Slow Control for Belle / Belle II

Mikihiko Nakao (KEK-IPNS)

mikihiko.nakao@kek.jp

September 30, 2010

5th DEPFET workshop, Valencia (from KEK)

Is everything other than clock / trigger / event timing related control is called "slow control"?

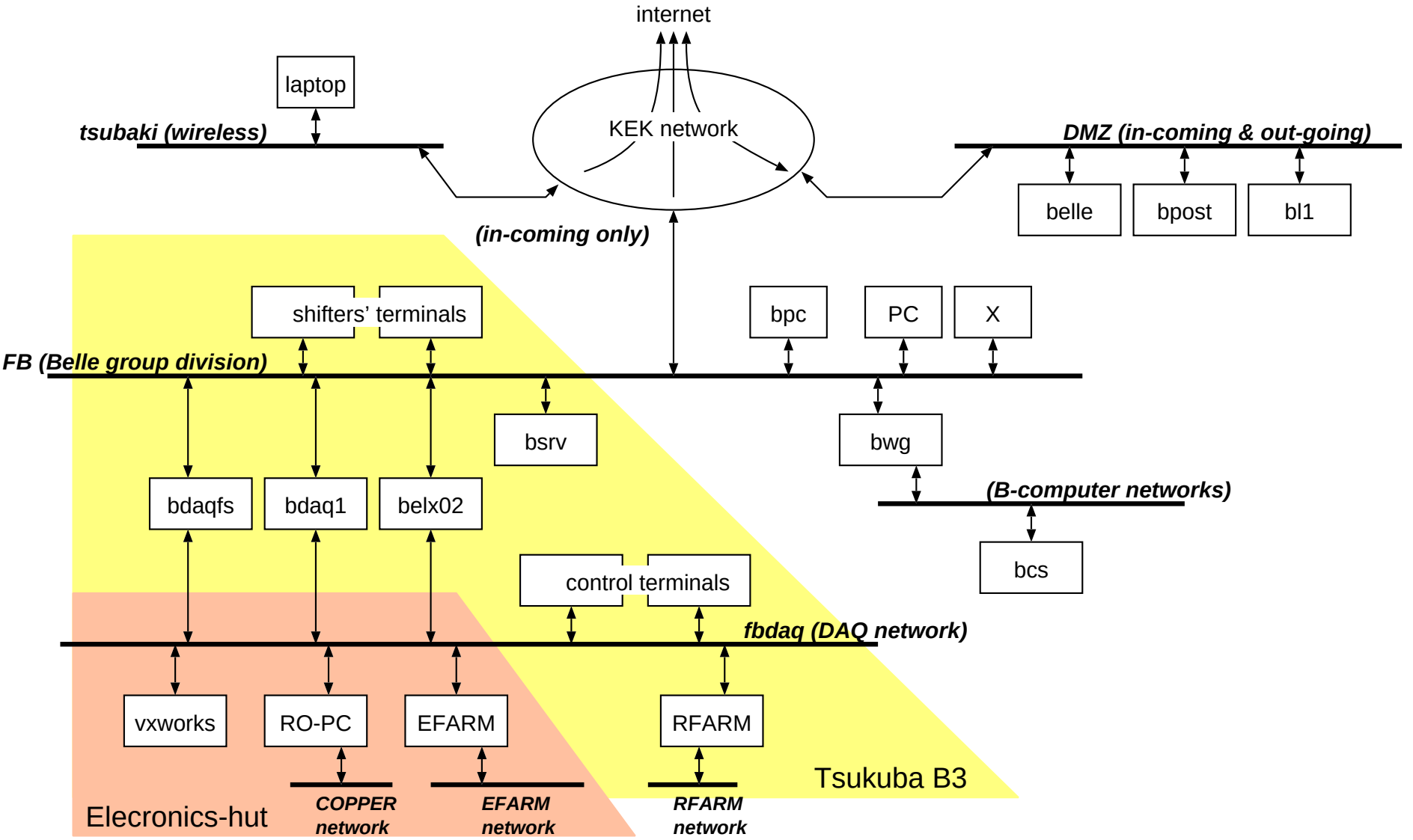
More on run control for which I was working will be covered.

Belle Slow Control

- Run control — start / stop runs, error handling, monitoring the data stream, including / excluding subsystems
- Implemented as "NSM" (network shared memory)
- HV control — based on "Konoe", developed by Kyoto
- Environment monitor — HP Hardware, read out by GPIB
- Temparature monitor — based on LabVIEW (Windows)
- Alarm system — temparature, humidity, voltages, etc
- Interface to KEKB — data sharing, various interlocks
- Online database (logger) — panther based
- GUI — Run control is based on Gtk+, HV control within Konoe (Tcl/Tk)
- Infrastructure — DAQ network and other sub-network segments, also many hardwired lines

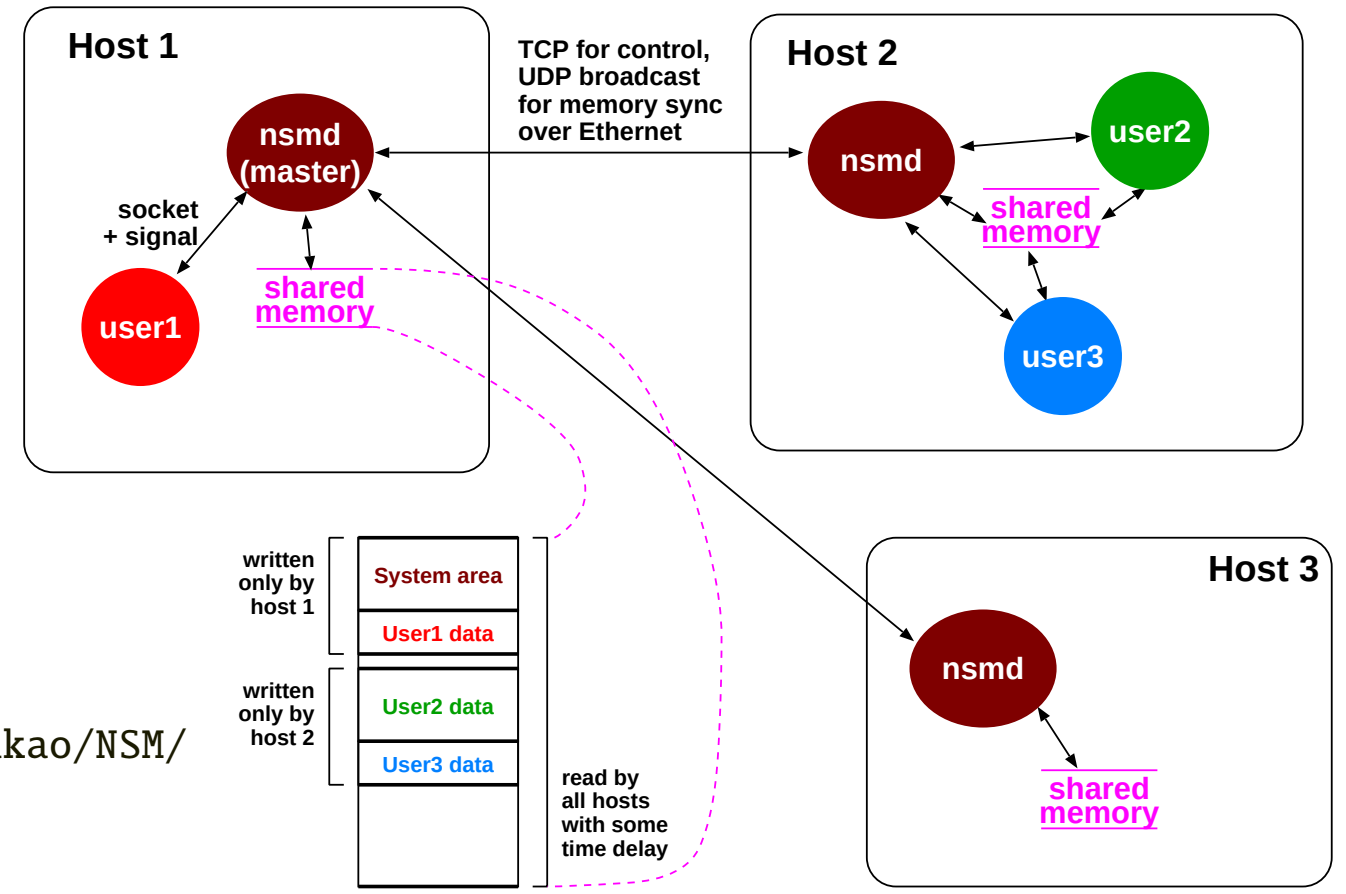
DAQ and other networks in KEK

Services: file server (NFS), DNS, NTP, network booting (rarp, bootparam, tftp, bootp, limited mailing, no httpd



Network Shared Memory

- Used in run control and sharing many control data
- Linux, Solaris and VxWorks (porting to Unix-like OS is easy)
- No security (and attack) protection, otherwise error free (but breaking down by a DoS attack is possible)
- Should be easy to use

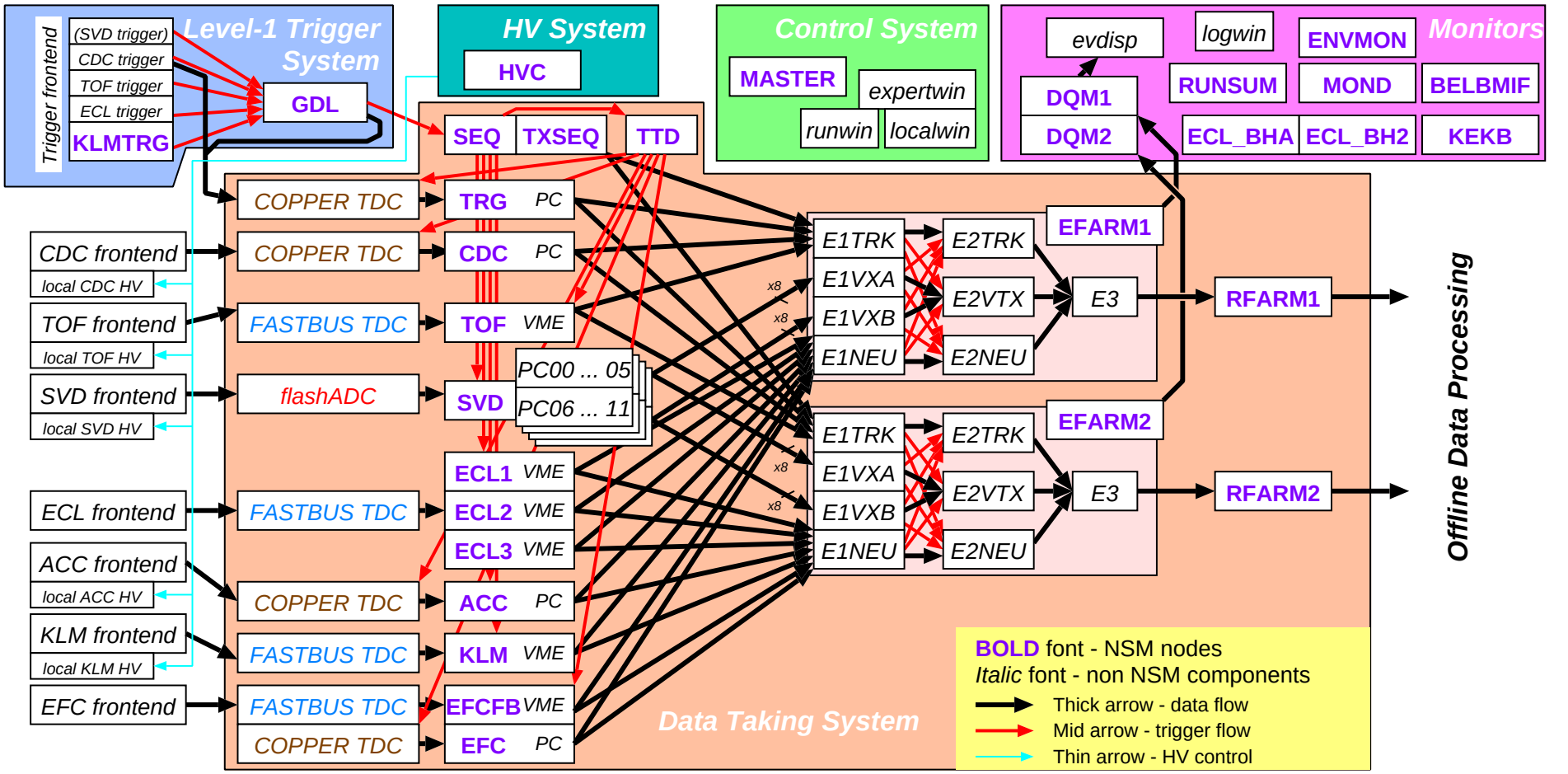


Reference:

`http://belle.kek.jp/~nakao/NSM/`

Belle DAQ under NSM

- All data-taking systems are under NSM
- All the monitoring data are distributed over NSM



Belle Operation and Run States

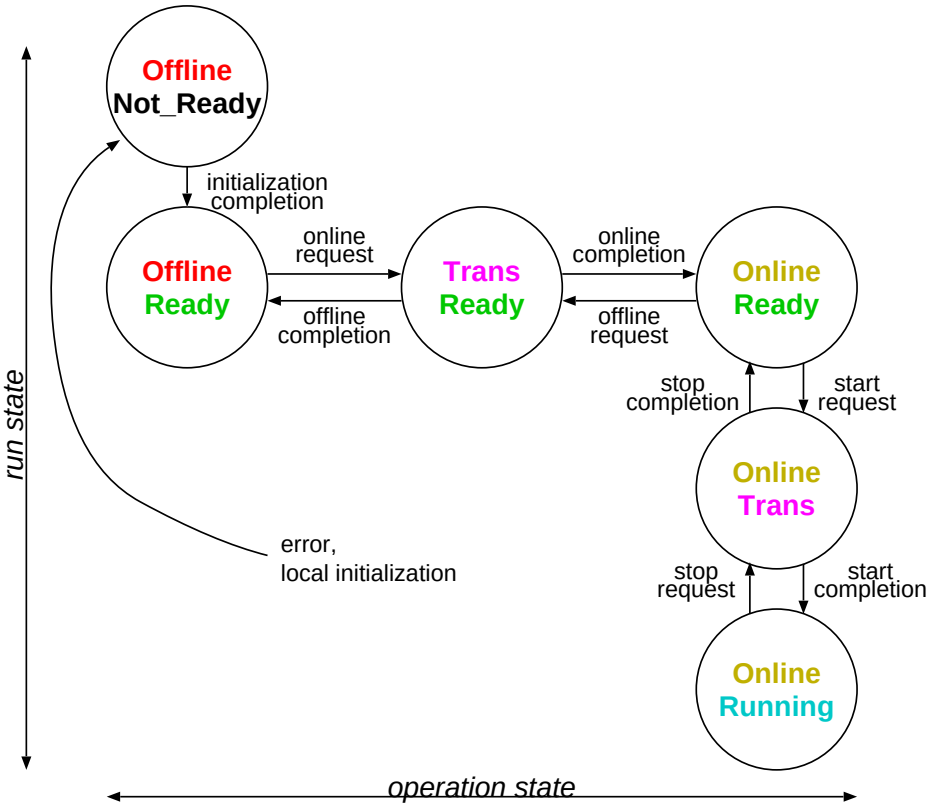
- ONLINE request

  Make sure everything is ready to start, e.g., start a receiver process to accept connection from upstream

- START request

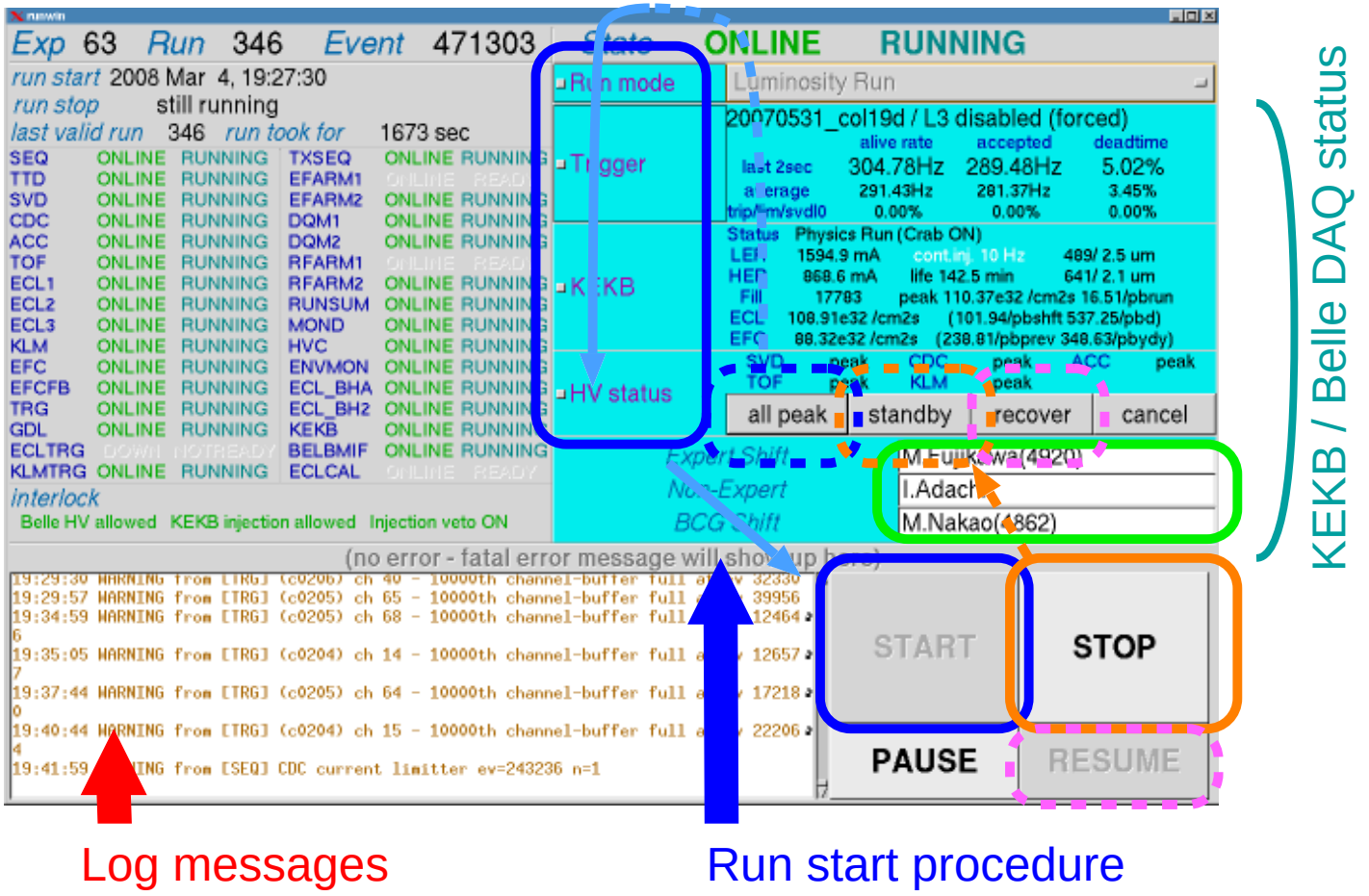
  Make the system ready to receive triggers, e.g., start the relevant process, make a connection to the receiver process

- *Doing more at START is easier and reliable, but has made the Belle startup procedure slower*



Belle Master Run Control

- Run / error handling sequence is “almost” a state machine hardcoded in a C++ program
- Everything in one panel, all info availabe via NSM



Detector Setup

Mostly localized within the subdetector

- Threshold and other parameter settings
- Not so many FPGA downloading (and firmware versions)
- Keeping track of the changes within the group
- Implementing the changes into the offline database
  (and sometimes hardcoded in the offline reconstruction program)
- Many things just happens at
  - Power on sequence
  - “ONLINE” request by the Master Run Control
  - “START” request by the Master Run Control
    (sometimes taking annoying time)
  - In a few cases, additional shell script for setup
    (e.g. ECL threshold setting through FASTBUS)

Interface to/from KEKB

Both fast and slow communication between KEKB and Belle were quite important for higher luminosity and safe Belle operation (and will be relevant for DEPFET at Belle II)

- From KEKB, fast to slow
  - RF clock (509 MHz), revolution signal (100 kHz)
  - Injection kicker signal
  - Permission to start the run (beam ready)
  - KEKB status (current, injection, bunch, etc), EPICS to NSM
- From Belle to KEKB, fast to slow
  - Injection inhibit interlock, beam abort when high background
  - Oscilloscope video for background / trigger signals
  - Luminosity (ECL) and other Belle info, NSM to EPICS
  - Vertex position measured by RFARM (full reconstruction)

Lessons for Belle II

- Run control and HV control were on different framework, could have been unified
- Lot of information was available to monitor the current status, but tracing back the past events was not easy
  - Mostly because of missing “tools”,
    and not because of missing information
  - Database and user interface should be designed these in mind

Detector readout setup at Belle II

- Run control
  - Faster run start — no firmware or parameter downloading at “START” (need to change the word) but there will be something for slow (O(min)) setup
- Signals for frontend setup
  - Frontend FPGA programming through JTAG

    Distribution and collection of JTAG signals using FTSW, actual implementation not well thought out
  - Frontend parameter setting through optical link

    Need a path of DAQ file server → readout PC → COPPER PrPMC → FINESSE → frontend
  - Monitoring through timing link for redundancy and “fast” slow control

NSM at Belle II?

- NSM was reliable at Belle, why not using it at Belle II
  - I think NSM is easy to use
  - Other packages (e.g. EPICS) would require another learning process. KEKB had lots of EPICS troubles at the startup
- Potential (solvable) problems
  - Extention over subnet — already done in SVD, EFARM, RFARM, COPPER systems
  - No standard interface to GUI / histogramming / database / web / scripting languages
  - Cannot handle too large data on shared memory (e.g., putting histograms on the shared memory is not supported)

Summary

- We had many lessons from 10-year Belle operation, from when more people are active, to when very few people interested in stable operation
- If we follow Belle tradition, we can construct a minimal slow control system as was done in Belle (and was quite surprisingly working), with many improvements
- A structured “slow control group” would be also possible, but the reality will not be much changed unless human resources are provided