PCA Housekeeping - Telemetry Simulator for PCA and SOC Software Testing

XTE-PCA Software Module

PCA-SIM-EXP-V5.0

Build 4.0

Version 5.0

November 21st 1994

by

Barry Giles

PCA XTE group

Code 666, NASA/GSFC, Greenbelt MD 20771

and

Vikram Savkoor

Hughes STX Corp.

4400 Forbes Blvd., Lanham, MD 20706

Foreword

The PCA science group is responsible for providing to the SOC an agreed set of software products together with their associated documentation and test data/results. Such deliveries are governed by the details to be found in the PCA IAD and ICD documents.

In order to facilitate the production of these deliverables a series of documents have been produced that are intended to contain as much relevant and accurate information as is available that pertains to the title and subject matter of each delivery item. These documents should also present and explain the concepts and plans for the use of the supplied information.

This document is concerned with the topic:-

PCA HOUSEKEEPING TELEMETRY SIMULATOR.

The PCA group will also produce additional software for its own use, in its own external environment. This software will also be provided to the SOC but is not governed by such rigorous product assurance requirements although it also requires documentation.

Table of Contents

1.0 INTRODUCTION

1.1 Identification of Document

This document describes a software program written in C that simulates the PCA experiment housekeeping telemetry on the XTE spacecraft.

1.2 Scope of Document

The PCA team at GSFC is required to deliver a number of software modules to the Science Operations Center (SOC) for the XTE mission. This document attempts to examine and pull together all of the aspects relating to the development of the PCA Housekeeping simulator software.

1.3 Purpose and Objectives of Document

This document is intended to describe:

- how to use and run the simulator program
- the expected output and the menu options to change it
- the arrays of tabulated typical data values

1.4 Document Status and Schedule

The major milestones in the history of the development of the PCA housekeeping simulator are outlined below. The build development plan is given in section 11.

Date	Version No.	Manual	Function	Distribution
1 / 1 / 93	1.0	NO	Limited Simulation	Limited
1 / 8 / 93	1.1	NO	Test Development	None
4 / 15 / 93	1.2	NO	Test Development	None
6 / 12 / 93	2.0	YES	Full Simulation	SOC delivery
6 / 22 / 93	2.1	YES	as 2.0 + menu & cmd. line	SOC delivery
10 / 29 / 93	2.2	YES	as 2.1 but revised time sequences & added in real conversions table	SOC delivery
3 / 31 / 94	3.0	YES	as 2.2 but parameters now specified in a data file rather than fixed at compile time. Cmd editor linked to drive simulator The C++ version has been discontinued	SOC delivery

Date	Version No.	Manual	Function	Distribution
6 / 8 / 94	4.0	YES	as 3.0 but revised conversions & data values Simulator was frozen at this level of sophistication	SOC delivery
11 / 21 / 94	5.0	YES	as 3.0 but revised conversions values conversion table split up into a section for each PCU	SOC delivery

1.5 Document Organization

Section 2 - provides references or bibliography for this document

Section 3 - describes the purpose and main capabilities of the software. It also describes how to control the various program options

Section 4 - explains the procedures for installing, tailoring, and initiating the software

Section 5 - describes how to start and terminate operation normally, and how to determine whether normal termination has occurred

Section 6 - describes each of the functions and their operations

Section 7 - lists and explains each possible error condition and associated message that may be encountered. describes the corresponding corrective actions to be taken

Section 8 - explains recovery procedures that the user may employ

Section 9 - provides an alphabetized list of the definitions for abbreviations and acronyms used in this document

Section 10 - provides an alphabetized list of definition for special terms used in the document

Section 11 - presents a brief development plan for the PCA housekeeping simulator

Section 12 - details the “canonical” test activities

Section 13 - provides the “Standard” data values for testing

Section 14 - contains appendices

2.0 RELATED DOCUMENTATION

2.1 Parent Documents

None

2.2 Applicable Documents

Relevant information to the general areas and detailed format of PCA Housekeeping data can be found in the following document:

PCA Housekeeping - Status, Use, Screen Functions, Recall Modes, SOC interface, XTE-PCA Software Module, PCA-EXP-MAN-V5.0, Barry Giles, PCA Team, November 21st, 1994

PCA Commands - Types, Sequences, Use, Configurations, Screen Functions, Verification, SOC interface, XTE-PCA Software Module, PCA-COM-EXP-V4.0, Barry Giles, PCA Team, June 18th, 1994

Interface Control Document for the PCA Command, Telemetry and Monitoring between the XTE Science Operation Center and the PCA Principal Investigator Team. XTE-SOC-PCA-ICD-V1.0, Rahim Choudhary et al., March 1994

Software Product Assurance Plan between the XTE Science Operation Center and the PCA Principal Investigator Team - Aileen Tousignant & Barry Giles, PCA team, February 12, 1993

2.3 Information Documents

The following documents, though not directly applicable, amplify or clarify the information presented in this volume:

NASA Software Documentation Standard Software Engineering Program, NASA-STD-2100-91, Technical Standards Div. Publication, July 29, 1991

XTE CCSDS Advanced Orbiting Systems Telemetry and Telecommand Implementation Document, August 7, 1992

XTE SOC Software Development Plan, SOC-STN-017, V0.2, September 7, 1993

3.0 OVERVIEW of PURPOSE and FUNCTIONS

An overview of the operation and function of the simulator is provided in the following sections. The complete simulator system is now comprised of six files:

- PCAhksim.c - The C program, menu or command line driven, screen & file output
- CommandVal.c - The C include file to update the commandable parameters using input piped from the Command Editor program. Reads the command instructions.
- Conversion.c - The C include file to read in the standard conversion parameter file
- ParaVal.c - The C include file to read in the file specifying the initial required telemetry parameter values
- pcaconv.dat - A data file of the same conversion parameters supplied to and used by the SOC H&S subsystem (the same file). These parameters are used to scale the physical numbers in the above data file to binary values which when converted back by H&S should reproduce the expected input values
- pcasimval.dat - A data file of the initial required telemetry parameter values

Note:

The simulator can be used as a stand alone packet generator or its output can be changed to reflect commands sent to it on <stdin> via a pipe by the PCA command editor.

3.1 Running the Simulator Program

There are three ways of running the program:

1. typing <PCAhksim> without options runs the program using the default command line settings
2. typing <PCAhksim -*****> runs the program using the selected command line options
3. typing <PCAhksim -m> displays the menu for further command selections

The simulator is always ready to receive commands on <stdout>, should any arrive, so no special option is needed to activate this feature.

3.2 Menu Selections

Starting the program with the menu switch selected offers a choice of nine options. The menu display also includes the current version number.

- The first four options can be either ON or OFF
- For the next three options concerning the clock only one can be set to ON at a time
- Pressing the <8> key runs the program
- Pressing the <9> key quits the program
- Pressing <Ctrl C> at any time will terminate a running program when the next packet is produced (up to 8 seconds delay). The exit will close any open output file and return to the menu if active or else quit to the UNIX prompt.

The menu options are:

```
        XTE - PCA HOUSEKEEPING SIMULATOR
                    VER 2.2

1    TEST MODE OUTPUT    [OFF]
2    SAVE OUTPUT FILE    [OFF]
3    NOISE COMPONENT     [OFF]
4    STAIRCASE RATES     [OFF]
5    FALSE CLOCK         [OFF]
6    REAL CLOCK          [ON]
7    HISTORY CLOCK       [OFF]
8    RUN
9    QUIT

Enter >
```

3.3 Menu Test / Setup Options

The first four of the nine options are for special test modes.

3.3.1 TEST MODE OUTPUT

The test output shown in Appendix A can be turned ON or OFF by toggling menu option 1. This output tabulation is only intended for testing during program development.

3.3.2 SAVE OUTOUT FILE

The final binary output file can be saved as a disc file for later replay into other programs. The writing of the file is controlled by toggling the menu option 2. The name of the standard output file is SIMOUT.DAT

3.3.3 NOISE COMPONENT

The parameters that can have noise superimposed on them are shown in the tabulation in Appendix B. The noise component can be turned ON and OFF by toggling menu option 3. Setting it to OFF means that EVERY digital parameter in a decoded housekeeping display (Health & Safety real time display) must have the value shown in the table. Floating point variables should be identical within digitization errors.

3.3.4 STAIRCASE RATES

When option 4 is set to ON five of the rates in sub address 16 which are presented in the Appendix are overwritten in a special way. The intention is to produce a regular and repeatable change in the counting rates in these channels. Every time a new CCSDS packet is built for a particular PCU the assigned rate channel is incremented by the value indicated in the table below. The count rates simply cycle modulo 255.

RATE CHANNEL	PCA No.	INCREMENT
evL1	1	1
evR1	2	2
evL2	3	4
evR2	4	8
evL3	5	16

3.4 Menu Timing Options

The next three menu options are used for selecting the way the program runs with respect to the real time clock in the workstation. Only one of these three at a time can be set to ON. The other two will be forced to OFF with a priority system taking the highest active one in the 5 - 7 range. The time code in the CCSDS packet header is assumed to start at 0 hrs on January the 1st in the current year.

3.4.1 FALSE CLOCK

To generate data rapidly set this option to ON. The clock time in the work station is then ignored and the program simply generates CCSDS packets at high speed with the timing field in consecutive headers incremented by 8 (seconds) starting from zero. The packet counter increments from zero each time the program is re-run.

3.4.2 REAL CLOCK

To generate data packets at 8 second intervals using the work station clock this option should be set to ON. This simulates the rate at which packets would be arriving from the real spacecraft.

The time code in the generated CCSDS packet headers is the real work station clock time. This is taken as starting from zero at 0 hrs on January the 1st in the current year. The packet counter increments from zero each time the program is re-run.

3.4.3 HISTORY CLOCK

This case is a hybrid of the previous two. The program produces packets for a set number of minutes (normally one orbits worth or ~92 minutes) in a quick “flash” and then slows to the 8 second output rate. This allows a “history” of data to exist for examining real time plots in the Health & Safety system. The “flash” period is set to “n” minutes using the #define HCLOCK n line in the program header.

The time code in the generated CCSDS packet headers commences at the work station clock time and increments without discontinuity through the transition from rapid “flash” to samples at 8 second intervals. The packet time would then be in advance of the actual work station clock time.

3.5 Command Line Options

The following set of characters are valid as command line options:

[- tToOnNsSfFrRhHmM]

Typing any character not in this list will result in an error and the program will terminate. It is not necessary to specify a choice for every ON/OFF selection when using the command line run option. The character codes are:.

COMMAND LINE OPTION	ON	OFF
TEST MODE OUTPUT	t	T
SAVE OUTPUT FILE	o	O
NOISE COMPONENT	n	N
STAIRCASE RATES	s	S
FALSE CLOCK	f	F
REAL CLOCK	r	R
HISTORY CLOCK	h	H
MENU DISPLAY	m	M

3.6 Simulation Parameters

The basic parameters are all assigned values as shown in the section on Standard Data and in Appendix B. These values are in some cases somewhat arbitrary but generally reflect realistic digital values within the RED and YELLOW housekeeping limits. Some “un-real” bit combination have been selected occasionally in order to exercise all bit locations and therefore all housekeeping decoding and display functions.

The data tables for the simulation can be easily changed since it exists as a separate file <pcasimval.dat>. It is recommended that a copy of this file be retained as the data values have been carefully chosen to be fairly representative of the PCA experiment.

NOTE

All the integer count rates returned from the PCA experiment are expressed as the sum counted over an 8 second integration period.

3.6.1 Simulation Noise Additions

The tables provided for Standard Data (see later) and also given in Appendix B show the noise contribution for each parameter where appropriate.

- All counting rates have a root N component added to simulate random noise. The simulated noise extends over the range +-6 sigma and assumes normal statistics.
- Many other parameters such as voltages, temperatures and pressures have noise added in the range +-6 sigma where the quantity sigma is also defined in the data tables

The noise levels can be edited to introduce different signal variability.

3.6.2 Conversion Parameters

The conversion parameter table given in Appendix C is the same as that used by the SOC H&S subsystem. The parameters convert the physical values described in Appendix A to binary values. Most conversions follow a linear law of the form:

$Y = Ax + B.$

Appendix B contains the Y's and Appendix C the A & B values. The program therefore uses a equation of the form

digital (binary) value $x = (Y - B) / A$

The Standard Data rates given in Appendix B are in counts per second so these are multiplied by 8 in the program to reflect the 8 second integration sums carried out in the PCA RIF units. The H&S subsystem will divide the incoming rate values by 8 to return them to c/sec.

3.6.3 Commandable Features

The PCA command editor can be run in a mode where command packets are output via a pipe. The simulator can the read these "input" command instructions.

Only the bit patterns for commandable features can be changed in this way:

- discriminator LLD, VLE, alpha & dither settings
- TPG settings & controls
- all HV settings & controls
- all relay settings
- override controls & "B" side choice
- HRM value
- Diagnostics

Control of the diagnostic command settings is possible but not very meaningful.

3.6.4 Simulator Realism & Level Of Intelligence

At present (for Build 3) the simulator has no intelligence at all. This means that although the command bits in the generated packets may change the other data items will not reflect this in any realistic way i.e. if the HV's are turned off the detector will still be producing x-ray counts.

Some refinements of this sort may be added for Build 4.

3.7 Program Output

The output of the program is simple a series of packets constructed from bytes. The user is encouraged to adapt and modify their own version of the program to either incorporate it in existing code or pipe the output to other running applications.

3.8 CCDS Packet Structure

It is not proposed to describe in detail here the complete CCSDS packet structure. These details can be found in the PCA Team document referenced earlier. Here we just describe the array names used for each part of the packet and the program code itself contains many instructive comment lines. The basic data structure of a packet for each PCU is:

- The CCSDS packet header (14 bytes)
- Rates used by HRM (32 bytes)
- Rates not used by HRM (24 bytes)
- Command and board status (12 bytes)
- Analog values (40 bytes)
- Diagnostics (20 bytes).

Each call for another packet generates typical status information and delivers a packet in a two-dimensional array C733P_master[*id*][*nbytes*], where the first column, with a range of 0 to 4, corresponds to the PCU detector and the second column relates to the 142 bytes of data for the corresponding detector. The data packet for each detector is generated by invoking a function which then invokes a series of other functions that generate the different parts of the packet by constructing a set of arrays. This series of arrays is defined below.

3.8.1 CCSDS Packet Header

A typical CCSDS packet header consisting of a total of 14 bytes is built into array sub_1[...]. The sample data of the parameters contained in the header is fixed except for the sequence count location and the time code location. The time code can be generated in a number of ways. The current time is obtained and reduced to seconds since the beginning of the year.

3.8.2 Count Rates used by HRM (sub add. 16)

The eight count rates obtained from the RIF which are used by the High Rate Monitor (HRM) are available through sub address 16 and are built into the array sub_16[...]. Each 32 bit word consists of a 24 bit counter for each individual rate which is split into three bytes and stored in the array.

The sample data of count rates for each detector is contained in a two-dimensional array RIF16[...][...]. Since the on-board HRM works over 8 second intervals all these rates are expressed in counts expected over an 8 second summation. An option exists to add some "root N" noise to the rates as they are built into the array.

3.8.3 Count Rates not used by HRM (sub add. 17)

The six additional count rates obtained from the RIF which are not used by the HRM are available through sub address 17 and are built into the array sub_17[...]. Each 32 bit word consists of a 24 bit counter for each individual rate which is split into three bytes and stored in the array.

The sample data of count rates for each detector is contained in a two-dimensional array RIF17[...][...]. Since the on-board HRM works over 8 second intervals all these rates are expressed in counts expected over an 8 second summation. An option exists to add some "root N" noise to the rates as they are built into the array.

3.8.4 Command and Board Status (sub add. 18)

The status of commands received by each RIF/PCU is available through sub address 18 and is built into array sub_18[...].

Enquiries to sub address 18 produce highly compressed bit structures. The sample data for each detector is contained in a two-dimensional array RIF18[...][...]. None of these parameters can have noise added to then.

3.8.5 Analog Information (sub add. 19)

The analog channels of each detector that monitor various parameters using RIF sub address 19 are built into array sub_19[...].

Sub address 19 provides 20, 16 bit words to define temperatures, voltages and pressures on each detector. The 16 bit word, with LS 12 bits containing the actual data and the remaining MS 4 bits set to zero, is split into two bytes and stored in the array. The sample information is for each detector is contained in the array RIF19[...][...]. An option exists to add some noise to the sample data before the 40 bytes are built into the array.

3.8.6 Diagnostic Status (sub add. 29)

The diagnostic information for each RIF/PCU is available through sub address 29 and is built into array sub_29[...].

Sub address 29 provides a total of 10, 16 bit words for each detector. The sample information for each detector is contained in array RIF29[...][...]. An option exist to add some "root N" noise to the diagnostic rate channels as they are built into the array.

3.9 Interface To CMD EDitor & HS Subsystem

The way in which the simulator interacts with external programs and inputs is illustrated in Figure1. The pipe connection into or out of the simulator are optional as it will run as a stand alone unit to generate data files composed of typical PCA housekeeping telemetry packets.

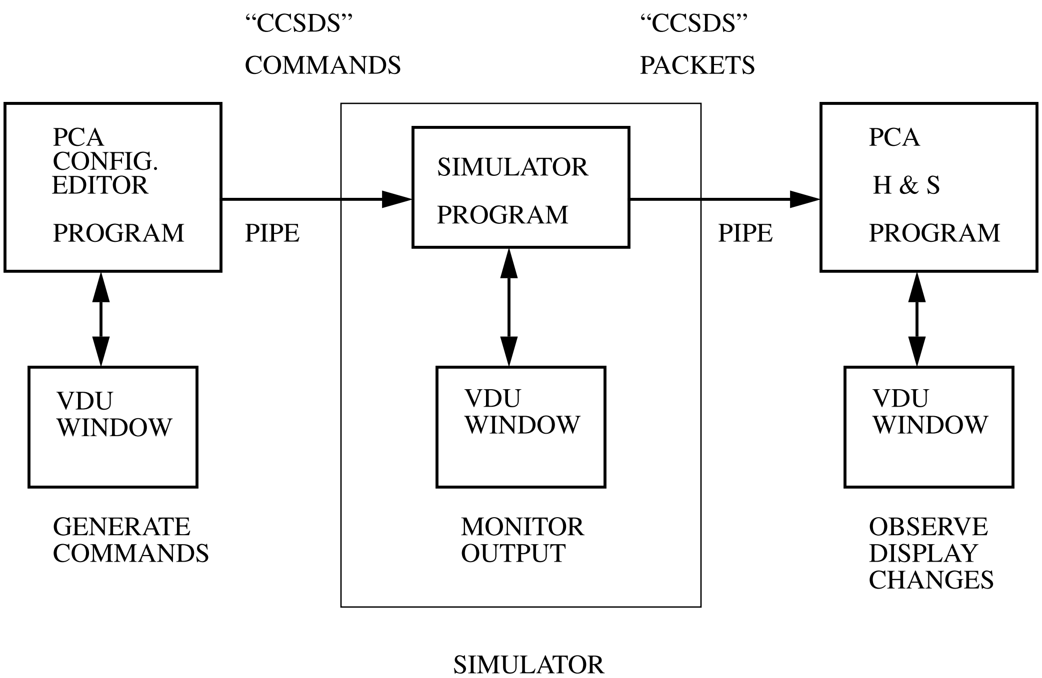

FIGURE 1. Interfaces to Simulator

3.9.1 Use of C Pipe Feature - MASTER TEST

The key points to remember are:

- Commands are only passed to the simulator from the PCA Configuration Editor when that output option is activated within the editor program. The output is on <stdout>
- The simulator is always ready to receive commands on its <stdin> should any arrive. This ability does not need to be activated
- The simulator outputs the generated packets on <stdout>. This output must be activated by an option in the simulator program
- The PCA housekeeping H&S display program can read packets on its <stdin>. This data input option must be selected from the programs menu structure

The final form of the execute command to perform an end to end test is then:

PCAconfigEd | PCAhksim | PCAdisplay

4.0 INSTALLATION and INITIALIZATION

Six files are required to compile the program:

- PCAhksim.c - main program
- Include files:
  CommandVal.c - read incoming commands from editor
  Conversion.c - reads conversion file
  ParaVal.c - reads parameter file
- pcaconv.dat - conversion file
- pcasimval.dat - parameter data file

To compile the program use a line like:

cc -o PCAhksim PCAhksim.c -lm

The command line settings shown are needed to link in the required libraries.

5.0 STARTUP and TERMINATION

The simulator is invoked by typing from the UNIX system prompt:

PCAhksim (uses default settings)

PCAhksim -m (provides menu options)

*PCAhksim -***** (allows full command line options)*

The program will take a moment to execute and will terminate normally after writing the output to a binary file if this option is active. Pressing <Ctrl C> at any time will terminate the program when the next packet is produced (possibly an 8 second delay). Any open output file will be closed and a return made to the UNIX prompt or the menu if that had been active.

See section 3 for a full description of the menu and command line options.

See section 9.3 shows how to link the simulator to other PCA software supplied to the SOF.

6.0 FUNCTIONS and their OPERATION

Not applicable to this software.

7.0 ERROR and WARNING MESSAGES

Normally the program will simply execute as expected.

There is one failure that may occur if the menu option is requested and the command window is not large enough to accept its required screen size.

An error message will say:

Segmentation fault (core dumped)

and return to the UNIX system prompt. The command window will have to be closed and another one opened. Then increase the command window size and try again.

8.0 RECOVERY STEPS

None - Simply restart the program.

9.0 ABBREVIATIONS and ACRONYMS

RIF - Remote InterFace Unit

PCA - Proportional Counter Array (all detectors)

PCU - Proportional Counter Unit (single detector)

HRM - High Rate Monitor

10.0 GLOSSARY

None

11.0 DEVELOPMENT BUILD PLAN

The development of the PCA telemetry simulator is intended to follow the series of Delivery Levels given below. In the past level 1 has corresponded to SOC Build 1 and level 2 to Build 2 but this will not necessarily continue for all future deliveries.

Level 1 - released as Build 1 (Framework, 8/3/93)

- Built the full packet structure with headers and counters for all PCU’s
- Specified a few parameters and varied them in a controlled way (staircase ramps). The remaining data fields were empty

Level 2 - delivered for Build 2 (End to End, 12/31/93)

- All parameters simulated
- Most parameters given typical values - these are specified in an external data file to facilitate changing them on the fly
- All parameters made slightly different in value (except some digital bits) to uniquely identify them for SOC subsystem testing - therefore some combinations were not realistic of a default or flight configuration
- For realism some parameters can have suitable noise superimposed. The noise component (root N for rates) can be toggled on and off
- This delivery provides “standard PCA data packets” and “standard output results”. All parameters have expected values that can be checked in the H&S and Science Monitoring subsystems
- The simulator uses exactly the same parameter conversion file as supplied to the SOC H&S subsystem. This ensures that a voltage of say 7.1 volts specified in Appendix B is converted to a binary value (in the packets) and then back again to 7.1 volts within H&S.

Level 3 - delivery for Build 3 (Mission Critical, 5/31/94)

- PCA desired configuration editor linked to simulator to change command item settings - no changes to count rates, analog readings or any other monitored data items
- The simulator will be delivered in a form that is convenient for the SOC to easily insert into their Data Generator Subsystem framework

Level 4 - Delivered for Build 4 (Full functionality, 11/30/94)

- No further developments made - simulator was frozen
- The conversion parameter file was split up into 5 parts, one for each address slot. This was done to keep the file structure in line with that used in the PCA H&S subsystem where these files originate

12.0 Canonical FITS Test Data & Test Plan

The SOF decided in mid 1994 to request that the instrument teams (IT) provide a sample set of packets to be:

- Passed through Ingest & SOF systems
- Produce a FITS file using XFF

The IT's were then to carefully inspect the output FITS file and confirm that it matched the expected form and content for the input data packets. Such a pair of approved data & FITS files would then form a "canonical" set. The intension was that the packet data could be re-run at any time and the new FITS file compared with the old. This would be expected to be identical unless some "error" had crept into the SOF code during further development.

PCA have produced a canonical set of data packets for PCU 1 (all PCU's are equivalent except for the APiD value). This standard set was generated using the Housekeeping Simulator program using the standard default set of parameter values. The test file consists of:

- 10 packets with constant parameter values
- 10 packets with the noise option turned on

The final XFF FITS file was checked using the FDUMP tool in the FTOOLS suite. This produces neat tabular output so that visual comparisons with the input values can be made. It is important that the simulator run generating the test packets uses the same set of conversion parameters as does the PCA accessor when XFF creates the FITS file. Changing some conversion values or moving detector addresses can lead to problems unless the file usage is consistent. Any such change invalidates the option of a simple yes/no file comparison between the new and old FITS files. Also a precise match with the default values in the simulator default file is not possible for any parameter that uses conversion terms (not a simple count or binary value). A match should exists however at about the 1 bit resolution level. All integer or binary values should be identical.

13.0 STANDARD SIMULATION TEST DATA

RATES - HRM

[......] - denotes rate channels used for staircase overwrite

Array	Bits	Mnemonic	PCU1	PCU2	PCU3	PCU4	PCU5	Noise
RIF16[][0]	24	evL1	[550]	600	650	700	750	$\sqrt{n}$
RIF16[][1]	24	evR1	530	[580]	630	680	730	$\sqrt{n}$
RIF16[][2]	24	evL2	320	340	[360]	380	400	$\sqrt{n}$
RIF16[][3]	24	evR2	315	335	355	[375]	395	$\sqrt{n}$
RIF16[][4]	24	evL3	60	70	80	90	[100]	$\sqrt{n}$
RIF16[][5]	24	evR3	55	65	75	85	95	$\sqrt{n}$
RIF16[][6]	24	evVX	1800	1850	1900	1950	2000	$\sqrt{n}$
RIF16[][7]	24	evVP	155	160	165	170	175	$\sqrt{n}$

RATES - non HRM

Array	Bits	Mnemonic	PCU1	PCU2	PCU3	PCU4	PCU5	Noise
RIF17[][0]	24	evVPvle	42	44	46	48	50	$\sqrt{n}$
RIF17[][1]	24	evXEvle	32	34	36	38	40	$\sqrt{n}$
RIF17[][2]	24	ev2ormore	22	24	26	28	30	$\sqrt{n}$
RIF17[][3]	24	evAlpha	180	185	190	195	200	$\sqrt{n}$
RIF17[][4]	24	evVXvle	12	14	16	18	20	$\sqrt{n}$
RIF17[][5]	24	evXEgood	1300	1350	1400	1450	1500	$\sqrt{n}$

COMMAND PARAMETERS

Array	Bits	Mnemonic	PCU1	PCU2	PCU3	PCU4	PCU5	Noise
RIF18[][0]	2	dsR2	0	1	2	3	0	N/A
RIF18[][1]	2	dsL2	1	2	3	0	1	N/A
RIF18[][2]	2	dsR1	2	3	0	1	2	N/A
RIF18[][3]	2	dsL1	3	0	1	2	3	N/A
RIF18[][4]	2	dsVP	0	11	2	3	0	N/A
RIF18[][5]	2	dsVX	1	2	3	0	1	N/A
RIF18[][6]	2	dsR3	2	3	0	1	2	N/A

Array	Bits	Mnemonic	PCU1	PCU2	PCU3	PCU4	PCU5	Noise
RIF18[][7]	2	dsL3	3	0	1	2	3	N/A
RIF18[][8]	1	ADCdither	0	1	0	0	0	N/A
RIF18[][9]	2	dsVle	0	11	2	3	1	N/A
RIF18[][10]	2	dsAlpha	3	12	1	0	2	N/A
RIF18[][11]	1	tpgR3	0	1	0	0	0	N/A
RIF18[][12]	1	tpgL3	0	0	1	0	0	N/A
RIF18[][13]	1	tpgR2	0	0	0	1	0	N/A
RIF18[][14]	1	tpgL2	0	0	0	0	1	N/A
RIF18[][15]	1	tpgR1	1	0	0	0	0	N/A
RIF18[][16]	1	tpgL1	0	1	0	0	0	N/A
RIF18[][17]	1	tpgAlpha	0	0	1	0	0	N/A
RIF18[][18]	1	tpgVP	0	0	0	1	0	N/A
RIF18[][19]	1	tpgVX	0	0	0	0	1	N/A
RIF18[][20]	1	tpgMode	0	1	0	0	0	N/A
RIF18[][21]	1	tpgEn	1	1	0	0	0	N/A
RIF18[][22]	4	cmdhvXE	15	8	4	0	1	N/A
RIF18[][23]	4	cmdhvPR	14	12	7	1	0	N/A
RIF18[][24]	1	plsPRon	0	0	1	0	0	N/A
RIF18[][25]	1	plsPRoff	0	0	0	1	0	N/A
RIF18[][26]	1	plsXEon	0	0	0	0	1	N/A
RIF18[][27]	1	plsXEoff	0	0	1	0	0	N/A
RIF18[][28]	1	plslvon	0	0	0	1	0	N/A
RIF18[][29]	1	plslvoff	0	0	0	0	1	N/A
RIF18[][30]	1	hvPRrly	1	1	1	1	0	N/A
RIF18[][31]	1	hvXErly	1	1	1	1	0	N/A
RIF18[][32]	1	lvpsrly	1	1	1	1	0	N/A
RIF18[][33]	1	hvPRon	1	1	1	1	0	N/A
RIF18[][34]	1	hvXEon	1	1	1	1	0	N/A
RIF18[][35]	1	se1B	0	1	0	0	0	N/A
RIF18[][36]	1	rateOv	0	0	1	0	0	N/A
RIF18[][37]	1	ppsOv	0	0	0	1	0	N/A
RIF18[][38]	1	hrmOv	0	0	0	0	1	N/A
RIF18[][39]	1	hvinh3	0	0	1	0	0	N/A
RIF18[][40]	1	hvinh2	0	0	0	1	0	N/A
RIF18[][41]	1	hvinh1	0	0	0	0	1	N/A
RIF18[][42]	1	hrmFlag	0	1	1	1	1	N/A
RIF18[][43]	24	hrm	2048	4096	8192	16384	32768	N/A

ANALOG PARAMETERS

Array	Bits	Mnemonic	PCU1	PCU2	PCU3	PCU4	PCU5	Noise
RIF19[][0]	12	tmpspare	0.00	0.00	0.00	0.00	0.00	0.25
RIF19[][1]	12	tmpRIF	20.50	21.20	20.70	19.80	20.00	0.25
RIF19[][2]	12	tmpADC	15.30	14.90	15.00	16.10	14.50	0.25
RIF19[][3]	12	tmpCSA	18.20	19.00	18.90	19.50	19.20	0.25
RIF19[][4]	12	tmpSA	11.00	11.80	10.80	11.20	11.50	0.25
RIF19[][5]	12	tmpTPG	23.50	22.90	22.50	23.30	23.00	0.25
RIF19[][6]	12	tmpLVPS	13.10	12.80	13.70	13.00	12.50	0.25
RIF19[][7]	12	tmpHVPS	17.50	16.80	16.50	17.00	17.90	0.25
RIF19[][8]	12	vap12	11.90	11.95	12.00	12.05	12.10	0.02
RIF19[][9]	12	vam12	-12.10	-12.05	-12.00	-11.95	-11.90	0.02
RIF19[][10]	12	vap6	5.92	5.96	6.00	6.04	6.08	0.02
RIF19[][11]	12	vam6	-6.08	-6.04	-6.00	-5.96	-5.92	0.02
RIF19[][12]	12	vdp12	11.80	11.90	12.01	12.10	12.20	0.05
RIF19[][13]	12	vdm12	-12.20	-12.10	-11.99	-11.90	-11.80	0.05
RIF19[][14]	12	vdp5	4.95	5.05	5.00	4.98	5.02	0.05
RIF19[][15]	12	vspare	0.00	0.00	0.00	0.00	0.00	0.05
RIF19[][16]	12	prsXE	845.0	840.0	848.0	850.0	843.0	0.75
RIF19[][17]	12	prsPR	795.0	797.0	800.0	799.0	803.0	0.75
RIF19[][18]	12	hvXE	2190.0	2050.0	1970.0	910.0	1910.0	0.50
RIF19[][19]	12	hvPR	2840.0	2800.0	2700.0	2580.0	1580.0	0.50

DIAGNOSTIC PARAMETERS

Array	Bits	Mnemonic	PCU1	PCU2	PCU3	PCU4	PCU5	Noise
RIF29[][0]	24	c1count	51	61	71	81	91	$\sqrt{n}$
RIF29[][1]	4	c1selN	1	2	3	4	5	N/A
RIF29[][2]	1	c1testbit	0	0	0	0	1	N/A
RIF29[][3]	1	c1status	0	1	0	1	0	N/A
RIF29[][4]	24	c2count	151	161	171	181	191	$\sqrt{n}$
RIF29[][5]	4	c2se1N	3	4	5	6	7	N/A
RIF29[][6]	1	c2testbit	0	0	0	0	1	N/A
RIF29[][7]	1	c2status	1	0	1	0	1	N/A

Array	Bits	Mnemonic	PCU1	PCU2	PCU3	PCU4	PCU5	Noise
RIF29[][8]	1	mckspeed	0	1	0	0	1	N/A
RIF29[][9]	1	mckgatechip1	0	1	1	1	1	N/A
RIF29[][10]	1	mckgatechip2	1	0	1	1	1	N/A
RIF29[][11]	3	mckselect	4	5	6	7	3	N/A
RIF29[][12]	8	lastcmd	201	211	221	231	241	N/A
RIF29[][13]	1	to1pps	1	1	1	0	1	N/A
RIF29[][14]	1	torate	0	1	0	1	1	N/A
RIF29[][15]	1	waitfor1pps	1	0	0	1	0	N/A
RIF29[][16]	1	twohrms	0	1	1	0	1	N/A
RIF29[][17]	1	onehrm	0	1	1	0	1	N/A

14.0 APPENDICES

APPENDIX A

Sample output of the PCA Status Telemetry Simulator in ‘TEST’ mode.

>

Running PCA Status Telemetry Simulator

CCSDS header data block

00001000	8	01011010	90
11000000	192	00000000	0
00000000	0	10000111	135
00000000	0	11000010	194
11000101	197	11001001	201
00000000	0	00000000	0
00000000	0	00000000	0

Sub Address 16 data block

00000000	0	00000000	0	00001111	15	10001111	143
00000000	0	00000000	0	00010000	16	10111101	189
00000000	0	00000000	0	00000110	6	00101101	45
00000000	0	00000000	0	00000110	6	11011010	218
00000000	0	00000000	0	00000010	2	00000100	4
00000000	0	00000000	0	00000001	1	11110111	247
00000000	0	00000000	0	00000110	6	01000100	68

00000000	0	00000000	0	00000011	3	00101000	40

Sub Address 17 data block

00000000	0	00000000	0	00000000	0	00100111	39
00000000	0	00000000	0	00000000	0	00111111	63
00000000	0	00000000	0	00000000	0	00011011	27
00000000	0	00000000	0	00000000	0	10101011	171
00000000	0	00000000	0	00000000	0	10011010	154
00000000	0	00000000	0	00011001	25	00100001	33

Sub Address 18 data block

00011011	27
00011011	27
00000011	3
00000010	2
00000001	1
11011011	219
00000000	0
00011111	31
00000001	1
00000010	2
00000010	2
10000000	128

Sub Address 19 data block

00000000	0	00000000	0
00001011	11	00000111	7
00001001	9	10001101	141
00000000	10	00101011	43
00001000	8	01001100	76
00001011	11	11001011	203
00001000	8	11011111	223
00001010	10	01001110	78
00001000	8	01001100	76
00000001	1	11011010	218
00000110	6	11000101	197
00000011	3	01010010	82
00001000	8	01110101	117
00000001	1	11110001	241
00000110	6	10100101	165
000001010	5	00110110	54
00000111	7	00101111	47
00001001	9	11111111	255
00001001	9	01101101	52
00000010	2	01101101	109

Sub Address 29 data block

00000000	0	01111000	120
01111000	120	01111000	120
00000000	0	00000000	0

00000000	0	00110000	48
00000000	0	11010010	210
11010010	210	11010010	210
00000000	0	00000000	0
00000000	0	00000000	21
00001100	12	00110010	50
00000000	0	00010100	20

Entire Packet Byte Length = 142

The following group of lines are always output to the screen for each packet from each detector while using the menu driven version.

CCSDS	14 bytes of packet header data
HRM rates	32 bytes of the 8 HRM rates
Non HRM rates	24 bytes of the 6 non HRM rates
Cmds & Status	12 bytes of command and status data
Temperatures	16 bytes of the 8 analog temperature readings
Voltages	16 bytes of the 8 analog voltage readings
Pressure & HV	8 bytes of the two analog pressure & HV readings
Diagnostics	20 bytes of the diagnostic information

A quick glance at the first line of each output block confirms that the simulator is running correctly.

1) Verify in the headers that the APid’s run in the sequence 90, 91, 92, 92, 94

2) Verify in the headers that the time code increments by 8 seconds

3) Verify in the rates that they are constant across time updates or vary a bit depending on the selected mode

APPENDIX B

PCA Status Telemetry Data - “include file” pcasimval.dat

(see following listing)

APPENDIX C

PCA Conversion Parameters - “data file” pcaconv.dat

(see following listing)