ITk DAQ Requirements

ITk DAQ Readout Group

September 18, 2017

Abstract

Summary of the DAQ requirements to operate and calibrate the ITk detector in the ATLAS experiment.

Revision History

Revision	Date	Author(s)	Description
0.1	10.07.2017	Heim	Initial version
0.2	09.08.2017	Gallop	Merge most of Strips document
0.3	18.09.2017	Heim	Adding Pixel comments and some revision

Contents

1 Introduction

This document summarises the DAQ requirements to operate and calibrate the ITk detector in the ATLAS experiment. There are significant differences between the Pixels and Strips implementations, but an attempt is made to unify the requirements. It does not intend to specify the implementation nor give enough detail to be used as a baseline for the implementation, the purpose of all information in this document is to motivate the requirements. Note that particularly the Pixel chip is still being specified, the information included in this document is inspired by the RD53A demonstrator chip, but as it is just a demonstrator chip and will be superseded by a specific ITk Pixel readout chip, some features might not be implemented in RD53A or might change in the future readout chip due to test results from RD53A.

Note to Pixel people: I redefined in an effort to simplify "module" definitions some features which might seem strange to those who know the detail. Note: I do not understand the purpose of the "ITk units", hence I left them out for now. Generally "higher-level DAQ" is meant to be ITk sw and "lower-level DAQ" the system directly attached to the link.

2 Other documents

This document is to be read in conjunction with the TDAQ interfaces document [4]. Note that that document emphasises the flow of data over the front-end interface, whereas this tries to describe a broad view of the system, from the point of view of ITk.

The DAQ interface section of the Strips TDR [3] currently describes a superset of the following (for both Pixels and Strips). but this document should be made to supersede that description.

3 Pixel Overview

The ITk Pixel detector (from a DAQ perspective) is composed modules of 4 readout chips. Each of these modules is connected to one slow (160 Mbps) TTC downlink and 1,2, or 4 fast (5.12 Gbps) uplinks (depending on in which layer of the detector it is mounted). Modules are mounted on two types of structures: barrel staves which distribute modules over z at fixed ϕ and r, and end-cap rings which distribute modules over ϕ at fixed z and r. Modules at positive z will be connected to a patch panel on the A-side of the ATLAS detector and modules at negative z will be connected to C-side. Modules which are connected at the side of the detector from the same structure are fed by one fast

(2.56 Gbps) TTC downlink, which is split up by a multiplexer ASIC[1] into up to 16 slow TTC links which connect to every module. From the user/software perspective this fast TTC link should be transparent. The uplink and downlink of the same module should be handled in the same lower level DAQ instance.

It is important that the uplink and downlink of one chip end up in the same lower-level readout unit, this means that there is strong asymmetry of uplinks to downlinks in the lower-level readout unit. For instance in the innermost layer there are up to 64 chips per one 2.56 Gbps TTC downlink, but each of these chips has a 5.12 Gbps uplink. This results in a 1 downlink and 64 uplink mapping and it is important that all of these 64 links end up in the same lower-level readout unit.

3.1 Downlink

The readout chip recovers the 160 MHz clock from the 160 Mbps command stream. A custom encoding ensures that enough transitions are present in the bitstream for the CDR circuitry to work properly. It consists of a continuous stream of 16-bit frames. The protocol specifies the following commands:

- 15×Trigger: one 16-bit frame sent at 160 Mbps covers 4 bunch crossing, 15 trigger commands cover all possible trigger permutations, includes 5-bit trigger tag, broadcast.
- ECR: 1 × 16-bit frame, event counter reset, broadcast. (Aligned to 4-bc frame)
- BCR: 1 × 16-bit frame, bunch crossing counter reset, broadcast. (Aligned to 4-bc frame)
- Global Pulse: 2 × 16-bit frame, includes 4-bit chip id and 4-bit data.
- Calib. Pulse: 3 × 16-bit frame, includes 4-bit chip id and 15-bit data.
- Write Register: 4 × 16-bit frame, includes 4-bit chip id, 9-bit register address, and 16-bit data.
- Write Register: 12 × 16-bit frame, includes 4-bit chip id, 9-bit register address, and 96-bits of data.
- Read Register: 3 × 16-bit frame, includes 4-bit chip id, 9-bit register address
- Sync: 1 × 16-bit frame, synchronisation frame, used as idle frame, needs to be sent periodically.

The payload of frames is split up in 5-bit fields which are encoded via a custom encoding to 8-bit to achieve DC balance. Commands which consists of multiple frames do not need

[1] Could be an lpGBT, but might also be a GBT-like custom ASIC.

to be sent consecutively (but still in order), i.e. 'Write Register' can be used to fill the gaps in between triggers. An ECR will reset large parts of the chip including the command decoder and therefore cancel any command which was in process of being sent. It will also delete any data or triggers in the pipeline, hence need to wait for buffers to empty before sending ECR.

3.2 Uplink

Each uplink is transmitting at 5.12 Gbps using the Aurora 64b66b protocol[2]. The DAQ needs to facilitate the necessary means to synchronise to the data stream, which might require sending commands to the chip it is connected to. There are two types of Aurora frames which can be identified by their sync header, data frames and register frames. Register frames will only be sent after a specific number of data frames. If there is not data to send out, a idle frame will be sent which also identifies as a data frame by it's sync header. All register frame will contain a 4-bit status code.

In order to request a resynchronisation, for instance in case the DAQ fails to sync up in time, specific registers in the chip have to written.

4 Strips Overview

The Strips detector [3] is made up of staves (barrel) and petals (end-cap). There are two kinds of staves, the outer layers having a coarser granularity due to lower occupancy (long strips vs short strips), but all petals are the same. These are similar from a DAQ point of view, differing by the number of devices connected.

A stave is made up of 14 modules on each side, corresponding to 28 data streams per side for the inner barrels (14 data streams for the outer layer). The end-cap petals have a less regular structure, with 9 modules per petal side but similar total bandwidth requirements (see appendix).

Within a module, a hybrid consists of a number of ABCStar front-end ASICs [1], accessed via an HCCStar ASIC [2] in a star configuration. For each HCCStar, there are between 6 and 11 ABCStar ASICs. Additionally, power and bias settings are controlled via the AMAC ASIC. Monitoring of power and temperature is through a combination of the AMAC and the HCCStar.

The stave/petal control and data path is implemented by the lpGBT. Each long strip stave and petal is controlled by two lpGBTs, one for each side. For the short strip stave there

[2] https://www.xilinx.com/support/documentation/ip_documentation/aurora_64b66b_protocol_spec_sp011.pdf

are two lpGBTs on each side, in order to increase the uplink bandwidth. The downlink of only one of these is used, the uplink clock is slaved to the second.

4.1 Downlink

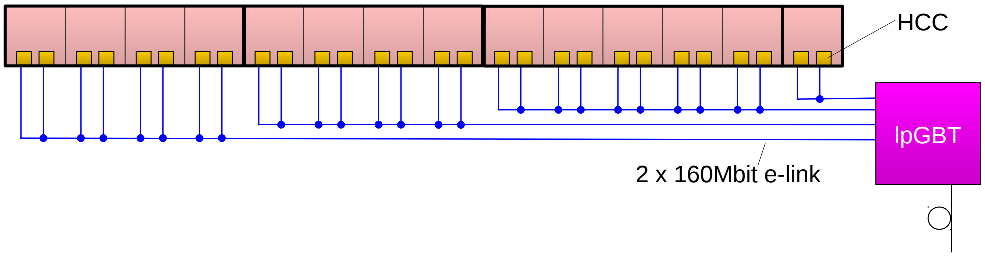

Figure 1: Strips downlink, showing distribution of TTC data to segments of 4, 4, 5 and 1 module on a stave

The commands to are encoded using a custom protocol. This encodes the L0A, command and BCRs (hence LCB) into a single 160 Mbit e-link. A second e-link is used to send L1 and R3 readout commands when necessary. Within a stave/petal side, the modules are divided into 4 groups and a separate version of the LCB and L1/R3 signals is sent to each group.

The LCB protocol encodes commands using 6b/8b over a 4 BC frame, this allows 12 bits for each frame. The logical commands sent using the LCB protocol are as follows.

- L0A with tag. 4-bit mask + 7-bit L0 tag
- Bunch crossing counter reset
- Synchronous resets
- Calibration pulse
- Digital pulse
- Register write
- Register read
- Hit counter commands

Multi-frame commands can be interleaved with other commands, so for example register writes can be inserted dynamically in between triggers.

4.2 Uplink

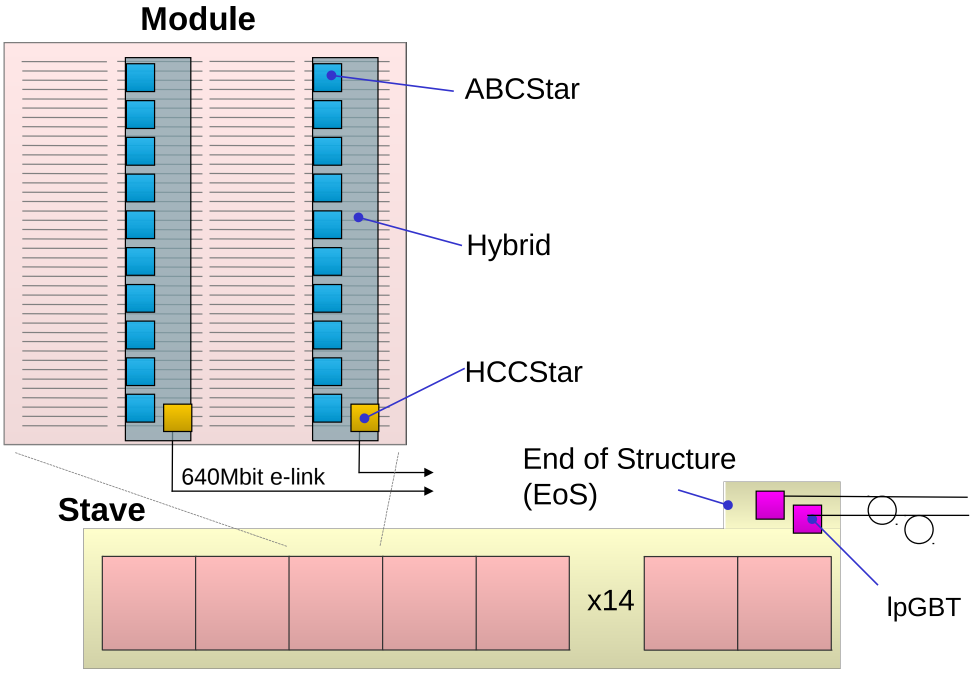

Figure 2: Strips uplinks on short-strip stave, showing 14 modules driving 28 e-links into 2 lpGBTs

Each module generates data on 1 or 2 640 Mbit e-links (one per HCCStar) which are multiplexed with data from the other modules on the stave/petal side by the lpGBT, for a maximum of 28 e-links from a short-strip stave side.

The line encoding uses a custom packet format, optionally encoded using 8b/10b with k-codes for start/end of packet and idle. Alternatively, packets are delimited by a preamble and trailer.

The predominant type (by data volume) will be event-data, but other types include register-readback, occupancy counter data, DCS data and warnings/alerts. Data for an event will arrive in one packet, taking up between about 10 and 60 bytes (depending on occupancy). Due to the variable length and queuing in the HCCStar, data from different hybrids will arrive at different times. Register data is readout using different modes, depending on whether a single chip is addressed, and whether the command is to be interleaved with event data.

4.3 Power Control and Monitoring

In the Strip detector, the lpGBTx provides the only, with the exception of interlocks, path for controlling and monitoring the detector. During power up of the detector, the DC-DC converter on each module must first be enabled in order to power the HCCStar and the ABCStar chips. This is done by the AMAC ASIC which communicates with the DCS system by means of an lpGBTx slow control adapter (SCA) channel and the downlink fibres. AMAC also includes several ADC channels which allow on-module currents, temperatures and voltages to be monitored throughout the powering procedure. Only once the module is powered can communication be initialised via the standard e-links.

5 Operation

This section will discuss all requirement which are important for the operation of the detector in data taking conditions. Note that this is a logical mode of operation and may not be wholly separate from other modes of operation.

5.1 Trigger Schemes

The system must deal with both potential trigger schemes. In particular:

- L0 at 1 MHz (L0 latency: 12.5 μs)
- L0 at up to 4 MHz + L1 at up to 800 kHz (L0 latency: 12.5 μs, L1 latency: 32 μs)

Most of this document applies equally to both cases. Find a way to accentuate differences where required

5.1.1 Pixels

The outer detector layers can operate at an L0 rate of up to 4 MHz. The inner layers can operate in two modes: either at an L0 rate of up to 1 MHz, or at an L0 rate of up to 4 MHz in conjunction with an L1 trigger. The system can handle up to 16 consecutive triggers as long as the bandwidth is not saturated. Each trigger command (up to 4 consecutive triggers per trigger command) needs to be sent with a tag generated by the DAQ (e.g. lower bits of trigger id), which needs to be saved in combination with the bunch crossing id and number of trigger. To read out the inner layers (1st, and 2nd) at an L0 rate of up to 4 MHz, an L0 trigger is sent to the chip to buffer the data inside the chip and to perform the read out of the data an L1 trigger has to be sent with the matching trigger tag. The outer layers (3rd, 4th, and 5th) can handle a 4 MHz L0 trigger rate out-of-the-box.

Possibly clarify interaction with R3? i.e. this allows the L0 data from the outer layers to be used as input to L1-track? R3 might be used to filter the data sent to L1-track?

5.1.2 Strips

For the single level trigger, only L0A is to be generated off-detector. The low-order bits of the L0ID are encoded to identify the BC using an L0 tag. Due to the LCB protocol, L0As in consecutive bunch crossings (within the 4 BC frame) will be given consecutive L0 tags. A readout request incorporating the appropriate L0 tag is generated internally to the FE ASICs.

For the dual-level trigger, L0A, L1A and R3 come from TTC. L0A is handled similarly to the single level case, but readout is not automatic. The L1A is translated into a readout request which identifies data associated with a particular L0A using the tag. The R3 is translated into a similar readout request that is directed at particular hybrids on the detector.

Not that the bandwidth for R3 and L1 data is shared. We expect the R3 to read out 10% of modules, so a 4 MHz L0A uses 400 kHz of bandwidth, leaving 600 kHz for L1. For L1 readout at 800 kHz, this suggests an L0A of 2 MHz.

5.2 Downlink

Pixels: The 160 Mbps command stream needs to be synchronous to the 40 MHz LHC clock as the readout chip will generate its internal 40 MHz clock from the recovered 160 MHz clock. During operation the downlink is fed by multiple prioritized pipelines:

1. ECR pipeline: send ECR (timing)
2. Trigger pipeline: send trigger command if there was a trigger in the last four bunch crossings (≈16 Mbps)
3. Sync pipeline: send sync if sync frame has not been sent the last 32 frames (≈10 Mbps)
4. Priority command: send high priority commands received from higher level control
5. Trickle configuration: constantly sends configuration to chips

Higher level DAQ needs to have access to the bitstream to the tricker configuration buffer memory at all times to perform possible necessary changes to the configuration in response to monitoring data. It should be possible enable/disable the trickle configuration for any of the chips which is connected to the same TTC link, in case it's configuration being modified or is being taken out of the run. Initial configuration of the system can be performed via the priority command pipeline.

5.2.1 Trickle configuration

Pixels: It is expected under HL-LHC conditions to have an SEU rate in the pixel registers which results in 1% of all pixels failing in 10 s. As the cumulative fraction of failing pixels should never exceed 1% (and there are more failure modes of pixels than SEUs), it is necessary to reconfigure the full pixel matrix at least every second or more often. With a trickle configuration (continuous reconfiguration) one configuration of a single chip requires ≈0.1 ms (assuming a bandwidth of 130 Mbps), as there are 4 chips on one TTC link this rises to 0.4 ms. This is already on the level of what is demanded bye SEUs and it therefore necessary to perform a continuous reconfiguration.

Strips: Putting some numbers to this, for the Strips case. In order to trickle the configuration into the detector, we send write register commands when there are no L0As. Given a trigger rate of 4 MHz, we send 357 L0A per orbit, which uses up to 1428 BCs. This leaves 2136 bunches, which can be used to send 59 write commands (36 BC each). A complete reconfiguration ($\sim 10k$ registers for the modules in one TTC domain) is therefore possible in 170 orbits or 15.2 ms. Using a more conservative scheme, sending 1 register per orbit would take 0.9 s.

Because of how the encoding scheme operates, this can be done dynamically. The encoder can check for triggers and insert part of the write register command if there are no L0As for this frame.

5.2.2 Configuration Size

It may be useful for allocation of FIFO sizes to note the size of the bit-stream needed to configure the detector.

A few different numbers are presented in table 2.

Pixels The readout chip has 1 kB of global register and 157 kB of pixel register memory. A complete bitstream to write the full configuration into one chip is approx. 350 kB (depends on exact implementation), i.e. the bitstream to configure a full module (all chips attaches to one TTC link) is 1.4 MB.

	Pixel	Strips
Bits per channel	8	7
Channels per chip	160 k	256
Channel bytes per chip	157 k	224
Global bytes per chip	1024	44
Total bytes per chip	158 k	268
Bits per register write	64/192	55
Max chips per TTC e-link	4	100
Max bits per TTC e-link	11.2 M	368 k
Max chips per TTC o-link	64	280
Max bits per TTC o-link	175 MB	1.03 M
Chips per sub-detector	6400	233856
Bits per sub-detector (unframed)	7.7 G	862 M
Bytes per sub-detector (unframed)	987 M	108 M

Table 2: Approximate size of configuration data.

Strips There are 67*32-bit registers per FE chip in the current specifcation (made up of 11 chip-global registers and channel registers that account for a further 56). Configuration is sent to a maximum of 100 chips over one e-link (5 short strip modules on the barrel stave). Sending 6.7k registers at 36 BCs per register takes about 6 ms. Note that this excludes configuration for the HCCStar and AMAC.

5.2.3 Timing

It needs to be possible to adjust the downlink in 160 MHz clock steps to align the commands frames to the 40 MHz bunch crossing clock. The trigger input stream should be adjustable in 40 MHz clock steps. Finer delay to adjust for sub 160 MHz clock steps will be handled inside the chip. The delay from trigger stream input to trigger command output needs to be constant and deterministic.

Note that both Pixel and Strips downlink encodings stretch commands over 4 BC frames, so alignment of the BC reset is done by changing this delay.

5.2.4 Maskability

The trigger and ECR pipeline can only be enabled on a per module basis. All other pipeline should be able to be enabled/disabled on a per chip basis.

5.2.5 Addressing

While some of the commands mixed with trigger signals may be global in nature (e.g. a counter reset), many of these commands will contain data specific to certain modules on a downlink. Particularly for calibration and fast module configuration, the trigger signals on the downlink should be independently addressable at the e-link level within a downlink, so that the configuration can be sent under local control of the ITk Control Unitor via a preloaded command buffer in the unit-that-sends-the-downlink-data.

Pixels The connectivity of one chip should be fully defined by its e-link and chip-id. The DAQ will use the corresponding e-link and take of the sub e-link addressing in sw.

Strips example By way of an example, the present strip stave design configures up to 100 FE chips through a single e-link (5 modules of 20 chips). The address field of these register write commands uniquely defines the chip within the e-link, but not the module within the set of all possible Pixel or Strip modules. The address will likely define the target module only within a certain e-link (in order for it to use a minimum number of bits). Therefore, commands should be routed to the desired e-link of the desired optical downlink using external information.

5.3 Uplink

The data received from the chip needs to be matched with trigger tag. No data frame will be sent twice, i.e. if specific data will be requested at another point in time it needs to be retained by the DAQ system. The delay from sending the trigger command to receiving the data is variable and depends on the links occupancy. however the chip will mix the data of multiple triggers.

5.4 Routing

Pixels: TBD, endpoints unclear

The data should be routed to the following units:

- A copy of a programmable percentage of the data should be routed to the monitoring, to check the detector status.
- All register frames should be sent to control to check the status code and act accordingly.

- Some register frames (depending on address) should be sent to DCS as they contain chip internal ADC readings.

Packets should be routed to destinations based on the type of the data arriving. Other factors easily extracted from the packet, for instance L0ID, BCID, occupancy (size of packet), errors might also be useful to feed into the routing decision.

In the dual trigger case, appropriate data should be sent to L1-track. Specific requirements on latency etc. are out of scope of this document.

5.5 Monitoring

Pixels: TBD, endpoints unclear

Some configurable percentage (up to 100%) of data will need to be passed to the ITk Analysis Unit which will monitor the quality of the data. This could be the same unit which buffers the data pending a L1 trigger, but the unit buffering the data will need access to all the trigger information, at least in order to check counters. Also, the routing of data on the uplink from the affected modules may be reprogrammed to send all data to the associated ITk Analysis Unit.

The primary purpose is to look for faults in modules that may not be obvious from examining data available at the output of the event builder and to respond to any warnings issued by the on-detector electronics. The collection of this data must have no effect on the physics data flow.

5.5.1 Collect and Analyse Physics Monitor Data

Pixels: TBD, endpoints unclear

In addition to chip occupancies, this monitoring should include measuring the occupancy of the front-end link, i.e. how much more data can be sent down this link. This is a function of the raw event size, and the space between packets. Another useful variable to keep track of is the latency, how much time has passed between the transmission of the trigger and reception of the last data for this event.

5.5.2 Triggers that are not Useful for Data Analysis

It will also be useful to collect data based on conditions when no hits are expected, for example looking for stuck memory cells, or based on low level event characteristics like unusually large events, which could be due to a threshold misconfiguration (caused by an SEU).

5.6 Configuration

Pixels: Exact states unknown, used to be driven by DCS, all new with serial powering.

Query, should mentions of configuration above be moved here?

Sending configuration commands to the detector needs to be done in a few different circumstances.

- **Power-up/Standby:** When the detector is turned on it is not configured. As part of the start-up sequence, the configuration should be send to the detector.
- **Regular reconfiguration:** In order to protect against degradation of the module configuration due to SEUs, it is proposed to continually send configuration commands to the detector making use of the time between triggers.
- **Module recovery:** This is similar to the power-up case, but for one module while operations continue on the rest of the detector.

In two of these cases, control of the downlink is shared between the normal TTC and the unit generating the reconfiguration stream. For module recovery, each downlink should be treated separately to avoid conflicts with the recovery of multiple modules.

5.7 Diagnostics

Some of the register frame status codes, register values, or issues with the uplink should be handled in a fast manner in the lower-level DAQ. Any action performed by the lower-level DAQ needs to be communicated to the higher-level DAQ, e.g. disabling a link due to sync loss.

The full link data should be available for diagnostics. In particular framing information to diagnose misbehaving links, and this should be possible in parallel with normal data taking.

5.7.1 Maskability

Each uplink should be maskable in case the attached module is disabled, not yet configured, or other link failures which lead to high traffic.

5.8 Stop-less Module Recovery

Module recovery should be possible without stopping data-taking for the remainder of the detector. This should follow from a combination of the addressing, masking and configuration points above.

1. Notice a problem with a module (via monitoring or error detection etc).
2. Mask off TTC commands being sent to a module.
3. Send start-up and configuration commands.
4. Re-check link for problem.
5. Re-enable TTC commands to this module.

The full recovery cycle should be performed in a time in the order a few seconds to avoid excessive data loss and instabilities caused by a misbehaving module.

5.9 Trigger sources

For testing purposes it should be possible to go into data taking mode using different kind of trigger sources, e.g. cyclic triggers, external trigger sources, or the chip internal self trigger.

5.9.1 Multiple triggers

A feature dropped from SCT vintage chips for the upgrade is the ability to read out 3BCs together, which is not needed due to the implementation of the contiguous L0 requirement. For timing studies, it should be possible to send a sequence of contiguous L0A in place of one, up to 64. Note that this might violate the complex dead time for ultimate conditions, but is only required when the occupancy or trigger rate is low enough not to saturate the read-out bandwidth.

5.10 Control Hierarchy

For testing purposes it should be possible to run different parts of the detector at the same time with different data-taking configurations. Should be as specific as possible about what different data-taking configurations (trigger sources?) and the levels at which control should be moved around For instance, running calibrations on a stave while taking cosmic data with the remainder of the detector.

The minimum unit should be the stave/petal in Strips, and the module in Pixels.

5.11 Busy

The front-end chip protocols and buffer sizes are designed so as not to fill up when working with nominal event occupancies and trigger distributions. If a buffer nevertheless fills up and event information is dropped, this will be notified either by a flag in the next data packet, or in the read of a status register.

In order to mitigate against unexpectedly high occupancies, the front-end chips are expected (details TBC) to contain a configurable max. event size, if an event is to be read out which is larger than this the event will be chopped at the programmed size and the remaining data will be thrown away.

As the response time for raising BUSY is relatively long compared to the time to fill a buffer, it is expected that these errors will be integrated into the data quality system and not raise BUSY.

Should we keep this? It may be useful to generate BUSY based on aggregation of a large numbers of these errors, with a possible pre-scale. This could happen either within the data path itself, or from ITk Control Unit(this might also happen when configuring the detector before a run).

In particular, when triggers are generated locally errors such as these might also be acted on to gate the local trigger.

There should be full monitoring of any locally generated BUSY.

5.12 Event Building

Due to the trigger tagging scheme, the event data from the detector contains only the L0 tag and a portion of the BCID. It is therefore necessary to record details (BCID, L0ID and possibly L1ID) of the events corresponding to the triggers sent to the detector. This allows the reconstruction of the full trigger information.

As part of this reconstruction, the following should be checked for every event:

- L0 tag is correct (bad tags to be dropped with an error)
- Every link sent a data packet
- (for Strips) the HCC didn't flag a missing chip
- (for Strips) BCID is correct for this L0 tag

- Other sanity checks to be defined (for instance, monotonic ordering of cluster addresses)
- Should ano large events for moinitoring.

5.13 ROI

Strip only!

The exact form of ROI information from the trigger system and any possible latency between the availability of the L0 signal and the ROI information are yet to be finalized, but the procedure can be split in the following way:

- An identifier will be sent from the ROI system in terms of specific geometric segments of the ITk
- This geometric identifier will be mapped to an ITk identifier
- The ITk identifier specifies a particular uplink, which corresponds to a specific module located somewhere in the ITk volume
- The ITk identifier also specifies a particular downlink e-link, which is sent to a particular detector module

We should specify where this mapping should occur and how it should be configured.

[We believe that maintenance of this mapping will be the responsibility of the ITk since the allocation of uplinks, fibers, etc. will be its responsibility].

In the Strips case, a priority trigger is sent to the detector with a reference to an L0 tag. Therefore the translation of ROI information must be performed prior to the formation of the final downlink data stream.

[In the Pixels case data intended for L1 should be filtered out of the L0 data stream, using this ROI information.]

In both cases, the translation of ROI segment identifier to ITk uplink identifier will need to be made somewhere between the Trigger System and the units servicing the downlinks or uplinks. If the translation is to be made for the downlinks, it will need to be performed in a very short time in order to meet latency requirements. For the alternative case, there will be a little more time before the data arrives from the uplink, but the directing of data to L1-Track or not to L1-Track must still be fast enough to meet the L1-Track latency requirement.

6 Calibration

Calibration of the detector requires the chips to be configured with special settings and the injection of calibration pulses.

6.1 Downlink

During calibration two actions are performed:

1. Send a configuration bitstream containing the whole or parts of configuration to each chip.
2. Send (one or multiple) trigger commands and (one) global/calib. pulse commands spaced by a programmable but fixed delay. This bitstream is sent out with a fixed frequency.

The higher-level DAQ will take control of preparing the configuration bitstream. The trigger and calibration pulse bitstream can be prepared by the higher-level DAQ, but should be stored by the lower-level DAQ to ensure the fixed delay and frequency (at least 512-bit, to be automatically repeated by a programmable amount or for a fixed time). Instead of sending the trigger and calibration pulse commands with a fixed frequency, they can also be sent in a dynamic manner once the data has been received by the uplink to speed up the calibration procedure.

6.2 Uplink

All of the data has to be sent to the higher-level DAQ for analysis. To ensure short calibrations times this should happen at the full uplink bandwidth.

6.3 Control Hierarchy

It should be possible to run different kinds of calibration scans on different detector parts at the same time. The smallest group would be the module in Pixels (NB this is a subset of lpGBT for TTC) or the stave/petal in Strips.

7 Downlink routing modes

From ITK sw side: we first need to understand the contraints of the ”sw-ROD” system. Then we can define the routing models. So this, or more precisely any constraints on sw and routing, should be supplied by TDAQ

The idea is that this is common to operations and calibration, as it's presenting a way to implement both. But, how does it integrate with the overall structure

Strips: Signals sent to detector are encoded onto the physical downlinks to the FE chips. They can be logically split into:

- BC clock
- L0
- L1 (PR and LP, using tag)
- Read register
- Write register
- BCR
- Other commands (resets etc)
- DCS
  - Read and write registers on FE chips
  - Read and write registers on power chips

There are various potential sources for these signals.

- TTC
- DCS
- Ctl&Cfg (a.k.a. ITk Control Unit)
  - From operator control console (diagnostics)
  - Automatic responses to incoming data being monitored
  - As a response to status reported by FE ASICs
  - Periodic resets, reconfigurations
- Local

The 40 MHz system clock must run continuously as long as the FE ASICs are powered.

We can define different modes where commands can be sent from particular sources and merged. For instance, when merging regular reads from DCS with re-writing configuration.

	Operation	Calibration	Raw	Configuration
BC	TTC	TTC/Control	TTC/Control	TTC
L0A	TTC	Control	Control	None
L1A	TTC	Control	Control	None
Registers	Control/DCS	Local/DCS	Control	Control
BCR	TTC	TTC/Control	Control	None
DCS registers	DCS	DCS	Control	DCS/Control

Table 3: Routing of control data. Depending on the mode (column), logical commands (rows) might be sourced from different controllers.

In this case, it should be possible to switch modes without interrupting the coding on the FE link.

Note that this might be affected by addressing requirements noted in section 5.2.5; a mode should be applicable at the link level.

For instance, table 3.

The FE protocol will encode these signals in a deterministic way. For instance in the Strips LCB protocol bunches of 4 L0s are encoded with a tag into a 160 Mbit stream encoded using 6b/8b. It should be noted that with this scheme, the codes used for register access are exclusive to L0 triggers. This means these commands should be stored in a FIFO for interleaving into the data when no L0s are to be sent.

8 DCS

Detector safety is assured by an independent system, but both Strips and Pixels have (to different extents) information that is relevant to DCS). For Strips this is part of the control path.

Needs review from DCS side

8.1 Pixels

No vital detector control will be sent via the module down/uplinks, however DCS might issue sporadic register read commands specific chip register for extra information (e.g. internal temperature or voltage). As the uplink data stream contains a fixed amount of register frames, all registers which contain information interesting to the DCS should be sent to it in any case.

8.2 Strips

For detector safety, there is an interlock on the temperature of the cooling pipe to turn off power to the detector. For all other control and readout the data goes through the same lpGBT.

The lpGBT EC link will be used to communicate with a bus of power chips (AMAC). This is likely to be implemented using bit-banging. The data from the AMAC will arrive asynchronously wrt to the clock, but at a lower rate.

Some of the GBT-SCA functionality of the lpGBT will be used (eg ADC and some IOs).

As monitoring of the detector status is via lpGBT, it is likely there will be watchdog running in case this is lost. Therefore, loss of lpGBT link should be signalled to DCS. Receiving data from register reads will also reset the watchdog. If there is no response the power will be shut off. While the detector is turning on or off this should be short, but once stable a couple of minutes should be OK TBC!.

It should be noted that if off-detector lpGBT is not on UPS, we lose the links if we lose power in the counting room.

For Strips, DCS data will be sent up the uplinks and commands from DCS will be sent via the downlinks. This uplink data needs to be deciphered and sent to the DCS port at all times, even if the full TDAQ infrastructure is not operational, and commands from DCS and from the control console must be sent to the detector at all times as well. This requires the logic that mixes trigger and ITk commands for the downlink to be operational at all times even if TTC or other trigger processing is not operational. In fact, there needs to be some fail-safe or interlock that prevents powering up of parts of the detector if these data paths between DCS and the detector are not operational.

To make this point clear, the DCS data, both monitor data and configuration read-back data, will come up the same e-links shared with the event data. It will be in packets just like other register read-back data, but with a different address indicating that it is DCS data.

9 Implementation Notes

Here are presented some notes on possible implementations based on the above.

9.1 LTI Bandwidth

The LTI proposal has a large output bandwidth. The following is based on driving 8 PON networks, each of which can potentially be split into 32.

Is it possible to do all FE protocol encoding (from TTC commands) in the LTI and use the FELIX as a pure pass-through.

It looks like this would require too large a bandwidth to be workable. Therefore the FELIX needs to do at least some protocol encoding.

9.1.1 Strips

Each lpGBT drives 4 domains with LCB protocol (160 Mbit). For configuration these must all be independent.

Of the 80 user bits available from each PON this uses 16 bits per lpGBT, allowing 5 lpGBTs per PON. This therefore implies more than one PON per FELIX, and no splitting of PONs. Another implication is that DCS commands are now sent via the LTI.

Including L1 and R3 in this calculation further increases the required bandwidth. Mapping from ROI to R3 commands are unique to each stave. For most purposes the L1 command can be common.

9.1.2 Pixels

review numbers by Pixel. Useage unlcear.

9.1.3 Command Distribution

An alternative to a full FE-link encoding in LTI would be to make use of the user bits to encode commands (for instance write/read register), the values for which are filled in inside the FELIX. Again, commands sent from DCS should be taken into account.

9.2 Inter-block Commands

In several places, blocks are described separately, but it is important to note the need for software communication between. For instance sending trigger information to data handler for accounting purposes, or from monitoring to config & control for feedback.

	ABCStar	HCC	Modules per petal	Cumulative ABCStar per petal
R0	8 + 9	2	1	17
R1	10 + 11	2	1	38
R2	12	2	1	50
R3	7 + 7	4	2	78
R4	8	2	2	94
R5	9	2	2	112

Table 4: Table of petal chip counts per module.

References

[1] ABCStar, **EDMS**: *AT2-IS-CD-0002*

[2] HCCStar, **EDMS**: *AT2-IS-CD-0003*

[3] ITk Strips TDR, **CDS**: *ATL-COM-UPGRADE-2017-006*

[4] Interfaces with Detector Front-End Systems **EDMS**: *ATL-D-ES-0051*

10 Appendix

10.1 More Strips numbers

The ITk Strip Detector has 384 petals (192 for each end, with 32 in each of 6 disks). There are 256 long-strip staves (outer two barrels) with one lpGBTx per side and 136 short-strip staves (inner two barrels) with two lpGBTx per side. This is 768 lpGBTs for the petals, 512 for the long-strip staves and 544 for the short-strip staves. Totalling 1552 downlink and 1824 uplink fibres.

There are 14 modules on a barrel stave. For the short-strips, this means 280 ABCStar ASICs. For the long-strips, drop a factor of 2, 140 ASICs. This excludes HCCStar.

For ~~the~~ each petal we have 112 ABCStar. These are connected to 14 HCCStars, matching the 14 lpGBT uplinks. Due to the tapering geometry, the inner hybrids have more chips than the outer. So: