TITLE:	DPU-ASW Management of the DCU ERROR REG content with possible off-line recovery actions		
Date:	20/10/2021	**Issue:**	1.1
Reference:	EUCL-IBO-TN-7-035		
Custodian:	Eduardo Medinaceli		

Contributors:	Function:	Date:	Signature:
Raoul Grimoldi	Industry Partner OHB-I		
Elio Mangraviti	Industry Partner OHB-I		
Andrea Balestra	NI-DPU-ASW developer		
Ruben Farinelli	NI-DPU-ASW developer		
Eduardo Medinaceli	DPU-ASW developer / manager		
Chiara Sirignano			
Approved by:	**Function:**	**Date:**	**Signature:**
Stefano Dusini	NI-AIV Manager deputy		
Authorized by:	**Function:**	**Date:**	**Signature:**
Luca Valenziano	Italian NISP Project Manager		

1. Change-log

Issue	Date	Page	Description Of Change
1.0	01/10/2021	all	First issue
1.1	20/10/2021	- Section 7 - Section 9 - Appendixes	- Removed TBDs, and added implementation details - Added - Added

Table of Contents

2. Documents

Applicable documents

AD	Title / Author	Document Reference	Issue	Date
1	Warm Electronics to Focal Plane ICD	EUCL-OPD-ICD-7-001	4.8	06/04/2018
2	NI-DPU ASW Requirement Specifications	EUCL-OPD-RS-7-001	4.6	30/04/2020
3	Digital Control Unit Firmware ICD	EUCL-OPD-ICD-7-002	2.2	05/04/2019

Reference documents

RD	Title / Author	Document Reference	Issue	Date
0	NISP Acronyms List	EUCL-IAP-LI-1-001	2.0	04/05/2013
1	NISP DPU/DCU HW Design Description Document	EUCL-OTO-RP-7-001	4.0	11/07/2016
2	NISP DPU_DCU BSW SW Driver API Interface Document	EUCL-IBO-DOC-7-001	4.0	27/10/2016
3	NISP DCU FPGA Design Description	EUC-TN-ATI-036	4.0	10/04/2018
4	NI-DPU DCU ASW Design Definition Document	EUCL-OPD-RP-7-001	5.4	22/06/2020
5	NISP Flight User Manual Issue	EUCL-IBO-MA-7-005	2.1	03/31/2021
6	NI-DPU ASW: blocking communication issue between SCE and DCU	EUCL-IBO-NCR-7-030	1.0	28/09/2021
7	EUCLID PLM TVAC - Phase 50c - NISP CRT issue	EUCL-ASFT-NCR-1-3-00090	1.0	29/05/2021
8	DPU ASW Static & Unit Test Report	EUCL-OPD-PL-7-005	2.1	17/07/2020

3. Acronyms

See [RD0]

4. On-line documentation

An on-line copy of this document can be found at:

https://euclid.baltig-pages.infn.it/DPU-ASW/DPUASW_errorhandling.html

5. DCU - Sidecar interface

The front-end interface that connects the DCU and the SIDECAR is split into two main sections: science data interface and TMTC interface.
A comprehensive description of these I/F can be found in Section 6.9.2 of [RD3]. Information about DCU drivers handling this interface is shown in [RD2]; while the double sampling mechanism implemented in the SIDECAR science I/F with the DCU can be found in [RD1].
For details on the DPU-ASW management of the DCU-Sidecar interface see [RD4].

6. Description of the DCU ERROR REGISTER handling strategy

Following NISP NCR [RD6] and PLM NCR [RD7] a strategy for handling each one of the errors present in the DCU_ERROR_REG was given by OHB-I (09/10/2021), including bits which do not correspond to errors i.e., bits 0 to 7 also reported in the same DCU register. The recovery actions, indicated in the next table on the column RECOVERY ACTIONS were also provided. The DPU-ASW team with OHB-I agreed the DPU-ASW actions to be implemented associated to each error (see the column BIT of the next table) on 09/13/2021. These actions are indicated in the column FOSESEE DPU-ASW ACTIONS; for completeness the old DPU-ASW (v1.3.7 used during CSL test campaign) actions are indicated in the last column.

DCU_ERROR_REG (DCU register area 0x0020)

The values in the register are latched and are reset only when the DCU_CMD_RESET_ERROR_REG command is issued: so, the register reports all the flags that have been raised since the last ERROR_RESET command [RD3].

Table 1, DCU_ERROR_REG description, DPU-ASW foresee actions, plus different options for the recovery action sorted with increasing effects over the NI-Focal Plane state.

BIT	NAME	DEFAULT	DESCRIPTION	RECOVERY ACTIONS	FORESEE DPU-ASW ACTIONS	CURRENT (v1.3.7) DPU-ASW ACTIONS
31	SIDECAR_CMD_DONE	0	This bit is raised after the correct execution of a command directed to the SIDECAR. When the start command (SEND_CMD, bit1 of DCU_COMMAND_REG) is sent by the DPU this flag is cleared and it will be raised to '1' only when the current command is executed correctly. NOTE: This status bit is copied also in the error register in order to allow the SW to monitor the execution of the commands and the error occurred with only a single access.	Comment: in case this error is detected the command sent to the SCE shall be considered not valid. Possible recovery actions are: • Reset DCU_ERROR_REG • new communication trial or • reinitialization of the DCU and SCE without power cycle • Reset DCU_ERROR_REG	- Reset TMTC I/F - Return ERROR	- Return ERROR

				• SCE soft boot (DCU abort + reset of error reg and SCE soft boot) or • Power cycle the DCU		
30 : 21	reserved	0	-	Comment: these bits are always '0'.	- No action comment: the content is stored in the telemetry register	- No action comment: the content is stored in the telemetry register
20	SDRAM_single_err	0	This bit goes high when a single error has been detected and corrected on the SDRAM memory.	Comment: Single errors are corrected by the EDAC. Recovery action • none	- No action	- Return error
19	SDRAM_double_err	0	This bit goes high when a double error has been detected and not corrected on the SDRAM memory.	Comment: multiple errors cannot be corrected by the EDAC, so corruption of data is expected. Comment2: This error should be monitored on Ground, because most probably errors in images will be present. Recovery action: an analysis is needed only in	- No action	- Return error

				case of repeated occurrences and condition described in Comment2 satisfied: • DCU power cycle		
18	LCL_prot_triggered	0	This bit goes high when the signal LCL_STAT goes low for at least 1ms after the DCDC_power_on command is received. After this failure all the SIDECAR power enable are instantly switch off. REMARK: The configuration SIDECAR_keep_ON must be cleared (if it was enabled) before to reset this flag. REMARK: to recover this error the DCU must be reset.	Recovery action • DCU power on and SCE boot	- Return error comment: the isolation of the error is handled by FDIR 28V NISP-FDIR-221	- Return error comment: the isolation of the error is handled by FDIR 28V NISP-FDIR-221
17	HK_prot_triggered	0	This bit goes high when an overvoltage or overcurrent is detected on a protected channel. After this failure all the SIDECAR power enable are instantly switch off and the related information are saved in	Recovery action • DCU power on and SCE boot	- Return ERROR comment: the isolation of the error is handled by SCE Th FDIR NISP-FDIR-205	- Return ERROR comment: the isolation of the error is handled by SCE Th FDIR NISP-FDIR-205

			the HK registers (last acquired value on the SPI channel and SPI channel number). REMARK: The configuration SIDECAR_keep_ON must be cleared (if it was enabled) before to reset this flag. REMARK: to recover this error the DCU must be reset.			
16	Invalid_2V5_val	0	This bit goes high when the command "SET_2V5D" is received and the value of register 2V5_value is outside of the allowed range. This protection can be disabled for debug purpose writing '1' in OVERRIDE_DAC_PROT field of the HK_CONFIG_REG register.	Recovery action: an analysis is needed. To be checked after programming of the 2V5 only. In case the bit has been asserted, an attempt to program a value outside allowed range has been detected and the value has been rejected. A correct value shall be programmed.	- Return error	- Return error
15	Invalid_VDDA_val	0	This bit goes high when the command "SET_VDDA" is received and the value of register VDDA_value is outside of	Recovery action: an analysis is needed. To be checked after programming of the VDDA only.	- Return error	- Return error

			the allowed range. This protection can be disabled for debug purpose writing '1' in OVERRIDE_DAC_PROT field of the HK_CONFIG_REG register.	In case the bit has been asserted, an attempt to program a value outside allowed range has been detected and the value has been rejected. A correct value shall be programmed.		
14	UART_CRC_err	0	This bit goes high when a CRC error has been detected in the UART TMTC interface between DCU and DRB.	Comment: in case this error is detected on the UART I/F by the DCU, the command is not executed. The status of the DCU may be not the one expected by the ASW. Possible actions are: • reinitialization of the DCU (without powering off SCE) • Reset DCU_ERROR_REG • SCE soft boot (DCU abort + reset of error reg and SCE soft boot) or • Power cycle	- Return error	- Return error
13	UART_sync_err	0	This bit goes high when an invalid synchronization word has	See above	- Return error	- Return error

			been received from the DRB.			
12	UART_timeout	0	This bit goes high when a timeout has been detected in the UART TMTC interface between DCU and DRB.	See above	- Return error	- Return error
11	TMTC_IF_CRC_err	0	This bit goes high when a CRC error is detected into the reply packet from the SIDECAR.	Comment: In case this error is detected on the SCE TMTC I/F by the DCU, the data received shall be considered not valid. Recovery actions are: • Reset DCU_ERROR_REG • new communication trial or • reinitialization of the DCU (without powering off SCE) • Reset DCU_ERROR_REG • SCE soft boot (DCU abort + reset of error reg and SCE soft boot) or	- Reset TMTC I/F - Return error	- Reset TMTC I/F - Return error

				• Power cycle the DCU Note: CRC is available on block read and write only		
10	TMTC_IF_parity_err	0	This bit goes high when a parity error is detected into the reply packet from the SIDECAR.	Comment: in case this error is detected on the SCE TMTC I/F by the DCU, the data received shall be considered not valid. Recovery Actions: • Reset DCU_ERROR_REG • new communication trial or • reinitialization of the DCU (without powering off SCE) • Reset DCU_ERROR_REG • SCE soft boot (DCU abort + reset of error reg and SCE soft boot) or • Power cycle the DCU	- Reset TMTC I/F - Return error	- Reset TMTC I/F - Return error

				Note: parity is available on single, or block read and write		
9	TMTC_IF_reply_to	0	This bit goes high when a word of the reply packet from the SIDECAR is not received within 255 clock cycle at 10MHz (a bit more of the time needed to receive 2 words) after the previous one. The TMTC interface keep waiting for the arrival of the following words until the end of message.	Comment: In case this error is detected on the SCE TMTC I/F by the DCU, the communication shall be considered not completed. The status of the SCE may be not the one expected by the ASW Recovery Actions: • Reset DCU_ERROR_REG • new communication trial or • reinitialization of the DCU (without powering off SCE) • Reset DCU_ERROR_REG • SCE soft boot (DCU abort + reset of error reg and SCE soft boot) or	- Reset TMTC I/F - Return error	- Reset TMTC I/F - Return error

				• Power cycle the DCU		
8	TMTC_IF_ACK_pulse_to	0	This bit goes high when the acknowledge pulse of a transmitted word is not received within a 1023 clock cycle at 10MHz. The TMTC interface keep waiting for the arrival of the pulse.	Comment: In case this error is detected on the SCE TMTC I/F by the DCU, the communication shall be considered not completed. The status of the SCE may be not the one expected by the ASW. Possible actions are: • Reset DCU_ERROR_REG • new communication trial or • reinitialization of the DCU (without powering off SCE) • Reset DCU_ERROR_REG • SCE soft boot (DCU abort + reset of error reg and SCE soft boot) or	- Reset TMTC I/F - Return error	- Reset TMTC I/F - Return error

				• Power cycle the DCU		
7	Row_timeout	0	This bit goes high when the SIDECAR has not transmitted a valid science data packet within the defined timeout, during the execution of a MACC operation. The timeout is disabled between two consecutive frames.	Comment: Row timeout shall be considered an indication of a problem on the communication with the SCE.	- No action	- No action
6	Line_number_error	0	This bit goes high when the line number received in the header of the packet is not equal to the internal row counter.	Comment: Line Number Error shall be considered an indication of a problem on the communication with the SCE.	- No action	- Return error
5	Science_IF_CRC_err_1	0	This bit goes high when a transmission error has occurred in the science packet and, consequently, a wrong CRC is received. This error is referred only to FIFO_1. This error can be triggered also if a valid synch and block length are received but the transmission by the SCE	Comment: Being the harness delay not known, the Science_IF has been designed to sample on rising and falling edge of the SCE_CLK. At least one of the two sampling times will result in a correct data sampling. So, it is expected that in normal condition FIFO0 or FIFO1 can signal anomalies but not both. So one FIFO only signalling anomalies	- No action	- Return error

			is interrupted in the half of the packet.	shall not cause any reaction by the ASW, just reporting in TLM. Both FIFOs signalling anomalies shall be considered an indication of a problem on the communication with the SCE.		
4	Invalid_block_length_1	0	This bit goes high if, after the reception of a valid Synch, an invalid block size is detected. This error is referred only to FIFO_1.	See above.	- No action	- Return error
3	Science_IF_Sync_err_1	0	This bit goes high when the SIDECAR science bus leaves the idle state and the first received byte is different from the synch word. This error is referred only to FIFO_1	See above.	- No action	- Return error
2	Science_IF_CRC_err_0	0	This bit goes high when a transmission error has occurred in the science packet and, consequently, a wrong CRC is received. This error is referred only to FIFO_0. This error can be triggered also if a valid synch and block length are received but the	See above.	- No action	- Return error

			transmission by the SCE is interrupted in the half of the packet.			
1	Invalid_block_length_0	0	This bit goes high if, after the reception of a valid Synch, an invalid block size is detected. This error is referred only to FIFO_0	See above.	- No action	- Return error
0	Science_IF_Sync_err_0	0	This bit goes high when the SIDECAR science bus leaves the idle state and the first received byte is different from the synch word. This error is referred only to FIFO_0	See above.	- No action	- Return error

7. Implementation Proposal

To handle all the errors (besides the ones related to the FIFOs, Row_timeout, Line_number_error, SDRAM_single_err, SDRAM_double_err) with a common strategy: SCE Read and SCE Write functions will return error stopping the execution of the command implementing any of these functions and providing in TLM the error register (DCU_ERROR_REG).
In the case when TM/TC errors the TM/TC I/F will be reset, and the error register will be cleared. This implies that in those cases (bits 8-11) is possible that there will be no visibility of the DCU_ERROR_REG with the error (bits rise) because it can be cleared, but a TM(5,2) related to the associated command will be issued. This choose allows to restore cyclic operations using the TM/TC interface in the case the reset solves the issue.
The recovery action for all the other cases always implies the reset of the error register followed by the sequences described by OHB in column **RECOVERY ACTIONS** of Table 1.

The intervention in the code is minimized and the recovery procedures are implemented using the **FORESEE DPU-ASW ACTIONS**. A copy of Table 1 with all the information will be documented also in [RD5].

To avoid recursive issues while performing consecutive exposures operation related to data production e.g., inside a command pipeline where several exposure commands are sent, the error severity related to a failure of the exposure command will be increased. In the former implementation a TM(5,2) was rise provoking an event PUS(5,2) which do not stop the command pipeline; in the new implementation instead a TM(5,3) will be rise generating an event PUS(5,3) leading the NI-ICU into SAFE state stopping the command pipeline.
Therefore, if during execution any of the following commands fails:
SCE_EXP-NISC0403 will issue error SCE_EEXP (0x73D0009) and a PUS(5,3) event,
SCE_KTC_EXP-NISC0402 will issue error SCE_EKTCEXP (0x73D0011) and a PUS(5,3) event,
and SCE_IPC_EXP-NISC0401will issue error SCE_EIPCEXP (0x73D0019) and a PUS(5,3) event.
These error assignments can be found on the on-line documentation at
https://euclid.baltig-pages.infn.it/DPU-ASW/DPUASW_errorhandling.html#t06

This mechanism is independent to the DPU's watchdog systems that prevent a system halt due to the non-production of data by the exposure command (WD_EoE) and the watchdog related to the data processing of a Dither (WD_EoD) documented in RD-4.

DPU-ASW Implementation

A distinction between different errors and 'warnings' is done according to OHB input summarized in the following table:

DCU_ERROR_REG bit	action
- FIFO-like errors (bits 0 - 5)	no action;
- line number (bit 6)	no action;
- row timeout (bit7)	no action;
- TMTC reset (bits 8 – 11)	resetTMTC(); return error;
- UART error, (bits 12 - 14)	return error;
- invalid power supply V (bits 15, 16)	return error;
- LCL protections (bits 17, 18)	return error;
- SDRAM errors, (bits19 - 20)	no action;
- reserved (unused) bits (bits 21 - 30)	no action;
- CMD done (bit31=1)	no action;
- CMD not done (bit 31=0)	resetTMTC(); return error;

- The clear of the DCU_ERROR_REG will be done automatically only when the TM/TC I/F is reset internally (TBD)
- In all the other cases when an error is returned, the clear of the register is associated to the recovery action.

Figure 1 shows the implementation philosophy in the DPU-ASW of the new handling of the DCU_ERROR_REG.

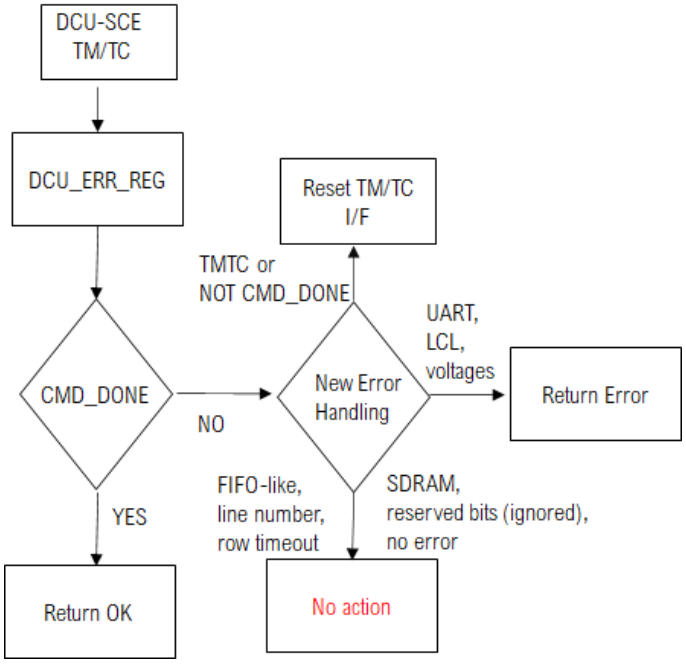


Figure 1, DPU-ASW new DCU_ERROR_REG handling strategy

The UML timeline diagram of Figure 2 shows the new error injection mechanism associated to errors during execution of commands SCE_EXP, SCE_KTCEXP, and SCE_IPCEXP denoted in the figure as SCE_EXP().

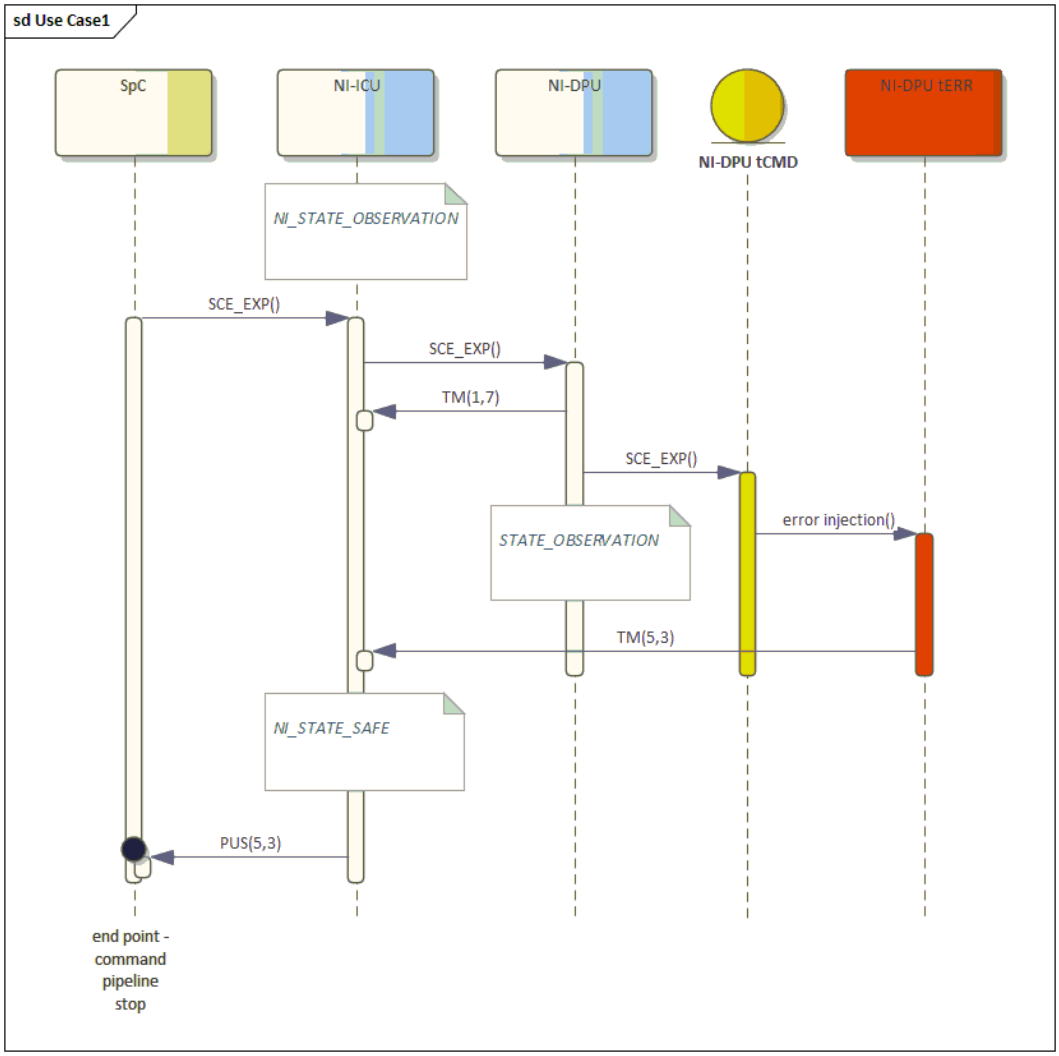


Figure 2, UML timeline diagram showing the implementation of the error injection associated to errors during execution of command SCE_EXP, SCE_KTC and SCE_IPC (represented with SCE_EXP()).The Spacecraft (SpC), the NISP ICU, NISC DPU, and the DPU-ASW Error task (tERR) and command task (tCMD) are represented with lifelines. Events TM(1,7) positive command verification, and error TM(5,3) generating an event PUS(5,3) are also shown; as well as the change of state of NISP associated to this last event. The end-point shows the stop of the SpC's command pipeline.

The configuration control entries associated to this implementation are (attached in Appendix E), on-line versions at:

https://issues.infn.it/jira/browse/EUNIDPUASW-165

https://issues.infn.it/jira/browse/EUNIDPUASW-166

8. NISP Telemetry

The DCU_ERROR_REG (32 bit) is saved for each DCU in telemetry using the following parameters according to MIB-3.68:

For DPU 1:
NIST0905, NIST1033, NIST1161, NIST1289, NIST1417, NIST1545, NIST1673, NIST1801
(parameter description: NISP_DPUAG_DC1-8_Error_Reg)
For DPU 2:
NIST5001, NIST5129, NIST5257, NIST5385, NIST5513, NIST5641, NIST5769, NIST5897
(parameter description: NISP_DPUBG_DC1-8_Error_Reg)

9. Software implementation tests

The version including the management of the DCU_ERROR_REG was extensively tested at LAM using a cryogenic setup composed by the DPU-EQM equipped with 3 SCEs. Also, a custom delay line was used to force the sampling the data using FIFO_0 or FIFO_1. The results can be found in Appendix A.

The continuous integration engine was applied to the software, including unit testing, and static testing. All the result are included in Appendixes B, C, and D.

The software changes are tracked using the DPU-ASW configuration control toolkit, and entries related to the new implementation are reported in Appendix E.

Appendix A – Test results using DPU-EQM at LAM at nominal conditions

Here are reported the slides discussed during the two NRBs where the results of the preliminary tests of the new implementation were presented. In the slides are described the different setups used during the tests at LAM. The first presentation shows the 'proof of concept' results, where FIFO-like errors were not considered in the error management. The second presentation 'Implementation proposal', shows the results of the final implementation of the DCU_ERRO_REG management implementation, and results of the error handling are shown for different errors injected using a debugger connected to the DPU-EQM. A Monte Carlo study simulating the injection of different errors and their classification is also included.

Proof of concept test

Results of the DPU-ASW tests during the NISP investigation at LAM, 13-18 September 2021

E. Medinaceli DPU-ASW manager for the NISP team

Software Implementation

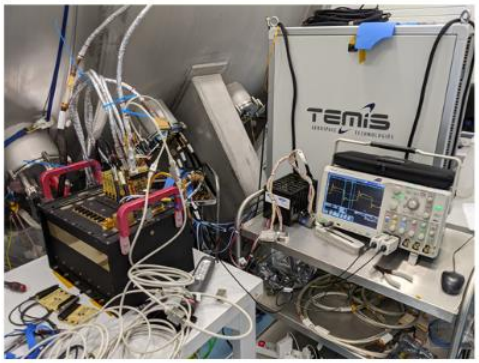
The provisionary patch DPU-ASWv1.3.7.1 was used only to prove the concept of the SW modification following NCR EUCL-ASFT-NCR-1-3-00090.
In this version internal actions triggered by "errors" related to FIFO_0 and FIFO_1 were excluded from the ReadSCE and WriteSCE functions

Test Setup

Using the LAM setup:
- DPU-EQM equipped with 4 SCEs operating at different temperatures (cryo/warm)
- DPU-ASWv1.3.7.1 was loaded in the DPU-EQM E2PROM (image 2) using the NISP MEM load service (1553 I/F PUS Service 6)
- No debugging tools were used during the test

Equipment:
- DPU-EM using 4 DCUs
- JIG board connected to DCU1
- delay line connected to DCU1 (with different lengths)
- 4 flight-like SCEs connected using ERIOS feedthrough
- flight-like length of NISP harnesses

Auxiliar equipment:
- Markury electronics (stand-alone SCE readout system)
- Oscilloscope interfaced with JIG
- ICU-EQM powered by ICU-TE
- CCS-SCOE



Test cases

The following error conditions seen at CSL were tested triggering both FIFO_0 and FIFO_1 errors independently and once at the time on DCU1. Broadcast exposures were done using 4 DCUs

(1) HSK stop while images acquisition
(2) consecutive exposures (using one or more detectors) when any FIFO error is generated
(3) ASICs current increase using sequence: SCE_EXP with the 'CSL error condition', ALIVENESS check, 2nd SCE_EXP

Test results

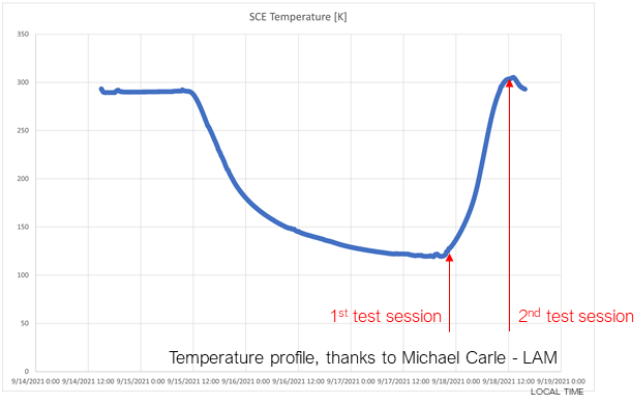
Test cases (1), (2) and (3) were executed using 4 DCUs. Only in DCU1 were injected (fake) "communication errors", inducing the data sampling on FIFO_0 and in FIFO_1 using different length of dedicated delay lines.

1st test session on 17/09/2021
started at 7:30 PM (UTC) at SCE_T ~ 122 K

2nd test session on 18/09/2021
started at 8:50 AM (UTC) at SCE_T ~ 302 K

- see next slides

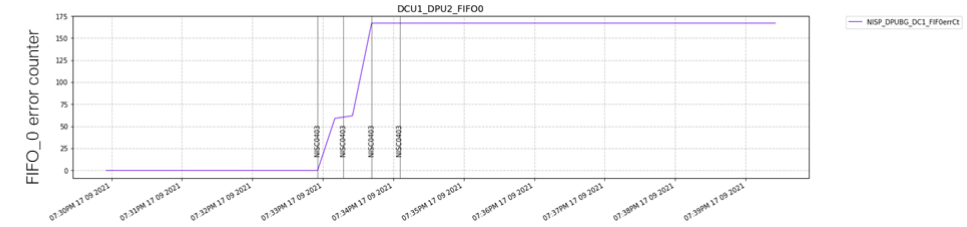
Plots prepared with C. Sirignano



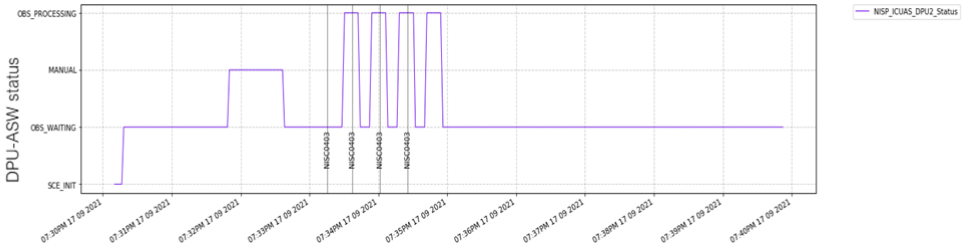
Temperature profile, thanks to Michael Carle - LAM

Tests using DPU ASWv1.3.7.1 injecting FIFO_0 errors @LAM 17/9/2021 Using 1 dither composed by 4 MACC(2,2,2)

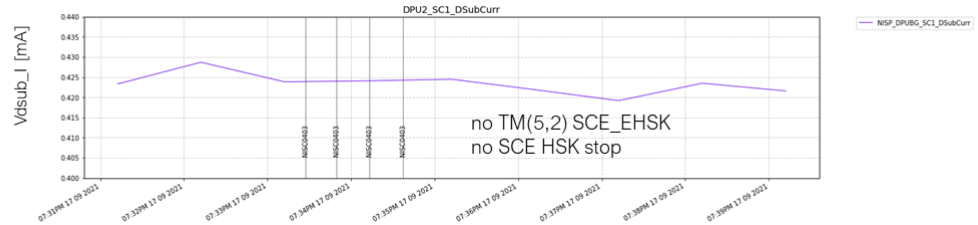
Test case (2)

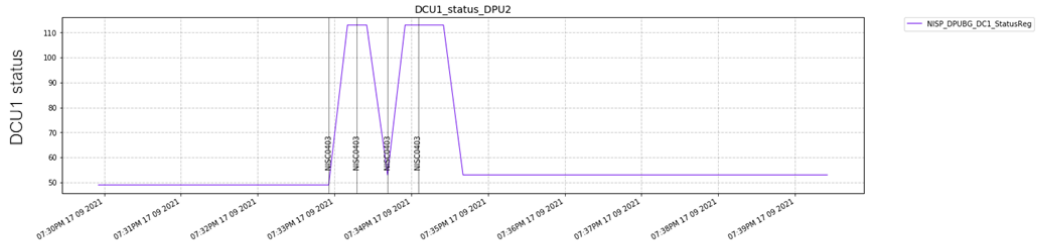


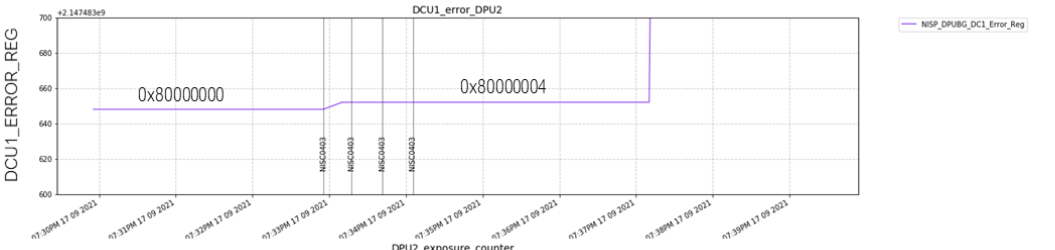
vertical lines indicates EXPOSURE command -NISC0403

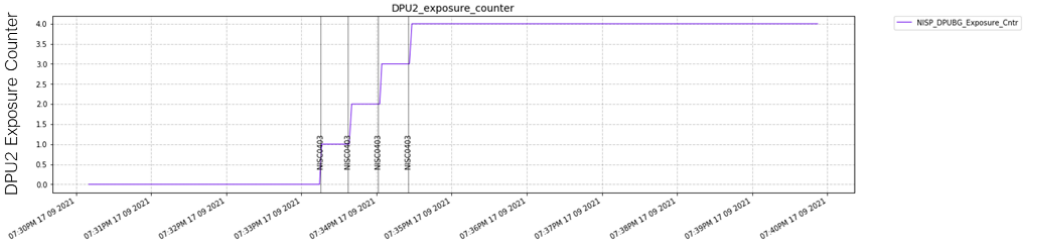


Test case (1)



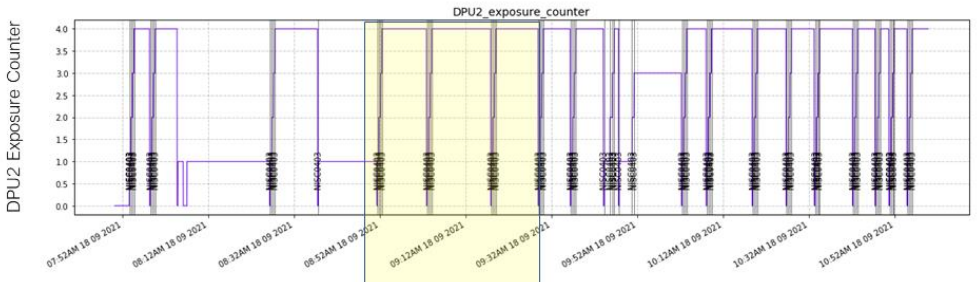






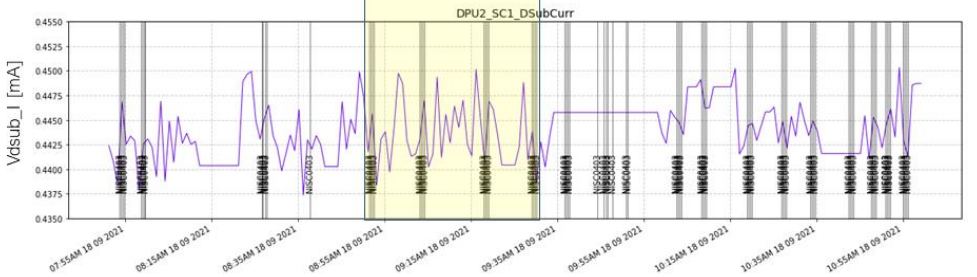
Tests using DPU ASW version 1.3.7.1 injecting FIFO_0 errors @LAM 18/9/2021 using 3xDither of 4xMACC(2,2,2) = 12 broadcast exposures

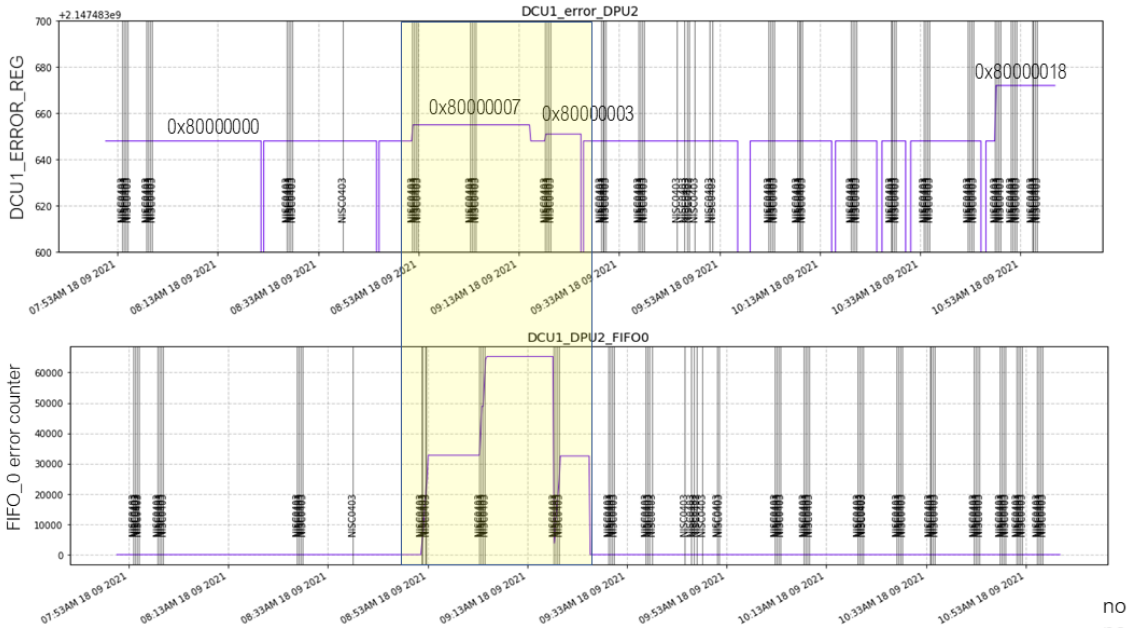
Test case (2)



no exposure errors
no SCE HSK stop
no TM(5,2)

Test case (1)

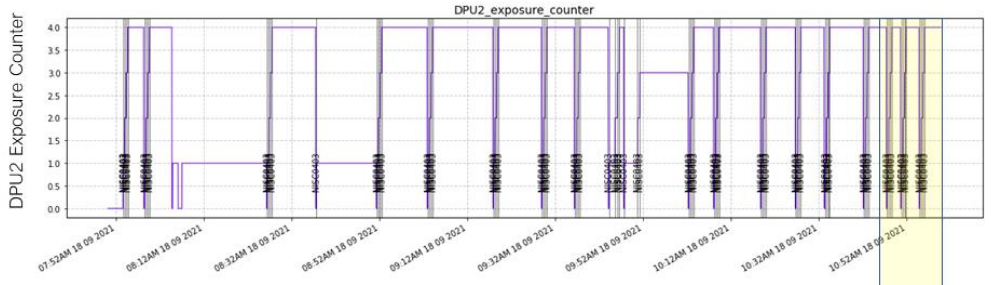




no exposure errors
no SCE HSK stop
no TM(5,2)

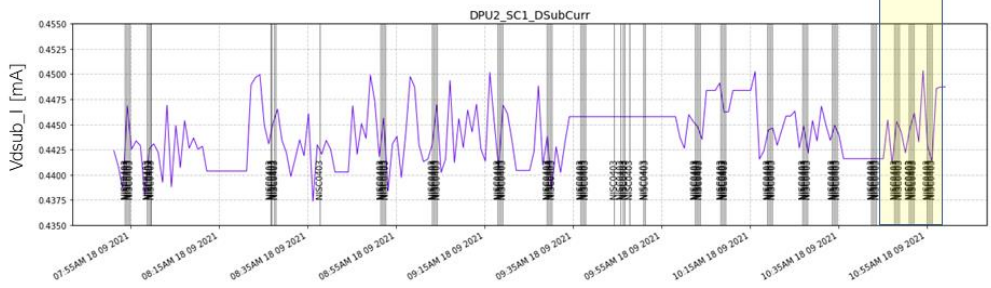
Tests using DPU ASW version 1.3.7.1 injecting FIFO_0 errors @LAM 18/9/2021 using 3xDither of 4xMACC(2,2,2) = 12 broadcast exposures

Test case (2)



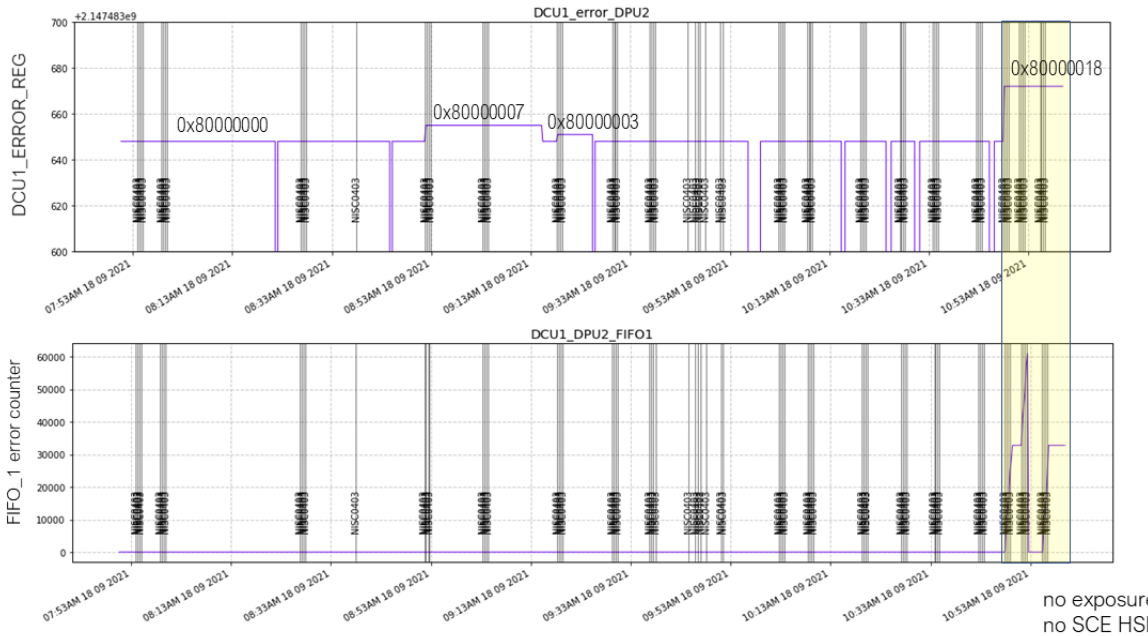
no exposure errors
no SCE HSK stop
no TM(5,2)

Test case (1)



Tests using DPU ASW version 1.3.7.1 injecting FIFO_1 errors @LAM 18/9/2021 using 3xDither of 4xMACC(2,2,2) = 12 broadcast exposures

Test cases (1) and (2)

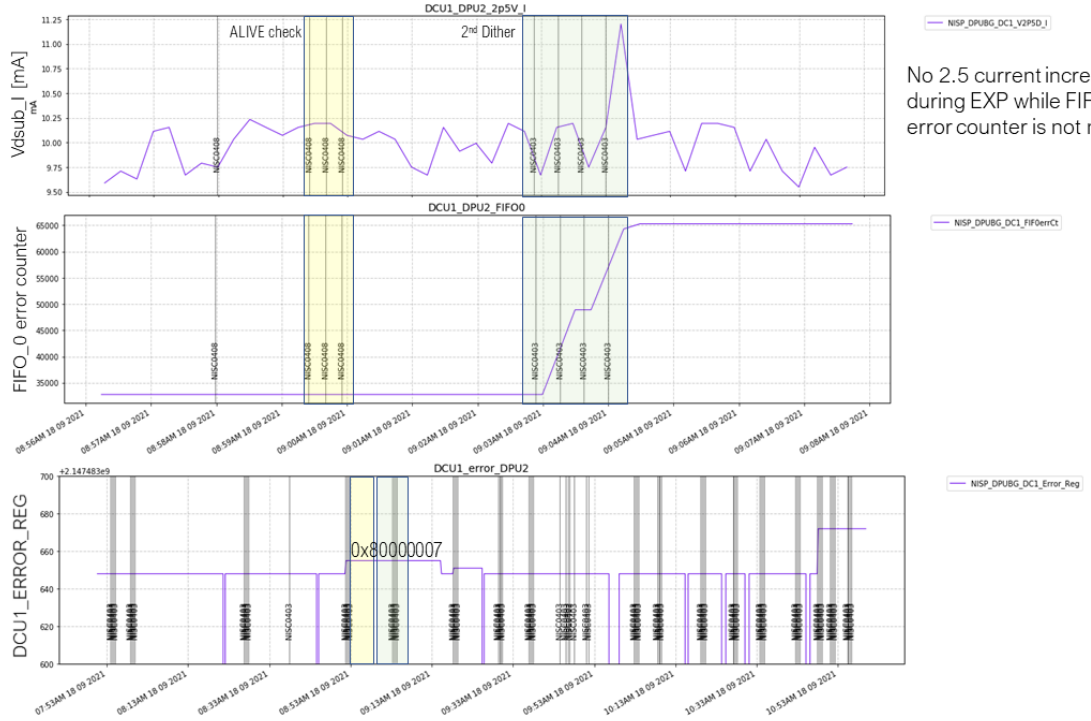


no exposure errors
no SCE HSK stop
no TM(5,2)

Tests using DPU ASW version 1.3.7.1 injecting FIFO_0 errors

Using the sequence: SCE_EXP NISC0403, SCE_ALIVE NISC0408, SCE_EXP NISC0403

Test case (3)

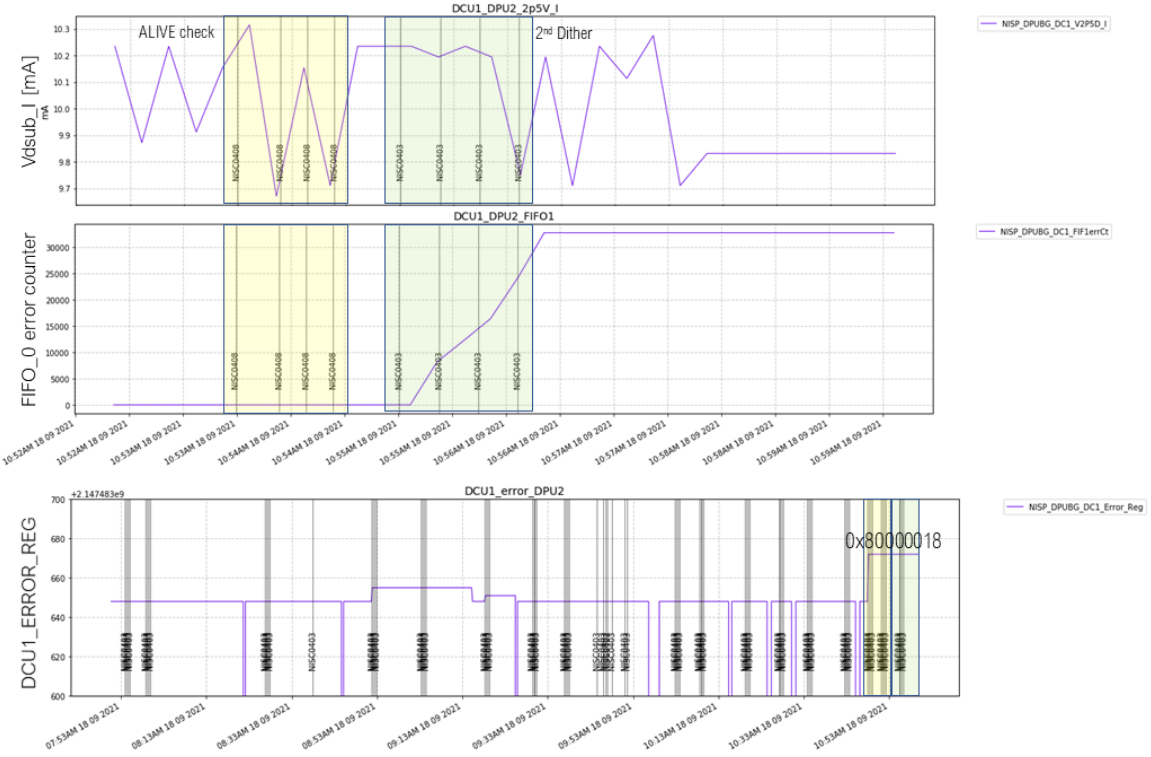


No 2.5 current increase during EXP while FIFO_0 error counter is not null

Tests using DPU ASW version 1.3.7.1 injecting FIFO_1 errors

Using the sequence: SCE EXP NISC0403, SCE ALIVE NISC0408, SCE EXP NISC0403

Test case (3)



Conclusions

The error conditions encountered during CSL's PLM test campaign were solved with the new 'SW concept of design'.
All the scenarios were tested injecting both FIFO_0 and FIFO_1 'errors' independently in a single DCU while operating with 4 DCUs/SCEs.
Verifications were performed at an operational temperature of SCE ~122 K ("cold") and at "warm" SCE ~302 K.

Test Summary:

Test condition: (1) HSK stop during image acquisition, (2) consecutive exposures after any FIFO error occurrence:

- no stop of the SCE HSK retrieval during 4 consecutive broadcast acquisitions using 4 detectors, one reporting FIFO_0 'errors' at cold
- no stop of the SCE HSK retrieval during 12 consecutive broadcast acquisitions using 4 detectors, one reporting FIFO_0 'errors' at warm
- no stop of the SCE HSK retrieval during 12 consecutive broadcast acquisitions using 4 detectors, one reporting FIFO_1 'errors' at warm

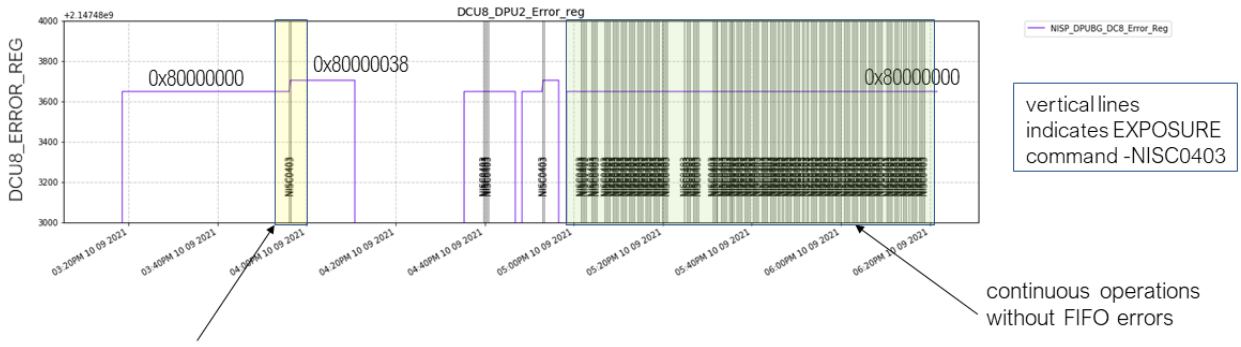
Test condition: (3) ASIC current increase using the sequence: SCE_EXP (with presence of FIFO 'errors'), ALIVENESS check, SCE_EXP

- no SCE current increase (any power line) following the trigger sequence when FIFO_0 'errors' were injected in DCU1, at warm
- no SCE current increase (any power line) following the trigger sequence when FIFO_1 'errors' were injected in DCU1, at warm

Next Activities

A new DPU-ASW version will be prepared considering the 'concept of design' described before. It will also include the new input provided by OHB-I.
This activities will be associated to a SW NCR, therefore OHB inputs and the new SW implementation will be discussed in a NRB.
Further SW investigations related to the new SW version will be held next week at LAM using the same setup at room temperature.

Repetition of the error condition with DPU-ASWv1.3.7 (used at CSL) during LAM setup verification



ERROR CONDITION

Date	status_reg	FIFO_0_err_counter	FIFO_1_err_counter	error_reg
2021-09-10 15:56:41.544000	0x31	0	0	0x80000000
2021-09-10 15:56:56.544000	0xB5	0	7477	0x80000038

First Exposure command 15:56:41
First TM(5,2) and HK4 Freeze 15:56:55

Second exposure command not executed with error code SCE_EEXP } same CSL error condition

Implementation proposal test

Results of the DPU-ASW dedicated tests during the NISP investigation at LAM, 27-30 September 2021

E. Medinaceli DPU-ASW manager for the NISP team

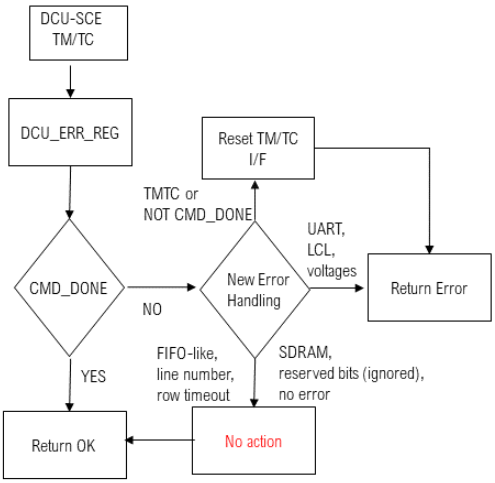
DCU_ERROR_REG treatment implementation

A distinction between different errors and 'warnings' is done according to OHB input:

DCU_ERROR_REG bit	action
- FIFO-like errors (bits 0 - 5)	no action;
- line number (bit 6)	no action;
- row timeout (bit7)	no action;
- TMTC reset (bits 8 – 11)	resetTMTC(); return error;
- UART error, (bits 12 - 14)	return error;
- invalid power supply V (bits 15, 16)	return error;
- LCL protections (bits 17, 18)	return error;
- SDRAM errors, (bits19 - 20)	no action;
- reserved (unused) bits (bits 21 - 30)	no action;
- CMD done (bit31=1)	no action;
- CMD not done (bit 31=0)	resetTMTC(); return error;

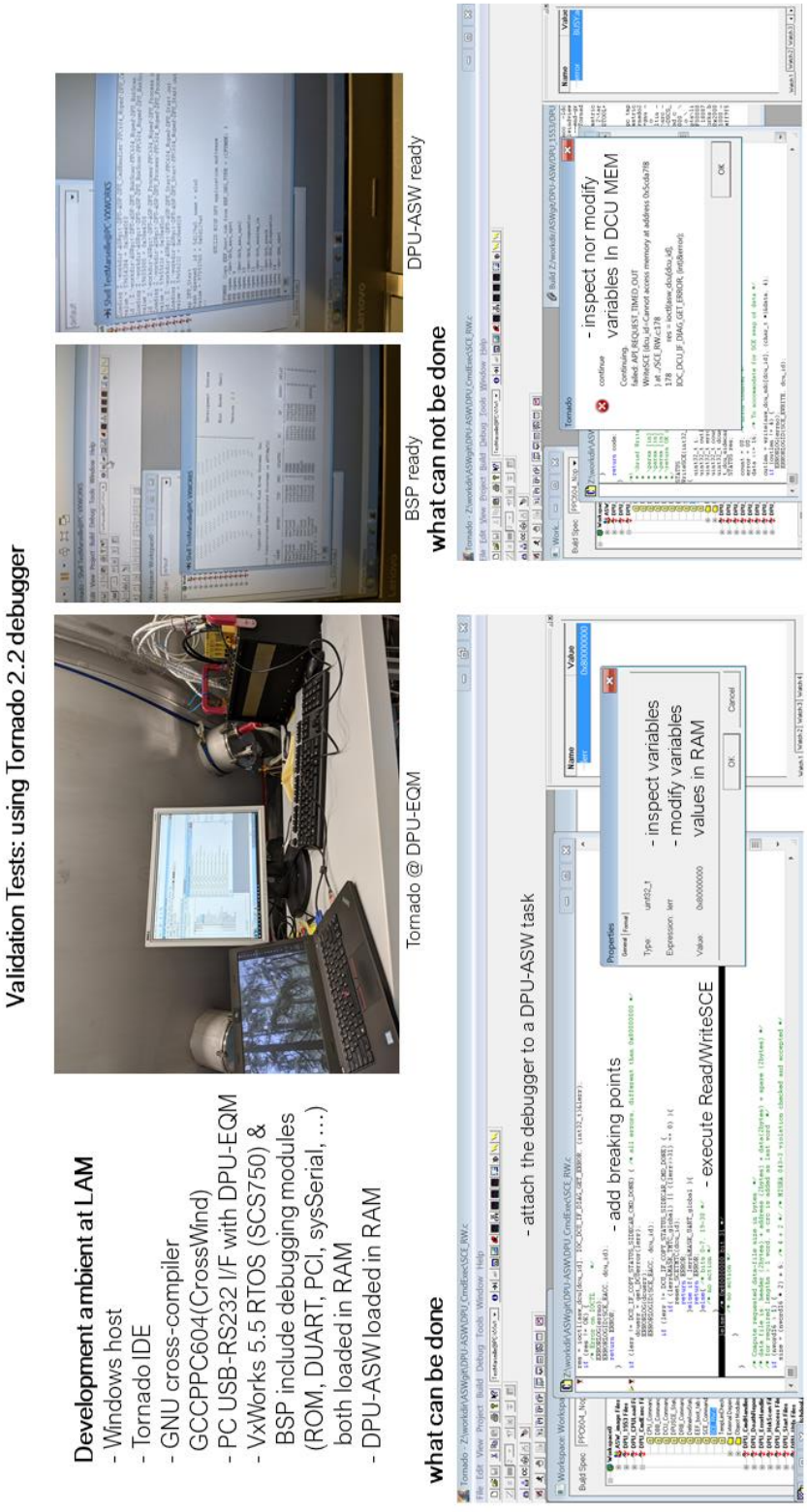
(blue actions differs from the OLD error treatment)

SCE Read & SCE Write internal 'error' handling

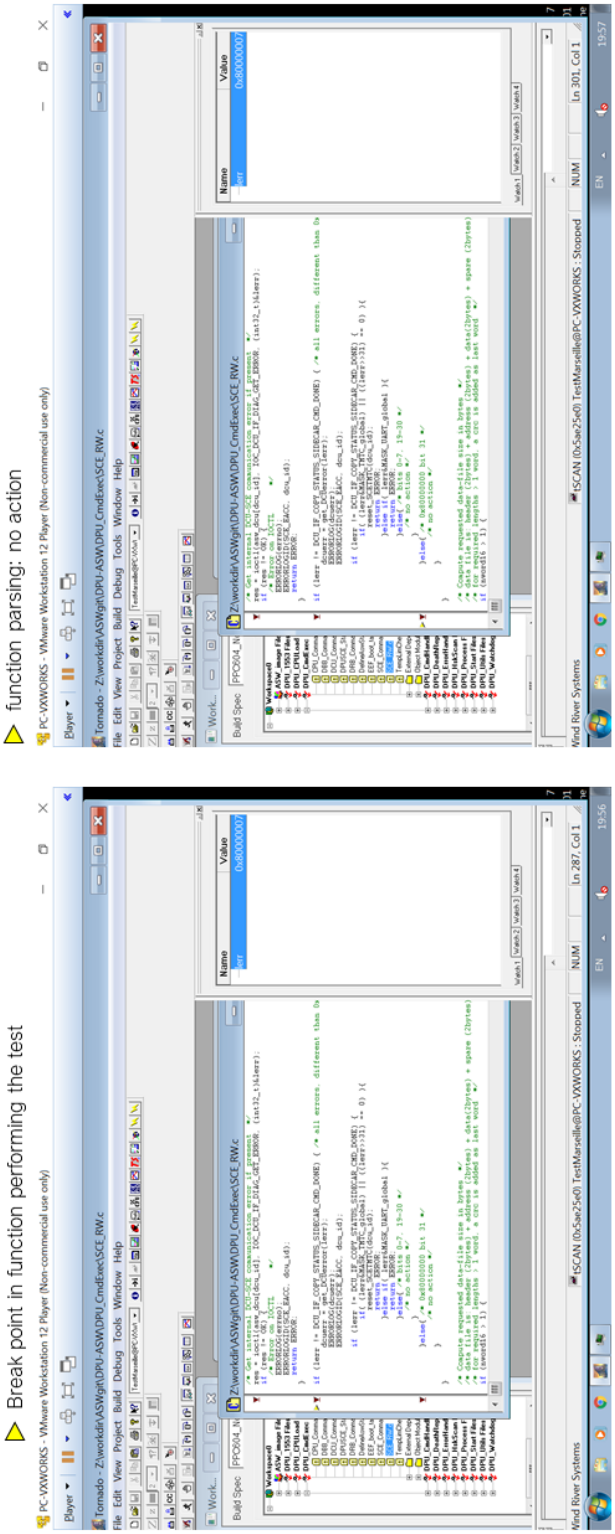


- Severity of HW errors were rediscussed with OHB (10/8/2021)
- The clear of the DCU_ERROR_REG will be done automatically only when the TM/TC I/F is reset internally → could lead to visibility of TM/TC errors
- In all the other cases when an error is returned, the clear of the register is associated to the recovery action
- TM(5,3) will be assigned when SCE_EXP command fails (SCE_KTC_EXP, SCE_IPC_EXP)

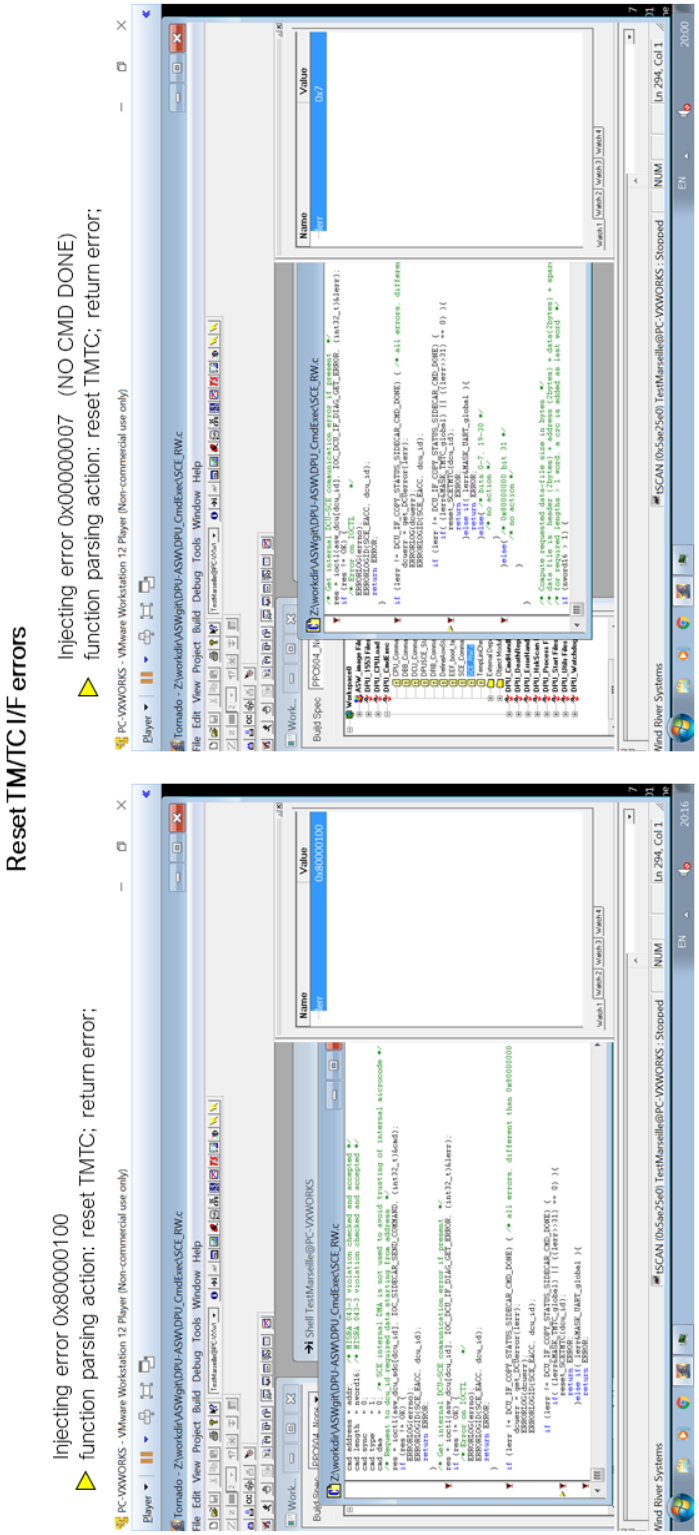
updated since NRB of 9/29/2021



FIFO-like errors example, using error 0x80000007

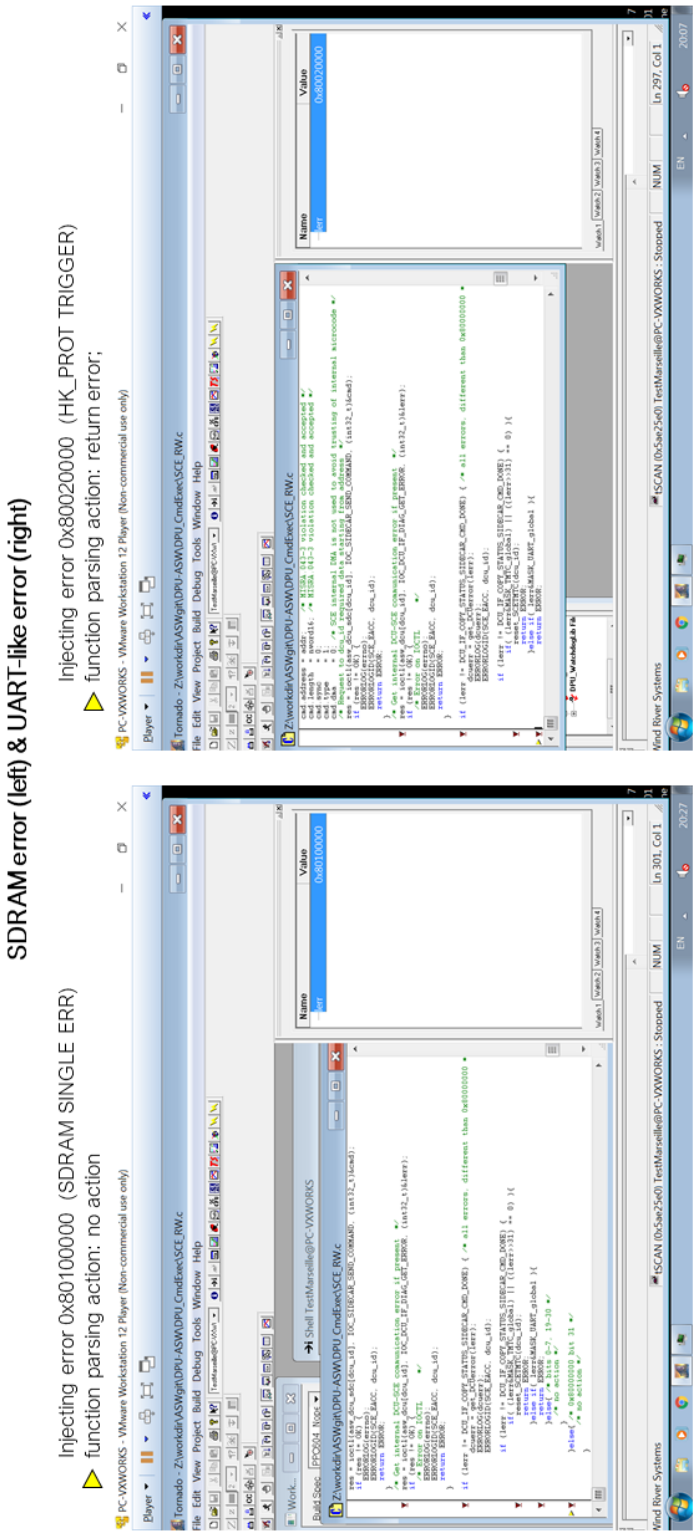


- NO error injected in TLM(5,2) verified
- DCU_ERROR_REG saved in TLM



- Error injected in TLM(5,2) verified
- DCU_ERROR_REG saved in TLM
- Also, the reset of the DCU_ERROR_REG was tested verifying the TLM value after the reset

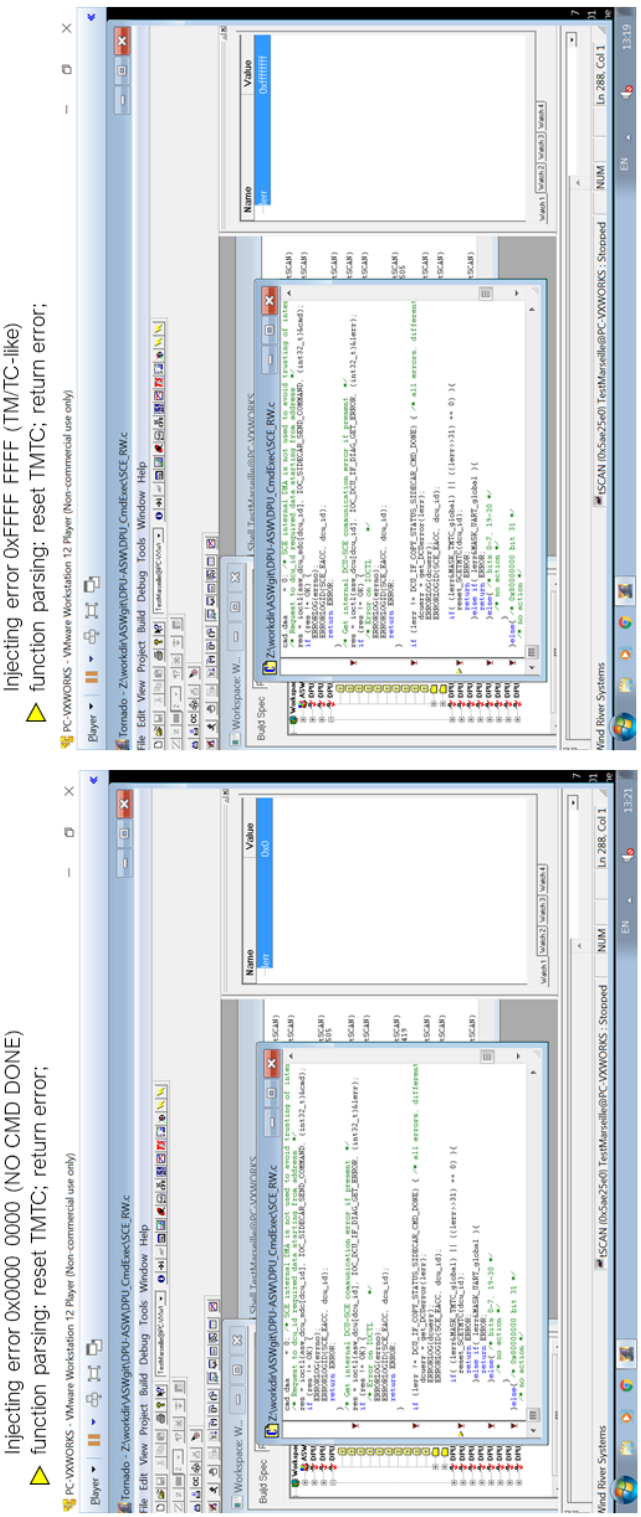
SDRAM error (left) & UART-like error (right)



- NO error injected in TLM(5,2) verified
- DCU_ERROR_REG saved in TLM

- error injected in TLM(5,2) verified
- DCU_ERROR_REG saved in TLM

Unit Test like errors: 0x0000 0000 (left) and 0xFFFF FFFF (right)



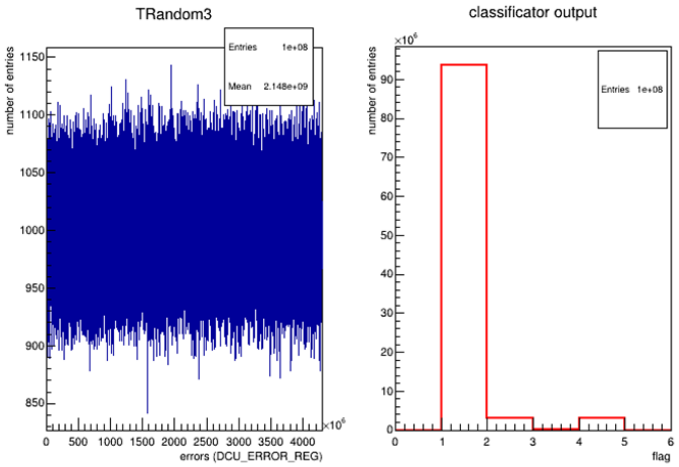
- Error injected in TLM(5,2) verified
- DCU_ERROR_REG saved in TLM
- Also, the reset of the DCU_ERROR_REG was tested verifying the TLM value after the reset

Monte Carlo simulation of error injection and classification

Simulations of DCU_ERROR_REG content

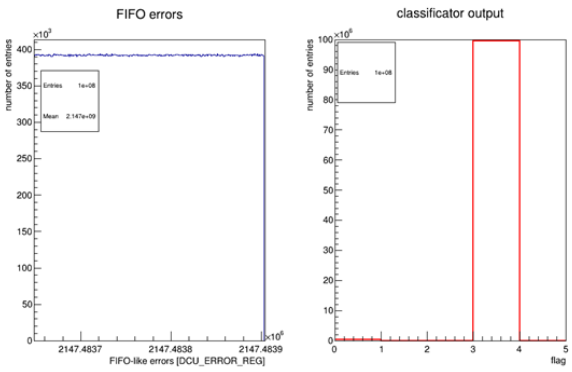
$100x10^6$ events were simulated,
all combinatory (left plot)
between 0x0000000
and 0xFFFFFFFF (all 32 bits)

classificator results (right plot):	action
flag: = 0, 0x80000000 OK	no action
= 1, TMTC reset (bits 8 – 11)	resetTMTC(); return error;
= 2, UART error, (bits 12 - 14)	return error;
invalid voltages, (bits 15, 16)	
protections (bits 17, 18)	
= 3, FIFO-like errors (bits 0 - 5)	no action
line number (bit 6)	
row timeout (bit7)	
= 4, no ACK (bit 31)	resetTMTC(); return error;
= 5, SDRAM errors, (bits19 - 20)	no action
reserved bits (bits 21 - 30)	



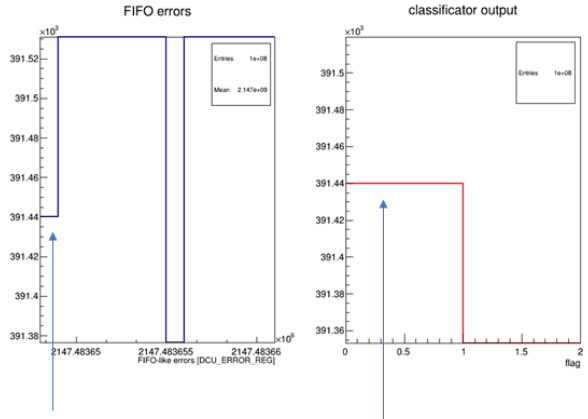
command done + FIFO-like + line number error + row timeout

$100x10^6$ events were simulated, all combinatory – see 'FIFO errors plot' (bottom left)
between 0x8000000 and 0x800000FF = 1000 0000 0000 0000 0000 0000 1111 1111
(command done + FIFO-like + line number error + row timeout)



Classificator results:
flag = 3 ≡ FIFO-like errors (no action)
flag = 0 ≡ command done ok (0x8000000) (no action)

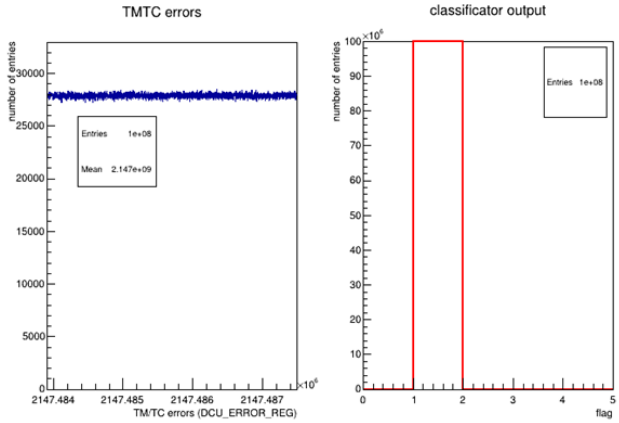
Zoom-in



Entries equal to 0x80000000:
391440 generated as command done
all classified with flag = 0

TM/TC errors

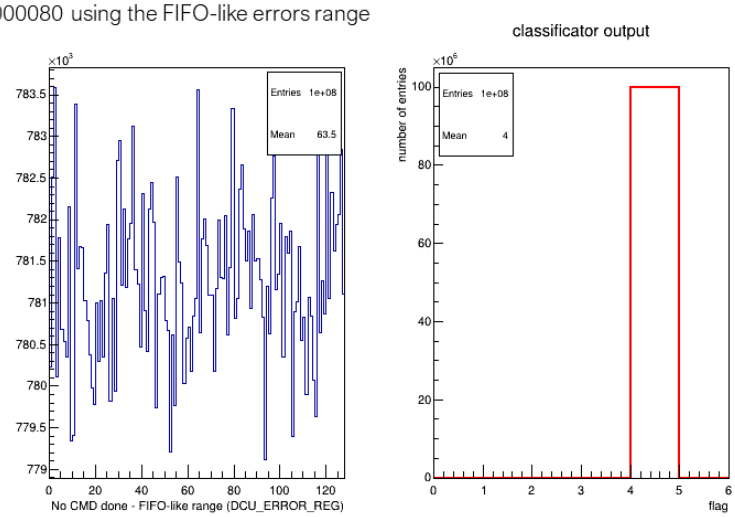
100x10^6 events were simulated (all combinatory – see 'TMTC errors plot')
between 8000 0100 and 8000 0F00 = 1000 0000 0000 0000 0000 0001 0000 0000 - 1000 0000 0000 0000 0000 1111 0000 0000



Classificator results:
flag = 1 ≡ TM/TC errors = return error

No command done errors (using the FIFO-like errors' range)

100x10^6 events were simulated (all combinatory – see 'No CMD done plot')
between 0x00000000 and 0x00000080 using the FIFO-like errors range



Classificator results:
flag = 5 ≡ No CMD done = resetTMTC(); return error;

Appendix B – Unit Tests

The following is the output of the Parasoft C/C++ test suit implementing the unit testing of only functions involved on the DCU_ERROR_REG management:

PARASOFT C/C++test® Report

User configuration: **Load Test Results (Files)**
10/12/21 11:05:34

TEST EXECUTION

Test Project Name	Tasks			Executed Test Cases		
	Fix Runtime Error Detection Violations	Fix Unit Test Problems	Review Unit Test Outcomes	Passed	Failed	Total
DPU_CmdExec	0	0	22	22	0	22
Total [0:00:00]	0	0	22	**22**	**0**	**22**

SCE Read and Write functions were correctly tested with a coverage of 100%

SCE_RW.c -- 100% [123/123 executable lines]
- ReadSCE -- 100% [42/42 executable lines]
- WriteSCE -- 100% [59/59 executable lines]
- reset_SCETMTC -- 100% [22/22 executable lines]

Unit Test report of the ASW functions ReadSCE and WriteSCE after DPU-ASW bug detection

FILE COVERAGE: 100% (123/123 EXECUTABLE LINES)

Notes: source file contains three functions, the total number of tests is divided in the following way
WriteSCE (12 tests)
ReadSCE (6 tests)
Reset_SCETMTC (4 tests)

This document reports updated Unit Tests only for functions WriteSCE and ReadSCE while tests for Reset_SCETMTC are reported in RD-8.

```
STATUS WriteSCE(int32_t dcu_id, uint32_t addr, uint32_t data)
```

FUNCTION COVERAGE: 100% (59/59 EXECUTABLE LINES)

TEST CASE: WriteSCE_1
Stub of the function *outlen = write(asw_dcu_sdc[dcu_id], (char_t *)&data, 4)*
returning outlen = 1
STATUS: -1

TEST CASE: WriteSCE_2
stub of the function *ioctl(asw_dcu_sdc[dcu_id], IOC_SIDECAR_SEND_COMMAND, (int)&cmd)*
returning OK+1 (==1)
STATUS: 1

TEST CASE: WriteSCE_3
dcu_id=5
Stub of the function *ioctl(asw_dcu[dcu_id], IOC_DCU_IF_DIAG_GET_ERROR, (int)&error)*
returning OK+1 (=1)
STATUS: 1

TEST CASE: WriteSCE_4
dcu_id=5
Stub of the *function ioctl(asw_dcu[dcu_id], IOC_DCU_IF_DIAG_GET_ERROR, (int)&error)*
returning OK and assign error=0x64
This means that bit 31 is 0 and verifies the condition
if((error&MASK_TMTC_global) || ((error>>31) == 0))
STATUS: -1

TEST CASE: WriteSCE_5
dcu_id=5
Stub of the *function ioctl(asw_dcu[dcu_id], IOC_DCU_IF_DIAG_GET_ERROR, (int)&error)*
returning OK and assigning error=0x800F0000 (bit 31 and 18-21)
This verifies the condition "if (error&MASK_UART_global)"
STATUS: -1

TEST CASE: WriteSCE_6
dcu_id=5
Stub of the *function ioctl(asw_dcu[dcu_id], IOC_DCU_IF_DIAG_GET_ERROR, (int)&error)*
returning OK and assugning error=0x80000001 (bit 0and 31)
STATUS: 0

TEST CASE: WriteSCE_7
When calling the function *ioctl(asw_dcu[dcu_id], IOC_DCU_IF_DIAG_GET_ERROR, (int)&error)*
assign error=DCU_IF_COPY_STATUS_SIDECAR_CMD_DONE (bit 31)
STATUS: 0

TEST CASE: WriteSCE_8
dcu_id=DPU_BRD_DEV
Stub of the function *ioctl(asw_dcu[i], IOC_DCU_IF_DIAG_GET_ERROR, (int)&error)*
returning -1
STATUS: -1

TEST CASE: WriteSCE_9
dcu_id=DPU_BRD_DEV
Stub of the *function ioctl(asw_dcu[i], IOC_DCU_IF_DIAG_GET_ERROR, (int)&error)*
returning OK and set error=0x0
this satifies condition if((error&MASK_TMTC_global) || ((error>>31) == 0))
STATUS: -1

TEST CASE: WriteSCE_10
dcu_id=DPU_BRD_DEV
Stub of *the function ioctl(asw_dcu[i], IOC_DCU_IF_DIAG_GET_ERROR, (int)&error)*
returning OK and set error 0x80010000
this satisfies the condition
if(error&MASK_UART_global)
STATUS: -1

TEST CASE: WriteSCE_11
dcu_id=DPU_BRD_DEV
Stub of the function *ioctl(asw_dcu[i], IOC_DCU_IF_DIAG_GET_ERROR, (int)&error)*
returning OK and set error=0x80000001 (bit 0 and 31)
STATUS: 0

TEST CASE: WriteSCE_12
dcu_id=DPU_BRD_DEV
Stub of the function *ioctl(asw_dcu[i], IOC_DCU_IF_DIAG_GET_ERROR, (int)&error)*
returning OK and set error=DCU_IF_COPY_STATUS_SIDECAR_CMD_DONE
STATUS: 0

```
STATUS  ReadSCE(int32_t dcu_id, uint32_t addr, uint32_t nword16)
```

FUNCTION COVERAGE: 100% (42/42 EXECUTABLE LINES)

TEST CASE: ReadSCE_1
Stub of the function *ioctl(asw_dcu_sdc[dcu_id], IOC_SIDECAR_SEND_COMMAND, (int32_t)&cmd)*
returning OK+1 (=1)
STATUS: -1

TEST CASE: ReadSCE_2
Stub of the function *ioctl(asw_dcu[dcu_id], IOC_DCU_IF_DIAG_GET_ERROR, (int32_t)&lerr)*
returning OK+1 (=1)
STATUS: -1

TEST CASE: ReadSCE_3
Stub of the *function ioctl(asw_dcu[dcu_id], IOC_DCU_IF_DIAG_GET_ERROR, (int32_t)&lerr)*
returning OK and set lerr=0
STATUS: -1

TEST CASE: ReadSCE_4
Stub of the function *ioctl(asw_dcu[dcu_id], IOC_DCU_IF_DIAG_GET_ERROR, (int32_t)&lerr)*
returning OK and set lerr=0x80010000
This satisfies the condition if(error&MASK_UART_global)
STATUS: -1

TEST CASE: ReadSCE_5
Stub of the function *ioctl(asw_dcu[dcu_id], IOC_DCU_IF_DIAG_GET_ERROR, (int32_t)&lerr)*

returning OK and set lerr=0x80000001
STATUS: 0

TEST CASE: ReadSCE_6
Stub of the *function ioctl(asw_dcu[dcu_id], IOC_DCU_IF_DIAG_GET_ERROR, (int32_t)&lerr)*
returning OK and set lerr=DCU_IF_COPY_STATUS_SIDECAR_CMD_DONE
Stub of the function *outlen = read(asw_dcu_sdc[dcu_id], (char_t *)sidecarData, size)*
returning outlen != size
STATUS -1

Appendix C – Static Tests – Cross Compiling & CLANG analyzer

The following is the output of the Jenkins suit implementing the cross-compilation and CLANG analyzer results concerning functions involved on the DCU_ERROR_REG management:

Started by user JenkinsUser
Running as SYSTEM
Building in workspace /var/lib/jenkins/workspace/DPU ASW
[WS-CLEANUP] Deleting project workspace...
[WS-CLEANUP] Done
using credential 21ab0769-a9c3-45d3-8acd-cf3761da9905
Cloning the remote Git repository
Cloning repository https://baltig.infn.it/euclid/DPU-ASW.git
 > git init /var/lib/jenkins/workspace/DPU ASW # timeout=10
Fetching upstream changes from https://baltig.infn.it/euclid/DPU-ASW.git
 > git --version # timeout=10
using GIT_ASKPASS to set credentials
 > git fetch --tags --progress -- https://baltig.infn.it/euclid/DPU-ASW.git +refs/heads/*:refs/remotes/origin/* # timeout=10
 > git config remote.origin.url https://baltig.infn.it/euclid/DPU-ASW.git # timeout=10
 > git config --add remote.origin.fetch +refs/heads/*:refs/remotes/origin/* # timeout=10
 > git config remote.origin.url https://baltig.infn.it/euclid/DPU-ASW.git # timeout=10
Fetching upstream changes from https://baltig.infn.it/euclid/DPU-ASW.git
using GIT_ASKPASS to set credentials
 > git fetch --tags --progress -- https://baltig.infn.it/euclid/DPU-ASW.git +refs/heads/*:refs/remotes/origin/* # timeout=10
 > git rev-parse refs/remotes/origin/master^{commit} # timeout=10
 > git rev-parse refs/remotes/origin/origin/master^{commit} # timeout=10
Checking out Revision 896839296cd9937541143828acbf3d3deec6c96c (refs/remotes/origin/master)
 > git config core.sparsecheckout # timeout=10
 > git checkout -f 896839296cd9937541143828acbf3d3deec6c96c # timeout=10
Commit message: "added TM(5,3) errors to SCE_EXP, SCE_KTCEXP, and SCEIPCEXP"
 > git rev-list --no-walk c3214ad43fb91412f8ee78d0f3a6bd51acab2ad3 # timeout=10
[DPU ASW] $ /bin/bash /tmp/jenkins8135916845810769394.sh

==========

```
[DPU ASW] $ /bin/bash /tmp/jenkins2206036317123280442.sh
powerpc-linux-gnu-gcc-6 -c -o SCE_RW.o -g -O0 -mcpu=604 -mstrict-align -ansi -fno-builtin -I. -
I/home/andrea/Tornado2.2/target/h/ -I./../DPU_Include -I./../BasicSW/BSP -I./../BasicSW/HDSW/include -
DCPU=PPC604 -DTOOL_FAMILY=gnu -DTOOL=gnu -mlongcall -Wall -Wextra -Wuninitialized -Wshadow -Wundef -
Wformat=2 -Wlogical-op -fstrict-overflow -Wstrict-overflow=5 -Wcast-qual -Wdisabled-optimization -Winit-self -
Wmissing-include-dirs -Wredundant-decls -fdiagnostics-show-option -mcpu=604 -mstrict-align SCE_RW.c

============================================================

scan-build: Using '/usr/lib/llvm-3.8/bin/clang' for static analysis
/usr/share/clang/scan-build-3.8/bin/../libexec/ccc-analyzer -c -o SCE_RW.o -g -O0 -mcpu=604 -mstrict-align -ansi -
fno-builtin -I. -I/home/andrea/Tornado2.2/target/h/ -I./../DPU_Include -I./../BasicSW/BSP -
I./../BasicSW/HDSW/include -DCPU=PPC604 -DTOOL_FAMILY=gnu -DTOOL=gnu -mlongcall -Wall -Wextra -
Wuninitialized -Wshadow -Wundef -Wformat=2 -Wlogical-op -fstrict-overflow -Wstrict-overflow=5 -Wcast-qual -
Wdisabled-optimization -Winit-self -Wmissing-include-dirs -Wredundant-decls -fdiagnostics-show-option -
mcpu=604 -mstrict-align SCE_RW.c

scan-build: Removing directory '/tmp/scan-build-2021-10-13-162207-3419-1' because it contains no reports.
scan-build: No bugs found.

============================================================

Finished: SUCCESS
```

The overall results for DPU-ASWv1.3.8 can be found at:

https://euclid.baltig-pages.infn.it/DPU-ASW/Jenkins/JenkinsBuildDPU-ASWv1.3.8.txt

Appendix D – Static Test - Polyspace test suit analysis

The following is the output of Polyspace results concerning functions involved on the DCU_ERROR_REG management:

Summary By File

File	Defects (Reviewed)
C:\Users\e_med\workdir\EUCLID\ASW_TAGS\ASWv1.3.8\DPU_CmdExec\SCE_RW.c	0 (0)

```
Verifying sources compliance ...
************************************************************
*** Verifying GNU C 6.x sources
************************************************************
Verifying sources ...
Verifying SCE_RW.c (37/54)
************************************************************
```

*** Bug-finder - Global analysis done
****

Ending at: Thu Oct 14 16:03:03 2021
User time for global: 00:00:04.20 (4.2real, 4.2u + 0s (0.08gc))
User time for polyspace-bug-finder: 00:07:39 (459.39real, 459.39u + 0s (0.61gc))

*** End of Polyspace analysis

The overall results for DPU-ASWv1.3.8 can be found at:

https://euclid.baltig-pages.infn.it/DPU-ASW/Polyspace/ASW_test_BugFinder.html

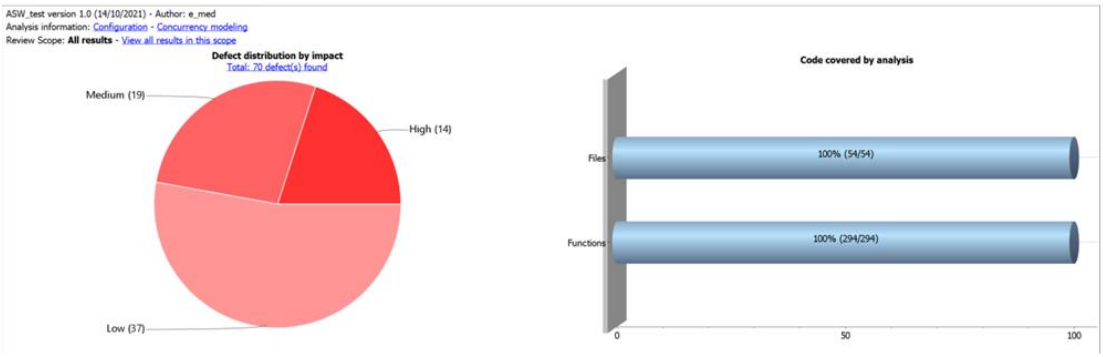
Global summary:

Polyspace Bug Finder

Detailed Report for Project: ASW_test

Report Author: e_med

Analysis Author(s): e_med
Polyspace Version(s): Polyspace Bug Finder 3.3 (R2020b Update 5)
Project Version(s): 1.0



- all cases already documented/discussed for DPU-ASWv1.3.7 in EUCL-OPD-PL-7-005 DPU ASW Static & Unit Test Report
- none related to the new implementation

Analysis Author(s): e_med Polyspace Version(s): Polyspace Bug Finder 3.3 (R2020b Update 5) Project Version(s): 1.0 Result Folder(s): C:\Users\e_med\Documents\Polyspace_Workspace\ASW_test\Module_1\BF_Result

Published 14-Oct-2021 16:21:05

Appendix E – Software Configuration Control entries

[EUNIDPUASW-165] new DCU_ERROR_REG handling Created: 11/Oct/21 Updated: 14/Oct/21 Resolved: 11/Oct/21

Status:	Done
Project:	Application software for the Euclid NISP DPU
Component/s:	Commands
Affects Version/s:	DPU-ASW_v1.3.7.1
Fix Version/s:	DPU-ASW_v1.3.7.2

Type:	New Feature	**Priority:**	Minor
Reporter:	Eduardo Medinaceli Villegas	**Assignee:**	Eduardo Medinaceli Villegas
Resolution:	Done	**Votes:**	0
Labels:	None		
Remaining Estimate:	Not Specified		
Time Spent:	Not Specified		
Original Estimate:	Not Specified		

Attachments:	2ndTestSession@LAM.pdf

Description

Function get_DCUerror(uint32_t errorreg) called in DCU_ERROR_REG test done in ReadSCE and WriteSCE removed.

Codes removed:

```
#define IOCDCU_ERROR_PREFIX 0xA6800000 /* ASW DCU error prefix */
#define IOCDCU_IF_ERROR_OP_NOT_COMPLETED (IOCDCU_ERROR_PREFIX) /* SCE Operation not completed */
#define IOCDCU_IF_ERROR_SDRAM_SINGLE_ERR (IOCDCU_ERROR_PREFIX + (1U << 19U)) /* 0xA6880000 */
#define IOCDCU_IF_ERROR_SDRAM_DOUBLE_ERR (IOCDCU_ERROR_PREFIX + (1U << 18U)) /* 0xA6840000 */
#define IOCDCU_IF_ERROR_HK_PROT_TRIGGERED (IOCDCU_ERROR_PREFIX + (1U << 17U)) /* 0xA6820000 */
#define IOCDCU_IF_ERROR_INVALID2V5D (IOCDCU_ERROR_PREFIX + (1U << 16U)) /* 0xA6810000 */
#define IOCDCU_IF_ERROR_INCVALID_VDDA (IOCDCU_ERROR_PREFIX + (1U << 15U)) /* 0xA6808000 */
#define IOCDCU_IF_ERROR_UART_CRC (IOCDCU_ERROR_PREFIX + (1U << 14U)) /* 0xA6804000 */
#define IOCDCU_IF_ERROR_UART_SYNC (IOCDCU_ERROR_PREFIX + (1U << 13U)) /* 0xA6802000 */
#define IOCDCU_IF_ERROR_UART_TIMEOUT (IOCDCU_ERROR_PREFIX + (1U << 12U)) /* 0xA6801000 */
#define IOCDCU_IF_ERROR_TMTC_CRC (IOCDCU_ERROR_PREFIX + (1U << 11U)) /* 0xA6800800 */
#define IOCDCU_IF_ERROR_TMTC_PARITY (IOCDCU_ERROR_PREFIX + (1U << 10U)) /* 0xA6800400 */
#define IOCDCU_IF_ERROR_TMTC_TIMEOUT (IOCDCU_ERROR_PREFIX + (1U << 9U)) /* 0xA6800200 */
```

```
#define IOCDCU_IF_ERROR_TMTC_ACK_PULSE (IOCDCU_ERROR_PREFIX + (1U << 8U)) /* 0xA6800100 */
#define IOCDCU_IF_ERROR_ROW_TIMEOUT (IOCDCU_ERROR_PREFIX + (1U << 7U)) /* 0xA6800080 */
#define IOCDCU_IF_ERROR_LINE_NUMBER (IOCDCU_ERROR_PREFIX + (1U << 6U)) /* 0xA6800040 */
#define IOCDCU_IF_ERROR_SCIENCE_CRC_1 (IOCDCU_ERROR_PREFIX + (1U << 5U)) /* 0xA6800020 */
#define IOCDCU_IF_ERROR_INVALID_LEN_1 (IOCDCU_ERROR_PREFIX + (1U << 4U)) /* 0xA6800010 */
#define IOCDCU_IF_ERROR_SCIENCE_SYNC_1 (IOCDCU_ERROR_PREFIX + (1U << 3U)) /* 0xA6800008 */
#define IOCDCU_IF_ERROR_SCIENCE_CRC_0 (IOCDCU_ERROR_PREFIX + (1U << 2U)) /* 0xA6800004 */
#define IOCDCU_IF_ERROR_INVALID_LEN_0 (IOCDCU_ERROR_PREFIX + (1U << 1U)) /* 0xA6800002 */
#define IOCDCU_IF_ERROR_SCIENCE_SYNC_0 (IOCDCU_ERROR_PREFIX + (1U << 0U)) /* 0xA6800001 */
```

Comments

Comment by Eduardo Medinaceli Villegas [11/Oct/21]

- Tested new implementation of the DCU_ERROR_REG handling using bitmasks
  see attached slides [^2ndTestSession@LAM.pdf]

Comment by Eduardo Medinaceli Villegas [14/Oct/21]

ReadSCE and WriteSCE (broadcast and non) functions of SCE_RW.C implements the following test over the DCE_ERROR_REG:

```
/* DCU_ERROR_REG handling*/
MASK_TMTC_global = 0x00000F00; /* bits 8-11 */
MASK_UART_global = 0x0007F000; /* bits 12-18 UART, Voltages, protections */

if (error != DCU_IF_COPY_STATUS_SIDECAR_CMD_DONE) { /* all errors, different than 0x80000000 */
  if( (error&MASK_TMTC_global) || ((error>>31) == 0) )
  {
    reset_SCETMTC(dcu_id); ERRORLOGID(SCE_EACC, dcu_id); return ERROR;
  }
  else if( error&MASK_UART_global )
  {
    ERRORLOGID(SCE_EACC, dcu_id); return ERROR;
  }
  else{ /* bits 0-7, 19-30 */
    /* no action */
 }
} else { /* 0x80000000 bit 31 */
/* no action */
}
```

[EUNIDPUASW-166] errors associated to EXPOSURE commands

Created: 11/Oct/21 Updated: 14/Oct/21 Resolved: 11/Oct/21

Status:	Done
Project:	Application software for the Euclid NISP DPU
Component/s:	error_handling
Affects Version/s:	DPU-ASW_v1.3.7.2
Fix Version/s:	DPU-ASW_v1.3.8

Type:	New Feature	**Priority:**	Major
Reporter:	Eduardo Medinaceli Villegas	**Assignee:**	Eduardo Medinaceli Villegas
Resolution:	Done	**Votes:**	0
Labels:	None		
Remaining Estimate:	Not Specified		
Time Spent:	Not Specified		
Original Estimate:	Not Specified		

Attachments:	DCU_ERR_REG_handling_UML.gif FinalStrategy.png

Description

added TM(5,3) errors to SCE_EXP, SCE_KTCEXP, and SCEIPCEXP

Comments

Comment by Eduardo Medinaceli Villegas [14/Oct/21]

Injected TM(5,3) in:

```
ASWFullErrorLog(errno, sce_ID, taskIdSelf(), cmd1553ID, SCE_EIPCEXP, crit_high); /* SCE_IPCEXP */
ASWFullErrorLog(errno, sce_ID, taskIdSelf(), cmd1553ID, SCE_ETEXP, crit_high); /* SCE_ETEXP */
ASWFullErrorLog(errno, sce_ID, taskIdSelf(), cmd1553ID, SCE_EEXP, crit_high); /* SCE_EXP */
ASWFullErrorLog(errno, sce_ID, taskIdSelf(), cmd1553ID, SCE_EKTCEXP, crit_high); /* SCE_KTCEXP */
```