

# Introduction CDB and RDB

---

- CDB global tag **v6.3=2025DataV0** as of **now**
  - ▷ global tag for **entire** 2025 data taking period
  - ▷ alignment information for all detectors (to be updated, see below)
  - ▷ detsetup (B field - not used outside of minalyzer/vertexing)
  - ▷ pixelquality (cf Haris' talk)
- CDB **and** RDB available from
  - ▷ REST: pc11740.psi.ch
  - ▷ JSON: merlin7:/data/project/mu3e/cdb
  - ▷ JSON: mu3ebe:/data/mu3e/cdb
  - ▷ sync'ed from pc11740, mu3ebe rsync'ed from merlin7
- CDB next steps
  - ▷ include tile channel quality information (**eventually also fibres?**)
  - ▷ add to CDB: pixel time calibration and cosmic-track based alignment
  - ▷ more, continued, validation
- **Missing**
  - ▷ **fibres good (or bad) run list** and **channel quality list**

# mu3eUtil/cdb updates (dev)

- **calPixelQualityLM** with

- ▶ bool isChipDead(unsigned int chipid, int row = -1, int col = -1);
- ▶ bool isLinkBad(unsigned int chipid, int ilink);
- ▶ bool isLinkDead(unsigned int chipid, int ilink);
- ▶ enum Status, see below
- more if desired

- **calPixelTimeCalibration**

- ▶ CDB interface to Nik's pixel time calibration
- ▶ initial payload based on single **largecalib.calib**
- ▶ **yield identical results for**
  - file initialization
  - CDB initialization
- ▶ **GT/tag/payload not yet in CDB!**  
(this afternoon)
- ▶ unclear to me how to propagate this code  
(mu3eAna/anaComba/. . . )

```
calibration ReadCalibrationCDB(){
    calibration calib;
    cdbAbs *pDB(0);
    std::string gt = "datav6.3=2025V0";
    std::string db = "/Users/ursl/data/mu3e/cdb";
    int verbose(1000);
    pDB = new cdbJSON(gt, db, verbose);
    Mu3eConditions *pDC = Mu3eConditions::instance(gt, pDB);
    pDC->setRunNumber(3812);
    calAbs *cal = pDC->getCalibration("pixeltimecalibration_");
    calPixelTimeCalibration* cpt = dynamic_cast<calPixelTimeCalibration*>(cal);

    for(uint chip =0; chip < NCALIBRATIONCHIPS; chip++){
        chipcalibration cc;
        int globalChipID = IDfromChipNr(chip);
        for(uint sector = 0; sector < NSECTORS; sector++){
            for(uint tot = 0; tot < NTOTBINS; tot++){
                cc[sector][tot].mean = cpt->getMean(globalChipID, sector, tot);
                cc[sector][tot].meanerr = cpt->getMeanErr(globalChipID, sector, tot);
                cc[sector][tot].sigma = cpt->getSigma(globalChipID, sector, tot);
                cc[sector][tot].sigmaerr = cpt->getSigmaErr(globalChipID, sector, tot);
            }
        }
        calib[IDfromChipNr(chip)] = cc;
    }
    return calib;
}
```

# Runs: good, bad, undecided, . . .

---

- "good" runs in 2025 datataking period
    - ▷ Jak's pixel list  
(though no difference between "good" and "bad" runs wrt track  $p$  distribution)
    - ▷ Elizaveta's tiles list
    - ▷ **no list for fibres**
  - "bad" runs
    - ▷ all links affected by LVDS errors
    - ▷ way too much noise  
(so far based on vertex criteria only)
  - Example bad run criteria for VTX
    - ▷ number of good VTX pixels  $< 10^6$
    - ▷ number of VTX links with disqualifying LVDS errors  $> 40$
- ⇒ Included in RDB
- ▷ REST on pc11740
  - ▷ JSON on merlin7/mu3ebe, mu3eUtil accessible (or your own JSON parsing)

# Runs: Good, Bad, Not Bad, Any, Unset

Mu3e Run Database

pc11740.psi.ch/rdb/?onlySignificant=yes&runClass=Beam&dqVtx=1&dqTil=1

Relaunch to update

## Mu3e Run Database

Significant Runs only ☒

Run class Beam

Run range min max

Max runs limit

Search comments search in comments

Apply Filters

Data Quality Filters

Mu3e Any

Beam Any

Vertex Good

Pixel Any

Fibres Any

Tiles Good

Link Any

Showing the last 27 runs (significant runs only)

	Run	Start time	End time	Sig.	Class	Events	Shift crew	Shift Comments/RunInfo Comments	Resources
<input type="checkbox"/>	<a href="#">4898</a>	Fri Jun 20 13:13:46 2025	Fri Jun 20 13:15:38 2025	✓	Beam	174646	LM, ES, MS, NZ	Beam with current best values (TH 126, single TDAC config) - Long run over lunch	<a href="#">ov</a>
<input type="checkbox"/>	<a href="#">4896</a>	Fri Jun 20 13:09:32 2025	Fri Jun 20 13:11:24 2025	✓	Beam	170577	LM, ES, MS, NZ	Beam with current best values (TH 126, single TDAC config) - Long run over lunch	<a href="#">ov</a>
<input type="checkbox"/>	<a href="#">4894</a>	Fri Jun 20 13:05:17 2025	Fri Jun 20 13:07:09 2025	✓	Beam	181751	LM, ES, MS, NZ	Beam with current best values (TH 126, single TDAC config) - Long run over lunch	<a href="#">ov</a>
<input type="checkbox"/>	<a href="#">4893</a>	Fri Jun 20 13:03:10 2025	Fri Jun 20 13:05:03 2025	✓	Beam	171049	LM, ES, MS, NZ	Beam with current best values (TH 126, single TDAC config) - Long run over lunch	<a href="#">ov</a>
<input type="checkbox"/>	<a href="#">4892</a>	Fri Jun 20 13:01:03 2025	Fri Jun 20 13:02:56 2025	✓	Beam	189603	LM, ES, MS, NZ	Beam with current best values (TH 126, single TDAC config) - Long run over lunch	<a href="#">ov</a>
<input type="checkbox"/>	<a href="#">4889</a>	Fri Jun 20 12:54:42 2025	Fri Jun 20 12:56:34 2025	✓	Beam	184239	LM, ES, MS, NZ	Beam with current best values (TH 126, single TDAC config) - Long run over lunch	<a href="#">ov</a>
<input type="checkbox"/>	<a href="#">4888</a>	Fri Jun 20 12:52:35 2025	Fri Jun 20 12:54:27 2025	✓	Beam	176593	LM, ES, MS, NZ	Beam with current best values (TH 126, single TDAC config) - Long run over lunch	<a href="#">ov</a>
<input type="checkbox"/>	<a href="#">4887</a>	Fri Jun 20 12:50:28 2025	Fri Jun 20 12:52:20 2025	✓	Beam	185499	LM, ES, MS, NZ	Beam with current best values (TH 126, single TDAC config) - Long run over lunch	<a href="#">ov</a>
<input type="checkbox"/>	<a href="#">4886</a>	Fri Jun 20 12:48:21 2025	Fri Jun 20 12:50:13 2025	✓	Beam	166220	LM, ES, MS, NZ	Beam with current best values (TH 126, single TDAC config) - Long run over lunch	<a href="#">ov</a>
<input type="checkbox"/>	<a href="#">4885</a>	Fri Jun 20 12:46:14 2025	Fri Jun 20 12:48:06 2025	✓	Beam	175028	LM, ES, MS, NZ	Beam with current best values (TH 126, single TDAC config) - Long run over lunch	<a href="#">ov</a>
<input type="checkbox"/>	<a href="#">4884</a>	Fri Jun 20 12:44:07 2025	Fri Jun 20 12:45:59 2025	✓	Beam	181612	LM, ES, MS, NZ	Beam with current best values (TH 126, single TDAC config) - Long run over lunch	<a href="#">ov</a>

Run Numbers (CSV)

Built with Cursor

# Example of JSON RDB document

```
{
  "_id": "685542f4d3363bdaee372443",
  "BOR": {
    "Run number": 4898,
    "Start time": "Fri Jun 20 13:13:46 2025",
    "Shift crew": "LM, ES, MS, NZ",
    "Run Class": "Beam",
    "Beam Blocker Open": true,
    "Mu3e Magnet": 1,
    .. snip ..
  },
  "EOR": {
    "Stop time": "Fri Jun 20 13:15:38 2025",
    "Events": 174646,
    "File size": 3174616428,
    "Uncompressed data size": 3847312505,
    "Comments": "Beam with current best values (TH 126, single TDAC config) - Long run over lunch"
  },
  "History": [
    {
      "date": "2025/06/20 13:16:04",
      "comment": "Database entry inserted "
    }
  ],
  "Attributes": [
    {
      "DataQuality": {
        "mu3e": "-1",
        "beam": "-1",
        "vertex": 1,
        "pixel": "-1",
        "fibres": "-1",
        "tiles": 1,
        "calibration": "-1",
        "links": "-1",
        "version": "unset"
      },
      .. snip ..
    },
    {
      "RunInfo": {
        "date": "2025/06/20 13:24:57",
        "Significant": "true",
        .. snip ..
      }
    }
  ],
  "Resources": [
    {
      "type": "application/pdf",
      "fileId": "685c1dd2d3363bdaee373a1d",
      "filename": "summary-4898.pdf",
      "uploadDate": "2025-06-25T16:03:30.215Z",
      "description": "ov"
    },
    .. snip ..
  ]
}
```

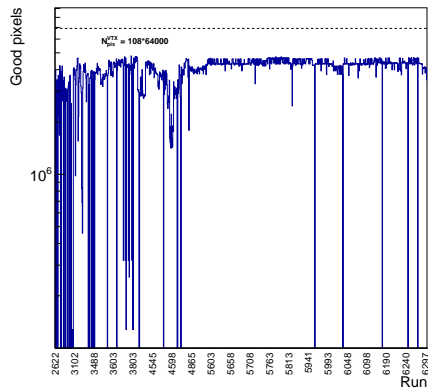
([mu3ebe:/data/mu3e/cdb/runrecords/runRecord\\_4898.json](mu3ebe:/data/mu3e/cdb/runrecords/runRecord_4898.json))

# Runs: beam, cosmics, . . .

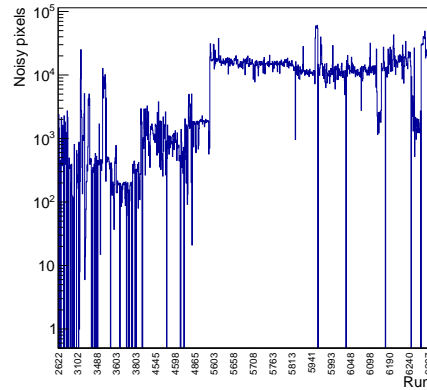
- Run periods

- ▶ **beam**: 2622 - 6301 with magnet ON
  - except for **cosmics**: 5845 - 5890 (5890 is partially cosmics/beam)

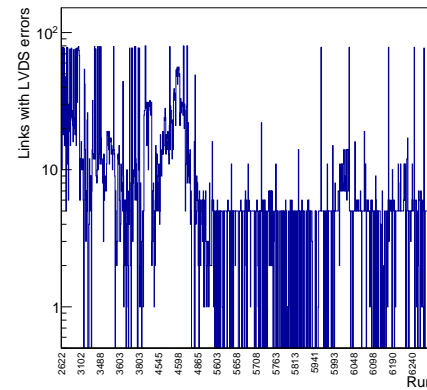
Total number of good pixels per run



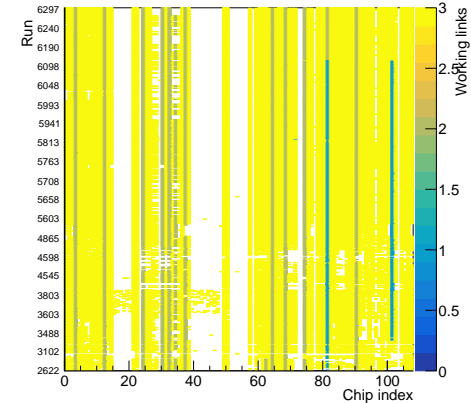
Total number of noisy pixels per run



Number of links with LVDS errors per run

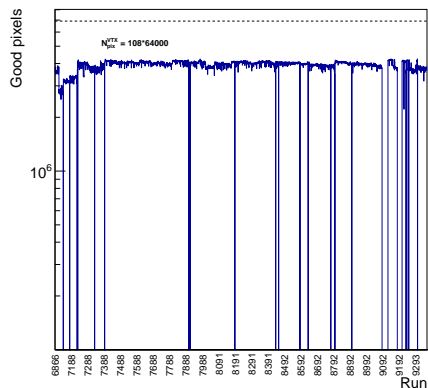


Working links per chip

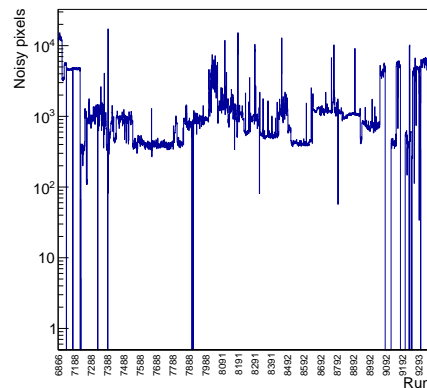


- ▶ **cosmics**: 6865 - 9361 with magnet OFF

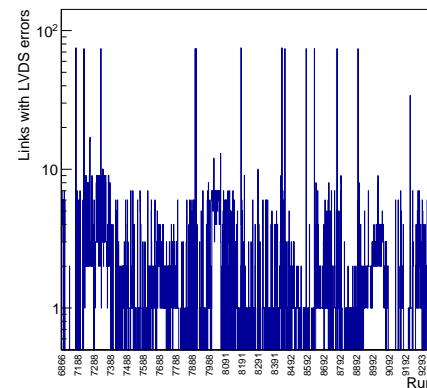
Total number of good pixels per run



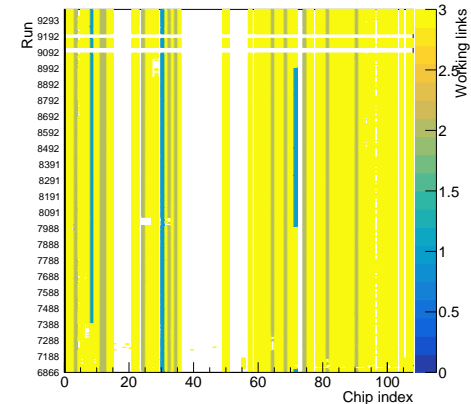
Total number of noisy pixels per run



Number of links with LVDS errors per run



Working links per chip



# Leftovers

# LVDS errors (from Midas metadata)

---

- `tools/midasMeta/mu3e_midas_meta` results in
  - ▷ `linkMatrix`  
mapping from chip submatrix A,B,C to link offset 0,1,2. Can be "E" (error)
  - ▷ `linkMask`  
link masked? (1 = unmasked, 0 = masked)
  - ▷ `lvdsErrRates`  
maximum error rate of link during run in Hz, > 10 Hz considered bad

- Observations

Comment	linkMatrix	linkMask	lvdsErrRates	Conclusion
Good	CAB	111	0,0,0	Good data
Unmasked Lnk (err rate)	CBA	111	36,17,0	Reject entire chip
One masked err link	ECB	011	0,0,0	Keep active links
Submatrix duplicated	BEB	111	big,big,big	Reject entire chip
Masked large err rate	BEC	000	big,big,big	Reject entire chip
"Masked err links"	EEE	000	0,0,0	Reject entire chip
"Unmasked 3 err link"	EEE	111	0,0,0	Look at lvdsErrRates



# PixelQualityLM

---

- "PixelQualityLM" to capture quality of
  - ▷ chips, links, columns, pixels  
0 = good, 1 = noisy, 2 = suspect, 3 = declared bad,  
4 = LVDS errors on link, 5 = LVDS errors from other link,  
8 = no hits, 9 = masked
- Algorithm (currently)
  - ▷ **links**: Use LVDS errors from midas meta data and hit counts  
"4" if link has LVDS error rate  $> 10$  Hz or (unmasked) 'E'  
"5" if other unmasked link has LVDS error rate  $> 10$  Hz  
"8" if no hits on link  
"9" if link masked
  - ▷ **columns**: determine average column hit count  
"9" if column hit count  $< 10\%$  of average(!)  
"3" if more than 40 noisy pixels in column
  - ▷ **pixels**: determine average chip hit count  
"1" if pixel hit count  $> 10\sigma$  above average

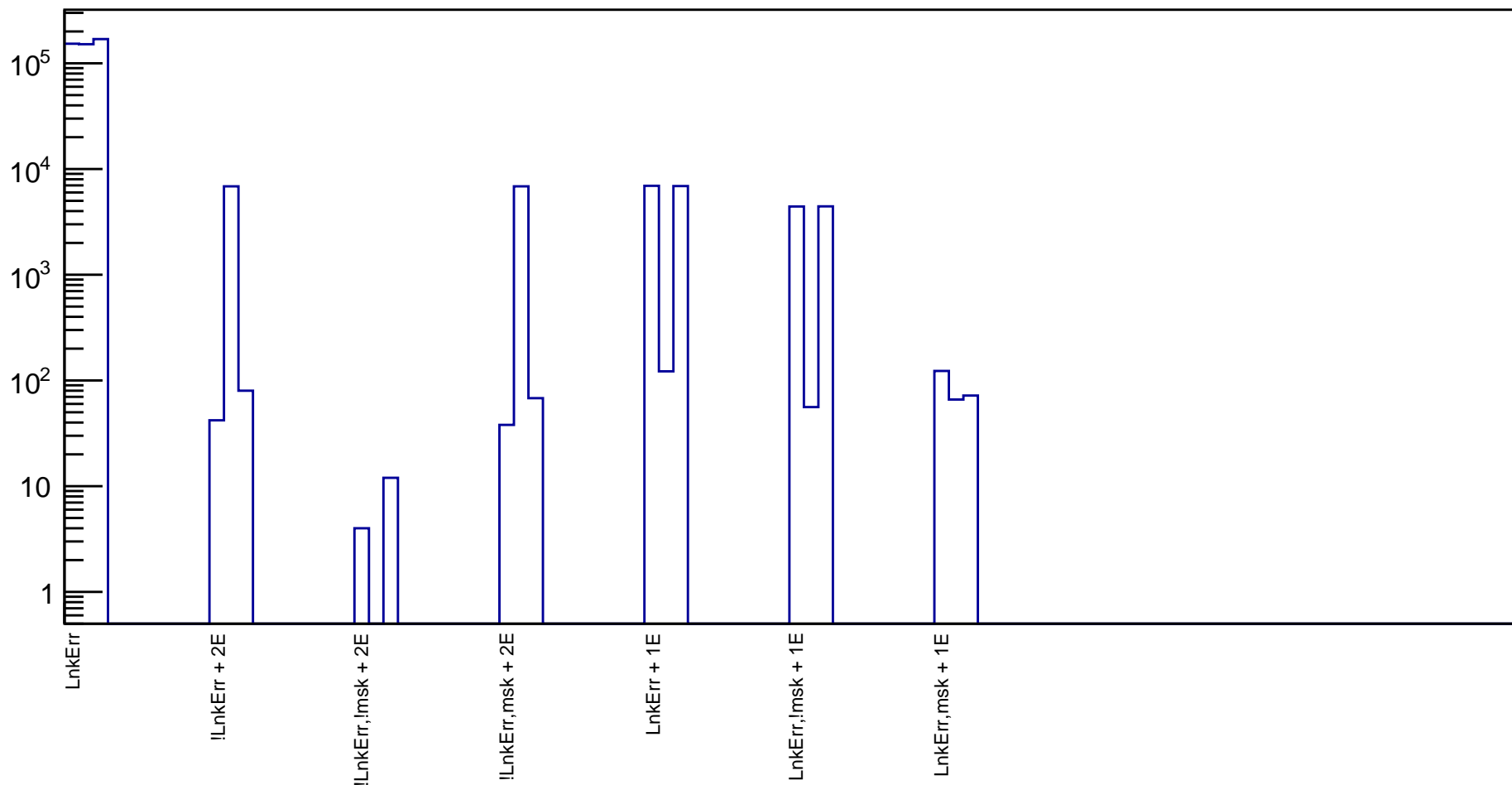
→ improvements possible!
- Trying to extract "good" run list and sync with Jak

# ABC Link Error Status (all sig. runs)

- **Very preliminary!!**

- ▷ mostly not interesting for P&S, but essential for my validation
- ▷ mostly driven by (2)3 chips: (35), 161, 1126

abcMatrix



# Without those chips

---

abcMatrix

