R3B DCS 2020

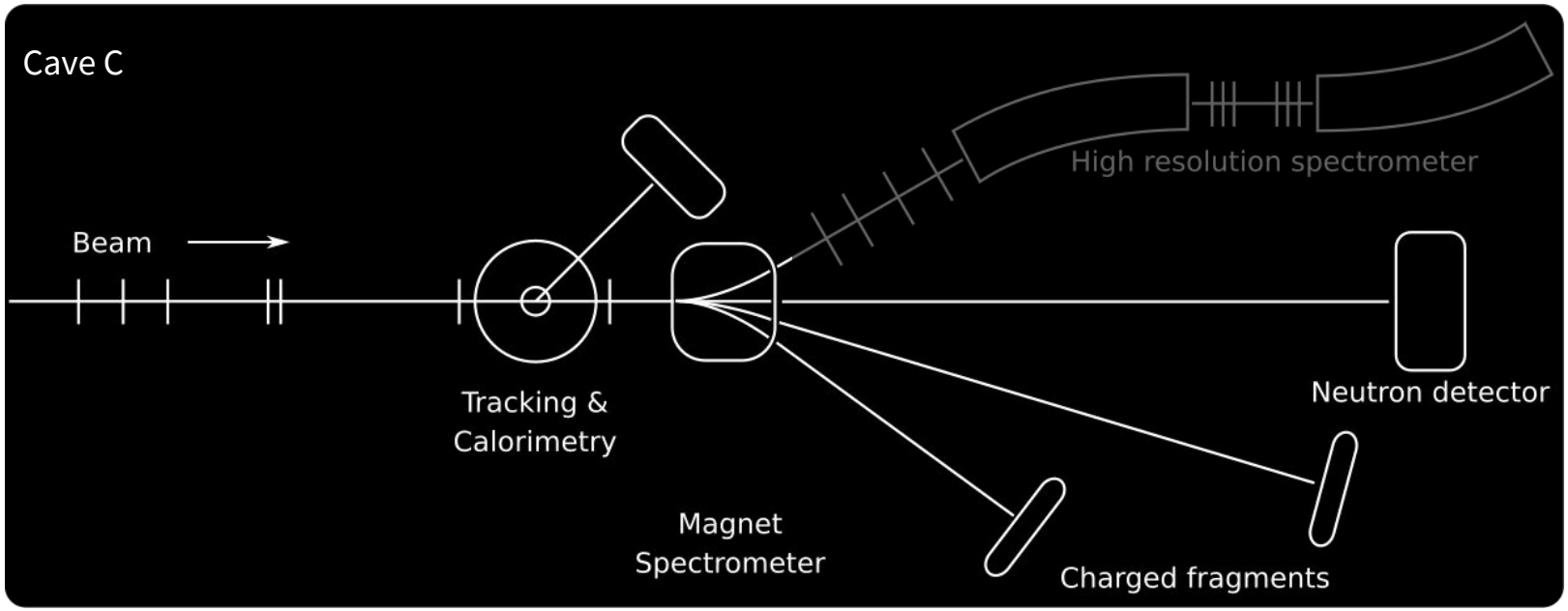
Bastian Löher (GSI) - C²DCS Meeting December 2020

Find the newest version of this presentation here:
https://docs.google.com/presentation/d/1RBTOeYpQERHvMuevA-VZ1FMwkWF1K2C-lw9JbC_kQvw
Additional slides here:
http://web-docs.gsi.de/~bloeher/rep/2019_r3b_meeting/bloeher.html

The R3B setup

Reactions with relativistic beams in inverse kinematics

- fixed target experiment, allows high-resolution kinematically complete measurements
- tracking of incoming beam & outgoing fragments (neutrons, light and heavy charged fragments)



The R3B setup - Under control

Scope

- Goal is to control / monitor as much as needed for remote operation, minimize access to Cave C
- Parameters under control / monitoring:
  - Detector front-end cards
  - High / Low voltage power supplies
  - Power distribution units / power strips
  - VME / NIM crates
  - Motor controllers
  - Pressure gauges
  - Gas systems
  - Temperatures
  - Scopes
  - GSI FESA Accelerator parameters
  - R3B DAQ
  - some more I forgot…
- **Today**: Bits and pieces that have solved some of our problems

The R3B setup - Under control

Tools

- Building blocks
  - normal IOCs
  - PyEpics/bash - mostly small scripts
  - caproto - run control application
  - libca - interface with C (e.g. DAQ, helpers)
- Visualisation
  - old fashioned MEDM (CSS too cumbersome over SSH)
  - r3bcavalcade: SDL-based, ADL-parsing high performance GUI
- R3B specific
  - Monitoring tools
  - Control tools
  - Automation helpers

R3B Monitoring

Components

- Parameter Archiving - epics_collector
- Detector status - high level IOCs
- Common user entry point - whatsup
- Visual feedback - colorspill

New in 2020

- High performance visuals - r3bcavalcade
- Who is watching the watchers?

R3B Monitoring

epics_collector

- Problem in 2016:
  - want to store information from EPICS
  - not using CSS, BEAUTY not an option
  - archiver appliance seems too large
  - channel archiver old and busted?
  - no need for *relational* DB
  - no need for *millions* of PVs
- Additional request:
  - Store parameter data in the DAQ data stream
- Solution:
  - generic C library that connects to and monitors groups of PVs based on PV lists
  - several applications:
    - watcher (like camonitor)
    - archiver (using influxdb as backend)
    - drasi/mbs (acting as DAQ node, writing to data stream)
  - total 5k lines of code

R3B Monitoring

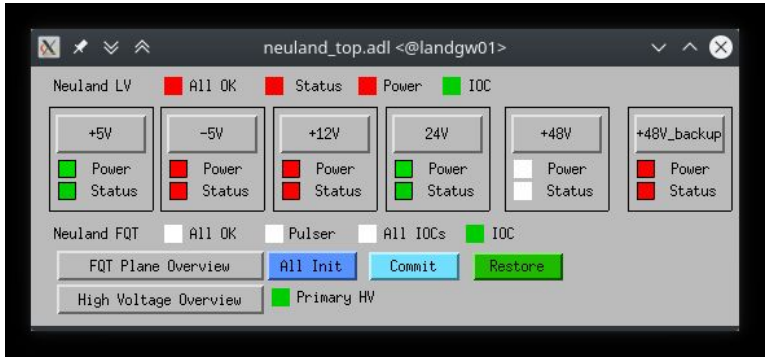
archiver

- Database: influxdb
- Visualisation: Chronograf
- Configuration via PV lists
  - automatically generated
- Currently logging ~600 PVs
- Command line interface to retrieve values
- Automatic restart

R3B Monitoring

high level IOCs

- Problem:
  - EPICS favors divide and conquer strategy → many IOCs for a single detector
  - Users would like to know the big picture
- Solution:
  - Per-detector high-level (soft) IOCs
  - Collection of:
    - seq records → mostly triggering fanout actions (e.g. “All On”, “Init”)
    - calc records → mostly fan-in / summary of status information
  - Definition of “Standard records for NUSTAR DCS”, that every IOC should implement
    - :ioc:status → bi record OK / not OK
    - :ioc:error → state record with error message
    - :ioc:state → state record with name of state
- Status:
  - Almost too much maintenance, need to automate more



R3B Monitoring

single entry point

- Problem:
  - EPICS favors divide and conquer strategy → many IOCs for a single detector
  - Users would like to know where to start
  - Information scattered through the Wiki → hard to remember
- Solution:
  - Top level GUI element that links to everything else
  - Type ‘whatsup’ anywhere, and the user sees what’s up

- Today it looks a bit different, moving to auto-generated GUI

R3B Monitoring

colorspill

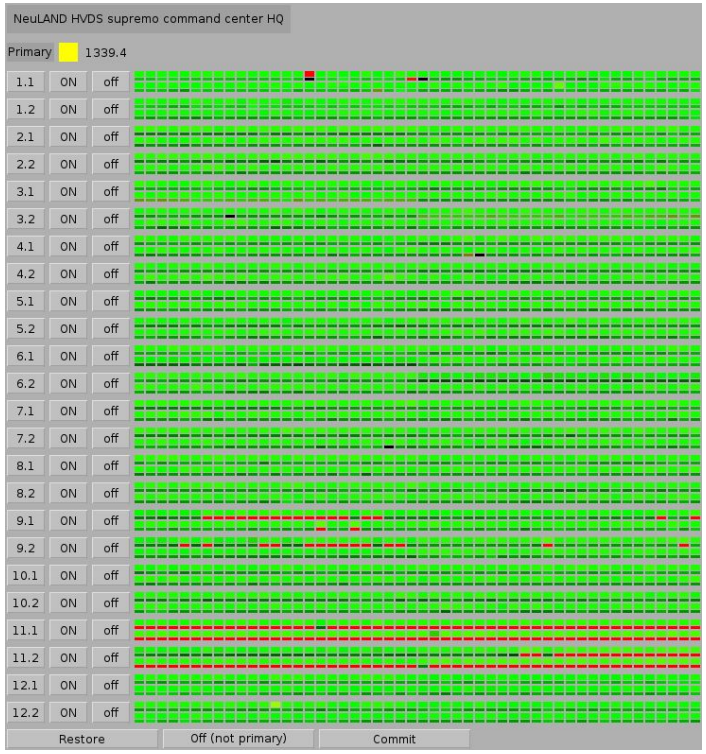
- Problem:
  - Life in the Messhütte can get boring at 3 am
    → Failures may go unnoticed
    → Long time to react
- Solution:
  - Install EPICS controlled RGB flood lights
  - Change colors based on system status
  - Beam on/off
  - DAQ failure
  - ...

Amazon

R3B Monitoring

r3bcavalcade

- Problem:
  - Neuland detector has many channels (currently 2400, planned 6000)
  - Users want to check voltage status on a single overview
  - MEDM goes into 100% CPU usage, slow update
- Solution:
  - Replace MEDM in this case with an SDL-based program
  - Add also current monitoring
  - Updates @ ~10 Hz
  - CPU usage < 10%
  - Also fine over SSH
- Alternative?
  - How does phoebus fare with that many elements?



R3B Monitoring

quis custodiet ipsos custodes?

- Problem:
  - Things will go down / fail / reboot / get stuck
  - IOCs and services should come up again after failure
  - Would like to check status on a web page
- Solution(s):
  - procServ + cron: works! but no easy status overview
  - screen + cron: works! but one can use procServ, too
  - systemd: works! haven’t tried yet
  - supervisord: works! simple to set up, simple web interface, extendable
- Any suggestions from your side? What works, what doesn’t?

R3B Control

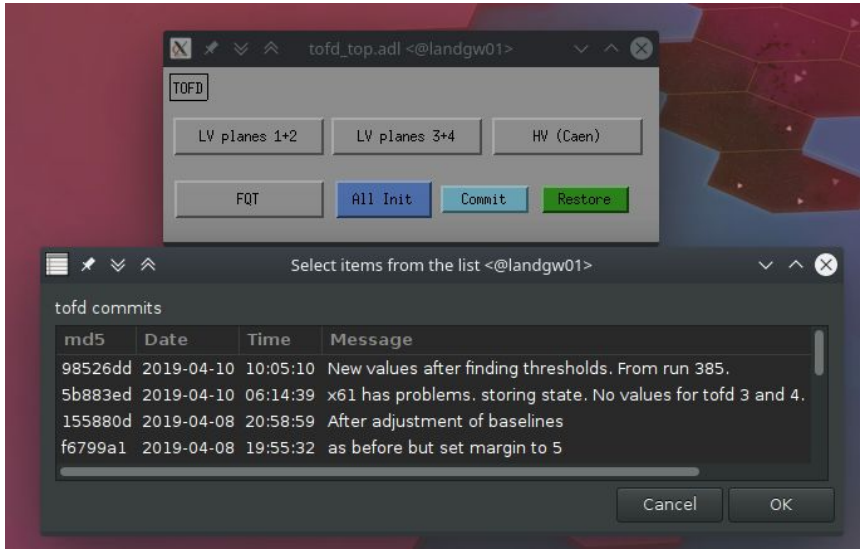
Tasks

- Parameter setpoints - commit/restore
- Controlling the DAQ - run control
- Getting input from the DAQ - adhocsoftioc

R3B Control

commit/restore

- Problem:
  - BURT is OK, but that’s about it
  - autosave / bumpless reboot does just that
  - Users would like to save and restore detector state, and save a note (timestamps too?)
- Solution:
  - git to the rescue!
  - Input: automatically generated PV lists
  - Central slow control repositories
  - One-button commit (+ commit message)
  - One-button restore (choose from list)
    - List is basic zenity



R3B Control

run control

- Problem:
  - DAQ and controls are two separate things
  - Users would like to collect information when starting / stopping data taking
- Solution:
  - run_control
    - interface to DAQ (start / stop / status)
    - big, friendly buttons
    - gather meta information from EPICS, FESA, files
    - generate electronic logbook entry, run log
    - extensible via plugins
    - multi-user support
  - written in Python
  - using caproto to dynamically build an IOC
  - controlled via MEDM
  - run once, control from anywhere

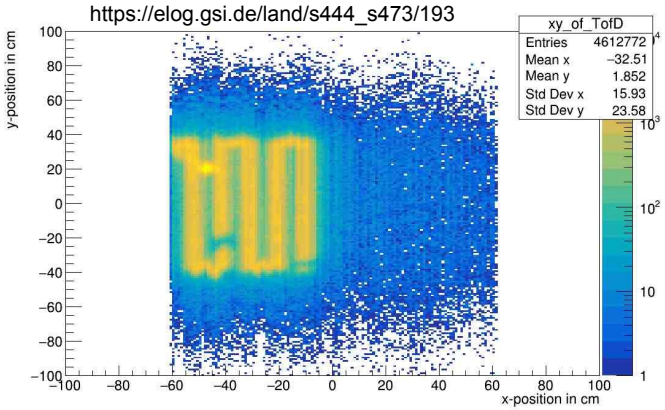
ID	Date	Author	Category	Experiment	Phase	Subject
566	Thu Apr 11 09:05:19 2019	runcontrol	**Runs**	s454	Experiment	r3b run0410 stop

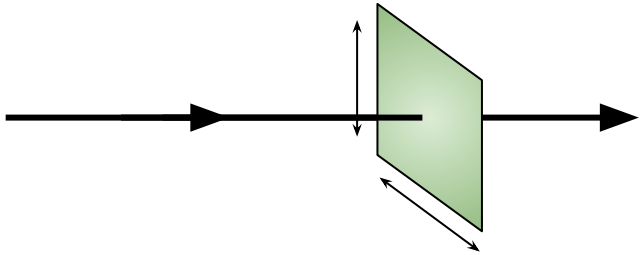
```
Experiment: s454
Run: 410 stopped
Message:
On shift: Konstanze & Hans & Bastii
Beam ion: 16O
Beam energy: 500.0 AMeV
GLAD current: 2337.0 A
----- Pipeline r3b_main (0) BEGIN -----
Pipeline: r3b_main (0)
Stop time: 2019-04-11 09:05:18.913343
Run time: 57224.538261175156 s
----- Pipeline r3b_main (0) END -----
TrloII Scalers (r3b_trloii) begin-of-spill
-------------------------------------------------------------------------------
Input                    #             before DT   after DT   after DS  DT [%]
-------------------------=-----------------------------------------------------
  0:          0 ToFd34OR   # ToFd12m             0          0          0   -
  1:          0            # ToFd12OR            0          0          0   -
  2:          0            # ToFd34OR            0          0          0   -
```

R3B Control

input from the DAQ

- Problem:
  - DAQ and controls are two separate things
  - EPICS should display information from the DAQ (scaler values)
  - EPICS should react on DAQ status (start / stop motion, Messhütte lights)
- Idea:
  - add EPICS PVs to parts of the DAQ software
  - listen to DAQ PVs to control other parts
- Problem 2:
  - writing an add-on CA server in C is a bit of a hassle
- Solution:
  - adhocsoftioc: dbgen + fmioc
  - dbgen: generate a DB file during runtime
  - fmioc: fork a softIoc that uses the new DB
  - use normal libca to use the softIoc





Beam has spill structure
Normally, sweep run produces ‘holes’
Motion control knows about beam state
→ Sweep run without spill structure

Helpers

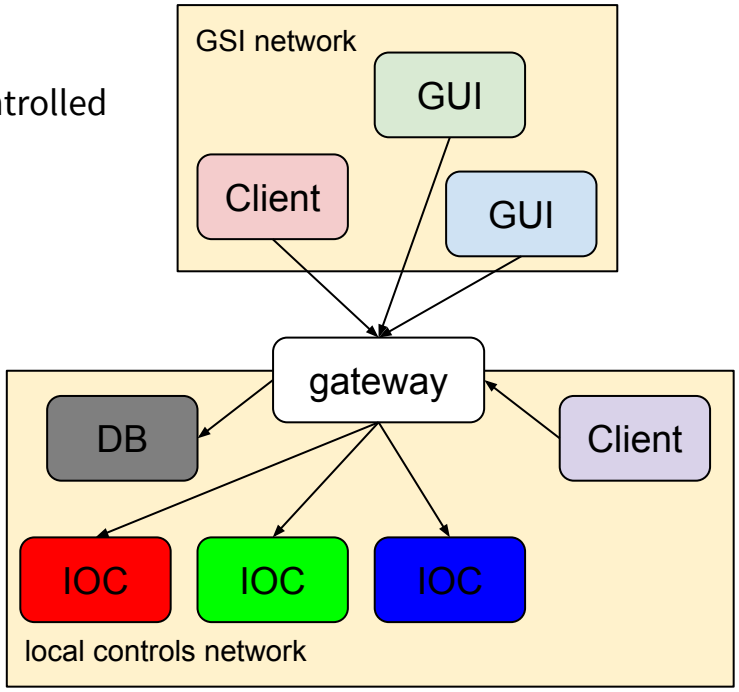
Toolbox

- A new gateway - r3bcagw
- DNS / DHCP - r3b_sc_hosts
- Generators - r3bmap
- Network boot
- Power cycling
- Raspberry Pi I2C - nohadcon

Helpers

Local network + gateway(s)

- Pre-2016 many problems:
  - new devices need to be registered with central IT
    → think twice, if the new device really needs to be controlled
  - IT complains about strange data from our devices
    → frequent bug hunting
- Since 2016:
  - private slow control network with our own DNS
  - misbehaving hardware is contained
  - one gateway as fixed entry point
  - stop leaking EPICS broadcasts into GSI network
- Problem:
  - Standard EPICS gateway slow with many rules
  - Gateway should do aliasing (1000s of rules)
- Solution:
  - New gateway implementation: r3bcagw
  - Uses lists of aliases (automatically generated)
  - Much faster, but still a work in progress
- Currently moving to new hardware



Helpers

DNS / DHCP

- Problem:
  - We are lazy
  - DNS / DHCP uses same/similar information in several files
  - Should be kept in sync
- Solution:
  - Keep information in one place
  - Generate files for BIND and DHCP server → r3b_sc_hosts
  - Handles several subnets
  - Handles grouping and options for e.g. PXE boot

Helpers

Generators

- Problem:
  - We are lazy
  - DB files, ADL files, PV lists are repetitive
- Solution:
  - Keep around fundamental mapping information (e.g. which detector channel is connected where?)
  - Generate DB, ADL, PV lists from this info and install in appropriate places
  - Collect all scripts in one repository → r3bmap
  - make && make install
  - Generates 6 MB of files from 20 kB of input
  - Also generates files for data unpacking and analysis
- Great boost to maintainability & flexibility
  - Most actions require a change in a single file + rebuild + install
    - Hardware replacement (hostname change)
    - Mapping change
    - ...

Helpers

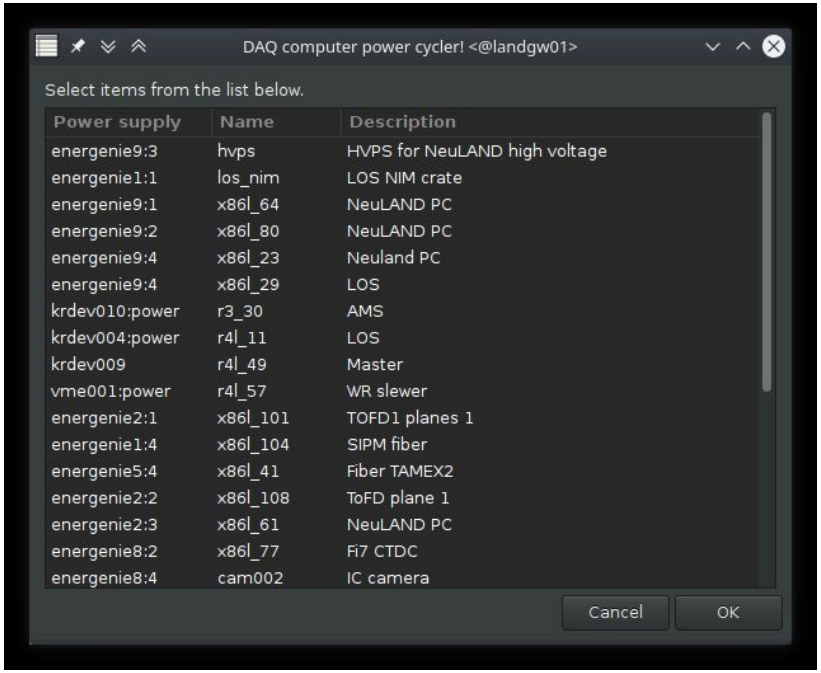
Network booting

- Problem:
  - We are lazy
  - Many installations are similar (especially for raspberry Pi SBCs)
- Solution:
  - Install boot server with common root file system
  - Boot from there instead of local disks
  - Start services / IOCs based on hostname
  - Time for setting up a new raspberry Pi host ~1 minute

Helpers

Power cycling

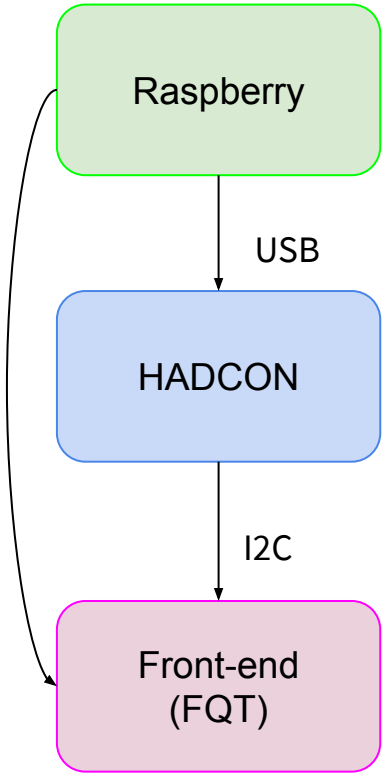
- Problem:
  - We are lazy
  - Things lock up, one needs to pull the power cord
- Solution:
  - Connect everything to power distribution units (PDU)
  - Pull the plug over ethernet
  - Works also for VME / NIM crates (via SNMP)



Helpers

Raspberry Pi / I2C

- Problem:
  - We are lazy and we are impatient
  - Use Raspberry Pi to control Front-end settings
  - Init cycle takes ~10-15 minutes (threshold finding)
    (This used to be ~1 h with LabView, before EPICS)
- Solution:
  - Remove HADCON, talk I2C directly
  - Communication 4x faster
- Side effect:
  - Use standard tools for debugging I2C comm



Summary

NUSTAR DCS web page:
http://web-docs.gsi.de/~land/nustar-dcs

Device support:
http://web-docs.gsi.de/~land/nustar-dcs/epics_device_support.html

- **Almost full control over Cave C**
- **Run control / logging**
- **Fast saving / restoring of values**
- **Everything under revision control**
- **Automation, where possible**

The R3B DAQ / controls team

- Hans Törnqvist, Bastian Löher, Haik Simon, Håkan Johansson

Questions

- **Cosylab? Any experience?**
- **Per-detector gateway?**
- **Anomaly detection?**