1. INTRODUCTION

This document specifies the set of tests to be performed to check the correct integration of the SPIRE warm electronics equipment with the SPIRE FPU simulator (stand alone configuration). It also assesses on the checks to perform to verify the success or failure of a particular test.

1.1 Scope

To judge the success or failure of a set of bench tests by checking that:

- Commands generated for a particular test by CUS and sent via SCOS2000 are correctly received and executed by the correspondent DRCU subsystem unit.
- No error or exception reports were generated during the execution of the commands.
- The appropriate telemetry was generated and the correspondent parameters changed in an expected manner.

1.2 Reference Documents

RD01 SPIRE Data ICD (SPIRE-RAL-PRJ-001078), Issue 1.1, 25th May2004

RD02 SPIRE EGSE-ILT Startup Procedures (SPIRE-RAL-DOC-001630), Issue 0.7, 24th June 2003

RD03 SPIRE DRCU Swith ON Procedure (SPIRE-RAL-PRC-002222), Issue 0.1, 10th June 2004

RD04 DRCU/DPU Interface Control Document (Sap-SPIRE-CCa-076-02 Issue 1.0 14th Feb 2003

RD05 SPIRE DRCU Integration Test specification Issue1 (SPIRE-RAL-DOC-001799 5th Sep 2003

RD06 EGSE TestControl User Guide Draft 5 (PICC-ME-MN-002 16th December 2003)

1.3 Bench Test Configuration

1.3.1 SPIRE IEGSE Setup

- Connect warm electronics following Figure 1.
- CDMS Simulator version 2.5 (**installed**) on WINDOWS2000 (Gordon) **Login Info needed**
- SCOS 2000 2.3e Patch Level 5 + TOPE - running on a Linux SuSE 7.3 system. MIBs required:
  - SPIRE CQM Issue2 MIB (**installed**) on IEGSE machine **hoss2k4-1 (IP address: 192.168.202.102).**
  - CDMS Simulator MIB (**to be merged with CQM Issue 2)**.
- OBS Issue2 (**installed** written to EEPROM)
- SPIRE build 515 of the HCSS (**installed**) on a Linux SuSE 9.1 system (IEGSE machine **hosqla4-2)** (**IP address 192.168.202.111**) – includes the EGSE router and gateway.
- QLA – running on **hosqla4-2** (**IP address 192.168.202.111**)
- EGSE Test Tool PacketDisplay running on **hoslqla4-2**
- Test Control scripts (to be copied onto /home/ssop23e/tcl/TC directory on **hoss2k4-1**)

- CUS scripts to imported into HCSS database (db already created : **SPIRE_IEGSE1 on hossqla4-2**)

The following table must be filled in when the test is performed

Date:	
Time:	
Test conductor:	
Machine:	

1.3.2 SPIRE EGSE Setup Checks

Before any test can be executed the following applications must be running in the correspondent machines, to have information of how to achieve this refer to document RD02. Fill the blanks as appropriate when the test is performed:

SPIRE IEGSE component	Version	Status	HIEGSE Machine	Check	Comments
EGSE router	515	Running			
EGSE Gateway	515	Running			
CDMS Simulator	2.5	Running			
SCOS2000	2.3 patch level 5	Running			
TOPE environment	?	Started (EXIF task running)			
Packet Display	515	Running			
TM Ingestion	515	Running			

2. ON BOARD SOFTWARE STARTUP AND OBS CHEKS

To power ON the DPU and start up the OBS follow these steps:

Step #	Action	Comments	Check
1	Power ON 28V main power supply to HSDPU	Wait for voltage reading to stabilize at 28.8 V	
2	Power ON secondary power supply (button on the bottom right part of power supply)	Wait for current to stabilize at 0.42 A. At this moment an Event Report (5, 2) is received on packet display. The last word in the packet contains the number of errors found in the memory checks and should be 0.Check it.	
3	From SCOS TOPE environment run PROC-OPER-SPIRE-DPU-ON.tcl procedure	The command sent to the DPU is: FORCE_BOOT If the command is succesfull HK reports start appearing in the packet display.4 Event packets are generated at this moment and they account for the fact that the DRCU is still not ON so there is no subsytem response.	

Once the OBS is running HK reports should be generated automatically and packet display tool should show telemetry packets of (type, subtype) = (3, 25) with APID 0x500 (critical HK) and 0x502 (nominal HK). The Telemetry Display page DPU AND OBS PARAMETERS on SCOS 2000 displays all the DPU and OBS specific parameters from the nominal HK report. The Telemetry Display page CRITICAL HK PARAMETERS displays the entire contents of the critical HK report.

2.1 HK Generation Rate check

On the DPU AND OBS PARAMETERS display page check the following parameters:

Nominal HK parameter Name	Expected Behaviour	Actual Behaviour	Success/Failure	Comments
THSK	Incrementing once every second			
TM2N	Incrementing once every second			
TSYNC	Incrementing once every second			

On the CRITICAL HK REPORT display page check:

Critical HK parameter Name	Expected Behaviour	Actual Behaviour	Success/Failure	Comments
ALL	Updating once every 2 seconds			

2.2 DPU and OBS Parameter Monitoring

Some further monitoring checks must be performed before carrying on with the integration tests. These checks are described in the table below.

Nominal HK parameter Name	Expected Value	Actual Value	Success/Failure	Comments
OBSVER	OBS version as specified in the release note 0x2000 for version 2.0			
TMMODE	0 – Normal TM Mode			
DPUP5V	~5.0 V			
DPUP15V	~14.70 V			
DPUM15V	~-14.98 V			
DPUTEMP	~304.68K			
DPUP2_5V	~2.48V			

2.3 DRCU switch ON procedure

In order to switch on the DRCU follow these steps:

Step #	Action	Comments	Check
1	In TOPE run procedure **clear_SPIRE_HK_report.tcl**	Single click over the procedure the hit Run.Two commands to stop the HK request are sent to the OBS.Check then commands are successful. The SPIRE HK generation stops in the packet display.	
2	Power on the main power supply (Button on the top rear right side of the bench looking from in front)	The red light PRIME power on the front will come on.	
3	Power on the secondary power supply.Pull down (loaded) the spring in the front side of the DRCU.	The green light of the secondary power supply will come on.	
4	In TOPE run procedure **define_SPIRE_new_HK_report.tcl**	Single click over the procedure the hit Run.Two commands to restart the HK request are sent to the OBS.Check then commands are successful. The SPIRE HK generation restarts in the packet display. In the DPU AND OBS PARAMETERS display in SCOS,the HK parameter **MONSTAT=5**	

At this moment the DRCU is ON and both the DCU and the SCU are ready to receive and execute instrument commands.

3. TEST SPECIFICATION

3.1 Pre-test conditions

Perform the following action before the execution of the tests:

Step #	Action	HIEGSE Machine	Comments	Check
1	Start Test Control Server	TBD machine	Type "tescontrol-server" in a console in the appropiate machine and hit return.	

3.2 Test specification

For each integration test the behaviour of the relevant subsystem housekeeping parameters is monitored and compared to an expected value after the test is executed. Tables for each of the tests are located in Annex1 at the end of this document.

The following test will be executed for the warm electronics stand alone test:

1. **Test_SPIRE_BENCH01:** DCU High Speed Link Test in nominal mode (fixed number of frames).
2. **Test_SPIRE_BENCH02:** DCU High Speed Link Test in test pattern mode (fixed number of frames).
3. **Test_SPIRE_BENCH03:** DCU High Speed Link Test in nominal mode (1 min continuous generation).
4. **Test_SPIRE_BENCH04:** SCU High Speed Link Test in nominal mode (fixed number of frames).
5. **Test_SPIRE_BENCH05:** SCU High Speed Link Test in test pattern mode (fixed number of frames).
6. **Test_SPIRE_BENCH06:** SCU High Speed Link Test in nominal mode (1 min continuous generation).
7. **Test_SPIRE_BENCH07**: DCU High Speed Link Test in nominal mode (long period generation)
8. **Test_SPIRE_BENCH08:** Maximum telemetry rate test
9. **Test_SPIRE_BENCH09:** Maximum number of packets test

3.3 Test execution

To execute one of these particular tests single click over its name in the TOPE window and press Run button (just beside).A window will pop-up requesting for the test procedure parameters.

3.4 Test description and purpose

Test_SPIRE_BENCH01: DCU High Speed Link Test in nominal mode (fixed number of frames)

A fixed number of DCU nominal science frames are requested at nominal bias and sampling frequencies for both photometer and spectrometer.
This is done for each one of the DCU science types (bands) both photometer and spectrometer.
The purpose is to check that the DCU produces the number of frames requested.

Test_SPIRE_BENCH02: DCU High Speed Link Test in test pattern mode (fixed number of frames)

A fixed number of DCU test pattern frames are requested at nominal bias and sampling for both photometer and spectrometer.
This is done for each one of the DCU science types (bands) both photometer and spectrometer.
The purpose is to check that the DCU produces the correct frames as specified in RD01.

Test_SPIRE_BENCH03: DCU High Speed Link Test in nominal mode (continuous generation)

Continuous generation of DCU nominal science frames is requested at nominal bias and sampling frequencies for both photometer and spectrometer for a period of time.
This is done for each one of the DCU science types (bands) both photometer and spectrometer.
The purpose is to check that the DCU produces data continuously at the highest rate possible for a small period of time.

Test_SPIRE_BENCH04: SCU High Speed Link Test in nominal mode (fixed number of frames)

A fixed number of SCU science frames are requested at nominal sampling frequency.
The purpose is to check that the SCU produces the number of frames requested.

Test_SPIRE_BENCH05: SCU High Speed Link Test in test pattern mode (fixed number of frames)

A fixed number of SCU test pattern frames are requested at nominal sampling frequency.
The purpose is to check that the SCU produces the correct frames as specified in RD01.

Test_SPIRE_BENCH06: SCU High Speed Link Test in nominal mode (continuous generation)

Continuous generation of SCU nominal science frames is requested at nominal sampling frequency for a period of time.
The purpose is to check that the SCU produces data continuously for a small period of time.

Test_SPIRE_BENCH07: DCU High Speed Link Test in nominal mode (continuous generation for a long period)

Continuous generation of DCU nominal science frames is requested at the highest sampling possible for photometer or spectrometer for a period of time.

The purpose is to check that the DCU produces data continuously at the highest rate possible for as long as possible period of time.

Test_SPIRE_BENCH08: Maximum number of packets test:

A maximum of 27 sub-frames are allocated for the prime instrument in the normal telemetry mode per second. Continuous generation of DCU full photometer nominal science frames is requested at 25 Hz sampling possible taking into account that 1 sub-frame must be allocated for HK and another for possible commands acknowledgement or event reports.
The purpose is to verify this particular requirement.

Test_SPIRE_BENCH09: Maximum telemetry rate test:

The maximum average telemetry rate available is 130Kbps. A higher rate is permitted for a period of time. Sampling of full spectrometer nominal science frames at ~ 140 Hz is requested to check for this telemetry high rate.
The purpose is to verify this particular requirement.

NOTE: No MCU testing is to be carried out in the CQM model as the OBS is compliant with QM1 models (both SCU and MCU are QM1 models) whereas the MCU is model QM0.This means that the frames produced by the MCU don't have the correct length expected by the OBS and event packets would be generated unnecessarily through out the test.

4. WARM ELECTRONICS STAND ALONE CONFIGURATION

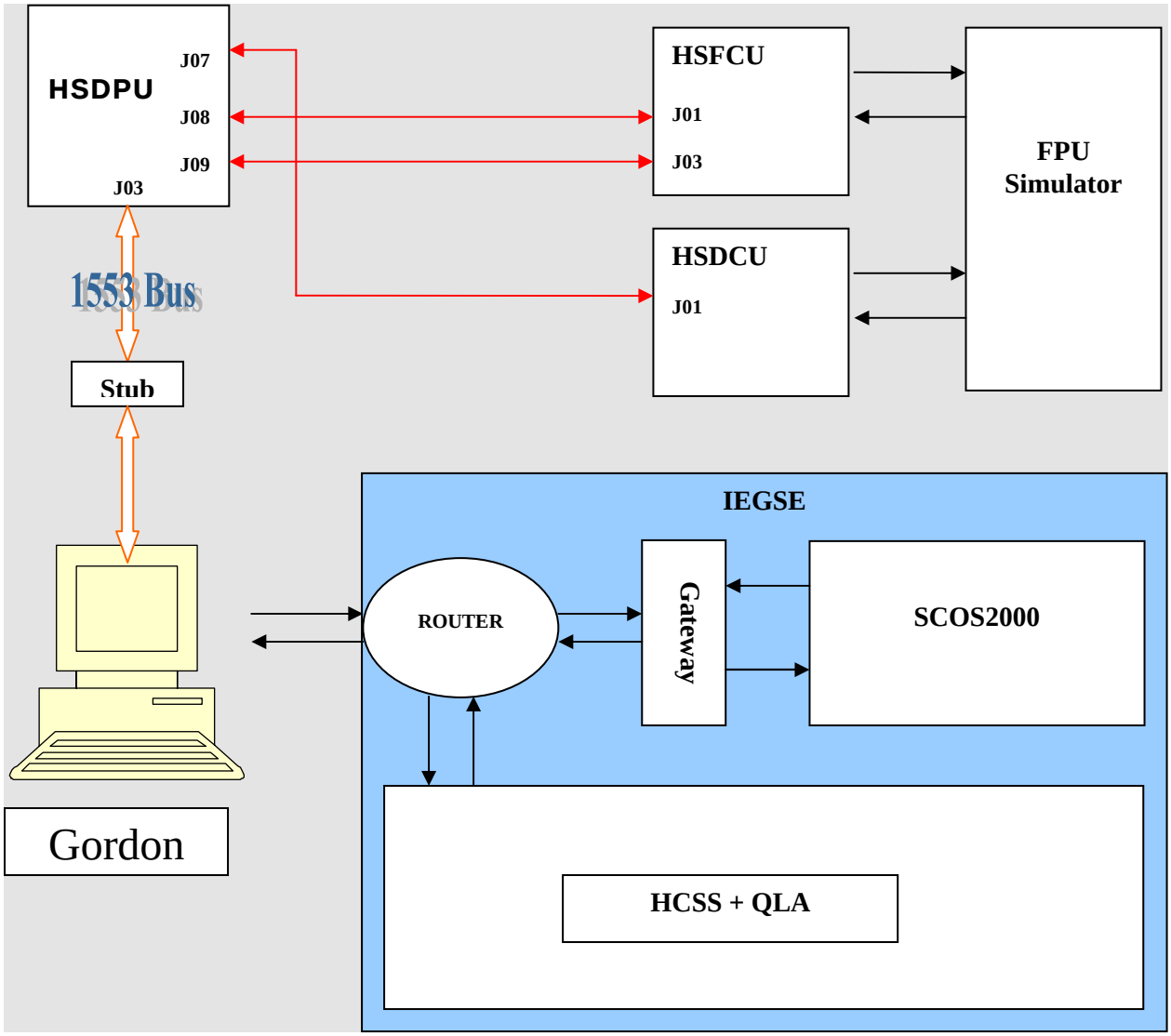

Figure 1 Spire Warm Electronics and IEGSE configuration for SPIRE_BENCH tests

5. ANNEXE1

Test_SPIRE_BENCH01

Test Name	Test Description	HK Key Parameter	Number of frames expected	Actual Value Before/After	Number of frames received	Comments	Test Result Success/Fail
Test_SPIRE_BENCH 01.tcl	Refer to 3.3	**DCUFRAMECNT**	**700**				

Start time @:
End time @:
OBSID:
Commets:

Test_SPIRE_BENCH02

Test Name	Test Description	HK Key Parameter	Number of frames expected	Actual Value Before/After	Number of frames received	Comments	Test Result Success/Fail
Test_SPIRE_BENCH02.tcl	Refer to 3.3	**DCUFRAMECNT**	**700**				

Start time @:
End time @:
OBSID:
Commets:

Test_SPIRE_BENCH03

Test Name	Test Description	HK Key Parameter	Number of frames expected	Actual Value Before/After	Number of frames received	Comments	Test Result Success/Fail
Test_SPIRE_BENCH03.tcl	Refer to 3.3	**DCUFRAMECNT**	**~1400**				

Start time @:
End time @:
OBSID:
Commets:

Test_SPIRE_BENCH04

Test Name	Test Description	HK Key Parameter	Number of frames expected	Actual Value Before/After	Number of frames received	Comments	Test Result Success/Fail
Test_SPIRE_BENCH 04.tcl	Refer to 3.3	**SCUFRAMECNT**	**31**				

Start time @:
End time @:
OBSID:
Commets:

Test_SPIRE_BENCH05

Test Name	Test Description	HK Key Parameter	Number of frames expected	Actual Value Before/After	Number of frames received	Comments	Test Result Success/Fail
Test_SPIRE_BENCH05.tcl	Refer to 3.3	**SCUFRAMECNT**	**31**				

Start time @:
End time @:
OBSID:
Commets:

Test_SPIRE_BENCH06

Test Name	Test Description	HK Key Parameter	Number of frames expected	Actual Value Before/After	Number of frames received	Comments	Test Result Success/Fail
Test_SPIRE_BENCH06.tcl	Refer to 3.3	**DCUFRAMECNT**	**~4000**				

Start time @:
End time @:
OBSID:
Commets:

Test_SPIRE_BENCH07

Test Name	Test Description	HK Key Parameter	Number of frames expected	Actual Value Before/After	Number of frames received	Comments	Test Result Success/Fail
Test_SPIRE_BENCH07.tcl	Refer to 3.3	**DCUFRAMECNT**					

Start time @:
End time @:
OBSID:
Commets:

Test_SPIRE_BENCH08

Test Name	Test Description	HK Key Parameter	Number of frames expected	Actual Value Before/After	Number of frames received	Comments	Test Result Success/Fail
Test_SPIRE_BENCH08.tcl	Refer to 3.3	**DCUFRAMECNT**	**700**				

Start time @:
End time @:
OBSID:
Commets:

Test_SPIRE_BENCH09

Test Name	Test Description	HK Key Parameter	Number of frames expected	Actual Value Before/After	Number of frames received	Comments	Test Result Success/Fail
Test_SPIRE_BENCH09.tcl	Refer to 3.3	**DCUFRAMECNT**					

Start time @:
End time @:
OBSID:
Commets:

6. ANNEXE2

Test control scripts for warm electronics integration tests

```
# ----
# Test_SPIRE_BENCH01
# ----


# @Author      Asier Abreu
# @Date        11/11/04
# @Version     1.0
# @Purpose     DCU Nominal data generation fixed number of frames,test control
# @param photbiasfreq float 200 Enter Phot Bias Frequence [0.15 - 305](Hz)
# @param photsampfreq float 43 Enter Phot Sampling Frequence [38 - 305](Hz)
# @param specbiasfreq float 160 Enter Spec Bias Frequence [0.15 - 305](Hz)
# @param specsampfreq float 80 Enter SpecSampling Frequence [38 - 305](Hz)

catch {unset newcmdList}
catch {unset params}
set phase 0
set ph0 [expr $phase * 255.0 / 360.0]
set ph1 [round $ph0]
set phase [format %04x $ph1]
set frames 100
set ftime 0
set params(photbiasfreq) [expr $photbiasfreq]
set params(photsampfreq) [expr $photsampfreq]
set params(specbiasfreq) [expr $specbiasfreq]
set params(specsampfreq) [expr $specsampfreq]
set params(phase) [expr $phase]
set params(frames) [expr $frames]
set params(ftime) [expr $ftime]
set cmdList [getObservationCommands Test_SPIRE_BENCH01 params]
source /home/sops23e/tcl/TC/CUS2hex.tcl
CUS2hex $cmdList
sendObservationCommands $newcmdList
#foreach n $newcmdList {
# putlog $n
# }
```

```
# ----
# Test_SPIRE_BENCH02
# ----


# @Author      Asier Abreu
# @Date        11/11/04
# @Version     1.0
# @Purpose     DCU Test pattern data generation fixed number of frames,test control
# @param photbiasfreq float 200 Enter Phot Bias Frequence [0.15 - 305](Hz)
# @param photsampfreq float 43 Enter Phot Sampling Frequence [38 - 305](Hz)
# @param specbiasfreq float 160 Enter Spec Bias Frequence [0.15 - 305](Hz)
# @param specsampfreq float 80 Enter SpecSampling Frequence [38 - 305](Hz)

catch {unset newcmdList}
catch {unset params}
set phase 0
set ph0 [expr $phase * 255.0 / 360.0]
set ph1 [round $ph0]
set phase [format %04x $ph1]
set frames 100
set ftime 0
set params(photbiasfreq) [expr $photbiasfreq]
set params(photsampfreq) [expr $photsampfreq]
set params(specbiasfreq) [expr $specbiasfreq]
set params(specsampfreq) [expr $specsampfreq]
set params(phase) [expr $phase]
set params(frames) [expr $frames]
set params(ftime) [expr $ftime]
set cmdList [getObservationCommands Test_SPIRE_BENCH02 params]
source /home/sops23e/tcl/TC/CUS2hex.tcl
CUS2hex $cmdList
sendObservationCommands $newcmdList
#foreach n $newcmdList {
# putlog $n
# }
```

```
# ----
# Test_SPIRE_BENCH03
# ----


# @Author      Asier Abreu
# @Date        11/11/04
# @Version     1.0
# @Purpose     DCU Nominal data generation continuous,test control
# @param photbiasfreq float 200 Enter Phot Bias Frequence [0.15 - 305](Hz)
# @param photsampfreq float 43 Enter Phot Sampling Frequence [38 - 305](Hz)
# @param specbiasfreq float 160 Enter Spec Bias Frequence [0.15 - 305](Hz)
# @param specsampfreq float 80 Enter SpecSampling Frequence [38 - 305](Hz)

catch {unset newcmdList}
catch {unset params}
set phase 0
set ph0 [expr $phase * 255.0 / 360.0]
set ph1 [round $ph0]
set phase [format %04x $ph1]
set frames 0
set ftime 60
set params(photbiasfreq) [expr $photbiasfreq]
set params(photsampfreq) [expr $photsampfreq]
set params(specbiasfreq) [expr $specbiasfreq]
set params(specsampfreq) [expr $specsampfreq]
set params(phase) [expr $phase]
set params(frames) [expr $frames]
set params(ftime) [expr $ftime]
set cmdList [getObservationCommands Test_SPIRE_BENCH03 params]
source /home/sops23e/tcl/TC/CUS2hex.tcl
CUS2hex $cmdList
sendObservationCommands $newcmdList
#foreach n $newcmdList {
# putlog $n
# }
```

```
# ----
# Test_SPIRE_BENCH04
# ----

# @Author      Asier Abreu
# @Date        11/11/04
# @Version     1.0
# @Purpose     SCU Nominal data generation fixed number of frames
# @param scuframes integer 31 Number of SCU frames [0 - 31]
# @param framerate integer 0 SCU framerate f(Hz)=80/(1+framerate)
catch {unset newcmdList}
catch {unset params}
set ftime 0
set params(scuframes) [expr $scuframes]
set params(framerate) [expr $framerate]
set params(ftime) [expr $ftime]
set cmdList [getObservationCommands Test_SPIRE_BENCH04 params]
source /home/sops23e/tcl/TC/CUS2hex.tcl
CUS2hex $cmdList
sendObservationCommands $newcmdList
#foreach n $newcmdList {
# putlog $n
# }

# ----
# Test_SPIRE_BENCH05
# ----

# @Author      Asier Abreu
# @Date        11/11/04
# @Version     1.0
# @Purpose     SCU Test Pattern data generation fixed number of frames
# @param scuframes integer 31 Number of SCU frames [0 - 31]
# @param framerate integer 0 SCU framerate f(Hz)=80/(1+framerate)
catch {unset newcmdList}
catch {unset params}
set ftime 0
set params(scuframes) [expr $scuframes]
set params(framerate) [expr $framerate]
set params(ftime) [expr $ftime]
set cmdList [getObservationCommands Test_SPIRE_BENCH05 params]
source /home/sops23e/tcl/TC/CUS2hex.tcl
CUS2hex $cmdList
sendObservationCommands $newcmdList
#foreach n $newcmdList {
# putlog $n
# }
# ----
# Test_SPIRE_BENCH05
# ----
```

```
# @Author      Asier Abreu
# @Date        11/11/04
# @Version     1.0
# @Purpose     SCU Nominal data generation fixed number of frames
# @param framerate integer 0 SCU framerate f(Hz)=80/(1+framerate)
# @param ftime integer 60 Time for data generation
catch {unset newcmdList}
catch {unset params}
set scuframes 0
set params(scuframes) [expr $scuframes]
set params(framerate) [expr $framerate]
set params(ftime) [expr $ftime]
set cmdList [getObservationCommands Test_SPIRE_BENCH06 params]
source /home/sops23e/ tcl/TC/CUS2hex.tcl
CUS2hex $cmdList
sendObservationCommands $newcmdList
#foreach n $newcmdList {
# putlog $n
# }
```

7. ANNEXE3

Cus scripts for warm electronics integration tests

```
mode Test_SPIRE_BENCH01 {
   string dmode = "PF_Data"; //Full Photometer default
   double photbiasfreq = 200.0; //Default Master Clock divisor for PhotBiasFreq 199.9 Hz
   double photsampfreq = 15.34; //Default Sampling divisor for PhotSamplingFreq 15.38 Hz
   double specbiasfreq = 160.0; //Default Master Clock divisor for SpecBiasFreq  Hz
   double specsampfreq = 80.0; //Default Sampling divisor for SpecSamplingFreq 80 Hz
   int frames = 0x0; //Number of frames to request
   int phase = 0x0; //Phase to 0 degrees
   int ftime = 60; //Time for the generation
}{
   ClearObs();
   delay(1);
   StartObs();
   delay(1);
   string[] modes =
["PF_Data","SF_Data","PSW_Data","PMW_Data","PLW_Data","SSW_Data","SLW_Data"];
   for(int i = 0 .. length(modes) - 1) {

DCU_PhotSpec_QLA_NEW(modes[i],photbiasfreq,photsampfreq,specbiasfreq,specsampfreq,frames,p
hase,ftime);
      delay(1);
   }
   EndObs();
}
```

The CUS script for tests Test_SPIRE_BENCH02 and Test_SPIRE_BENCH03 are exactly the same and is the test control script which determines what type of telemetry is requested via the test procedure parameters.

```
mode Test_SPIRE_BENCH04 {
   int scuframes = 0x1f; //Number of scu frames
   int framerate = 0; // f(hz)=80/(1+framerate)
   int ftime = 1; //Data generation time If conitnuous
}{
   delay(1);
   ClearObs();
   delay(1);
   StartObs();
   delay(1);
   SCU_Nom_Data(framerate,scuframes,ftime);
   delay(1);
   EndObs();
}

mode Test_SPIRE_BENCH05 {
   int scuframes = 0x1f; //Number of scu frames
   int framerate = 0; // f(hz)=80/(1+framerate)
   int ftime = 1; //Data generation time If conitnuous
}{
   delay(1);
   ClearObs();
   delay(1);
   StartObs();
   delay(1);
   SCUTest_Patt(framerate,scuframes,ftime);
   delay(1);
   EndObs();
}
```

The CUS script for tests Test_SPIRE_BENCH06 is exactly the same as Test_SPIRE_BENCH04 and is the test control script which determines what type of telemetry is requested via the test procedure parameters.