# *Feasibility Studies of a Level-1 Tracking Trigger for ATLAS*

Matt Warren
Ilija Bizjak, Richard Brenner (Uppsala), Gordon Crone,
Nikos Konstantinidis, Mark Sutton (Sheffield)

TWEPP, 23 September 2009

# *Introduction*

- Tracking trigger for ATLAS upgrade is relatively new idea
  - Not included in current upgrade design to date
  - Physics case needs work
- 2 options are under investigation

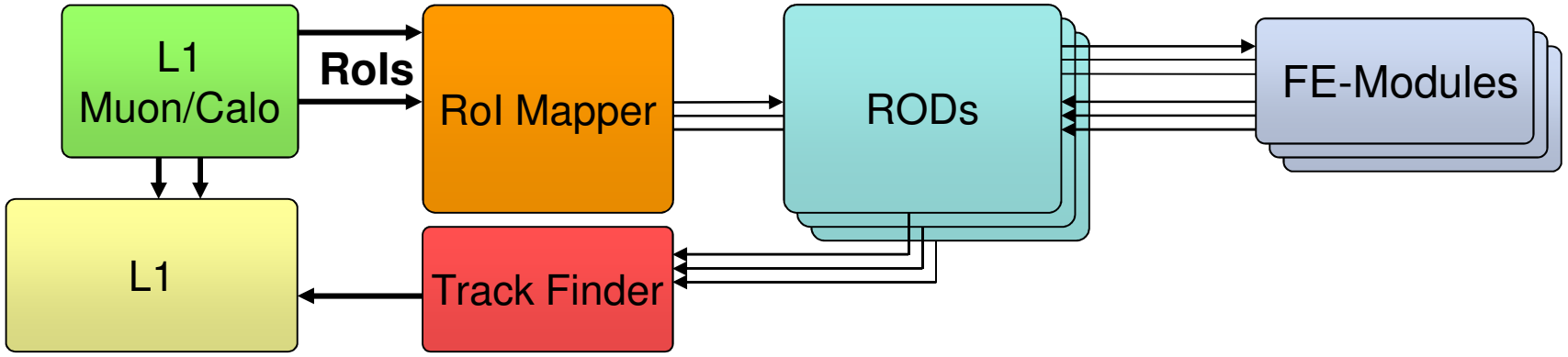
(presuming 40MHz readout of whole detector too difficult)

1) Auto/local event selection with special layers
   - On detector logic selects good events
   - Can readout every BC (if interesting)
   - Modules have both sides connected
   - Ideally layers are linked on-detector(!)
   - Early studies show high readout rates required
   - Still developing …

2) Regional Readout
   - A portion of the detector is readout prior to an L1A being issued
   - Fast/low latency – contributes to L1 decision
   - The focus of this talk …

# Regional Readout

- Level-1 Muon/Calo system identifies a “region of interest” - RoI
- RoI mapped to set of front-end modules
- Regional Readout Request (R3) signal is sent to these modules
  - Data is copied from mid-pipeline, jumps to head of any queues
- Modules transfer regional data over a prioritised channel to Readout Driver (ROD) off-detector
- The regional data in intercepted and forward to the track finder.
- Track finder contributes to L1 decision



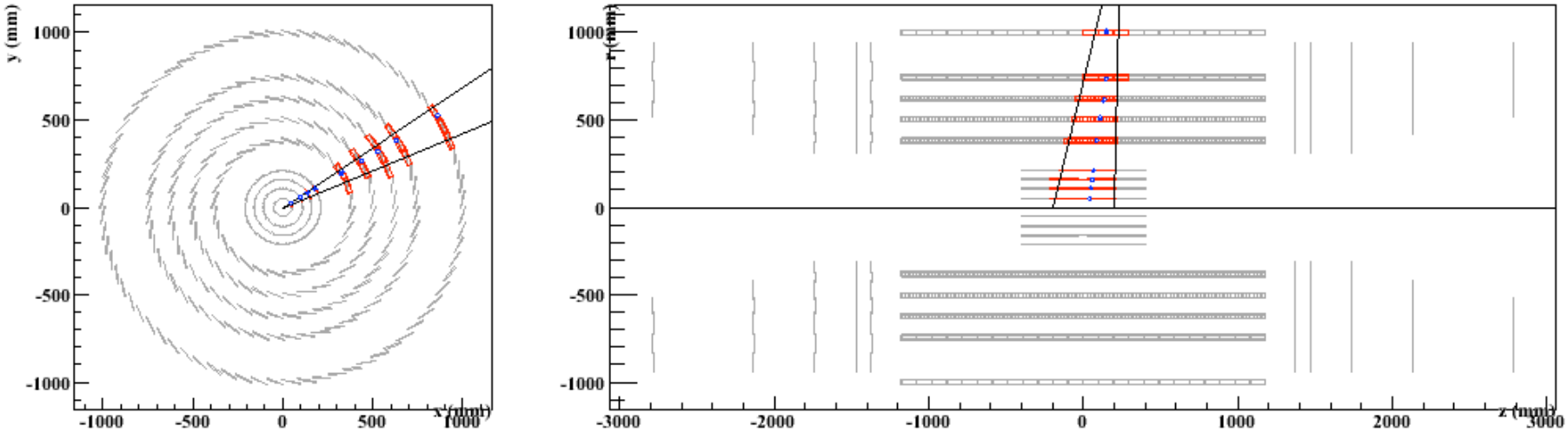
# Rates and Expectations

Presumptions:

- L1 rate stays at 100kHz
- R3 rate at 400-500kHz

Simulation results indicate:

- An RoI contains ~1% of modules in the detector (tracker)
  - Effect on bandwidth minimised
- ~4 RoIs/BC




RoI: Δφ=0.2, Δη=0.2 at Calo Δz=40cm at beam line

# Challenges

There are many hurdles

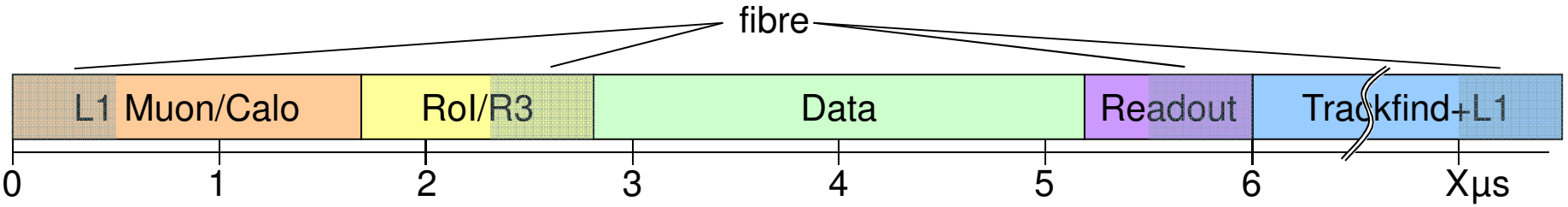
- Latency (overall)
  - FE pipeline length is limited – for Level-1 we need be fast
- Data Volume/Readout Rate/Dead-time
  - More data = longer readout
- Data Transfer (+ Synchronisation)
  - Ideally we want separate links, but compromises are needed
- Regional Readout Request distribution
  - Targeted fast-commands need more infrastructure
- Off-detector readout/track-finder
  - Needs fast/synchro path to L1

# *Latency (Overall)*

- Latency is affected by almost all components – see next slides
- FEICs have finite pipelines, defining the system latency
  - Current ATLAS has 3.2us (128 BC)
  - Upgrade would prefer more (6.4us is common), but at what cost?
  - Track trigger definitely needs more (double round-trip to module)

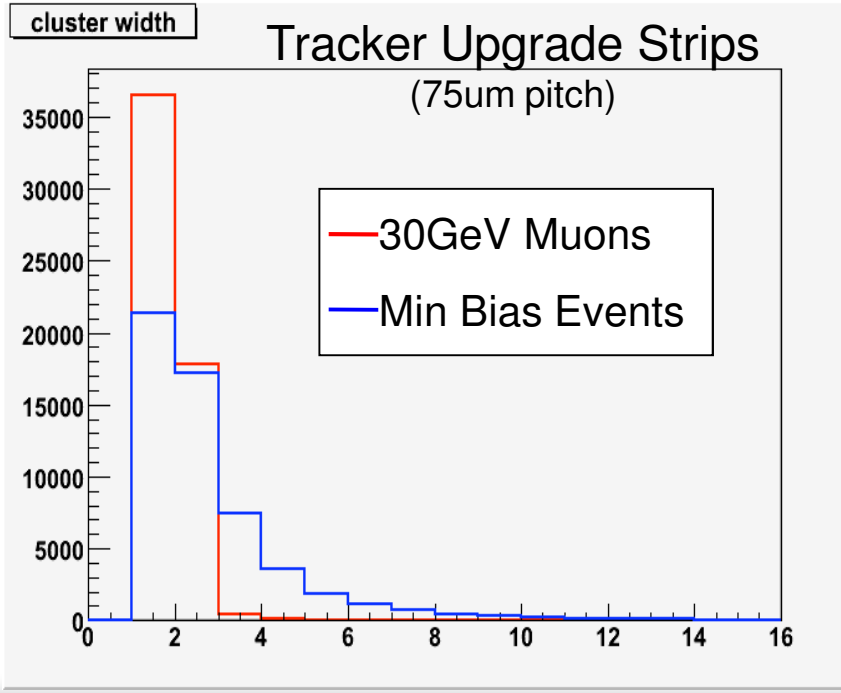
Initial estimates:

| | |
|---|---|
| BC → RoI | 1700ns (incl. 500ns fibre) |
| Decode RoI/Send R3 | 1150ns (incl. 500ns fibre) |
| Data Volume | 2375ns |
| Readout | 825ns (incl. 500ns fibre) |
| Track Finder + L1 | 1500ns+? (incl. 500ns fibre) |
| **Total** | **7550ns+** |



# Data Volume

- Largest contribution to overall latency
- Defines dead-time of a module
  - Low-latency = no queuing of data
- Data reduction prior to sending from module
  - Only using <3 strip clusters (high-pT tracks)
  - Sending central cluster only
- Cap number of events/chip/module
  - 3/chip; 15/column (1280 strips)
  - <5% of 'good' hits lost
  - Allows efficient data packaging
    - small hit counter + hits
- ~120 bits/event/column
  - Will reduce with bigger chips

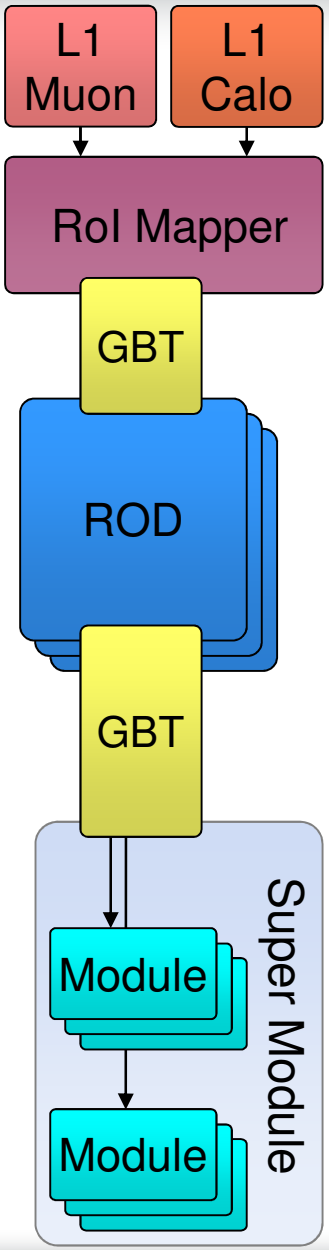


# *Data Transfer (+ Synchronisation)*

- Ideally regional data has a dedicated path off-detector
  - Lowest and fixed latency
  - No congestion
- But resources are limited
  - Bandwidth
  - Signal routing on-detector (copper)
  - Signalling off detector (fibre) – power
- Share readout "channel" with 'normal' data
  - Bad for Latency+Syncho: Must wait for in-progress event to finish
  - Reduce this effect by using small packets
    - e.g. 10b = <start><event/regional><8b data>

# Regional Readout Request Distribution

- Synchronous to event-BC (like L1 trigger signal)
  - Acts on pipeline data
- TTC is designed to broadcast fixed latency
  - Need a new mechanism
- Broadcast (or multicast) lists of RoIs to RODs
  - Using GBT in counting room
  - <4k RoI in detector = 12b RoI ID, so 10/GBT frame
- ROD converts to link/bitmap of connected modules
  - Custom LUT on each ROD
- GBT transfers to detector
  - Trigger bit identifies R3, data contains bitmap
- Bitmap divided across zones and serialised
  - Adds dead-time. But minor effect (<200ns)
  - Readout dead-time means double R3's not useful



# *Off-detector Readout / Track Finder*

- ROD routes regional data packets to track-finder
  - Detected as serial stream arrives – low latency
  - Data volume is low, many input links can share single output link
    - Need to investigate/optimise to minimise queuing
    - Data with earlier BCIDs could be prioritised
    - Link ID overhead
- Track-finder is divided geographically across detector
  - Quadrants, barrel/ec/both, with some overlap
  - Track finder processors assigned to individual BCIDs
  - "Bingo" technique – as data arrives it is used
    - Don't need to wait for whole event
  - Re-sync at output (makes Level-1 happy)
    - Any available track info is forwarded to L1 a FIXED time after BC
    - Late arriving data/processing is discarded

# *Summary / Conclusion*

- Regional Readout is new, and only on paper
  - The upgrade design is a moving target
  - Track trigger needs to be integrated into each component
    - ASICs, Hybrid, Modules, Stave, Tapes, SMC, GBT, ROD, TTC …
- No show stoppers … yet!
  - Need input from the real experts
- Fits with current upgrade design, except:
  - We need more latency!
  - But <10µs
- All comments and ideas welcome …

# *The End*

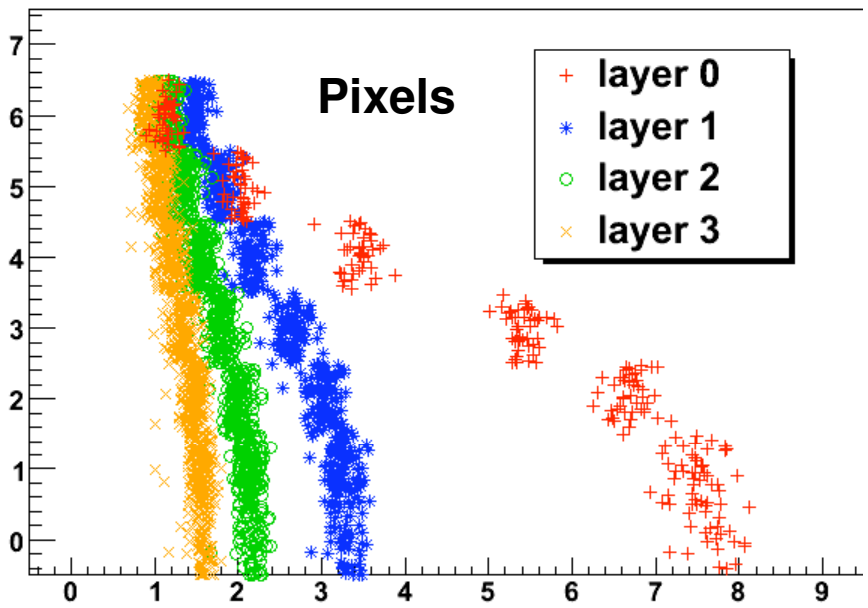
Thank you.

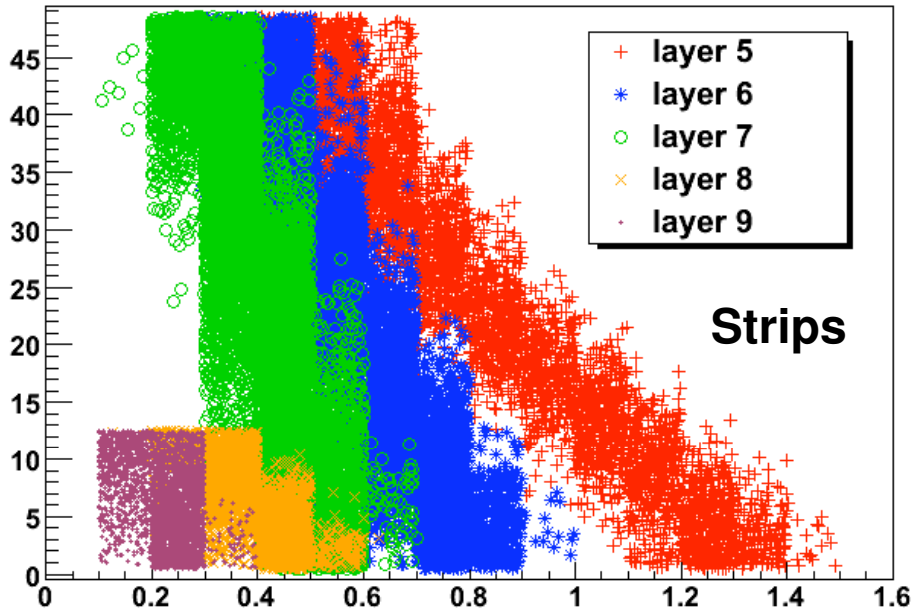
Questions?

Backup slides/more detail …

# How often a module in an RoI?

Fraction of RoIs (in %) containing a module




- Central modules in layers near the beam line are more frequently inside an RoI
- Work ongoing to determine how many layers

# Glossary

**Regional Readout *(RR)*** is a system to readout only a subset of the detector

**Regional Readout Request *(R3)*** is the targeted RR command

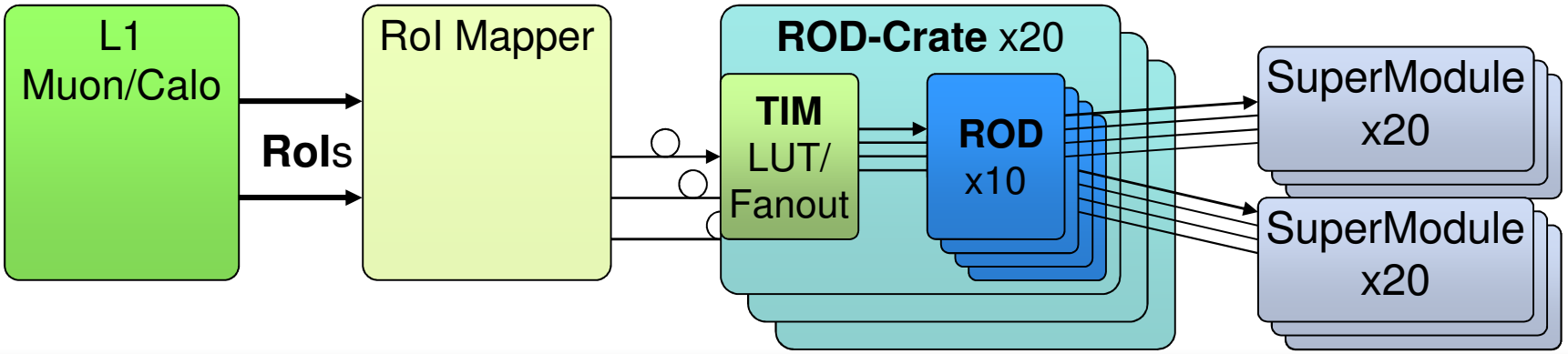
– R3 initiates readout of **Regional Data *(RD)*** from the front-end

**Region of Interest *(RoI)*** info from Level-1 Trigger is used to generate R3s

**RoI Mapper *(RoIM)*** maps L1-RoI to RR RoI and distributes to **ROD-Crates**

Inside a **ROD-Crate**:

- **TTC Interface Module *(TIM)*** – sends targeted R3/RoI to each ROD
- **~10 Readout Drivers *(RODs)*** – LUT of RoI/stave – sends R3s to stave



# *Detector Layout (Upgrade SCT)*

A **Super-Module** (**Stave**) comprises 2 Sides

A **Super-Module Side** (**Stave-side**) comprises

- 12 Modules (= Wafers)
- 1 Super Module Controller (SMC)
- 1 "Tape" – flex-circuit for whole stave inter-connect

**[Note, when referring to a Stave, usually we mean a Stave-side]**

A **Module** comprises

- 1 Wafer (4x25mm strips or 1x100mm strips)
- 2 Hybrids (1 for long-strips)

A **Hybrid** comprises

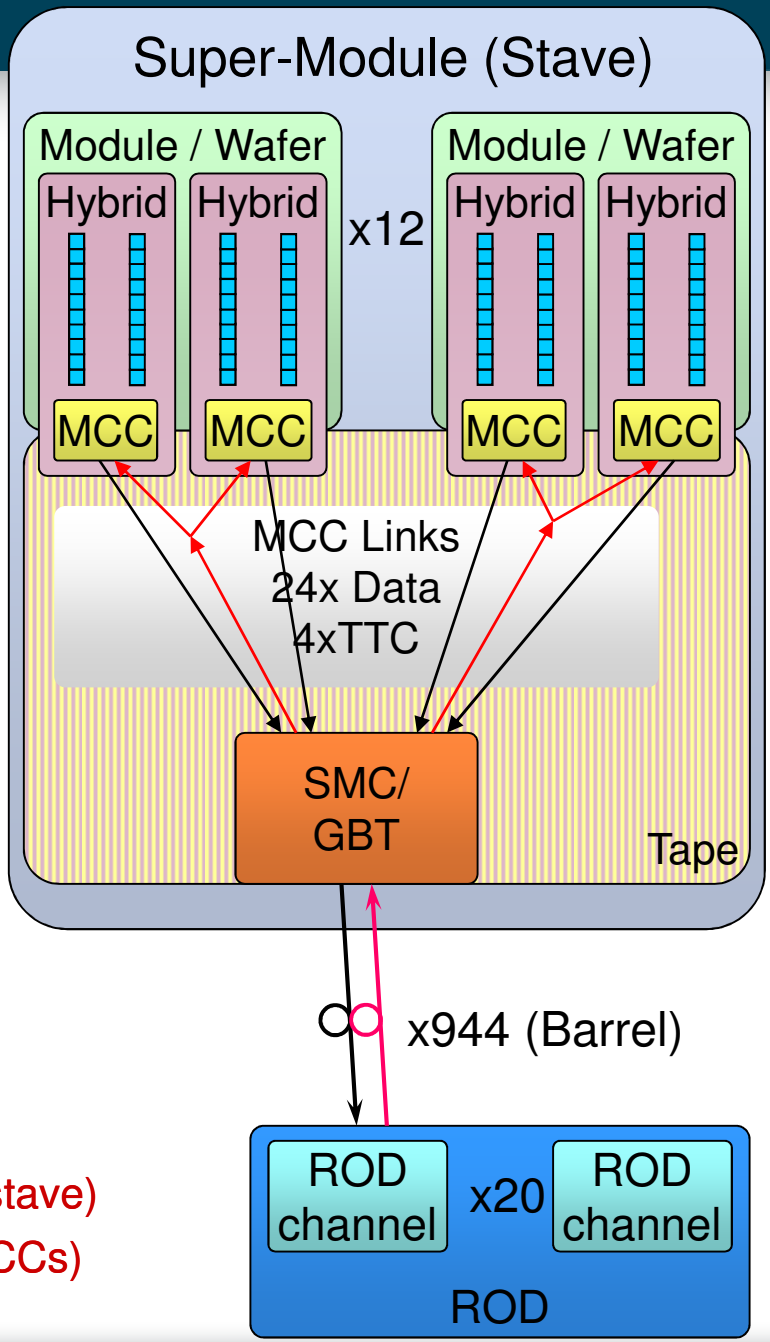
- 2 Columns of 10 FEICs (1 for long strips)
- 1 Module Controller Chip (MCC) driving a point-to-point readout link to the SMC (Cu, 160Mb)

The **SMC** aggregates 24 MCC links (12 for long) into 1 GBT link connected to a ROD off-detector (fibre, 3.84Gbs)

- Channels/ROD unknown, but ~20 fits current rack alloc.

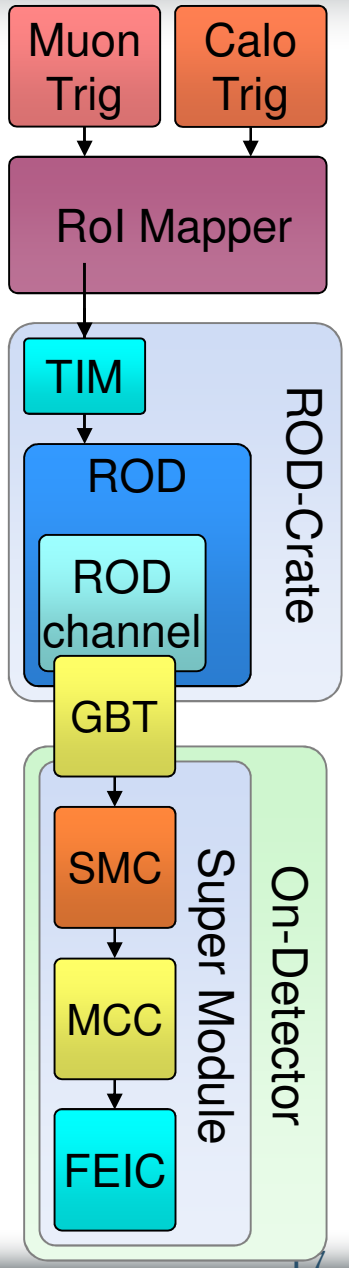
**Control Path:**

- ROD distributes custom R3 map for each ROD-channel (stave)
- SMC decodes and distributes to modules (or individual MCCs)



# Region Readout Request (R3) Flow

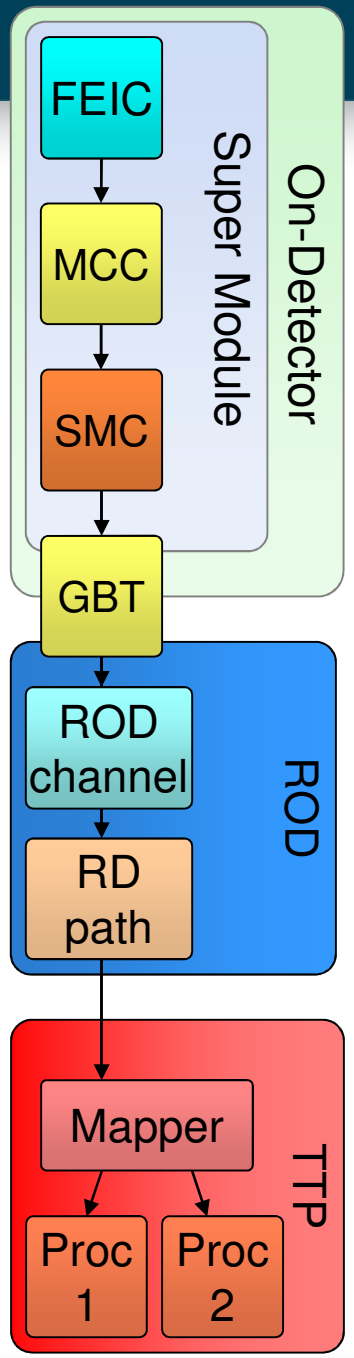
- RoI Mapper
  - L1Muon/Calo RoI Interface – convert and synchronise RoIs
  - Stave/Module/ROD Mapper (LUT)
  - Fanout/Driver to ROD-Crate OR Interface to TTC?
- ROD-Crate
  - LUT/fanout on TIM to RODs
  - LUT/route on ROD to individual Stave-sides
- GBT
  - Use a trigger bit to identify as R3 packet
  - If R3, then data in packet contrains bit-map of MCCs in RoI
  - Takes Stave TTC "Zones" into account
- SMC
  - Interfaces to R3 interface to TTC control lines on tape
  - Provides R3 word format as needed (serialised)
- Tape
  - Transferred on TTC lines
  - Path split into 4 zones – 4 sets of lines (6 MCCs/zone)
- MCC
  - Distributes R3 to FEICs



# Regional Data (RD) Readout Flow

- MCC receives R3
  - Initiates data readout – formulates headers, stores BCID
  - Forwards to FEICs
- FEIC
  - copies data from mid-pipeline to track-trig logic
  - data reduction applied
  - Data priority-queued for sending ASAP
  - MCC forwards down stave to SMC/GBT/ROD as normal data
- ROD
  - Header identifies regional-data packet: ROD directs it to output channel
- Track Trigger Processor (TTP)
  - Data sorted by BCID/RoI and sent to processing units

NOTE: For latency reasons, only 1 R3 can be handled at a time – no queuing



# RoI/R3 Distribution to Stave

- Synchronous distribution desired
  - Reduced complexity in the FEICs
- <5 RoIs per event (whole detector)
  - likely better to distribute RoI-ID to RODs
- RoI Mapper interfaces to L1 Muon/Calo systems
  - Synchronises to single BC/event
  - Converts to common RoI-ID format if needed
  - Sends list of RoI-IDs to each TIM
    - Could make use of TTC infrastructure?
  - Using GBT without error-correction gives us 120 bits/BC
    - Presuming <4k RoI in the detector, we can send 10x12bit RoI-ID/BC
- TIM decodes RoI-ID and sends custom list to each RODs
  - OR broadcasts to RoI-ID to all RODs
- RODs map RoI-ID to custom R3 bitmaps for each link
  - Each ROD is configured with custom list of “connected RoIs” and stave-map
- R3 is sent via GBT to Stave
  - Dedicated GBT trigger bit acts as “Stave-R3”
  - If Stave-R3 set, then data for that packet contains MCC bit-map

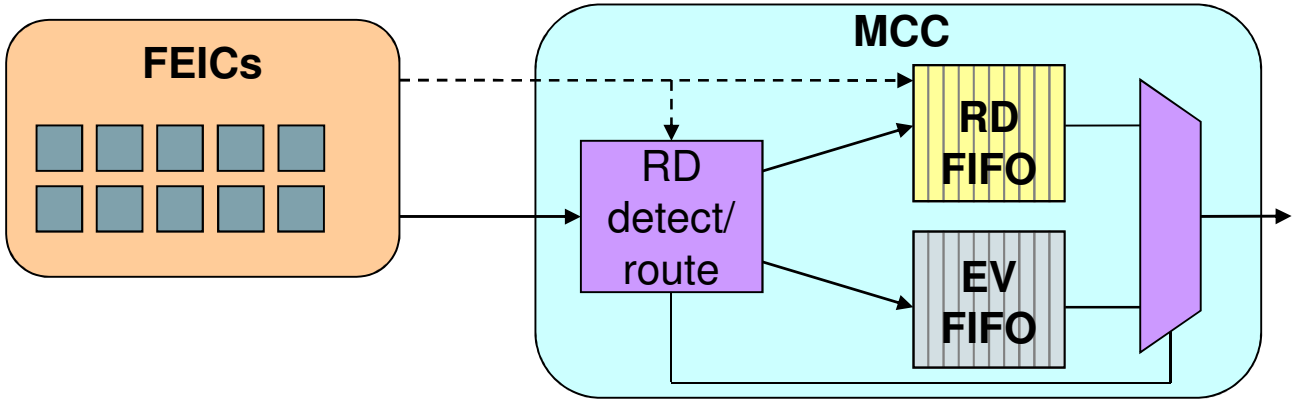
# On-Stave R3 Distribution

- Stave signals are distributed via copper on a flex-circuit
- At the end is the SMC (GBT inside) and opto hardware
  - –SMC very much undefined
  - –GBT functionality is still developing
- TTC signals bussed in 4 zones on the stave (LVDS)
  - –i.e. no point-to-point signalling
  - –Zone = 6 MCC/zone,
  - –R3 is transmitted serially at 40MHz (L1A/R3 share 80Mb line)
    - Need 6b map to ID all MCCs individually + start-bit, so 175s latency for R3 distribution
    - Consider 6 zone TTC, 160Mb distribution?
- No extra capacity for addition signal tracks or bandwidth
  - –Or at least that's what we are told
- MCC receives R3
  - –Initiates readout – headers, queue-jump
  - –Forwards to FEICs
- Data signalling from the MCC modules is point-to-point
  - –1 link/MCC = 24 LVDS pairs @ 160Mb each

# MCC / FEIC

MCC manages regional-readout:

- Generates header
- Distribute R3 to FEICs
- Pauses normal FIFO and switches to RD FIFO
- Adds Header packet:
  - BCID, R3ID (matches L1ID?)
- Ignores second R3 if busy (or inserts empty event)




FEICs have extra logic:

- Copies event from pipeline
- Selects only useful hits (see later slide on compression)
- Reads out data independent of any queues (private queue)
  - OR (more radical) - dedicated RD-links on hybrid to MCC

# FEIC Data Format

- A large component of RR latency is in transferring the data
- We presume data will be transferred in packets
- Smaller packets are better for RR latency
  - RD might have to wait for whole packet to finish, so this defines latency
  - Presuming whole packets are sent from the FEICs, the regional data needs to factorise neatly to avoid wasted bits (xN chips)
- e.g.10b packet:
  - 1 start-bit
  - 1 type bit 0: event data 1: regional data
  - 8 data bits
- When transmitting data a regional data packet is inserted at the front of the queues – only wait for packet in transmission

# Off-Detector Regional Data Flow

- SMC forwards RD as per normal data top RODs
- ROD scans data as close to input as possible
  - Scan for packets with RD bit set
  - Route data to Track-finder instead of ROD event processor
- For lowest latency, a separate TF-link-out/Stave-link-in is ideal
  - but data-volumes are low, so sharing needs to be investigated
- On a shared link data should be grouped and tagged by link-ID,BCID
  - Waiting for large chunks of event ROD has latency implication
  - Sending data in small chunks add tagging overhead
- Using an async Track-Finder means events can be queued
  - Events subject to higher BCID offsets can be prioritised
  - Need to make sure bandwidth out is sufficient for average readout rates
- Geography needs to be considered:
  - Staves in same RoI region ideally don't share RODs
  - For the barrel this makes sense – arranged radially, EC??
  - Every third stave on same ROD

# Track-Finder

Off-detector system for finding tracks/stubs

Not defined, but we have some ideas:

- Segmented geographically, with overlap
- Separate systems for each segment
- Each RoI (BCID) gets it's own processing unit

Track-finder essentially "interfaces" regional data to Level-1, so:

- Async input (bingo mechanism)
  - Will build a tracklet set as data arrives – not reliant on synchro
    - Assign a separate processing unit per event
    - TF latency defines number of units required
    - As packets arrive from the FE processing can begin
      - No need to wait for full 'event'
- Synchronous output
  - Send AVAILABLE data to L1 a fixed latency after BC
    - i.e. tracks found are sent, late track are lost – no waiting…

# Readout Architecture

Two new readout architectures are interesting:

1. Tracks in L1
   - regional readout contributes to L1 decision
   - Minimise latency (2-5us)
   - Longer pipelines on detector
2. Level 2 buffer on detector (lets face it L1.5/L2 are the same thing)
   - Regional data contributes to L2 decision
   - Less latency constrained (but only a little less: ~10us)
   - Higher rate L1 (400KHz)
   - Larger FE buffers can keep data awaiting L2 decision, most is rejected
   - The interesting bit: stave bandwidth goes DOWN!
     - L2 rate at 10-40kHz

Essentially it boils down to longer pipelines vs. bigger buffers