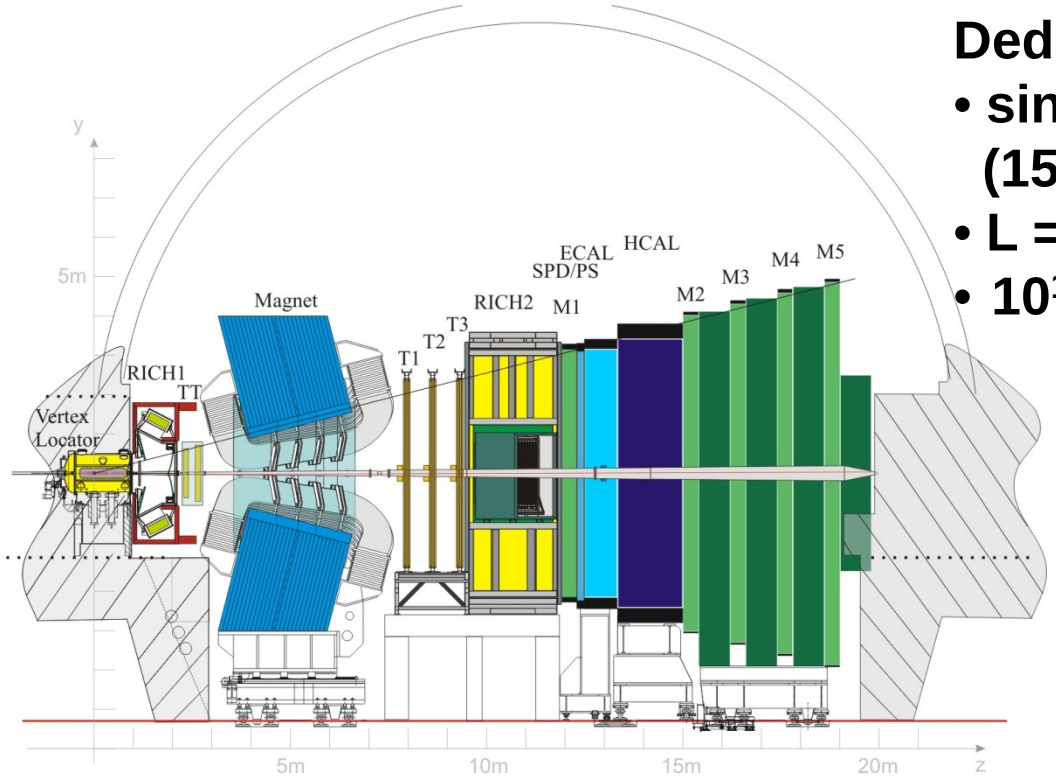
Control and Operation of the LHCb Readout Boards using Embedded Microcontrollers and the PVSSII SCADA System

Stefan Koestner
on behalf of the
LHCb Online Group
(email: Stefan.Koestner@cern.ch)

IEEE - Nuclear Science Symposium
San Diego, Oct. 31st 2006

The LHCb experiment



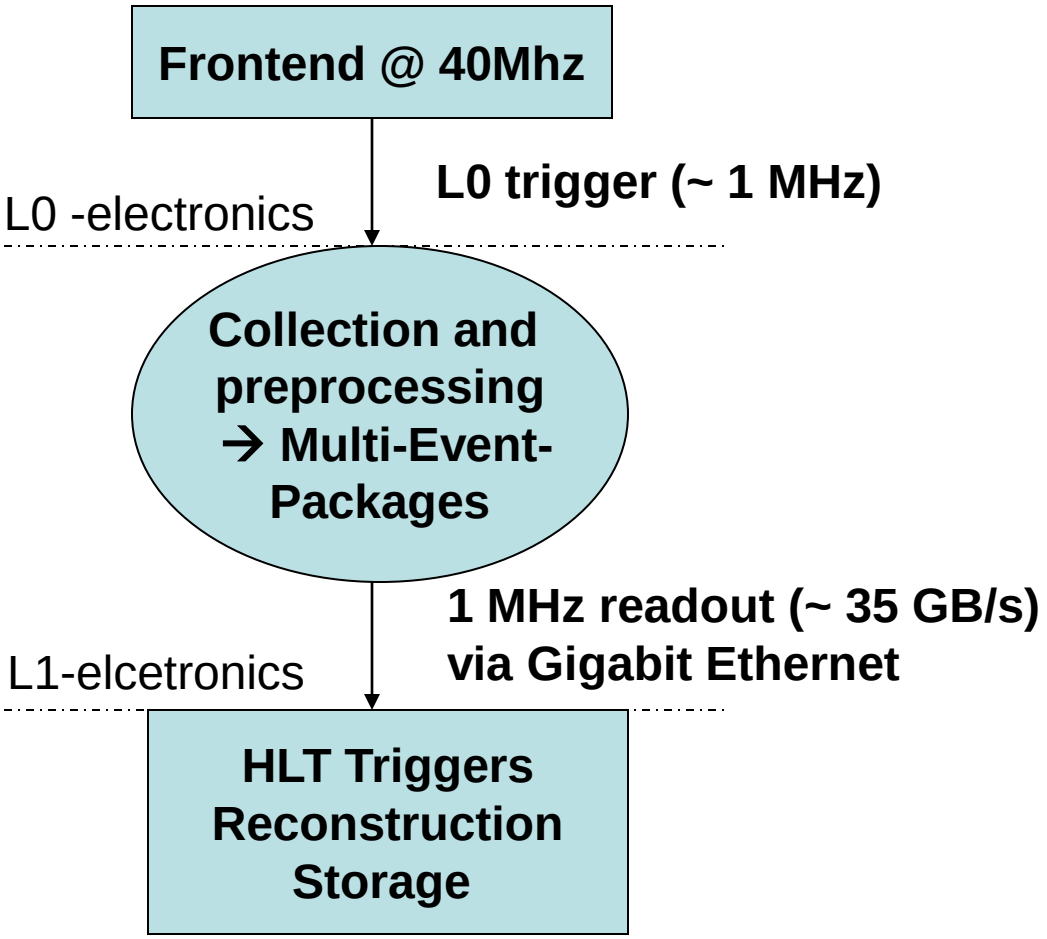
Dedicated to 'B-physics':

- **single arm forward spectrometer (15 - 300 mrad)**
- **$L = 2 \times 10^{32}\ cm^{-2}\ s^{-1}$ (~1 evt/bx)**
- **$10^{12}\ b\bar{b}$-pairs (~100 kHz)**

fraction of interest just 10^{-5}
→ efficient trigger necessary to enrich the data sample! (implemented in 2 levels)

- **Ten sub-detectors with approx. 1.1 million readout channels in total**
- **average total event size (at average occupancy) is 35 kB (after ZS)**

Trigger and DAQ system

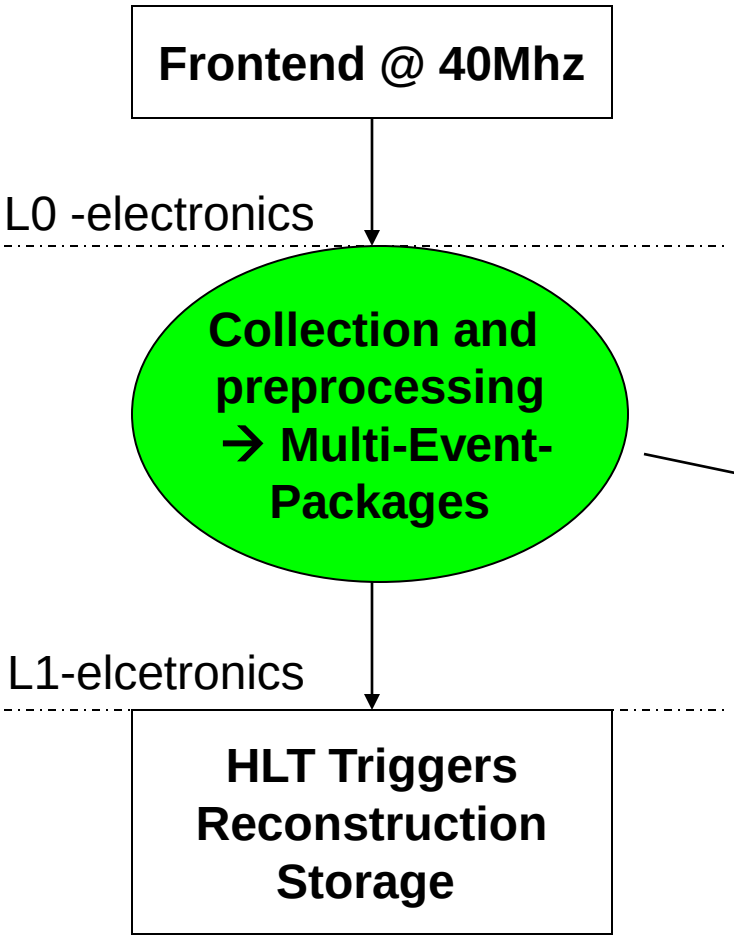


L0 trigger:
4 µs fixed latency
time synchronized to FE-chips (Calos, Muon & Pile-up Veto)

Push-protocol supposing that following stage has enough buffer.
Centralized throttle via TFC-system (synchronous).

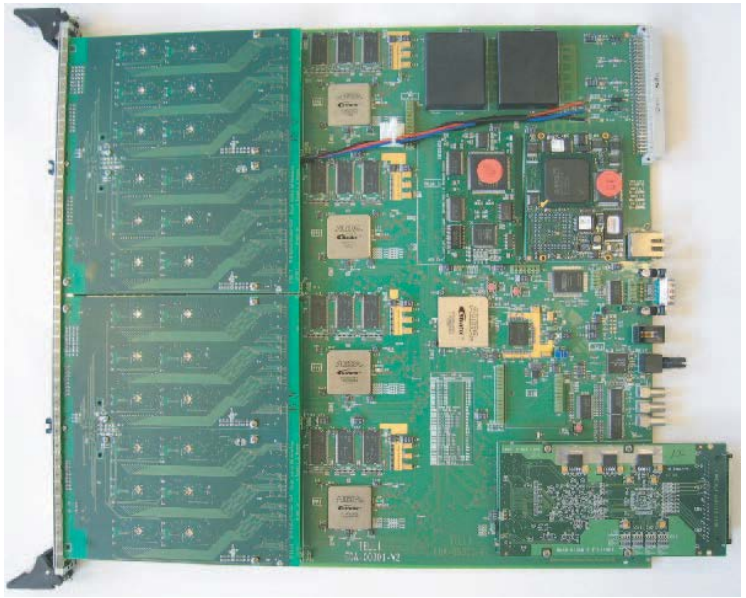
HLT trigger:
~2kHz to tape

TELL1 - board

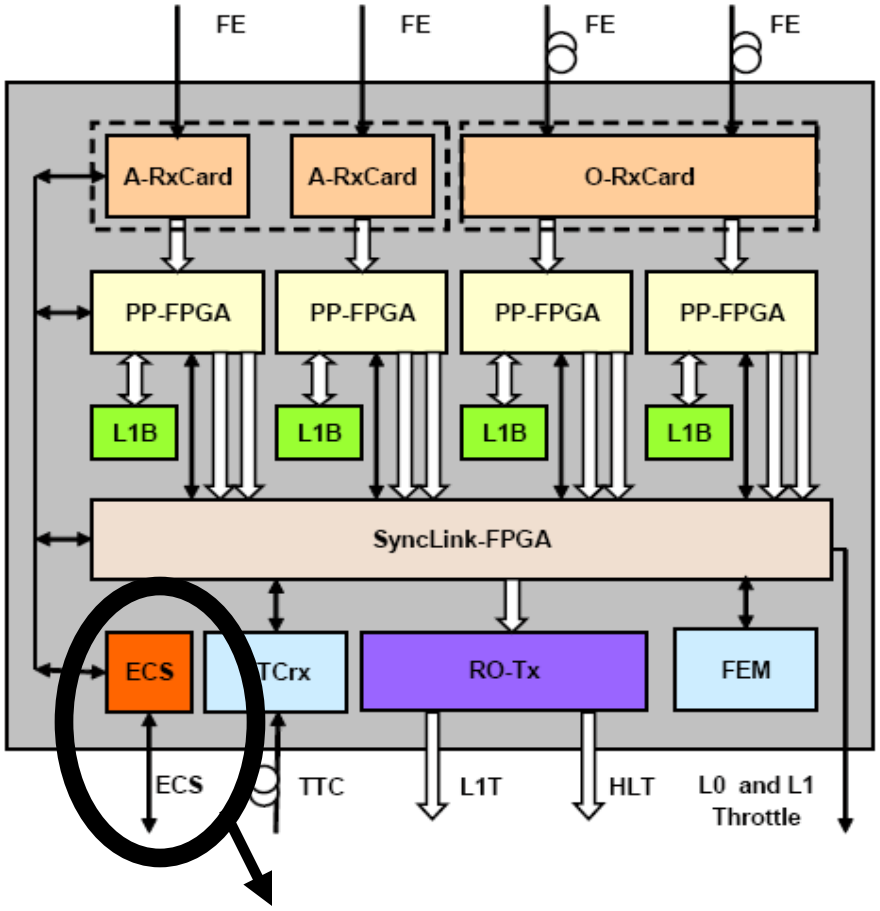


Trigger ELectronics Level 1 board:

- **FPGA based**
- **adopted by almost all subdetectors**
- **in total ~300 boards**
- **with ~400 registers & memory blocks**



TELL1 - board



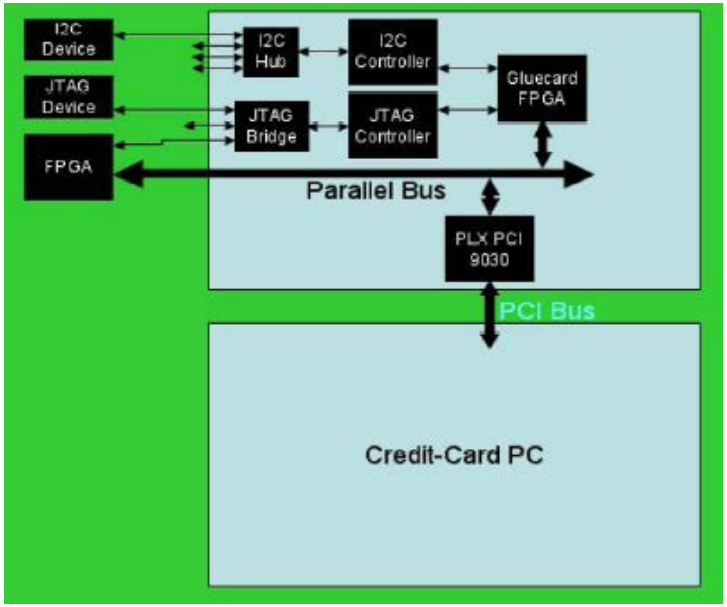
- ADC conversion
- Synchronisation, reordering, pedestal subtraction
- Common mode suppression
- Zero Suppression
- Multi Event Packing
- IP Framing
- Gigabit Ethernet Link

- ECS interface via creditcard sized PC connected to 10/100 Ethernet
- Controls network separated from Data network
- isolated access path → robustness

Credit Card PC

4” - SM520PCX (Digitallogic) CCPC:

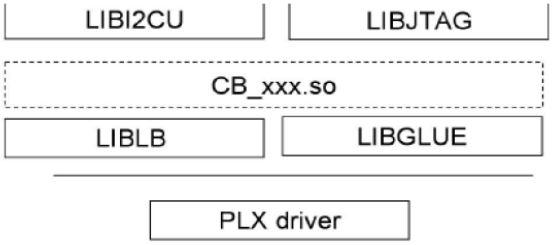
- **i486 compatible microcontroller (AMD ELAN 520)**
- **Linux Kernel at 133 MHz**
- **reading from local bus 20 MB/s**
- **filesystem shared over server**



Access to board via gluecard over a PLX PCI9030 bridge:

Parallel bus (8/16/32), 3 JTAG chains (2 MHz), 4 I^2C lines and 9 GPIO lines

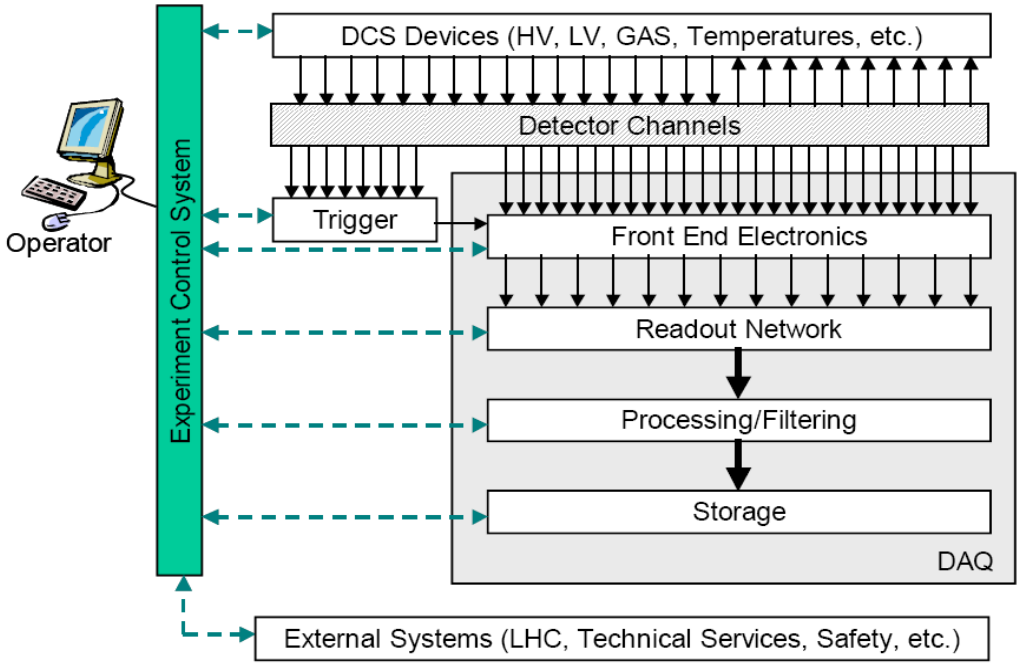
low level libraries under SLC4 (maintenance)



Experiment Control System

A uniform and homogenous control system based on the 'Joint Controls Project' (JCOP) framework - tailored for LHCb needs:

- **eases development as well as operation**
- **distributed and scalable system**
- **coherent interfaces**



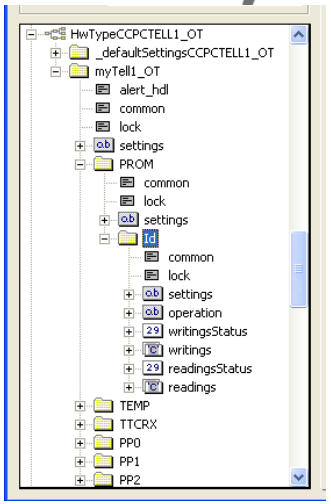
Configuration, monitoring and operation of the 'whole' experiment:

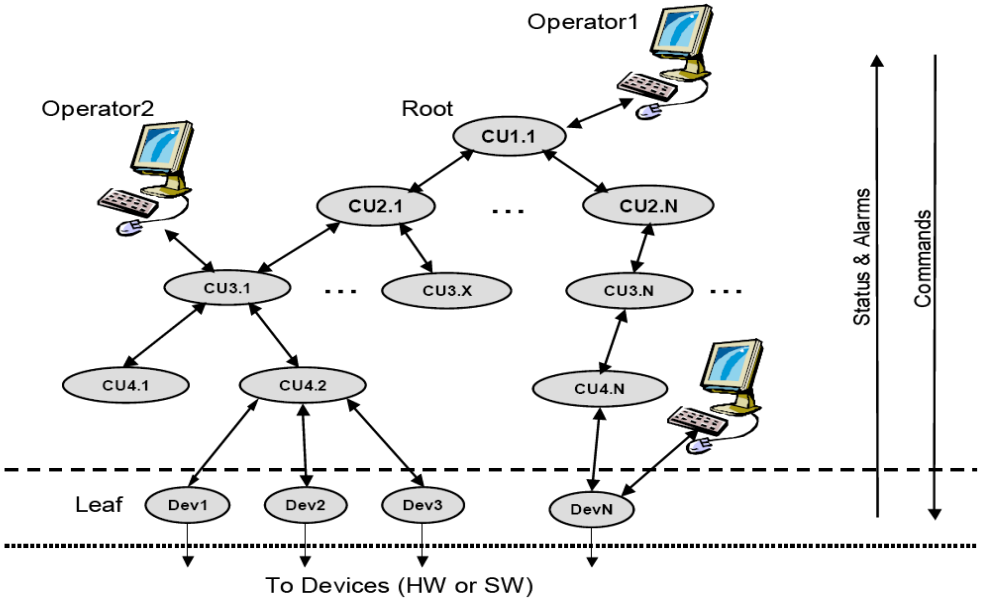
- **Data acquisition & trigger**
- **Detector Operations (DCS)**
- **Experimental Infrastructure**
- **Interaction with accelerator & CERN Safety System**

Industrial SCADA (PVSSII)

The CERN's common JCOP framework is built upon an industrial SCADA system: PVSSII

- **data (registers or probes) is stored in 'data points' (=complex structures) connected to the PVSS internal memory-DB**
- **Change of these entries triggers a callback function**
- **Upon certain thresholds alerts can be launched**





Extended to allow for finite state modeling & hierarchical control:

- **intuitive model of subsystem**
- **several levels of abstraction**
- **sub systems can run alone**
- **auto-recovery from errors**

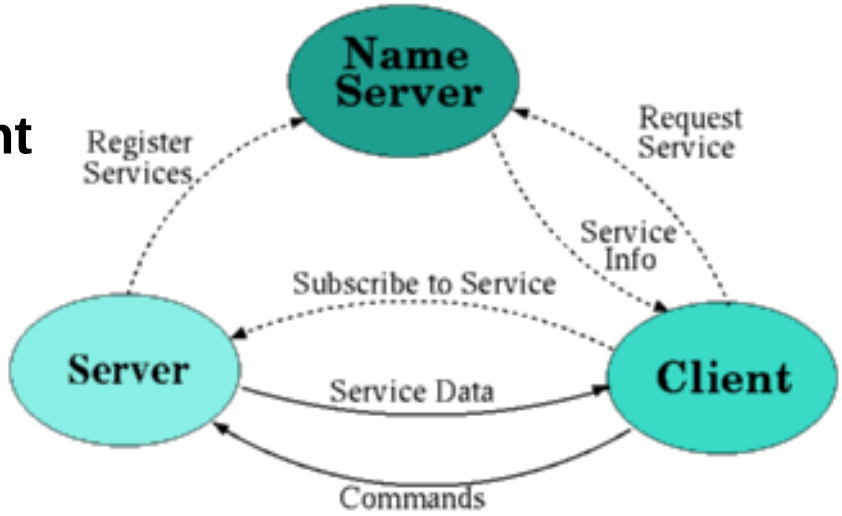
Communication Layer - DIM

Interface from ECS to hardware realized with the 'Distributed Information Management' System (DIM) – a portable lightweight communication layer:

DIM server running on CCPC publishes services to DIM Name Server (DNS) from where the client (ECS) can subscribe to it.

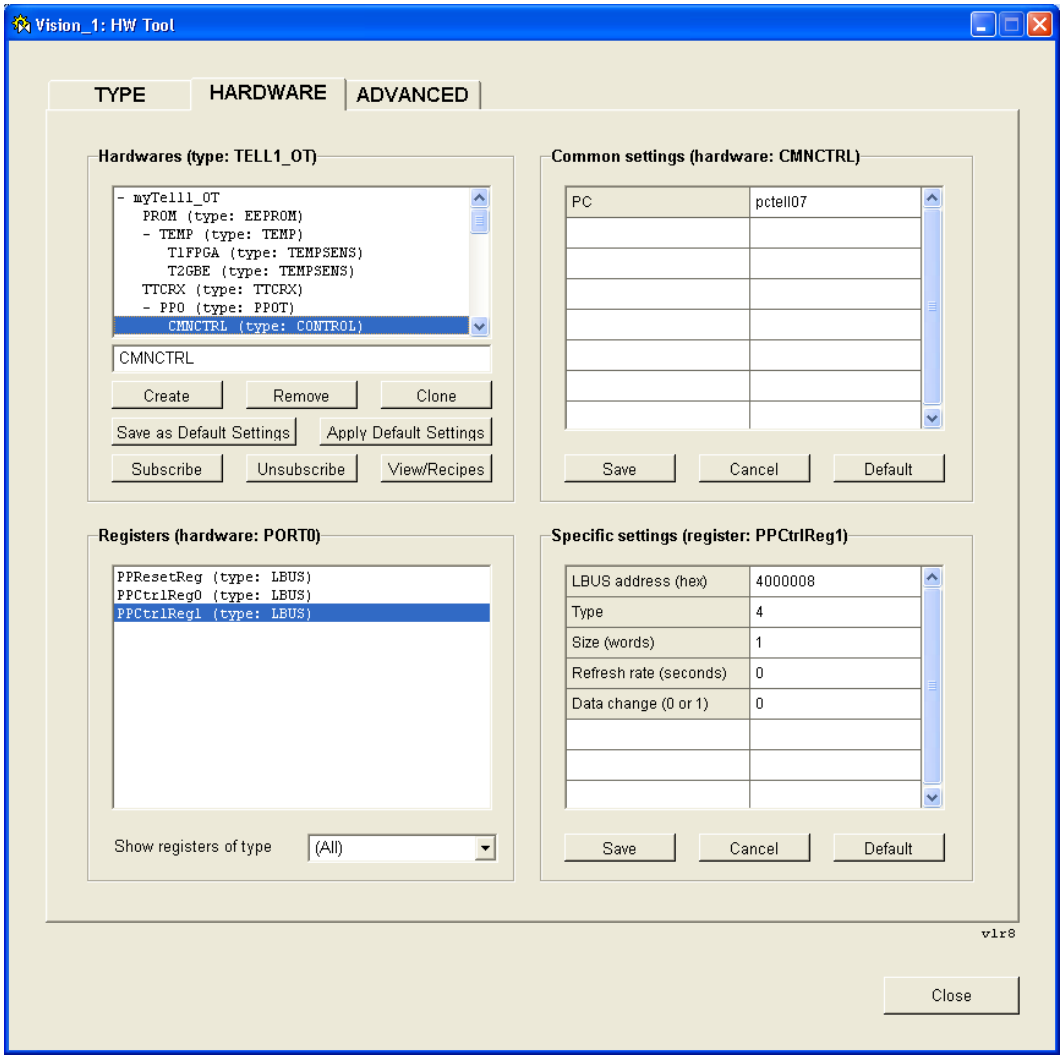
Data exchange peer to peer from server to client.

Client sends commands (write/read) – server updates services (data/status)



- **Portability: Clients can be installed on any machine just specifying DNS node (no need to take care of connectivity)**
- **Robustness: If server crashes it can easily republish on DNS node**

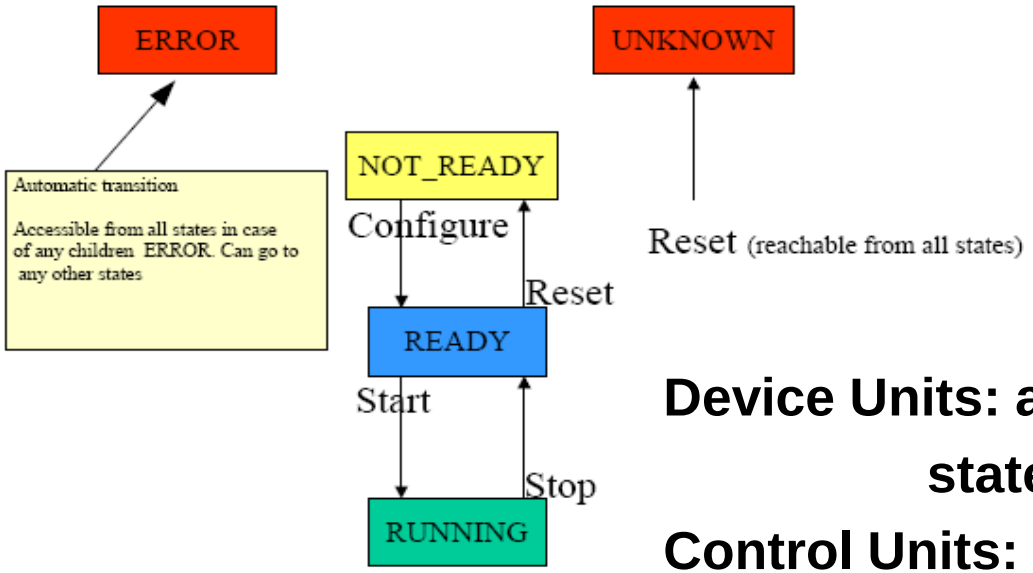
Abstraction Layer - FwHw



Introduced a tool to model hardware as PVSS-datapoints:

- **hides diversity and complexity of the various hardware/bus types**
- **takes care of the communication with the Tell1 (using DIM)**
- **framework function calls by register name**

Finite State Machines



Further abstraction as finite state machines (SMI++):

defined transition from one state of the board into another with possibility to autorecover

Device Units: act on hardware (data points)

state can be triggered by hardware

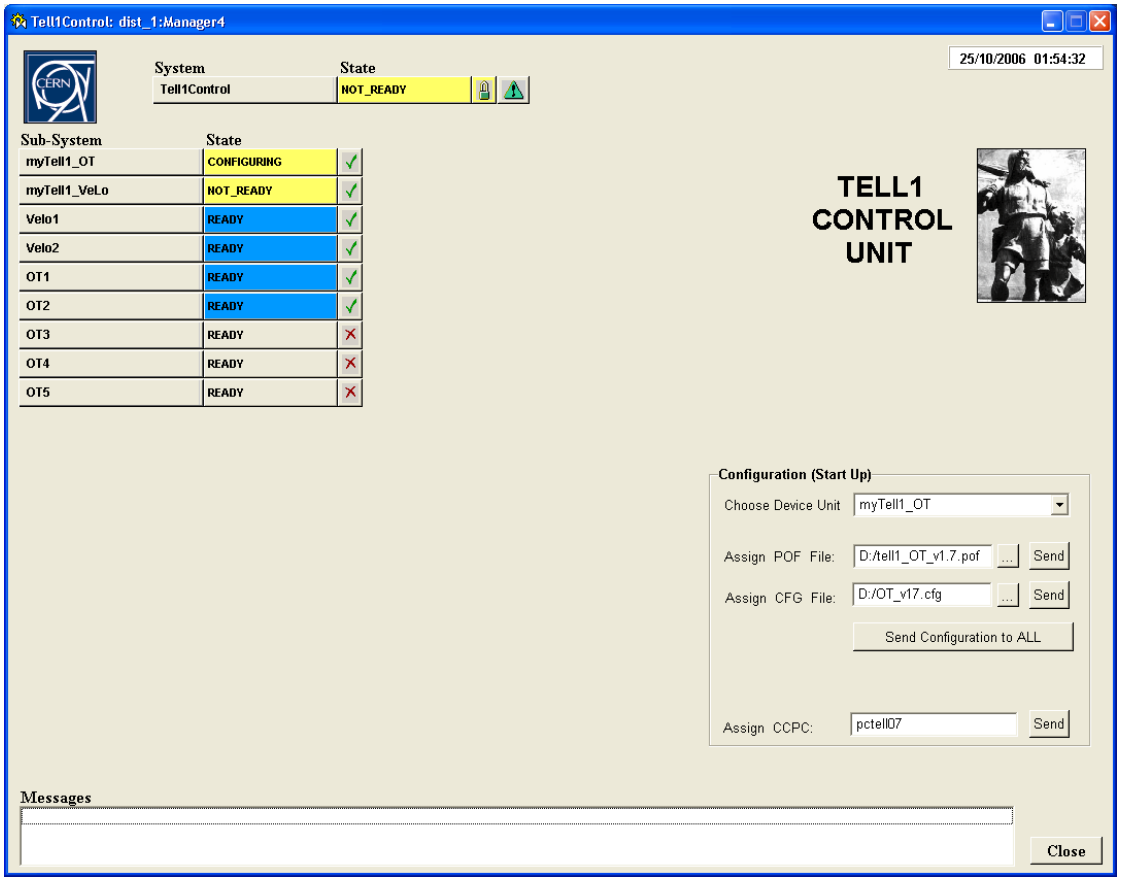
Control Units: send commands to children (DU)

state transition when children change

- **Firmware can be downloaded to board in state NOT_READY**
- **Configuration of board (mainly writing registers) via 'recipes' downloaded from Configuration Database (different settings for different run conditions)**

Tell1 Control Unit

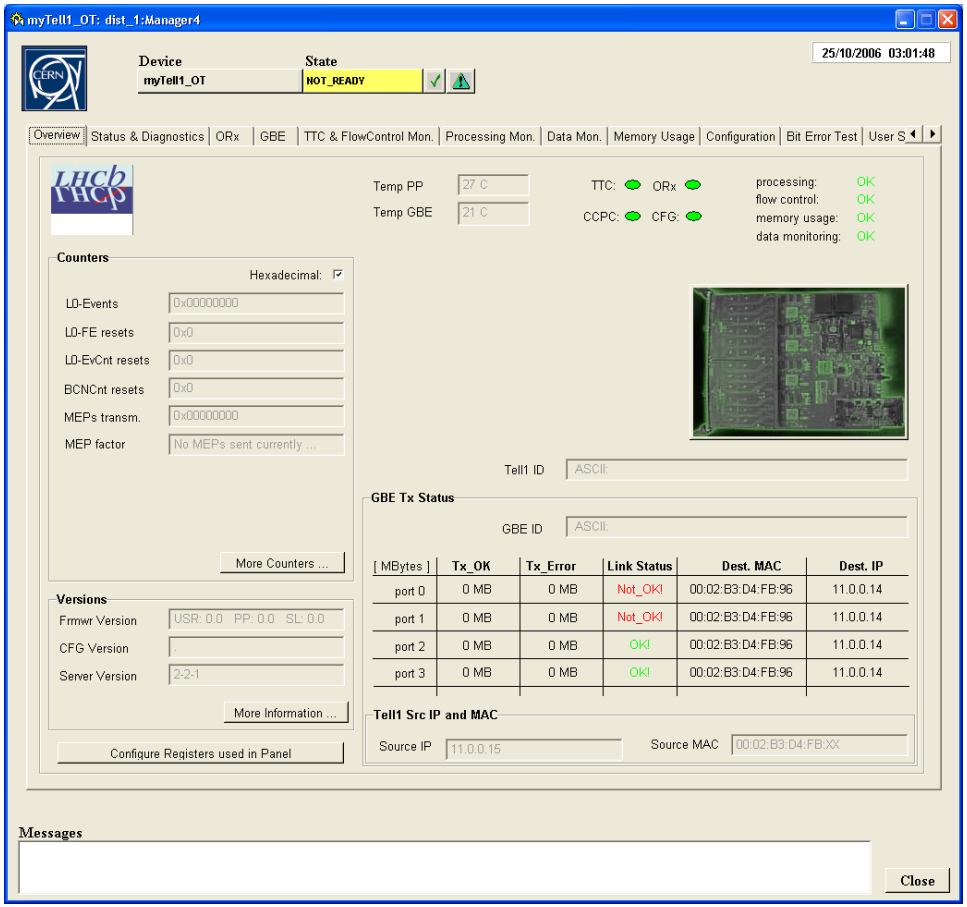
An example for a Tell1 CU panel (under construction)



- states of children (DUs) define state of CU
- commands are propagated downwards
- partitioning of the system is a key feature (partitions can have their own trigger)
- clicking on device unit allows to open operator panels for the device (boards)

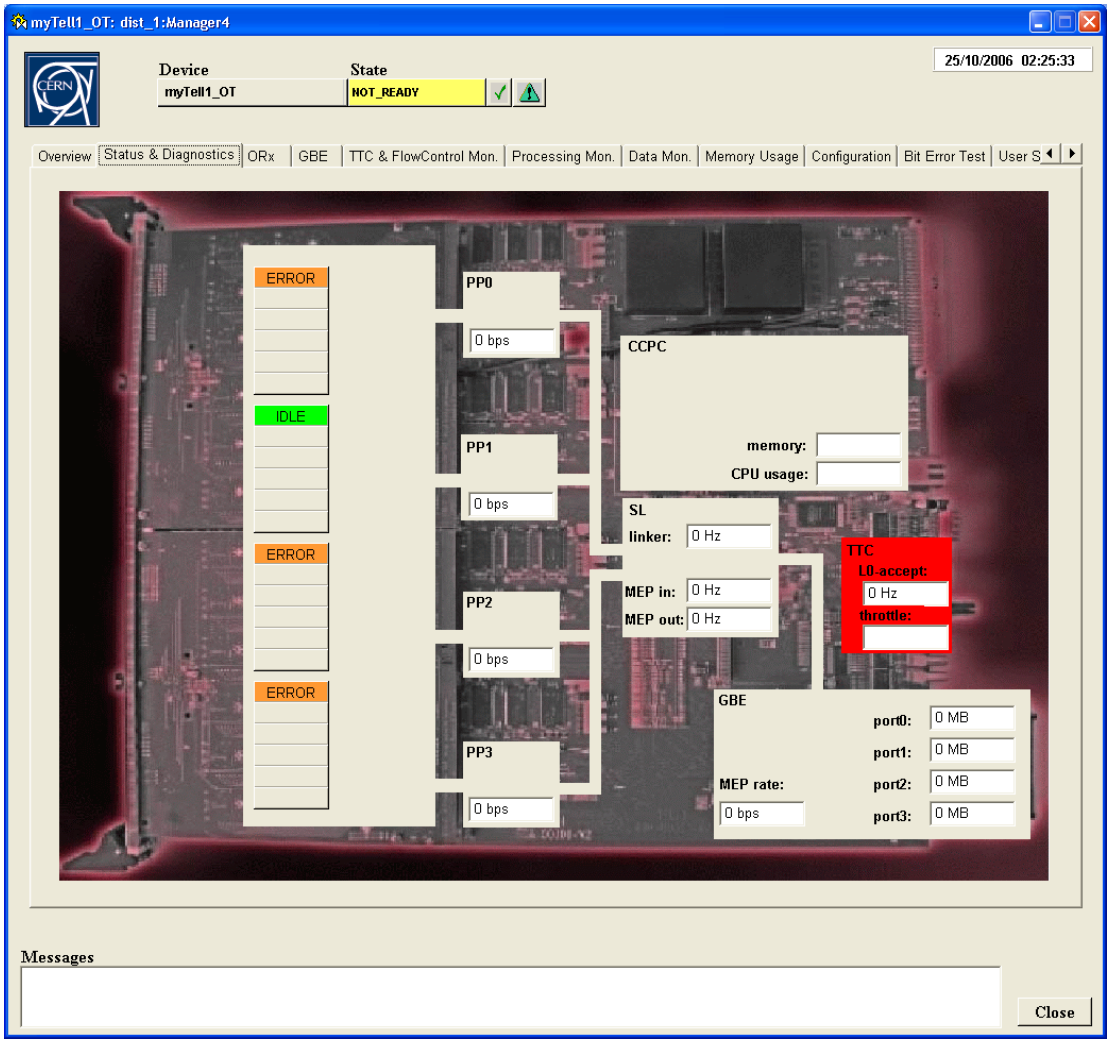
Tell1 Device Unit

Screenshot of a Tell1 DU panel – operator interface:



- identifies board type
- evaluates malfunctioning parts of the board
- quick overview of important counters and registers
- reconfiguration of refresh rate

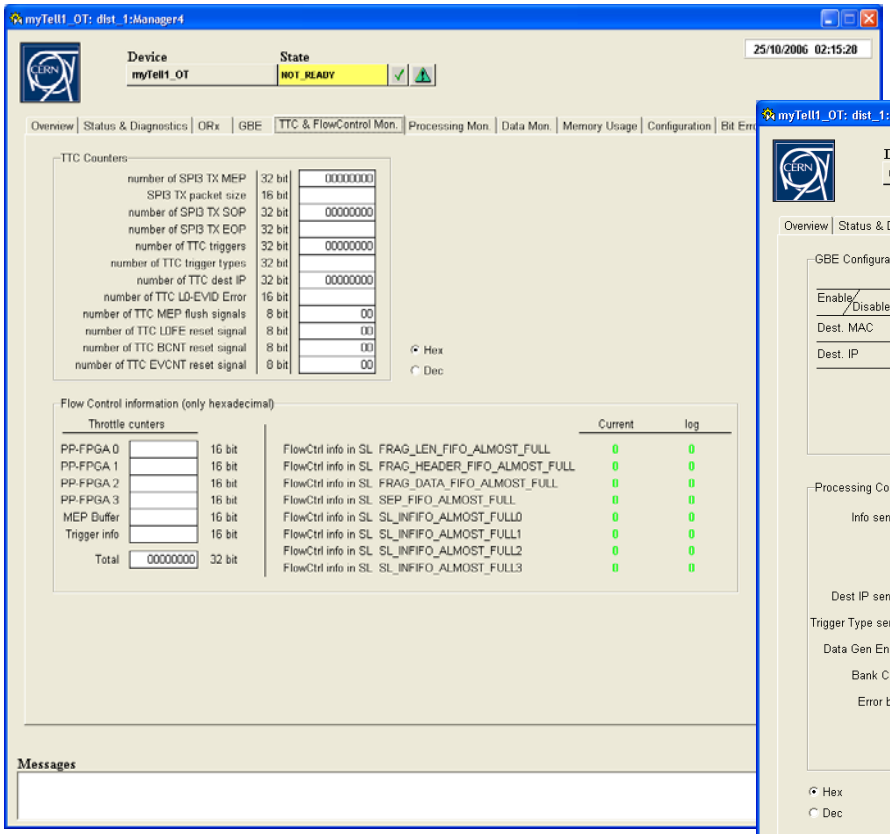
Tell1 Device Unit

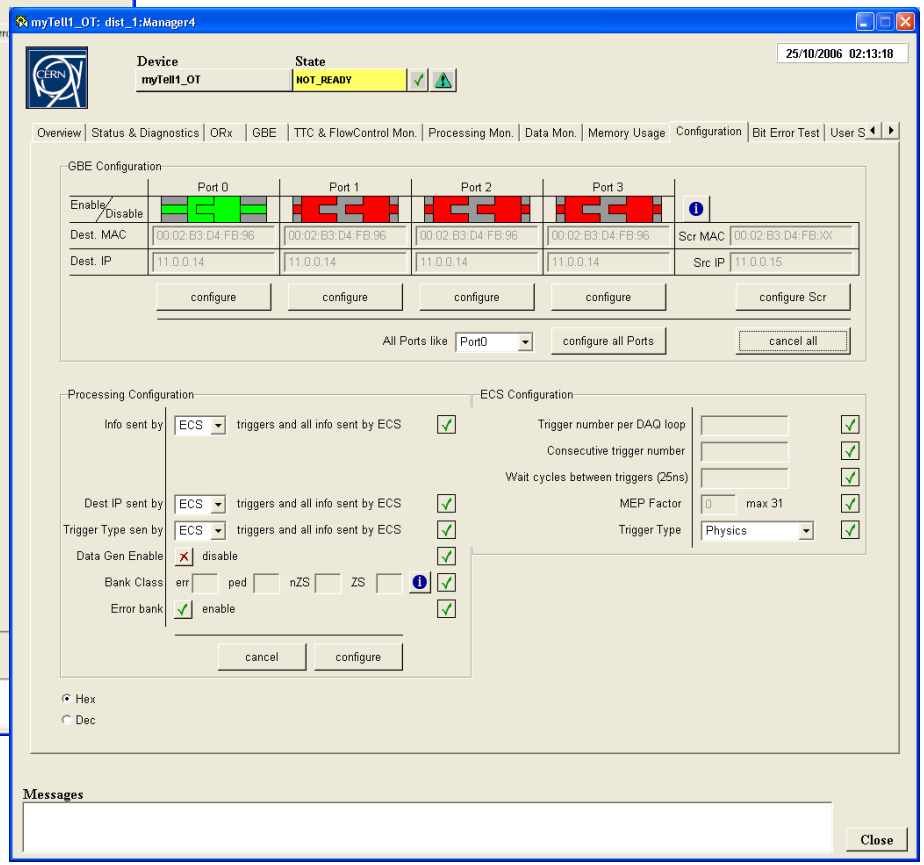


- registers (=data points) can be connected to callback functions which allow to evaluate the functionality of subcomponents (inside Ctrl-script)
- error recovery or alerts can be implemented

Tell1 Device Unit

- many panels to monitor registers – grouped in sub panels
- panels for interaction and reconfiguration of registers





Conclusions

- Presented the basic concepts for implementing the LHCb readout boards into the Experiment Control System with a focus on the various layers and frameworks used
- The system is currently under test for commissioning of the sub-detectors (refinements still ongoing)
- A generic design was of major importance to allow for the usage by different subdetectors and board types
- Further wizards to facilitate board implementation by non experts