r e p o r t

MIT Kavli Institute
One Kendall Square, Building 300
Cambridge MA 02139–4307
Tel: 617-253-7277
Fax: 617-253-8084

To: ACIS Science Operations Team

From: Peter Ford, NE83-545 <pgf@space.mit.edu>

Date: August 8th 2014

Subject: Regression tests for the ACIS Engineering Unit (v 1.1)

1. Introduction

A series of high-level tests have been developed to validate the performance of the ACIS engineering unit (hardware simulator). The tests assume that the sub-units – PSMC, DEA and DPA – have been powered-up and are drawing the expected current from the external 28V supplies.

2. Test 1 – DEA (video boards and board 11) and DPA (BEP-A and FEPs)

In this test, BEP-A is cold-booted, loaded with flight patches and *deaeng*, and warm-booted. 6 CCDs and 6 FEPs are powered up and a timed-exposure science run is begun using a parameter block whose *eventThreshold*, *splitThreshold* and *videoOffset* fields are chosen to generate a few event candidates per node per exposure. The run is made in Faint-Bias mode, which reports the *initialOverclocks* from each bias map without having to report the bias maps themselves. Once at least 30 exposures have been received from each CCD, the run is terminated with a *stopScience* command. The science run is then repeated with a different set of 6 CCDs, including the 4 CCDs what were not used in the first run. At the end of the first run, the "*test1.tcl*" *expect* script prints a table of commanded offsets and average bias overclocks, *viz.*

	Video Offsets				Initial Overclocks			
ccd	A	B	C	D	A	B	C	D
0	145	37	38	38	272	347	499	602
3	35	38	38	38	387	538	481	421
4	44	30	49	42	514	394	762	638
5	41	35	42	35	349	444	493	385
7	35	42	42	41	383	379	550	391
9	38	38	38	38	581	407	376	520

Note that node A of CCD_0 is given a larger *videoOffset* than the rest in order to reduce the number of event candidates that it would otherwise report. It is suspected that the terminating input load to this node may be faulty. The second test run produces a similar table, *e.g.*,

	Video Offsets				Initial Overclocks			
ccd	A	B	C	D	A	B	C	D
0	145	37	38	38	295	347	494	602
1	55	30	55	25	302	573	266	360
2	40	30	50	50	653	646	485	546
3	35	38	38	38	380	536	474	417
6	33	48	45	32	492	635	186	554
8	25	45	13	35	403	462	733	624

The *expect* script tests many functions of the EU: warm and cold booting, patch loading – it dumps the load and validates the result – FEP booting, loading and execution, CCD selection, etc. It would be useful to add a test of the backup DEA interface board ("Board 12") powered by the DEA-B output from the PSMC, but this mode is currently unavailable.

3. Test 2 – DPA (BEP-A and BEP-B, FEPs) and Image Loader

In the second test, commanded by the "*test2.tcl*" *expect* script, the DEA and its video boards are by-passed and pixels are sent to the FEPs via the Image Loader. The test starts by cold-booting BEP-A, loading flight patches and *dearepl*, and warm-booting. The image loader is commanded to generate a simple bias image and a timed-exposure science run is started in faint-bias mode. Once the "`SWSTAT_FEP_STARTDATA`" signal is received from software housekeeping, commands are sent to the Image Loader to generate an image containing a small number of event candidates. Once at least 30 exposures have been received from each CCD, the run is terminated with a *stopScience* command. The expect script reports the *videoOffset* and *initialOverclock* values for each (dummy) CCD and bias map, *e.g.*,

	Video Offsets				Initial Overclocks			
ccd	A	B	C	D	A	B	C	D
0	50	50	50	50	200	300	400	500
1	50	50	50	50	200	300	400	500
2	50	50	50	50	200	300	400	500
3	50	50	50	50	200	300	400	500
5	50	50	50	50	200	300	400	500
7	50	50	50	50	200	300	400	500

The test is now performed a second time using BEP-B in place of BEP-A. This necessitates selecting the second BEP, cold booting it, loading the patches, and warm booting. After performing the science run a second time, the same script should report the same table values as before. Of course, these numbers say nothing about the state of health of the EU, beyond its ability to read the output of the Image Loader, but it tests the latter and also the Pixel Switch (the unit that switches the FEPs input between the DEA and the Image Loader.)

4. Applicable Documents

- ACIS DPA/DEA Interface Control Document, MIT 36–02205, Revision C, March 10, 1995.
- ACIS Software IP&CL Structure Definition Notes, MIT 36–53204.0204, Revision N, March 15, 2001.
- ACIS SI Software Detailed Design Specification (As-Built), MIT 36–53200, February 3, 2000.

5. Glossary

bcmd	A program and associated language for generating ACIS commands.
BEP	ACIS Back-End Processor.
DEA	Detector Electronics Assembly: 10 video boards and 2 interface boards (one active, one spare).
DPA	Digital Processor Assembly: the 6 FEPs and two BEPs (one active, one spare).
FEP	ACIS Front-End Processor.
PSMC	ACIS Power Supply and Mechanism Controller.

6. Appendix – test1.tcl

```
#! /usr/bin/env expect

set target [lindex $argv 0]
set toolsdir  {/nfs/acis/h2/pgf/acis/patches/tools}
set release {release-F-opt-G}
set patchdir "/nfs/acis/h3/acisfs/patchbld/$release/release/dist"
set stddir "$patchdir/standard-$release"
set reldir "$patchdir/options-$release"
set timeout 10
```

```
set maxexp 30
set nloop 0
set last_fep 5
set cmdid 5000
set patches  "$stddir/standard.bcmd\
              $reldir/opt_smtimedlookup.bcmd\
              $reldir/opt_eventhist.bcmd\
              $reldir/opt_cc3x3.bcmd\
              $reldir/opt_compressall.bcmd\
              $reldir/opt_txings.bcmd\
              $reldir/opt_tlmio.bcmd\
              $reldir/opt_printswhouse.bcmd\
              $reldir/opt_deaeng.bcmd\
              ./release54.bcmd"
# ------------------------------------------------------------------------
# Assign thresholds and video offsets to CCDs
proc set_parms {ccds} {
    global thresh split voff iccd
    foreach ii [list \
       { 0 {25 12 13 12} {10 5 5 5} {145 37 38 38} } \
       { 1 { 8  8  8  8} { 5 5 5 5} {55 30 55 25} } \
       { 2 { 8  8  8  9} { 5 5 5 5} {40 30 50 50} } \
       { 3 {11 12 11 13} { 5 5 5 5} {35 38 38 38} } \
       { 4 {12 12 12 12} { 5 5 5 5} {44 30 49 42} } \
       { 5 {11 11 12 11} { 8 8 8 8} {41 35 42 35} } \
       { 6 { 8  8  9 10} { 5 5 5 5} {33 48 45 32} } \
       { 7 {12 12 12 12} { 8 8 8 8} {35 42 42 41} } \
       { 8 { 8  8  8 11} { 5 5 5 5} {25 45 13 35} } \
       { 9 {12 12 12 12} { 5 5 5 5} {38 38 38 38} } \
       {10 { 0  0  0  0} { 0 0 0 0} { 0  0  0  0} } \
    ] {
       set ccd [lindex $ii 0]
       set iccd($ccd) 0
       lassign $ii nn thr($ccd) spl($ccd) vof($ccd)
    }
    foreach ii {0 1 2 3 4 5} {
       set ccd [lindex $ccds $ii]
       set thresh($ii) $thr($ccd)
       set split($ii) $spl($ccd)
       set voff($ii) $vof($ccd)
    }
}
# ------------------------------------------------------------------------
# Construct DEA board 11 housekeeping block
proc dea_cmd {cmdid slot} {
  set str    "load $cmdid dea $slot {\n"
  append str " deaBlockId= 1\n"
  append str " sampleRate  = 10\n"
  foreach ii {{0 7} {12 12} {15 20} {25 40}} {
    for {lassign $ii a b} {$a <= $b} {incr a} {
      append str "  queries   = {\n"
      append str "    ccdId   = 10\n"
      append str "    queryId = $a\n"
      append str "  }\n"
    }
```

```
  }
  return "$str}\n"
}
# ------------------------------------------------------------------------
# Set up
# ------------------------------------------------------------------------

# ---- Embed the Procedure Library ----
source $toolsdir/lib/lib-exp/runtest_support.tcl

# ---- Start the Command Pipe ----
spawn $toolsdir/bin/cmdclient $env(ACISSERVER)
set cmd_id $spawn_id

# ---- Start the Telemetry Pipe ----
spawn $toolsdir/bin/tlmclient $env(ACISSERVER)
sleep 1

# ---- Select DEA input to FEPs ----
system make ACISSERVER=$env(ACISSERVER) deaselect

# ---- Restart the Processor ----
cold_boot
load_patch_list $patches
warm_boot

# ---- Check the patch load ----
source confirm_patch_load.tcl
confirm_patch_load $patches

# ---- Load DEA housekeeping block ----
send -i $cmd_id [dea_cmd [incr cmdid] 4]
command_echo 1 13 {Load DEA housekeeping block}
sleep 1

# ---- Start DEA housekeeping ----
send -i $cmd_id "start [incr cmdid] dea 4\n"
command_echo 1 18 {Start DEA housekeeping}
# ------------------------------------------------------------------------
# Part 1: Start science runs
# ------------------------------------------------------------------------

foreach ccds {{0 3 4 5 7 9} {0 1 2 3 6 8}} {

# ---- Setup ccd parameters ----
set_parms $ccds
puts "\nTesting CCDs $ccds\n"

# ---- Power down all boards ----
power_off_boards

# ---- Power up requested FEPs and DEAs ----
power_on_boards $ccds
expect {
    -timeout 60
```

```
    -re ".*SWSTAT_FEPMAN_ENDLOAD: $last_fep\[\r\n]*" {}
    timeout {}
}

# ---- Load TE parameter block ----
send -i $cmd_id "load [incr cmdid] te 4 {
  parameterBlockId           = 0x00fff024
  fepCcdSelect               = $ccds
  fepMode                    = 2 # 0 # 3 # FEP_TE_MODE_EV5x5
  bepPackingMode             = 1 # 0 # BEP_TE_MODE_FAINTBIAS
  onChip2x2Summing           = 0
  ignoreBadPixelMap          = 0
  ignoreBadColumnMap         = 0
  recomputeBias              = 1
  trickleBias                = 0
  subarrayStartRow           = 0
  subarrayRowCount           = 1023
  overclockPairsPerNode      = 8
  outputRegisterMode         = 0 # QUAD_FULL
  ccdVideoResponse           = 0 0 0 0 0 0
  primaryExposure            = 31
  secondaryExposure          = 0
  dutyCycle                  = 0
  fep0EventThreshold         =   $thresh(0)
  fep1EventThreshold         =   $thresh(1)
  fep2EventThreshold         =   $thresh(2)
  fep3EventThreshold         =   $thresh(3)
  fep4EventThreshold         =   $thresh(4)
  fep5EventThreshold         =   $thresh(5)
  fep0SplitThreshold         =   $split(0)
  fep1SplitThreshold         =   $split(1)
  fep2SplitThreshold         =   $split(2)
  fep3SplitThreshold         =   $split(3)
  fep4SplitThreshold         =   $split(4)
  fep5SplitThreshold         =   $split(5)
  lowerEventAmplitude        = 5
  eventAmplitudeRange        = 50
  gradeSelections            = 0xffffffff 0xffffffff 0xffffffff 0xffffffff
                               0xffffffff 0xffffffff 0xffffffff 0x7fffffff
  windowSlotIndex            = 65535
  histogramCount             = 0
  biasCompressionSlotIndex   =    0    3    1    1    0    0
  rawCompressionSlotIndex    = 2
  ignoreInitialFrames        = 5
  biasAlgorithmId            =    1    1    1    1    1    1
  biasArg0                   =    2    2    2    2    2    2
  biasArg1                   =    5    5    5    5    5    5
  biasArg2                   =   20   20   20   20   20   20
  biasArg3                   =    0    0    0    0    0    0
  biasArg4                   =   20   20   20   20   20   20
  fep0VideoOffset            =   $voff(0)
  fep1VideoOffset            =   $voff(1)
  fep2VideoOffset            =   $voff(2)
  fep3VideoOffset            =   $voff(3)
  fep4VideoOffset            =   $voff(4)
```

```
  fep5VideoOffset            =   $voff(5)
  deaLoadOverride            = 0x00000000
  fepLoadOverride            = 0x00000000
}
"
command_echo 1 9 {Load TE pBlock}

# ---- Start the science run ----
send -i $cmd_id "start [incr cmdid] te 4\n"
command_echo 1 14 {Start TE Science}

# ---- Dump Huffman table ----
send -i $cmd_id "dump [incr cmdid] huffman\n"
command_echo 1 35 {Dump Huffman Tables}

# ---- Wait for exposure ----
expect {
    -timeout 300
    -re "exposure\[^\r]*\
       ccdId=(\[0-9])\[^\r]*\
       exposureNumber=(\[0-9x]+)\[^\r]*\
       \[\r\n]*" {
       set nccd $expect_out(1,string)
       if {$iccd($nccd) == 0 || $expect_out(2,string) < $maxexp} {
           set iccd($nccd) 1
           exp_continue
       }
    }
    timeout {fail { run timed out }}
}

# ---- Stop the science run ----
send -i $cmd_id "stop [incr cmdid] science\n"
expect {
    -timeout 3600
    -re "scienceReport.*terminationCode=\[0-9]+\[\r\n]*" {}
     timeout { fail { timeout waiting for scienceReport } }
}

# ---- Display statistics ----
system make filt N=$nloop
incr nloop

}

system mv pkts.raw $target.raw
pass " Job $target succeeded "

# ------------------------------------------------------------------------
# Done
# ------------------------------------------------------------------------
```

7. Appendix — test2.tcl

```
#! /usr/bin/env expect

set target [lindex $argv 0]
set toolsdir  {/nfs/acis/h2/pgf/acis/patches/tools}
set release {release-F-opt-G}
set patchdir "/nfs/acis/h3/acisfs/patchbld/$release/release/dist"
set stddir "$patchdir/standard-$release"
set reldir "$patchdir/options-$release"
set timeout 10
set ccds {0 1 2 3 5 7}
set cmdid 5000
set last_fep 5
set maxexp 30
set nr  -1
set patches  "$stddir/standard.bcmd\
              $reldir/opt_smtimedlookup.bcmd\
              $reldir/opt_eventhist.bcmd\
              $reldir/opt_cc3x3.bcmd\
              $reldir/opt_compressall.bcmd\
              $reldir/opt_txings.bcmd\
              $reldir/opt_tlmio.bcmd\
              $reldir/opt_printswhouse.bcmd\
              $reldir/opt_dearepl.bcmd\
              ./release53.bcmd"
# ------------------------------------------------------------------------
# Construct DEA board 11 housekeeping block
proc dea_cmd {cmdid slot} {
  set str    "load $cmdid dea $slot {\n"
  append str " deaBlockId= 1\n"
  append str " sampleRate  = 10\n"
  foreach ii {{0 7} {12 12} {15 20} {25 40}} {
    for {lassign $ii a b} {$a <= $b} {incr a} {
      append str "  queries   = {\n"
      append str "    ccdId   = 10\n"
      append str "    queryId = $a\n"
      append str "  }\n"
    }
  }
  return "$str}\n"
}
# ------------------------------------------------------------------------
# Set up
# ------------------------------------------------------------------------

# ---- Embed the Procedure Library ----
source $toolsdir/lib/lib-exp/runtest_support.tcl

# ---- Start the Command Pipe ----
spawn $toolsdir/bin/cmdclient $env(ACISSERVER)
set cmd_id $spawn_id

# ---- Start the Telemetry Pipe ----
spawn $toolsdir/bin/tlmclient $env(ACISSERVER)
```

```
sleep 1

# ---- Select DEA input to FEPs ----
system make ACISSERVER=$env(ACISSERVER) loaderselect

# ---- Repeat test for both BEPs ----
foreach bep {A B} {

# ---- Restart the Processor ----
send -i $cmd_id "select bep $bep\n"
cold_boot
load_patch_list $patches
warm_boot

# ---- Check the patch load ----
source confirm_patch_load.tcl
confirm_patch_load $patches

# ---- Load DEA housekeeping block ----
send -i $cmd_id [dea_cmd [incr cmdid] 4]
command_echo 1 13 {Load DEA housekeeping block}
sleep 1

# ---- Start DEA housekeeping ----
send -i $cmd_id "start [incr cmdid] dea 4\n"
command_echo 1 18 {Start DEA housekeeping}

# ------------------------------------------------------------------------
# Part 1: Start science runs
# ------------------------------------------------------------------------

puts "\nTesting CCDs $ccds with BEP-$bep\n"

# ---- Power off boards ----
power_off_boards
sleep 5

# ---- Power up requested FEPs and DEAs ----
power_on_boards $ccds
expect {
    -timeout 70
    -re ".*SWSTAT_FEPMAN_ENDLOAD: $last_fep\[\r\n]*" {}
    timeout { if {$reboot} { fail { Failed to power up } } }
}

# ---- Load TE parameter block ----
send -i $cmd_id "load [incr cmdid] te 4 {
  parameterBlockId           = 0x00fff024
  fepCcdSelect               = $ccds
  fepMode                    = 2 # FEP_TE_MODE_EV3x3
  bepPackingMode             = 1 # BEP_TE_MODE_FAINTBIAS
  onChip2x2Summing           = 0
  ignoreBadPixelMap          = 0
  ignoreBadColumnMap         = 0
  recomputeBias              = 1
```

```
  trickleBias               = 0
  subarrayStartRow          = 0
  subarrayRowCount          = 1023
  overclockPairsPerNode     = 8
  outputRegisterMode        = 0 # QUAD_FULL
  ccdVideoResponse          = 0 0 0 0 0 0
  primaryExposure           = 31
  secondaryExposure         = 0
  dutyCycle                 = 0
  fep0EventThreshold        =   30   30   30   30
  fep1EventThreshold        =   30   30   30   30
  fep2EventThreshold        =   30   30   30   30
  fep3EventThreshold        =   30   30   30   30
  fep4EventThreshold        =   30   30   30   30
  fep5EventThreshold        =   30   30   30   30
  fep0SplitThreshold        =   13   13   13   13
  fep1SplitThreshold        =   13   13   13   13
  fep2SplitThreshold        =   13   13   13   13
  fep3SplitThreshold        =   13   13   13   13
  fep4SplitThreshold        =   13   13   13   13
  fep5SplitThreshold        =   13   13   13   13
  lowerEventAmplitude       = 20
  eventAmplitudeRange       = 4000
  gradeSelections           = 0xffffffff 0xffffffff 0xffffffff 0xffffffff
                              0xffffffff 0xffffffff 0xffffffff 0x7fffffff
  windowSlotIndex           = 65535
  histogramCount            = 0
  biasCompressionSlotIndex  =    0    3    1    1    0    0
  rawCompressionSlotIndex   = 2
  ignoreInitialFrames       = 5
  biasAlgorithmId           =    1    1    1    1    1    1
  biasArg0                  =    2    2    2    2    2    2
  biasArg1                  =    5    5    5    5    5    5
  biasArg2                  =   20   20   20   20   20   20
  biasArg3                  =    0    0    0    0    0    0
  biasArg4                  =   20   20   20   20   20   20
  fep0VideoOffset           =   50   50   50   50
  fep1VideoOffset           =   50   50   50   50
  fep2VideoOffset           =   50   50   50   50
  fep3VideoOffset           =   50   50   50   50
  fep4VideoOffset           =   50   50   50   50
  fep5VideoOffset           =   50   50   50   50
  deaLoadOverride           = 0x00000000
  fepLoadOverride           = 0x00000000
}
"
command_echo 1 9 {Load TE pBlock}

# ---- Load bias maps ----
system make bias ACISSERVER=$env(ACISSERVER)

# ---- Start the science run ----
send -i $cmd_id "start [incr cmdid] te 4\n"
command_echo 1 14 {Start TE Science}
```

```
# ---- Dump Huffman table ----
send -i $cmd_id "dump [incr cmdid] huffman\n"
command_echo 1 35 {Dump Huffman Tables}

# ---- Wait for exposure ----
set sw 0
expect {
    -timeout 300
    -re "SWSTAT_FEP_STARTDATA\[^\r]*\[\r\n]*" {
       system make image ACISSERVER=$env(ACISSERVER)
       incr sw
       exp_continue
    }
    -re "exposure\[^\r]*\
       ccdId=(\[0-9])\[^\r]*\
       exposureNumber=(\[0-9x]+)\[^\r]*\
       \[\r\n]*" {
       set nccd $expect_out(1,string)
       if {$sw == 0 || $expect_out(2,string) < $maxexp} {
           exp_continue
       }
    }
    timeout {fail { run timed out }}
}

# ---- Stop the science run ----
send -i $cmd_id "stop [incr cmdid] science\n"
wait_stop_science

# ---- Display statistics ----
system make filt N=[incr nr]
}

# ---- Reset for BEP-A ----
send -i $cmd_id "select bep a\n";
warm_boot
system mv pkts.raw $target.raw

pass " Job $target succeeded "

# ------------------------------------------------------------------------
# Done
# ------------------------------------------------------------------------
```