SuperB Event Data Chain

Architecture Proposal

S. Luitz, SuperB Collaboration Meeting

LNF, March 2012

Event Data Chain Architecture Proposal

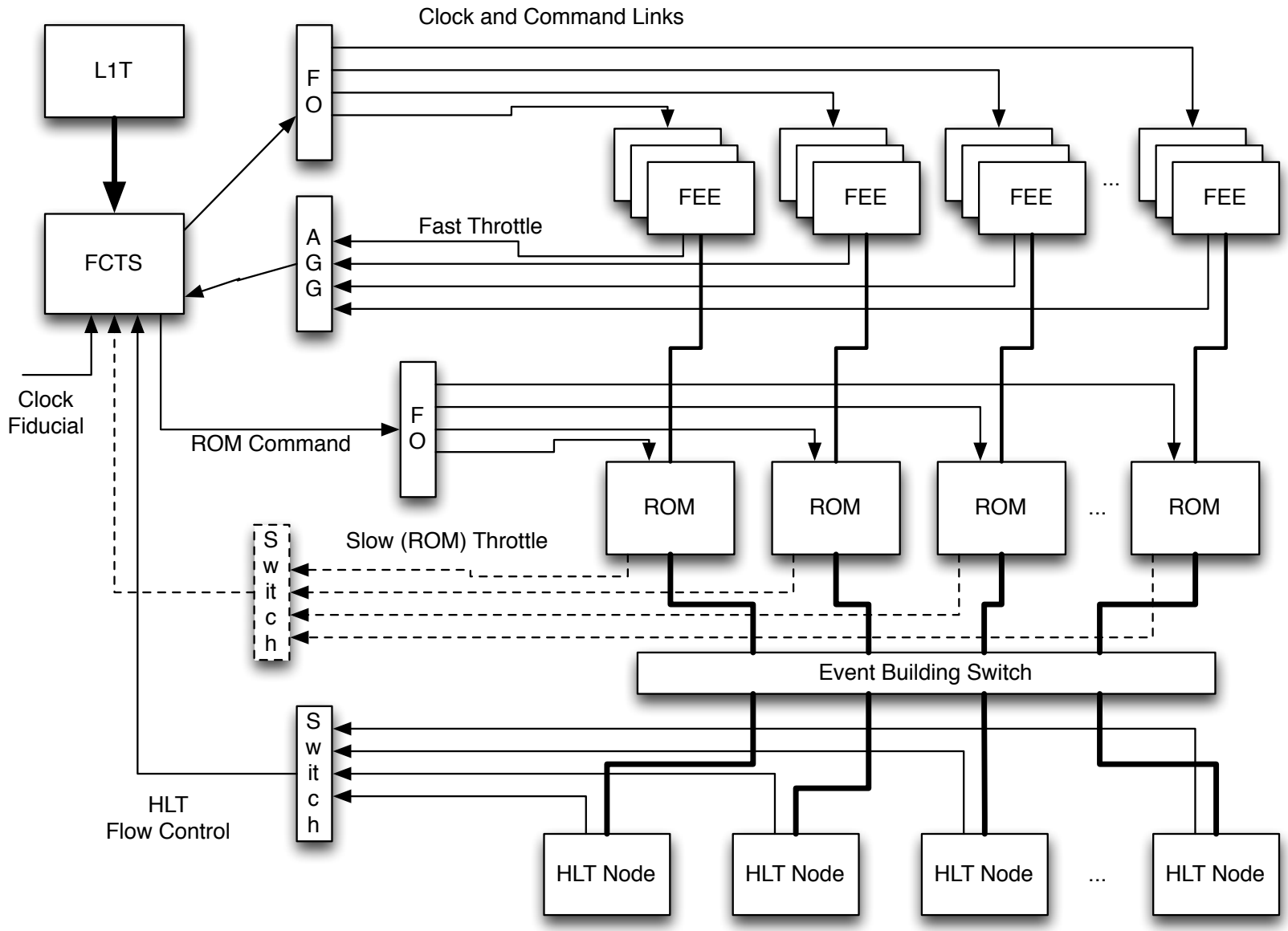
- Last week Daniel and I spent some time thinking about the event data chain implementation
- Assumptions
  - 150kHz L1 Accept Rate
  - 100-200 kByte event size

Guiding Principles

- Keep the system simple and flexible
- Building blocks should be replaceable or upgradeable without having to change the rest of the system
- End-to-end Data Flow architecture
  - No "artificial" split into DAQ and "Online"
    - Trigger and Event Data Path (L1T -> disk)
    - Support systems (ECS, RC, DC, …)
    - Hardware / Software

How it works

- L1 trigger sends decisions to FCTS
  - Fixed latency
- FCTS broadcasts
  - Readout commands to FEEs (fixed latency)
  - Copy of readout command + additional event information to the ROMs (fixed latency not required)
    - Including HLT destination node
- FEEs send corresponding fragment data to ROMs
- ROMs combine event fragments and associate with additional event information
  - No out-of order processing (fully pipelined)
- ROMs send event data to destination HLT node determined by FCTS
- HLT processes events, writes output to per-node stream (disk)
- 3 flow control mechanisms
  - Fast throttle FEE -> FCTS (return path of command link)
  - Slow throttle ROM -> FCTS (ethernet) (might be optional)
  - HLT flow control HLT -> FCTS (ethernet) (request, sliding window protocol)



SuperB

SLAC NATIONAL ACCELERATOR LABORATORY

2 Types of Command Protocols

- Separate command links from FCTS to FEE and ROMs
- FEE command links
  - Transmit commands *and clock?*
  - Fixed latency
  - Radiation-hard
  - Minimize command length – simple commands
  - N-bit "tag" for error detection
- ROM command links
  - Transmit "ROM commands" (no clock)
  - No radiation requirements
  - Variable latency – can even derandomize
    - Copy of FEE command (allows for consistency check)
    - Unique event identifier (timestamp)
    - Full trigger word
    - Destination HLT node "address"
- Separating the links also keeps complicated decisions (e.g. HLT node assignments) out of the fixed-latency path
- Could use the same link technology

Error Detection & Recovery

- We still need to think through the various possible failure modes and how to deal with them / recover, e.g.:
  - Loss of clock / loss of sync
  - Loss of L1 Accept command to FEE
  - Bit error(s) in L1 Accept command
  - …
- Details of response depend on implementation and failure modes of links, clock distribution, etc.
- Questions
  - Where / how do we detect errors?
  - Do we need to stop the complete DAQ to resync?
  - What reliability of detection do we need (e.g # of tag bits)
  - …

FCTS Input from Accelerator

- Similar to BaBar
- RF clock, divide by 8 to obtain the experiment master clock (~59.5 MHz)
- Revolution "fiducial"
  - so that we can synchronize our time counter to the revolution phase
- HER and LER trickle injection shot signals
  - Would allow L1 trigger inhibit around injection bunches
  - Need to record time (including phase) of the last HER/LER injection shot with every event (so that HLT and offline filters can be applied to suppress high-background bunches around injection shots)
    - Through FCTS? Through trigger / trigger readout?

Event Time Stamp

- Event identifier unique over lifetime of experiment
- BaBar: 56 bit counting at ~60MHz
  - wraps every ~20 years
  - clock derived from RF (divided by 8)
  - fixed relationship to revolution fiducial
  - coarse time initialized by wall clock time (e.g. from an NTP source)
    - Initialization: Stop counter, load initial time, arm counter, start at next revolusion
- Propose to do the same …

Event Builder

- Basic design choices
  - Push event builder
    - A la BaBar or LHCb
    - ROMs push event fragments into the HLT
    - Load balancing controlled by FCTS
      - Note: BaBar didn't load balance
  - Request-based event builder
    - HLT nodes pull events from ROMs
  - Connection-less vs. connection-based
- Proposal: Connectionless & Push
  - Allows maximum flexibility for ROM implementation

SuperB SLAC NATIONAL ACCELERATOR LABORATORY

HLT Load Balancing

- Sequencing of HLT node assignments done by FCTS (in firmware)
- Round-robin (?) over available nodes
- Every HLT node periodically announces its number of free input queue slots to the FCTS (over ethernet)
  - i.e. how many more events is it willing to take
  - FCTS maintains a per node window / counter
    - Decrements with every event sent to this node
    - Stop sending events to this node when reaching 0
- Think of this as a sliding window or generic event request implementation
- Provides dead HLT-node removal and load balancing
- Should be very simple and robust
  - Light-weight version of what LHCb is doing

To MEP or not to MEP?

- Do we need multi-event packets?
- Naïve calculation:
  - 200 kByte events from 60 ROMs
- → 3.3 kByte per ROM average
  - > 2 Ethernet standard MTUs
  - Event with jumbo frames packing limited to 2 events per frame
- Probably not necessary for Ethernet
  - Build capability into FCTS (& ROM) in case we use a different transport?

SuperB SLAC NATIONAL ACCELERATOR LABORATORY