

RICH700d in Oracle and Hydra

[RICH700d in Oracle and Hydra](#)

[Develoments until July 2017](#)

[The current status:](#)

[To Do next:](#)

[Resources until July 2017 :](#)

[Hydra2:](#)

[Oracle:](#)

[Develoments in October/November 2017](#)

[Geometry index definitions](#)

[Rich700 unpacker](#)

[Database interface and Oracle tables](#)

[HRich700Trb3Lookup](#)

[HRich700ThresholdPar](#)

[The current status](#)

[To do](#)

[Develoments in December 2017](#)

[Tests with COSY beamtime data in HYDRA online monitoring framework](#)

[Preparation of the mapping](#)

[Adjustements in the online monitoring](#)

[Bugfixes and improvements in the preliminary unpacker](#)

[First results](#)

[To do](#)

[Resources since November 2017](#)

[Hydra2](#)

[Hades online monitoring application for cosy data test](#)

[Oracle](#)

Develoments until July 2017

Currently the hydra2 unpacker will just map (trbnetaddress, channel) into some global (pixel_column, pixel_row) coordinates. This means one can get the "absolute" x-y location of each pixel on the sensitive plane. Note that we do not have 6 separate hades sectors in this stage, they can be calculated from (pixel_column, pixel_row) in the next steps to match the (exisiting)subsequent analysis.

Concerning Oracle, internally this is organized with a table RICH700_TRB3LOOKUP_DATA that maps (trbnetaddress, channel) to a global photomultiplier id pmt_id and the local pixel index pmt_pixel on each pmt. The (pixel_column, pixel_row) coordinates are *currently* taken from the table RICH700_PIXEL_GRID via a global pixel_id index that is related to (pmt_id, pmt_pixel) by table RICH700_PIXEL.

Later one could handle such evaluation of (pixel_column, pixel_row) via the pmt_id itself, since the PMT location should be defined in the corresponding geometry classes of HGEOM account.

Additionally, one could relate a dedicated carrier backplane via id to each PMT, which might be useful for backplane related slow control data later. For such intentions, the table RICH700_PMT has been defined as a suggestions for discussion. Note that this is empty and not used!

The current status:

1. The relevant tables in Oracle were created and filled on the test database. As initial mapping the set up of June2017 "photon beam time" has been imported via sql here.
2. The hydra2 framework compiles with the described changes for rich700d against Oracle db-hades-test
3. As user rich_ana2@db-hades-test, it is possible to export the parameter container Rich700Trb3Lookup to root format with macro init_rich700.C, and into ascii format with macro init_rich700_ascii.C (see sources below).
4. With macro writeParToOra.C (see below), it is possible to read Rich700Trb3Lookup from ascii file and insert it to Oracle as user rich_ana2@ db-hades-test. A new treestyle parameter version is generated and can be validated as user rich_ana2 by the web interface at https://hades-db.gsi.de/pls/hades_webdbs-test/hanal2.htreepar_mtn.entry_form_versions
5. The newly validated parameter can be exported from Oracle to ascii again with init_rich700_ascii.C and this shows up identical with the previously imported one. Therefore the parameter container io seems to work!

To Do next:

1. Test the very trb3 unpacking with such parameter container by means of HRich700Trb3Unpacker class.
2. Discuss the future organization of pmt ids, pixel ids, row, columns etc. with respect to the geometry and the simulation classes, and the further rich analysis.
3. Develop another (treestyle?) parameter container to cover the threshold and calibration parameters required for rich700 pmt and trb3. The validation procedures should also be discussed (automatically on daq start as before? Manually on web interface after each change?).
4. Plan if and how the carrier backplane related information from the slow control import has to be used for the rich700 analysis?

Resources until July 2017 :

Hydra2:

The relevant changes in hydra2 are available in such test repository: <https://subversion.gsi.de/dabc/rich700/hydra2/>

This concerns additional classes in hydra2/rich:

- hrich700trb3lookup
- hrich700trb3unpacker

and changes in hydra2/rich:

- richdef.h (new category catRich700Raw).
- hrichparrootfileio
- hrichparasciifileio

and modifications in

- hydra2/ora/hrichparora2io
- hydra2/base/datasource/htrbnetdef.h (defined preliminary rich700 address range that would match for the test beamtime data)

My test macros for reading and writing parameter containers are here: <https://subversion.gsi.de/dabc/rich700/test/>

Oracle:

The scripts for creation of the Oracle objects (tables etc.) are kept here: https://subversion.gsi.de/dabc/rich700/oracle/rich_ana2/

Note that the initial photon beamtime mapping as excel sheet at https://subversion.gsi.de/dabc/rich700/oracle/rich_ana2/scripts/geometry_cp.xlsx.) This has been implemented as user rich_ana2 on the gsi Oracle test data base db-hades-test. The actual tables etc. can be browsed as usual with the webdocumentation at

https://hades-db.gsi.de/pls/hades_webdbs-test/hades_util.hxwww_doc.show_account?p_id=17

The treestyle parameter container Rich700Trb3Lookup can also be inspected in the webdb gui, e.g. https://hades-db.gsi.de/pls/hades_webdbs-test/hanal2.htreepar_query.entry_form_sets

Please also remember the discussion minutes at <http://jspc29.x-matter.uni-frankfurt.de:9001/p/RICH700-Oracle>

-- JoernAdamczewski - 05 Jul 2017

Develoments in October/November 2017

Geometry index definitions

All PMTs are identified by a global **pmt_id** [1...576]. This links the PMT to a corresponding geometry object by name. PMT number 1 is the location at the *lower right corner* of the rich700 sensitive plane when looking in beam downstream direction ($x=-609.5$ mm, $y=-609.5$ mm) . This grid of virtual PMT positions has 24x24 locations. Note that the first *real* PMT mounted has the id number 9 (about $x=-185.5$ mm , $y=-609.5$ mm). The index numbering is then with incrementing in x direction (moving left when looking beam downstream). At **pmt_id** =25 the next y row above continues.

The pixels on each PMT are numbered by a local **pixel_id** [1...64]. Pixel id 1 begins at the lower right corner of each PMT (looking beam downstream), i.e. at minimum x and minimum y position. The pixel numbering follows increasing x coordinates, at pixel_id=9 the next y row begins above the previous one. From **pmt_id** and **pixel_id** a global raster position of each pixel may be calculated.

Rich700 unpacker

The hydra2 unpacker is implemented as class **HRich700Trb3Unpacker**. This is a subclass of previously existing **HTrb3Unpacker**. The first stage of unpacking is done with previously existing class **HTrb3TdcUnpacker**. This class converts the leading and trailing edge hits from the hld file into a list of "records" of class **HTrb3TdcUnpacker::ChannelRec** for each tdc channel. The **HRich700Trb3Unpacker** just scans this list of hits and evaluates them further to fill the new output event category "catRich700Raw" with data objects **HRich700Raw** for every single pixel that was hit.

The mapping between (**trbnetaddress**, **channel**) into (**pmt_id** , **pixel_id**) is done by means of a tree-style parameter container class **HRich700Trb3Lookup**. This keeps for each dirich TDC an object of class **HRich700Trb3LookupTdc** which contains for each channel a **HRich700Trb3LookupChan** structure with the mapping to (pmt,pixel).

Additionally, for each pixel (**pmt_id**, **pixel_id**) time threshold parameters are kept in a tree-style parameter container **HRich700ThresholdPar**. The trb3 hit threshold parameters (**t_min**, **t_max**, **tot_min** , **tot_max** , **flag**) for each pixel are kept in class **HRich700PixelThreshold**. A nonzero value of "flag" will suppress the pixel completely.

Database interface and Oracle tables

Interface to read/write the parameter container to ascii and root file have been implemented in hydra2. An example asciii file can be found at <https://subversion.gsi.de/dabc/rich700/test/Rich700ParamsNov2017.txt>

The Oracle tables are tested as account RICH_ANA2 on the hades test data base (https://hades-db.gsi.de/pls/hades_webdbs-test/hades_util.hxsite.main)

HRich700Trb3Lookup

Oracle tables RICH700_TRB3LOOKUP_DATA, RICH700_TRB3LOOKUP, RICH700_TRB3LOOKUP_VERS implement the tree-style parameter container pattern. Corresponding oracle views RICH700_TRB3LOOKUP_DATA_VIEW and RICH700_LOOKUP_VERS_AT_DATE provide the reading interface to hydra2. The view HT_RICH700_LOOKUP_DATA allows a display of parameter set in the web interface for validation.

The ASCII file interface has the column format:

```
trbnet-address channel pmt pixel
```

HRich700ThresholdPar

Oracle tables RICH700_THRESHOLD_DATA, RICH700_THRESHOLD, RICH700_THRESHOLD_VERS implement the tree-style parameter container pattern. Corresponding oracle views RICH700_THRESHOLD_DATA_VIEW and RICH700_THRESHOLD_VERS_AT_DATE provide the reading interface to hydra2. The views HT_RICH700_THRESHOLD_PARTS and HT_RICH700_THRESHOLD_DATA_VIEW allow a display of parameter set in the web interface for

validation. Here each PMT can be shown separately.

The ASCII file interface has the column format:

```
pmt pixel T_min (ns) T_max (ns) Tot_min (ns) Tot_max (ns) Flag
```

The current status

1. The relevant tables in Oracle were created and filled on the test database. As initial mapping the set up of June2017 "photon beam time" has been imported via sql here.
2. The hydra2 framework compiles with the described changes for rich700d against Oracle db-hades-test
3. As user rich_ana2@db-hades-test, it is possible to export the parameter containers Rich700Trb3Lookup and Rich700ThresholdPar to root format with macro `init_rich700.C`, and into ascii format with macro `init_rich700_ascii.C`.
4. With macro `writeParToOra.C` it is possible to read Rich700Trb3Lookup from ascii file and insert it to Oracle as user rich_ana2@ db-hades-test. A new treestyle parameter version is generated and can be validated as user rich_ana2 by the web interface at https://hades-db.gsi.de/pls/hades_webdbs-test/hanal2.htreepar_mtn.entry_form_versions
5. With macro `writeThresholdParToOra.C` it is possible to read Rich700ThresholdPar from ascii file and insert it to Oracle as user rich_ana2@ db-hades-test. A new treestyle parameter version is generated and can be validated as user rich_ana2 by the web interface at https://hades-db.gsi.de/pls/hades_webdbs-test/hanal2.htreepar_mtn.entry_form_versions
6. The newly validated parameter can be exported from Oracle to ascii again with `init_rich700_ascii.C` and this shows up identical with the previously imported one. Therefore the parameter container io seems to work!

To do

1. Test the very trb3 unpacking with such parameter container by means of HRich700Trb3Unpacker class. Here a real hld file from test beam data may be used. A simple monitor histogram in hydra may display the pixel hit map with row and column positions on the PMT plane.
2. Improve the HRich700Trb3Unpacker concerning the RICH group requirements.
3. Implement a geometry container for rich700 in hydra2 and Oracle.
4. Define the trbnet address range reserved for rich700. This has to be put into hydra2 (`hydra2/base/datasource/htrbnetdef.h`) and oracle (range check of tables).

Develoments in December 2017

Tests with COSY beamtime data in HYDRA online monitoring framework

The unpacker HRich700Trb3Unpacker has been run in the HADES online monitoring (QA) framework.

Preparation of the mapping

The test beam mapping from COSY (geometry.dat file by Christian Pauly) has been converted such that the PMT ids used in the beamtime (1...12) are mapped to PMT ids of the future HADES RICH plane with corresponding relative orientation. For this test we chose the right lower corner of the HADES PMTs (i.e. minimum x and minimum y coordinates in beam reference system), beginning from PMT 9 up to PMT 107 on the other edge.

This conversion was done by newly developed helper script `convertgeometry.pl`. The output of this script are two HADES database ascii files for parameter containers `Rich700Trb3Lookup` and `Rich700ThresholdPar`. These parameter containers could be stored to Oracle test data base by ROOT macros `writeParTo0ra.C` and `writeThresholdParTo0ra.C` as described above. They could be validated using the Oracle web interface methods.

Adjustements in the online monitoring

The standard beamtime monitoring from `hydra2/online` has been reduced to only show the RICH detector:

- The online analysis server `hadesonlineserver.cc` is configured to read just from the sample file of COSY2017 beamtime `rich_thrscan_no_WLS_thr_70_17308211044.hld`. It will use the ASCII files as parameter input for the `Rich700Trb3Lookup` and `Rich700ThresholdPar` containers.
- The monitoring server component `createHades.C` drops all other detector components except for the RICH700. It defines the `HRich700Trb3Unpacker` for the subevent (hub) id mask used at COSY (0x8100). It just uses one taskset `HRich700TaskSet` that has been modified in `hydra2` framework to also accept "real" data.
- The monitoring server component `Rich.C` drops all standard histograms for the moment. Instead it defines new histograms `hRich700PMT` and `hRich700Pixel` and fills them from the data of the new event category `catRich700Raw` that keeps the trb3 hits for each pixel
- The monitoring client configuration has been defined in `ClientConfigRich700_PMTonly.xml` such that it will show only the two new histograms to check the unpacking and mapping

Bugfixes and improvements in the preliminary unpacker

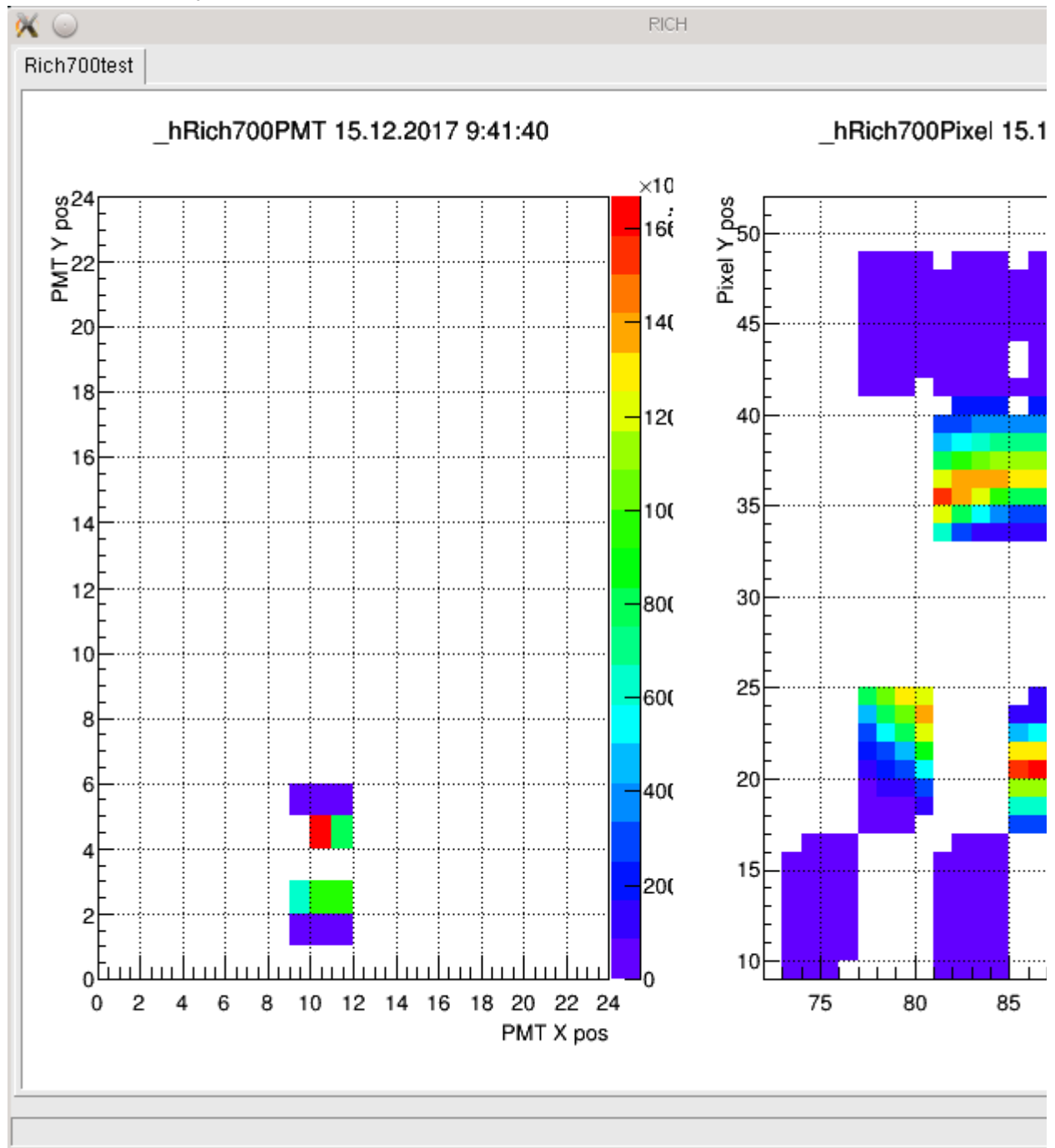
During the tests it was necessary to modify the first draft of the `HRich700Trb3Unpacker`:

- The setup of the single `HTrb3TdcUnpacker` components (for each TDC address) has to be done from the `Rich700Trb3Lookup` parameter container in `reinit()` function, not in the `init()`. Otherwise the parameter container is not yet initialized from the database (the input ascii files resp.).
- The time selection of hits by the default `Rich700ThresholdPar` was deactivated by a compile time define `RICH700_NO_TIME_CUT`. The cosy data contains negative hit times with respect to the trigger time at channel 0, so all hits would be suppressed with this setup (**TODO: define a reasonable initial time cut parameter container and validate it in the database**)
- The slot range of the category `catRich700Raw` (for class `HRich700Raw`) has to be extended by one to also cover indices 524 (for PMT id) and 64 (for pixel id). Database mapping starts

with index 1 for both, so we do the same in the matrix category and just ignore the index 0 slot.

First results

It is working! See the snapshot of the hades online monitoring window processing the COSY beamtime data in hydra:



Left histogram shows the mocked up PMT positions in the future RICH700 backplane array, right histogram the pixel positions on such plane (zoomed to the interesting region with our 12 testbeam PMTs)

To do

- In addition to the event structure `catRich700Raw`, it is required to also fill the structure `catRichCal` already in the `HRich700Trb3Unpacker`. This requires a geometrical transformation of the pixel positions into the sector, row, column scheme used for the old Rich.
- Once the `catRichCal` is filled by the `HRich700Trb3Unpacker`, the online monitoring can activate again the other rich tasksets and histograms. This allows to check if the mapping from `catRich700Raw` to `catRichCal` is working.

Note that the code resources on the webserver mentioned below have been updated to this most recent versions

-- JoernAdamczewski - 15 Dec 2017

Resources since November 2017

Hydra2

A git snapshot of my current developer version of hydra2 with all changes is available at <http://web-docs.gsi.de/~adamczew/git/hydra2dev.tar.gz>

After unpacking, the file `mydefault_hydra.sh` has to be adjusted:

- The `HADDIR` location has to be changed to the private hydra2 installation directory. With these changes it should compile on gsi linux cluster.
- For **compilation without Oracle**, the `export USES_ORACLE=yes` has to be deactivated to `no`
- For compilation outside gsi linux cluster, the environment `R00TSYS` and `CERN_ROOT` must be set to the installation locations of root and cernlib
- For compilation outside gsi linux cluster, the helper function environment `hsc-functions.sh` has to be copied from gsi

The test macros for reading and writing parameter containers are in a separate repository:

<https://subversion.gsi.de/dabc/rich700/test/>

Please note that you need the RICH_ANA2 account login for the hades Oracle test database to invoke the macros!

Hades online monitoring application for cosy data test

A tarball containing the modified hades online monitoring setup for testing the unpacker and mapping with the COSY testbeam data can be found here: <http://web-docs.gsi.de/~adamczew/git/hadesonline.tar.gz>

The geometry files and conversion script `convertgeometry.pl`, the initial setup and the generated ascii parameter container files can be found at <http://web-docs.gsi.de/~adamczew/git/cosydata.tar.gz>

-- JoernAdamczewski - 15 Dec 2017

Oracle

This has been implemented as user rich_ana2 on the gsi Oracle test data base db-hades-test. The actual tables etc. can be browsed as usual with the webdocumentation at

https://hades-db.gsi.de/pls/hades_webdbs-test/hades_util.hxwww_doc.show_account?p_id=17

The treestyle parameter containers Rich700Trb3Lookup and Rich700ThresholdPar can also be inspected in the webdb gui, e.g. https://hades-db.gsi.de/pls/hades_webdbs-test/hanal2.htreepar_query.entry_form_sets

A git snapshot of the relevant Oracle setup scripts for account rich_ana2 is available at <http://web-docs.gsi.de/~adamczew/git/oradev.tar.gz>

-- JoernAdamczewski - 02 Nov 2017

This topic: Computing > WebHome > OracleRICH700

Topic revision: 15 Dec 2017, JoernAdamczewski

Copyright © by the contributing authors. All material on this collaboration platform is the property of the contributing authors.
Ideas, requests, problems regarding Hades Wiki? Send feedback

