ----
expect {
    -re ".*SWSTAT_FEP_EXECMEM: $last_fep\[\r\n]" {}
    timeout {}
}

# ---- Load CC 3x3 Mode -----
send -i $cmd_id "load 0 cc 4 {
    parameterBlockId             = 0xffffffff
    fepCcdSelect                 = $ccd_list
    fepMode                      = 2 # FEP_CC_MODE_EV3x3
    bepPackingMode               = 0 # BEP_CC_MODE_FAINT
    ignoreBadColumnMap           = 1
    recomputeBias                = 0
    trickleBias                  = 0
    rowSum                       = 0
    columnSum                    = 0
    overclockPairsPerNode        = 8
    outputRegisterMode           = $quad_mode
    ccdVideoResponse             =     0     0     0     0     0     0
```

```
    fep0EventThreshold           =   100   100   100   100
    fep1EventThreshold           =   100   100   100   100
    fep2EventThreshold           =   100   100   100   100
    fep3EventThreshold           =   100   100   100   100
    fep4EventThreshold           =   100   100   100   100
    fep5EventThreshold           =   100   100   100   100
    fep0SplitThreshold           =    50    50    50    50
    fep1SplitThreshold           =    50    50    50    50
    fep2SplitThreshold           =    50    50    50    50
    fep3SplitThreshold           =    50    50    50    50
    fep5SplitThreshold           =    50    50    50    50
    fep4SplitThreshold           =    50    50    50    50
    fep5SplitThreshold           =    50    50    50    50
    lowerEventAmplitude          = 0
    eventAmplitudeRange          = 1500
    gradeSelections              = 0x9
    windowSlotIndex              = 4
    rawCompressionSlotIndex      = 0
    ignoreInitialFrames          = 2
    biasAlgorithmId              =     0     0     0     0     0     0
    biasRejection                =     0     0     0     0     0     0
    fep0VideoOffset              =    65    65    65    65
    fep1VideoOffset              =    65    65    65    65
    fep2VideoOffset              =    65    65    65    65
    fep3VideoOffset              =    65    65    65    65
    fep4VideoOffset              =    65    65    65    65
    fep5VideoOffset              =    65    65    65    65
    deaLoadOverride              = 0
    fepLoadOverride              = 0
}
"
command_echo 1 10 "load cc"

# ---- Load CC Window -----
send -i $cmd_id "load 0 window1d 4 {
    windowBlockId              = 0xffffffff
    windows = {
        ccdId                    = 0
        ccdColumn                = 510
        width                    = 0
        sampleCycle              = 0
        lowerEventAmplitude      = 0
        eventAmplitudeRange      = 24570
    }
    windows = {
        ccdId                    = 1
        ccdColumn                = 510
        width                    = 0
        sampleCycle              = 0
        lowerEventAmplitude      = 0
        eventAmplitudeRange      = 24570
    }
    windows = {
        ccdId                    = 2
        ccdColumn                = 510
        width                    = 0
        sampleCycle              = 0
        lowerEventAmplitude      = 0
        eventAmplitudeRange      = 24570
    }
    windows = {
        ccdId                    = 3
        ccdColumn                = 510
        width                    = 0
```

```
        sampleCycle             = 0
        lowerEventAmplitude     = 0
        eventAmplitudeRange     = 24570
    }
    windows = {
        ccdId                   = 4
        ccdColumn               = 510
        width                   = 0
        sampleCycle             = 0
        lowerEventAmplitude     = 0
        eventAmplitudeRange     = 24570
    }
    windows = {
        ccdId                   = 5
        ccdColumn               = 510
        width                   = 0
        sampleCycle             = 0
        lowerEventAmplitude     = 0
        eventAmplitudeRange     = 24570
    }
    windows = {
        ccdId                   = 6
        ccdColumn               = 510
        width                   = 0
        sampleCycle             = 0
        lowerEventAmplitude     = 0
        eventAmplitudeRange     = 24570
    }
    windows = {
        ccdId                   = 7
        ccdColumn               = 510
        width                   = 0
        sampleCycle             = 0
        lowerEventAmplitude     = 0
        eventAmplitudeRange     = 24570
    }
    windows = {
        ccdId                   = 8
        ccdColumn               = 510
        width                   = 0
        sampleCycle             = 0
        lowerEventAmplitude     = 0
        eventAmplitudeRange     = 24570
    }
    windows = {
        ccdId                   = 9
        ccdColumn               = 510
        width                   = 0
        sampleCycle             = 0
        lowerEventAmplitude     = 0
        eventAmplitudeRange     = 24570
    }
}
"
command_echo 1 12 "load window1d"

# ---- Start the Bias Run ----
send -i $cmd_id "start 0 cc bias 4\n"
command_echo 1 17 "start bias run"
set timeout 360
wait_stop_science

# ---- Load the Event Image ----
system "make image"
```

```
# ---- Start the Science Run ----
send -i $cmd_id "start 0 cc 4\n"
command_echo 1 16 "start science run"

# ---- Wait for Exposure Records ----
set expo -1
set expo1 -1
set events 0
set stats 0
set ended 0

expect {
    -re "exposureCc\[^\r]*\
        exposureNumber=(\[0-9]+)\[^\r]*\
        eventsSent=(\[0-9]+)\[^\r]*\
        discardEventAmplitude=(\[0-9]+)\[^\r]*\
        discardWindow=(\[0-9]+)\[^\r]*\
        discardGrade=(\[0-9]+)\[\r\n]*" {

        set expo $expect_out(1,string)
        set nevt $expect_out(2,string)
        set namp $expect_out(3,string)
        set ngrd $expect_out(4,string)
        set nwin $expect_out(5,string)

        set events [expr $events + $nevt + $namp + $ngrd + $nwin]
        exp_continue
    }

    -re ".*SWSTAT_FILT_FEP_\[^\r\n]*" {
        if {$stats == 0} {
            send -i $cmd_id "stop 0 science\n"
            incr stats
        }
        exp_continue
    }

    -re "scienceReport.*terminationCode=\[0-9]+\[\r\n]*" {
        set expo1 $expo
        exp_continue
    }

    -re "swHousekeeping\[^\r\n]*" {
        if {$expo1 < 0} {
            exp_continue
        }
        if {$ended == 0} {
            incr ended
            exp_continue
        }
    }

    timeout {
        fail "No exposure record"
    }
}

# inspect packets, accumulate the SWSTAT_FILT_* counts
system "psci -s -p pkts.raw > foo.log"
set fid [open foo.log r]
set stats 0
for {set id 91} {$id<145} {incr id} {set stat($id) 0 ; set title($id) ""}
while {[gets $fid line] >= 0} {
```

```
    if {[regexp {^ *swStatisticId *= ([0-9]*) \# (SWSTAT_FILT_.*)$} \
         $line all id name]} {
        if {[gets $fid line] >= 0} {
            if {[regexp {^ *count *\= ([0-9]*).*$} $line all count]} {
                set stat($id) [expr $stat($id) + $count]
                set title($id) $name
                set stats [expr $stats + $count]
            }
        }
    }
}
system "rm -f foo.log"

# examine the individual statistics
foreach id [lsort -integer [array names stat]] {
    set ii [expr ($id - 91) % 9]
    if {($ii == 0 && $stat($id) != 5) ||
        ($ii == 8 && $stat($id) != 1) ||
        ($ii != 0 && $ii != 8 && $stat($id) != 6)} {
        fail "incorrect number of $title($id) messages ($stat($id))"
    }
}

# ---- Report fate ----
if {$events > 0 && $events == $stats} {
    pass "$events events through exposure $expo1, $stats correct grade reports"
} else {
    fail "$events events through exposure $expo1, $stats correct grade reports"
}
```

```
/* ============================================================
 *
 * $$Source: /acis/h3/acisfs/configcntl/patches/smtimedlookup/smtimedlookup.C,v $$
 *
 * Patch Name: Table-driven TE mode Source File
 *
 * Description:
 *  This patch replaces three methods of the SmTimedExposure class with
 *  functions that use lookup tables to determine the subsequent science
 *  mode action, based on the method pointers contained in the tables.
 *
 *  This patch makes it much simpler to design and test other optional
 *  patches that add new science modes. It is only necessary for these
 *  patches to replace the appropriate table elements with pointers to
 *  new or replacement methods. The alternative would be for each
 *  combination of optional patches to maintain its own command parsing
 *  logic, which would become increasingly cumbersome as the number of
 *  new modes increased.
 *
 *  The smtimedlookup.C file defines and implements the following:
 *
 *    Test_SmTimedExposure    - a "friendly" subclass of SmTimedExposure
 *                              which provides replacement functions for
 *                              the setupProcess, setupFepBlock and
 *                              terminate methods, using lookup tables.
 *      Test_SmTimedExposure  - This constructor is provided to avoid
 *                              compiler/linker complaints. Never invoked.
 *      ~Test_SmTimedExposure - This constructor is provided to avoid
 *                              compiler/linker complaints. Never invoked.
 *      setupProcess          - This function replaces the
 *                              SmTimedExposure::setupProcess function.
 *                              It extracts the value of the FepMode field
 *                              from the current TE parameter block and
 *                              uses it as an index into the lookup table
 *                              smTimedLookupMode[]. If null, an error is
 *                              posted to S/W housekeeping and Histogram
 *                              mode is assumed. Otherwise, the table
 *                              field is interpreted as the address of a
 *                              function that is called to set up the mode.
 *      setup3x3              - This function performs the set-up for
 *                              all TE 3x3 event modes. Its address is
 *                              provided in the appropriate elements of
 *                              the smTimedLookupMode[] table.
 *      setup5x5              - This function performs the set-up for
 *                              the TE 5x5 event mode. Its address is
 *                              provided in the appropriate element of
 *                              the smTimedLookupMode[] table.
 *      setupFepBlock         - This function replaces the
 *                              SmTimedExposure::setupFepBlock function.
 *                              It extracts the value of the FepMode field
 *                              from the current TE parameter block and
 *                              uses it as an index into the lookup table
 *                              smTimedSetupFep[]. The table field is
 *                              interpreted as the address of a function
 *                              that is called to set up the mode.
 *      setupFepTeRaw         - This function performs the FEP setup for
 *                              TE Raw mode. Its address is provided in
 *                              the appropriate element of the
 *                              smTimedSetupFep[] table.
 *      setupFepTeHist        - This function performs the FEP setup for
 *                              TE Raw Histogram mode. Its address is
 *                              provided in the appropriate element of the
 *                              smTimedSetupFep[] table.
 *      setupFepTe3x3         - This function performs the FEP setup for
```

```
 *                              TE 3x3 event mode. Its address is provided
 *                              in the appropriate element of the
 *                              smTimedSetupFep[] table.
 *      setupFepTe5x5         - This function performs the FEP setup for
 *                              TE 5x5 event mode. Its address is provided
 *                              in the appropriate element of the
 *                              smTimedSetupFep[] table.
 *      terminate             - This function replaces the
 *                              SmTimedExposure::terminate function.
 *                              It extracts the value of the FepMode field
 *                              from the current TE parameter block and
 *                              uses it as an index into the lookup table
 *                              smTimedTerminate[]. The table field is
 *                              interpreted as the address of a function
 *                              that is called to set up the mode.
 *      normalTermination     - This function performs normal post-run
 *                              cleanup for all default TE modes. Its address
 *                              is provided in the appropriate element of the
 *                              smTimedSetupFep[] table.
 *
 *  The following lookup tables are declared static and are not
 *  assigned to a class:
 *
 *    smTimedLookupMode[16]   - functions to perform setup for a science
 *                              mode, indexed by FepMode.
 *    smTimedSetupFep[16]     - functions to perform FEP setup for a science
 *                              mode, indexed by FepMode.
 *    smTimedLookup3x3[16]    - functions to perform setup for a 3x3 event
 *                              science mode, indexed by BepPackingMode.
 *    smTimedLookup5x5[16]    - functions to perform setup for a 5x5 event
 *                              science mode, indexed by BepPackingMode.
 *    smTimedTerminate[16]    - functions to perform post-run processing for
 *                              a science mode, indexed by FepMode.
 *
 * References:
 *   Refer to the 1.5 release of filesscience/smtimedexposure.C
 *
 * $$Log: smtimedlookup.C,v $
 * $Revision 1.6  2002/06/13 17:32:40  pgf
 * $Add documentation.
 * $
 * $Revision 1.5  2001/04/07 02:00:37  pgf
 * $Fix more null tests.
 * $
 * $Revision 1.4  2001/04/06 21:02:54  pgf
 * $Define the constructor.
 * $
 * $Revision 1.3  2001/04/06 20:47:46  pgf
 * $Add dummy constructor.
 * $
 * $Revision 1.2  2001/04/06 05:07:06  pgf
 * $Perform correct test for null method pointer.
 * $
 * $Revision 1.1  2001/04/05 18:43:34  pgf
 * $Initial version for testing.
 * $
 * $Revision 1.1  2001/04/05 15:38:32  pgf
 * $Initial version for testing.
 * $$
 * ========================================================== */

#ifndef private
#define private public
#endif
```

```
#ifndef protected
#define protected public
#endif

#include "ipcl/interface.h"
#include "filesscience/smtimedexposure.H"
#include "filesswhouse/swhousekeeper.H"

extern SmTimedExposure smTimedExposure;

// ###################################################
// ###################################################

class Test_SmTimedExposure : public SmTimedExposure
{
public:
  Test_SmTimedExposure();
  ~Test_SmTimedExposure();
  Boolean setupProcess();
  void setup3x3();
  void setup5x5();
  Boolean setupFepBlock(FepId fep, FEPparmBlock& fepblock);
  void setupFepTeRaw(FepId fep, FEPparmBlock& fepblock);
  void setupFepTeHist(FepId fep, FEPparmBlock& fepblock);
  void setupFepTe3x3(FepId fep, FEPparmBlock& fepblock);
  void setupFepTe5x5(FepId fep, FEPparmBlock& fepblock);
  void terminate();
  void normalTermination();
};

typedef void (Test_SmTimedExposure::* lookupPtr) ();
typedef void (Test_SmTimedExposure::* setupFepPtr) (FepId fep, FEPparmBlock& fepblock);

// -------------------------------------------------
// -- FepMode branch table --
// -------------------------------------------------
static lookupPtr smTimedLookupMode[16] = {
  (lookupPtr) &SmTimedExposure::setupRaw,
  (lookupPtr) &SmTimedExposure::setupHist,
  &Test_SmTimedExposure::setup3x3,
  &Test_SmTimedExposure::setup5x5,
};

// -------------------------------------------------
// -- FepMode branch table for setupFepBlock --
// -------------------------------------------------
static setupFepPtr smTimedSetupFep[16] = {
  &Test_SmTimedExposure::setupFepTeRaw,
  &Test_SmTimedExposure::setupFepTeHist,
  &Test_SmTimedExposure::setupFepTe3x3,
  &Test_SmTimedExposure::setupFepTe5x5,
};

// -------------------------------------------------
// -- BepMode branch table for 3x3 FepMode --
// -------------------------------------------------
static lookupPtr smTimedLookup3x3[16] = {
  (lookupPtr) &SmTimedExposure::setupFaint3x3,
  (lookupPtr) &SmTimedExposure::setupFaintBias3x3,
  (lookupPtr) &SmTimedExposure::setupGraded,
};

// -------------------------------------------------
```

```
// -- BepMode branch table for 5x5 FepMode --
// ------------------------------------------------
static lookupPtr smTimedLookup5x5[16] = {
  (lookupPtr) &SmTimedExposure::setupFaint5x5,
  (lookupPtr) &SmTimedExposure::setupFaintBias3x3,
  (lookupPtr) &SmTimedExposure::setupGraded,
};

// ------------------------------------------------
// -- FepMode branch table for terminate() --
// ------------------------------------------------
static lookupPtr smTimedTerminate[16] = {
  &Test_SmTimedExposure::normalTermination,
  &Test_SmTimedExposure::normalTermination,
  &Test_SmTimedExposure::normalTermination,
  &Test_SmTimedExposure::normalTermination,
};

// ------------------------------------------------
Test_SmTimedExposure::Test_SmTimedExposure()
// ------------------------------------------------
{
};

// ------------------------------------------------
Test_SmTimedExposure::~Test_SmTimedExposure()
// ------------------------------------------------
{
};

// ------------------------------------------------
void Test_SmTimedExposure::setup3x3()
// ------------------------------------------------
{
  // --- Select type of BEP event packing ---
  TeBepMode bepMode = TeBepMode(teBlock.get_Bep_Packing_Mode());

  // -- lookup BEP mode table --
  if (smTimedLookup3x3[bepMode] == lookupPtr(0)) {
    swHousekeeper.report(SWSTAT_TE_BAD_BEP_MODE, bepMode);
    bepMode = BEP_TE_MODE_FAINT;
  }

  // -- set up the BEP --
  (smTimedLookup3x3[bepMode])();
}

// ------------------------------------------------
void Test_SmTimedExposure::setup5x5()
// ------------------------------------------------
{
  // --- Select type of BEP event packing ---
  TeBepMode bepMode = TeBepMode(teBlock.get_Bep_Packing_Mode());

  // -- lookup BEP mode table --
  if (smTimedLookup5x5[bepMode] == lookupPtr(0)) {
    swHousekeeper.report(SWSTAT_TE_BAD_BEP_MODE, bepMode);
    bepMode = BEP_TE_MODE_FAINT;
  }

  // -- set up the BEP --
  (smTimedLookup5x5[bepMode])();
}
```

```
// ------------------------------------------------
Boolean Test_SmTimedExposure::setupProcess()
// ------------------------------------------------
{
  // ---- Govern overall mode by what the FEP should produce ----
  TeFepMode fepMode = TeFepMode(teBlock.get_Fep_Mode());

  // -- lookup FEP mode table --
  if (smTimedLookupMode[fepMode] == lookupPtr(0)) {
    swHousekeeper.report(SWSTAT_TE_BAD_FEP_MODE, fepMode);
    fepMode = FEP_TE_MODE_HIST;
  }

  // -- set up the FEP --
  (smTimedLookupMode[fepMode])();

  // ---- Check if setup parameters in range ----
  Boolean retval = teBlock.isValid();

  // ---- If invalid, log reason for end of run ----
  if (retval == BoolFalse)
    {
      setTerminationReason(SMTERM_PROC_PARM_INVALID);
    }

  return retval;
}

// ------------------------------------------------
void Test_SmTimedExposure::setupFepTeRaw(FepId fep, FEPparmBlock& fepblock)
// ------------------------------------------------
{
  fepblock.type = FEP_TIMED_PARM_RAW;
  fepblock.nhist = 0;
}

// ------------------------------------------------
void Test_SmTimedExposure::setupFepTeHist(FepId fep, FEPparmBlock& fepblock)
// ------------------------------------------------
{
  fepblock.type = FEP_TIMED_PARM_HIST;
  fepblock.nhist = teBlock.get_Histogram_Count();
}

// ------------------------------------------------
void Test_SmTimedExposure::setupFepTe3x3(FepId fep, FEPparmBlock& fepblock)
// ------------------------------------------------
{
  fepblock.type = FEP_TIMED_PARM_3x3;
  fepblock.nhist = 0;
}

// ------------------------------------------------
void Test_SmTimedExposure::setupFepTe5x5(FepId fep, FEPparmBlock& fepblock)
// ------------------------------------------------
{
  fepblock.type = FEP_TIMED_PARM_5x5;
  fepblock.nhist = 0;
}

// ------------------------------------------------
Boolean Test_SmTimedExposure::setupFepBlock(FepId fep, FEPparmBlock& fepblock)
// ------------------------------------------------
{
```

```
  DebugProbe probe;

  TeFepMode fepMode = TeFepMode(teBlock.get_Fep_Mode());

  if (smTimedSetupFep[fepMode] == setupFepPtr(0))
    {
      return BoolFalse;
    }

  (smTimedSetupFep[fepMode])(fep, fepblock);

  unsigned sum = (teBlock.get_On_Chip_2x2_summing() == 0) ? 1 : 2;
  fepblock.nrows = (teBlock.get_Subarray_Row_Count()+1)/sum;
  fepblock.ncols = 1024/sum;

  fepblock.quadcode = FEP_QUAD_ABCD;

  if (teBlock.get_Output_Register_Mode() == QUAD_AC)
    {
      fepblock.quadcode = FEP_QUAD_AC;
      fepblock.ncols /= 2;
    }
  else if (teBlock.get_Output_Register_Mode() == QUAD_BD)
    {
      fepblock.quadcode = FEP_QUAD_BD;
      fepblock.ncols /= 2;
    }
  else
    {
      fepblock.ncols /= 4;
    }

  fepblock.noclk = teBlock.get_Overclock_Pairs_Per_Node() * 2;

  fepblock.btype    = fepBiasType(teBlock.get_Bias_Algorithm_Id(fep));

  fepblock.thresh[0] = fepThresh[fep][0];
  fepblock.thresh[1] = fepThresh[fep][1];
  fepblock.thresh[2] = fepThresh[fep][2];
  fepblock.thresh[3] = fepThresh[fep][3];

  fepblock.bparm[0] = teBlock.get_Bias_Arg_0(fep);
  fepblock.bparm[1] = teBlock.get_Bias_Arg_1(fep);
  fepblock.bparm[2] = teBlock.get_Bias_Arg_2(fep);
  fepblock.bparm[3] = teBlock.get_Bias_Arg_3(fep);
  fepblock.bparm[4] = teBlock.get_Bias_Arg_4(fep);

  fepblock.nskip = teBlock.get_Duty_Cycle();
  fepblock.initskip = 1 + teBlock.get_Ignore_Initial_Frames();

  return BoolTrue;
}

// ------------------------------------------------
void Test_SmTimedExposure::terminate()
// ------------------------------------------------
{
  TeFepMode fepMode = TeFepMode(teBlock.get_Fep_Mode());

  if (smTimedTerminate[fepMode] == lookupPtr(0))
    {
      swHousekeeper.report(SWSTAT_TE_BAD_FEP_MODE, fepMode);
      fepMode = FEP_TE_MODE_HIST;
    }
```

```
  (smTimedTerminate[fepMode])();
}

// ------------------------------------------------
void Test_SmTimedExposure::normalTermination()
// ------------------------------------------------
{
    DebugProbe probe;

    // ---- Flush all exposure records ----
    flushProcesses();

    // ---- Form and issue report ----
    Tf_Science_Report form;

    if (waitForPkt(form) == BoolFalse)
    {
        return;
    }

    setupScienceReport(form);

    form.post();
}
```

```
#! /usr/bin/env expect
#
# $Source: /acis/h3/acisfs/configcntl/patches/smtimedlookup/testsuite/smoke/runtest.tcl,v $
#
# Test of smtimedlookup patch alone
#

puts "Welcome to smtimedlookup/testsuite/smoke/runtest.tcl"

# ---- Launch the command and telemetry server processes ----
set basedir [lindex $argv 0]    ; # patch base directory
set tools [lindex $argv 1]      ; # tool directory
set patchdir [lindex $argv 2]   ; # patches under test
set quad_mode "0 \# QUAD_ABCD"  ; # desired outputRegisterMode
set ccd_list "7 0 1 2 3 6"      ; # desired fepCcdSelect
set last_fep 5                  ; # last FEP read out

# ---- Embed procedure library ----
source $basedir/$tools/lib/lib-exp/runtest_support.tcl

# ---- Start command pipe ----
spawn $basedir/$tools/bin/cmdclient $env(ACISSERVER)
set cmd_id $spawn_id

# ---- Start telemetry pipe ----
spawn $basedir/$tools/bin/tlmclient $env(ACISSERVER)
sleep 1

# ---- Apply patches ----
cold_boot
load_patch_list "$basedir/$tools/share/opt_tlmio.bcmd\
                 $basedir/$tools/share/opt_printswhouse.bcmd\
                 $basedir/$tools/share/opt_dearepl.bcmd\
                 $basedir/$tools/share/standard.bcmd\
                 $basedir/$patchdir/opt_smtimedlookup.bcmd"
warm_boot

# ---- Power on FEPs and CCDs ----
power_on_boards "$ccd_list"

# ---- While powering up, load the Bias Image ----
system make bias

# ---- Wait for FEPs to finish powering ----
expect {
    -re ".*SWSTAT_FEP_EXECMEM: $last_fep\[\r\n]" { sleep 10 }
    timeout {}
}

# ---- Load TE 3x3 CTI1 Mode -----
send -i $cmd_id "load 0 te 4 {
    parameterBlockId             = 0xffffffff
    fepCcdSelect                 = $ccd_list
    fepMode                      = 2 # FEP_TE_MODE_EV3x3
    bepPackingMode               = 0 # BEP_TE_MODE_FAINT
    onChip2x2Summing             = 0
    ignoreBadPixelMap            = 1
    ignoreBadColumnMap           = 1
    recomputeBias                = 0
    trickleBias                  = 0
    subarrayStartRow             = 0
    subarrayRowCount             = 1023
    overclockPairsPerNode        = 8
    outputRegisterMode           = $quad_mode
```

```
    ccdVideoResponse              =     0     0     0     0     0     0
    primaryExposure               = 33
    secondaryExposure             = 0
    dutyCycle                     = 0
    fep0EventThreshold            =   100   100   100   100
    fep1EventThreshold            =   100   100   100   100
    fep2EventThreshold            =   100   100   100   100
    fep3EventThreshold            =   100   100   100   100
    fep4EventThreshold            =   100   100   100   100
    fep5EventThreshold            =   100   100   100   100
    fep0SplitThreshold            =    50    50    50    50
    fep1SplitThreshold            =    50    50    50    50
    fep2SplitThreshold            =    50    50    50    50
    fep3SplitThreshold            =    50    50    50    50
    fep5SplitThreshold            =    50    50    50    50
    fep4SplitThreshold            =    50    50    50    50
    fep5SplitThreshold            =    50    50    50    50
    lowerEventAmplitude           = 0
    eventAmplitudeRange           = 65535
    gradeSelections               = 0xffffffff 0xffffffff 0xffffffff 0xffffffff\
                                    0xffffffff 0xffffffff 0xffffffff 0xffffffff
    windowSlotIndex               = 65535
    histogramCount                = 0
    biasCompressionSlotIndex      =     3     3     1     1     1     1
    rawCompressionSlotIndex       = 0
    ignoreInitialFrames           = 2
    biasAlgorithmId               =     1     1     1     1     1     1
    biasArg0                      =     1     1     1     1     1     1
    biasArg1                      =     0     0     0     0     0     0
    biasArg2                      =     0     0     0     0     0     0
    biasArg3                      =    26    26    50    50    50    50
    biasArg4                      =    20    20    20    20    20    20
    fep0VideoOffset               =    65    65    65    65
    fep1VideoOffset               =    65    65    65    65
    fep2VideoOffset               =    65    65    65    65
    fep3VideoOffset               =    65    65    65    65
    fep4VideoOffset               =    65    65    65    65
    fep5VideoOffset               =    65    65    65    65
    deaLoadOverride               = 0
    fepLoadOverride               = 0
}
"
command_echo 1 9 "load te"

# ---- Start bias-only Science Run ----
send -i $cmd_id "start 0 te bias 4\n"
command_echo 1 15 "start bias-only run"
wait_stop_science

# ---- Load the Event Image ----
system make image

# ---- Start the Science Run ----
send -i $cmd_id "start 0 te 4\n"
command_echo 1 14 "start science run"

# ---- Wait for Exposure Records ----
set timeout 360
expect {
    -re "exposureTe.*\[^\r\n]*" {
        send -i $cmd_id "stop 0 science\n"
    }
    -re "scienceReport\[^\r]*\
        terminationCode=(\[0-9]+)\[\r\n]*" {
```

```
        fail "BEP termination code $expect_out(1,string)"
    }
    timeout { fail "No exposure record" }
}

wait_stop_science

# ---- Report fate ----
pass "science run completed"
```

```
/* ============================================================
 *
 * $$Source: /acis/h3/acisfs/configcntl/patches/untricklebias/untricklebias.C,v $$
 *
 * Patch Name: untricklebias - execute bias thief from science task
 *
 * Description:
 *
 *   For reasons unknown, the BEP has occasionally run the science and
 *   bias thief tasks simultaneously. This causes the FEPs to start
 *   searching for x-ray events while the BEP is copying their bias
 *   maps to telemetry. If the threshold crossing freqency is
 *   sufficiently high, this can trigger an error in the FEP firmware
 *   leading to a "T-plane latchup" condition.
 *
 *   The untricklebias patch prevents this behavior by ensuring that
 *   the FEP bias maps are never accessed by the BiasThief task.
 *   Instead, the science task is given these functions.
 *
 *   The main routine of the bias thief task is repaced by
 *   Test_BiasThief::goTaskEntry, which does nothing beyond waking up
 *   whenever the task monitor tells it to, but goes back to sleep
 *   again immediately.
 *
 *   Where necessary, the remaining BiasThief methods that are called
 *   from the science task are replaced by methods that do not notify
 *   the bias thief task that a change has been made. The
 *   trickleTeBias and trickleCcBias do not need to be patched, but
 *   the checkMonitor method must be replaced with a version that is
 *   appropriate for being called from the science task. Note that
 *   it test the EV_SM_BIAS_ABORT_RUN in the event mask: this is
 *   the value appropriate for a science task abort.
 *
 *   The untricklebias.C file defines and implements the following:
 *
 *     Test_BiasThief           - a subclass of BiasThief which replaces
 *                                most of the methods of that class with
 *                                methods that can be called from the
 *                                science manager task. Only goTaskEntry
 *                                remains to be executed as the bias
 *                                thief task.
 *
 *       Test_BiasThief         - this constructor is provided to avoid
 *                                compiler/linker complaints. Never invoked.
 *
 *       goTaskEntry            - is the only surviving method to be
 *                                run in the Bias Thief task. It does
 *                                nothing except respond to queries from
 *                                the task monitor.
 *
 *       biasReady              - sets the abortFlag and busyFlag, but
 *                                doesn’t notify the bias thief task.
 *
 *       abort                  - sets abortFlag but doesn’t notify the
 *                                bias thief task.
 *
 *       checkMonitor           - checks whether an abort signal or a
 *                                monitor query has been sent to the
 *                                science task. If so, it handles them
 *                                and returns.
 *
 *       doBiasTrickle          - this method is called from the science
 *                                task and calls the existing BiasThief
 *                                tasks for each available FEP.
```

```
 *
 *     Test_ScienceMode         - a subclass of ScienceMode that
 *                                replaces its waitForBiasTrickle method.
 *
 *       waitForBiasTrickle     - this method invokes the doBiasTrickle
 *                                method in the new Test_BiasThief class.
 *
 * References:
 *    Refer to the 1.5 release of filesscience/smtimedexposure.C and to
 *    the method of FEP patching used in the cc3x3 patch.
 *
 * $$Log: untricklebias.C,v $
 * $Revision 1.5  2002/06/13 17:45:49  pgf
 * $Add comments.
 * $
 * $Revision 1.4  2001/04/06 21:04:22  pgf
 * $Define the constructor.
 * $
 * $Revision 1.3  2001/04/06 20:47:10  pgf
 * $Add dummy constructor.
 * $
 * $Revision 1.2  2001/04/06 03:11:32  pgf
 * $Fix bug in global names.
 * $
 * $Revision 1.1  2001/04/05 22:43:43  pgf
 * $Initial test release.
 * $$
 * ========================================================== */

#include "filesscience/sciencemode.H"
#include "filesscience/smtimedexposure.H"
#include "filesscience/smcontclocking.H"
#include "filesswhouse/swhousekeeper.H"

//---------------------------------------------------------------------------
//---------------------------------------------------------------------------

class Test_BiasThief : public BiasThief
{
private:
  enum Test_Events { EV_SM_BIAS_ABORT_RUN = 1 << 1 };
public:
  Test_BiasThief(unsigned taskid);
  virtual void goTaskEntry();
  virtual void biasReady();
  virtual void abort();
  virtual Boolean checkMonitor();
  static Boolean doBiasTrickle(BiasThief& thief);
};

Test_BiasThief::Test_BiasThief(unsigned taskid) : BiasThief(taskid) {}

//---------------------------------------------
void Test_BiasThief::goTaskEntry()
//---------------------------------------------
{
  DebugProbe probe;
  // ---- FOREVER ----
  for (;;)
    {
      // --- Wait for query from task monitor ---
      unsigned caught = waitForEvent (EV_TASKQUERY);
      // --- Respond to monitor queries ---
      if (caught & EV_TASKQUERY)
```

```
        {
          taskMonitor.respond ();
        }
    } // END FOREVER
}

// set flags to indicate bias to be written
//-----------------------------------------------
void Test_BiasThief::biasReady()
//-----------------------------------------------
{
  abortFlag = BoolFalse;
  busyFlag = BoolTrue;
}

// no need to abort the thief
//---------------------------------------
void Test_BiasThief::abort()
//---------------------------------------
{
  abortFlag = BoolFalse;
}

// call this method from time to time to intercept task
// manager polling and aborts.
//-------------------------------------------------
Boolean Test_BiasThief::checkMonitor()
//-------------------------------------------------
{
  Boolean retval = BoolTrue;

  // get science task event mask
  unsigned mask = EV_TASKQUERY | EV_SM_BIAS_ABORT_RUN;
  unsigned caught = taskManager.queryCurrentTask()->requestEvent(mask);
  // respond to task poll
  if (caught & EV_TASKQUERY) {
    taskMonitor.respond ();
  }
  // abort sent by stopScience or RADMON inhibit
  if (caught & EV_SM_BIAS_ABORT_RUN) {
    retval = BoolFalse;
  }
  return retval;
}

// Copy the bias maps to telemetry
//-------------------------------------------------
Boolean Test_BiasThief::doBiasTrickle(BiasThief& thief)
//-------------------------------------------------
{
  Boolean retval = BoolTrue;

  if (thief.isBusy()) {
    // -- Trickle bias for each FEP --
    for (unsigned fepid = 0; fepid < FEP_COUNT; fepid++) {
      // - FEP has no bias -
      if (thief.fepInfo[fepid].base == 0) {
        continue;       // Skip to next FEP
      }
      // - Trickle mode type -
      if (thief.modetype == 0) {        // Timed Exposure
        if (thief.trickleTeBias (FepId(fepid)) == BoolFalse) {
          retval = BoolFalse;
          break;        // Aborted
```

```
        }
      } else {
        // Continuous Clocking
        if (thief.trickleCcBias (FepId(fepid)) == BoolFalse) {
          retval = BoolFalse;
          break;        // Aborted
        }
      } // END IF timed exposure ELSE continuous clocking
    } // END FOR all feps
    // --- No longer busy ---
    thief.busyFlag = BoolFalse;
  } // END IF isBusy()

  return retval;
}

//-------------------------------------------------------------------------
//-------------------------------------------------------------------------

class Test_ScienceMode : public ScienceMode
{
public:
  Boolean waitForBiasTrickle();
};

// Instead of waiting for the biasThief task to complete,
// go and execute it in the current scienceManager task.
//---------------------------------------------------------
Boolean Test_ScienceMode::waitForBiasTrickle()
//---------------------------------------------------------
{
  return Test_BiasThief::doBiasTrickle(biasThief);
}

//-------------------------------------------------------------------------
//-------------------------------------------------------------------------
```

```
#! /bin/env expect
#
# $Source: /acis/h3/acisfs/configcntl/patches/untricklebias/testsuite/fix-hw/runtest.tcl,v $
#
# Test of bias and event processing -- with untricklebias patch
#

puts "Welcome to untricklebias/testsuite/fix-hw/runtest.tcl"

# ---- Launch the command and telemetry server processes ----
set basedir [lindex $argv 0]    ; # patch base directory
set tools [lindex $argv 1]      ; # tool directory
set patchdir [lindex $argv 2]   ; # patches under test
set first_fep 0                 ; # first FEP under test
set last_fep 5                  ; # last FEP under test
set quad_mode "0 \# QUAD_ABCD"  ; # desired outputRegisterMode
set ccd_list "7 5 0 1 2 3"      ; # desired fepCcdSelect

# ---- Embed procedure library ----
source $basedir/$tools/lib/lib-exp/runtest_support.tcl

# ---- Procedure to Wait for Start of bias processing ----
proc wait_bias_start {} {
    set timeout 360
    expect {
        -re "SWSTAT_FEP_STARTBIAS.*\[\r\n]*" {
        }
        -re "dataTeBiasMap.*\[\r\n]*" {
            fail "Bias processing not terminated prematurely"
        }
        -re "exposureTe.*\[\r\n]*" {
            fail "Bias not terminated prematurely"
        }
        -re "scienceReport\[^\r]*\
            terminationCode=(\[0-9]+)\[\r\n]*" {
            fail "BEP termination code $expect_out(1,string)"
        }
        timeout {
            fail "No bias record"
        }
    }
}

# ---- Procedure to Wait for First Bias Record ----
proc wait_bias { fep } {
    set timeout 1000
    expect {
        -re "dataTeBiasMap\[^\r]*\
            fepId=(\[0-9]+)\[^\r]*\
            dataPacketNumber=0\[\r\n]*" {
            if {$expect_out(1,string) != $fep} {
                exp_continue
            }
        }
        -re "exposureTe.*\[\r\n]*" {
            fail "Bias not terminated prematurely"
        }
        -re "scienceReport\[^\r]*\
            terminationCode=(\[0-9]+)\[\r\n]*" {
            fail "BEP termination code $expect_out(1,string)"
        }
        timeout {
            fail "No bias record"
        }
```

```
    }
}

# ---- Procedure to Wait for First Exposure Record ----
proc wait_exp { fep } {
    set timeout 1000
    expect {
        -re "exposureTe\[^\r]*\
            fepId=(\[0-9])\[^\r\n]*" {
            if {$expect_out(1,string) != $fep} {
                exp_continue
            }
        }
        -re "scienceReport\[^\r]*\
            terminationCode=(\[0-9]+)\[\r\n]*" {
            fail "BEP termination code $expect_out(1,string)"
        }
        timeout {
            fail "No exposure record"
        }
    }
}

# ---- Procedure to Wait for stopScience and test its code ----
proc wait_stop_science_and_test { retc } {
    set timeout 360
    set inter 0

    expect {
        -re "dataTeBiasMap\[^\r]*pixelCount=\[0-9]+\[\r\n]*" {
                if {$inter == 1} {
                        fail "Bias packets interleaved with event packets"
                }
                exp_continue
        }
        -re "exposureTe\[^\r]*discardGrade=\[0-9]+\[\r\n]*" {
                set inter 1
                exp_continue
        }
        -re "dataTe\[^\r]*dataPacketNumber\[0-9]+\[\r\n]*" {
                set inter 1
                exp_continue
        }
        -re "scienceReport\[^\r]*terminationCode=(\[0-9]+)\[\r\n]*" {
            set rc $expect_out(1,string)
            if {$rc != $retc} {
                fail "Unexpected terminationCode $rc (expected $retc)"
            }
        }
        timeout {}
    }
}

# ---- Procedure to Assert RADMON, Verify it, Deassert it, Verify it.
proc toggle_radmon { retc } {
    global cmd_id
    send -i $cmd_id "set radiationmonitor high\n"

    # ---- Wait for RADMON Confirmation ----
    expect {
        -re "SWSTAT_SCI_INHIBIT_ON.*\[\r\n]*" {}
        timeout {fail "No RADMON ENABLE confirmation"}
    }
```

```
    # ---- Wait for the scienceReport
    wait_stop_science_and_test $retc

    # ---- Restart the Suspended Science Run ----
    send -i $cmd_id "set radiationmonitor low\n"
    expect {
        -re "SWSTAT_SCI_INHIBIT_OFF.*\[\r\n]*" {}
        timeout {fail "No RADMON DISABLE confirmation"}
    }
}

# ---- Start command pipe ----
spawn $basedir/$tools/bin/cmdclient $env(ACISSERVER)
set cmd_id $spawn_id

# ---- Start telemetry pipe ----
spawn $basedir/$tools/bin/tlmclient $env(ACISSERVER)
sleep 1

# ---- Select Input from Image Loader ----
system make loaderselect

# ---- Apply patches ----
cold_boot
load_patch_list "$basedir/$tools/share/opt_tlmio.bcmd\
                 $basedir/$tools/share/opt_printswhouse.bcmd\
                 $basedir/$tools/share/opt_dearepl.bcmd\
                 $basedir/$tools/share/standard.bcmd\
                 $basedir/$patchdir/opt_untricklebias.bcmd"
warm_boot

# ---- Power on FEPs and CCDs ----
power_on_boards "$ccd_list"

# ---- Wait for FEPs to finish powering ----
expect {
    -re ".*SWSTAT_FEPMAN_ENDLOAD: $last_fep\[\r\n]*" {}
    timeout { fail "Power-up Failure" }
}

# ---- Load Pblock for Faint Timed-Exposure Mode ----
send -i $cmd_id "load 0 te 4 {
    parameterBlockId             = 0x00000014
    fepCcdSelect                 = $ccd_list
    fepMode                      = 2 # FEP_TE_MODE_EV3x3
    bepPackingMode               = 2 # BEP_TE_MODE_GRADED
    onChip2x2Summing             = 0
    ignoreBadPixelMap            = 0
    ignoreBadColumnMap           = 0
    recomputeBias                = 1
    trickleBias                  = 1
    subarrayStartRow             = 0
    subarrayRowCount             = 99
    overclockPairsPerNode        = 8
    outputRegisterMode           = $quad_mode
    ccdVideoResponse             =     0     0     0     0     0     0
    primaryExposure              = 5
    secondaryExposure            = 0
    dutyCycle                    = 0
    fep0EventThreshold           =   100   100   100   100
    fep1EventThreshold           =   100   100   100   100
    fep2EventThreshold           =   100   100   100   100
    fep3EventThreshold           =   100   100   100   100
    fep4EventThreshold           =   100   100   100   100
```

```
    fep5EventThreshold           =   100   100   100   100
    fep0SplitThreshold           =    50    50    50    50
    fep1SplitThreshold           =    50    50    50    50
    fep2SplitThreshold           =    50    50    50    50
    fep3SplitThreshold           =    50    50    50    50
    fep5SplitThreshold           =    50    50    50    50
    fep4SplitThreshold           =    50    50    50    50
    fep5SplitThreshold           =    50    50    50    50
    lowerEventAmplitude          = 0
    eventAmplitudeRange          = 65535
    gradeSelections              = 0xffffffff 0xffffffff 0xffffffff 0xffffffff\
                                   0xffffffff 0xffffffff 0xffffffff 0xffffffff
    windowSlotIndex              = 65535
    histogramCount               = 0
    biasCompressionSlotIndex     =     3     3     1     1     1     1
    rawCompressionSlotIndex      = 0
    ignoreInitialFrames          = 2
    biasAlgorithmId              =     1     1     1     1     1     1
    biasArg0                     =     1     1     1     1     1     1
    biasArg1                     =     0     0     0     0     0     0
    biasArg2                     =     0     0     0     0     0     0
    biasArg3                     =    26    26    50    50    50    50
    biasArg4                     =    20    20    20    20    20    20
    fep0VideoOffset              =    65    65    65    65
    fep1VideoOffset              =    65    65    65    65
    fep2VideoOffset              =    65    65    65    65
    fep3VideoOffset              =    65    65    65    65
    fep4VideoOffset              =    65    65    65    65
    fep5VideoOffset              =    65    65    65    65
    deaLoadOverride              = 0
    fepLoadOverride              = 0
}
"
command_echo 1 9 "load te"

# ---- Test 1: stopScience during bias map creation ----
system make bias RUN=1
send -i $cmd_id "start 0 te 4\n"
command_echo 1 14 "start science run"
wait_bias_start
send -i $cmd_id "stop 0 science\n"
wait_stop_science_and_test 1

# ---- Test 2: double stopScience during bias map creation ----
system make bias RUN=2
send -i $cmd_id "start 0 te 4\n"
command_echo 1 14 "start science run"
wait_bias_start
send -i $cmd_id "stop 0 science\n"
sleep 1
send -i $cmd_id "stop 0 science\n"
wait_stop_science_and_test 1

# ---- Test 3: startScience during bias map creation ----
system make bias RUN=3
send -i $cmd_id "start 0 te 4\n"
command_echo 1 14 "start science run"
wait_bias_start
send -i $cmd_id "start 0 te 4\n"
command_echo 1 14 "start science run"
wait_stop_science_and_test 6
wait_bias $first_fep
system make image
wait_exp $last_fep
```

```
send -i $cmd_id "stop 0 science\n"
wait_stop_science_and_test 1

# ---- Test 4: assert/deassert RADMON during bias map creation ----
system make bias RUN=4
send -i $cmd_id "start 0 te 4\n"
command_echo 1 14 "start science run"
wait_bias_start
toggle_radmon 3
wait_bias $first_fep
system make image
wait_exp $last_fep
send -i $cmd_id "stop 0 science\n"
wait_stop_science_and_test 1

# ---- Test 5: stopScience during bias map telemetering ----
system make bias RUN=5
send -i $cmd_id "start 0 te 4\n"
command_echo 1 14 "start science run"
wait_bias $first_fep
send -i $cmd_id "stop 0 science\n"
wait_stop_science_and_test 1

# ---- Test 6: double stopScience during bias map telemetering ----
system make bias RUN=6
send -i $cmd_id "start 0 te 4\n"
command_echo 1 14 "start science run"
wait_bias $first_fep
send -i $cmd_id "stop 0 science\n"
sleep 1
send -i $cmd_id "stop 0 science\n"
wait_stop_science_and_test 1

# ---- Test 7: startScience during bias map telemetering ----
system make bias RUN=7
send -i $cmd_id "start 0 te 4\n"
command_echo 1 14 "start science run"
wait_bias $first_fep
send -i $cmd_id "start 0 te 4\n"
command_echo 1 14 "start science run"
wait_stop_science_and_test 4
wait_bias $first_fep
system make image
wait_exp $last_fep
send -i $cmd_id "stop 0 science\n"
wait_stop_science_and_test 1

# ---- Test 8: assert/deassert RADMON during bias map telemetering ----
system make bias RUN=8
send -i $cmd_id "start 0 te 4\n"
command_echo 1 14 "start science run"
wait_bias $first_fep
toggle_radmon 3
wait_bias $first_fep
system make image
wait_exp $last_fep
send -i $cmd_id "stop 0 science\n"
wait_stop_science_and_test 1

# ---- Test 9: stopScience during event processing ----
system make bias RUN=9
send -i $cmd_id "start 0 te 4\n"
command_echo 1 14 "start science run"
wait_bias $first_fep
```

```
system make image
wait_exp $last_fep
send -i $cmd_id "stop 0 science\n"
wait_stop_science_and_test 1

# ---- Test 10: double stopScience during event processing ----
system make bias RUN=10
send -i $cmd_id "start 0 te 4\n"
command_echo 1 14 "start science run"
wait_bias $first_fep
system make image
wait_exp $last_fep
send -i $cmd_id "stop 0 science\n"
sleep 1
send -i $cmd_id "stop 0 science\n"
wait_stop_science_and_test 1

# ---- Test 11: startScience during event processing ----
system make bias RUN=11a
send -i $cmd_id "start 0 te 4\n"
command_echo 1 14 "start science run"
wait_bias $first_fep
system make image
wait_exp $last_fep
system make bias RUN=11b
send -i $cmd_id "start 0 te 4\n"
command_echo 1 14 "start science run"
wait_stop_science_and_test 15
wait_bias $first_fep
system make image
wait_exp $last_fep
send -i $cmd_id "stop 0 science\n"
wait_stop_science_and_test 1

# ---- Test 12: assert/deassert RADMON during event processing ----
system make bias RUN=12a
send -i $cmd_id "start 0 te 4\n"
command_echo 1 14 "start science run"
wait_bias $first_fep
system make image
wait_exp $last_fep
system make bias RUN=12b
toggle_radmon 15
wait_bias $first_fep
system make image
wait_exp $last_fep
send -i $cmd_id "stop 0 science\n"
wait_stop_science_and_test 1

# ---- Report fate ----
pass "all tests successful"
```

```
#! /bin/env expect
#
# $Source: /acis/h3/acisfs/configcntl/patches/untricklebias/testsuite/fix-hw/runtestcc.tcl,v
 $
#
# Test of CC bias and event processing -- with untricklebias patch
#

puts "Welcome to untricklebias/testsuite/fix-hw/runtestcc.tcl"

# ---- Launch the command and telemetry server processes ----
set basedir [lindex $argv 0]    ; # patch base directory
set tools [lindex $argv 1]      ; # tool directory
set patchdir [lindex $argv 2]   ; # patches under test
set first_fep 0                 ; # first FEP under test
set last_fep 5                  ; # last FEP under test
set quad_mode "0 \# QUAD_ABCD"  ; # desired outputRegisterMode
set ccd_list "7 5 0 1 2 3"      ; # desired fepCcdSelect

# ---- Embed procedure library ----
source $basedir/$tools/lib/lib-exp/runtest_support.tcl

# ---- Wait for FEPs to finish powering ----
proc wait_fepload { fep } {
    set timeout 360
    expect {
        -re ".*SWSTAT_FEPMAN_ENDLOAD: $fep\[\r\n]*" {}
        timeout { fail "Power-up Failure" }
    }
}

# ---- Procedure to Wait for Start of bias processing ----
proc wait_bias_start {} {
    set timeout 360
    expect {
        -re "SWSTAT_FEP_STARTBIAS.*\[\r\n]*" {
        }
        -re "dataCcBiasMap.*\[\r\n]*" {
            fail "Bias processing not terminated prematurely"
        }
        -re "exposureCc.*\[\r\n]*" {
            fail "Bias not terminated prematurely"
        }
        -re "scienceReport\[^\r]*\
            terminationCode=(\[0-9]+)\[\r\n]*" {
            fail "BEP termination code $expect_out(1,string)"
        }
        timeout {
            fail "No bias record"
        }
    }
}

# ---- Procedure to Wait for First Bias Record ----
proc wait_bias { fep } {
    set timeout 1000
    expect {
        -re "dataCcBiasMap\[^\r]*\
            fepId=$fep\[^\r\n]*" {
        }
        -re "exposureCc.*\[\r\n]*" {
            fail "Bias not terminated prematurely"
        }
        -re "scienceReport\[^\r]*\
```

```
            terminationCode=(\[0-9]+)\[\r\n]*" {
            fail "BEP termination code $expect_out(1,string)"
        }
        timeout {
            fail "No bias record"
        }
    }
}

# ---- Procedure to Wait for First Exposure Record ----
proc wait_exp { fep } {
    set timeout 1000
    expect {
        -re "exposureCc\[^\r]*\
            fepId=$fep\[^\r\n]*" {
        }
        -re "scienceReport\[^\r]*\
            terminationCode=(\[0-9]+)\[\r\n]*" {
            fail "BEP termination code $expect_out(1,string)"
        }
        timeout {
            fail "No exposure record"
        }
    }
}

# ---- Procedure to Wait for stopScience and test its code ----
proc wait_stop_science_and_test { retc1 retc2 } {
    set timeout 360
    set inter 0

    expect {
        -re "dataCcBiasMap\[^\r]*pixelCount=\[0-9]+\[\r\n]*" {
                if {$inter == 1} {
                        fail "Bias packets interleaved with event packets"
                }
                exp_continue
        }
        -re "exposureCc\[^\r]*discardGrade=\[0-9]+\[\r\n]*" {
                set inter 1
                exp_continue
        }
        -re "dataCc\[^\r]*dataPacketNumber\[0-9]+\[\r\n]*" {
                set inter 1
                exp_continue
        }
        -re "scienceReport\[^\r]*terminationCode=(\[0-9]+)\[\r\n]*" {
            set rc $expect_out(1,string)
            if {$rc != $retc1 && $rc != $retc2} {
                fail "Unexpected terminationCode $rc (expected $retc1)"
            }
        }
        timeout {}
    }
}

# ---- Procedure to Assert RADMON, Verify it, Deassert it, Verify it.
proc toggle_radmon { retc } {
    global cmd_id
    send -i $cmd_id "set radiationmonitor high\n"

    # ---- Wait for RADMON Confirmation ----
    expect {
        -re "SWSTAT_SCI_INHIBIT_ON.*\[\r\n]*" {}
```

```
        timeout {fail "No RADMON ENABLE confirmation"}
    }

    # ---- Wait for the scienceReport
    wait_stop_science_and_test 3 $retc

    # ---- Restart the Suspended Science Run ----
    send -i $cmd_id "set radiationmonitor low\n"
    expect {
        -re "SWSTAT_SCI_INHIBIT_OFF.*\[\r\n]*" {}
        timeout {fail "No RADMON DISABLE confirmation"}
    }
}

# ---- Start command pipe ----
spawn $basedir/$tools/bin/cmdclient $env(ACISSERVER)
set cmd_id $spawn_id

# ---- Start telemetry pipe ----
spawn $basedir/$tools/bin/tlmclient $env(ACISSERVER)
sleep 1

# ---- Select Input from Image Loader ----
system make loaderselect

# ---- Apply patches ----
cold_boot
load_patch_list "$basedir/$tools/share/opt_tlmio.bcmd\
                 $basedir/$tools/share/opt_printswhouse.bcmd\
                 $basedir/$tools/share/opt_dearepl.bcmd\
                 $basedir/$tools/share/standard.bcmd\
                 $basedir/$tools/share/opt_cc3x3.bcmd\
                 $basedir/$patchdir/opt_untricklebias.bcmd"
warm_boot

# ---- Power on FEPs and CCDs ----
power_on_boards "$ccd_list"
wait_fepload $last_fep

# ---- Load Pblock for Faint Continuously-Clocked Mode ----
send -i $cmd_id "load 0 cc 4 {
    parameterBlockId             = 0x00000014
    fepCcdSelect                 = $ccd_list
    fepMode                      = 2 # FEP_CC_MODE_EV3x3
    bepPackingMode               = 0 # BEP_CC_MODE_FAINT
    ignoreBadColumnMap           = 0
    recomputeBias                = 1
    trickleBias                  = 1
    rowSum                       = 0
    columnSum                    = 0
    overclockPairsPerNode        = 8
    outputRegisterMode           = $quad_mode
    ccdVideoResponse             =     0     0     0     0     0     0
    fep0EventThreshold           =   100   100   100   100
    fep1EventThreshold           =   100   100   100   100
    fep2EventThreshold           =   100   100   100   100
    fep3EventThreshold           =   100   100   100   100
    fep4EventThreshold           =   100   100   100   100
    fep5EventThreshold           =   100   100   100   100
    fep0SplitThreshold           =    50    50    50    50
    fep1SplitThreshold           =    50    50    50    50
    fep2SplitThreshold           =    50    50    50    50
    fep3SplitThreshold           =    50    50    50    50
    fep5SplitThreshold           =    50    50    50    50
```

```
    fep4SplitThreshold          =    50    50    50    50
    fep5SplitThreshold          =    50    50    50    50
    lowerEventAmplitude         = 0
    eventAmplitudeRange         = 24570
    gradeSelections             = 0xf
    windowSlotIndex             = 65535
    rawCompressionSlotIndex     = 0
    ignoreInitialFrames         = 2
    biasAlgorithmId             =     0     0     0     0     0     0
    biasRejection               =     0     0     0     0     0     0
    fep0VideoOffset             =    65    65    65    65
    fep1VideoOffset             =    65    65    65    65
    fep2VideoOffset             =    65    65    65    65
    fep3VideoOffset             =    65    65    65    65
    fep4VideoOffset             =    65    65    65    65
    fep5VideoOffset             =    65    65    65    65
    deaLoadOverride             = 0
    fepLoadOverride             = 0
}
"
command_echo 1 10 "load cc"

# ---- Test 1: stopScience during bias map creation ----
system make ccbias RUN=1
send -i $cmd_id "start 0 cc 4\n"
command_echo 1 16 "start science run"
wait_bias_start
send -i $cmd_id "stop 0 science\n"
wait_stop_science_and_test 1 1

# ---- Test 2: double stopScience during bias map creation ----
system make ccbias RUN=2
send -i $cmd_id "start 0 cc 4\n"
command_echo 1 16 "start science run"
wait_bias_start
send -i $cmd_id "stop 0 science\n"
sleep 1
send -i $cmd_id "stop 0 science\n"
wait_stop_science_and_test 1 15

# ---- Test 3: startScience during bias map creation ----
system make ccbias RUN=3
send -i $cmd_id "start 0 cc 4\n"
command_echo 1 16 "start science run"
wait_bias_start
send -i $cmd_id "start 0 cc 4\n"
command_echo 1 16 "start science run"
wait_stop_science_and_test 6 6
system make ccimage
wait_fepload $last_fep
wait_exp $last_fep
send -i $cmd_id "stop 0 science\n"
wait_stop_science_and_test 1 1

# ---- Test 4: assert/deassert RADMON during bias map creation ----
system make ccbias RUN=4
send -i $cmd_id "start 0 cc 4\n"
command_echo 1 16 "start science run"
wait_bias_start
toggle_radmon 6
wait_bias $first_fep
system make ccimage
wait_exp $last_fep
send -i $cmd_id "stop 0 science\n"
```

```
wait_stop_science_and_test 1 1

# ---- Test 5: stopScience during bias map telemetering ----
system make ccbias RUN=5
send -i $cmd_id "start 0 cc 4\n"
command_echo 1 16 "start science run"
wait_bias $first_fep
send -i $cmd_id "stop 0 science\n"
wait_stop_science_and_test 1 1

# ---- Test 6: double stopScience during bias map telemetering ----
system make ccbias RUN=6
send -i $cmd_id "start 0 cc 4\n"
command_echo 1 16 "start science run"
wait_bias $first_fep
send -i $cmd_id "stop 0 science\n"
sleep 1
send -i $cmd_id "stop 0 science\n"
wait_stop_science_and_test 1 15

# ---- Test 7: startScience during bias map telemetering ----
system make ccbias RUN=7
send -i $cmd_id "start 0 cc 4\n"
command_echo 1 16 "start science run"
wait_bias $first_fep
send -i $cmd_id "start 0 cc 4\n"
command_echo 1 16 "start science run"
wait_stop_science_and_test 6 6
system make ccimage
wait_fepload $last_fep
wait_exp $last_fep
send -i $cmd_id "stop 0 science\n"
wait_stop_science_and_test 1 1

# ---- Test 8: assert/deassert RADMON during bias map telemetering ----
system make ccbias RUN=8
send -i $cmd_id "start 0 cc 4\n"
command_echo 1 16 "start science run"
wait_bias $first_fep
toggle_radmon 6
wait_bias $first_fep
system make ccimage
wait_exp $last_fep
send -i $cmd_id "stop 0 science\n"
wait_stop_science_and_test 1 1

# ---- Test 9: stopScience during event processing ----
system make ccbias RUN=9
send -i $cmd_id "start 0 cc 4\n"
command_echo 1 16 "start science run"
wait_bias $first_fep
system make ccimage
wait_exp $last_fep
send -i $cmd_id "stop 0 science\n"
wait_stop_science_and_test 1 1

# ---- Test 10: double stopScience during event processing ----
system make ccbias RUN=10
send -i $cmd_id "start 0 cc 4\n"
command_echo 1 16 "start science run"
wait_bias $first_fep
system make ccimage
wait_exp $last_fep
send -i $cmd_id "stop 0 science\n"
```

```
sleep 1
send -i $cmd_id "stop 0 science\n"
wait_stop_science_and_test 1 15

# ---- Test 11: startScience during event processing ----
system make ccbias RUN=11a
send -i $cmd_id "start 0 cc 4\n"
command_echo 1 16 "start science run"
wait_bias $first_fep
system make ccimage
wait_exp $last_fep
system make ccbias RUN=11b
send -i $cmd_id "start 0 cc 4\n"
command_echo 1 16 "start science run"
wait_stop_science_and_test 15 15
system make ccimage
wait_fepload $last_fep
wait_exp $last_fep
send -i $cmd_id "stop 0 science\n"
wait_stop_science_and_test 1 1

# ---- Test 12: assert/deassert RADMON during event processing ----
system make ccbias RUN=12a
send -i $cmd_id "start 0 cc 4\n"
command_echo 1 16 "start science run"
wait_bias $first_fep
system make ccimage
wait_exp $last_fep
system make ccbias RUN=12b
toggle_radmon 15
wait_bias $first_fep
system make ccimage
wait_exp $last_fep
send -i $cmd_id "stop 0 science\n"
wait_stop_science_and_test 1 1

# ---- Report fate ----
pass "all tests successful"
```

```
#! /bin/env expect
#
# $Source: /acis/h3/acisfs/configcntl/patches/untricklebias/testsuite/fix-hw/runtestall.tcl,
v $
#
# Test of bias and event processing -- with untricklebias patch
#

puts "Welcome to untricklebias/testsuite/fix-hw/runtestall.tcl"

# ---- Launch the command and telemetry server processes ----
set basedir [lindex $argv 0]    ; # patch base directory
set tools [lindex $argv 1]      ; # tool directory
set patchdir [lindex $argv 2]   ; # patches under test
set first_fep 0                 ; # first FEP under test
set last_fep 5                  ; # last FEP under test
set quad_mode "0 \# QUAD_ABCD"  ; # desired outputRegisterMode
set ccd_list "7 5 0 1 2 3"      ; # desired fepCcdSelect

# ---- Embed procedure library ----
source $basedir/$tools/lib/lib-exp/runtest_support.tcl

# ---- Procedure to Wait for Start of bias processing ----
proc wait_bias_start {} {
    set timeout 360
    expect {
        -re "SWSTAT_FEP_STARTBIAS.*\[\r\n]*" {
        }
        -re "dataTeBiasMap.*\[\r\n]*" {
            fail "Bias processing not terminated prematurely"
        }
        -re "exposureTe.*\[\r\n]*" {
            fail "Bias not terminated prematurely"
        }
        -re "scienceReport\[^\r]*\
            terminationCode=(\[0-9]+)\[\r\n]*" {
            fail "BEP termination code $expect_out(1,string)"
        }
        timeout {
            fail "No bias record"
        }
    }
}

# ---- Procedure to Wait for First Bias Record ----
proc wait_bias { fep } {
    set timeout 1000
    expect {
        -re "dataTeBiasMap\[^\r]*\
            fepId=(\[0-9]+)\[^\r]*\
            dataPacketNumber=0\[\r\n]*" {
            if {$expect_out(1,string) != $fep} {
                exp_continue
            }
        }
        -re "exposureTe.*\[\r\n]*" {
            fail "Bias not terminated prematurely"
        }
        -re "scienceReport\[^\r]*\
            terminationCode=(\[0-9]+)\[\r\n]*" {
            fail "BEP termination code $expect_out(1,string)"
        }
        timeout {
            fail "No bias record"
```

```
        }
    }
}

# ---- Procedure to Wait for First Exposure Record ----
proc wait_exp { fep } {
    set timeout 1000
    expect {
        -re "exposureTe\[^\r]*\
            fepId=(\[0-9])\[^\r\n]*" {
            if {$expect_out(1,string) != $fep} {
                exp_continue
            }
        }
        -re "scienceReport\[^\r]*\
            terminationCode=(\[0-9]+)\[\r\n]*" {
            fail "BEP termination code $expect_out(1,string)"
        }
        timeout {
            fail "No exposure record"
        }
    }
}

# ---- Procedure to Wait for stopScience and test its code ----
proc wait_stop_science_and_test { retc } {
    set timeout 360
    set inter 0

    expect {
        -re "dataTeBiasMap\[^\r]*pixelCount=\[0-9]+\[\r\n]*" {
                if {$inter == 1} {
                        fail "Bias packets interleaved with event packets"
                }
                exp_continue
        }
        -re "exposureTe\[^\r]*discardGrade=\[0-9]+\[\r\n]*" {
                set inter 1
                exp_continue
        }
        -re "dataTe\[^\r]*dataPacketNumber\[0-9]+\[\r\n]*" {
                set inter 1
                exp_continue
        }
        -re "scienceReport\[^\r]*terminationCode=(\[0-9]+)\[\r\n]*" {
            set rc $expect_out(1,string)
            if {$rc != $retc} {
                fail "Unexpected terminationCode $rc (expected $retc)"
            }
        }
        timeout {}
    }
}

# ---- Procedure to Assert RADMON, Verify it, Deassert it, Verify it.
proc toggle_radmon { retc } {
    global cmd_id
    send -i $cmd_id "set radiationmonitor high\n"

    # ---- Wait for RADMON Confirmation ----
    expect {
        -re "SWSTAT_SCI_INHIBIT_ON.*\[\r\n]*" {}
        timeout {fail "No RADMON ENABLE confirmation"}
    }
```

```
    # ---- Wait for the scienceReport
    wait_stop_science_and_test $retc

    # ---- Restart the Suspended Science Run ----
    send -i $cmd_id "set radiationmonitor low\n"
    expect {
        -re "SWSTAT_SCI_INHIBIT_OFF.*\[\r\n]*" {}
        timeout {fail "No RADMON DISABLE confirmation"}
    }
}

# ---- Start command pipe ----
spawn $basedir/$tools/bin/cmdclient $env(ACISSERVER)
set cmd_id $spawn_id

# ---- Start telemetry pipe ----
spawn $basedir/$tools/bin/tlmclient $env(ACISSERVER)
sleep 1

# ---- Select Input from Image Loader ----
system make loaderselect

# ---- Apply patches ----
cold_boot
load_patch_list "$basedir/$tools/share/opt_tlmio.bcmd\
                 $basedir/$tools/share/opt_printswhouse.bcmd\
                 $basedir/$tools/share/opt_dearepl.bcmd\
                 $basedir/$tools/share/standard.bcmd\
                 $basedir/$tools/share/opt_cc3x3.bcmd\
                 $basedir/$tools/share/opt_ccignore.bcmd\
                 $basedir/$tools/share/opt_compressall.bcmd\
                 $basedir/$tools/share/opt_ctireport1.bcmd\
                 $basedir/$tools/share/opt_ctireport2.bcmd\
                 $basedir/$tools/share/opt_eventhist.bcmd\
                 $basedir/$tools/share/opt_reportgrade1.bcmd\
                 $basedir/$tools/share/opt_smtimedlookup.bcmd\
                 $basedir/$tools/share/opt_teignore.bcmd\
                 $basedir/$patchdir/opt_untricklebias.bcmd"
warm_boot

# ---- Power on FEPs and CCDs ----
power_on_boards "$ccd_list"

# ---- Wait for FEPs to finish powering ----
expect {
    -re ".*SWSTAT_FEPMAN_ENDLOAD: $last_fep\[\r\n]*" {}
    timeout { fail "Power-up Failure" }
}

# ---- Load Pblock for Faint Timed-Exposure Mode ----
send -i $cmd_id "load 0 te 4 {
    parameterBlockId             = 0x00000014
    fepCcdSelect                 = $ccd_list
    fepMode                      = 2 # FEP_TE_MODE_EV3x3
    bepPackingMode               = 2 # BEP_TE_MODE_GRADED
    onChip2x2Summing             = 0
    ignoreBadPixelMap            = 0
    ignoreBadColumnMap           = 0
    recomputeBias                = 1
    trickleBias                  = 1
    subarrayStartRow             = 0
    subarrayRowCount             = 99
    overclockPairsPerNode        = 8
```

```
    outputRegisterMode          = $quad_mode
    ccdVideoResponse            =     0     0     0     0     0     0
    primaryExposure             = 5
    secondaryExposure           = 0
    dutyCycle                   = 0
    fep0EventThreshold          =   100   100   100   100
    fep1EventThreshold          =   100   100   100   100
    fep2EventThreshold          =   100   100   100   100
    fep3EventThreshold          =   100   100   100   100
    fep4EventThreshold          =   100   100   100   100
    fep5EventThreshold          =   100   100   100   100
    fep0SplitThreshold          =    50    50    50    50
    fep1SplitThreshold          =    50    50    50    50
    fep2SplitThreshold          =    50    50    50    50
    fep3SplitThreshold          =    50    50    50    50
    fep5SplitThreshold          =    50    50    50    50
    fep4SplitThreshold          =    50    50    50    50
    fep5SplitThreshold          =    50    50    50    50
    lowerEventAmplitude         = 0
    eventAmplitudeRange         = 65535
    gradeSelections             = 0xffffffff 0xffffffff 0xffffffff 0xffffffff\
                                  0xffffffff 0xffffffff 0xffffffff 0xffffffff
    windowSlotIndex             = 65535
    histogramCount              = 0
    biasCompressionSlotIndex    =     3     3     1     1     1     1
    rawCompressionSlotIndex     = 0
    ignoreInitialFrames         = 2
    biasAlgorithmId             =     1     1     1     1     1     1
    biasArg0                    =     1     1     1     1     1     1
    biasArg1                    =     0     0     0     0     0     0
    biasArg2                    =     0     0     0     0     0     0
    biasArg3                    =    26    26    50    50    50    50
    biasArg4                    =    20    20    20    20    20    20
    fep0VideoOffset             =    65    65    65    65
    fep1VideoOffset             =    65    65    65    65
    fep2VideoOffset             =    65    65    65    65
    fep3VideoOffset             =    65    65    65    65
    fep4VideoOffset             =    65    65    65    65
    fep5VideoOffset             =    65    65    65    65
    deaLoadOverride             = 0
    fepLoadOverride             = 0
}
"
command_echo 1 9 "load te"

# ---- Test 1: stopScience during bias map creation ----
system make bias RUN=1
send -i $cmd_id "start 0 te 4\n"
command_echo 1 14 "start science run"
wait_bias_start
send -i $cmd_id "stop 0 science\n"
wait_stop_science_and_test 1

# ---- Test 2: double stopScience during bias map creation ----
system make bias RUN=2
send -i $cmd_id "start 0 te 4\n"
command_echo 1 14 "start science run"
wait_bias_start
send -i $cmd_id "stop 0 science\n"
sleep 1
send -i $cmd_id "stop 0 science\n"
wait_stop_science_and_test 1

# ---- Test 3: startScience during bias map creation ----
```

```
system make bias RUN=3
send -i $cmd_id "start 0 te 4\n"
command_echo 1 14 "start science run"
wait_bias_start
send -i $cmd_id "start 0 te 4\n"
command_echo 1 14 "start science run"
wait_stop_science_and_test 6
wait_bias $first_fep
system make image
wait_exp $last_fep
send -i $cmd_id "stop 0 science\n"
wait_stop_science_and_test 1

# ---- Test 4: assert/deassert RADMON during bias map creation ----
system make bias RUN=4
send -i $cmd_id "start 0 te 4\n"
command_echo 1 14 "start science run"
wait_bias_start
toggle_radmon 3
wait_bias $first_fep
system make image
wait_exp $last_fep
send -i $cmd_id "stop 0 science\n"
wait_stop_science_and_test 1

# ---- Test 5: stopScience during bias map telemetering ----
system make bias RUN=5
send -i $cmd_id "start 0 te 4\n"
command_echo 1 14 "start science run"
wait_bias $first_fep
send -i $cmd_id "stop 0 science\n"
wait_stop_science_and_test 1

# ---- Test 6: double stopScience during bias map telemetering ----
system make bias RUN=6
send -i $cmd_id "start 0 te 4\n"
command_echo 1 14 "start science run"
wait_bias $first_fep
send -i $cmd_id "stop 0 science\n"
sleep 1
send -i $cmd_id "stop 0 science\n"
wait_stop_science_and_test 1

# ---- Test 7: startScience during bias map telemetering ----
system make bias RUN=7
send -i $cmd_id "start 0 te 4\n"
command_echo 1 14 "start science run"
wait_bias $first_fep
send -i $cmd_id "start 0 te 4\n"
command_echo 1 14 "start science run"
wait_stop_science_and_test 4
wait_bias $first_fep
system make image
wait_exp $last_fep
send -i $cmd_id "stop 0 science\n"
wait_stop_science_and_test 1

# ---- Test 8: assert/deassert RADMON during bias map telemetering ----
system make bias RUN=8
send -i $cmd_id "start 0 te 4\n"
command_echo 1 14 "start science run"
wait_bias $first_fep
toggle_radmon 3
wait_bias $first_fep
```

```
system make image
wait_exp $last_fep
send -i $cmd_id "stop 0 science\n"
wait_stop_science_and_test 1

# ---- Test 9: stopScience during event processing ----
system make bias RUN=9
send -i $cmd_id "start 0 te 4\n"
command_echo 1 14 "start science run"
wait_bias $first_fep
system make image
wait_exp $last_fep
send -i $cmd_id "stop 0 science\n"
wait_stop_science_and_test 1

# ---- Test 10: double stopScience during event processing ----
system make bias RUN=10
send -i $cmd_id "start 0 te 4\n"
command_echo 1 14 "start science run"
wait_bias $first_fep
system make image
wait_exp $last_fep
send -i $cmd_id "stop 0 science\n"
sleep 1
send -i $cmd_id "stop 0 science\n"
wait_stop_science_and_test 1

# ---- Test 11: startScience during event processing ----
system make bias RUN=11a
send -i $cmd_id "start 0 te 4\n"
command_echo 1 14 "start science run"
wait_bias $first_fep
system make image
wait_exp $last_fep
system make bias RUN=11b
send -i $cmd_id "start 0 te 4\n"
command_echo 1 14 "start science run"
wait_stop_science_and_test 15
wait_bias $first_fep
system make image
wait_exp $last_fep
send -i $cmd_id "stop 0 science\n"
wait_stop_science_and_test 1

# ---- Test 12: assert/deassert RADMON during event processing ----
system make bias RUN=12a
send -i $cmd_id "start 0 te 4\n"
command_echo 1 14 "start science run"
wait_bias $first_fep
system make image
wait_exp $last_fep
system make bias RUN=12b
toggle_radmon 15
wait_bias $first_fep
system make image
wait_exp $last_fep
send -i $cmd_id "stop 0 science\n"
wait_stop_science_and_test 1

# ---- Report fate ----
pass "all tests successful"
```

MIT CSR ACIS	**ENGINEERING CHANGE ORDER**	**ECO No. 36–1030**

CENTER FOR SPACE RESEARCH
MASSACHUSETTS INSTITUTE OF TECHNOLOGY

DWG. NO.	NEW REV.	DRAWING TITLE
36-58021.02	C	Flight Software Optional Patch Release C Certification

REASON FOR CHANGE:
Certification of standard patch release B along with the following optional patches from release C: *smtimedlookup*, *compressall*, *eventhist* patch, and *cc3x3* patch.

DESCRIPTION OF CHANGE:
Three patch combinations are certified as release B–C–C:

(a) *cc3x3, eventhist, and smtimedlookup*. This is similar to the *cc3x3+eventhist* certification for release B–B–B, with the addition of *smtimedlookup* to permit other timed-exposure modes to be added at a later date.

(b) *cc3x3*, *eventhist*, *compressall*, and *smtimedlookup*.
The certification tests are taken from the optional release C suite, the only difference being that the tests are conducted after loading the full 4–patch set.

(c) *cc3x3*, *eventhist*, *compressall*, *untricklebias*, and *smtimedlookup*.
The certification tests are taken from the optional release C suite, the only difference being that the tests are conducted after loading the full 5–patch set.

	SIGNATURE	DATE	REMARKS:
ORIGINATOR	Peter Ford	03/21/03	Final sign-off version
MECHANICAL			
ELECTRICAL			
SOFTWARE			
STRUCTURE			
FABRICATION			
SCIENCE			
SYSTEMS ENG.			
QUALITY			
PROJ. ENGINEER			
DEPUTY PM			
PROJ. MANAGER			

TITLE: ACIS eventhist, cc3x3, smtimedlookup Patch Certification Release Notes

DOCUMENT NUMBER: 36-58021.02 REVISION: C

ORIGINATOR: Peter G. Ford <pgf@space.mit.edu>

LETTER	SCO NO.	DESCRIPTION	APPROVED	DATE
C	36-1030	Certify CC3x3/EventHist/smTimedL	RFG	03/21/2003

==========

Title: ACIS eventhist, cc3x3, smtimedlookup Patch Certification Release Notes for Version C

Software Change Order: 36-1030

Build Date: Wed Feb 19 00:50:52 EST 2003
Part Number: 36-58021.02
Version: C
CVS Tag: cc3x3+eventhist-B-C-C

Std Number: 36-58010
Std Version: B
Std Tag: release-B
Std SCO: 36-1019

Opt Number: 36-58020
Opt Version: C
Opt Tag: review-release-B-opt-C
Opt SCO: 36-1022

IPCL Number: 36-53204.0204
IPCL Version: N
IPCL CVS Tag: release-N

Description:

This certification verifies the operation of Continuous Clocking 3x3 Patch in conjunction with the Event Histogram and smTimedLookup Patches.

The certification consists of three tests, copied from the original test runs during the Options Release. The tests have been modified to load all three optional patches, rather than just one of them, and to clean up some false failures due to timing/pattern matching issues in the tests.

The tests verify that the patch modes run as they did during the original test when they are both installed into the system.

The Continuous Clocking 3x3 (cc3x3) test consists of two parts. The first launches a CC3x3 run, whereas the second runs CC1x3. This suite performs CC1x3 tests to verify that the modifications to the existing BEP Continuous Clocking functions do not break the existing CC1x3 functionality. Since the FEP software only contains CC3x3 code during CC3x3 runs (this is verified by the CC1x3 run), and no BEP functions used by Timed Exposure are modified by the patch, the Timed Exposure modes do not need to be re-tested as part of this certification.

Each test sends a series of events on the right side of each quadrant (the original test was derived from the test for the rquad bug fix), and verifies that the mode runs nominally, and produces the expected event list. Since the "stop" critereon for the test is a little fuzzy, the runs tend to produce additional exposures that aren’t in the file used to check the run’s event output. "diff" used in the test produces mismatches on the additional exposures produced by the test run. Manual check of the run data shows that the event lists are replicated correctly by the run. Later, a "wrapping"

comparison may be developed to eliminate this manual step.

The Event Histogram test uses a similar strategy to the CC3x3 test. It starts an Event Histogram run, and sends in a series of standard events. It then compares the resulting quadrant histograms with an example file to verify the results.

One caveat that arose during the review of the Optional patches is that, when the standard patch "zap1expo" is present, which it should always be, the first exposure of event histogram mode will not contain any events. This will cause the first histogram from each FEP quadrant to appear to have been integrated for 1 less frame time than subsequent quadrant histograms. This is different than Raw Histogram mode, which is not affected by the "zap1expo" patch. The histogram example file used for this certification assumes that no events are sent during exposure 2 (the first "real" exposure of the run).

The smTimedExposure patch is tested by merely running a timed-exposure faint run, verifying that the bias and event detection phases have been invoked, and then stopping the run.

Included Patches:

- eventhist
- cc3x3
- smtimedlookup

Test Support Patches:

- printswhouse
- dearepl
- tlmio

Test Results:

- smtimedlookup --> PASS
- cc3x3 --> PASS
- eventhist --> PASS
- eventhist --> PASS

TITLE: ACIS eventhist, cc3x3, compressall, smtimedlookup Patch Certification Release Notes

DOCUMENT NUMBER: 36-58021.02 REVISION: C

ORIGINATOR: Peter G. Ford <pgf@space.mit.edu>

LETTER	SCO NO.	DESCRIPTION	APPROVED	DATE
C	36-1030	Certify CC3x3/EventHist/smTimedL	RFG	03/21/2003

==========

Title: ACIS eventhist, cc3x3, compressall, smtimedlookup Patch Certification Release Notes f
or Version C

Software Change Order: 36-1030

Build Date: Wed Feb 19 02:16:20 EST 2003
Part Number: 36-58021.02
Version: C
CVS Tag: cc3x3+eventhist+compressall-B-C-C

Std Number: 36-58010
Std Version: B
Std Tag: release-B
Std SCO: 36-1019

Opt Number: 36-58020
Opt Version: C
Opt Tag: review-release-B-opt-C
Opt SCO: 36-1022

IPCL Number: 36-53204.0204
IPCL Version: N
IPCL CVS Tag: release-N

Description:
This certification verifies the operation of the
Continuous Clocking 3x3, Event Histogram, Compress
All, and Science Mode Timed Lookup Patches.

The certification consists of two tests, copied
from the original test run during the Options
Release. The tests have been modified to load
all four optional patches, rather than just one
at a time, and to clean up some false failures due
to timing/pattern matching issues in the tests.

The tests verify that the patch modes run as they
did during the original test when they are
both installed into the system.

The Continuous Clocking 3x3 (cc3x3) test consists of two
parts. The first launches a CC3x3 run, whereas the second
runs CC1x3. This suite performs CC1x3 tests to verify that
the modifications to the existing BEP Continuous Clocking
functions do not break the existing CC1x3 functionality. Since
the FEP software only contains CC3x3 code during CC3x3 runs (this
is verified by the CC1x3 run), and no BEP functions used by Timed Exposure
are modified by the patch, the Timed Exposure modes do not need to
be re-tested as part of this certification.

Each test sends a series of events on the right side of each quadrant
(the original test was derived from the test for the rquad
bug fix), and verifies that the mode runs nominally, and
produces the expected event list. Since the "stop" critereon
for the test is a little fuzzy, the runs tend to produce
additional exposures that aren’t in the file used to check
the run’s event output. "diff" used in the test produces
mismatches on the additional exposures produced by the test
run. Manual check of the run data shows that the event lists
are replicated correctly by the run. Later, a "wrapping"

comparison may be developed to eliminate this manual step.

The Event Histogram test uses a similar strategy to the CC3x3 test. It starts an Event Histogram run, and sends in a series of standard events. It then compares the resulting quadrant histograms with an example file to verify the results.

One caveat that arose during the review of the Optional patches is that, when the standard patch "zap1expo" is present, which it should always be, the first exposure of event histogram mode will not contain any events. This will cause the first histogram from each FEP quadrant to appear to have been integrated for 1 less frame time than subsequent quadrant histograms. This is different than Raw Histogram mode, which is not affected by the "zap1expo" patch. The histogram example file used for this certification assumes that no events are sent during exposure 2 (the first "real" exposure of the run).

The smTimedExposure patch is tested by merely running a timed-exposure faint run, verifying that the bias and event detection phases have been invoked, and then stopping the run.

The Compress All patch is tested by copying an image to the image loader that contains several very "noisy" rows that are known to be incompressible by the Huffman tables. A timed-exposure raw-mode run is executed and the pixelCount field of the dataTeRaw packets of a couple of raw frames is monitored. The test fails if pixelCount is ever zero.

Included Patches:
- eventhist
- cc3x3
- compressall
- smtimedlookup

Test Support Patches:
- printswhouse
- dearepl
- tlmio

Test Results:
- smtimedlookup --> PASS
- cc3x3 --> PASS
- eventhist --> PASS
- eventhist --> PASS
- compressall --> PASS

TITLE: ACIS untricklebias, eventhist, cc3x3, compressall, smtimedlookup Patch Certification Release Notes

DOCUMENT NUMBER: 36-58021.02 REVISION: C

ORIGINATOR: Peter G. Ford <pgf@space.mit.edu>

LETTER	SCO NO.	DESCRIPTION	APPROVED	DATE
C	36-1030	Certify CC3x3/EventHist/smTimedL	RFG	03/21/2003

==========

Title: ACIS untricklebias, eventhist, cc3x3, compressall, smtimedlookup Patch Certification Release Notes for Version C

Software Change Order: 36-1030

Build Date: Wed Feb 19 04:55:02 EST 2003
Part Number: 36-58021.02
Version: C
CVS Tag: cc3x3+eventhist+compressall+untricklebias-B-C-C

Std Number: 36-58010
Std Version: B
Std Tag: release-B
Std SCO: 36-1019

Opt Number: 36-58020
Opt Version: C
Opt Tag: review-release-B-opt-C
Opt SCO: 36-1022

IPCL Number: 36-53204.0204
IPCL Version: N
IPCL CVS Tag: release-N

Description:

This certification verifies the operation of the Continuous Clocking 3x3, Event Histogram, Compress All, Untrickle Bias, and Science Mode Timed Lookup Patches.

The certification consists of two tests, copied from the original test run during the Options Release. The tests have been modified to load all four optional patches, rather than just one at a time, and to clean up some false failures due to timing/pattern matching issues in the tests.

The tests verify that the patch modes run as they did during the original test when they are both installed into the system.

The Continuous Clocking 3x3 (cc3x3) test consists of two parts. The first launches a CC3x3 run, whereas the second runs CC1x3. This suite performs CC1x3 tests to verify that the modifications to the existing BEP Continuous Clocking functions do not break the existing CC1x3 functionality. Since the FEP software only contains CC3x3 code during CC3x3 runs (this is verified by the CC1x3 run), and no BEP functions used by Timed Exposure are modified by the patch, the Timed Exposure modes do not need to be re-tested as part of this certification.

Each test sends a series of events on the right side of each quadrant (the original test was derived from the test for the rquad bug fix), and verifies that the mode runs nominally, and produces the expected event list. Since the "stop" critereon for the test is a little fuzzy, the runs tend to produce additional exposures that aren’t in the file used to check the run’s event output. "diff" used in the test produces mismatches on the additional exposures produced by the test run. Manual check of the run data shows that the event lists are replicated correctly by the run. Later, a "wrapping"

```
        comparison may be developed to eliminate this manual step.

        The Event Histogram test uses a similar strategy to
        the CC3x3 test. It starts an Event Histogram run, and
        sends in a series of standard events. It then compares
        the resulting quadrant histograms with an example file
        to verify the results.

        One caveat that arose during the review of the Optional
        patches is that, when the standard patch "zap1expo" is
        present, which it should always be, the first exposure
        of event histogram mode will not contain any events.
        This will cause the first histogram from each FEP quadrant
        to appear to have been integrated for 1 less frame time than
        subsequent quadrant histograms. This is different than Raw
        Histogram mode, which is not affected by the "zap1expo"
        patch. The histogram example file used for this certification
        assumes that no events are sent during exposure 2 (the
        first "real" exposure of the run).

        The smTimedExposure patch is tested by merely running a
        timed-exposure faint run, verifying that the bias and event
        detection phases have been invoked, and then stopping the run.

        The Compress All patch is tested by copying an image to
        the image loader that contains several very "noisy" rows
        that are known to be incompressible by the Huffman tables.
        A timed-exposure raw-mode run is executed and the
        pixelCount field of the dataTeRaw packets of a couple of
        raw frames is monitored. The test fails if pixelCount is
        ever zero.

        The Untrickle Bias patch is tested by a pair of expect scripts,
        each of which performs 12 tests, one in TE mode, the other in
        CC mode. Each test starts a science run and then terminates it
        in one of the possible ways, viz:

         1: stopScience during bias map creation
         2: double stopScience during bias map creation
         3: startScience during bias map creation
         4: assert/deassert RADMON during bias map creation
         5: stopScience during bias map telemetering
         6: double stopScience during bias map telemetering
         7: startScience during bias map telemetering
         8: assert/deassert RADMON during bias map telemetering
         9: stopScience during event processing
        10: double stopScience during event processing
        11: startScience during event processing
        12: assert/deassert RADMON during event processing

        The tests fail unless all steps complete and return the anticipated
        scienceReport return codes.

-----------------------------------------------------------------------
Included Patches:
        untricklebias
        eventhist
        cc3x3
        compressall
        smtimedlookup

-----------------------------------------------------------------------
Test Support Patches:
        printswhouse
```

```
        dearepl
        tlmio

------------------------------------------------------------------------
Test Results:
        smtimedlookup --> PASS
        cc3x3 --> PASS
        eventhist --> PASS
        eventhist --> PASS
        compressall --> PASS
        untricklebias --> PASS
        untricklebias --> PASS
```