1. Slow Control Test: Voltage and Current check

The *test.cgi* calls the executable called *get_i_v* which has as output a JSON file in which all the current and voltage values are reported (the data that come out with the command are reported slowc -a). In this example the executable does not take any arguments as input.

The JSON response consists of:

Item of JSON file	**MEANING**
PMT1_HVM	Value of the PMT1 voltage
PMT1_CM	Value of the PMT1 current
PMT1_TM	T value of the PMT1
PMT2_HVM	Value of the PMT2 voltage
PMT2_CM	Value of the PMT2 current
PMT2_TM	T value of the PMT2
PMT3_HVM	Value of the PMT3 voltage
PMT3_CM	Value of the PMT3 current
PMT3_TM	T value of the PMT3
PMT4_HVM	Value of the PMT4 voltage

PMT4_CM	Value of the PMT4 current
PMT4_TM	T value of the PMT4
PMT5_HVM	Value of the PMT5 voltage
PMT5_CM	Value of the PMT5 current
PMT5_TM	T value of the PMT5
PMT6_HVM	Value of the PMT6 voltage
PMT6_CM	Value of the PMT6 current
PMT6_TM	T value of the PMT6
V_1V0	Value of the 1V0 voltage
I_1V0	Value of the 1V0 current
V_1V2	Value of the 1V2 voltage
I_1V2	Value of the 1V2 current
V_1V8	Value of the 1V8 voltage
I_1V8	Value of the 1V8 current
V_3V3	Value of the 3V3 voltage
I_3V3	Value of the 3V3 current
I_3V3_SC	Value of the 3V3 SC current

V_AN_P5V	Value of the AN P5V voltage
I_P5V_ANA	Value of the AN P5V current
V_N5V_ANA	Value of the N5V ANA voltage
I_N5V_ANA	Value of the N5V ANA current
V_GPS_5V	Value of the GPS voltage
I_GPS_5V	Value of the GPS current
V_RADIO_12V	Value of the RADIO voltage
I_RADIO_12V	Value of the RADIO current
V_PMTS_12V	Value of the PMTS voltage
I_PMTS_12V	Value of the PMTS current
V_EXT1_24V	Value of the EXT1 voltage
V_EXT2_24V	Value of the EXT2 voltage
I_V_INPUTS	Value of the I_V_INPUTS current
BAT1	Values BAT1 of the signal at the TPCB connector
BAT2	Values BAT2 of the signal at the TPCB connector
EXT_TEMP	Values EXT_TEMP of the signal at the TPCB connector
BAT_CENT	Values BAT_CENT of the signal at the TPCB connector

BAT_OUT	Values BAT_OUT of the signal at the TPCB connector
LOADCURR	Values LOADCURR current (TPCB)
SP_VOLT	Values of SP VOLT voltage (TPCB)
SP_CURR	Values of SP CURR current (TPCB)
P12V_LI	
P12V_HI1	
P12V_HI2	
P12V_HI3	
T_AIR	Values of T of air
P	Values of P of air
TW	Values of TW (this value needs to be multiplied by 0.1K)

In the case where it is necessary to divide the values related to the PMT by the fixed values of the low-voltage, two different executables can be created.

2. Slow Control Test: HV_Test

The *test.cgi* calls the executable *HV_Test*, this routine sets all DACs to zero and requests the read-back voltages for each PMTs output. Then it sets the DACs to maximum value and requests the voltages again. The difference of two values is returned in JSON format for each PMTs:

Item of JSON file	**MEANING**

PMT1_HVM_diff	Difference related to PMT1
PMT2_HVM_diff	Difference related to PMT2
PMT3_HVM_diff	Difference related to PMT3
PMT4_HVM_diff	Difference related to PMT4
PMT5_HVM_diff	Difference related to PMT5
PMT6_HVM_diff	Difference related to PMT6

3. Slow Control Test: fpga version

The *test.cgi* calls the executable *get_fpga_version* which reads the FPGA version from the register ID_REG_ADDR and outputs a JSON file with the following item:

Item of JSON file	**MEANING**
VERSION	Date of last compilation as reported in the FPGA records.

4. FPGA interface Test: test of GPS

The *test.cgi* calls the executable GPS_serial that checks the GPS serial port. It has as output a JSON file which has only one item that changes depending on the result of the check:

Item of JSON file	**MEANING**
message	Ok Ok
message	failed message

5. Usb interface Test: umount_usb

The *test.cgi* calls the executable *umount_usb*: which has no input parameters and replies with a JSON file that can have three kinds of message:

Item of JSON file	**MEANING**
message	/usb alredy unmounted (if the usb is already unmounted)
message	/usb unmounted (if the usb is correctly unmounted)

6. Usb interface Test: mount_usb

The *test.cgi* calls the executable *mount_usb*: which has no input parameters and replies with a JSON file that can have three kinds of message:

Item of JSON file	**MEANING**
message	/usb already mounted (if the usb is already mounted)
message	OK (if the usb is correctly mounted)
message	ERROR (if an error occurred in mounting usb)

7. Front-end Test: signal test.

The *test.cgi* calls the executable record that which has as input the values of the thresholds on the PMTs and responds with a JSON file that shows the following values for each ADC:

Item of JSON file	**MEANING**
baseline	Baseline average value for each adc.
RMS	Baseline rms value for each adc.
Pulsepos	Value of pulse position for each adc.
valuepeak	Value of peak for each adc.

8. FPGA interface Test: Trigger test.

The *test.cgi* calls the executable Trig_test that which has as input the number of pulse and replies with a JSON file:

Item of JSON file	**MEANING**
message	passed
message	overflow(range is [1...255])
message	failed

9. FPGA interface Test: test of Radio

The *test.cgi* calls the executable Radio_serial that checks the Radio serial port. It has as output a JSON file which has only one item that changes depending on the result of the check:

Item of JSON file	**MEANING**
message	Ok Ok
message	failed message