CMS Global Calorimeter Trigger

University of Bristol

S. Galagadera, A. Shah, S. Madani
Rutherford Laboratory

→ CMS Level-1 Trigger
→ GCT Design
→ Module implementation
→ Firmware
→ Lessons Learnt
CMS Level-1 Trigger

→ Level-1 uses muon & calo data only
  - Tracking data too large / complex
  - Local pattern recognition is possible

→ Strategy
  - Identify highest-pt physics objects (e/γ, μ, jets, τ jets, energy flow)
  - Use $p_t$ and topological cuts at final stage

→ Fully pipelined digital electronic system
  - Impossible to make decision in 25ns
  - All data stored on detector during fixed L1 latency, read out upon L1A
  - Memory constraints give max latency 3.2μs

→ Output of Level-1
  - Single bit: accept / reject
  - Triggers may be ‘throttled’ for technical reasons – but otherwise, ~ zero deadtime
  - On L1A, event data proceed to HLT
Calo Trigger Algorithms

e/γ (hit tower + max neighbour):
- 2-tower Et; hit tower passes H/E cut
- Hit tower: 2x5 strip with >90% Et in 5x5 (FG)

Isolated e/γ added criteria:
- All 9 towers pass FG and H/E
- One 'corner' group of EM towers < Thr

Jet or τ:
- ΣEt of 12x12 trig tower sliding window
- Central 4x4 Et > each neighbour
τ (isolated narrow deposit) added criteria:
- all 9 regions have 'τ pattern' deposit

Total / missing Et uses 4x4 granularity
Total "Ht" uses found jets only


Trigger Architecture

→ GCT functions
  - Sort $e/\gamma$, iso. $e/\gamma$
  - Find, categorize, sort jets
  - Global energy sums
  - Lumi monitoring
  - DAQ interface

→ System interfaces
  - Input: 3000b / BX
  - To GT: 300b / BX
  - DAQ: 200MB/s
  - TTC, TTS, VME

→ Off-detector system
GCT Design Issues

→ Small, complex, one-off system
  - Single rack, no ‘mass production’ issues
  - Many board types -> cost, complexity, testability, spares…
  - Strategy: use a **generic module** for all processor functions
    - Enabled by very high performance FPGAs (Virtex-2 chosen)

→ Very large data density
  - Compact, highly interconnected system
  - Physically impossible to route input signals into single crate
  - Strategy: fast copper **serial links** for all IO
  - Use similar technology for module interconnect (**no backplane**)  

→ Reliability is crucial
  - GCT is a single point of failure for CMS
  - Strategy: multiple redundant control paths to all boards
  - Strategy: **monitoring & self test** at chip, board, system levels

→ Must maintain the maximum possible **flexibility and programmability**
Trigger Processor Module

→ Basic building block
→ 9U x 400mm VME
→ 34.5 Gbps input
  ■ 24 x DS92LV16
→ 8.5 Gbps output
  ■ 6 x DS92LV16
→ 60 Gbps sharing
  ■ 6 x VSC7226
→ Processing
  ■ 4 x XC2V3000
  ■ Algo logic at 160MHZ
  ■ 160/320 Mbps SSTL
  ■ control / DAQ bus (wishbone derivative)
TPM Implementation (v1)

- Anti-flex bars
- Input Rx / FPGA
- Debug connectors
- DC-DC Conversion
- Proc FPGAs
- Control/CF/VME
- Data sharing
Module Interconnect

- 3.2Gbps diff. LVPECL, duplex
  - Vitesse VSC7226
- Up to 24 links per slot + clock
- Foamed PTFE cable
  - No crosstalk, good impedance ctrl
  - Fully reconfigurable (no tracks)
  - But - halogens (airtight enclosure)
MI Implementation

200ps/div (3.2Gbps); Measured BER < 10^{-13}
Input Module

→ Function
- Retime and synchronize 80MHz parallel ECL data
- Bit-by-bit deskew allows use of low cost cables
- Serialize + send to TPM

→ Implementation
- Simple, low-cost module
- Wide common mode range receivers
- Single Virtex-2 FPGA oversamples and formats data

→ Clock / control
- Provided by TPM
- Data capture / playback for debugging of system

QuickTime™ and a TIFF (Uncompressed) decompressor are needed to see this picture.
IM Implementation

Serializers
Integration Tests
Communications Module

→ Function

- Distribute master clock throughout system
- Accommodate CMS-specific control / DAQ interfaces

→ Timing

- Single TTCrq daughterboard for GCT system (fibre input)
- 40MHz clock distributed via point-to-point rear links
- TTC control signals distributed to ‘DAQ TPM’ and thence to system

→ DAQ

- Slink-64 daughtercard for final DAQ system
- VME64 interface for ‘private DAQ’ during commissioning

→ Other interfaces

- Link to throttling system in case of buffer overflow or error
- Controls multidrop JTAG backplane
- General-purpose control / monitoring ports
CM Implementation

- DAQ data from TPM
- Control FPGA
- TTCrq
- Clock / JTAG output
- SLink-64
Control / Monitoring

→ Wishbone-based control bus
  - Allows open source cores
  - VME64 <-> Wishbone core developed
  - Bus extender serialises W/B protocol, over shared control / DAQ ring bus
  - DAQ transactions look like wishbone traffic
  - Multiple bus masters allowed, including onboard soft CPU

→ JTAG
  - Used for FPGA config, system test
  - Also useful for debugging via Xilinx Chipscope

→ Monitoring / DAQ
  - Every FPGA has large capture / playback buffers
  - Flexible online capture to CMS DAQ
Firmware Issues

→ Issues
  - GCT has a lot of firmware (200+ FPGAs, all unique)
  - Change / config control is important
  - Commercial CAD solutions do not support this use case well

→ Strategy
  - Use techniques and tools borrowed from large software projects
  - Encapsulation, modularisation, interface definition

→ Fbuild
  - Build system allows GCT system to be automatically ‘made’ from components
  - Includes: building on cluster; ‘simulation’ build, core customisation
  - Can share configuration control with GCT software, simulation tools
Lessons Learnt (so far!)

→ Design
   - The ‘generic module’ approach is useful and cost-effective
     • Trade-off in board versus system complexity seems to have paid off
   - Very high density communication over serial links can be made to work
     • Power, clock, signal path quality are all important

→ Implementation
   - Getting an 18-layer 400mm board fabricated reliably is not so easy
     • Rework becomes more and more difficult
   - Dedicated debugging connectors / nets are now essential
   - Integration and board checkout take longer than you think
     • Almost everything is needed to get anything to work

→ Software / firmware
   - GCT boards are almost a ‘software system’ now
   - Automate and integrate firmware / software build from the start

→ JTAG
   - Does the job, but actually becoming difficult to implement reliably!
Summary

→ GCT
  ■ Small but complex system, challenging to implement at low cost
  ■ Generic module / configurable backplane approach used

→ Project status
  ■ All modules implemented in final or prototype versions
  ■ System integration and test now proceeding
  ■ No critical show-stoppers so far

→ Lessons from implementation
  ■ Process of board design -> fab -> test is now very long and challenging
  ■ Built in capacity for debug / self-test / monitor is essential
  ■ Automated approach needed to firmware build and config control

→ GCT ready for action in 2006!
  ■ Integration with CMS L1 trigger starting in 2005
Driven by LHC physics conditions
- Decays of rare and heavy particles against large “soft” QCD b/g
- Many decays involve intermediate $W / Z$; $H -> \gamma\gamma$ also important
- Identify high-$p_t$ leptons* and photons (*including $\tau$)
- Low $p_t$ thresholds motivated by efficiency for $W / Z /$ light Higgs

Trigger combinations
- >20GeV limit on single-lepton thresholds due to quark decay + $\pi^0$ b/g
- Most interesting states decay to two or more trigger objects – can use lower thresholds for objects in combination
- Find trigger objects locally, combine and cut only at last stage

Large uncertainties in background (and perhaps signal)
- Flexibility and control of rate are both vital
- All trigger thresholds and conditions must be programmable
- Trigger architecture is fixed, but this is a function of detector geometry

Must have high and well-understood efficiency
- Need to include overlapping and minbias triggers to measure $\varepsilon$
Backup: GCT Location

120m below PA5 (Cessy)

CMS experimental cavern (construction phase)

CMS counting room (trigger + DAQ)

2μs fibre path