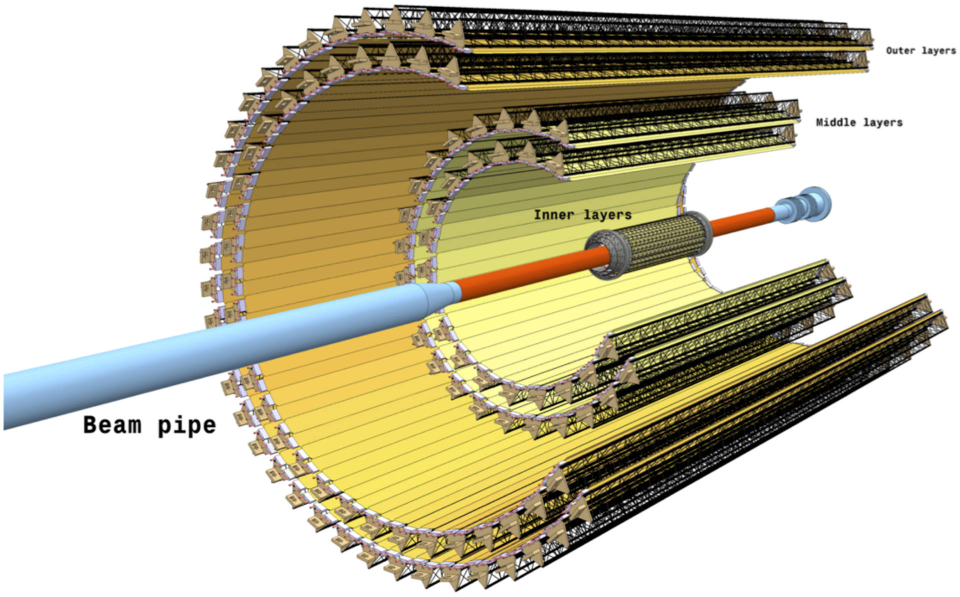
ALICE ITS Upgrade
HIC and Stave Testing Manual

27th August 2018

Abstract

This document serves as manual for the testing of the HICs and Staves of the ALICE ITS Upgrade. The first section describes the procedure to be followed for all tests; the second section gives a detailed description of all scans performed in the tests, including output data and possible problems. In the appendix the software setup procedure and simple database routines are described.



Contents

1 Procedure

1.1 Testing Steps

The following list summarises the different steps that have to be performed for the test of each single device (HIC, half-stave or stave). The list assumes that the setup has been prepared correctly.

1. Verify that the device exists in the database
2. Verify that all previous activities are present in the database history of the device (cf. Section B.2). If previous activities are missing / not closed, investigate the reason with the responsible site and have the activities created / closed if possible. If not possible, report the problem and postpone the testing.
3. Connect the device to the test setup. Be particularly careful with the cross-cables when connecting the power adaptors and with the delicate Firefly connectors.
4. Double-check the connections.
5. Run the appropriate test from the GUI.
6. After successful running of the test program select to write the result to the database.
7. For each HIC you will get a window showing the previous classification (i.e. classification from previous test, *before* starting this test program), the classification of each scan, including the cuts that led to this classification and the resulting classification *after* this test. Make sure that you understand *why* your HIC has received a given classification.
8. If available, check the pdf-output for each HIC and scan, in particular for scans that are not classified as gold. (Reference plots to follow)
9. While writing to the database, check the terminal for a login prompt. The login should show your service account name and is required for the copying of the data to eos.
10. After writing to the database, check that the activity has been created (possibly before closing the GUI).
11. Check that the activity has the correct components attached (input and output).
12. Check that the result corresponds to the classification shown in the GUI and that the activity has been closed.
13. Check that the data has been copied to eos correctly. This can be done through the link in the database entry. (After the installation of the offline script and chron job this step can be substituted by a periodical check on eos).

During the execution of the test please keep a log containing *at least* any abnormal events (e.g. exceptions or error messages) and any deviations from the standard procedure.

1.2 Investigating and Reporting Problems

All problems in the execution of the test or storing of the data have to be reported to the bug report mailing list
`alice-its-qa-software-bugreport@cern.ch`.

In order to find the source of the problem, please try to give as complete information as possible:

- Exact description of the problem, including error messages.
- Which test where you doing, when the problem occured.
- At which stage of the test did the problem occur
- Complete terminal output (available in log file `gui_<date>_<time>.log`)
- Config file used
- Any abnormal message from the test software, both in the GUI and in the terminal window

The mailing list can also be used for technical questions that are not strictly bug reports. Since the mailing list contains all code developers it is the best way to obtain a fast answer.

2 Test Description

This section gives details about each scan performed on both the HICs and staves. For each scan carried out in the tests it provides: a short description of the test, a list of the measured parameters, which parameters are written to files and the database, the cut values applied to the parameters, the output files and a collection of possible error messages with explanations. This section will also present an outline of the classification procedure for the HICs and staves.

2.1 Testing General

2.1.1 Test Execution and Test Data

When a new test is selected in the graphical user interface, a list of scans is initialised after the selection of the test type. The list contains all scans foreseen for the selected testing step. After pressing the start button, the scan list is executed one by one; in parallel to each scan the corresponding analysis is started, which analyses the scan data and produces a scan result. At the end of each scan, the tested HICs are classified for the individual scan, according to pre-defined cuts; at this stage also result and data files are written.

Once the complete scan list is finished, the test software gives a summary on the scan statuses and asks the user whether the results are to be written to database. In this case the database record is created and populated. Furthermore all local files are copied to eos.

2.1.2 HIC Classification

At the end of each scan the output parameters are used to generate the classification for the scan, details of these parameters and the cuts implemented for the scans are given in the subsequent sections. The scans can be classified as follows: gold, silver, bronze or not working (red). The cuts have been chosen so that they are in line with the cuts of the chip tests. Once all the scans are complete, the final HIC and stave classification is taken from the worst scan. This final classification cannot be better than any classification given in a previous step. However, the classification can be better than a test of the same type. This means that a retest can improve the classification.

For the HICs and staves the classifications are: gold, silver, bronze, partially working, partially working category B, no back-bias, no back-bias category B or not working. For more details about each class see table 1. A HIC is classified with back bias not working (3rd column of two cases) if either the back bias tripped during the I-V-curve in the power test or any of the scans with back bias failed (scan result Red).

2.1.3 Understanding Test Results and Classification

There are several means to understand the test results and why a given HIC has received a certain classification. This section describes these means in general, scan specific points are described in the sections of the individual scans. This section only contains the aspects of regular scan execution, possible problems in the execution of the scan itself are discussed in the next section.

Table 1: A table of the classifications used for HICs.

Worst scan result	Number of working chips OB/IB	Back bias working	HIC class
Gold	14/9	✓	Gold
Silver	14/9	✓	Silver
Bronze	14/9	✓	Bronze
Gold/Silver/Bronze	13/8	✓	Partially working
Gold/Silver/Bronze	12/7	✓	Partially working cat B
Gold/Silver/Bronze[1]	14/9	✗	No back bias
Gold/Silver/Bronze[1]	12 or 13/7 or 8	✗	No back bias cat B

[1] Only scans without back bias are considered.

Scan Classification and Cuts The scan classifications are visible in different places. Firstly, the colour of the scan in the scan progress list on the left hand side of the GUI window indicates the classification of the *worst* HIC in any given scan (e.g. if one HIC of a half-stave is classified silver for a given scan and all other HICs are classified gold, the scan will be coloured silver in the scan list).

Then, before writing to the database the user is presented with a classification window for each HIC. The window contains the classification of the HIC from previous tests, the classification of the HIC in each of the scans and the resulting final classification. When clicking on any of the scan classifications, the user gets a detailed lists of the cuts failed by this HIC in the selected test. The list contains cut name, cut value and actual value for each of the cuts. **This window should always be the first tool to understand why a given HIC has received a certain classification.** For future reference both the classification per scan and the failed cuts are attached in two files (Classification.dat and FailedCuts.txt, resp.) to the database activity. In addition the scan classifications are also attached to the database as parameters in order to facilitate their use in the preparation of yield histograms.

Result Files Each scan produces one result file per HIC, which is saved locally, attached to the database activity and copied to eos. This result file contains at the beginning a section of the scan conditions (see below) and then a summary of the scan results. The summary typically consists in average / sum values for the entire HIC, followed by the corresponding values chip-by-chip. The result files can be used e.g. as a starting point to investigate which chip degraded the HIC classification and for what reason. All result file names follow the naming convention `<ScanName>Result_<Date>_<Time>_[conditions].dat`, where `conditions` is an optional field for scans that are intentionally performed under several different conditions, e.g. varied supply voltages.

Conditions Data The first lines of the result files contain information on the conditions under which the scan was performed. These lines are particularly helpful in case of problems or where the scan does not yield the expected results. The conditions data contain:

- The software and firmware version
- The voltages and currents measured by the power board
- The set voltages on the power board
- The analogue voltage measured by the chips. The voltages are given both chip-by-chip and averaged over all chips.
- The temperature measured by the chips (chip-by-chip and HIC-average)

All voltages and temperature are given for the beginning and for the end of the scan.

Data Files In some cases the scan saves additional data files with more detailed information (threshold maps, I-V-curve, hit maps...). These data files can be used to obtain further understanding of the performance of a given HIC. The data files are not attached to the database activity, but saved locally and then copied to eos. A description of these scan-specific files is given in the sections for the individual scans.

2.2 Impedance Test

The impedance test is carried out to identify any shorts that occur between the voltage supplies of the HIC. These are digital (DVDD), analogue (AVDD) and back bias (BIAS). It is described here for completeness, but it is not executed through the same test interface as the other tests and the result does not follow the classification summarised in table 1.

2.2.1 Description

An IV curve is conducted for each voltage supply. For the DVDD and AVDD supplies, the voltage range is 0–0.2 V in steps of 0.01 V, taking 20 measurement points. While for the BIAS supply, the voltage range is 0–4 V in steps of 0.08 V, taking 50 measurement points. During all the IV curves conducted, the 3 supply channels on the power supply have a compliance limit of 100 mA for DVDD and AVDD and 10 mA for BIAS. All channels are used in linked fuse mode, i.e. if one channel reaches compliance all channels are set to 0 V and turned off. At each point the resistance is calculated using $R = V/I$. If the measured current is zero, then the resistance is set to be 1 MΩ. The final resistance is the sum of the resistances from non-zero current values, divided by the total number of measurement points. Sometimes the BIAS IV curve will measure a current of 0 A at all points, after which the user is requested to measure the resistance using a multimeter, input the result to the database by hand and close the entry.

2.2.2 Parameters and Files

There are three data files saved for the impedance test, one for each voltage supply. The list of parameters are shown in table 2.

Table 2: A table of the impedance test parameters and their cuts.

Parameter	DB parameter	In result file	Cut	Comment
AVDD impedance	yes	no	NOK if < 100 Ω	
BIAS impedance	yes	no	NOK if < 100 Ω	
DVDD impedance	yes	no	NOK if < 100 Ω	

2.3 Initial Register Readback

The very first test is an initial register readback. This occurs after the type of test has been chosen and the configuration is loaded, before the actual start of the scan program. The aim of this test is to see if the chips are powered correctly and are able to respond. If any chip does not respond, then it is masked for the remainder of the test. As a result you will see a display in the GUI with all non-working chips marked red (and all working chips green). The number of chips that have been found working in this readback is the number used for the final HIC classification (cf. table 1).

2.3.1 Errors and Troubleshooting

Red chips. Some or all of the chips do not respond to the initial test. In particular if the entire HIC or one row of 7 chips is not working there is a high probability for a setup problem rather than a problematic HIC.

- Cause: firefly and power cables are not attached properly or are faulty.
- Solution: check the cable connections or change cable.
- Cause: there is not enough power supplied to the module. Check the voltage across digital and analogue with a multimeter, is it below 1.62 V?
- Solution: has the calibration been performed? Does the power adaptor make good contact with the cross cables?
- Cause: analogue and/or digital reach compliance, possible short.
- Solution: check cross cable resistances, if any are below 100 Ω there is a short and the HIC not working. If possible, use a thermal camera to locate short and inspect under microscope to see if damage can be fixed, e.g. a broken capacitor on FPC.
- Cause: powerboard trip.
- Solution: Reset any powerboard trips e.g. over temperature.

Table 3: A table of the power test parameters and their cuts.

Parameter	DB parameter	In result file	Cut	Comment
Iddd after switch on	no	yes	> 50 mA	Cut only to exclude “dead” HICs
Idda after switch on	no	yes	> 20 mA	
Iddd with clock	no	yes	IB: 300 < Iddd < 550 mA OB: 500 < Iddd < 850 mA	
Idda with clock	no	yes	IB: 70 < Idda < 180 mA OB: 120 < Idda < 250 mA	
Iddd after config	yes	yes		
Idda after config	yes	yes		
Ibias at 0 V	yes	yes		
Ibias at 3 V	yes	yes	Silver if > 10 mA	
Max. bias voltage	(yes)	yes	no back bias if < 4 V	(in file only if tripped before 4 V)
IV curve	N/A	yes (separate file)		

2.4 Power test

The objective of the power test is to make sure that the HIC or stave operate within the functional limits for the voltage and current.

2.4.1 Description

The first step of the test is to switch off the power and clock to the HIC. Once this is done, the HIC is sequentially powered on, the clock is turned on and the HIC is configured. During these steps the voltage drop is corrected for and both the digital and analogue voltages and currents are recorded. After this, an IV curve of the back bias is performed from 0–4 V. For the IV curve a software current limit of 15 mA has been set. In case the current exceeds this limit, the HIC is classified as back bias not working and the maximum reachable voltage (i.e. the last point before the trip) is saved.

2.4.2 Parameters and Files

A table of the parameters for this scan and their respective cuts is given in table 3. A summary of the parameters, the test conditions and the classification are saved in the PowerTestResult.dat file. The IV curve is saved in a separate file, IVCurve.dat.

2.4.3 Errors and Troubleshooting

A trip during the scan. The digital, analogue and/or back bias channel reaches compliance during the scan. In back bias case Ibias > 15 mA.

- Cause: possible short.
- Solution: check cross cable resistances, if any are below 100 Ω there is a short and the HIC not working. Compare with impedance test results. If possible, use a thermal camera to locate short and inspect under microscope to see if damage can be fixed, e.g. a broken capacitor on FPC.

2.5 DCTRL Measurement

The goal of the scan is to directly measure the DCTRL driver strength and ensure that it is above an acceptable level. It also measures whether the driver strength increases linearly with the corresponding DAC setting.

2.5.1 Description

The scan is performed on all chips with a control interface. For the inner barrel this is all the chips, while for the outer barrel this is the master chips, chips 0 and 8. There are 16 measurement points from 0 to 15, representing all the DCTRL driver settings, for which a measurement of the peak-to-peak, amplitude, rise time and fall time are done. These measurements are performed separately for both positive and negative DCTRL lines. The results are then analysed with the data taken at a DCTRL setting of 0 neglected. A plot of the amplitude as a function of DCTRL setting is fit and the slope, intercept, χ^2 and correlation coefficient are extracted. The rise and fall times are defined as the time it takes to go from 20% to 80% of the peak height.

2.5.2 Parameters and Files

The parameters of the scan are shown in table 4, with the worst values for each HIC being uploaded to the DB. A summary of the parameters, the test conditions and the classification are saved in the DctrlScanResult.dat file. The raw data is saved in a separate file, DCtrlMeasurement.dat, which contains the chip number, driver setting, peak of positive, peak of negative, amplitude of positive, amplitude of negative, rise time of positive, rise time of negative, fall time of positive and fall time of negative.

2.5.3 Errors and Troubleshooting

IO Exception (13): Permission denied. Cannot run the DCTRL measurement and get an error straight away.

- Cause: computer cannot talk to the oscilloscope.
- Solution: need to add to <username> to the dialout group: ”sudo usermod -a -G dialout <username>” then log out and log back in again. Check with: ”groups <username>”..

Table 4: A table of the DCTRL measurement parameters and their cuts.

Parameter	DB parameter	In result file	Cut	Comment
worst max. amplitude	yes	yes	Not working if $< 300\,\mathrm{mV}$	
worst slope	yes	yes	Not working if $< 20\,\mathrm{mV/DAC}$	
worst χ^2	yes	yes	Not working if > 0.05	
worst correlation	yes	yes		
worst rise time	yes	yes	Silver if $> 10\,\mathrm{ns}$	
worst fall time	yes	yes	Silver if $> 10\,\mathrm{ns}$	
worst χ^2 ratio	yes	yes		Comparison with previous test
worst slope ratio	yes	yes		Comparison with previous test

The oscilloscope is not found by the PC

- Cause: USB connection
- Solution: Check the USB cable connection
- Cause: Wrong USB mode set in the oscilloscope
- Solution: Set the correct USB mode according to the setup manual

A channel has a systematic offset. For example, one channel is giving higher results than the other channels.

- Cause: termination resistor.
- Solution: check or replace termination resistor.

A channel is not working. This can happen permanently or intermittently.

- Cause: problem with the soldering of one of the components on the readout adaptor, as shown in figure 1.
- Solution: resolder component.
- Cause: bad cables.
- Solution: check connection or replace cables.

All channels working on oscilloscope, but test fails: the oscilloscope triggers, waveforms are visible and change amplitude, but the scan fails.

- Cause: Cables swapped
- Solution: Verify in the raw data file that the amplitudes is non-zero (several 100 mV, but nearly constant). This happens if the LEMO cables are not connected in the right order. Connect the cables according to the order described in the setup manual.

Trigger not received. Oscilloscope waits for trigger from MOSAIC but never receives it.

- Cause: MOSIAC not properly configured.
- Solution: Check correct NIM solder pads are soldered on MOSAIC. i.e. to use NIM 3 as an output, pad 4 needs to be soldered, see figure 3 on page 5/1-5 of the MOSIAC manual.
- Solution: MOSIAC firmware needs to be the latest version.

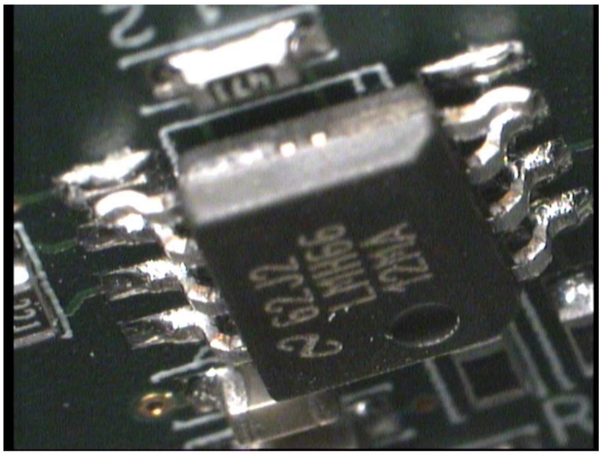


Figure 1: An example of problematic soldering on the readout adaptor.

2.6 Fifo Scan

The FIFO scan is carried out in order to test the communication over the control interface by means of a series of read-back operations. Since the read-back is performed on the region memories, also defects in these memories are detected.

2.6.1 Description

Within the ALPIDE chip, FIFO buffers are located in the periphery circuitry and store the addresses of the hit pixels, which are enqueued via the ADDR bus. The FIFO buffers are 16 bits wide, meaning they can hold multiple 16-bit numbers. For this reason, the FIFO scan inputs four hexadecimal numbers [1]; 0x0000 (0, 0000 or 0000 0000 0000), 0x5555 (21845 or 0101 0101 0101 0101), 0xaaaa (43690 or 1010 1010 1010 1010) and 0xffff (65535 or 1111 1111 1111 1111). The four hex numbers are then read out.

The chip passes the test if the readout is the same as the output, indicating a successful test. If the test is not successful there are two different types of unsuccessful readback, an exception and an error. An exception is when the MOASAIC cannot read or decode the data sent from the chip. These typically indicate a communication problem possibly due to the setup or a weak DCTRL link of one of the master chips. While an error is when the readback was completed but the value is different from the written one. This indicates a problem with a single chip. The entire FIFO scan is repeated with ±10% of the nominal supply voltage.

2.6.2 "FIFO exceptions"

One of the parameters counted during the FIFO test are the so-called "FIFO exceptions". These are exceptions registered by the MOSAIC board in the communication with the chips over the control interface. The reason for the exception can be either that the answer from the chip cannot be deciphered or that the chip ID received in the answer from the chip is not the expected one. Differently than the FIFO errors, which require

[1] Hexadecimal is a base 16 system where each hex digit (0,1,2,3,4,5,6,7,8,9,A, B, C, D, E, F) represents a 4-bit binary sequence. Each hex code starts with '0x' to signify the following four digits are in hex.

a successful communication with the chips, the FIFO exceptions therefore imply that a reliable communication through the control interface is not possible. Very often the reason for this is a connection problem - either in the Firefly or Eyespeed cable or in the power connector.

2.6.3 Parameters and Files

A summary file is created and is called FifoScanResult.dat and contains the usual information. It also details the number of exceptions and error in each pattern for the whole HIC as well as chip by chip. The three DB parameters are: FIFO errors, FIFO exceptions and chips with FIFO errors, whose name changes according to the different voltage settings. A table giving all the parameters is found in table 5.

2.6.4 Errors and Troubleshooting

Exceptions. Scan is classified as not working (red) due to exceptions found in the HIC exclusively found in chips 0-6.

- Cause: Problem in the communication through the control interface.
- Solution: Check cable connections and repeat test.

Table 5: A table of the FIFO scan parameters and their cuts.

Parameter	DB parameter	In result file	Cut	Comment
FIFO exceptions per HIC	yes	yes	Not working if > 0	
FIFO exceptions per chip	no	yes		
FIFO errors per HIC	yes	yes	Not working if > 0	
FIFO errors per HIC and pattern	no	yes		
FIFO errors per chip and pattern	no	yes		
Number of chips with FIFO errors	yes	yes	Not working if > 1	(Obsolete)

2.7 Digital Scan

The digital scan is carried out to ascertain the quality of the pixel response across the whole matrix of the chip and to test the readout path through the high-speed data link.

2.7.1 Description

The digital scan injects digital signals into the front-end electronics of each pixel and counts the number of hits recorded; testing the response of the electronics to a hit. This is performed over a certain subset of pixels chosen by the user. The default is to have 512 mask stages for the whole matrix meaning that one row of pixels is pulsed at a time. The number of injections is by default set to 50 but can be set by the user in the in the config by changing "NINJ". The scan is repeated for three settings of the digital supply voltage: nominal, +10% and -10%.

2.7.2 Readout Problems

During the digital scan, but also any other scan that uses the data readout (threshold scan, threshold tuning, noise occupancy, readout test ...), a number of readout errors are counted. These are:

- 8b10b errors: each time the MOSAIC encounters a corrupt 8b10b word, a corresponding flag is set in the event data. The number of occurrences of these flags is counted as 8b10b errors.
- Timeouts: for N trigger commands sent to the alpide chips, the software expects (N times number of chips) events back. If this number is not received within a given time, the event creates a timeout and the software moves to the next step. After a configurable number of timeouts the scan is aborted. The reason for timeouts can be various, e.g. data corruption on the high-speed link, triggers not received by the chips or single chips blocking the readout of a HIC.
- Corrupt events: if an event is received correctly, but the data cannot be decoded within the Alpide data format, this is counted as a corrupt event.

In addition to the errors listed above, which are related to the readout path, there is a fourth category, which is usually caused by a defect within a single chip:

- Priority encoder errors: this category contains cases where either a single pixel sent its address twice or the order in which the address were sent is not correct. Both problems are caused by the priority encoder in the affected double column and could in many cases be solved by masking the pixel. The addresses of the affected pixels are saved in the stuck pixel file.

2.7.3 Parameters and Files

Once the scan is complete, three files are saved, a summary file, a raw data file for each chip and a list of stuck pixels. StuckPixel.dat contains the chip ID, the pixel region, the double column and the pixel address. In the raw data file, the column, row and number

Table 6: A table of the digital scan parameters and their cuts.

Parameter	**DB parameter**	**In result file**	**Cut**	**Comment**
Timeouts per HIC	yes	yes	Not working if > 0	
8b10b errors per chip	yes	yes		
Corrupt events per HIC	yes	yes	Not working if > 0	
Priority encoder errors per HIC	yes	(yes)		= stuck pixels
Bad pixels per HIC	yes	yes		
Bad pixels per chip	yes	yes	Silver if > 50 Bronze if > 2100 Not working if > 5243 (0.1%)	
Dead pixels per HIC	no	no		
Dead pixels per chip	no	no		
Stuck pixels per HIC	(yes)	yes		= priority encoder error (address attached in file)

of hits are writen to Digital.dat. Only pixels with non-zero hit values are recorded. The summary file is named DigitalScanResult.dat and contains the test conditions, parameters for the HIC and the different types of pixels for each chip. The parameters for the test are: timeouts, 8b10b errors, corrupt events, priority encoder errors, dead pixels and stuck pixels. These are listed in table 6. Timeouts, 8b10b errors and corrupt events are linked to data transmission problems. A bad pixel is defined as a pixel that a non-zero number of hits not equal to "NINJ", 50. For pixels which record 0 hits, these are labelled as dead pixels. Pixels that have an error in the priority encoder are labelled as stuck pixels.

2.7.4 Errors and Troubleshooting

Not working classification due to one or more chips.

- Cause: one or more of the chips fails a bronze cut and is labelled not working.
- Solution: mask the problem chips(s) and retest.

Table 7: A table of the digital white frame scan parameters and their cuts.

Parameter	DB parameter	In result file	Cut	Comment
Timeouts	no	yes		
8b10b errors	no	yes		
Corrupt events	no	yes		
Unmaskable pixels per HIC	yes	yes		
Unmaskable stuck pixels per HIC[1]	yes	yes	Silver > 0 Bronze if > 2 Not working if > 5	
Unmaskable pixels per chip	no	yes		
Unmaskable stuck pixels per chip	no	yes		

2.8 Digital White Frame

The aim of the digital white frame is to identify the pixels that cannot be masked and if any of those are stuck pixels as well.

2.8.1 Description

Similar to the digital scan, a set charge is injected into the pixels a certain number of times however, for this scan, all the pixels are masked.

2.8.2 Parameters and Files

There are two parameters that this scan saves. The first one is the number of unmaskable pixels. As long as the unmaskable pixel is not noisy, then it will not pose too much of a problem. However, the second parameter, unmaskable stuck pixels, do cause a problem. If a pixel in this category is hit, then the whole double column is lost. This means that if there are several unmaskable stuck pixels in the same double column then only one double column is lost. If the unmaskable stuck pixels are spread out over several double columns, then this will result in a large number of pixels being unusable. Table 7 shows the parameters for the digital white frame scan. The files saved for the scan are DigitalWFScanResult.dat, a summary file; StuckPixels.dat, a list of stuck pixels; and UnmaskedPixels.dat, a list of unmaskable pixels.

subsubsectionErrors and Troubleshooting See errors in digital scan, section 2.7.4.

[1] Only counted if in different double columns

2.9 Threshold Scan

A threshold scan sets out to determine the threshold and noise of each pixel in units of electrons.

2.9.1 Description

Each pixel has an in-built comparator circuit which compares the charge collected in the active region to a preset threshold value. Once the charge collected surpasses this preset threshold, the pixel outputs a positive signal. The threshold scan inputs 50 analogue signals of the same magnitude into the active region of each pixel for varying charge values and records the pixel output. A plot of the pixel output as a function of the threshold is produced. In theory this should be a step function but the error on the threshold, noise, smooths the transition from a step function to a S-curve. There are two methods implemented by the GUI to extract the parameters from the S-curves, a speedy fit and root fit. The type of fit can be selected before the whole scan is started. The speedy fit method uses the differential of the S-curve which gives a Gaussian centred on the actual threshold value and the width gives a measure of the noise. For the root fit, each S-curve is fit with an error function. The threshold value where the S-curve reaches half its maximum is the threshold value and the width of the error function is the noise.

The scan is performed at two back bias voltages of 0 V and 3 V. For the HIC tests, there are two threshold scans per voltage setting, one before the tuning with nominal setting, this is not part of the HIC classification, and one after with tuned setting, this is used for classification. So the whole procedure is: threshold scan, VCASN tuning, ITHR tuning and threshold scan (tuned). More details about the tuning procedure are found in section 2.10

2.9.2 Parameters and Files

Each threshold scan returns one ThresholdScanResult.dat file and a Threshold_FitResults.dat file. The Threshold_FitResults.dat file contains the threshold and noise values for each pixel worked out using the method above. The file is of the form: <column> <row> <threshold> <noise> $< \chi^2 >$. The ThresholdScanResult.dat file contains the usual test conditions, the parameters and the classification of the threshold scan. For each chip it gives the number of pixels without hits and without threshold, hot pixels, average threshold, threshold RMS, average noise and noise RMS. Table 8 shows where these parameters are stored and what conditions dictate the scan classification. A plot showing the threshold value for each pixel is shown in figure 2.

2.9.3 Errors and Troubleshooting

All chips show large number of dead pixels.

- Cause: analogue supply voltage is too low.
- Solution: check supply voltage at cross-cables is above 1.62 V. Check contact of power adaptor with cross-cables.

Not working classification due to one or more chips.

Table 8: The threshold scan parameters and how the HIC classification relates to them.

Parameter	DB Parameter?	In result file?	Cut	Comment
Dead Pixels per HIC	yes	no		
Pixels without threshold per HIC	yes	no		
Average noise per HIC	yes	no		
Max chip noise per HIC	yes	no		
Min chip average threshold per HIC	yes	no		
Max chip average threshold per HIC	yes	no		
Pixels without hits per chip	no	yes	Silver if > 50 Bronze if > 2100 Not working if > 5243 (1%)	IB: 0.1% OB: 1 dcol
Pixels without threshold per chip	no	yes	Silver if > 5243 Bronze if > 26214 Not working if > 52429	
Hot pixels per chip	no	yes		
Average threshold per chip	no	yes		
Threshold RMS per chip	no	yes	Not working if > 30e	
Average noise per chip	no	yes	Silver if > 10e	
Noise RMS per chip	no	yes		
Threshold RMS/Mean			Bronze if > 0.3 Not working if > 0.5	
Avg. chip threshold - deviation from target	yes	yes		

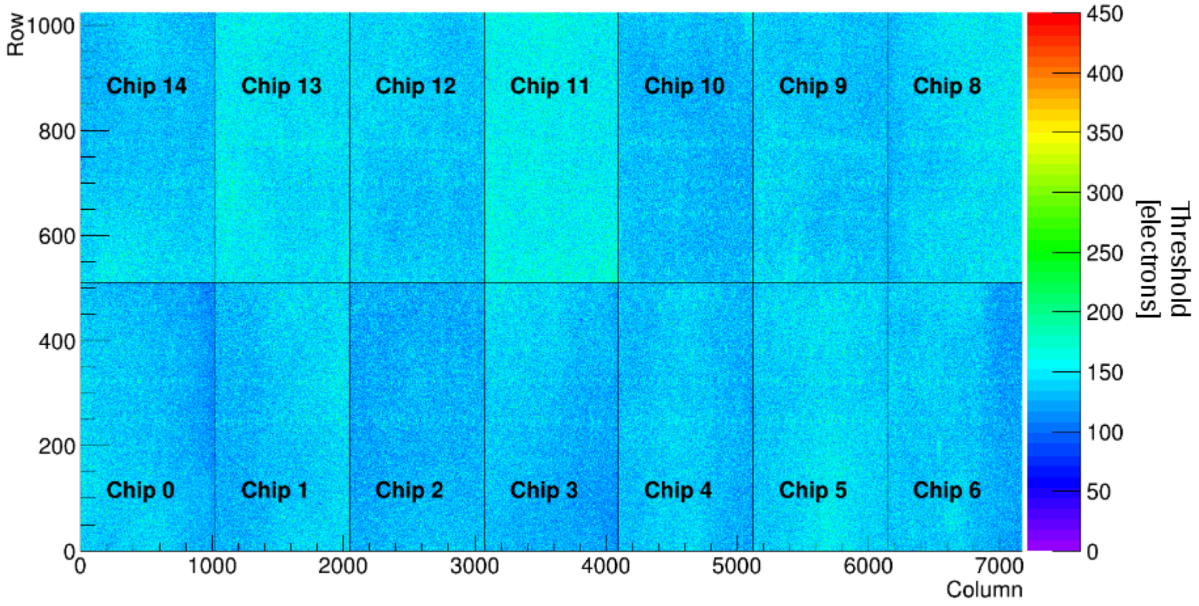


Figure 2: Threshold values for every pixel of a HIC. A uniform distribution is desirable so each pixel responds in the same way for a given event.

- Cause: one or more of the chips fails a bronze cut and is labelled not working.
- Solution: mask the problem chips(s) and retest. Also check the results of the previous threshold tunings (VCASN and ITHR): are the results reasonable?

Table 9: A table of the threshold tuning scan parameters and their cuts.

Parameter	DB parameter	In result file	Cut	Comment
Minimum chip avg VCASN/ITHR	yes	no		
Maximum chip avg VCASN/ITHR	yes	no		

2.10 Threshold Tunning

The tuning is used to slightly adjust the comparator settings of the different chips in a HIC so that a uniform response across the whole HIC is achieved. It can also help in reducing the fake hit rate.

2.10.1 Description

There are two types of tuning done: VCASN and ITHR. VCASN controls the baseline of the comparator. A higher VCASN means a higher baseline, hence a lower threshold. ITHR is a current that discharges the comparator. By decreasing ITHR, the threshold is lowered because there is a slower reaction from the feedback mechanism discharging the comparator. The scan procedure for the tuning has the same principle as the threshold scan, except this time the charge is fixed to the target threshold and either VCASN or ITHR are varied. In order to speed up the process, the scan is performed over every 64th row, i.e. 8 mask stages.

2.10.2 Parameters and Files

After the tuning is complete a summary file is written, VCASNTune.dat and ITHRTune.dat. These files contain the usual scan conditions as well as the average and RMS of VCASN or ITHR for each chip. The parameters which are saved are shown in table 9.

2.10.3 Errors and Troubleshooting

See section 2.9.3.

Table 10: A table of the noise occupancy parameters and their cuts.

Parameter	DB parameter	In result file	Cut	Comment
No. of noisy pixels per HIC	yes	yes	Silver if > 50 Bronze if > 2100 Not working if > 5243	
Noise occupancy per HIC	yes	yes		
No. of noisy pixels per chip	no	yes		
Noise occupancy per chip	no	yes		

2.11 Noise Occupancy

The goal of the noise occupancy scan is to calculate the noise occupancy and determine the noisy pixels of the chips in the HIC.

2.11.1 Description

Once the thresholds for the pixels has been set, the noise occupancy scan is performed. In the scan the full matrix is read out without injections for a total of 1×10^5 triggers. The noise occupancy of a chip is given by the number of hits divided by the number of pixels times the number of triggers. Per HIC, the noise occupancy is the total occupancy from each chip divided by the number of chips. The noise occupancy for a single pixel is calculated as the number of hits divided by the number of triggers; if the noise occupancy is above a configurable threshold, the pixel is considered noisy. The noise occupancy scan is first performed with no pixels masked and then with the noisy pixels masked both at a back bias of 0 V and 3 V.

2.11.2 Parameters and Files

Once the scan is complete a summary file is created called NoiseOccResult.dat along with a list of any noisy pixels in a separate file, NoisyPixels.dat. The list of parameters are shown in table 10.

2.12 General Issues and Troubleshooting

This section describes general issues that can occur in a variety of scans. The issues are divided into system issues, problems with the control interface and problems with the data readout.

2.12.1 System Issues

IPBus Error. The GUI terminates.

- Cause: no communication between MOSAIC and powerboard.
- Solution: check readout cable, is the powerboard on?

2.12.2 Problems with the control interface

2.12.3 Problems with the data readout

2.12.4 Database Issues

No connection to Database: All database interactions fail.

Possible causes and solutions:

- Check the validity of your kerberos token and if necessary renew it. (In the output check for messages of the type "Your kerberos credentials expired. Suggest you run kinit"). In case of doubt use klist to list all tokens with their validity.
- Check whether the database is down (this can be checked e.g. by trying to access the webinterface). In case it is down, please report to Cesar Ceballos Sanchez.

Component not found: the GUI complains that the component has not been found in the database.

In this case you can still perform the test, but will not be able to write to database. Therefore, please check why the component is not in the database and restart the test.

Component has open activities: the GUI informs you *before* the start of the test that the component has other open activities.

This means that other activities in the components history have not been closed yet. The GUI will therefore not be able to attach the component to the testing activity and hence will leave the activity open. The information before the test gives you the opportunity to solve this issue before attempting to write to the database. The test can already be started, but the issue should be resolved before selecting to write to the DB after completion of the scan list.

Problem in writing to the database: the GUI informs you that a problem has occurred during writing to the DB.

Here two cases have to be distinguished. You should check in the web interface of the database and with help of the error messages that are given in the dialogue window, which of the two cases occurred, and act accordingly.

- A problem occurred *during* creation of the activity. If you do not find any activity corresponding to the performed test in the database, you can try to rewrite. This problem might be due to a missing/expired kerberos token; therefore before re-attempting to write, please check that you have a valid token.
- A problem occurred *after* creation of the activity. In this case you will find an open activity related to your test in the database. Here you need to understand the reason for the problems (e.g. from the list of error messages in the window) and fix the activity in the web interface of the database before closing it. If you are unsure, please contact an expert. **Never manually close an activity without checking that both input and output components are set.**

A Software Installation

This section describes the procedure to prepare and perform the installation of the new-alpide-software.

A.1 Prerequisites

The software is currently supported to run only on CentOS CERN 7. This reference installations are used to test software after every commit to the repository.

- **CentOS CERN 7:**

  ```
  yum install -y gcc gcc-c++ make cmake cmake3 tar zlib wget subversion
      clang \
    git libusb1-devel tinyxml-devel qt5-qtbase-devel krb5-workstation
        cern-get-sso-cookie
  yum install -y centos-release-scl
  yum update -y
  yum install -y llvm-toolset-7 devtoolset-7
  wget -O /opt/root.tar.gz \
    https://root.cern.ch/download/root_v6.14.00.Linux-centos7-x86_64-
        gcc4.8.tar.gz
  cd /opt/ ; tar xzfv root.tar.bz
  ```

  ROOT has to be loaded using

  ```
  source /opt/root/bin/thisroot.sh .
  ```

  Do not install ROOT using `yum` as the packages are broken!

  Activate the `llvm-toolset-7` and `devtoolset-7`:

  ```
  source scl_source enable llvm-toolset-7
  source scl_source enable devtoolset-7
  ```

For other operation systems, please see https://gitlab.cern.ch/alice-its-alpide-software/alice-its-docker-containers. Please note that we cannot support other operating systems that CentOS CERN 7.

A.2 Installation

The software is available in a gitlab repository:

`https://gitlab.cern.ch/alice-its-alpide-software/new-alpide-software`

To check it out for the first time use

```
git clone https://username@gitlab.cern.ch/alice-its-alpide-software/new-alpide-
    software.git
```

with your user name. Note that your account needs to be added to a list of users in order to being able to access the repository. Versions are updated regularly, make sure you have checked out the latest version. After the clone you will find the software in a directory `new-alpide-software`, an additional subdirectory `analysis` contains standard macros to analyse the data.

Preferably, the software is compiled using a cmake-based build flow. For details see the README.md in the software repository.

A.3 Database Access

In order to establish connection with the database Kerberos needs to be configured. In CC7 the configuration can be done as below

```
sudo cp new-alpide-alpide-software/Doc/etc/krb5.conf /etc/krb5.conf
```

After the configuration the user needs to create a ticket. This is done by using the credentials as following

```
kinit <service account>
```

The user can check whether a ticket was created or not and until when it is valid, with the command

```
klist
```

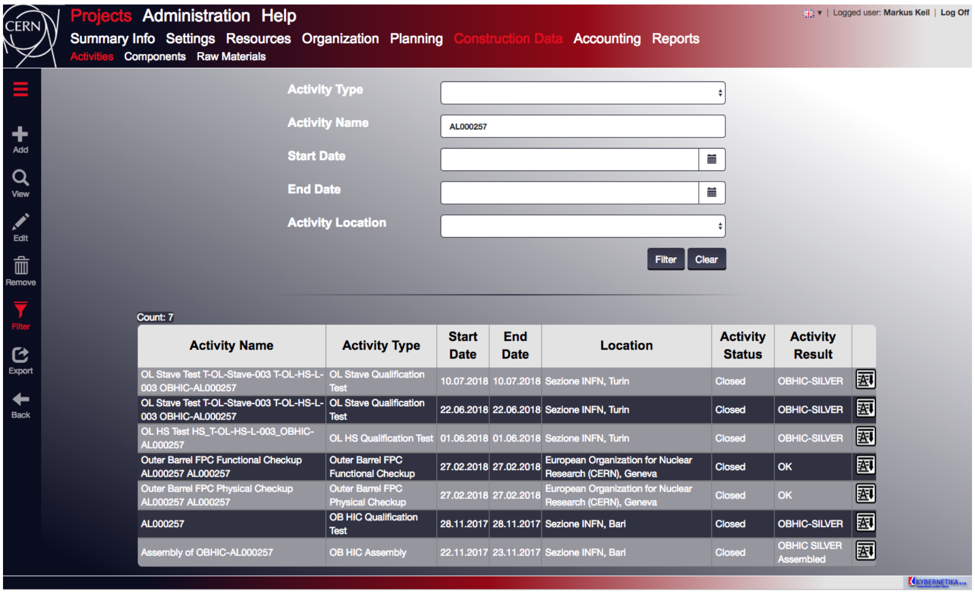
Once the ticket expires (after 3 days e.g.) the user needs to create a new one as before by using knit

B DatabaseProcedures

This section describes a few simple procedures for the database and its web interface, that can be useful when checking the integrity of the data within the database. The web interface can be reached via the address https://alucms.web.cern.ch/Projects/Project.aspx, choosing project "ITS". The procedures described here are only those, which have to be used before or after each test to check the data of a given device in the database. A full manual of the database interface is available through the `Help` menu item at the top of the database web interface.

B.1 Searching for Activities

A list of activities is obtained through the links `Construction Data`→`Activities` from the top menu of the web interface. You can use the `Filter` button on the left hand side to reduce the number of activities displayed. The names of all testing activities contain the name of the tested device. This can be used to find all (testing) activities performed on a given HIC, independently of whether the input/output components have been set in the activities. The example in Fig. 3 shows a search for all activities whose name contain AL000257. Note that the activity name field accepts substrings, but is case sensitive.


CERN
Projects Administration Help
Logged user: Markus Keil | Log Off
Summary Info Settings Resources Organization Planning Construction Data Accounting Reports
Activities Components Raw Materials
Add
View
Edit
Remove
Filter
Export
Back
Filter
Clear
KYBERNETIKA

Activity Type
Activity Name: AL000257
Start Date
End Date
Activity Location

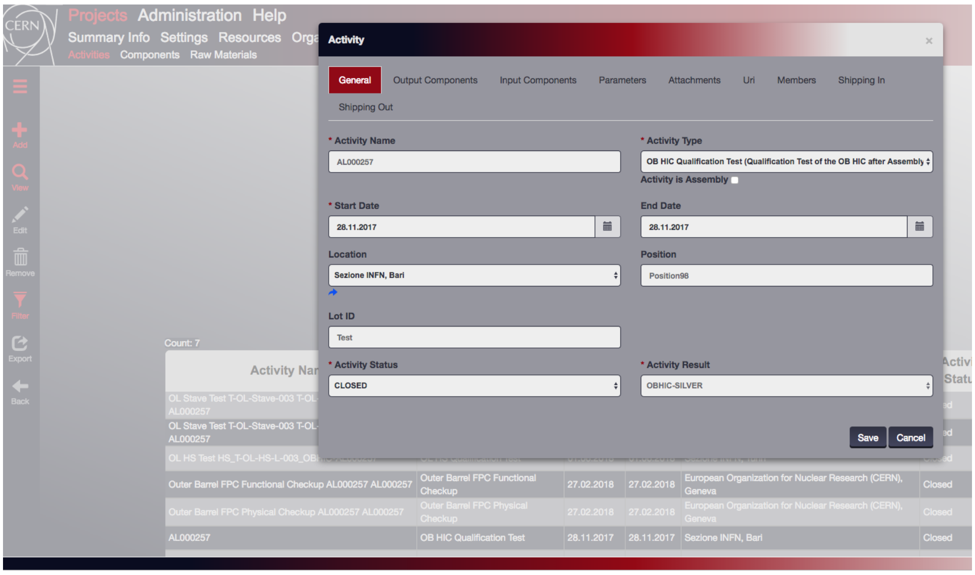
Count: 7

Activity Name	Activity Type	Start Date	End Date	Location	Activity Status	Activity Result	
OL Stave Test T-OL-Stave-003 T-OL-HS-L-003 OBHIC-AL000257	OL Stave Qualification Test	10.07.2018	10.07.2018	Sezione INFN, Turin	Closed	OBHIC-SILVER	
OL Stave Test T-OL-Stave-003 T-OL-HS-L-003 OBHIC-AL000257	OL Stave Qualification Test	22.06.2018	22.06.2018	Sezione INFN, Turin	Closed	OBHIC-SILVER	
OL HS Test HS_T-OL-HS-L-003_OBHIC-AL000257	OL HS Qualification Test	01.06.2018	01.06.2018	Sezione INFN, Turin	Closed	OBHIC-SILVER	
Outer Barrel FPC Functional Checkup AL000257 AL000257	Outer Barrel FPC Functional Checkup	27.02.2018	27.02.2018	European Organization for Nuclear Research (CERN), Geneva	Closed	OK	
Outer Barrel FPC Physical Checkup AL000257 AL000257	Outer Barrel FPC Physical Checkup	27.02.2018	27.02.2018	European Organization for Nuclear Research (CERN), Geneva	Closed	OK	
AL000257	OB HIC Qualification Test	28.11.2017	28.11.2017	Sezione INFN, Bari	Closed	OBHIC-SILVER	
Assembly of OBHIC-AL000257	OB HIC Assembly	22.11.2017	23.11.2017	Sezione INFN, Bari	Closed	OBHIC SILVER Assembled	

Figure 3: Searching for activities in the web interface of the production DB

The details of all activities can be expanded by clicking on the "Detail" button on the right of the activity or, in a more detailed view, through the "View" button in the menu on the left hand side and selecting the check box of the activity. The latter opens a detail window as shown in Fig. 4.

In a similar way activities can be modified by using the "Edit" button from the left-hand menu, however **before closing any activity manually the correctness of the data imperatively needs to be checked. This concerns in particular the setting of the input AND output components of the activity and the activity result.**



* Activity Name: AL000257

* Activity Type: OB HIC Qualification Test (Qualification Test of the OB HIC after Assembly

Activity is Assembly □

* Start Date: 28.11.2017

End Date: 28.11.2017

Location: Sezione INFN, Bari

Position: Position98

Lot ID: Test

* Activity Status: CLOSED

* Activity Result: OBHIC-SILVER

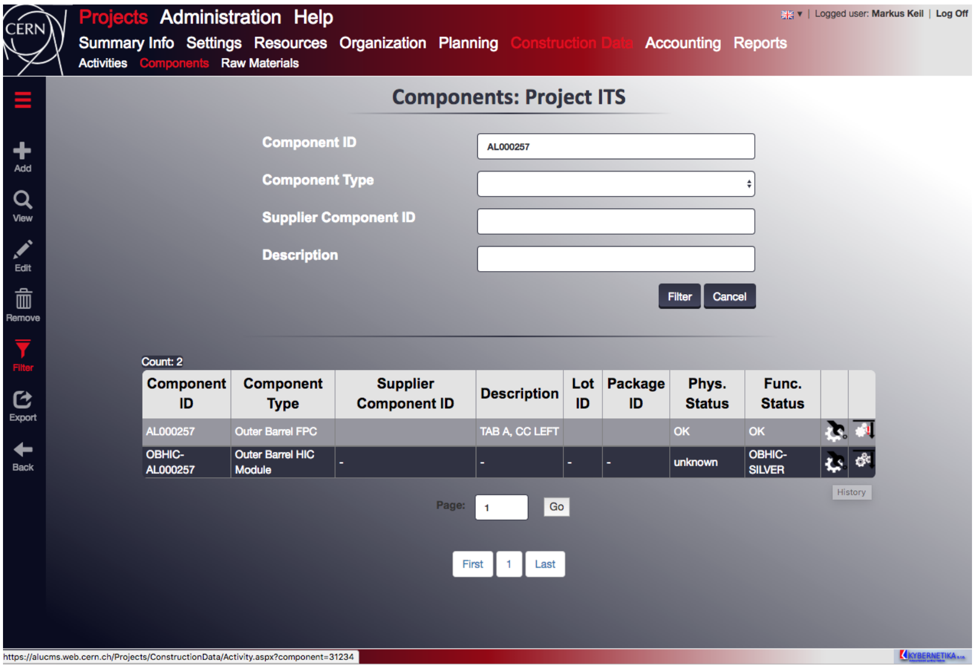
Count: 7

Outer Barrel FPC Functional Checkup AL000257 AL000257	Outer Barrel FPC Functional Checkup	27.02.2018	27.02.2018	European Organization for Nuclear Research (CERN), Geneva	Closed
Outer Barrel FPC Physical Checkup AL000257 AL000257	Outer Barrel FPC Physical Checkup	27.02.2018	27.02.2018	European Organization for Nuclear Research (CERN), Geneva	Closed
AL000257	OB HIC Qualification Test	28.11.2017	28.11.2017	Sezione INFN, Bari	Closed

Figure 4: Activity details for an OB HIC qualification test.

B.2 Searching for Components

Similar to the activities the web interface also allows to search for components entered into the database. This is done by selecting the menu items `Construction Data`→`Components`. Also here a filter can be applied on the list of components. This is shown in Fig. 5 for the same name from the activity search example.


CERN
Projects Administration Help
Logged user: Markus Keil | Log Off
Summary Info Settings Resources Organization Planning Construction Data Accounting Reports
Activities Components Raw Materials
Add
View
Edit
Remove
Filter
Export
Back
Filter
Cancel
History
Page: 1 Go
First 1 Last
https://alucms.web.cern.ch/Projects/ConstructionData/Activity.aspx?component=31234
KYBERNETIKA

Components: Project ITS

Component ID: AL000257

Component Type:

Supplier Component ID:

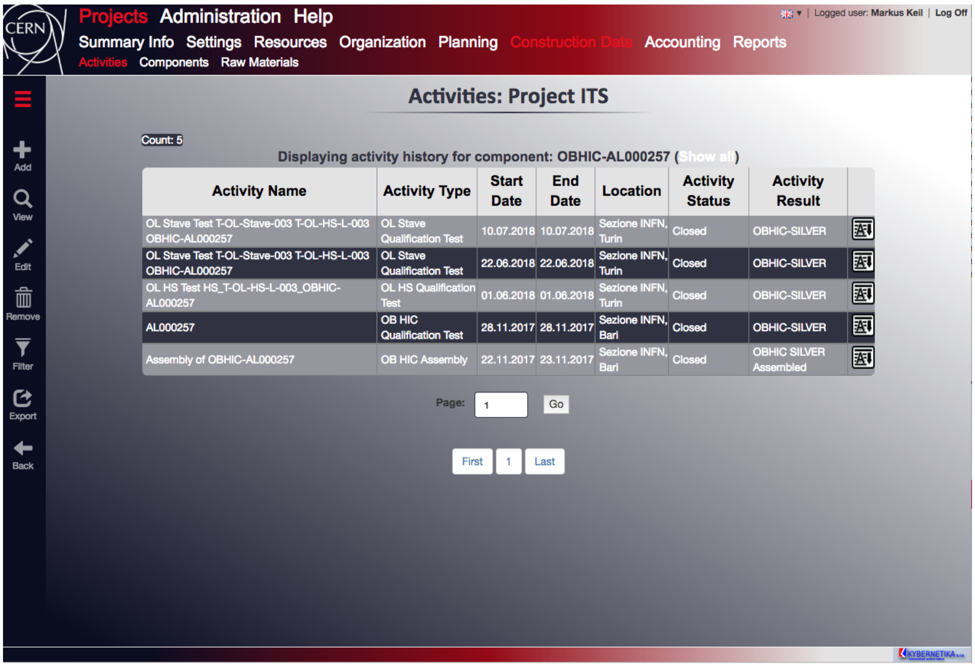
Description:

Count: 2

Component ID	Component Type	Supplier Component ID	Description	Lot ID	Package ID	Phys. Status	Func. Status		
AL000257	Outer Barrel FPC		TAB A, CC LEFT			OK	OK		
OBHIC-AL000257	Outer Barrel HIC Module	-	-	-	-	unknown	OBHIC-SILVER		

Figure 5: Searching for components in the web interface of the production DB

An important point in the component list as shown in Fig. 5 is the history button on the right hand side (second button from the right, gear wheel with wrench). Clicking this button leads to the activity history of the component, i.e. a list of all activities that are linked to this component. Figure 6 shows this history for the example component OBHIC-AL000257.



Projects Administration Help
Logged user: Markus Keil | Log Off
Summary Info Settings Resources Organization Planning Construction Data Accounting Reports
Activities Components Raw Materials
Add
View
Edit
Remove
Filter
Export
Back

Activities: Project ITS

Count: 5

Displaying activity history for component: OBHIC-AL000257 (Show all)

Activity Name	Activity Type	Start Date	End Date	Location	Activity Status	Activity Result	
OL Stave Test T-OL-Stave-003 T-OL-HS-L-003 OBHIC-AL000257	OL Stave Qualification Test	10.07.2018	10.07.2018	Sezione INFN, Turin	Closed	OBHIC-SILVER	
OL Stave Test T-OL-Stave-003 T-OL-HS-L-003 OBHIC-AL000257	OL Stave Qualification Test	22.06.2018	22.06.2018	Sezione INFN, Turin	Closed	OBHIC-SILVER	
OL HS Test HS_T-OL-HS-L-003_OBHIC-AL000257	OL HS Qualification Test	01.06.2018	01.06.2018	Sezione INFN, Turin	Closed	OBHIC-SILVER	
AL000257	OB HIC Qualification Test	28.11.2017	28.11.2017	Sezione INFN, Bari	Closed	OBHIC-SILVER	
Assembly of OBHIC-AL000257	OB HIC Assembly	22.11.2017	23.11.2017	Sezione INFN, Bari	Closed	OBHIC SILVER Assembled	

Figure 6: Display of the history of component OBHIC-AL000257

It is good practice to compare the activities shown in the component's history with those found via the procedure described in B.1, in order to find activities who have not been closed and/or whose component have not been set correctly. In the example shown above all activities have been correctly closed since all the HIC activities of the activity list (Fig. 3) also appear in the component's history (Note that the two additional activities in Fig. 3 refer to the FPC with the same name).

C Further Literature