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
V_EXT1_24V	**Value of the voltage of EXT1**
I_V_INPUTS	**Value of the current of EXT1**
SP_VOLT	**Value of the SP voltage**
SP_CURR	**Value of the SP current**
V_1V0	**Value of the 1V voltage**
I_1V0	**Value of the 1V 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_SC	**Value of the 3V3 current**
V_GPS_5V	**Value of the GPS voltage**
I_GPS_5V	**Value of the GPS current**
V_RADIO_12V	**Value of the Radio 12V voltage**
I_RADIO_12V	**Value of the Radio 12V current**
V_PMTS_12V	**Value of the PMTs 12V voltage**
I_PMTS_12V	**Value of the PMTs 12V current**
LOADCURR	**Value of the Load current**
PMT1_HVM	**Value of the PMT1 voltage**
PMT1_CM	**Value of the PMT1 current**
PMT2_HVM	**Value of the PMT2 voltage**
PMT2_CM	**Value of the PMT2 current**
PMT3_HVM	**Value of the PMT3 voltage**
PMT3_CM	**Value of the PMT3 current**
PMT4_HVM	**Value of the PMT4 voltage**
PMT4_CM	**Value of the PMT4 current**

PMT5_HVM	**Value of the PMT5 voltage**
PMT5_CM	**Value of the PMT5 current**
PMT6_HVM	**Value of the PMT6 voltage**
PMT6_CM	**Value of the PMT6 current**

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: 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.

3. 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

4. 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)

5. 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)

6. 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.