SPIRE-AST-PRC-002490

Title: **EGSE Configuration Procedure**

CI-No: 142 210

Prepared by:	S. Ilsen / M. Koelle	Date: 21 June 2005
Checked by:	C. Schlosser	1.7.05
Product Assurance:	for R. Stritter	7.7.05
Configuration Control:	W. Wietbrock	2.7.05
Project Management:	Dr. W. Fricke	07/07/05

Distribution: See Distribution List (last page)

Issue	Date	Sheet	Description of Change	Release
1	26.04.05		Initial Issue	
2	02.06.05		Clarification of TCL scripts with Flow Diagram Change EGSE_CONFIG to EGSE_CONFIG_AUTO Change EGSE_OFFLINE to EGSE_OFFLINE_AUTO Update INSTR_POWER_ON Update INSTR_POWER_OFF Added PACS_POWER_ON and PACS_POWER_OFF	
3	21.06.05		Added Procedure order of execution	

Table of Contents

Table of Figures

No figures

List of Tables

No Tables

1 Abbreviations

PLM SCOE	**P**ayload **M**odule (Power SCOE)
CDMU DFE	**C**ommand & **D**ata **M**anagement **U**nit **D**ata **F**ront **E**nd
PSU	**P**ower **S**upply **U**nit
EGSE	**E**lectrical **G**round **S**upport **E**quipment
GUI	**G**raphical **U**ser **I**nterface
HK	**H**ousekeeping
TM	**T**ele**m**etry
TC	**T**elec**c**ommand
LCL	**L**atching **C**urrent **L**imiter (Power Switch)

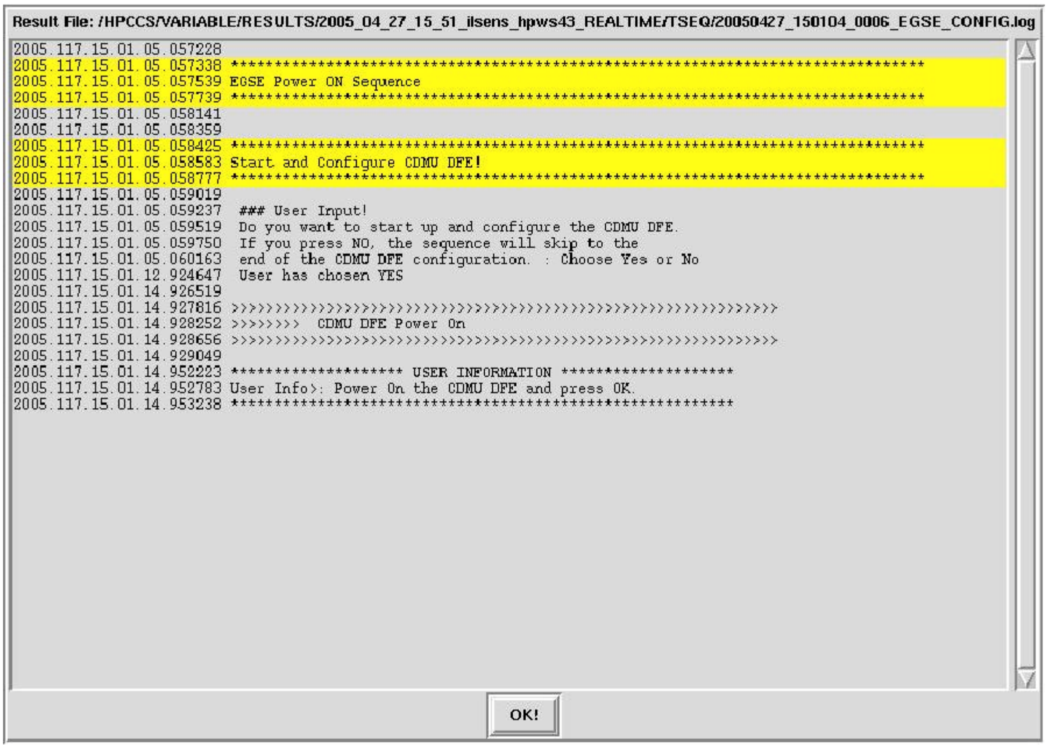
2 General description

This document describes briefly how the PLM EGSE, consisting of the PLM SCOE and CDMU DFE is setup. The procedure is split up into 4 different parts and for each part a specific TCL script is made.

1. EGSE Configuration before test (EGSE_CONFIG_AUTO.tcl)
2. EGSE de-configuration (EGSE_OFFLINE_AUTO.tcl)
3. Instruments Power On (INSTR_POWER_ON.tcl)
4. Instruments Power Off (INSTR_POWER_OFF.tcl)

The tcl scripts, mentioned above, are used to guide the CCS test conductor through the start up and configuration process of the PLM EGSE. The scripts will make sure the PLM EGSE is setup similarly each time a test is started. They will generate a detailed log files containing TM data from the CDMU DFE and PLM SCOE. This data can be used to verify if the PLM EGSE is setup correctly. Apart from this specific log file, also a standard CCS log file is generated.

The tcl sequences are visualised as an ‘interactive log’. This means that the user is asked specific questions, which can be answered with OK, YES or NO, etc... The answer of the user is then immediately displayed in the log and an action is executed. This can be seen in the figure below.



The TCL scripts are using EGSE specific commands and lots of GUI specific code. This may provide great difficulty to read the code. For this purpose, each script is visualised in a flowchart.

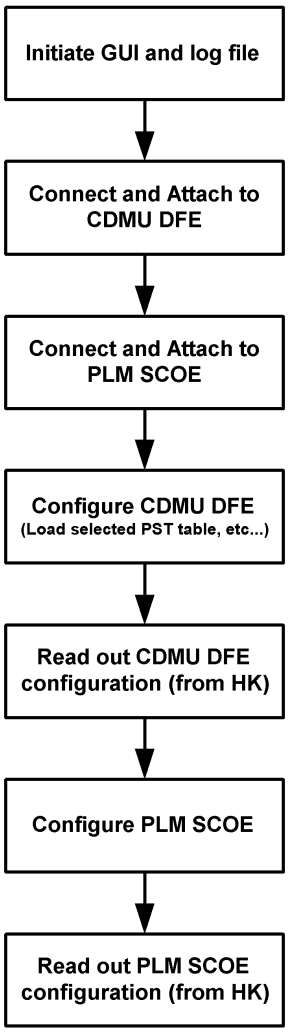
3 Order of Execution

Step-No.	Test-Step-Description	Comments	P	N
1	Make sure the CCS is switched on and a REALTIME session has been started	Test session name:		
2	Power on the CDMU DFE platform			
3	Power on the PLM SCOE platform			
4	Power on the CDMU DFE workstation and wait for the BIST to finish.	If BIST failed restart CDMU DFE platform and workstation		
5	Power on the PLM SCOE workstation and wait for the BIST to finish.	If BIST failed restart PLM SCOE platform and workstation		
6	Execute “EGSE_CONFIG_AUTO.tcl”			
7	Execute “SubscribeParams.tcl”	Wait until TCL status on CCS has changed from RUN to WAIT. This may take up to 10 minutes.		
8	Execute “INSTR_POWER_ON.tcl” or an instrument specific power on sequence.			
9	Perform the instrument test (see instrument procedures)			
10	Execute “INSTR_POWER_OFF.tcl” or an instrument specific power off sequence.			
11	Execute “EGSE_OFFLINE_AUTO.tcl”			
12	The PLM EGSE is now offline and ready to be shut down.			

4 Detailed description of TCL's scripts

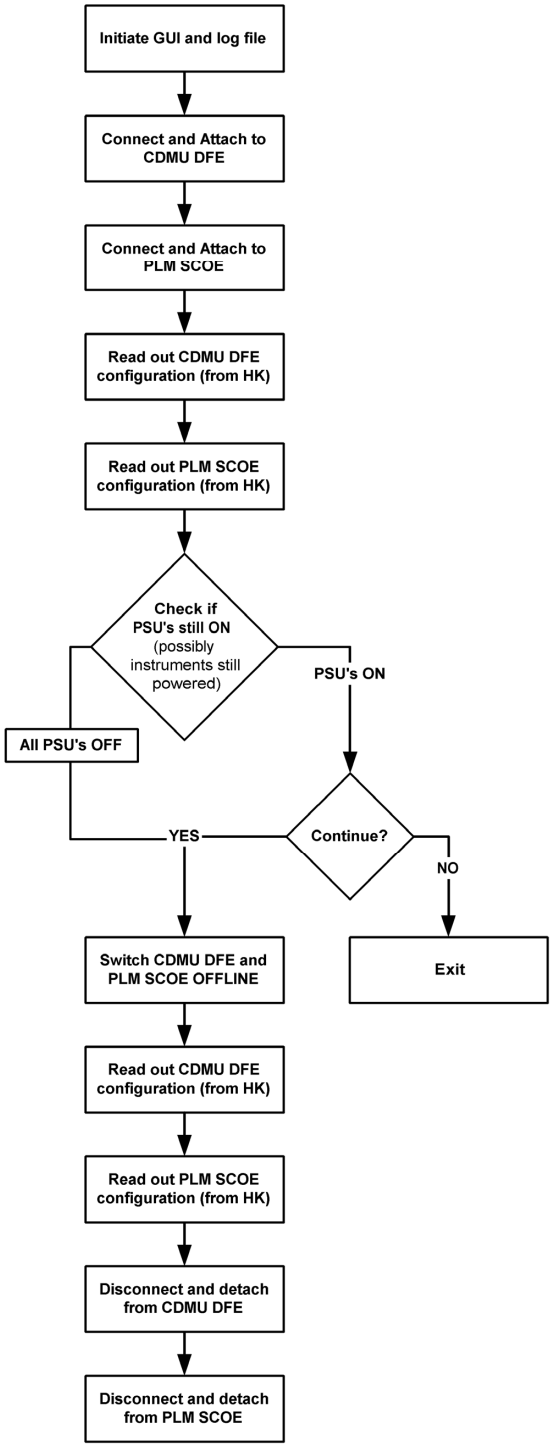
4.1 EGSE_CONFIG_AUTO.tcl (see appendix 1)

This script is used to power on the CDMU DFE and PLM SCOE is the right order. Also the configuration of both modules is included. This means that at the end of the script the CDMU DFE and PLM SCOE are fully configured and in ONLINE, remote commanding mode. A flowchart that visualises the script is shown below:



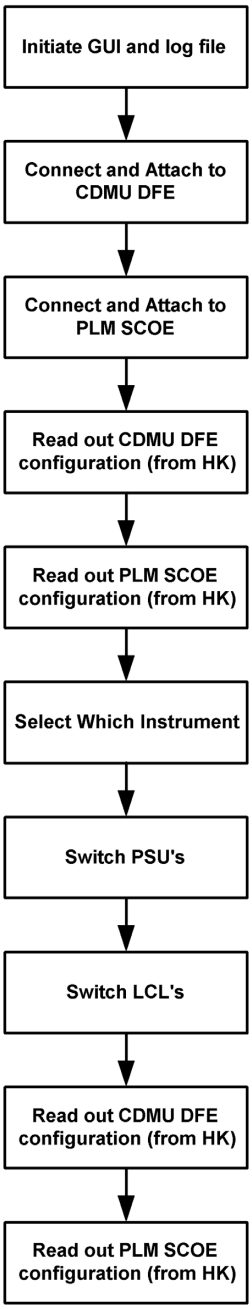
4.2 EGSE_OFFLINE_AUTO.tcl (see appendix 2)

This script will make sure that the CDMU DFE and PLM SCOE are switched to offline mode. This script can be used at the end of a test session and stops any TM coming from the CDMU and switches all LCL's to OFF (if not done already using the INSTR_POWER_OFF sequence). A flowchart that visualises the script is shown below:



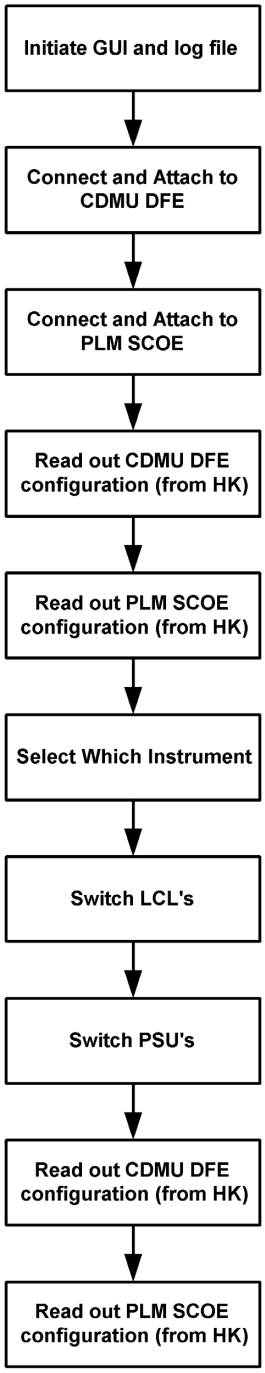
4.3 INSTR_POWER_ON.tcl (see appendix 3)

This script is used to power on a selected instrument(s). The ‘power on’-sequence (order in which LCL are switched ON) is by default set to the information given in the ‘power on’ procedures from the instrumenters. This default order can however be changed before or during the power on. Also the option to use redundant power instead of primary is included. The result of this sequence is a log file containing the voltages and currents of all LCL’c of the PLM SCOE. Also the CDMU configuration is read out and included in the log. A flowchart that visualises the script is shown below:



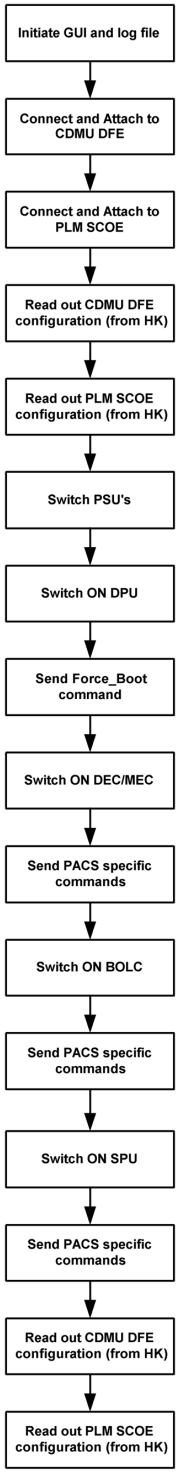
4.4 INSTR_POWER_OFF.tcl (see appendix 4)

This script is used to power off a selected instrument(s). The 'power off'-order is by default set to the order defined in the 'power off'procedures provided by the instrumenters. This order can be however manually changed. A clear log is generated containing all important parameters from PLM SCOE and CDMU DFE. A flowchart that visualises the script is shown below:



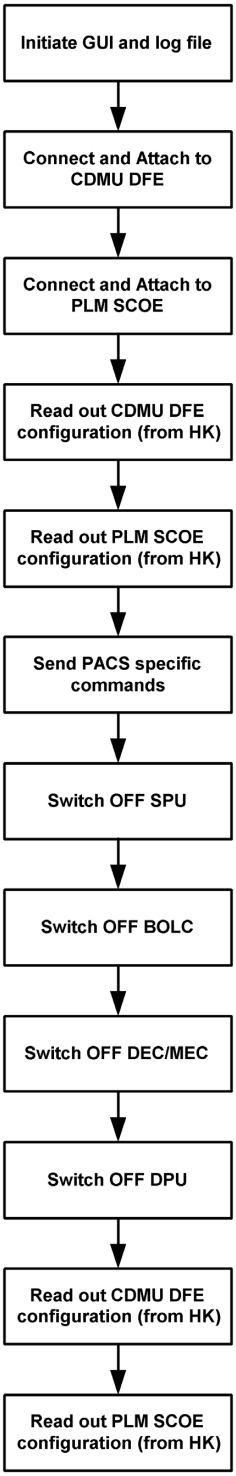
4.5 PACS_POWER_ON.tcl (see appendix 5)

For the PACS instrument a special automated power on sequence was demanded. This script is actually a derived INSTR_POWER_ON.tcl including some PACS commands. These specific PACS commands are extracted from the PACS_Switch_on_CCS.tcl script. A flowchart that visualises the script is shown below:



4.6 PACS_POWER_OFF.tcl (see appendix 6)

Similar to PACS_POWER_ON. The tcl is based on the INSTR_POWER_OFF.tcl script with addition of some PACS specific commands extracted from PACS_Switch_off_CCS.tcl



Appendix 1: EGSE_CONFIG.tcl

```
#######################################################################
# File: $Id: EGSE_CONFIG_AUTO.tcl,v 1.3 2005/06/01 14:39:42 ilsens Exp $
#
# Description:
#
#               Power On EGSE (CDMUDFE & PLM SCOE) using interactive log
#
#
# Last edited by: $Author: ilsens $ on $Date: 2005/06/01 14:39:42 $.
#
#
#######################################################################

# automatically set the revision. do not edit this
setrevision {$Id: EGSE_CONFIG_AUTO.tcl,v 1.3 2005/06/01 14:39:42 ilsens Exp $}

#######################################################################
# start of test sequence
#######################################################################

global env
global go_on
global auto_path

set go_on -1

lappend auto_path "/HPCCS/VARIABLE/CONFIG/TCL/Alenia"

# ------------------------------
# creation of the output file
# ------------------------------
gettime sec usec
set fn_time [clock format $sec -format %Y%m%d_%H%M%S -gmt true]
set fn_name [file tail [info script]]
set fn_name [string trim $fn_name ".tcl"]
set fn_add "GUI_log"
set fn_ID "XXXX"
append FileName $fn_time "_" $fn_ID "_" $fn_name "_" $fn_add ".txt"
putlog $FileName
set outfile [open $env(HPCCSTESTRES)/TSEQ/$FileName w]

# This is the mandatory procedures to abtain the output log window
setup_win

newTest_gen "EGSE CONFIG Sequence"
logm ""

newTest_gen "Check of CDMU DFE and PLM SCOE"
logm ""

logm ""
logm "Connecting to CDMU DFE"
connect CDMUDFE
waittime +00.00.02.000000
logm "Attaching to CMDU DFE"
attach CDMUDFE
waittime +00.00.01.000000
logm ""
#Check if BIST was successful?
logm "Checking if CDMU DFE BIST was OK"
waittime +00.00.02.000000
logm ""
set Status_CDMU_BIST [getrawvalue [fetch YM024944]]
if {$Status_CDMU_BIST == 1} {
        logm ">>> RESULT : CDMU DFE BIST OK, continuing EGSE_CONFIG."
        waittime +00.00.02.000000
} else {
        logm ">>> RESULT : CDMU DFE BIST Not OK, stopping EGSE_CONFIG."
        waittime +00.00.03.000000
        finish_TS
}

logm ""
logm "Connecting to PLM SCOE"
connect PLMSCOE
waittime +00.00.02.000000
logm "Attaching to PLM SCOE"
attach PLMSCOE
waittime +00.00.01.000000
logm ""
#Check if BIST was successful?
logm "Checking if PLM SCOE BIST was OK"
waittime +00.00.02.000000
logm ""
```

```
set Status_PLM_BIST [getrawvalue [fetch YM024942]]
if {$Status_PLM_BIST == 1} {
        logm ">>> RESULT : PLM SCOE BIST OK, continuing EGSE_CONFIG."
        waittime +00.00.02.000000
} else {
        logm ">>> RESULT : PLM SCOE BIST Not OK, stopping EGSE_CONFIG."
        waittime +00.00.03.000000
        finish_TS
}

newTest_gen "Configuring CDMU DFE"
logm ""

logm "Switching CDMUDFE to ONLINE mode"
tcsend YC002944
waittime +00.00.01.000000
logm ""

set result "0"
logm ""
set pst(0) "Unlisted PST file"
set pst(1) "HIFI_prime_inst.PST"
set pst(2) "SPIRE_prime_inst.PST"
set pst(3) "PACS_prime_inst.PST"
set pst(4) "PACS_burst_mode.PST"
set pst(5) "PACS_SPIRE_par.PST"

logm ""
logm "Available PST tables:"
logm "1. $pst(1)"
logm "2. $pst(2)"
logm "3. $pst(3)"
logm "4. $pst(4)"
logm "5. $pst(5)"
logm ""

set checkPST 1
while {$checkPST == 1} {
        set result [inputbox ">>> Please enter the number of the required PST table. Enter 0 for an unlisted."]
        # check if PST is valid
        if {$result == 1 || $result == 2 || $result == 3 || $result == 4 || $result == 5 || $result == 0} {
                set checkPST "0"
                logm ""
                logm "You have selected $result : $pst($result)"
                set pst_name $pst($result)
                logm ""
        } else {
                logm " !!! The chosen number is not valid."
                logm " !!! You can chose between 1,2,3,4,5 or 0."
                logm ""
        }
}

if {$result != 0} {
        logm "Loading $pst_name file on CDMU DFE"
        logm ""
        tcsend YC057944 [list YP571944 $pst_name]
        waittime +00.00.05.000000
        # do a check. This will make sure that a missing PST table is detected"
        set Status_CDMU_PSTfileName [getrawvalue [fetch YM809944]]
        set pst_file [string range $Status_CDMU_PSTfileName 0 12]
        set pst_shouldbe [string range $pst_name 0 12]
        set equal_result [string equal $pst_file $pst_shouldbe]
        if {$equal_result == 1} {
                logm "The PST table is loaded on the CDMU DFE."
        } else {
                logm "The PST table is NOT loaded on the CDMU DFE. Please load it manually."
                set result 0
        }
}
logm ""

if {$result == 0} {
        logm "Switching to Local Commanding Mode"
        tcsend YC004944
        waittime +00.00.01.000000
        logm ""
        set result "0"
        set result [infom ">>> Please load and apply the PST table of your choice and press OK."]
        logm "Switching to Remote Commanding Mode"
        tcsend YC005944
        waittime +00.00.01.000000
        logm ""
}

logm "Enabling PST file execution."
tcsend YC058944 {YP010944 "1"}
waittime +00.00.01.000000
```

```
logm ""
logm "Enabling TM Queue."
tcsend YC068944 {YP010944 "1"}
waittime +00.00.01.000000
logm ""
logm "Enabling TM Polling."
tcsend YC067944 {YP010944 "1"}
waittime +00.00.01.000000
logm ""
logm "Enabling TC Queue."
tcsend YC060944 {YP010944 "1"}
waittime +00.00.01.000000
logm ""
logm "Enabling SA Queue."
tcsend YC064944 {YP010944 "1"}
waittime +00.00.01.000000
logm ""
logm "Enabling SA Reading."
tcsend YC066944 {YP010944 "1"}
waittime +00.00.01.000000
logm ""
logm "Enabling Low Level Time Synchronisation."
tcsend YC05D944 {YP010944 "1"}
waittime +00.00.01.000000
logm ""
set result "0"
set result [infom ">>> Please Enable the Busmonitor (Set Online Mode and Start New Acquisition) and press OK."]

waittime +00.00.05.000000

### Reading out CDMUDFE Settings
logm ">>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>"
logm ">>>>>>>>  Reading out CDMUDFE Settings"
logm ">>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>"
logm ""

set Status_CDMU_OnLine [getrawvalue [fetch YM777944]]
logm "Status_CDMU_OnLine is $Status_CDMU_OnLine (extracted from TLM YM777944)"

set Status_CDMU_TMpolling [getrawvalue [fetch YM780944]]
logm "Status_CDMU_TMpolling is $Status_CDMU_TMpolling (extracted from TLM YM780944)"

set Status_CDMU_SAreadActive [getrawvalue [fetch YM781944]]
logm "Status_CDMU_SAreadActive is $Status_CDMU_SAreadActive (extracted from TLM YM781944)"

set Status_CDMU_SAqueueActive [getrawvalue [fetch YM782944]]
logm "Status_CDMU_SAqueueActive is $Status_CDMU_SAqueueActive (extracted from TLM YM782944)"

set Status_CDMU_TMqueueActive [getrawvalue [fetch YM783944]]
logm "Status_CDMU_TMqueueActive is $Status_CDMU_TMqueueActive (extracted from TLM YM783944)"

set Status_CDMU_TCqueueActive [getrawvalue [fetch YM784944]]
logm "Status_CDMU_TCqueueActive is $Status_CDMU_TCqueueActive (extracted from TLM YM784944)"

set Status_CDMU_PSTfileName [getrawvalue [fetch YM809944]]
logm "Status_CDMU_PSTfileName is $Status_CDMU_PSTfileName (extracted from TLM YM809944)"

set Status_CDMU_PSTrunning [getrawvalue [fetch YM829944]]
logm "Status_CDMU_PSTrunning is $Status_CDMU_PSTrunning (extracted from TLM YM829944)"

waittime +00.00.03.000000

newTest_gen "Configuring PLM SCOE"
logm ""

logm "Switching PLM SCOE to ONLINE mode"
tcsend YC002942
waittime +00.00.01.000000
logm ""
waittime +00.00.05.000000

### Reading out PLM SCOE Settings
logm ">>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>"
logm ">>>>>>>>  Reading out PLM SCOE Settings"
logm ">>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>"
logm ""

set Status_PLM_OnLine [getrawvalue [fetch YM018942]]
logm "Status_PLM_OnLine is $Status_PLM_OnLine (extracted from TLM YM018942)"

set Status_PLM_PSU1_Master [getrawvalue [fetch YM129942]]
logm "Status_PLM_PSU1_Master is currently $Status_PLM_PSU1_Master (extracted from TLM YM129942)"

set Status_PLM_PSU1_Slave [getrawvalue [fetch YM145942]]
logm "Status_PLM_PSU1_Slave is currently $Status_PLM_PSU1_Slave (extracted from TLM YM145942)"

set Status_PLM_PSU2_Master [getrawvalue [fetch YM177942]]
logm "Status_PLM_PSU2_Master is currently $Status_PLM_PSU2_Master (extracted from TLM YM177942)"
```

```
set Status_PLM_PSU2_Slave [getrawvalue [fetch YM193942]]
logm "Status_PLM_PSU2_Slave is currently $Status_PLM_PSU2_Slave (extracted from TLM YM193942)"

set Status_PLM_LCL1_V [getrawvalue [fetch YM228942]]
logm "Status_PLM_LCL1_V is currently $Status_PLM_LCL1_V (extracted from TLM YM228942)"

set Status_PLM_LCL1_I [getrawvalue [fetch YM232942]]
logm "Status_PLM_LCL1_I is currently $Status_PLM_LCL1_I (extracted from TLM YM232942)"

set Status_PLM_LCL2_V [getrawvalue [fetch YM244942]]
logm "Status_PLM_LCL2_V is currently $Status_PLM_LCL2_V (extracted from TLM YM244942)"

set Status_PLM_LCL2_I [getrawvalue [fetch YM248942]]
logm "Status_PLM_LCL2_I is currently $Status_PLM_LCL2_I (extracted from TLM YM248942)"

set Status_PLM_LCL3_V [getrawvalue [fetch YM260942]]
logm "Status_PLM_LCL3_V is currently $Status_PLM_LCL3_V (extracted from TLM YM260942)"

set Status_PLM_LCL3_I [getrawvalue [fetch YM264942]]
logm "Status_PLM_LCL3_I is currently $Status_PLM_LCL3_I (extracted from TLM YM264942)"

set Status_PLM_LCL4_V [getrawvalue [fetch YM276942]]
logm "Status_PLM_LCL4_V is currently $Status_PLM_LCL4_V (extracted from TLM YM276942)"

set Status_PLM_LCL4_I [getrawvalue [fetch YM280942]]
logm "Status_PLM_LCL4_I is currently $Status_PLM_LCL4_I (extracted from TLM YM280942)"

set Status_PLM_LCL5_V [getrawvalue [fetch YM292942]]
logm "Status_PLM_LCL5_V is currently $Status_PLM_LCL5_V (extracted from TLM YM292942)"

set Status_PLM_LCL5_I [getrawvalue [fetch YM296942]]
logm "Status_PLM_LCL5_I is currently $Status_PLM_LCL5_I (extracted from TLM YM296942)"

set Status_PLM_LCL6_V [getrawvalue [fetch YM308942]]
logm "Status_PLM_LCL6_V is currently $Status_PLM_LCL6_V (extracted from TLM YM308942)"

set Status_PLM_LCL6_I [getrawvalue [fetch YM312942]]
logm "Status_PLM_LCL6_I is currently $Status_PLM_LCL6_I (extracted from TLM YM312942)"

set Status_PLM_LCL7_V [getrawvalue [fetch YM324942]]
logm "Status_PLM_LCL7_V is currently $Status_PLM_LCL7_V (extracted from TLM YM324942)"

set Status_PLM_LCL7_I [getrawvalue [fetch YM328942]]
logm "Status_PLM_LCL7_I is currently $Status_PLM_LCL7_I (extracted from TLM YM328942)"

set Status_PLM_LCL8_V [getrawvalue [fetch YM340942]]
logm "Status_PLM_LCL8_V is currently $Status_PLM_LCL8_V (extracted from TLM YM340942)"

set Status_PLM_LCL8_I [getrawvalue [fetch YM344942]]
logm "Status_PLM_LCL8_I is currently $Status_PLM_LCL8_I (extracted from TLM YM344942)"

set Status_PLM_LCL9_V [getrawvalue [fetch YM356942]]
logm "Status_PLM_LCL9_V is currently $Status_PLM_LCL9_V (extracted from TLM YM356942)"

set Status_PLM_LCL9_I [getrawvalue [fetch YM360942]]
logm "Status_PLM_LCL9_I is currently $Status_PLM_LCL9_I (extracted from TLM YM360942)"

set Status_PLM_LCL10_V [getrawvalue [fetch YM372942]]
logm "Status_PLM_LCL10_V is currently $Status_PLM_LCL10_V (extracted from TLM YM372942)"

set Status_PLM_LCL10_I [getrawvalue [fetch YM376942]]
logm "Status_PLM_LCL10_I is currently $Status_PLM_LCL10_I (extracted from TLM YM376942)"

set Status_PLM_LCL11_V [getrawvalue [fetch YM388942]]
logm "Status_PLM_LCL11_V is currently $Status_PLM_LCL11_V (extracted from TLM YM388942)"

set Status_PLM_LCL11_I [getrawvalue [fetch YM392942]]
logm "Status_PLM_LCL11_I is currently $Status_PLM_LCL11_I (extracted from TLM YM392942)"

set Status_PLM_LCL12_V [getrawvalue [fetch YM404942]]
logm "Status_PLM_LCL12_V is currently $Status_PLM_LCL12_V (extracted from TLM YM404942)"

set Status_PLM_LCL12_I [getrawvalue [fetch YM408942]]
logm "Status_PLM_LCL12_I is currently $Status_PLM_LCL12_I (extracted from TLM YM408942)"

set Status_PLM_LCL13_V [getrawvalue [fetch YM420942]]
logm "Status_PLM_LCL13_V is currently $Status_PLM_LCL13_V (extracted from TLM YM420942)"

set Status_PLM_LCL13_I [getrawvalue [fetch YM424942]]
logm "Status_PLM_LCL13_I is currently $Status_PLM_LCL13_I (extracted from TLM YM424942)"

set Status_PLM_LCL14_V [getrawvalue [fetch YM436942]]
logm "Status_PLM_LCL14_V is currently $Status_PLM_LCL14_V (extracted from TLM YM436942)"

set Status_PLM_LCL14_I [getrawvalue [fetch YM440942]]
logm "Status_PLM_LCL14_I is currently $Status_PLM_LCL14_I (extracted from TLM YM440942)"
```

```
#End

finish_TS


######################################################################
# end of test sequence
######################################################################
# Changes:
# $Log: EGSE_CONFIG_AUTO.tcl,v $
# Revision 1.3  2005/06/01 14:39:42  ilsens
# (No log message)
#
# Revision 1.2  2005/06/01 12:41:42  ilsens
# (No log message)
#
# Revision 1.1  2005/06/01 11:51:36  ilsens
# initial version
#
#
######################################################################
```

Appendix 2: EGSE_OFFLINE.tcl

```
#####################################################################
# File: $Id: EGSE_OFFLINE_AUTO.tcl,v 1.2 2005/06/01 12:41:47 ilsens Exp $
#
# Description:
#
#               Swithc the EGSE (CDMU DFE and PLM SCOE) offline using interactive log
#
#
# Last edited by: $Author: ilsens $ on $Date: 2005/06/01 12:41:47 $.
#
#
#####################################################################

# automatically set the revision. do not edit this
setrevision {$Id: EGSE_OFFLINE_AUTO.tcl,v 1.2 2005/06/01 12:41:47 ilsens Exp $}

#####################################################################
# start of test sequence
#####################################################################

global env
global go_on
global auto_path

set go_on -1

lappend auto_path "/HPCCS/VARIABLE/CONFIG/TCL/Alenia"

# ------------------------------
# creation of the output file
# ------------------------------
gettime sec usec
set fn_time [clock format $sec -format %Y%m%d_%H%M%S -gmt true]
set fn_name [file tail [info script]]
set fn_name [string trim $fn_name ".tcl"]
set fn_add "GUI_log"
set fn_ID "XXXX"
append FileName $fn_time "_" $fn_ID "_" $fn_name "_" $fn_add ".txt"
putlog $FileName
set outfile [open $env(HPCCSTESTRES)/TSEQ/$FileName w]

# This is the mandatory procedures to abtain the output log window
setup_win

newTest_gen "EGSE OFFLINE Sequence"
logm ""

newTest_gen "Connect and attach to CDMU DFE and PLM SCOE"
logm ""

logm ""
logm "Connecting to CDMU DFE"
connect CDMUDFE
waittime +00.00.02.000000
logm "Attaching to CMDU DFE"
attach CDMUDFE
waittime +00.00.01.000000
logm ""

logm "Connecting to PLM SCOE"
connect PLMSCOE
waittime +00.00.02.000000
logm "Attaching to PLM SCOE"
attach PLMSCOE
waittime +00.00.01.000000
logm ""

logm ""
logm ">>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>"
logm ">>>>>>>>  Reading out CDMUDFE Settings"
logm ">>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>"
logm ""

set Status_CDMU_OnLine [getrawvalue [fetch YM777944]]
logm "Status_CDMU_OnLine is $Status_CDMU_OnLine (extracted from TLM YM777944)"

set Status_CDMU_TMpolling [getrawvalue [fetch YM780944]]
logm "Status_CDMU_TMpolling is $Status_CDMU_TMpolling (extracted from TLM YM780944)"

set Status_CDMU_SAreadActive [getrawvalue [fetch YM781944]]
logm "Status_CDMU_SAreadActive is $Status_CDMU_SAreadActive (extracted from TLM YM781944)"

set Status_CDMU_SAqueueActive [getrawvalue [fetch YM782944]]
logm "Status_CDMU_SAqueueActive is $Status_CDMU_SAqueueActive (extracted from TLM YM782944)"

set Status_CDMU_TMqueueActive [getrawvalue [fetch YM783944]]
```

```
logm "Status_CDMU_TMqueueActive is $Status_CDMU_TMqueueActive (extracted from TLM YM783944)"

set Status_CDMU_TCqueueActive [getrawvalue [fetch YM784944]]
logm "Status_CDMU_TCqueueActive is $Status_CDMU_TCqueueActive (extracted from TLM YM784944)"

set Status_CDMU_PSTfileName [getrawvalue [fetch YM809944]]
logm "Status_CDMU_PSTfileName is $Status_CDMU_PSTfileName (extracted from TLM YM809944)"

set Status_CDMU_PSTrunning [getrawvalue [fetch YM829944]]
logm "Status_CDMU_PSTrunning is $Status_CDMU_PSTrunning (extracted from TLM YM829944)"

logm ""
logm ">>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>"
logm ">>>>>>>>  Reading out PLM SCOE Settings"
logm ">>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>"
logm ""

set Status_PLM_OnLine [getrawvalue [fetch YM018942]]
logm "Status_PLM_OnLine is $Status_PLM_OnLine (extracted from TLM YM018942)"

set Status_PLM_PSU1_Master [getrawvalue [fetch YM129942]]
logm "Status_PLM_PSU1_Master is currently $Status_PLM_PSU1_Master (extracted from TLM YM129942)"

set Status_PLM_PSU1_Slave [getrawvalue [fetch YM145942]]
logm "Status_PLM_PSU1_Slave is currently $Status_PLM_PSU1_Slave (extracted from TLM YM145942)"

set Status_PLM_PSU2_Master [getrawvalue [fetch YM177942]]
logm "Status_PLM_PSU2_Master is currently $Status_PLM_PSU2_Master (extracted from TLM YM177942)"

set Status_PLM_PSU2_Slave [getrawvalue [fetch YM193942]]
logm "Status_PLM_PSU2_Slave is currently $Status_PLM_PSU2_Slave (extracted from TLM YM193942)"

set Status_PLM_LCL1_V [getrawvalue [fetch YM228942]]
logm "Status_PLM_LCL1_V is currently $Status_PLM_LCL1_V (extracted from TLM YM228942)"

set Status_PLM_LCL1_I [getrawvalue [fetch YM232942]]
logm "Status_PLM_LCL1_I is currently $Status_PLM_LCL1_I (extracted from TLM YM232942)"

set Status_PLM_LCL2_V [getrawvalue [fetch YM244942]]
logm "Status_PLM_LCL2_V is currently $Status_PLM_LCL2_V (extracted from TLM YM244942)"

set Status_PLM_LCL2_I [getrawvalue [fetch YM248942]]
logm "Status_PLM_LCL2_I is currently $Status_PLM_LCL2_I (extracted from TLM YM248942)"

set Status_PLM_LCL3_V [getrawvalue [fetch YM260942]]
logm "Status_PLM_LCL3_V is currently $Status_PLM_LCL3_V (extracted from TLM YM260942)"

set Status_PLM_LCL3_I [getrawvalue [fetch YM264942]]
logm "Status_PLM_LCL3_I is currently $Status_PLM_LCL3_I (extracted from TLM YM264942)"

set Status_PLM_LCL4_V [getrawvalue [fetch YM276942]]
logm "Status_PLM_LCL4_V is currently $Status_PLM_LCL4_V (extracted from TLM YM276942)"

set Status_PLM_LCL4_I [getrawvalue [fetch YM280942]]
logm "Status_PLM_LCL4_I is currently $Status_PLM_LCL4_I (extracted from TLM YM280942)"

set Status_PLM_LCL5_V [getrawvalue [fetch YM292942]]
logm "Status_PLM_LCL5_V is currently $Status_PLM_LCL5_V (extracted from TLM YM292942)"

set Status_PLM_LCL5_I [getrawvalue [fetch YM296942]]
logm "Status_PLM_LCL5_I is currently $Status_PLM_LCL5_I (extracted from TLM YM296942)"

set Status_PLM_LCL6_V [getrawvalue [fetch YM308942]]
logm "Status_PLM_LCL6_V is currently $Status_PLM_LCL6_V (extracted from TLM YM308942)"

set Status_PLM_LCL6_I [getrawvalue [fetch YM312942]]
logm "Status_PLM_LCL6_I is currently $Status_PLM_LCL6_I (extracted from TLM YM312942)"

set Status_PLM_LCL7_V [getrawvalue [fetch YM324942]]
logm "Status_PLM_LCL7_V is currently $Status_PLM_LCL7_V (extracted from TLM YM324942)"

set Status_PLM_LCL7_I [getrawvalue [fetch YM328942]]
logm "Status_PLM_LCL7_I is currently $Status_PLM_LCL7_I (extracted from TLM YM328942)"

set Status_PLM_LCL8_V [getrawvalue [fetch YM340942]]
logm "Status_PLM_LCL8_V is currently $Status_PLM_LCL8_V (extracted from TLM YM340942)"

set Status_PLM_LCL8_I [getrawvalue [fetch YM344942]]
logm "Status_PLM_LCL8_I is currently $Status_PLM_LCL8_I (extracted from TLM YM344942)"

set Status_PLM_LCL9_V [getrawvalue [fetch YM356942]]
logm "Status_PLM_LCL9_V is currently $Status_PLM_LCL9_V (extracted from TLM YM356942)"

set Status_PLM_LCL9_I [getrawvalue [fetch YM360942]]
logm "Status_PLM_LCL9_I is currently $Status_PLM_LCL9_I (extracted from TLM YM360942)"

set Status_PLM_LCL10_V [getrawvalue [fetch YM372942]]
logm "Status_PLM_LCL10_V is currently $Status_PLM_LCL10_V (extracted from TLM YM372942)"
```

```
set Status_PLM_LCL10_I [getrawvalue [fetch YM376942]]
logm "Status_PLM_LCL10_I is currently $Status_PLM_LCL10_I (extracted from TLM YM376942)"

set Status_PLM_LCL11_V [getrawvalue [fetch YM388942]]
logm "Status_PLM_LCL11_V is currently $Status_PLM_LCL11_V (extracted from TLM YM388942)"

set Status_PLM_LCL11_I [getrawvalue [fetch YM392942]]
logm "Status_PLM_LCL11_I is currently $Status_PLM_LCL11_I (extracted from TLM YM392942)"

set Status_PLM_LCL12_V [getrawvalue [fetch YM404942]]
logm "Status_PLM_LCL12_V is currently $Status_PLM_LCL12_V (extracted from TLM YM404942)"

set Status_PLM_LCL12_I [getrawvalue [fetch YM408942]]
logm "Status_PLM_LCL12_I is currently $Status_PLM_LCL12_I (extracted from TLM YM408942)"

set Status_PLM_LCL13_V [getrawvalue [fetch YM420942]]
logm "Status_PLM_LCL13_V is currently $Status_PLM_LCL13_V (extracted from TLM YM420942)"

set Status_PLM_LCL13_I [getrawvalue [fetch YM424942]]
logm "Status_PLM_LCL13_I is currently $Status_PLM_LCL13_I (extracted from TLM YM424942)"

set Status_PLM_LCL14_V [getrawvalue [fetch YM436942]]
logm "Status_PLM_LCL14_V is currently $Status_PLM_LCL14_V (extracted from TLM YM436942)"

set Status_PLM_LCL14_I [getrawvalue [fetch YM440942]]
logm "Status_PLM_LCL14_I is currently $Status_PLM_LCL14_I (extracted from TLM YM440942)"


newTest_gen "Switch Off PLM SCOE"
logm ""

set Status_PLM_PSU1_Master [getrawvalue [fetch YM129942]]
set Status_PLM_PSU1_Slave [getrawvalue [fetch YM145942]]
set Status_PLM_PSU2_Master [getrawvalue [fetch YM177942]]
set Status_PLM_PSU2_Slave [getrawvalue [fetch YM193942]]

logm "Checking current PLM SCOE status"
waittime +00.00.02.000000
logm ""
if {$Status_PLM_PSU1_Master == 1 || $Status_PLM_PSU2_Master == 1} {
		logm ""
		set result "0"
		set result [yesorno ">>> One (or both) PSU's is still powered. Are you sure to power down the PLM SCOE?"]
		logm ""
		if {$result != 5} {
				logm "Stopping EGSE_OFFLINE sequence"
				waittime +00.00.02.000000
				finish_TS
		}
}
logm ""
logm "Switching PLM SCOE to OFFLINE mode."
tcsend YC003942
waittime +00.00.03.000000

newTest_gen "Switch Off PLM SCOE"
waittime +00.00.01.000000

logm ""
logm "Switching CDMU DFE to OFFLINE mode."
tcsend YC003944
waittime +00.00.03.000000

logm ""
logm ">>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>"
logm ">>>>>>>>  Reading out CDMUDFE Settings"
logm ">>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>"
logm ""

set Status_CDMU_OnLine [getrawvalue [fetch YM777944]]
logm "Status_CDMU_OnLine is $Status_CDMU_OnLine (extracted from TLM YM777944)"

set Status_CDMU_TMpolling [getrawvalue [fetch YM780944]]
logm "Status_CDMU_TMpolling is $Status_CDMU_TMpolling (extracted from TLM YM780944)"

set Status_CDMU_SAreadActive [getrawvalue [fetch YM781944]]
logm "Status_CDMU_SAreadActive is $Status_CDMU_SAreadActive (extracted from TLM YM781944)"

set Status_CDMU_SAqueueActive [getrawvalue [fetch YM782944]]
logm "Status_CDMU_SAqueueActive is $Status_CDMU_SAqueueActive (extracted from TLM YM782944)"

set Status_CDMU_TMqueueActive [getrawvalue [fetch YM783944]]
logm "Status_CDMU_TMqueueActive is $Status_CDMU_TMqueueActive (extracted from TLM YM783944)"

set Status_CDMU_TCqueueActive [getrawvalue [fetch YM784944]]
logm "Status_CDMU_TCqueueActive is $Status_CDMU_TCqueueActive (extracted from TLM YM784944)"
```

```
set Status_CDMU_PSTfileName [getrawvalue [fetch YM809944]]
logm "Status_CDMU_PSTfileName is $Status_CDMU_PSTfileName (extracted from TLM YM809944)"

set Status_CDMU_PSTrunning [getrawvalue [fetch YM829944]]
logm "Status_CDMU_PSTrunning is $Status_CDMU_PSTrunning (extracted from TLM YM829944)"

logm ""
logm ">>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>"
logm ">>>>>>>>  Reading out PLM SCOE Settings"
logm ">>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>"
logm ""

set Status_PLM_OnLine [getrawvalue [fetch YM018942]]
logm "Status_PLM_OnLine is $Status_PLM_OnLine (extracted from TLM YM018942)"

set Status_PLM_PSU1_Master [getrawvalue [fetch YM129942]]
logm "Status_PLM_PSU1_Master is currently $Status_PLM_PSU1_Master (extracted from TLM YM129942)"

set Status_PLM_PSU1_Slave [getrawvalue [fetch YM145942]]
logm "Status_PLM_PSU1_Slave is currently $Status_PLM_PSU1_Slave (extracted from TLM YM145942)"

set Status_PLM_PSU2_Master [getrawvalue [fetch YM177942]]
logm "Status_PLM_PSU2_Master is currently $Status_PLM_PSU2_Master (extracted from TLM YM177942)"

set Status_PLM_PSU2_Slave [getrawvalue [fetch YM193942]]
logm "Status_PLM_PSU2_Slave is currently $Status_PLM_PSU2_Slave (extracted from TLM YM193942)"

set Status_PLM_LCL1_V [getrawvalue [fetch YM228942]]
logm "Status_PLM_LCL1_V is currently $Status_PLM_LCL1_V (extracted from TLM YM228942)"

set Status_PLM_LCL1_I [getrawvalue [fetch YM232942]]
logm "Status_PLM_LCL1_I is currently $Status_PLM_LCL1_I (extracted from TLM YM232942)"

set Status_PLM_LCL2_V [getrawvalue [fetch YM244942]]
logm "Status_PLM_LCL2_V is currently $Status_PLM_LCL2_V (extracted from TLM YM244942)"

set Status_PLM_LCL2_I [getrawvalue [fetch YM248942]]
logm "Status_PLM_LCL2_I is currently $Status_PLM_LCL2_I (extracted from TLM YM248942)"

set Status_PLM_LCL3_V [getrawvalue [fetch YM260942]]
logm "Status_PLM_LCL3_V is currently $Status_PLM_LCL3_V (extracted from TLM YM260942)"

set Status_PLM_LCL3_I [getrawvalue [fetch YM264942]]
logm "Status_PLM_LCL3_I is currently $Status_PLM_LCL3_I (extracted from TLM YM264942)"

set Status_PLM_LCL4_V [getrawvalue [fetch YM276942]]
logm "Status_PLM_LCL4_V is currently $Status_PLM_LCL4_V (extracted from TLM YM276942)"

set Status_PLM_LCL4_I [getrawvalue [fetch YM280942]]
logm "Status_PLM_LCL4_I is currently $Status_PLM_LCL4_I (extracted from TLM YM280942)"

set Status_PLM_LCL5_V [getrawvalue [fetch YM292942]]
logm "Status_PLM_LCL5_V is currently $Status_PLM_LCL5_V (extracted from TLM YM292942)"

set Status_PLM_LCL5_I [getrawvalue [fetch YM296942]]
logm "Status_PLM_LCL5_I is currently $Status_PLM_LCL5_I (extracted from TLM YM296942)"

set Status_PLM_LCL6_V [getrawvalue [fetch YM308942]]
logm "Status_PLM_LCL6_V is currently $Status_PLM_LCL6_V (extracted from TLM YM308942)"

set Status_PLM_LCL6_I [getrawvalue [fetch YM312942]]
logm "Status_PLM_LCL6_I is currently $Status_PLM_LCL6_I (extracted from TLM YM312942)"

set Status_PLM_LCL7_V [getrawvalue [fetch YM324942]]
logm "Status_PLM_LCL7_V is currently $Status_PLM_LCL7_V (extracted from TLM YM324942)"

set Status_PLM_LCL7_I [getrawvalue [fetch YM328942]]
logm "Status_PLM_LCL7_I is currently $Status_PLM_LCL7_I (extracted from TLM YM328942)"

set Status_PLM_LCL8_V [getrawvalue [fetch YM340942]]
logm "Status_PLM_LCL8_V is currently $Status_PLM_LCL8_V (extracted from TLM YM340942)"

set Status_PLM_LCL8_I [getrawvalue [fetch YM344942]]
logm "Status_PLM_LCL8_I is currently $Status_PLM_LCL8_I (extracted from TLM YM344942)"

set Status_PLM_LCL9_V [getrawvalue [fetch YM356942]]
logm "Status_PLM_LCL9_V is currently $Status_PLM_LCL9_V (extracted from TLM YM356942)"

set Status_PLM_LCL9_I [getrawvalue [fetch YM360942]]
logm "Status_PLM_LCL9_I is currently $Status_PLM_LCL9_I (extracted from TLM YM360942)"

set Status_PLM_LCL10_V [getrawvalue [fetch YM372942]]
logm "Status_PLM_LCL10_V is currently $Status_PLM_LCL10_V (extracted from TLM YM372942)"

set Status_PLM_LCL10_I [getrawvalue [fetch YM376942]]
logm "Status_PLM_LCL10_I is currently $Status_PLM_LCL10_I (extracted from TLM YM376942)"

set Status_PLM_LCL11_V [getrawvalue [fetch YM388942]]
```

```
logm "Status_PLM_LCL11_V is currently $Status_PLM_LCL11_V (extracted from TLM YM388942)"

set Status_PLM_LCL11_I [getrawvalue [fetch YM392942]]
logm "Status_PLM_LCL11_I is currently $Status_PLM_LCL11_I (extracted from TLM YM392942)"

set Status_PLM_LCL12_V [getrawvalue [fetch YM404942]]
logm "Status_PLM_LCL12_V is currently $Status_PLM_LCL12_V (extracted from TLM YM404942)"

set Status_PLM_LCL12_I [getrawvalue [fetch YM408942]]
logm "Status_PLM_LCL12_I is currently $Status_PLM_LCL12_I (extracted from TLM YM408942)"

set Status_PLM_LCL13_V [getrawvalue [fetch YM420942]]
logm "Status_PLM_LCL13_V is currently $Status_PLM_LCL13_V (extracted from TLM YM420942)"

set Status_PLM_LCL13_I [getrawvalue [fetch YM424942]]
logm "Status_PLM_LCL13_I is currently $Status_PLM_LCL13_I (extracted from TLM YM424942)"

set Status_PLM_LCL14_V [getrawvalue [fetch YM436942]]
logm "Status_PLM_LCL14_V is currently $Status_PLM_LCL14_V (extracted from TLM YM436942)"

set Status_PLM_LCL14_I [getrawvalue [fetch YM440942]]
logm "Status_PLM_LCL14_I is currently $Status_PLM_LCL14_I (extracted from TLM YM440942)"

#End

newTest_gen "Disconnect and detach from CDMU DFE and PLM SCOE"
logm ""

logm ""
logm "Disconnecting from CDMU DFE"
disconnect CDMUDFE
waittime +00.00.02.000000
logm "Detaching from CMDU DFE"
detach CDMUDFE
waittime +00.00.01.000000
logm ""

logm "Disconnecting from PLM SCOE"
disconnect PLMSCOE
waittime +00.00.02.000000
logm "Detaching from PLM SCOE"
detach PLMSCOE
waittime +00.00.01.000000
logm ""

finish_TS

######################################################################
# end of test sequence
######################################################################
# Changes:
# $Log: EGSE_OFFLINE_AUTO.tcl,v $
# Revision 1.2  2005/06/01 12:41:47  ilsens
# (No log message)
#
# Revision 1.1  2005/06/01 12:14:24  ilsens
# initial version
#
#
######################################################################
```

Appendix 3: INSTR_POWER_ON.tcl

```
#################################################################
# File: $Id: INSTR_POWER_ON.tcl,v 1.6 2005/06/02 10:05:03 ilsens Exp $
#
# Description:
#
#               Power ON the HERSCHEL instruments using an interactive log
#
#
# Last edited by: $Author: ilsens $ on $Date: 2005/06/02 10:05:03 $.
#
#
#################################################################

# automatically set the revision. do not edit this
setrevision {$Id: INSTR_POWER_ON.tcl,v 1.6 2005/06/02 10:05:03 ilsens Exp $}

#################################################################
# start of test sequence
#################################################################


global env
global go_on
global auto_path

set go_on -1

lappend auto_path "/HPCCS/VARIABLE/CONFIG/TCL/Alenia"

# ------------------------------
# creation of the output file
# ------------------------------
gettime sec usec
set fn_time [clock format $sec -format %Y%m%d_%H%M%S -gmt true]
set fn_name [file tail [info script]]
set fn_name [string trim $fn_name ".tcl"]
set fn_add "GUI_log"
set fn_ID "XXXX"
append FileName $fn_time "_" $fn_ID "_" $fn_name "_" $fn_add ".txt"
putlog $FileName
set outfile [open $env(HPCCSTESTRES)/TSEQ/$FileName w]

# This is the mandatory procedures to abtain the output log window
setup_win

logm ""
newTest_gen "Start of Instrument POWER ON sequence."
logm ""

logm " To run this script, the CDMU DFE and PLM SCOE should be"
logm " powered and configured. "
logm " To initiate, this script will connect and attach to the CDMUDFE"
logm " and PLM SCOE."
logm ""

logm "Connecting to CDMU DFE"
connect CDMUDFE
waittime +00.00.02.000000
logm "Attaching to CMDU DFE"
attach CDMUDFE
waittime +00.00.01.000000
logm ""

logm "Connecting to PLM SCOE"
connect PLMSCOE
waittime +00.00.02.000000
logm "Attaching to PLM SCOE"
attach PLMSCOE
waittime +00.00.01.000000

### Reading out CDMUDFE Settings
logm ">>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>"
logm ">>>>>>>>  Reading out CDMUDFE Settings"
logm ">>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>"
logm ""

set Status_CDMU_OnLine [getrawvalue [fetch YM777944]]
logm "Status_CDMU_OnLine is $Status_CDMU_OnLine (extracted from TLM YM777944)"

set Status_CDMU_TMpolling [getrawvalue [fetch YM780944]]
logm "Status_CDMU_TMpolling is $Status_CDMU_TMpolling (extracted from TLM YM780944)"

set Status_CDMU_SAreadActive [getrawvalue [fetch YM781944]]
logm "Status_CDMU_SAreadActive is $Status_CDMU_SAreadActive (extracted from TLM YM781944)"
```

```
set Status_CDMU_SAqueueActive [getrawvalue [fetch YM782944]]
logm "Status_CDMU_SAqueueActive is $Status_CDMU_SAqueueActive (extracted from TLM YM782944)"

set Status_CDMU_TMqueueActive [getrawvalue [fetch YM783944]]
logm "Status_CDMU_TMqueueActive is $Status_CDMU_TMqueueActive (extracted from TLM YM783944)"

set Status_CDMU_TCqueueActive [getrawvalue [fetch YM784944]]
logm "Status_CDMU_TCqueueActive is $Status_CDMU_TCqueueActive (extracted from TLM YM784944)"

set Status_CDMU_PSTfileName [getrawvalue [fetch YM809944]]
logm "Status_CDMU_PSTfileName is $Status_CDMU_PSTfileName (extracted from TLM YM809944)"

set Status_CDMU_PSTrunning [getrawvalue [fetch YM829944]]
logm "Status_CDMU_PSTrunning is $Status_CDMU_PSTrunning (extracted from TLM YM829944)"
logm ""

### Reading out PLM SCOE Settings
logm ">>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>"
logm ">>>>>>>>  Reading out PLM SCOE Settings"
logm ">>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>"
logm ""

set Status_PLM_OnLine [getrawvalue [fetch YM018942]]
logm "Status_PLM_OnLine is $Status_PLM_OnLine (extracted from TLM YM018942)"

set Status_PLM_PSU1_Master [getrawvalue [fetch YM129942]]
logm "Status_PLM_PSU1_Master is currently $Status_PLM_PSU1_Master (extracted from TLM YM129942)"

set Status_PLM_PSU1_Slave [getrawvalue [fetch YM145942]]
logm "Status_PLM_PSU1_Slave is currently $Status_PLM_PSU1_Slave (extracted from TLM YM145942)"

set Status_PLM_PSU2_Master [getrawvalue [fetch YM177942]]
logm "Status_PLM_PSU2_Master is currently $Status_PLM_PSU2_Master (extracted from TLM YM177942)"

set Status_PLM_PSU2_Slave [getrawvalue [fetch YM193942]]
logm "Status_PLM_PSU2_Slave is currently $Status_PLM_PSU2_Slave (extracted from TLM YM193942)"

set Status_PLM_LCL1_V [getrawvalue [fetch YM228942]]
logm "Status_PLM_LCL1_V is currently $Status_PLM_LCL1_V (extracted from TLM YM228942)"

set Status_PLM_LCL1_I [getrawvalue [fetch YM232942]]
logm "Status_PLM_LCL1_I is currently $Status_PLM_LCL1_I (extracted from TLM YM232942)"

set Status_PLM_LCL2_V [getrawvalue [fetch YM244942]]
logm "Status_PLM_LCL2_V is currently $Status_PLM_LCL2_V (extracted from TLM YM244942)"

set Status_PLM_LCL2_I [getrawvalue [fetch YM248942]]
logm "Status_PLM_LCL2_I is currently $Status_PLM_LCL2_I (extracted from TLM YM248942)"

set Status_PLM_LCL3_V [getrawvalue [fetch YM260942]]
logm "Status_PLM_LCL3_V is currently $Status_PLM_LCL3_V (extracted from TLM YM260942)"

set Status_PLM_LCL3_I [getrawvalue [fetch YM264942]]
logm "Status_PLM_LCL3_I is currently $Status_PLM_LCL3_I (extracted from TLM YM264942)"

set Status_PLM_LCL4_V [getrawvalue [fetch YM276942]]
logm "Status_PLM_LCL4_V is currently $Status_PLM_LCL4_V (extracted from TLM YM276942)"

set Status_PLM_LCL4_I [getrawvalue [fetch YM280942]]
logm "Status_PLM_LCL4_I is currently $Status_PLM_LCL4_I (extracted from TLM YM280942)"

set Status_PLM_LCL5_V [getrawvalue [fetch YM292942]]
logm "Status_PLM_LCL5_V is currently $Status_PLM_LCL5_V (extracted from TLM YM292942)"

set Status_PLM_LCL5_I [getrawvalue [fetch YM296942]]
logm "Status_PLM_LCL5_I is currently $Status_PLM_LCL5_I (extracted from TLM YM296942)"

set Status_PLM_LCL6_V [getrawvalue [fetch YM308942]]
logm "Status_PLM_LCL6_V is currently $Status_PLM_LCL6_V (extracted from TLM YM308942)"

set Status_PLM_LCL6_I [getrawvalue [fetch YM312942]]
logm "Status_PLM_LCL6_I is currently $Status_PLM_LCL6_I (extracted from TLM YM312942)"

set Status_PLM_LCL7_V [getrawvalue [fetch YM324942]]
logm "Status_PLM_LCL7_V is currently $Status_PLM_LCL7_V (extracted from TLM YM324942)"

set Status_PLM_LCL7_I [getrawvalue [fetch YM328942]]
logm "Status_PLM_LCL7_I is currently $Status_PLM_LCL7_I (extracted from TLM YM328942)"

set Status_PLM_LCL8_V [getrawvalue [fetch YM340942]]
logm "Status_PLM_LCL8_V is currently $Status_PLM_LCL8_V (extracted from TLM YM340942)"

set Status_PLM_LCL8_I [getrawvalue [fetch YM344942]]
logm "Status_PLM_LCL8_I is currently $Status_PLM_LCL8_I (extracted from TLM YM344942)"

set Status_PLM_LCL9_V [getrawvalue [fetch YM356942]]
logm "Status_PLM_LCL9_V is currently $Status_PLM_LCL9_V (extracted from TLM YM356942)"
```

```
set Status_PLM_LCL9_I [getrawvalue [fetch YM360942]]
logm "Status_PLM_LCL9_I is currently $Status_PLM_LCL9_I (extracted from TLM YM360942)"

set Status_PLM_LCL10_V [getrawvalue [fetch YM372942]]
logm "Status_PLM_LCL10_V is currently $Status_PLM_LCL10_V (extracted from TLM YM372942)"

set Status_PLM_LCL10_I [getrawvalue [fetch YM376942]]
logm "Status_PLM_LCL10_I is currently $Status_PLM_LCL10_I (extracted from TLM YM376942)"

set Status_PLM_LCL11_V [getrawvalue [fetch YM388942]]
logm "Status_PLM_LCL11_V is currently $Status_PLM_LCL11_V (extracted from TLM YM388942)"

set Status_PLM_LCL11_I [getrawvalue [fetch YM392942]]
logm "Status_PLM_LCL11_I is currently $Status_PLM_LCL11_I (extracted from TLM YM392942)"

set Status_PLM_LCL12_V [getrawvalue [fetch YM404942]]
logm "Status_PLM_LCL12_V is currently $Status_PLM_LCL12_V (extracted from TLM YM404942)"

set Status_PLM_LCL12_I [getrawvalue [fetch YM408942]]
logm "Status_PLM_LCL12_I is currently $Status_PLM_LCL12_I (extracted from TLM YM408942)"

set Status_PLM_LCL13_V [getrawvalue [fetch YM420942]]
logm "Status_PLM_LCL13_V is currently $Status_PLM_LCL13_V (extracted from TLM YM420942)"

set Status_PLM_LCL13_I [getrawvalue [fetch YM424942]]
logm "Status_PLM_LCL13_I is currently $Status_PLM_LCL13_I (extracted from TLM YM424942)"

set Status_PLM_LCL14_V [getrawvalue [fetch YM436942]]
logm "Status_PLM_LCL14_V is currently $Status_PLM_LCL14_V (extracted from TLM YM436942)"

set Status_PLM_LCL14_I [getrawvalue [fetch YM440942]]
logm "Status_PLM_LCL14_I is currently $Status_PLM_LCL14_I (extracted from TLM YM440942)"
logm ""


### Start Up Instruments
logm ""
newTest_gen "Power On Instruments"
logm ""
logm ""
logm ""
logm ">>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>"
logm ">>>>>>>>  Start Up Instruments"
logm ">>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>"
logm ""

set go_for_instrument "1"

set LCL_name(0) "N/A"
set LCL_name(1) "SPIRE HSDPU"
set LCL_name(2) "SPIRE HSFCU"
set LCL_name(3) "HIFI ICU"
set LCL_name(4) "HIFI LCU"
set LCL_name(5) "HIFI WEH"
set LCL_name(6) "HIFI WEV"
set LCL_name(7) "HIFI HRH"
set LCL_name(8) "HIFI HRV"
set LCL_name(9) "NCA"
set LCL_name(10) "CCU"
set LCL_name(11) "PACS BOLC"
set LCL_name(12) "PACS MEC"
set LCL_name(13) "PACS DPU"
set LCL_name(14) "PACS SPU"

set order(1) "first"
set order(2) "second"
set order(3) "third"
set order(4) "fourth"
set order(5) "fifth"
set order(6) "sixth"

while {$go_for_instrument == 1} {
        set result "0"
        set result [inputbox "Which instrument needs to be Powered? PACS, SPIRE, HIFI, CCU?"]
        set instrument $result
        logm " You have selected to power $instrument."

        if {$instrument == "CCU"} {
                set name_check "1"
                set PSU(1) "2"
                set PSU(2) ""
                set LCL(1) "10"
                set LCL(2) "0"
                set LCL(3) "0"
                set LCL(4) "0"
                set LCL(5) "0"
                set LCL(6) "0"
                set prim_red(1) "2"
```

```
		set prim_red(2) "2"
		set prim_red(3) "2"
		set prim_red(4) "2"
		set prim_red(5) "2"
		set prim_red(6) "2"
		set tot_LCL "1"
		set tot_PSU "1"

	} elseif {$instrument == "PACS"} {
		set name_check "1"
		set PSU(1) "2"
		set PSU(2) ""
		set LCL(1) "13"
		set LCL(2) "12"
		set LCL(3) "11"
		set LCL(4) "14"
		set LCL(5) "0"
		set LCL(6) "0"
		set prim_red(1) "2"
		set prim_red(2) "2"
		set prim_red(3) "2"
		set prim_red(4) "2"
		set prim_red(5) "2"
		set prim_red(6) "2"
		set tot_LCL "4"
		set tot_PSU "1"

	} elseif {$instrument == "SPIRE"} {
		set name_check "1"
		set PSU(1) "1"
		set PSU(2) ""
		set LCL(1) "1"
		set LCL(2) "0"
		set LCL(3) "0"
		set LCL(4) "0"
		set LCL(5) "0"
		set LCL(6) "0"
		set prim_red(1) "2"
		set prim_red(2) "2"
		set prim_red(3) "2"
		set prim_red(4) "2"
		set prim_red(5) "2"
		set prim_red(6) "2"
		set tot_LCL "2"
		set tot_PSU "1"

	} elseif {$instrument == "HIFI"} {
		set name_check "1"
		set PSU(1) "1"
		set PSU(2) "2"
		set LCL(1) "3"
		set LCL(2) "7"
		set LCL(3) "5"
		set LCL(4) "4"
		set LCL(5) "0"
		set LCL(6) "0"
		set prim_red(1) "2"
		set prim_red(2) "2"
		set prim_red(3) "2"
		set prim_red(4) "2"
		set prim_red(5) "2"
		set prim_red(6) "2"
		set tot_LCL "6"
		set tot_PSU "2"

	} else {
		set name_check "0"
		logm ""
		logm " >>> The selected instrument is not available in this power on sequence."
		logm ""
	}

	set LCL_V(1) "YM228942"
	set LCL_I(1) "YM232942"
	set LCL_V(2) "YM244942"
	set LCL_I(2) "YM248942"
	set LCL_V(3) "YM260942"
	set LCL_I(3) "YM264942"
	set LCL_V(4) "YM276942"
	set LCL_I(4) "YM280942"
	set LCL_V(5) "YM292942"
	set LCL_I(5) "YM296942"
	set LCL_V(6) "YM308942"
	set LCL_I(6) "YM312942"
	set LCL_V(7) "YM324942"
	set LCL_I(7) "YM328942"
	set LCL_V(8) "YM340942"
	set LCL_I(8) "YM344942"
```

```
		set LCL_V(9) "YM356942"
		set LCL_I(9) "YM360942"
		set LCL_V(10) "YM372942"
		set LCL_I(10) "YM376942"
		set LCL_V(11) "YM388942"
		set LCL_I(11) "YM392942"
		set LCL_V(12) "YM404942"
		set LCL_I(12) "YM408942"
		set LCL_V(13) "YM420942"
		set LCL_I(13) "YM424942"
		set LCL_V(14) "YM436942"
		set LCL_I(14) "YM440942"

		set PSU_Master(1) "YM129942"
		set PSU_Slave(1) "YM145942"
		set PSU_Master(2) "YM177942"
		set PSU_Slave(2) "YM193942"

		set txt_prim_red(1) "Primary"
		set txt_prim_red(2) "Primary"
		set txt_prim_red(3) "Primary"
		set txt_prim_red(4) "Primary"
		set txt_prim_red(5) "Primary"
		set txt_prim_red(6) "Primary"
		set txt_prim_red(7) "Primary"
		set txt_prim_red(8) "Primary"
		set txt_prim_red(9) "Primary"
		set txt_prim_red(10) "Primary"
		set txt_prim_red(11) "Primary"
		set txt_prim_red(12) "Primary"
		set txt_prim_red(13) "Primary"
		set txt_prim_red(14) "Primary"

		if {$name_check == 1} {

			set chkLCL(1) $LCL(1)
			set chkLCL(2) $LCL(2)
			set chkLCL(3) $LCL(3)
			set chkLCL(4) $LCL(4)
			set chkLCL(5) $LCL(5)
			set chkLCL(6) $LCL(6)

			set orderloop "1"
			while {$orderloop == 1} {
				logm ""
				logm " The current power on order is:"
				logm " ---------------------------------"
				set intLCL "1"
				while {$intLCL <= $tot_LCL} {
					if {$LCL($intLCL) != "0"} {
						set Status_Voltage [getrawvalue [fetch $LCL_V($LCL($intLCL))]]
						set Status_Current [getrawvalue [fetch $LCL_I($LCL($intLCL))]]
					} else {
						set Status_Voltage "N/A"
						set Status_Current "N/A"
					}
					if {$prim_red($intLCL) == "2"} {
						set txt_prim_red($LCL($intLCL)) "Primary"
					} elseif { $prim_red($intLCL) == "3"} {
						set txt_prim_red($LCL($intLCL)) "Secondary"
					} elseif {$prim_red($intLCL) == "4"} {
						set txt_prim_red($LCL($intLCL)) "Both Primary & Secondary"
					} else {
						set txt_prim_red($LCL($intLCL)) "Not Selected"
					}
					logm " $intLCL. LCL $LCL($intLCL)    $LCL_name($LCL($intLCL))    $txt_prim_red($LCL($intLCL))
Voltage: $Status_Voltage V    Current: $Status_Current A"
					set intLCL [expr $intLCL + 1]
				}
				logm ""

				set result "0"
				set result [yesorno "Do you want to change this order?"]
				logm ""

				if {$result == 5} {
					set intLCL "1"
					while {$intLCL <= $tot_LCL} {
						set result "0"
						set checkLCL "1"
						while {$checkLCL == 1} {
							set result [inputbox "Which LCL needs to be switched $order($intLCL)? Enter
0 for none."]

							# check if LCL is valid
							if {$result == $chkLCL(1) || $result == $chkLCL(2) || $result == $chkLCL(3)
|| $result == $chkLCL(4) || $result == $chkLCL(5) || $result == $chkLCL(6) || $result == "0"} {
								set checkLCL "0"
								logm " $result"
```

```
                                                                        } else {
                                                                        logm " !!! The chosen LCL ($result) is not valid for this instrument."
                                                                        logm " !!! You can chose between $chkLCL(1), $chkLCL(2), $chkLCL(3),
$chkLCL(4), $chkLCL(5), $chkLCL(6), 0."
                                                                        logm ""
                                                                        }
                                                                }
                                                                set LCL($intLCL) $result

                                                                set result "0"
                                                                set check_prim_red "1"
                                                                if {$LCL($intLCL) != "0"} {
                                                                        while {$check_prim_red == 1} {
                                                                                set result [inputbox "Select primary (2), redundant (3) or both (4)
power for LCL $LCL($intLCL)?"]
                                                                                # check if prim_red choice is valid
                                                                                if {$result == 2 || $result == 3 || $result == 4} {
                                                                                        set check_prim_red "0"
                                                                                        logm " $result"
                                                                                } else {
                                                                                logm " !!! You have to chose 2 for primary, 3 for redundant power
or 4 for both!"
                                                                                logm " "
                                                                                }
                                                                        }
                                                                }
                                                                set prim_red($intLCL) $result

                                                                set intLCL [expr $intLCL + 1]
                                                        }
                                                } else {
                                                        set orderloop "0"
                                                }
                                        }


                                        # Set some parameters needed to send the commands
                                        set paramYP431942 "1"
                                set paramYP433942 "2"

                                        set pin_prim_go "2"
                                        set pin_prim_ret "4"
                                        set pin_red_go "7"
                                        set pin_red_ret "9"

                                        set deviation "0.25"

                                        # do the actual power on
                                        set result "0"
                                        set result [yesorno "Do you want to enable the PSU(s)?"]
                                        logm ""

                                        if {$result == 5} {
                                                set intPSU "1"
                                                while {$intPSU <= $tot_PSU} {
                                                        # Execute Telecommand
                                           tcsend YC036942 checks {SPTV DPTV CEV} ack {ACCEPT } referby rYC036942 \
                                           {YP361942 1} \
                                        [list YP362942 [expr $PSU($intPSU) -1] ]
                                           logm " Sending Telecommand YC036942"

                                        # Control Execution
                                        logm " Synchronizing on SEV..."
                                        if { [waitfor -timeout 15000 { rYC036942 } -until { [getcompleted $rYC036942] } ]} {
                                                        logm " Synchronised on SEV for TC(s): YC036942"
                                        } else {
                                                logm " SEV synchronisation timed out for TC(s): YC036942"
                                        }
                                                        logm ""
                                                        logm " >>> Checking"
                                                        waittime 6.00
                                                        set Status_Master [getrawvalue [fetch $PSU_Master($PSU($intPSU))]]
                                                        set Status_Slave [getrawvalue [fetch $PSU_Slave($PSU($intPSU))]]
                                        logm " PSU $PSU($intPSU) Master status is currently $Status_Master (from $PSU_Master($PSU($intPSU)))"
                                        logm " PSU $PSU($intPSU) Slave status is currently $Status_Slave (from $PSU_Slave($PSU($intPSU)))"
                                        logm ""

                                                        # Do check
                                                        if {$Status_Master == 1 && $Status_Slave ==1} {
                                                                set check 1
                                                        } else {
                                                                set check 0
                                                        }

                                                        if {$check == 1} {
                                                                set result "0"
                                                                set result [infom "Check Successful! PSU $PSU($intPSU) has been enabled."]
                                                                logm ""
```

```
                                        set intPSU [expr $intPSU + 1]
                                } else {
                                        set result "0"
                                        set result [yesorno "PSU $PSU($intPSU) has not been enabled. Repeat this step?"]
                                        logm ""
                                        if {$result != 5} {
                                                set intPSU [expr $intPSU + 1]
                                        }
                                }
                        }
                } else {
                        set Status_Master [getrawvalue [fetch $PSU_Master(1)]]
                        set Status_Slave [getrawvalue [fetch $PSU_Slave(1)]]
                logm " PSU 1 Master status is currently $Status_Master (from $PSU_Master(1))"
                logm " PSU 1 Slave status is currently $Status_Slave (from $PSU_Slave(1))"
                        set Status_Master [getrawvalue [fetch $PSU_Master(2)]]
                        set Status_Slave [getrawvalue [fetch $PSU_Slave(2)]]
                logm " PSU 2 Master status is currently $Status_Master (from $PSU_Master(2))"
                logm " PSU 2 Slave status is currently $Status_Slave (from $PSU_Slave(2))"

                }

                # LCL's
                logm " >>> Start Enabling LCL's"
                logm ""
                set acLCL "1"
                while {$acLCL <= $tot_LCL} {
                        if {$LCL($acLCL) == "0"} {
                                set result "0"
                                set result [infom "No LCL is selected to be switched on as $order($acLCL)"]
                                logm ""
                        } else {
                                set result "0"
                                set result [yesorno "Do you want to enable LCL $LCL($acLCL)?"]
                                logm ""
                                if {$result == 5} {
                                        set enable_LCL "1"
                                        while {$enable_LCL == 1} {
                                                # Execute Telecommand
                                                tcsend YC040942 checks {SPTV DPTV CEV} ack NONE referby rYC040942 \
                                        {YP401942 1} \
                                [list YP402942 $LCL($acLCL)]
                                        logm " Sending Telecommand YC040942 to Enable Limiter"

                                # Control Execution
                                logm " Synchronizing on SEV..."
                                   if { [waitfor -timeout 15000 { rYC040942 } -until { [getcompleted $rYC040942] } ]} {
                                        logm " Synchronised on SEV for TC(s): YC040942"
                                        } else {
                                                logm " SEV synchronisation timed out for TC(s): YC040942"
                                    }
                                                logm ""

                                                # Execute Telecommand
                                                tcsend YC043942 checks {SPTV DPTV CEV} ack NONE referby rYC043942 \
                                                        [list YP431942 $paramYP431942]  \
                                                        [list YP432942 $LCL($acLCL)]  \
                                                        [list YP433942 $prim_red($acLCL)]

                                        logm " Sending Telecommand YC043942 to Set Limiter"

                                # Control Execution
                                        logm " Synchronizing on SEV..."
                                           if { [waitfor -timeout 15000 { rYC043942 } -until { [getcompleted $rYC043942] } ]} {
                                        logm " Synchronised on SEV for TC(s): YC043942"
                                        } else {
                                                logm " SEV synchronisation timed out for TC(s): YC043942"
                                    }
                                                logm ""

                                                logm " >>> Checking"
                                                waittime 6.00
                                                set Status_Voltage [getrawvalue [fetch $LCL_V($LCL($acLCL))]]
                                                set Status_Current [getrawvalue [fetch $LCL_I($LCL($acLCL))]]
                                                logm " LCL $LCL($acLCL) has currently a voltage of $Status_Voltage.(from
$LCL_V($LCL($acLCL)))"
                                                logm " LCL $LCL($acLCL) has currently a current of $Status_Current.(from
$LCL_I($LCL($acLCL)))"
                                                logm ""

                                                # Do check
                                                if {$Status_Voltage >= 27} {
                                                        set check 1
                                                } else {
                                                        set check 0
                                                }

                                                if {$check == 1} {
```

```
                                                                        set result "0"
                                                                        set result [infom "Check Successful! LCL $LCL($acLCL) has been
enabled."]
                                                                        logm ""
                                                                        set enable_LCL "0"
                                                                } else {
                                                                        set result "0"
                                                                        set result [yesorno "LCL $LCL($acLCL) has not been enabled.
Repeat this step?"]
                                                                        logm ""
                                                                        if {$result != 5} {
                                                                                set enable_LCL "0"
                                                                        }
                                                                }
                                                        }
                                                }
                                        }
                                        set acLCL [expr $acLCL + 1]
                                }

                                logm ""
                                logm " All selected LCL's for $instrument are powered."
                                logm ""

                                set result "0"
                                set result [yesorno "Do you want to power on another instrument?"]
                                logm ""
                                if {$result == 5} {
                                        set go_for_instrument "1"
                                } else {
                                        set go_for_instrument "0"
                                }
                        }
}

### Reading out PLM SCOE Settings
logm ">>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>"
logm ">>>>>>>>>  Reading out PLM SCOE Settings"
logm ">>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>"
logm ""

set Status_PLM_OnLine [getrawvalue [fetch YM018942]]
logm "Status_PLM_OnLine is $Status_PLM_OnLine (extracted from TLM YM018942)"

set Status_PLM_PSU1_Master [getrawvalue [fetch YM129942]]
logm "Status_PLM_PSU1_Master is currently $Status_PLM_PSU1_Master (extracted from TLM YM129942)"

set Status_PLM_PSU1_Slave [getrawvalue [fetch YM145942]]
logm "Status_PLM_PSU1_Slave is currently $Status_PLM_PSU1_Slave (extracted from TLM YM145942)"

set Status_PLM_PSU2_Master [getrawvalue [fetch YM177942]]
logm "Status_PLM_PSU2_Master is currently $Status_PLM_PSU2_Master (extracted from TLM YM177942)"

set Status_PLM_PSU2_Slave [getrawvalue [fetch YM193942]]
logm "Status_PLM_PSU2_Slave is currently $Status_PLM_PSU2_Slave (extracted from TLM YM193942)"

set Status_PLM_LCL1_V [getrawvalue [fetch YM228942]]
logm "Status_PLM_LCL1_V is currently $Status_PLM_LCL1_V (extracted from TLM YM228942)"

set Status_PLM_LCL1_I [getrawvalue [fetch YM232942]]
logm "Status_PLM_LCL1_I is currently $Status_PLM_LCL1_I (extracted from TLM YM232942)"

set Status_PLM_LCL2_V [getrawvalue [fetch YM244942]]
logm "Status_PLM_LCL2_V is currently $Status_PLM_LCL2_V (extracted from TLM YM244942)"

set Status_PLM_LCL2_I [getrawvalue [fetch YM248942]]
logm "Status_PLM_LCL2_I is currently $Status_PLM_LCL2_I (extracted from TLM YM248942)"

set Status_PLM_LCL3_V [getrawvalue [fetch YM260942]]
logm "Status_PLM_LCL3_V is currently $Status_PLM_LCL3_V (extracted from TLM YM260942)"

set Status_PLM_LCL3_I [getrawvalue [fetch YM264942]]
logm "Status_PLM_LCL3_I is currently $Status_PLM_LCL3_I (extracted from TLM YM264942)"

set Status_PLM_LCL4_V [getrawvalue [fetch YM276942]]
logm "Status_PLM_LCL4_V is currently $Status_PLM_LCL4_V (extracted from TLM YM276942)"

set Status_PLM_LCL4_I [getrawvalue [fetch YM280942]]
logm "Status_PLM_LCL4_I is currently $Status_PLM_LCL4_I (extracted from TLM YM280942)"

set Status_PLM_LCL5_V [getrawvalue [fetch YM292942]]
logm "Status_PLM_LCL5_V is currently $Status_PLM_LCL5_V (extracted from TLM YM292942)"

set Status_PLM_LCL5_I [getrawvalue [fetch YM296942]]
logm "Status_PLM_LCL5_I is currently $Status_PLM_LCL5_I (extracted from TLM YM296942)"

set Status_PLM_LCL6_V [getrawvalue [fetch YM308942]]
logm "Status_PLM_LCL6_V is currently $Status_PLM_LCL6_V (extracted from TLM YM308942)"
```

```
set Status_PLM_LCL6_I [getrawvalue [fetch YM312942]]
logm "Status_PLM_LCL6_I is currently $Status_PLM_LCL6_I (extracted from TLM YM312942)"

set Status_PLM_LCL7_V [getrawvalue [fetch YM324942]]
logm "Status_PLM_LCL7_V is currently $Status_PLM_LCL7_V (extracted from TLM YM324942)"

set Status_PLM_LCL7_I [getrawvalue [fetch YM328942]]
logm "Status_PLM_LCL7_I is currently $Status_PLM_LCL7_I (extracted from TLM YM328942)"

set Status_PLM_LCL8_V [getrawvalue [fetch YM340942]]
logm "Status_PLM_LCL8_V is currently $Status_PLM_LCL8_V (extracted from TLM YM340942)"

set Status_PLM_LCL8_I [getrawvalue [fetch YM344942]]
logm "Status_PLM_LCL8_I is currently $Status_PLM_LCL8_I (extracted from TLM YM344942)"

set Status_PLM_LCL9_V [getrawvalue [fetch YM356942]]
logm "Status_PLM_LCL9_V is currently $Status_PLM_LCL9_V (extracted from TLM YM356942)"

set Status_PLM_LCL9_I [getrawvalue [fetch YM360942]]
logm "Status_PLM_LCL9_I is currently $Status_PLM_LCL9_I (extracted from TLM YM360942)"

set Status_PLM_LCL10_V [getrawvalue [fetch YM372942]]
logm "Status_PLM_LCL10_V is currently $Status_PLM_LCL10_V (extracted from TLM YM372942)"

set Status_PLM_LCL10_I [getrawvalue [fetch YM376942]]
logm "Status_PLM_LCL10_I is currently $Status_PLM_LCL10_I (extracted from TLM YM376942)"

set Status_PLM_LCL11_V [getrawvalue [fetch YM388942]]
logm "Status_PLM_LCL11_V is currently $Status_PLM_LCL11_V (extracted from TLM YM388942)"

set Status_PLM_LCL11_I [getrawvalue [fetch YM392942]]
logm "Status_PLM_LCL11_I is currently $Status_PLM_LCL11_I (extracted from TLM YM392942)"

set Status_PLM_LCL12_V [getrawvalue [fetch YM404942]]
logm "Status_PLM_LCL12_V is currently $Status_PLM_LCL12_V (extracted from TLM YM404942)"

set Status_PLM_LCL12_I [getrawvalue [fetch YM408942]]
logm "Status_PLM_LCL12_I is currently $Status_PLM_LCL12_I (extracted from TLM YM408942)"

set Status_PLM_LCL13_V [getrawvalue [fetch YM420942]]
logm "Status_PLM_LCL13_V is currently $Status_PLM_LCL13_V (extracted from TLM YM420942)"

set Status_PLM_LCL13_I [getrawvalue [fetch YM424942]]
logm "Status_PLM_LCL13_I is currently $Status_PLM_LCL13_I (extracted from TLM YM424942)"

set Status_PLM_LCL14_V [getrawvalue [fetch YM436942]]
logm "Status_PLM_LCL14_V is currently $Status_PLM_LCL14_V (extracted from TLM YM436942)"

set Status_PLM_LCL14_I [getrawvalue [fetch YM440942]]
logm "Status_PLM_LCL14_I is currently $Status_PLM_LCL14_I (extracted from TLM YM440942)"

#End
logm "<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<"
logm "<<<<<<<  Power Sequence Ended!"
logm "<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<"

finish_TS


#####################################################################
# end of test sequence
#####################################################################
# Changes:
# $Log: INSTR_POWER_ON.tcl,v $
# Revision 1.6  2005/06/02 10:05:03  ilsens
# (No log message)
#
# Revision 1.5  2005/06/01 13:07:29  ilsens
# (No log message)
#
# Revision 1.4  2005/05/31 14:37:21  ilsens
# (No log message)
#
# Revision 1.3  2005/05/04 09:01:20  ilsens
# (No log message)
#
# Revision 1.2  2005/04/21 14:11:28  ilsens
# (No log message)
#
# Revision 1.1  2005/04/21 14:06:42  ilsens
# initial version
#
#
#####################################################################
```

Appendix 4: INSTR_POWER_OFF.tcl

```
###################################################################
# File: $Id: INSTR_POWER_OFF.tcl,v 1.5 2005/06/02 10:08:07 ilsens Exp $
#
# Description:
#
#               Power down the selected instruments using an interactive log
#
#
# Last edited by: $Author: ilsens $ on $Date: 2005/06/02 10:08:07 $.
#
#
###################################################################

# automatically set the revision. do not edit this
setrevision {$Id: INSTR_POWER_OFF.tcl,v 1.5 2005/06/02 10:08:07 ilsens Exp $}

###################################################################
# start of test sequence
###################################################################


global env
global go_on
global auto_path

set go_on -1

lappend auto_path "/HPCCS/VARIABLE/CONFIG/TCL/Alenia"

# ------------------------------
# creation of the output file
# ------------------------------
gettime sec usec
set fn_time [clock format $sec -format %Y%m%d_%H%M%S -gmt true]
set fn_name [file tail [info script]]
set fn_name [string trim $fn_name ".tcl"]
set fn_add "GUI_log"
set fn_ID "XXXX"
append FileName $fn_time "_" $fn_ID "_" $fn_name "_" $fn_add ".txt"
putlog $FileName
set outfile [open $env(HPCCSTESTRES)/TSEQ/$FileName w]

# This is the mandatory procedures to abtain the output log window
setup_win

logm ""
newTest_gen "Start of Instrument POWER OFF sequence."
logm ""

logm " To run this script, the CDMU DFE and PLM SCOE should be"
logm " powered and configured. "
logm " To initiate, this script will connect and attach to the CDMUDFE"
logm " and PLM SCOE."
logm ""

logm "Connecting to CDMU DFE"
connect CDMUDFE
waittime +00.00.02.000000
logm "Attaching to CMDU DFE"
attach CDMUDFE
waittime +00.00.01.000000
logm ""

logm "Connecting to PLM SCOE"
connect PLMSCOE
waittime +00.00.02.000000
logm "Attaching to PLM SCOE"
attach PLMSCOE
waittime +00.00.01.000000

### Reading out CDMUDFE Settings
logm ">>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>"
logm ">>>>>>>>  Reading out CDMUDFE Settings"
logm ">>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>"
logm ""

set Status_CDMU_OnLine [getrawvalue [fetch YM777944]]
logm "Status_CDMU_OnLine is $Status_CDMU_OnLine (extracted from TLM YM777944)"

set Status_CDMU_TMpolling [getrawvalue [fetch YM780944]]
logm "Status_CDMU_TMpolling is $Status_CDMU_TMpolling (extracted from TLM YM780944)"

set Status_CDMU_SAreadActive [getrawvalue [fetch YM781944]]
logm "Status_CDMU_SAreadActive is $Status_CDMU_SAreadActive (extracted from TLM YM781944)"
```

```
set Status_CDMU_SAqueueActive [getrawvalue [fetch YM782944]]
logm "Status_CDMU_SAqueueActive is $Status_CDMU_SAqueueActive (extracted from TLM YM782944)"

set Status_CDMU_TMqueueActive [getrawvalue [fetch YM783944]]
logm "Status_CDMU_TMqueueActive is $Status_CDMU_TMqueueActive (extracted from TLM YM783944)"

set Status_CDMU_TCqueueActive [getrawvalue [fetch YM784944]]
logm "Status_CDMU_TCqueueActive is $Status_CDMU_TCqueueActive (extracted from TLM YM784944)"

set Status_CDMU_PSTfileName [getrawvalue [fetch YM809944]]
logm "Status_CDMU_PSTfileName is $Status_CDMU_PSTfileName (extracted from TLM YM809944)"

set Status_CDMU_PSTrunning [getrawvalue [fetch YM829944]]
logm "Status_CDMU_PSTrunning is $Status_CDMU_PSTrunning (extracted from TLM YM829944)"
logm ""

### Reading out PLM SCOE Settings
logm ">>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>"
logm ">>>>>>>>  Reading out PLM SCOE Settings"
logm ">>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>"
logm ""

set Status_PLM_OnLine [getrawvalue [fetch YM018942]]
logm "Status_PLM_OnLine is $Status_PLM_OnLine (extracted from TLM YM018942)"

set Status_PLM_PSU1_Master [getrawvalue [fetch YM129942]]
logm "Status_PLM_PSU1_Master is currently $Status_PLM_PSU1_Master (extracted from TLM YM129942)"

set Status_PLM_PSU1_Slave [getrawvalue [fetch YM145942]]
logm "Status_PLM_PSU1_Slave is currently $Status_PLM_PSU1_Slave (extracted from TLM YM145942)"

set Status_PLM_PSU2_Master [getrawvalue [fetch YM177942]]
logm "Status_PLM_PSU2_Master is currently $Status_PLM_PSU2_Master (extracted from TLM YM177942)"

set Status_PLM_PSU2_Slave [getrawvalue [fetch YM193942]]
logm "Status_PLM_PSU2_Slave is currently $Status_PLM_PSU2_Slave (extracted from TLM YM193942)"

set Status_PLM_LCL1_V [getrawvalue [fetch YM228942]]
logm "Status_PLM_LCL1_V is currently $Status_PLM_LCL1_V (extracted from TLM YM228942)"

set Status_PLM_LCL1_I [getrawvalue [fetch YM232942]]
logm "Status_PLM_LCL1_I is currently $Status_PLM_LCL1_I (extracted from TLM YM232942)"

set Status_PLM_LCL2_V [getrawvalue [fetch YM244942]]
logm "Status_PLM_LCL2_V is currently $Status_PLM_LCL2_V (extracted from TLM YM244942)"

set Status_PLM_LCL2_I [getrawvalue [fetch YM248942]]
logm "Status_PLM_LCL2_I is currently $Status_PLM_LCL2_I (extracted from TLM YM248942)"

set Status_PLM_LCL3_V [getrawvalue [fetch YM260942]]
logm "Status_PLM_LCL3_V is currently $Status_PLM_LCL3_V (extracted from TLM YM260942)"

set Status_PLM_LCL3_I [getrawvalue [fetch YM264942]]
logm "Status_PLM_LCL3_I is currently $Status_PLM_LCL3_I (extracted from TLM YM264942)"

set Status_PLM_LCL4_V [getrawvalue [fetch YM276942]]
logm "Status_PLM_LCL4_V is currently $Status_PLM_LCL4_V (extracted from TLM YM276942)"

set Status_PLM_LCL4_I [getrawvalue [fetch YM280942]]
logm "Status_PLM_LCL4_I is currently $Status_PLM_LCL4_I (extracted from TLM YM280942)"

set Status_PLM_LCL5_V [getrawvalue [fetch YM292942]]
logm "Status_PLM_LCL5_V is currently $Status_PLM_LCL5_V (extracted from TLM YM292942)"

set Status_PLM_LCL5_I [getrawvalue [fetch YM296942]]
logm "Status_PLM_LCL5_I is currently $Status_PLM_LCL5_I (extracted from TLM YM296942)"

set Status_PLM_LCL6_V [getrawvalue [fetch YM308942]]
logm "Status_PLM_LCL6_V is currently $Status_PLM_LCL6_V (extracted from TLM YM308942)"

set Status_PLM_LCL6_I [getrawvalue [fetch YM312942]]
logm "Status_PLM_LCL6_I is currently $Status_PLM_LCL6_I (extracted from TLM YM312942)"

set Status_PLM_LCL7_V [getrawvalue [fetch YM324942]]
logm "Status_PLM_LCL7_V is currently $Status_PLM_LCL7_V (extracted from TLM YM324942)"

set Status_PLM_LCL7_I [getrawvalue [fetch YM328942]]
logm "Status_PLM_LCL7_I is currently $Status_PLM_LCL7_I (extracted from TLM YM328942)"

set Status_PLM_LCL8_V [getrawvalue [fetch YM340942]]
logm "Status_PLM_LCL8_V is currently $Status_PLM_LCL8_V (extracted from TLM YM340942)"

set Status_PLM_LCL8_I [getrawvalue [fetch YM344942]]
logm "Status_PLM_LCL8_I is currently $Status_PLM_LCL8_I (extracted from TLM YM344942)"

set Status_PLM_LCL9_V [getrawvalue [fetch YM356942]]
logm "Status_PLM_LCL9_V is currently $Status_PLM_LCL9_V (extracted from TLM YM356942)"
```

```
set Status_PLM_LCL9_I [getrawvalue [fetch YM360942]]
logm "Status_PLM_LCL9_I is currently $Status_PLM_LCL9_I (extracted from TLM YM360942)"

set Status_PLM_LCL10_V [getrawvalue [fetch YM372942]]
logm "Status_PLM_LCL10_V is currently $Status_PLM_LCL10_V (extracted from TLM YM372942)"

set Status_PLM_LCL10_I [getrawvalue [fetch YM376942]]
logm "Status_PLM_LCL10_I is currently $Status_PLM_LCL10_I (extracted from TLM YM376942)"

set Status_PLM_LCL11_V [getrawvalue [fetch YM388942]]
logm "Status_PLM_LCL11_V is currently $Status_PLM_LCL11_V (extracted from TLM YM388942)"

set Status_PLM_LCL11_I [getrawvalue [fetch YM392942]]
logm "Status_PLM_LCL11_I is currently $Status_PLM_LCL11_I (extracted from TLM YM392942)"

set Status_PLM_LCL12_V [getrawvalue [fetch YM404942]]
logm "Status_PLM_LCL12_V is currently $Status_PLM_LCL12_V (extracted from TLM YM404942)"

set Status_PLM_LCL12_I [getrawvalue [fetch YM408942]]
logm "Status_PLM_LCL12_I is currently $Status_PLM_LCL12_I (extracted from TLM YM408942)"

set Status_PLM_LCL13_V [getrawvalue [fetch YM420942]]
logm "Status_PLM_LCL13_V is currently $Status_PLM_LCL13_V (extracted from TLM YM420942)"

set Status_PLM_LCL13_I [getrawvalue [fetch YM424942]]
logm "Status_PLM_LCL13_I is currently $Status_PLM_LCL13_I (extracted from TLM YM424942)"

set Status_PLM_LCL14_V [getrawvalue [fetch YM436942]]
logm "Status_PLM_LCL14_V is currently $Status_PLM_LCL14_V (extracted from TLM YM436942)"

set Status_PLM_LCL14_I [getrawvalue [fetch YM440942]]
logm "Status_PLM_LCL14_I is currently $Status_PLM_LCL14_I (extracted from TLM YM440942)"
logm ""

### Start Up Instruments
logm ""
newTest_gen "Power On Instruments"
logm ""
logm ""
logm ""
logm ">>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>"
logm ">>>>>>>>  Start Up Instruments"
logm ">>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>"
logm ""

set go_for_instrument "1"

set LCL_name(0) "N/A"
set LCL_name(1) "SPIRE HSDPU"
set LCL_name(2) "SPIRE HSFCU"
set LCL_name(3) "HIFI ICU"
set LCL_name(4) "HIFI LCU"
set LCL_name(5) "HIFI WEH"
set LCL_name(6) "HIFI WEV"
set LCL_name(7) "HIFI HRH"
set LCL_name(8) "HIFI HRV"
set LCL_name(9) "NCA"
set LCL_name(10) "CCU"
set LCL_name(11) "PACS BOLC"
set LCL_name(12) "PACS MEC"
set LCL_name(13) "PACS DPU"
set LCL_name(14) "PACS SPU"

set order(1) "first"
set order(2) "second"
set order(3) "third"
set order(4) "fourth"
set order(5) "fifth"
set order(6) "sixth"

while {$go_for_instrument == 1} {
        set result "0"
        # In this future the 'All' option could be included (if SwitchOffAllLimiters command works)
        set result [inputbox "Which instrument needs to be Powered down? PACS, SPIRE, HIFI, CCU?"]
        set instrument $result
        logm " You have selected to power down $instrument."

        if {$instrument == "CCU"} {
                set name_check "1"
                set PSU(1) "2"
                set PSU(2) ""
                set LCL(1) "10"
                set LCL(2) "0"
                set LCL(3) "0"
                set LCL(4) "0"
                set LCL(5) "0"
                set LCL(6) "0"
                set prim_red(1) "2"
```

```
		set prim_red(2) "2"
		set prim_red(3) "2"
		set prim_red(4) "2"
		set prim_red(5) "2"
		set prim_red(6) "2"
		set tot_LCL "1"
		set tot_PSU "1"

	} elseif {$instrument == "PACS"} {
		set name_check "1"
		set PSU(1) "2"
		set PSU(2) ""
		set LCL(1) "14"
		set LCL(2) "11"
		set LCL(3) "12"
		set LCL(4) "13"
		set LCL(5) "0"
		set LCL(6) "0"
		set prim_red(1) "2"
		set prim_red(2) "2"
		set prim_red(3) "2"
		set prim_red(4) "2"
		set prim_red(5) "2"
		set prim_red(6) "2"
		set tot_LCL "4"
		set tot_PSU "1"

	} elseif {$instrument == "SPIRE"} {
		set name_check "1"
		set PSU(1) "1"
		set PSU(2) ""
		set LCL(1) "1"
		set LCL(2) "0"
		set LCL(3) "0"
		set LCL(4) "0"
		set LCL(5) "0"
		set LCL(6) "0"
		set prim_red(1) "2"
		set prim_red(2) "2"
		set prim_red(3) "2"
		set prim_red(4) "2"
		set prim_red(5) "2"
		set prim_red(6) "2"
		set tot_LCL "2"
		set tot_PSU "1"

	} elseif {$instrument == "HIFI"} {
		set name_check "1"
		set PSU(1) "1"
		set PSU(2) "2"
		set LCL(1) "4"
		set LCL(2) "5"
		set LCL(3) "7"
		set LCL(4) "3"
		set LCL(5) "0"
		set LCL(6) "0"
		set prim_red(1) "2"
		set prim_red(2) "2"
		set prim_red(3) "2"
		set prim_red(4) "2"
		set prim_red(5) "2"
		set prim_red(6) "2"
		set tot_LCL "6"
		set tot_PSU "2"

	} elseif {$instrument == "All"} {
		set name_check "0"
		set allLCL "1"
	} else {
		set name_check "0"
		set allLCL "0"
		logm ""
		logm " >>> The selected instrument is not available in this power down sequence."
		logm ""
	}

	set LCL_V(1) "YM228942"
	set LCL_I(1) "YM232942"
	set LCL_V(2) "YM244942"
	set LCL_I(2) "YM248942"
	set LCL_V(3) "YM260942"
	set LCL_I(3) "YM264942"
	set LCL_V(4) "YM276942"
	set LCL_I(4) "YM280942"
	set LCL_V(5) "YM292942"
	set LCL_I(5) "YM296942"
	set LCL_V(6) "YM308942"
	set LCL_I(6) "YM312942"
```

```
		set LCL_V(7) "YM324942"
		set LCL_I(7) "YM328942"
		set LCL_V(8) "YM340942"
		set LCL_I(8) "YM344942"
		set LCL_V(9) "YM356942"
		set LCL_I(9) "YM360942"
		set LCL_V(10) "YM372942"
		set LCL_I(10) "YM376942"
		set LCL_V(11) "YM388942"
		set LCL_I(11) "YM392942"
		set LCL_V(12) "YM404942"
		set LCL_I(12) "YM408942"
		set LCL_V(13) "YM420942"
		set LCL_I(13) "YM424942"
		set LCL_V(14) "YM436942"
		set LCL_I(14) "YM440942"

		set PSU_Master(1) "YM129942"
		set PSU_Slave(1) "YM145942"
		set PSU_Master(2) "YM177942"
		set PSU_Slave(2) "YM193942"

		set txt_prim_red(1) "Primary"
		set txt_prim_red(2) "Primary"
		set txt_prim_red(3) "Primary"
		set txt_prim_red(4) "Primary"
		set txt_prim_red(5) "Primary"
		set txt_prim_red(6) "Primary"
		set txt_prim_red(7) "Primary"
		set txt_prim_red(8) "Primary"
		set txt_prim_red(9) "Primary"
		set txt_prim_red(10) "Primary"
		set txt_prim_red(11) "Primary"
		set txt_prim_red(12) "Primary"
		set txt_prim_red(13) "Primary"
		set txt_prim_red(14) "Primary"

		if {$name_check == 1} {

			set chkLCL(1) $LCL(1)
			set chkLCL(2) $LCL(2)
			set chkLCL(3) $LCL(3)
			set chkLCL(4) $LCL(4)
			set chkLCL(5) $LCL(5)
			set chkLCL(6) $LCL(6)

			set orderloop "1"
			while {$orderloop == 1} {
				logm ""
				logm " The current power down order is:"
				logm " ---------------------------------"
				set intLCL "1"
				while {$intLCL <= $tot_LCL} {
				if {$LCL($intLCL) != "0"} {
						set Status_Voltage [getrawvalue [fetch $LCL_V($LCL($intLCL))]]
						set Status_Current [getrawvalue [fetch $LCL_I($LCL($intLCL))]]
					} else {
						set Status_Voltage "N/A"
						set Status_Current "N/A"
					}
					logm " $intLCL. LCL $LCL($intLCL)    $LCL_name($LCL($intLCL))     Voltage: $Status_Voltage V
Current: $Status_Current A"
					set intLCL [expr $intLCL + 1]
				}
				logm ""

				set result "0"
				set result [yesorno "Do you want to change this order?"]
				logm ""

				if {$result == 5} {
					set intLCL "1"
					while {$intLCL <= $tot_LCL} {
						set result "0"
						set checkLCL "1"
						while {$checkLCL == 1} {
							set result [inputbox "Which LCL needs to be switched $order($intLCL)? Enter
0 for none."]
							# check if LCL is valid
							if {$result == $chkLCL(1) || $result == $chkLCL(2) || $result == $chkLCL(3)
|| $result == $chkLCL(4) || $result == $chkLCL(5) || $result == $chkLCL(6) || $result == "0"} {
								set checkLCL "0"
								logm " $result"
							} else {
							logm " !!! The chosen LCL ($result) is not valid for this instrument."
							logm " !!! You can chose between $chkLCL(1), $chkLCL(2), $chkLCL(3),
$chkLCL(4), $chkLCL(5), $chkLCL(6), 0."
							logm ""
```

```
                                                }
                                        }
                                        set LCL($intLCL) $result
                                        set intLCL [expr $intLCL + 1]
                                }
                        } else {
                                set orderloop "0"
                        }
                }


                # Set some parameters needed to send the commands
                set paramYP431942 "1"
        set paramYP433942 "2"

                set pin_prim_go "2"
                set pin_prim_ret "4"
                set pin_red_go "7"
                set pin_red_ret "9"

                set deviation "0.25"

                # LCL's
                logm " >>> Disable LCL's"
                logm ""
                set acLCL "1"
                while {$acLCL <= $tot_LCL} {
                        if {$LCL($acLCL) == "0"} {
                                set result "0"
                                set result [infom "No LCL is selected to be switched on as $order($acLCL)"]
                                logm ""
                        } else {
                                set result "0"
                                set result [yesorno "Do you want to disable LCL $LCL($acLCL)?"]
                                logm ""
                                if {$result == 5} {
                                        set disable_LCL "1"
                                        while {$disable_LCL == 1} {
                                                # Execute Telecommand
                                                tcsend YC041942 checks {SPTV DPTV CEV} ack NONE referby rYC041942 \
                                        {YP411942 1} \
                                [list YP412942 $LCL($acLCL)]
                                        logm " Sending Telecommand YC041942 to Disable Limiter"

                                # Control Execution
                                        logm " Synchronizing on SEV..."
                                   if { [waitfor -timeout 15000 { rYC041942 } -until { [getcompleted $rYC041942] } ]} {
                                        logm " Synchronised on SEV for TC(s): YC041942"
                                                } else {
                                                        logm " SEV synchronisation timed out for TC(s): YC041942"
                                        }
                                                logm ""

                                                logm " >>> Checking"
                                                waittime 6.00
                                                set Status_Voltage [getrawvalue [fetch $LCL_V($LCL($acLCL))]]
                                                set Status_Current [getrawvalue [fetch $LCL_I($LCL($acLCL))]]
                                                logm " LCL $LCL($acLCL) has currently a voltage of $Status_Voltage.(from
$LCL_V($LCL($acLCL)))"
                                                logm " LCL $LCL($acLCL) has currently a current of $Status_Current.(from
$LCL_I($LCL($acLCL)))"
                                                logm ""

                                                # Do check
                                                if {$Status_Voltage <= "0.25"} {
                                                        set check 1
                                                } else {
                                                        set check 0
                                                }

                                                if {$check == 1} {
                                                        set result "0"
                                                        set result [infom "Check Successful! LCL $LCL($acLCL) has been
disabled."]
                                                        logm ""
                                                        set disable_LCL "0"
                                                } else {
                                                        set result "0"
                                                        set result [yesorno "LCL $LCL($acLCL) has not been disabled.
Repeat this step?"]
                                                        logm ""
                                                        if {$result != 5} {
                                                                set disable_LCL "0"
                                                        }
                                                }
                                        }
                                }
                        }
                }
```

```
                set acLCL [expr $acLCL + 1]
        }

        # do the actual power down of PSU's
        set result "0"
        set result [yesorno "Do you want to disable PSU(s)?"]
        logm ""

        if {$result == 5} {
                set intPSU "1"
                while {$intPSU <= $tot_PSU} {
                        set result "0"
                        set result [yesorno "Do you want to disable PSU $PSU($intPSU)?"]
                        logm ""
                        if {$result == 5} {
                                # Execute Telecommand
           tcsend YC037942 checks {SPTV DPTV CEV} ack {ACCEPT } referby rYC037942 \
                                {YP371942 1} \
                        [list YP372942 [expr $PSU($intPSU) -1] ]
           logm " Sending Telecommand YC037942"

                # Control Execution
                logm " Synchronizing on SEV..."
                if { [waitfor -timeout 15000 { rYC037942 } -until { [getcompleted $rYC037942] } ]} {
                                logm " Synchronised on SEV for TC(s): YC037942"
                } else {
                        logm " SEV synchronisation timed out for TC(s): YC037942"
                }
                                logm ""
                                logm " >>> Checking"
                                waittime 6.00
                                set Status_Master [getrawvalue [fetch $PSU_Master($PSU($intPSU))]]
                                set Status_Slave [getrawvalue [fetch $PSU_Slave($PSU($intPSU))]]
                logm " PSU $PSU($intPSU) Master status is currently $Status_Master (from $PSU_Master($PSU($intPSU)))"
                logm " PSU $PSU($intPSU) Slave status is currently $Status_Slave (from $PSU_Slave($PSU($intPSU)))"
                logm ""

                                # Do check
                                if {$Status_Master == 0 && $Status_Slave == 0} {
                                        set check 1
                                } else {
                                        set check 0
                                }

                                if {$check == 1} {
                                        set result "0"
                                        set result [infom "Check Successful! PSU $PSU($intPSU) has been disabled."]
                                        logm ""
                                        set intPSU [expr $intPSU + 1]
                                } else {
                                        set result "0"
                                        set result [yesorno "PSU $PSU($intPSU) has not been disabled. Repeat this
step?"]
                                        logm ""
                                        if {$result != 5} {
                                                set intPSU [expr $intPSU + 1]
                                        }
                                }
                        } else {
                                set intPSU [expr $intPSU + 1]
                        }
                }
        } else {
                set Status_Master [getrawvalue [fetch $PSU_Master(1)]]
                set Status_Slave [getrawvalue [fetch $PSU_Slave(1)]]
        logm " PSU 1 Master status is currently $Status_Master (from $PSU_Master(1))"
        logm " PSU 1 Slave status is currently $Status_Slave (from $PSU_Slave(1))"
                set Status_Master [getrawvalue [fetch $PSU_Master(2)]]
                set Status_Slave [getrawvalue [fetch $PSU_Slave(2)]]
        logm " PSU 2 Master status is currently $Status_Master (from $PSU_Master(2))"
        logm " PSU 2 Slave status is currently $Status_Slave (from $PSU_Slave(2))"
        }

        logm ""
        logm " Power down of $instrument is done."
        logm ""

        set result "0"
        set result [yesorno "Do you want to power down another instrument?"]
        logm ""
        if {$result == 5} {
                set go_for_instrument "1"
        } else {
                set go_for_instrument "0"
        }
} else {
        if {$allLCL == 1} {
                set result "0"
```

```
                                    set result [yesorno "Are you sure to switch off all LCL's?"]
                                    logm ""
                                    if {$result == 5} {
                                            #
                                            # Not working at the moment!
                                            #
                                            # Execute Telecommand
                                    tcsend YC033942 checks {SPTV DPTV CEV} ack {ACCEPT } referby rYC033942
                                            logm " Sending Telecommand YC033942"

                                    # Control Execution
                                            logm " Synchronizing on SEV..."
                                    if { [waitfor -timeout 15000 { rYC033942 } -until { [getcompleted $rYC033942] } ]} {
                                                    logm " Synchronised on SEV for TC(s): YC033942"
                                            } else {
                                                    logm " SEV synchronisation timed out for TC(s): YC033942"
                                    }
                                            logm ""
                                    }
                                    set PSU(1) "1"
                                    set PSU(2) "2"
                                    set tot_PSU "1"
                                    set intPSU "1"
                                    while {$intPSU <= $tot_PSU} {
                                            set result "0"
                                            set result [yesorno "Do you want to disable PSU $PSU($intPSU)?"]
                                            logm ""
                                            if {$result == 5} {
                                                    # Execute Telecommand
                                       tcsend YC037942 checks {SPTV DPTV CEV} ack {ACCEPT } referby rYC037942 \
                                                {YP371942 1} \
                                            [list YP372942 [expr $PSU($intPSU) -1] ]
                                       logm " Sending Telecommand YC037942"

                                    # Control Execution
                                    logm " Synchronizing on SEV..."
                                    if { [waitfor -timeout 15000 { rYC037942 } -until { [getcompleted $rYC037942] } ]} {
                                                    logm " Synchronised on SEV for TC(s): YC037942"
                                    } else {
                                            logm " SEV synchronisation timed out for TC(s): YC037942"
                                    }
                                                    logm ""
                                                    logm " >>> Checking"
                                                    waittime 6.00
                                                    set Status_Master [getrawvalue [fetch $PSU_Master($PSU($intPSU))]]
                                                    set Status_Slave [getrawvalue [fetch $PSU_Slave($PSU($intPSU))]]
                                    logm " PSU $PSU($intPSU) Master status is currently $Status_Master (from $PSU_Master($PSU($intPSU)))"
                                    logm " PSU $PSU($intPSU) Slave status is currently $Status_Slave (from $PSU_Slave($PSU($intPSU)))"
                                    logm ""

                                                    # Do check
                                                    if {$Status_Master == 0 && $Status_Slave == 0} {
                                                            set check 1
                                                    } else {
                                                            set check 0
                                                    }

                                                    if {$check == 1} {
                                                            set result "0"
                                                            set result [infom "Check Successful! PSU $PSU($intPSU) has been disabled."]
                                                            logm ""
                                                            set intPSU [expr $intPSU + 1]
                                                    } else {
                                                            set result "0"
                                                            set result [yesorno "PSU $PSU($intPSU) has not been disabled. Repeat this
step?"]
                                                            logm ""
                                                            if {$result != 5} {
                                                                    set intPSU [expr $intPSU + 1]
                                                            }
                                                    }
                                            } else {
                                                    set intPSU [expr $intPSU + 1]
                                            }
                                    }
                                    set go_for_instrument "0"
                            }
                    }
            }


            ### Reading out PLM SCOE Settings
            logm ">>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>"
            logm ">>>>>>>>>  Reading out PLM SCOE Settings"
            logm ">>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>"
            logm ""

            set Status_PLM_OnLine [getrawvalue [fetch YM018942]]
```

```
logm "Status_PLM_OnLine is $Status_PLM_OnLine (extracted from TLM YM018942)"

set Status_PLM_PSU1_Master [getrawvalue [fetch YM129942]]
logm "Status_PLM_PSU1_Master is currently $Status_PLM_PSU1_Master (extracted from TLM YM129942)"

set Status_PLM_PSU1_Slave [getrawvalue [fetch YM145942]]
logm "Status_PLM_PSU1_Slave is currently $Status_PLM_PSU1_Slave (extracted from TLM YM145942)"

set Status_PLM_PSU2_Master [getrawvalue [fetch YM177942]]
logm "Status_PLM_PSU2_Master is currently $Status_PLM_PSU2_Master (extracted from TLM YM177942)"

set Status_PLM_PSU2_Slave [getrawvalue [fetch YM193942]]
logm "Status_PLM_PSU2_Slave is currently $Status_PLM_PSU2_Slave (extracted from TLM YM193942)"

set Status_PLM_LCL1_V [getrawvalue [fetch YM228942]]
logm "Status_PLM_LCL1_V is currently $Status_PLM_LCL1_V (extracted from TLM YM228942)"

set Status_PLM_LCL1_I [getrawvalue [fetch YM232942]]
logm "Status_PLM_LCL1_I is currently $Status_PLM_LCL1_I (extracted from TLM YM232942)"

set Status_PLM_LCL2_V [getrawvalue [fetch YM244942]]
logm "Status_PLM_LCL2_V is currently $Status_PLM_LCL2_V (extracted from TLM YM244942)"

set Status_PLM_LCL2_I [getrawvalue [fetch YM248942]]
logm "Status_PLM_LCL2_I is currently $Status_PLM_LCL2_I (extracted from TLM YM248942)"

set Status_PLM_LCL3_V [getrawvalue [fetch YM260942]]
logm "Status_PLM_LCL3_V is currently $Status_PLM_LCL3_V (extracted from TLM YM260942)"

set Status_PLM_LCL3_I [getrawvalue [fetch YM264942]]
logm "Status_PLM_LCL3_I is currently $Status_PLM_LCL3_I (extracted from TLM YM264942)"

set Status_PLM_LCL4_V [getrawvalue [fetch YM276942]]
logm "Status_PLM_LCL4_V is currently $Status_PLM_LCL4_V (extracted from TLM YM276942)"

set Status_PLM_LCL4_I [getrawvalue [fetch YM280942]]
logm "Status_PLM_LCL4_I is currently $Status_PLM_LCL4_I (extracted from TLM YM280942)"

set Status_PLM_LCL5_V [getrawvalue [fetch YM292942]]
logm "Status_PLM_LCL5_V is currently $Status_PLM_LCL5_V (extracted from TLM YM292942)"

set Status_PLM_LCL5_I [getrawvalue [fetch YM296942]]
logm "Status_PLM_LCL5_I is currently $Status_PLM_LCL5_I (extracted from TLM YM296942)"

set Status_PLM_LCL6_V [getrawvalue [fetch YM308942]]
logm "Status_PLM_LCL6_V is currently $Status_PLM_LCL6_V (extracted from TLM YM308942)"

set Status_PLM_LCL6_I [getrawvalue [fetch YM312942]]
logm "Status_PLM_LCL6_I is currently $Status_PLM_LCL6_I (extracted from TLM YM312942)"

set Status_PLM_LCL7_V [getrawvalue [fetch YM324942]]
logm "Status_PLM_LCL7_V is currently $Status_PLM_LCL7_V (extracted from TLM YM324942)"

set Status_PLM_LCL7_I [getrawvalue [fetch YM328942]]
logm "Status_PLM_LCL7_I is currently $Status_PLM_LCL7_I (extracted from TLM YM328942)"

set Status_PLM_LCL8_V [getrawvalue [fetch YM340942]]
logm "Status_PLM_LCL8_V is currently $Status_PLM_LCL8_V (extracted from TLM YM340942)"

set Status_PLM_LCL8_I [getrawvalue [fetch YM344942]]
logm "Status_PLM_LCL8_I is currently $Status_PLM_LCL8_I (extracted from TLM YM344942)"

set Status_PLM_LCL9_V [getrawvalue [fetch YM356942]]
logm "Status_PLM_LCL9_V is currently $Status_PLM_LCL9_V (extracted from TLM YM356942)"

set Status_PLM_LCL9_I [getrawvalue [fetch YM360942]]
logm "Status_PLM_LCL9_I is currently $Status_PLM_LCL9_I (extracted from TLM YM360942)"

set Status_PLM_LCL10_V [getrawvalue [fetch YM372942]]
logm "Status_PLM_LCL10_V is currently $Status_PLM_LCL10_V (extracted from TLM YM372942)"

set Status_PLM_LCL10_I [getrawvalue [fetch YM376942]]
logm "Status_PLM_LCL10_I is currently $Status_PLM_LCL10_I (extracted from TLM YM376942)"

set Status_PLM_LCL11_V [getrawvalue [fetch YM388942]]
logm "Status_PLM_LCL11_V is currently $Status_PLM_LCL11_V (extracted from TLM YM388942)"

set Status_PLM_LCL11_I [getrawvalue [fetch YM392942]]
logm "Status_PLM_LCL11_I is currently $Status_PLM_LCL11_I (extracted from TLM YM392942)"

set Status_PLM_LCL12_V [getrawvalue [fetch YM404942]]
logm "Status_PLM_LCL12_V is currently $Status_PLM_LCL12_V (extracted from TLM YM404942)"

set Status_PLM_LCL12_I [getrawvalue [fetch YM408942]]
logm "Status_PLM_LCL12_I is currently $Status_PLM_LCL12_I (extracted from TLM YM408942)"

set Status_PLM_LCL13_V [getrawvalue [fetch YM420942]]
logm "Status_PLM_LCL13_V is currently $Status_PLM_LCL13_V (extracted from TLM YM420942)"
```

```
set Status_PLM_LCL13_I [getrawvalue [fetch YM424942]]
logm "Status_PLM_LCL13_I is currently $Status_PLM_LCL13_I (extracted from TLM YM424942)"

set Status_PLM_LCL14_V [getrawvalue [fetch YM436942]]
logm "Status_PLM_LCL14_V is currently $Status_PLM_LCL14_V (extracted from TLM YM436942)"

set Status_PLM_LCL14_I [getrawvalue [fetch YM440942]]
logm "Status_PLM_LCL14_I is currently $Status_PLM_LCL14_I (extracted from TLM YM440942)"

#End
logm "<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<"
logm "<<<<<<<  Power Sequence Ended!"
logm "<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<"

finish_TS



######################################################################
# end of test sequence
######################################################################
# Changes:
# $Log: INSTR_POWER_OFF.tcl,v $
# Revision 1.5  2005/06/02 10:08:07  ilsens
# (No log message)
#
# Revision 1.4  2005/06/01 13:07:19  ilsens
# (No log message)
#
# Revision 1.3  2005/05/31 14:37:15  ilsens
# (No log message)
#
# Revision 1.2  2005/04/21 14:11:23  ilsens
# (No log message)
#
# Revision 1.1  2005/04/21 14:06:58  ilsens
# initial version
#
#
######################################################################
```

Appendix 5: Log file of PACS_POWER_ON.tcl

```
######################################################################
# File: $Id: PACS_POWER_ON.tcl,v 1.10 2005/05/31 14:34:37 ilsens Exp $
#
# Description:
#
#               This TCL script is used to automatically power on the PACS instrument
#
#
# Last edited by: $Author: ilsens $ on $Date: 2005/05/31 14:34:37 $.
#
#
######################################################################

# automatically set the revision. do not edit this
setrevision {$Id: PACS_POWER_ON.tcl,v 1.10 2005/05/31 14:34:37 ilsens Exp $}

######################################################################
# start of test sequence
######################################################################

# ---------------------------------------------------------------
# This first part defines some procedures to enable the GUI
# Please do not edit them
# The script is based on an adapted Alenia script
# using an interactive log.
# ---------------------------------------------------------------

global env
global go_on
global auto_path
global outfile

set go_on -1

lappend auto_path "/HPCCS/VARIABLE/CONFIG/TCL/Alenia"

# ------------------------------
# creation of the output file
# ------------------------------
gettime sec usec
set fn_time [clock format $sec -format %Y%m%d_%H%M%S -gmt true]
set fn_name [file tail [info script]]
set fn_name [string trim $fn_name ".tcl"]
set fn_add "GUI_log"
set fn_ID "XXXX"
append FileName $fn_time "_" $fn_ID "_" $fn_name "_" $fn_add ".txt"
putlog $FileName
set outfile [open $env(HPCCSTESTRES)/TSEQ/$FileName w]

# ---------------------------------------------------------------
# This second part contains the code to connect and attach to the
#             - CDMUDFE
#             - PLMSCOE
# ---------------------------------------------------------------

# This is the mandatory procedures to abtain the output log window
setup_win

logm ""
newTest_gen "Start of PACS POWER ON sequence."
logm ""

logm "To run this script, the CDMU DFE and PLM SCOE should be"
logm "powered and configured. "
logm "To initiate, this script will connect and attach to the CDMUDFE"
logm "and PLM SCOE."
logm ""

### Connect and Attach CDMU DFE
logm ">>> Connecting to CDMU DFE."
connect CDMUDFE
waittime +00.00.03.000000
logm ">>> Attaching to CDMU DFE."
attach CDMUDFE
waittime +00.00.03.000000
logm ""

### Connect and Attach PLM SCOE
logm ">>> Connecting to PLM SCOE."
connect PLMSCOE
waittime +00.00.03.000000
logm ">>> Attaching to PLM SCOE."
attach PLMSCOE
waittime +00.00.03.000000
```

```
logm ""

### Reading out CDMUDFE Settings
logm ">>> Reading out CDMUDFE Settings"
logm ""

set Status_CDMU_OnLine [getrawvalue [fetch YM777944]]
logm "Status_CDMU_OnLine is $Status_CDMU_OnLine (extracted from TLM YM777944)"

set Status_CDMU_TMpolling [getrawvalue [fetch YM780944]]
logm "Status_CDMU_TMpolling is $Status_CDMU_TMpolling (extracted from TLM YM780944)"

set Status_CDMU_SAreadActive [getrawvalue [fetch YM781944]]
logm "Status_CDMU_SAreadActive is $Status_CDMU_SAreadActive (extracted from TLM YM781944)"

set Status_CDMU_SAqueueActive [getrawvalue [fetch YM782944]]
logm "Status_CDMU_SAqueueActive is $Status_CDMU_SAqueueActive (extracted from TLM YM782944)"

set Status_CDMU_TMqueueActive [getrawvalue [fetch YM783944]]
logm "Status_CDMU_TMqueueActive is $Status_CDMU_TMqueueActive (extracted from TLM YM783944)"

set Status_CDMU_TCqueueActive [getrawvalue [fetch YM784944]]
logm "Status_CDMU_TCqueueActive is $Status_CDMU_TCqueueActive (extracted from TLM YM784944)"

set Status_CDMU_PSTfileName [getrawvalue [fetch YM809944]]
logm "Status_CDMU_PSTfileName is $Status_CDMU_PSTfileName (extracted from TLM YM809944)"

set Status_CDMU_PSTrunning [getrawvalue [fetch YM829944]]
logm "Status_CDMU_PSTrunning is $Status_CDMU_PSTrunning (extracted from TLM YM829944)"
logm ""

### Reading out PLM SCOE Settings
logm ">>> Reading out PLM SCOE Settings"
logm ""

set Status_PLM_OnLine [getrawvalue [fetch YM018942]]
logm "Status_PLM_OnLine is $Status_PLM_OnLine (extracted from TLM YM018942)"

set Status_PLM_PSU1_Master [getrawvalue [fetch YM129942]]
logm "Status_PLM_PSU1_Master is currently $Status_PLM_PSU1_Master (extracted from TLM YM129942)"

set Status_PLM_PSU1_Slave [getrawvalue [fetch YM145942]]
logm "Status_PLM_PSU1_Slave is currently $Status_PLM_PSU1_Slave (extracted from TLM YM145942)"

set Status_PLM_PSU2_Master [getrawvalue [fetch YM177942]]
logm "Status_PLM_PSU2_Master is currently $Status_PLM_PSU2_Master (extracted from TLM YM177942)"

set Status_PLM_PSU2_Slave [getrawvalue [fetch YM193942]]
logm "Status_PLM_PSU2_Slave is currently $Status_PLM_PSU2_Slave (extracted from TLM YM193942)"

set Status_PLM_LCL1_V [getrawvalue [fetch YM228942]]
logm "Status_PLM_LCL1_V is currently $Status_PLM_LCL1_V (extracted from TLM YM228942)"

set Status_PLM_LCL1_I [getrawvalue [fetch YM232942]]
logm "Status_PLM_LCL1_I is currently $Status_PLM_LCL1_I (extracted from TLM YM232942)"

set Status_PLM_LCL2_V [getrawvalue [fetch YM244942]]
logm "Status_PLM_LCL2_V is currently $Status_PLM_LCL2_V (extracted from TLM YM244942)"

set Status_PLM_LCL2_I [getrawvalue [fetch YM248942]]
logm "Status_PLM_LCL2_I is currently $Status_PLM_LCL2_I (extracted from TLM YM248942)"

set Status_PLM_LCL3_V [getrawvalue [fetch YM260942]]
logm "Status_PLM_LCL3_V is currently $Status_PLM_LCL3_V (extracted from TLM YM260942)"

set Status_PLM_LCL3_I [getrawvalue [fetch YM264942]]
logm "Status_PLM_LCL3_I is currently $Status_PLM_LCL3_I (extracted from TLM YM264942)"

set Status_PLM_LCL4_V [getrawvalue [fetch YM276942]]
logm "Status_PLM_LCL4_V is currently $Status_PLM_LCL4_V (extracted from TLM YM276942)"

set Status_PLM_LCL4_I [getrawvalue [fetch YM280942]]
logm "Status_PLM_LCL4_I is currently $Status_PLM_LCL4_I (extracted from TLM YM280942)"

set Status_PLM_LCL5_V [getrawvalue [fetch YM292942]]
logm "Status_PLM_LCL5_V is currently $Status_PLM_LCL5_V (extracted from TLM YM292942)"

set Status_PLM_LCL5_I [getrawvalue [fetch YM296942]]
logm "Status_PLM_LCL5_I is currently $Status_PLM_LCL5_I (extracted from TLM YM296942)"

set Status_PLM_LCL6_V [getrawvalue [fetch YM308942]]
logm "Status_PLM_LCL6_V is currently $Status_PLM_LCL6_V (extracted from TLM YM308942)"

set Status_PLM_LCL6_I [getrawvalue [fetch YM312942]]
logm "Status_PLM_LCL6_I is currently $Status_PLM_LCL6_I (extracted from TLM YM312942)"

set Status_PLM_LCL7_V [getrawvalue [fetch YM324942]]
logm "Status_PLM_LCL7_V is currently $Status_PLM_LCL7_V (extracted from TLM YM324942)"
```

```
set Status_PLM_LCL7_I [getrawvalue [fetch YM328942]]
logm "Status_PLM_LCL7_I is currently $Status_PLM_LCL7_I (extracted from TLM YM328942)"

set Status_PLM_LCL8_V [getrawvalue [fetch YM340942]]
logm "Status_PLM_LCL8_V is currently $Status_PLM_LCL8_V (extracted from TLM YM340942)"

set Status_PLM_LCL8_I [getrawvalue [fetch YM344942]]
logm "Status_PLM_LCL8_I is currently $Status_PLM_LCL8_I (extracted from TLM YM344942)"

set Status_PLM_LCL9_V [getrawvalue [fetch YM356942]]
logm "Status_PLM_LCL9_V is currently $Status_PLM_LCL9_V (extracted from TLM YM356942)"

set Status_PLM_LCL9_I [getrawvalue [fetch YM360942]]
logm "Status_PLM_LCL9_I is currently $Status_PLM_LCL9_I (extracted from TLM YM360942)"

set Status_PLM_LCL10_V [getrawvalue [fetch YM372942]]
logm "Status_PLM_LCL10_V is currently $Status_PLM_LCL10_V (extracted from TLM YM372942)"

set Status_PLM_LCL10_I [getrawvalue [fetch YM376942]]
logm "Status_PLM_LCL10_I is currently $Status_PLM_LCL10_I (extracted from TLM YM376942)"

set Status_PLM_LCL11_V [getrawvalue [fetch YM388942]]
logm "Status_PLM_LCL11_V is currently $Status_PLM_LCL11_V (extracted from TLM YM388942)"

set Status_PLM_LCL11_I [getrawvalue [fetch YM392942]]
logm "Status_PLM_LCL11_I is currently $Status_PLM_LCL11_I (extracted from TLM YM392942)"

set Status_PLM_LCL12_V [getrawvalue [fetch YM404942]]
logm "Status_PLM_LCL12_V is currently $Status_PLM_LCL12_V (extracted from TLM YM404942)"

set Status_PLM_LCL12_I [getrawvalue [fetch YM408942]]
logm "Status_PLM_LCL12_I is currently $Status_PLM_LCL12_I (extracted from TLM YM408942)"

set Status_PLM_LCL13_V [getrawvalue [fetch YM420942]]
logm "Status_PLM_LCL13_V is currently $Status_PLM_LCL13_V (extracted from TLM YM420942)"

set Status_PLM_LCL13_I [getrawvalue [fetch YM424942]]
logm "Status_PLM_LCL13_I is currently $Status_PLM_LCL13_I (extracted from TLM YM424942)"

set Status_PLM_LCL14_V [getrawvalue [fetch YM436942]]
logm "Status_PLM_LCL14_V is currently $Status_PLM_LCL14_V (extracted from TLM YM436942)"

set Status_PLM_LCL14_I [getrawvalue [fetch YM440942]]
logm "Status_PLM_LCL14_I is currently $Status_PLM_LCL14_I (extracted from TLM YM440942)"
logm ""

# ----------------------------------------------------------------
# This third part contains the code to enable the necessary PSU
# ----------------------------------------------------------------

### Switch on PSU's (Power Supply Units on PLM SCOE)
logm ">>> Switch ON PSU(s)"
logm ""

set LCL_name(11) "PACS BOLC"
set LCL_name(12) "PACS MEC"
set LCL_name(13) "PACS DPU"
set LCL_name(14) "PACS SPU"
set LCL_V(1) "YM228942"
set LCL_I(1) "YM232942"
set LCL_V(2) "YM244942"
set LCL_I(2) "YM248942"
set LCL_V(3) "YM260942"
set LCL_I(3) "YM264942"
set LCL_V(4) "YM276942"
set LCL_I(4) "YM280942"
set LCL_V(5) "YM292942"
set LCL_I(5) "YM296942"
set LCL_V(6) "YM308942"
set LCL_I(6) "YM312942"
set LCL_V(7) "YM324942"
set LCL_I(7) "YM328942"
set LCL_V(8) "YM340942"
set LCL_I(8) "YM344942"
set LCL_V(9) "YM356942"
set LCL_I(9) "YM360942"
set LCL_V(10) "YM372942"
set LCL_I(10) "YM376942"
set LCL_V(11) "YM388942"
set LCL_I(11) "YM392942"
set LCL_V(12) "YM404942"
set LCL_I(12) "YM408942"
set LCL_V(13) "YM420942"
set LCL_I(13) "YM424942"
set LCL_V(14) "YM436942"
set LCL_I(14) "YM440942"
```

```
set PSU_Master(1) "YM129942"
set PSU_Slave(1) "YM145942"
set PSU_Master(2) "YM177942"
set PSU_Slave(2) "YM193942"

# Execute Telecommand
tcsend YC036942 checks {SPTV DPTV CEV} ack {ACCEPT } referby rYC036942 \
        {YP361942 1} {YP362942 1}
logm ">>> Sending Telecommand YC036942"
logm ""
logm ">>> Checking"
waittime 6.00
set Status_Master [getrawvalue [fetch $PSU_Master(2)]]
set Status_Slave [getrawvalue [fetch $PSU_Slave(2)]]
logm "PSU 2 Master status is currently $Status_Master (from $PSU_Master(2))"
logm "PSU 2 Slave status is currently $Status_Slave (from $PSU_Slave(2))"
logm ""


# ---------------------------------------------------------------
# This part contains the code to switch on the DPU
# ---------------------------------------------------------------

logm ">>> Switch ON DPU"
logm ""

# Execute Telecommand
tcsend YC040942 checks {SPTV DPTV CEV} ack NONE referby rYC040942 \
        {YP401942 1} {YP402942 13}
logm ">>> Sending Telecommand YC040942 to Enable Limiter 13 -> PACS DPU"
logm ""

# Execute Telecommand
tcsend YC043942 checks {SPTV DPTV CEV} ack NONE referby rYC043942 \
        {YP431942 1} {YP432942 13} {YP433942 2}
logm ">>> Sending Telecommand YC043942 to Set Limiter 13 -> PACS DPU"
logm ""

logm ">>> Checking"
waittime 6.00
set Status_Voltage [getrawvalue [fetch $LCL_V(13)]]
set Status_Current [getrawvalue [fetch $LCL_I(13)]]
logm "LCL 13 has currently a voltage of $Status_Voltage.(from $LCL_V(13))"
logm "LCL 13 has currently a current of $Status_Current.(from $LCL_I(13))"
logm ""

# ---------------------------------------------------------------
# This part contains the code extracted from the PACS power-on script (after DPU start)
# The code is adapted in such a way that the interactive log shows what's
# happening in the tCL (logm "xxx")
# ---------------------------------------------------------------

waittime 12

# wait for TEI Jitter
waittime 5

# DPU Force Boot
logm "Force Boot DPU"
tcsend PC032380
waittime 1

set result "0"
set result [infom "Please check if the force boot has been executed correctly and press OK."]
logm ""
logm ""

# ---------------------------------------------------------------
# This part contains the code to switch on the DECMEC
# ---------------------------------------------------------------

logm ">>> Switch ON DEC/MEC"
logm ""

# Execute Telecommand
tcsend YC040942 checks {SPTV DPTV CEV} ack NONE referby rYC040942 \
        {YP401942 1} {YP402942 12}
logm ">>> Sending Telecommand YC040942 to Enable Limiter 12 -> PACS DEC/MEC"
logm ""

# Execute Telecommand
tcsend YC043942 checks {SPTV DPTV CEV} ack NONE referby rYC043942 \
        {YP431942 1} {YP432942 12} {YP433942 2}
logm ">>> Sending Telecommand YC043942 to Set Limiter 12 -> PACS DEC/MEC"
logm ""

logm ">>> Checking"
waittime 6.00
```

```
set Status_Voltage [getrawvalue [fetch $LCL_V(12)]]
set Status_Current [getrawvalue [fetch $LCL_I(12)]]
logm "LCL 12 has currently a voltage of $Status_Voltage.(from $LCL_V(12))"
logm "LCL 12 has currently a current of $Status_Current.(from $LCL_I(12))"
logm ""

# ----------------------------------------------------------------
# This part contains the code extracted from the PACS power-on script (after DMC start)
# The code is adapted in such a way that the interactive log shows what's
# happening in the tCL (logm "xxx")
# ----------------------------------------------------------------

waittime 15

# wait for TEI Jitter
waittime 5

#DPU reset of 1355
logm "DPU reset of 1355"
tcsend PC025380
waittime 2

# Establish DPU --> DMC connection (DPU-START-OBCP, n=19)
logm "Establish DPU --> DMC connection (DPU-START-OBCP, n=19)"
tcsend PC012380 {PP012380 19} {PP010380 2} \
            {PP011380 1}  {PP017380 0} \
            {PP011380 2}  {PP017380 1}
waittime 4

#Copy DMC SW from EEPROM to RAM
#DMC_LLSW_LOAD_EEPROM

logm "Copy DMC SW from EEPROM to RAM"

tcsend PC198420  {PP087420 0x3 LO} {PP089420 0x0000 LO} \
             {PP088420 0x1 LO} {PP090420 0x6EE00 LO} \
             {PP091420 0x4000 LO}
waittime 2

logm "DMC_LLSW_LOAD_EEPROM"

tcsend PC198420  {PP087420 0x3 LO} {PP089420 0x8000 LO} \
             {PP088420 0x1 LO} {PP090420 0x8000 LO} \
             {PP091420 0x8000 LO}
waittime 2

#Start  DMC HLSW  See DPU UM
#OBCP number 21
#DPU-START-OBCP  procedureID   num of times
#              paramID      param value
#              paramID     param value

logm "Start DMC HLSW"

tcsend PC012380 {PP012380 21} {PP010380 3} \
            {PP011380 1}  {PP017380 0} \
                          {PP011380 2}  {PP017380 0x01 LO} \
                          {PP011380 3}  {PP017380 0x8032 LO}
waittime 10

#tcsend PC013380 {PP012380 21}
waittime 0.5

# Establish DPU --> DMC (DPU-START-OBCP, n=19)
#puts stdout "DPU starts link with DMC with DPU as slave"

logm "DPU starts link with DMC with DPU as slave"

tcsend PC012380 {PP012380 19} {PP010380 2} \
            {PP011380 1}  {PP017380 0} \
            {PP011380 2}  {PP017380 2}
waittime 3

logm ""
logm ""

# ----------------------------------------------------------------
# This part contains the code to switch on the BOLC
# ----------------------------------------------------------------

logm ">>> Switch ON BOLC"
logm ""

# Execute Telecommand
tcsend YC040942 checks {SPTV DPTV CEV} ack NONE referby rYC040942 \
             {YP401942 1} {YP402942 11}
logm ">>> Sending Telecommand YC040942 to Enable Limiter 11 -> PACS BOLC"
logm ""
```

```
# Execute Telecommand
tcsend YC043942 checks {SPTV DPTV CEV} ack NONE referby rYC043942 \
            {YP431942 1} {YP432942 11} {YP433942 2}
logm ">>> Sending Telecommand YC043942 to Set Limiter 11 -> PACS BOLC"
logm ""

logm ">>> Checking"
waittime 6.00
set Status_Voltage [getrawvalue [fetch $LCL_V(11)]]
set Status_Current [getrawvalue [fetch $LCL_I(11)]]
logm "LCL 11 has currently a voltage of $Status_Voltage.(from $LCL_V(11))"
logm "LCL 11 has currently a current of $Status_Current.(from $LCL_I(11))"
logm ""

# ----------------------------------------------------------------
# This part contains the code extracted from the PACS power-on script (after BOLC start)
# The code is adapted in such a way that the interactive log shows what's
# happening in the tCL (logm "xxx")
# ----------------------------------------------------------------

waittime 10

# wait for TEI Jitter
waittime 5

# DMC_RESET_SMCS_CHIP_2
logm "DMC_RESET_SMCS_CHIP_2"
tcsend PC202420
waittime 4

# Execute BOLC initialisation including frequency setting

logm "Execute BOLC initialisation including frequency setting"

# For CQM we only have groups 1,3,5,6. The corresponding bits
# are (2^0+2^2+2^4+2^5)=53, which is 35 in hex.
tcsend PC103420 {PP071420 0x0A000035 LO}
waittime 0.5

# All temperature sensors are set with "070000FF" only once cold !!
tcsend PC103420 {PP071420 0x070000FF LO}
waittime 0.5

waittime 5

logm "set image frequence to 20 Hz"

# set image frequence to 20 Hz
tcsend PC103420 {PP071420 0x0B020020 LO}
waittime 0.5

logm ""
logm ""

# ----------------------------------------------------------------
# This part contains the code to switch on the SPU
# ----------------------------------------------------------------

logm ">>> Switch ON SPU"
logm ""

# Execute Telecommand
tcsend YC040942 checks {SPTV DPTV CEV} ack NONE referby rYC040942 \
            {YP401942 1} {YP402942 14}
logm ">>> Sending Telecommand YC040942 to Enable Limiter 14 -> PACS SPU"
logm ""

# Execute Telecommand
tcsend YC043942 checks {SPTV DPTV CEV} ack NONE referby rYC043942 \
            {YP431942 1} {YP432942 14} {YP433942 2}
logm ">>> Sending Telecommand YC043942 to Set Limiter 14 -> PACS SPU"
logm ""

logm ">>> Checking"
waittime 6.00
set Status_Voltage [getrawvalue [fetch $LCL_V(14)]]
set Status_Current [getrawvalue [fetch $LCL_I(14)]]
logm "LCL 14 has currently a voltage of $Status_Voltage.(from $LCL_V(14))"
logm "LCL 14 has currently a current of $Status_Current.(from $LCL_I(14))"
logm ""

# ----------------------------------------------------------------
# This part contains the code extracted from the PACS power-on script (after SPUC start)
# The code is adapted in such a way that the interactive log shows what's
# happening in the tCL (logm "xxx")
# ----------------------------------------------------------------
```

```
waittime 15

# wait for TEI Jitter
waittime 5

#DPU reset of 1355
logm "DPU reset of 1355"
tcsend PC025380
waittime 4

# Establish DPU --> DMC (DPU-START-OBCP, n=19)
logm "DPU starts link with DMC with DPU as slave"
tcsend PC012380 {PP012380 19} {PP010380 2} \
            {PP011380 1}  {PP017380 0} \
            {PP011380 2}  {PP017380 2}
waittime 10

# Establish DPU --> blue SPU links (DPU-START-OBCP, n=19)
#DPU-START-OBCP  procedureID   num of times
#              paramID      param value
#              paramID     param value
logm "DPU starts link with (blue) SPUS with DPU as master"
tcsend PC012380 {PP012380 19} {PP010380 2} \
            {PP011380 1}  {PP017380 1} \
                          {PP011380 2}  {PP017380 1}
waittime 4

# Establish DPU --> red SPU links (DPU-START-OBCP, n=19)
#DPU-START-OBCP  procedureID   num of times
#              paramID      param value
#              paramID     param value
logm "DPU starts link with (red) SPUL with DPU as master"
tcsend PC012380 {PP012380 19} {PP010380 2} \
            {PP011380 1}  {PP017380 2} \
                          {PP011380 2}  {PP017380 1}
waittime 4

logm "LOAD SPU RED HLSW FROM EEPROM TO RAM"
#LOAD SPU RED HLSW FROM EEPROM TO RAM first chunk
# SEE SPU HLSW UM V.3.1 page 26
tcsend PC070390  {PP040390 0x03 LO} {PP042390 0x100 LO} \
             {PP041390 0x01 LO} {PP043390 0x100 LO} \
             {PP038390 0x1e0 LO}
waittime 2

#LOAD SPU RED HLSW FROM EEPROM TO RAM second chunk
tcsend PC070390 {PP040390 0x03 LO} {PP042390 0x300 LO} \
            {PP041390 0x01 LO} {PP043390 0x300 LO} \
                          {PP038390 0x700 LO}
waittime 2

#LOAD SPU RED HLSW FROM EEPROM TO RAM Third chunk
tcsend PC070390 {PP040390 0x03 LO} {PP042390 0xa00 LO} \
            {PP041390 0x01 LO} {PP043390 0xa00 LO} \
                          {PP038390 0xa600 LO}
waittime 2

logm "LOAD SPU BLUE HLSW FROM EEPROM TO RAM"
#LOAD SPU BLUE HLSW FROM EEPROM TO RAM first chunk
# SEE SPU HLSW UM V.3.1 page 26
tcsend PC069400  {PP058400 0x03 LO} {PP060400 0x100 LO}  \
             {PP059400 0x01 LO} {PP061400 0x100 LO} \
                          {PP056400 0x1e0 LO}
waittime 2

#LOAD SPU BLUE HLSW FROM EEPROM TO RAM second chunk
tcsend PC069400 {PP058400 0x03 LO} {PP060400 0x300 LO} \
            {PP059400 0x01 LO} {PP061400 0x300 LO} \
                          {PP056400 0x700 LO}
waittime 2

#LOAD SPU BLUE HLSW FROM EEPROM TO RAM Third chunk
tcsend PC069400 {PP058400 0x03 LO} {PP060400 0xa00 LO} \
            {PP059400 0x01 LO} {PP061400 0xa00 LO} \
                          {PP056400 0xa600 LO}
waittime 4

#Start  SPUS HLSW  See DPU UM Version1.3 page 68 (App. B )
#OBCP number 21       : number of parameters = 3
#DPU-START-OBCP  procedureID   num of times
#              paramID      param value
#              paramID     param value
logm "Start SPUS HLSW"
tcsend PC012380 {PP012380 21} {PP010380 3} \
            {PP011380 1}  {PP017380 1} \
                          {PP011380 2}  {PP017380 0x01 LO} \
                          {PP011380 3}  {PP017380 0xa02 LO}
waittime 3
```

```
# Establish DPU --> blue SPU links (DPU-START-OBCP)
#DPU-START-OBCP  procedureID   num of times
#             paramID      param value
#             paramID     param value
logm "DPU starts link with (blue) SPUS with DPU as slave"
tcsend PC012380 {PP012380 19} {PP010380 2} \
           {PP011380 1}  {PP017380 1} \
                         {PP011380 2}  {PP017380 2}
waittime 4

#Start  SPUL HLSW  See DPU UM Version1.3 page 68 (App. B )
#OBCP number 21       : number of parameters = 3
#DPU-START-OBCP  procedureID   num of times
#DPU-START-OBCP  procedureID   num of times
#             paramID      param value
#             paramID     param value
logm "Start SPUL HLSW"
tcsend PC012380 {PP012380 21} {PP010380 3} \
          {PP011380 1}  {PP017380 2} \
                         {PP011380 2}  {PP017380  0x01 LO} \
                         {PP011380 3}  {PP017380  0xa02 LO}
waittime 3

# Establish DPU --> red SPU links (DPU-START-OBCP)
#DPU-START-OBCP  procedureID   num of times
#             paramID      param value
#             paramID     param value
logm "DPU starts link with (red) SPUL with DPU as slave"
tcsend PC012380 {PP012380 19} {PP010380 2} \
           {PP011380 1}  {PP017380 2} \
                         {PP011380 2}  {PP017380 2}
waittime 5

# Establish connection SPUL-DMC, DMC as master
#SPUL_CONNECT_DMC
logm "Establish connection SPUL-DMC, DMC as master"
tcsend PC195390 {PP084390 0x22 LO}
waittime 1

# Establish connection SPUS-DMC, DMC as master
#SPUS_CONNECT_DMC
logm "Establish connection SPUS-DMC, DMC as master"
tcsend PC194400 {PP085400 0x22 LO}
waittime 2

#Establish connection DMC-SPURS DMC Master
#DMC_START_BLE_SPU_LINK
logm "Establish connection DMC-SPURS DMC Master"
tcsend PC197420 {PP086420 1}
waittime 1

#Establish connection DMC-SPURL DMC Master
#DMC_START_RED_SPU_LINK
logm "Establish connection DMC-SPURL DMC Master"
tcsend PC196420 {PP086420 1}
waittime 2

# DMC_SWON_TEMP_SENSORS
tcsend PC210420
waittime 0.5
logm "FPU T-sensors are activated"
logm ""
logm ""

### Reading out CDMUDFE Settings
logm ">>> Reading out CDMUDFE Settings"
logm ""

set Status_CDMU_OnLine [getrawvalue [fetch YM777944]]
logm "Status_CDMU_OnLine is $Status_CDMU_OnLine (extracted from TLM YM777944)"

set Status_CDMU_TMpolling [getrawvalue [fetch YM780944]]
logm "Status_CDMU_TMpolling is $Status_CDMU_TMpolling (extracted from TLM YM780944)"

set Status_CDMU_SAreadActive [getrawvalue [fetch YM781944]]
logm "Status_CDMU_SAreadActive is $Status_CDMU_SAreadActive (extracted from TLM YM781944)"

set Status_CDMU_SAqueueActive [getrawvalue [fetch YM782944]]
logm "Status_CDMU_SAqueueActive is $Status_CDMU_SAqueueActive (extracted from TLM YM782944)"

set Status_CDMU_TMqueueActive [getrawvalue [fetch YM783944]]
logm "Status_CDMU_TMqueueActive is $Status_CDMU_TMqueueActive (extracted from TLM YM783944)"

set Status_CDMU_TCqueueActive [getrawvalue [fetch YM784944]]
logm "Status_CDMU_TCqueueActive is $Status_CDMU_TCqueueActive (extracted from TLM YM784944)"

set Status_CDMU_PSTfileName [getrawvalue [fetch YM809944]]
```

```
logm "Status_CDMU_PSTfileName is $Status_CDMU_PSTfileName (extracted from TLM YM809944)"

set Status_CDMU_PSTrunning [getrawvalue [fetch YM829944]]
logm "Status_CDMU_PSTrunning is $Status_CDMU_PSTrunning (extracted from TLM YM829944)"
logm ""

### Reading out PLM SCOE Settings
logm ">>> Reading out PLM SCOE Settings"
logm ""

set Status_PLM_OnLine [getrawvalue [fetch YM018942]]
logm "Status_PLM_OnLine is $Status_PLM_OnLine (extracted from TLM YM018942)"

set Status_PLM_PSU1_Master [getrawvalue [fetch YM129942]]
logm "Status_PLM_PSU1_Master is currently $Status_PLM_PSU1_Master (extracted from TLM YM129942)"

set Status_PLM_PSU1_Slave [getrawvalue [fetch YM145942]]
logm "Status_PLM_PSU1_Slave is currently $Status_PLM_PSU1_Slave (extracted from TLM YM145942)"

set Status_PLM_PSU2_Master [getrawvalue [fetch YM177942]]
logm "Status_PLM_PSU2_Master is currently $Status_PLM_PSU2_Master (extracted from TLM YM177942)"

set Status_PLM_PSU2_Slave [getrawvalue [fetch YM193942]]
logm "Status_PLM_PSU2_Slave is currently $Status_PLM_PSU2_Slave (extracted from TLM YM193942)"

set Status_PLM_LCL1_V [getrawvalue [fetch YM228942]]
logm "Status_PLM_LCL1_V is currently $Status_PLM_LCL1_V (extracted from TLM YM228942)"

set Status_PLM_LCL1_I [getrawvalue [fetch YM232942]]
logm "Status_PLM_LCL1_I is currently $Status_PLM_LCL1_I (extracted from TLM YM232942)"

set Status_PLM_LCL2_V [getrawvalue [fetch YM244942]]
logm "Status_PLM_LCL2_V is currently $Status_PLM_LCL2_V (extracted from TLM YM244942)"

set Status_PLM_LCL2_I [getrawvalue [fetch YM248942]]
logm "Status_PLM_LCL2_I is currently $Status_PLM_LCL2_I (extracted from TLM YM248942)"

set Status_PLM_LCL3_V [getrawvalue [fetch YM260942]]
logm "Status_PLM_LCL3_V is currently $Status_PLM_LCL3_V (extracted from TLM YM260942)"

set Status_PLM_LCL3_I [getrawvalue [fetch YM264942]]
logm "Status_PLM_LCL3_I is currently $Status_PLM_LCL3_I (extracted from TLM YM264942)"

set Status_PLM_LCL4_V [getrawvalue [fetch YM276942]]
logm "Status_PLM_LCL4_V is currently $Status_PLM_LCL4_V (extracted from TLM YM276942)"

set Status_PLM_LCL4_I [getrawvalue [fetch YM280942]]
logm "Status_PLM_LCL4_I is currently $Status_PLM_LCL4_I (extracted from TLM YM280942)"

set Status_PLM_LCL5_V [getrawvalue [fetch YM292942]]
logm "Status_PLM_LCL5_V is currently $Status_PLM_LCL5_V (extracted from TLM YM292942)"

set Status_PLM_LCL5_I [getrawvalue [fetch YM296942]]
logm "Status_PLM_LCL5_I is currently $Status_PLM_LCL5_I (extracted from TLM YM296942)"

set Status_PLM_LCL6_V [getrawvalue [fetch YM308942]]
logm "Status_PLM_LCL6_V is currently $Status_PLM_LCL6_V (extracted from TLM YM308942)"

set Status_PLM_LCL6_I [getrawvalue [fetch YM312942]]
logm "Status_PLM_LCL6_I is currently $Status_PLM_LCL6_I (extracted from TLM YM312942)"

set Status_PLM_LCL7_V [getrawvalue [fetch YM324942]]
logm "Status_PLM_LCL7_V is currently $Status_PLM_LCL7_V (extracted from TLM YM324942)"

set Status_PLM_LCL7_I [getrawvalue [fetch YM328942]]
logm "Status_PLM_LCL7_I is currently $Status_PLM_LCL7_I (extracted from TLM YM328942)"

set Status_PLM_LCL8_V [getrawvalue [fetch YM340942]]
logm "Status_PLM_LCL8_V is currently $Status_PLM_LCL8_V (extracted from TLM YM340942)"

set Status_PLM_LCL8_I [getrawvalue [fetch YM344942]]
logm "Status_PLM_LCL8_I is currently $Status_PLM_LCL8_I (extracted from TLM YM344942)"

set Status_PLM_LCL9_V [getrawvalue [fetch YM356942]]
logm "Status_PLM_LCL9_V is currently $Status_PLM_LCL9_V (extracted from TLM YM356942)"

set Status_PLM_LCL9_I [getrawvalue [fetch YM360942]]
logm "Status_PLM_LCL9_I is currently $Status_PLM_LCL9_I (extracted from TLM YM360942)"

set Status_PLM_LCL10_V [getrawvalue [fetch YM372942]]
logm "Status_PLM_LCL10_V is currently $Status_PLM_LCL10_V (extracted from TLM YM372942)"

set Status_PLM_LCL10_I [getrawvalue [fetch YM376942]]
logm "Status_PLM_LCL10_I is currently $Status_PLM_LCL10_I (extracted from TLM YM376942)"

set Status_PLM_LCL11_V [getrawvalue [fetch YM388942]]
logm "Status_PLM_LCL11_V is currently $Status_PLM_LCL11_V (extracted from TLM YM388942)"
```

```
set Status_PLM_LCL11_I [getrawvalue [fetch YM392942]]
logm "Status_PLM_LCL11_I is currently $Status_PLM_LCL11_I (extracted from TLM YM392942)"

set Status_PLM_LCL12_V [getrawvalue [fetch YM404942]]
logm "Status_PLM_LCL12_V is currently $Status_PLM_LCL12_V (extracted from TLM YM404942)"

set Status_PLM_LCL12_I [getrawvalue [fetch YM408942]]
logm "Status_PLM_LCL12_I is currently $Status_PLM_LCL12_I (extracted from TLM YM408942)"

set Status_PLM_LCL13_V [getrawvalue [fetch YM420942]]
logm "Status_PLM_LCL13_V is currently $Status_PLM_LCL13_V (extracted from TLM YM420942)"

set Status_PLM_LCL13_I [getrawvalue [fetch YM424942]]
logm "Status_PLM_LCL13_I is currently $Status_PLM_LCL13_I (extracted from TLM YM424942)"

set Status_PLM_LCL14_V [getrawvalue [fetch YM436942]]
logm "Status_PLM_LCL14_V is currently $Status_PLM_LCL14_V (extracted from TLM YM436942)"

set Status_PLM_LCL14_I [getrawvalue [fetch YM440942]]
logm "Status_PLM_LCL14_I is currently $Status_PLM_LCL14_I (extracted from TLM YM440942)"
logm ""

logm ""
newTest_gen "PACS Power On Sequence has ended"
logm ""

finish_TS

##########################################################################
# end of test sequence
##########################################################################
# Changes:
# $Log: PACS_POWER_ON.tcl,v $
# Revision 1.10  2005/05/31 14:34:37  ilsens
# File created by manual import
#
# Revision 1.6  2005/05/23 08:30:13  ilsens
# (No log message)
#
# Revision 1.5  2005/05/20 13:56:29  ilsens
# (No log message)
#
# Revision 1.4  2005/05/20 13:25:00  ilsens
# (No log message)
#
# Revision 1.3  2005/05/20 13:22:58  ilsens
# initial version
#
#
##########################################################################
```

Appendix 6: Log file of PACS_POWER_OFF.tcl

```
######################################################################
# File: $Id: PACS_POWER_OFF.tcl,v 1.6 2005/05/31 14:34:24 ilsens Exp $
#
# Description:
#
#               This TCL script is used to automatically power down the PACS instrument
#
#
# Last edited by: $Author: ilsens $ on $Date: 2005/05/31 14:34:24 $.
#
#
######################################################################

# automatically set the revision. do not edit this
setrevision {$Id: PACS_POWER_OFF.tcl,v 1.6 2005/05/31 14:34:24 ilsens Exp $}

######################################################################
# start of test sequence
######################################################################


# ---------------------------------------------------------------
# This first part defines some procedures to enable the GUI
# Please do not edit them
# The script is based on an adapted Alenia script
# using an interactive log.
# ---------------------------------------------------------------

global env
global go_on
global auto_path
global outfile

set go_on -1

lappend auto_path "/HPCCS/VARIABLE/CONFIG/TCL/Alenia"

# ------------------------------
# creation of the output file
# ------------------------------
gettime sec usec
set fn_time [clock format $sec -format %Y%m%d_%H%M%S -gmt true]
set fn_name [file tail [info script]]
set fn_name [string trim $fn_name ".tcl"]
set fn_add "GUI_log"
set fn_ID "XXXX"
append FileName $fn_time "_" $fn_ID "_" $fn_name "_" $fn_add ".txt"
putlog $FileName
set outfile [open $env(HPCCSTESTRES)/TSEQ/$FileName w]

# This is the mandatory procedures to abtain the output log window
setup_win

logm ""
newTest_gen "Start of PACS POWER ON sequence."
logm ""

logm "To run this script, the CDMU DFE and PLM SCOE should be"
logm "powered and configured. "
logm "To initiate, this script will connect and attach to the CDMUDFE"
logm "and PLM SCOE."
logm ""

### Connect and Attach CDMU DFE
logm ">>> Connecting to CDMU DFE."
connect CDMUDFE
waittime +00.00.03.000000
logm ">>> Attaching to CDMU DFE."
attach CDMUDFE
waittime +00.00.03.000000
logm ""

### Connect and Attach PLM SCOE
logm ">>> Connecting to PLM SCOE."
connect PLMSCOE
waittime +00.00.03.000000
logm ">>> Attaching to PLM SCOE."
attach PLMSCOE
waittime +00.00.03.000000
logm ""

### Reading out CDMUDFE Settings
logm ">>> Reading out CDMUDFE Settings"
logm ""
```

```
set Status_CDMU_OnLine [getrawvalue [fetch YM777944]]
logm "Status_CDMU_OnLine is $Status_CDMU_OnLine (extracted from TLM YM777944)"

set Status_CDMU_TMpolling [getrawvalue [fetch YM780944]]
logm "Status_CDMU_TMpolling is $Status_CDMU_TMpolling (extracted from TLM YM780944)"

set Status_CDMU_SAreadActive [getrawvalue [fetch YM781944]]
logm "Status_CDMU_SAreadActive is $Status_CDMU_SAreadActive (extracted from TLM YM781944)"

set Status_CDMU_SAqueueActive [getrawvalue [fetch YM782944]]
logm "Status_CDMU_SAqueueActive is $Status_CDMU_SAqueueActive (extracted from TLM YM782944)"

set Status_CDMU_TMqueueActive [getrawvalue [fetch YM783944]]
logm "Status_CDMU_TMqueueActive is $Status_CDMU_TMqueueActive (extracted from TLM YM783944)"

set Status_CDMU_TCqueueActive [getrawvalue [fetch YM784944]]
logm "Status_CDMU_TCqueueActive is $Status_CDMU_TCqueueActive (extracted from TLM YM784944)"

set Status_CDMU_PSTfileName [getrawvalue [fetch YM809944]]
logm "Status_CDMU_PSTfileName is $Status_CDMU_PSTfileName (extracted from TLM YM809944)"

set Status_CDMU_PSTrunning [getrawvalue [fetch YM829944]]
logm "Status_CDMU_PSTrunning is $Status_CDMU_PSTrunning (extracted from TLM YM829944)"
logm ""

### Reading out PLM SCOE Settings
logm ">>> Reading out PLM SCOE Settings"
logm ""

set Status_PLM_OnLine [getrawvalue [fetch YM018942]]
logm "Status_PLM_OnLine is $Status_PLM_OnLine (extracted from TLM YM018942)"

set Status_PLM_PSU1_Master [getrawvalue [fetch YM129942]]
logm "Status_PLM_PSU1_Master is currently $Status_PLM_PSU1_Master (extracted from TLM YM129942)"

set Status_PLM_PSU1_Slave [getrawvalue [fetch YM145942]]
logm "Status_PLM_PSU1_Slave is currently $Status_PLM_PSU1_Slave (extracted from TLM YM145942)"

set Status_PLM_PSU2_Master [getrawvalue [fetch YM177942]]
logm "Status_PLM_PSU2_Master is currently $Status_PLM_PSU2_Master (extracted from TLM YM177942)"

set Status_PLM_PSU2_Slave [getrawvalue [fetch YM193942]]
logm "Status_PLM_PSU2_Slave is currently $Status_PLM_PSU2_Slave (extracted from TLM YM193942)"

set Status_PLM_LCL1_V [getrawvalue [fetch YM228942]]
logm "Status_PLM_LCL1_V is currently $Status_PLM_LCL1_V (extracted from TLM YM228942)"

set Status_PLM_LCL1_I [getrawvalue [fetch YM232942]]
logm "Status_PLM_LCL1_I is currently $Status_PLM_LCL1_I (extracted from TLM YM232942)"

set Status_PLM_LCL2_V [getrawvalue [fetch YM244942]]
logm "Status_PLM_LCL2_V is currently $Status_PLM_LCL2_V (extracted from TLM YM244942)"

set Status_PLM_LCL2_I [getrawvalue [fetch YM248942]]
logm "Status_PLM_LCL2_I is currently $Status_PLM_LCL2_I (extracted from TLM YM248942)"

set Status_PLM_LCL3_V [getrawvalue [fetch YM260942]]
logm "Status_PLM_LCL3_V is currently $Status_PLM_LCL3_V (extracted from TLM YM260942)"

set Status_PLM_LCL3_I [getrawvalue [fetch YM264942]]
logm "Status_PLM_LCL3_I is currently $Status_PLM_LCL3_I (extracted from TLM YM264942)"

set Status_PLM_LCL4_V [getrawvalue [fetch YM276942]]
logm "Status_PLM_LCL4_V is currently $Status_PLM_LCL4_V (extracted from TLM YM276942)"

set Status_PLM_LCL4_I [getrawvalue [fetch YM280942]]
logm "Status_PLM_LCL4_I is currently $Status_PLM_LCL4_I (extracted from TLM YM280942)"

set Status_PLM_LCL5_V [getrawvalue [fetch YM292942]]
logm "Status_PLM_LCL5_V is currently $Status_PLM_LCL5_V (extracted from TLM YM292942)"

set Status_PLM_LCL5_I [getrawvalue [fetch YM296942]]
logm "Status_PLM_LCL5_I is currently $Status_PLM_LCL5_I (extracted from TLM YM296942)"

set Status_PLM_LCL6_V [getrawvalue [fetch YM308942]]
logm "Status_PLM_LCL6_V is currently $Status_PLM_LCL6_V (extracted from TLM YM308942)"

set Status_PLM_LCL6_I [getrawvalue [fetch YM312942]]
logm "Status_PLM_LCL6_I is currently $Status_PLM_LCL6_I (extracted from TLM YM312942)"

set Status_PLM_LCL7_V [getrawvalue [fetch YM324942]]
logm "Status_PLM_LCL7_V is currently $Status_PLM_LCL7_V (extracted from TLM YM324942)"

set Status_PLM_LCL7_I [getrawvalue [fetch YM328942]]
logm "Status_PLM_LCL7_I is currently $Status_PLM_LCL7_I (extracted from TLM YM328942)"

set Status_PLM_LCL8_V [getrawvalue [fetch YM340942]]
logm "Status_PLM_LCL8_V is currently $Status_PLM_LCL8_V (extracted from TLM YM340942)"
```

```
set Status_PLM_LCL8_I [getrawvalue [fetch YM344942]]
logm "Status_PLM_LCL8_I is currently $Status_PLM_LCL8_I (extracted from TLM YM344942)"

set Status_PLM_LCL9_V [getrawvalue [fetch YM356942]]
logm "Status_PLM_LCL9_V is currently $Status_PLM_LCL9_V (extracted from TLM YM356942)"

set Status_PLM_LCL9_I [getrawvalue [fetch YM360942]]
logm "Status_PLM_LCL9_I is currently $Status_PLM_LCL9_I (extracted from TLM YM360942)"

set Status_PLM_LCL10_V [getrawvalue [fetch YM372942]]
logm "Status_PLM_LCL10_V is currently $Status_PLM_LCL10_V (extracted from TLM YM372942)"

set Status_PLM_LCL10_I [getrawvalue [fetch YM376942]]
logm "Status_PLM_LCL10_I is currently $Status_PLM_LCL10_I (extracted from TLM YM376942)"

set Status_PLM_LCL11_V [getrawvalue [fetch YM388942]]
logm "Status_PLM_LCL11_V is currently $Status_PLM_LCL11_V (extracted from TLM YM388942)"

set Status_PLM_LCL11_I [getrawvalue [fetch YM392942]]
logm "Status_PLM_LCL11_I is currently $Status_PLM_LCL11_I (extracted from TLM YM392942)"

set Status_PLM_LCL12_V [getrawvalue [fetch YM404942]]
logm "Status_PLM_LCL12_V is currently $Status_PLM_LCL12_V (extracted from TLM YM404942)"

set Status_PLM_LCL12_I [getrawvalue [fetch YM408942]]
logm "Status_PLM_LCL12_I is currently $Status_PLM_LCL12_I (extracted from TLM YM408942)"

set Status_PLM_LCL13_V [getrawvalue [fetch YM420942]]
logm "Status_PLM_LCL13_V is currently $Status_PLM_LCL13_V (extracted from TLM YM420942)"

set Status_PLM_LCL13_I [getrawvalue [fetch YM424942]]
logm "Status_PLM_LCL13_I is currently $Status_PLM_LCL13_I (extracted from TLM YM424942)"

set Status_PLM_LCL14_V [getrawvalue [fetch YM436942]]
logm "Status_PLM_LCL14_V is currently $Status_PLM_LCL14_V (extracted from TLM YM436942)"

set Status_PLM_LCL14_I [getrawvalue [fetch YM440942]]
logm "Status_PLM_LCL14_I is currently $Status_PLM_LCL14_I (extracted from TLM YM440942)"
logm ""


# ------------------------------------------------------------------
# The following list of commands sets all BOL detector biases to 0
# to prepare the unit for switch-off. This is essential in order to avoid
# any detector damage.
# ------------------------------------------------------------------

# Reset bias for all groups sequentially
logm "Reset bias for all groups sequentially"

#    Set all groups bol bias 02 (VL)       to 0.000 Volt (0)
tcsend PC103420 {PP071420 0x00020000 LO}; waittime 0.5

#    Set all groups bol bias 05 (VCH)      to 0.000 Volt (0)
tcsend PC103420 {PP071420 0x00050000 LO}; waittime 0.5

#    Set all groups bol bias 01 (VH)       to 0.000 Volt (0)
tcsend PC103420 {PP071420 0x00010000 LO}; waittime 0.5

#    Set all groups bol bias 03 (VRL)      to 0.000 Volt (0)
tcsend PC103420 {PP071420 0x00030000 LO}; waittime 0.5

#    Set all groups bol bias 04 (VINJ)     to 0.000 Volt (0)
tcsend PC103420 {PP071420 0x00040000 LO}; waittime 0.5

#    Set all groups bol bias 06 (VDL)      to 0.000 Volt (0)
tcsend PC103420 {PP071420 0x00060000 LO}; waittime 0.5

#    Set all groups bol bias 08 (VGL)      to 0.000 Volt (0)
tcsend PC103420 {PP071420 0x00080000 LO}; waittime 0.5

#    Set all groups bol bias 07 (VSS)      to 0.000 Volt (0)
tcsend PC103420 {PP071420 0x00070000 LO}; waittime 0.5

#    Set all groups bol bias 16 (VDD)      to 0.000 Volt (0)
tcsend PC103420 {PP071420 0x00100000 LO}; waittime 0.5

#    Set all groups bol bias 15 (VGG)      to 0.000 Volt (0)
tcsend PC103420 {PP071420 0x000F0000 LO}; waittime 0.5

#    Set all groups bol bias 09 (CKRLH)    to 0.000 Volt (0)
tcsend PC103420 {PP071420 0x00090000 LO}; waittime 0.5

#    Set all groups bol bias 10 (CKRLL)    to 0.000 Volt (0)
tcsend PC103420 {PP071420 0x000A0000 LO}; waittime 0.5

#    Set all groups bol bias 11 (VDECX-H)  to 0.000 Volt (0)
```

```
tcsend PC103420 {PP071420 0x000B0000 LO}; waittime 0.5

#    Set all groups bol bias 12 (VDECX-L)  to 0.000 Volt (0)
tcsend PC103420 {PP071420 0x000C0000 LO}; waittime 0.5

#    Set all groups bol bias 13 (VSMS-H)   to 0.000 Volt (0)
tcsend PC103420 {PP071420 0x000D0000 LO}; waittime 0.5

#    Set all groups bol bias 14 (VSMS-L)   to 0.000 Volt (0)
tcsend PC103420 {PP071420 0x000E0000 LO}; waittime 0.5

#    Set all groups bol bias 18 (VDL-BU)   to 0.000 Volt (0)
tcsend PC103420 {PP071420 0x00120000 LO}; waittime 0.5

#    Set all groups bol bias 20 (VH-BLIND) to 0.000 Volt (0)
tcsend PC103420 {PP071420 0x00140000 LO}; waittime 0.5

#    Set all groups bol bias 19 (VGL-BU)   to 0.000 Volt (0)
tcsend PC103420 {PP071420 0x00130000 LO}; waittime 0.5

#    Set all groups bol bias 17 (VSS-BU)   to 0.000 Volt (0)
tcsend PC103420 {PP071420 0x00110000 LO}; waittime 0.5

waittime 1.0

#    Set all groups bol bias 21 (VDD-PROT-CL) OFF (0)
tcsend PC103420 {PP071420 0x00150000 LO}; waittime 0.5

#    Set all groups bol bias 22 (VDD-PROT-BU) OFF (0)
tcsend PC103420 {PP071420 0x00160000 LO}; waittime 0.5

#    Set all groups bol bias 23 (GND-BU) OFF (0)
tcsend PC103420 {PP071420 0x00170000 LO}; waittime 0.5

logm "BOL biases are set to zero"

# Now BOLC is prepared for switch-off
logm "Now BOLC is prepared for switch-off"

#--------------------------------
#  Set temperature probes off
#--------------------------------
logm "Set temperature probes off"
tcsend PC103420 {PP071420 0x07000000 LO}
waittime 0.5

#--------------------------------
# Set all groups to OFF
#--------------------------------
logm "Set all groups to OFF"
tcsend PC103420 {PP071420 0x0A000000 LO}
waittime 2.0


# ------------------------------------------------------------------
# This part contains the code to configure the switching of the LCL power switches
# ------------------------------------------------------------------

set LCL_V(1) "YM228942"
set LCL_I(1) "YM232942"
set LCL_V(2) "YM244942"
set LCL_I(2) "YM248942"
set LCL_V(3) "YM260942"
set LCL_I(3) "YM264942"
set LCL_V(4) "YM276942"
set LCL_I(4) "YM280942"
set LCL_V(5) "YM292942"
set LCL_I(5) "YM296942"
set LCL_V(6) "YM308942"
set LCL_I(6) "YM312942"
set LCL_V(7) "YM324942"
set LCL_I(7) "YM328942"
set LCL_V(8) "YM340942"
set LCL_I(8) "YM344942"
set LCL_V(9) "YM356942"
set LCL_I(9) "YM360942"
set LCL_V(10) "YM372942"
set LCL_I(10) "YM376942"
set LCL_V(11) "YM388942"
set LCL_I(11) "YM392942"
set LCL_V(12) "YM404942"
set LCL_I(12) "YM408942"
set LCL_V(13) "YM420942"
set LCL_I(13) "YM424942"
set LCL_V(14) "YM436942"
set LCL_I(14) "YM440942"

set PSU_Master(1) "YM129942"
```

```
set PSU_Slave(1) "YM145942"
set PSU_Master(2) "YM177942"
set PSU_Slave(2) "YM193942"

# ----------------------------------------------------------------
# This part contains the code to switch off the SPU
# ----------------------------------------------------------------

logm ">>> Switch OFF SPU"
logm ""

# Execute Telecommand
tcsend YC041942 checks {SPTV DPTV CEV} ack NONE referby rYC041942 \
            {YP411942 1} \
            [list YP412942 14]
logm " Sending Telecommand YC041942 to Disable Limiter 14 PACS SPU"
logm ""

logm ">>> Checking"
waittime 6.00
set Status_Voltage [getrawvalue [fetch $LCL_V(14)]]
set Status_Current [getrawvalue [fetch $LCL_I(14)]]
logm "LCL 14 has currently a voltage of $Status_Voltage.(from $LCL_V(14))"
logm "LCL 14 has currently a current of $Status_Current.(from $LCL_I(14))"
logm ""

waittime 0.5

# ----------------------------------------------------------------
# This part contains the code to switch off the BOLC
# ----------------------------------------------------------------

logm ">>> Switch OFF BOLC"
logm ""

# Execute Telecommand
tcsend YC041942 checks {SPTV DPTV CEV} ack NONE referby rYC041942 \
            {YP411942 1} \
            [list YP412942 11]
logm " Sending Telecommand YC041942 to Disable Limiter 11 PACS BOLC"
logm ""

logm ">>> Checking"
waittime 6.00
set Status_Voltage [getrawvalue [fetch $LCL_V(11)]]
set Status_Current [getrawvalue [fetch $LCL_I(11)]]
logm "LCL 11 has currently a voltage of $Status_Voltage.(from $LCL_V(11))"
logm "LCL 11 has currently a current of $Status_Current.(from $LCL_I(11))"
logm ""

waittime 0.5

# ----------------------------------------------------------------
# This part contains the code to switch off the DECMEC
# ----------------------------------------------------------------

logm ">>> Switch OFF DECMEC"
logm ""

# Execute Telecommand
tcsend YC041942 checks {SPTV DPTV CEV} ack NONE referby rYC041942 \
            {YP411942 1} \
            [list YP412942 12]
logm " Sending Telecommand YC041942 to Disable Limiter 12 PACS DECMEC"
logm ""

logm ">>> Checking"
waittime 6.00
set Status_Voltage [getrawvalue [fetch $LCL_V(12)]]
set Status_Current [getrawvalue [fetch $LCL_I(12)]]
logm "LCL 12 has currently a voltage of $Status_Voltage.(from $LCL_V(12))"
logm "LCL 12 has currently a current of $Status_Current.(from $LCL_I(12))"
logm ""

waittime 0.5

# ----------------------------------------------------------------
# This part contains the code to switch off the DPU
# ----------------------------------------------------------------

logm ">>> Switch OFF DPU"
logm ""

# Execute Telecommand
tcsend YC041942 checks {SPTV DPTV CEV} ack NONE referby rYC041942 \
            {YP411942 1} \
            [list YP412942 13]
logm " Sending Telecommand YC041942 to Disable Limiter 13 PACS DPU"
```

```
logm ""

logm ">>> Checking"
waittime 6.00
set Status_Voltage [getrawvalue [fetch $LCL_V(13)]]
set Status_Current [getrawvalue [fetch $LCL_I(13)]]
logm "LCL 13 has currently a voltage of $Status_Voltage.(from $LCL_V(13))"
logm "LCL 13 has currently a current of $Status_Current.(from $LCL_I(13))"
logm ""

waittime 0.5

logm "PACS is off"

### Reading out CDMUDFE Settings
logm ">>> Reading out CDMUDFE Settings"
logm ""

set Status_CDMU_OnLine [getrawvalue [fetch YM777944]]
logm "Status_CDMU_OnLine is $Status_CDMU_OnLine (extracted from TLM YM777944)"

set Status_CDMU_TMpolling [getrawvalue [fetch YM780944]]
logm "Status_CDMU_TMpolling is $Status_CDMU_TMpolling (extracted from TLM YM780944)"

set Status_CDMU_SAreadActive [getrawvalue [fetch YM781944]]
logm "Status_CDMU_SAreadActive is $Status_CDMU_SAreadActive (extracted from TLM YM781944)"

set Status_CDMU_SAqueueActive [getrawvalue [fetch YM782944]]
logm "Status_CDMU_SAqueueActive is $Status_CDMU_SAqueueActive (extracted from TLM YM782944)"

set Status_CDMU_TMqueueActive [getrawvalue [fetch YM783944]]
logm "Status_CDMU_TMqueueActive is $Status_CDMU_TMqueueActive (extracted from TLM YM783944)"

set Status_CDMU_TCqueueActive [getrawvalue [fetch YM784944]]
logm "Status_CDMU_TCqueueActive is $Status_CDMU_TCqueueActive (extracted from TLM YM784944)"

set Status_CDMU_PSTfileName [getrawvalue [fetch YM809944]]
logm "Status_CDMU_PSTfileName is $Status_CDMU_PSTfileName (extracted from TLM YM809944)"

set Status_CDMU_PSTrunning [getrawvalue [fetch YM829944]]
logm "Status_CDMU_PSTrunning is $Status_CDMU_PSTrunning (extracted from TLM YM829944)"
logm ""

### Reading out PLM SCOE Settings
logm ">>> Reading out PLM SCOE Settings"
logm ""

set Status_PLM_OnLine [getrawvalue [fetch YM018942]]
logm "Status_PLM_OnLine is $Status_PLM_OnLine (extracted from TLM YM018942)"

set Status_PLM_PSU1_Master [getrawvalue [fetch YM129942]]
logm "Status_PLM_PSU1_Master is currently $Status_PLM_PSU1_Master (extracted from TLM YM129942)"

set Status_PLM_PSU1_Slave [getrawvalue [fetch YM145942]]
logm "Status_PLM_PSU1_Slave is currently $Status_PLM_PSU1_Slave (extracted from TLM YM145942)"

set Status_PLM_PSU2_Master [getrawvalue [fetch YM177942]]
logm "Status_PLM_PSU2_Master is currently $Status_PLM_PSU2_Master (extracted from TLM YM177942)"

set Status_PLM_PSU2_Slave [getrawvalue [fetch YM193942]]
logm "Status_PLM_PSU2_Slave is currently $Status_PLM_PSU2_Slave (extracted from TLM YM193942)"

set Status_PLM_LCL1_V [getrawvalue [fetch YM228942]]
logm "Status_PLM_LCL1_V is currently $Status_PLM_LCL1_V (extracted from TLM YM228942)"

set Status_PLM_LCL1_I [getrawvalue [fetch YM232942]]
logm "Status_PLM_LCL1_I is currently $Status_PLM_LCL1_I (extracted from TLM YM232942)"

set Status_PLM_LCL2_V [getrawvalue [fetch YM244942]]
logm "Status_PLM_LCL2_V is currently $Status_PLM_LCL2_V (extracted from TLM YM244942)"

set Status_PLM_LCL2_I [getrawvalue [fetch YM248942]]
logm "Status_PLM_LCL2_I is currently $Status_PLM_LCL2_I (extracted from TLM YM248942)"

set Status_PLM_LCL3_V [getrawvalue [fetch YM260942]]
logm "Status_PLM_LCL3_V is currently $Status_PLM_LCL3_V (extracted from TLM YM260942)"

set Status_PLM_LCL3_I [getrawvalue [fetch YM264942]]
logm "Status_PLM_LCL3_I is currently $Status_PLM_LCL3_I (extracted from TLM YM264942)"

set Status_PLM_LCL4_V [getrawvalue [fetch YM276942]]
logm "Status_PLM_LCL4_V is currently $Status_PLM_LCL4_V (extracted from TLM YM276942)"

set Status_PLM_LCL4_I [getrawvalue [fetch YM280942]]
logm "Status_PLM_LCL4_I is currently $Status_PLM_LCL4_I (extracted from TLM YM280942)"

set Status_PLM_LCL5_V [getrawvalue [fetch YM292942]]
logm "Status_PLM_LCL5_V is currently $Status_PLM_LCL5_V (extracted from TLM YM292942)"
```

```
set Status_PLM_LCL5_I [getrawvalue [fetch YM296942]]
logm "Status_PLM_LCL5_I is currently $Status_PLM_LCL5_I (extracted from TLM YM296942)"

set Status_PLM_LCL6_V [getrawvalue [fetch YM308942]]
logm "Status_PLM_LCL6_V is currently $Status_PLM_LCL6_V (extracted from TLM YM308942)"

set Status_PLM_LCL6_I [getrawvalue [fetch YM312942]]
logm "Status_PLM_LCL6_I is currently $Status_PLM_LCL6_I (extracted from TLM YM312942)"

set Status_PLM_LCL7_V [getrawvalue [fetch YM324942]]
logm "Status_PLM_LCL7_V is currently $Status_PLM_LCL7_V (extracted from TLM YM324942)"

set Status_PLM_LCL7_I [getrawvalue [fetch YM328942]]
logm "Status_PLM_LCL7_I is currently $Status_PLM_LCL7_I (extracted from TLM YM328942)"

set Status_PLM_LCL8_V [getrawvalue [fetch YM340942]]
logm "Status_PLM_LCL8_V is currently $Status_PLM_LCL8_V (extracted from TLM YM340942)"

set Status_PLM_LCL8_I [getrawvalue [fetch YM344942]]
logm "Status_PLM_LCL8_I is currently $Status_PLM_LCL8_I (extracted from TLM YM344942)"

set Status_PLM_LCL9_V [getrawvalue [fetch YM356942]]
logm "Status_PLM_LCL9_V is currently $Status_PLM_LCL9_V (extracted from TLM YM356942)"

set Status_PLM_LCL9_I [getrawvalue [fetch YM360942]]
logm "Status_PLM_LCL9_I is currently $Status_PLM_LCL9_I (extracted from TLM YM360942)"

set Status_PLM_LCL10_V [getrawvalue [fetch YM372942]]
logm "Status_PLM_LCL10_V is currently $Status_PLM_LCL10_V (extracted from TLM YM372942)"

set Status_PLM_LCL10_I [getrawvalue [fetch YM376942]]
logm "Status_PLM_LCL10_I is currently $Status_PLM_LCL10_I (extracted from TLM YM376942)"

set Status_PLM_LCL11_V [getrawvalue [fetch YM388942]]
logm "Status_PLM_LCL11_V is currently $Status_PLM_LCL11_V (extracted from TLM YM388942)"

set Status_PLM_LCL11_I [getrawvalue [fetch YM392942]]
logm "Status_PLM_LCL11_I is currently $Status_PLM_LCL11_I (extracted from TLM YM392942)"

set Status_PLM_LCL12_V [getrawvalue [fetch YM404942]]
logm "Status_PLM_LCL12_V is currently $Status_PLM_LCL12_V (extracted from TLM YM404942)"

set Status_PLM_LCL12_I [getrawvalue [fetch YM408942]]
logm "Status_PLM_LCL12_I is currently $Status_PLM_LCL12_I (extracted from TLM YM408942)"

set Status_PLM_LCL13_V [getrawvalue [fetch YM420942]]
logm "Status_PLM_LCL13_V is currently $Status_PLM_LCL13_V (extracted from TLM YM420942)"

set Status_PLM_LCL13_I [getrawvalue [fetch YM424942]]
logm "Status_PLM_LCL13_I is currently $Status_PLM_LCL13_I (extracted from TLM YM424942)"

set Status_PLM_LCL14_V [getrawvalue [fetch YM436942]]
logm "Status_PLM_LCL14_V is currently $Status_PLM_LCL14_V (extracted from TLM YM436942)"

set Status_PLM_LCL14_I [getrawvalue [fetch YM440942]]
logm "Status_PLM_LCL14_I is currently $Status_PLM_LCL14_I (extracted from TLM YM440942)"
logm ""

logm ""
newTest_gen "PACS Power Off Sequence has ended"
logm ""

finish_TS


#####################################################################
# end of test sequence
#####################################################################
# Changes:
# $Log: PACS_POWER_OFF.tcl,v $
# Revision 1.6  2005/05/31 14:34:24  ilsens
# File created by manual import
#
# Revision 1.2  2005/05/23 08:30:20  ilsens
# (No log message)
#
# Revision 1.1  2005/05/23 07:50:38  ilsens
# initial version
#
#
#####################################################################
```

E N D O F D O C U M E N T