Western Norway
University of
Applied Sciences

ITS SystemC Simulations

Busy and data rate simulations of pp and PbPb at different event rates

Simon Voigt Nesbø

March 1st, 2018

Summary of ALPIDE busy mechanisms

- ALPIDE chip asserts BUSY_ON when:
  - 3 MEBs are in use in triggered mode
  - 2 MEBs are in use in continuous mode
  - Frame FIFO reaches ALMOST_FULL1 watermark (48 events)
  - **No data is lost when BUSY_ON is asserted.**

- Busy violation:
  - Occurs when all 3 MEBs are in use, the chip is busy and it receives a new trigger
  - This can also happen in continuous mode (theoretically)
  - **Data is lost**

- Flushed incomplete:
  - Occurs in continous mode when:
  - 2 MEBs are already in use, the chip is busy and it receives a new trigger
  - **Data is lost**

Summary of ALPIDE busy mechanisms

- READOUT_ABORT/DATA_OVERRUN mode:
  - Occurs when the frame FIFO reaches critical levels
  - Data discarded and empty frames transmitted
  - **Data is lost (lots of it)**

- FATAL mode:
  - Occurs when the frame FIFO overflows
  - All future data frames will be marked FATAL (until it is reset)
  - **Data is essentially garbage**

- Not observed in these simulations at reasonable event rates

Effect of «missing chips» on reconstruction

- The level of systematic error on the acceptance is on the percentage-level[1]
- If the fraction of missing chips for an event is lower than 1E-3, it will not affect physics
- Since “missing events” will be marked with busy violation flag, it will be possible for reconstruction to take this into account
- Full reconstruction should start working with contributions from a minimum of 4 layers

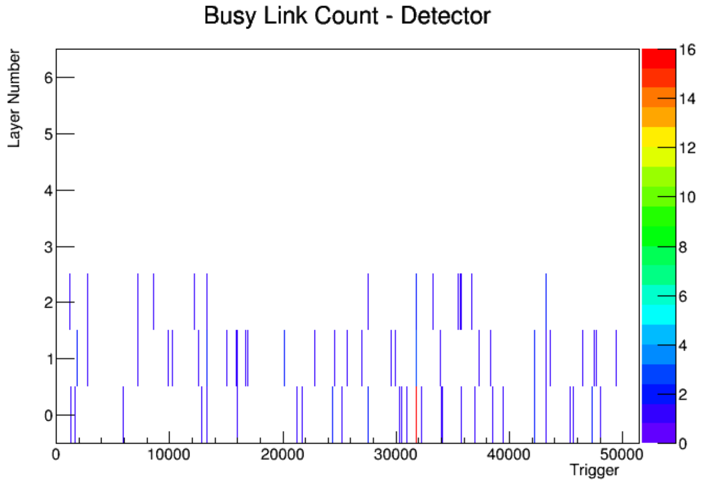
[1] Private communication with Ruben Shahoyan

PbPb simulations (triggered) – SystemC simulation setup

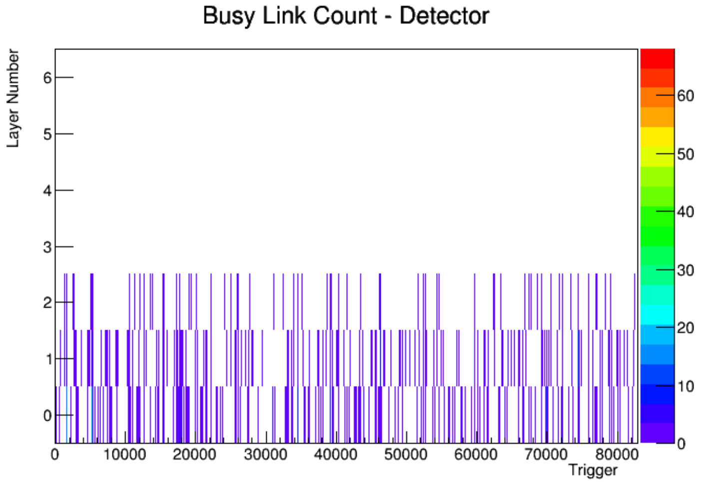
- Inner Barrel simulated
- Event rates: 50kHz, 100kHz, 150kHz
- 100,000 events for each simulation[1]
- PbPb MC event pool: 1,000 min-bias PbPb events (HIJING)
- QED MC event pool: 100,000 events
- Triggered mode
  - Strobe length: 100 ns
  - Trigger filter time: 1230 ns
- Pixel dead time: 200 ns
- Pixel active time: 6000 ns
- Minimum busy cycles: 8 (200 ns, the default)

[1] (stopped 50kHz and 100kHz sim at ~50k and ~80k events)

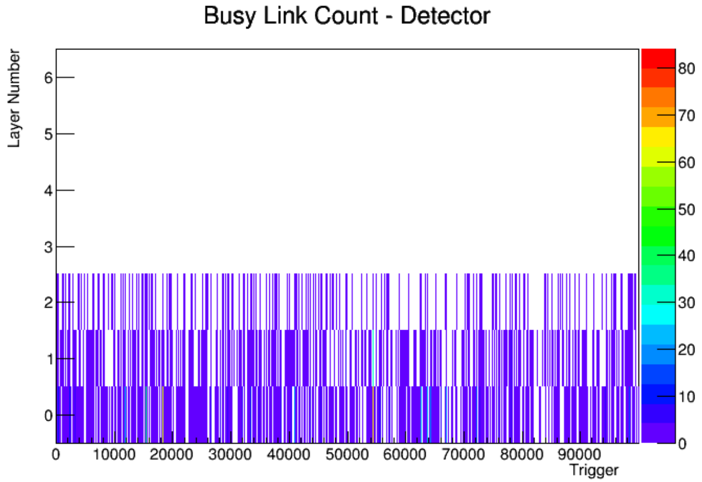
PbPb simulations (triggered) – Results



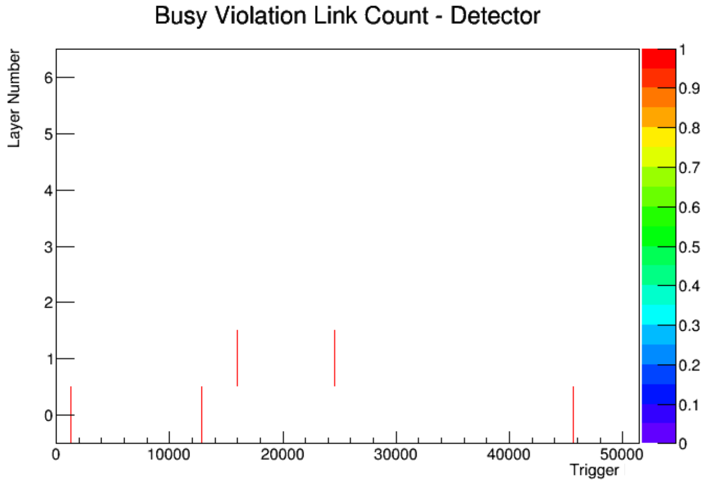
Busy link count, per layer, per trigger, 50kHz triggered



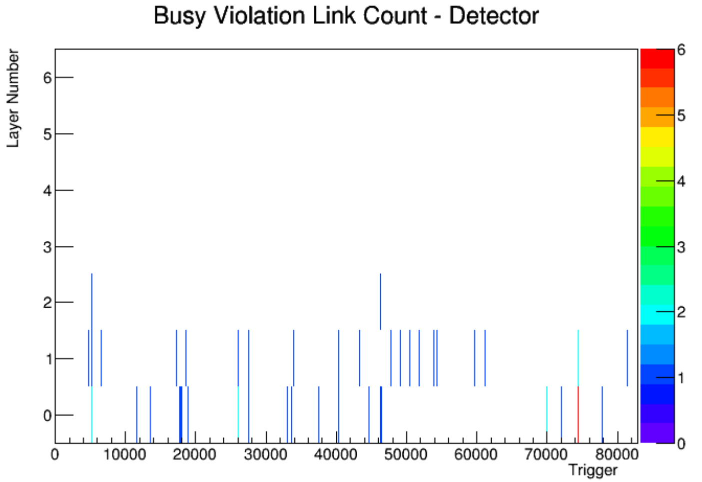
Busy link count, per layer, per trigger, 100kHz triggered



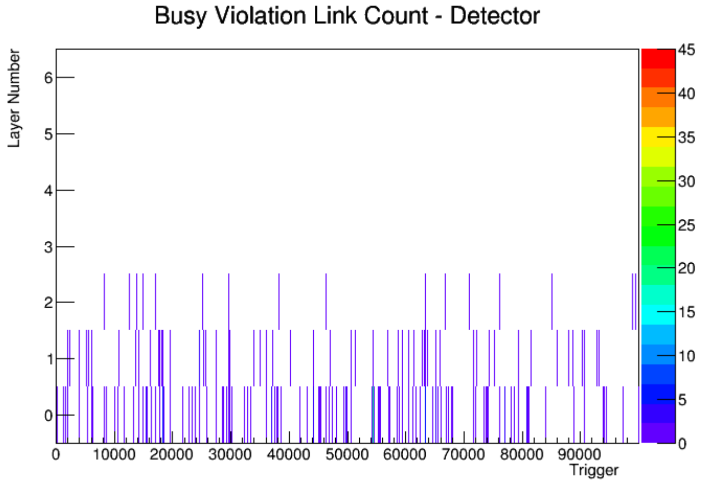
Busy link count, per layer, per trigger, 150kHz triggered



Busy link counter, per layer, per trigger, 50kHz triggered

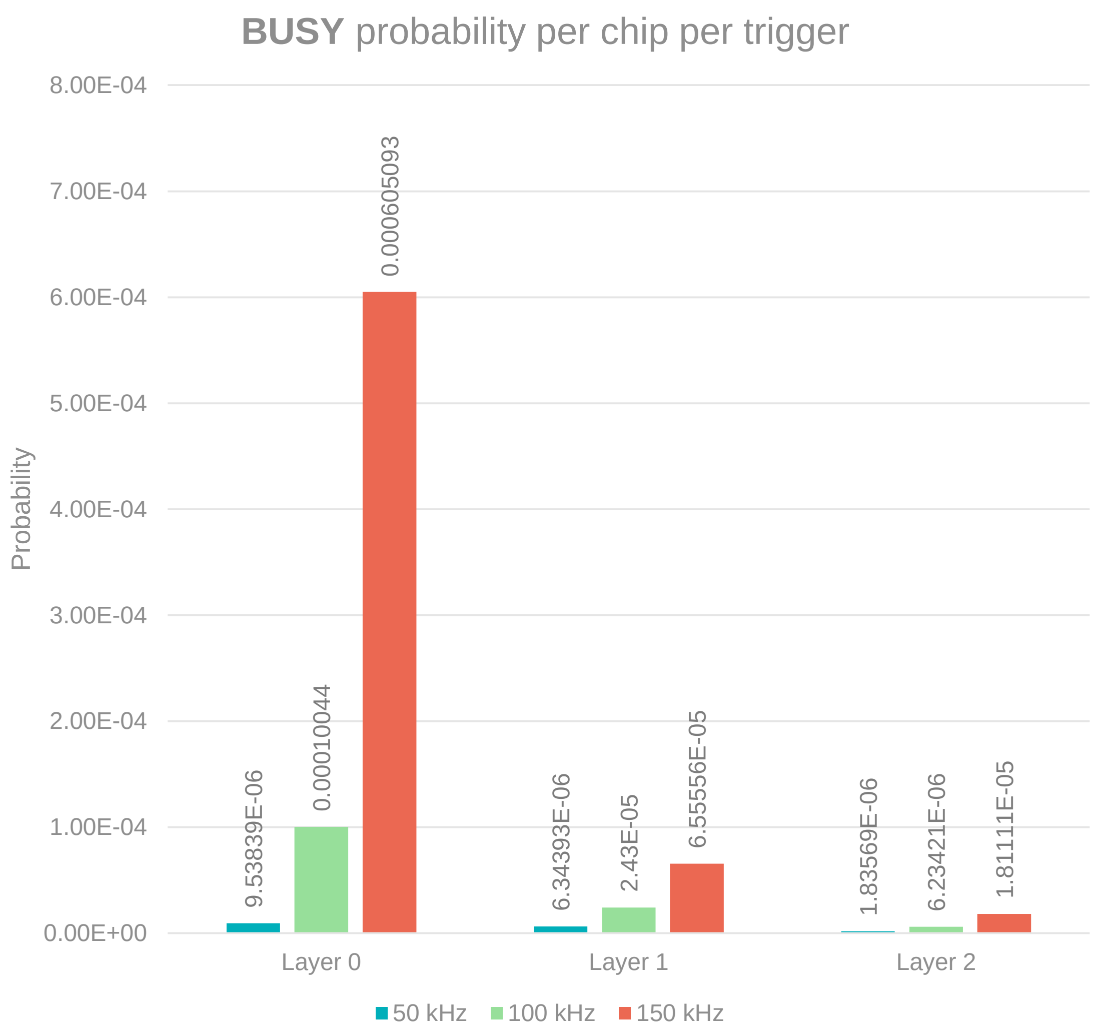


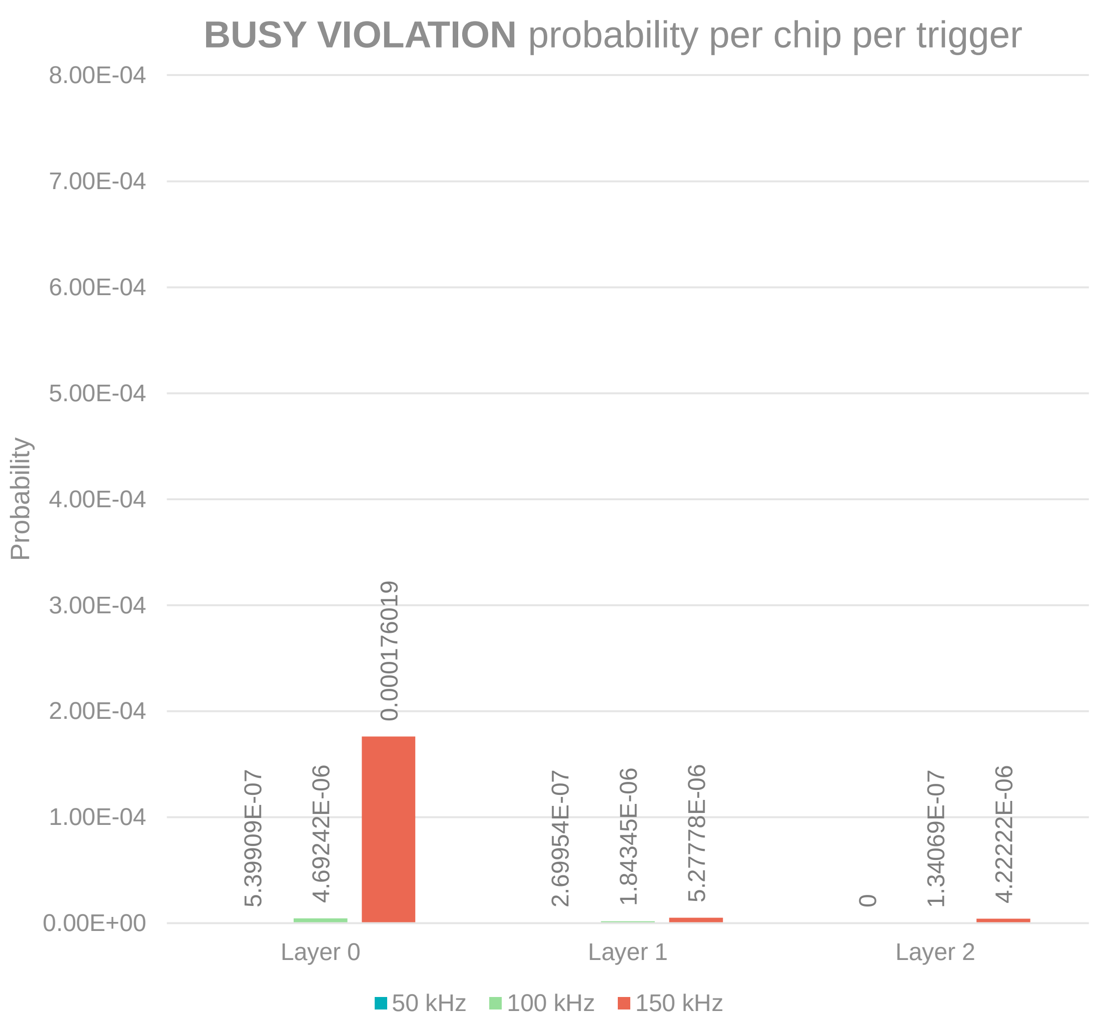
Busy link counter, per layer, per trigger, 100kHz triggered



Busy link counter, per layer, per trigger, 150kHz triggered

PbPb simulations (triggered) – Results





High event-rate pp simulations

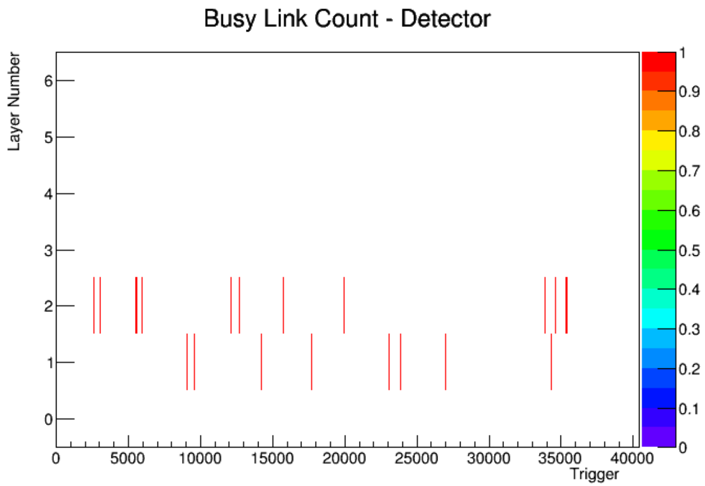
There is interest in probing production of rare heavy nuclei, such as helium/anti-helium, in proton-proton collisions

- To have a realistic chance of observing this, ALICE needs to run at higher interaction rates, maybe over 1MHz
- ALICE upgrade is designed for 200 kHz pp, ITS for 400 kHz.
- To investigate the viability of the upgraded ITS at interaction rates up to 5 MHz, simulations were run using the SystemC model

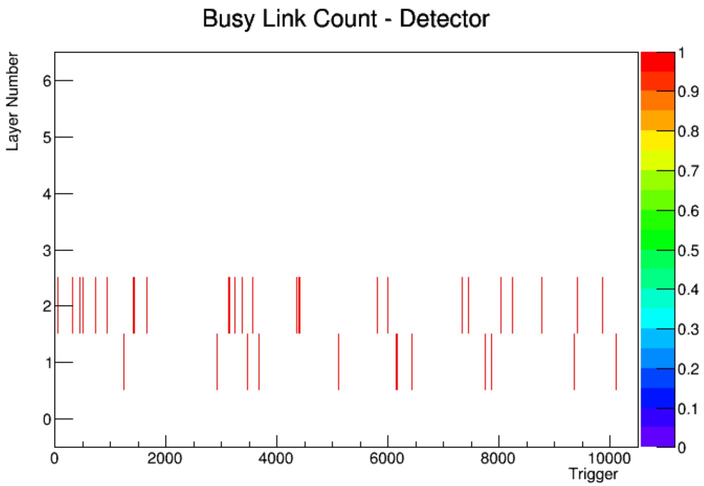
High event-rate pp simulations – SystemC simulation setup

- Inner Barrel simulated
- Event rates: 500kHz, 750kHz, 1MHz, 1.25MHz, 1.5MHz, 1.75MHz, 2MHz, and 5MHz
- 100,000 events for each simulation
- MC event pool: 10,000 min-bias pp events (Pythia)
- Continuous mode
  - Strobe length: 4900 ns
  - Strobe gap: 100 ns
- Pixel dead time: 200 ns
- Pixel active time: 6000 ns
- Minimum busy cycles: **160** (4000 ns)

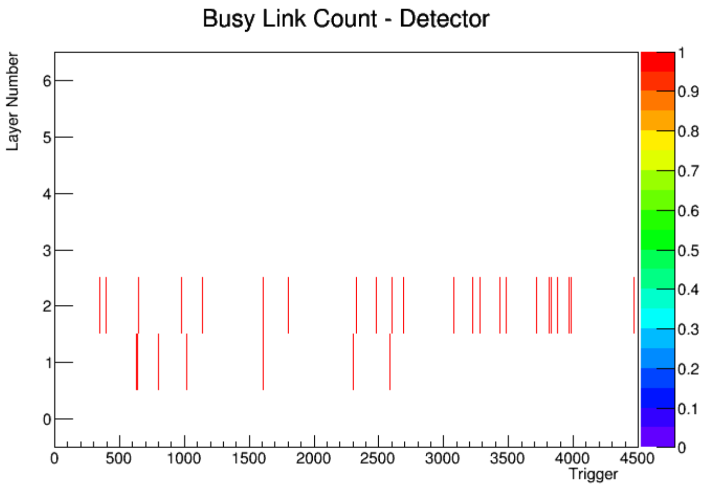
High event-rate continuous pp simulations – Results



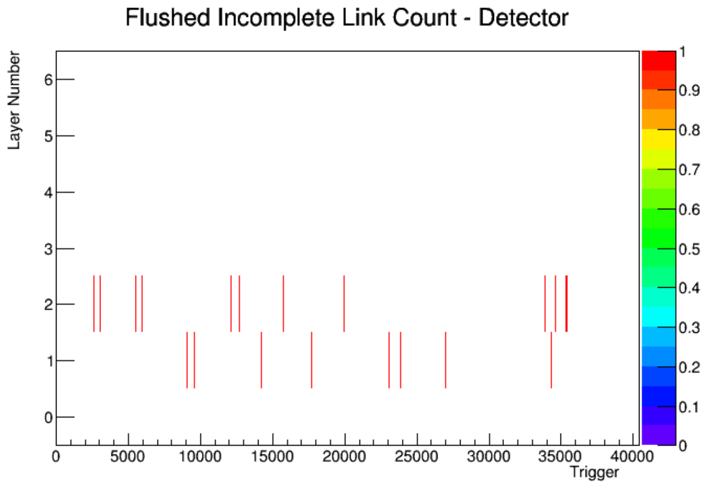
Busy link count, per layer, per trigger, 500kHz continuous



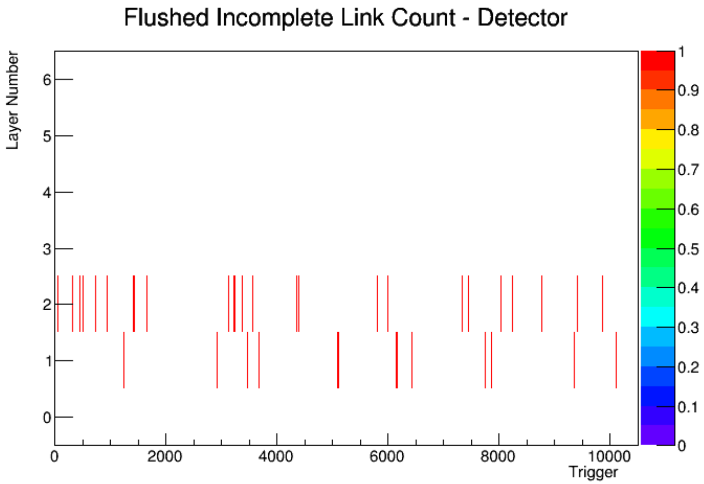
Busy link count, per layer, per trigger, 2000kHz continuous



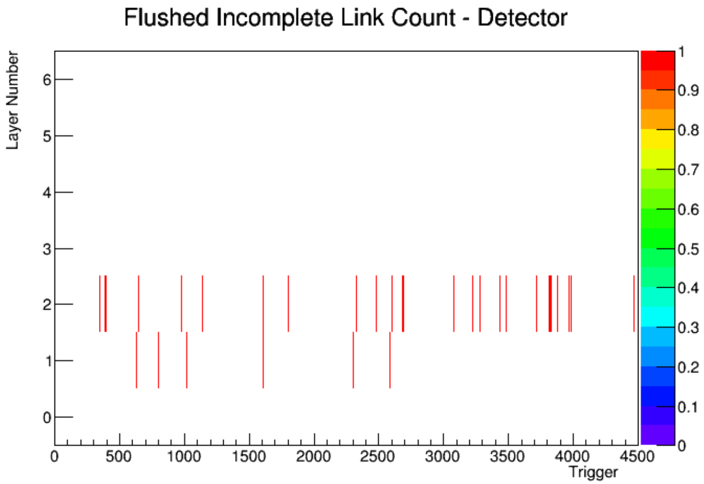
Busy link count, per layer, per trigger, 5000kHz continuous



Flush incompl count, per layer, per trigger, 500kHz continuous



Flush incompl count, per layer, per trigger, 2000kHz continuous



Flush incompl count, per layer, per trigger, 5000kHz continuous

High event-rate continuous pp simulations – Results

- The busies are due to 3 “monster” events with a lot of hits for 3 specific chips in layer 1 and 2
- Each of these seems to lead to a flush incomplete
- We had a pool of 10,000 MC events, and simulated 100,000 interaction events
- On average each of these “monster” events would be included 10 times
- 100,000 interaction events simulated regardless of event rate – that’s why we see the same amount of busy regardless of event rate
- Probably due to secondary interactions in the chips
  - More likely in layers that are further out?

Conclusions from preliminary simulations

- Not critical for nominal event rates
  - Busy violation count below acceptance level
  - No busy handling, or something very simple, is probably sufficient
- OB not simulated, and noise not included (on the order of 1E-8)

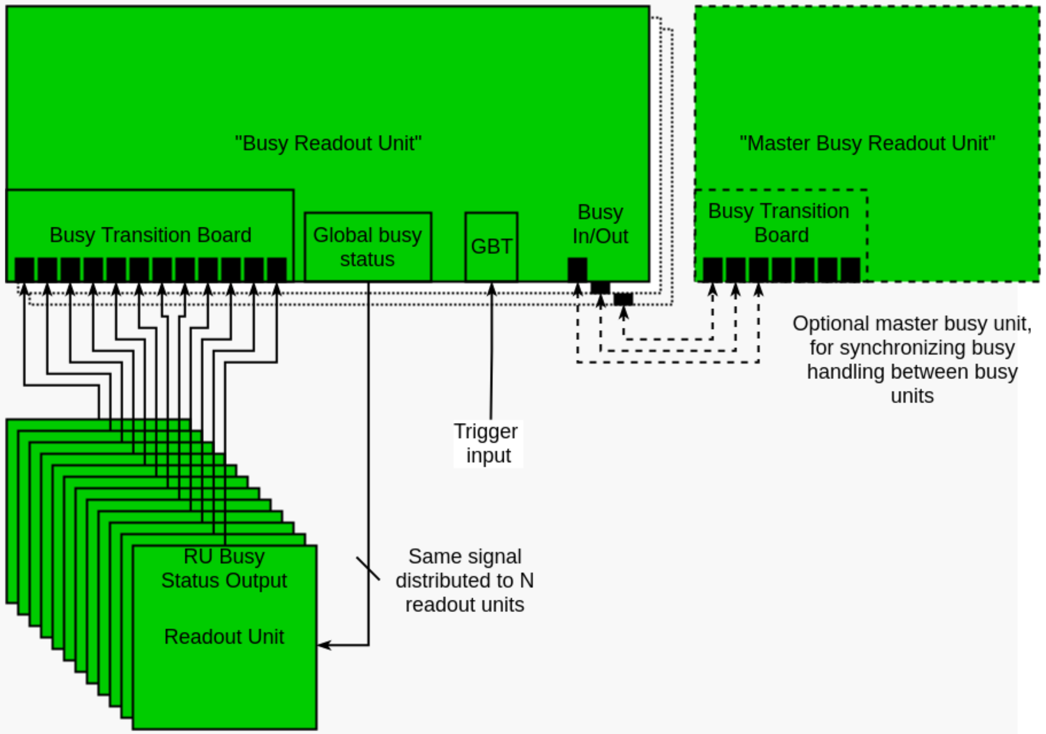
pp simulations at high event rate

- Appears that ALPIDEs can sustain high event rate pp in continuous mode
  - Whether it is possible to reconstruct events from this is a different question
- The only busies in these pp simulations come from those 3 events with secondary interactions
  - Not due to high occupancy
- A flush in one link now and then has little impact on the overall readout efficiency of the detector
- A busy subsystem that mainly just monitors busy/busyv/flush/etc seems sufficient in pp

Busy subsystem conceptional idea

- Only action at our disposal is to not send the next trigger/strobe if we want to avoid BUSYV/FLUSH
- Count number of busy ALPIDE data links in each RU
- Global busy status/action based on threshold of busy links
  - Skip next trigger/strobe if over threshold
- Have a modified RU configured as a «busy module»:
  - Use a new transition board to accept direct busy input from several RUs
  - Star topology → Little/no delay, no synchronization issues
  - Reuse RU design
  - Easier firmware design

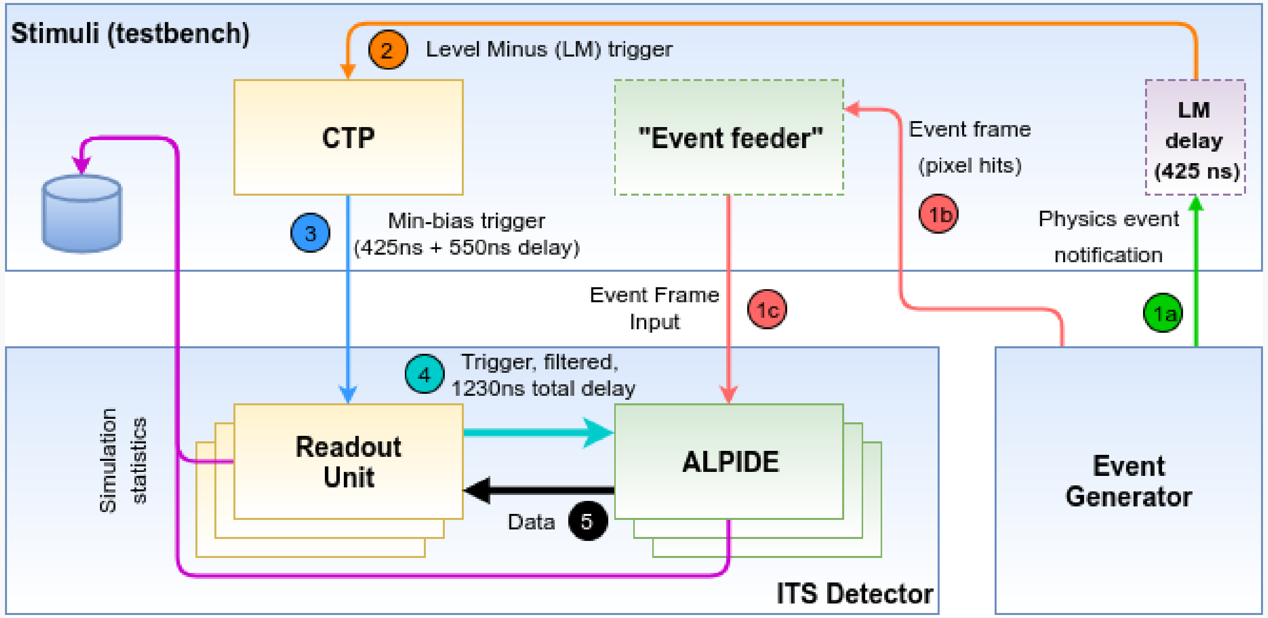
Busy board based on existing Readout Unit



Backup

Introduction

- Simulation model of readout chain in Alpide chip
- Initially intended for busy simulations only
- More general purpose at this point



Accuracy of model

- Aims to be a relatively accurate model, close to a 1:1 copy of real chip in SystemC
- Top Readout Unit (TRU) and Region Readout Unit (RRU)
  - Full model of FSMs, based on diagrams from Alpide EDR presentation
- Custom clustering method in C++/SystemC, based on interpretation of Alpide manual and data format
- Frame ReadOut and Management Unit (FROMU)
  - No direct counterpart to FROMU in code, but similar functionality implemented
- Data Management Unit (DMU)
  - Currently no direct counterpart to DMU
- There is a 4-word deep FIFO representing DMU FIFO
- Data Transfer Unit (DTU)
  - No DTU in the model. Serializing and encoding not necessary for our purposes (busy simulations)
  - DTU «implemented» with a dummy delay

Minimum busy time

- There is a setting for “minimum busy width” in the ALPIDE chip:
  - Register **0x001B - Minimum Busy Width**
  - Number of clock cycles the chip has to be in busy state internally before it asserts BUSY_ON
  - Defaults to 8 clock cycles (ie. 200 ns)

- In triggered mode, the time the busy subsystem has to make a decision is:
  - trigger_filtering_time – minimum_busy_time = 1230 ns – 200 ns = 1030 ns

- In continuous mode, with 5 us strobe, the time is:
  - 5000 ns – 200 ns = 4800 ns

- If the busy subsystem is designed to make an action in 1 us, it makes sense to increase minimum busy time in continuous mode. With 160 clock cycles (4000 ns):
  - 5000 ns – 4000 ns = 1000 ns

PbPb simulations – Event generation

A pool of discrete MC events for PbPb and QED were generated using the itsuTestBench in AliRoot:

- Using Hijing for random particle generation, and GEANT4 for tracking and detector response
  - Random events were picked from the pool using a uniform distribution, and fed to the detector in SystemC
  - Random time between events is picked from an exponential distribution, with lambda = 1 / avg_event_rate (time)
- QED events generated with 250 ns integration time
  - Luminosity: 6E27, Integration time: 250 ns
  - QED event as input to the chips continuously at 250 ns intervals, independent of triggers and interaction events

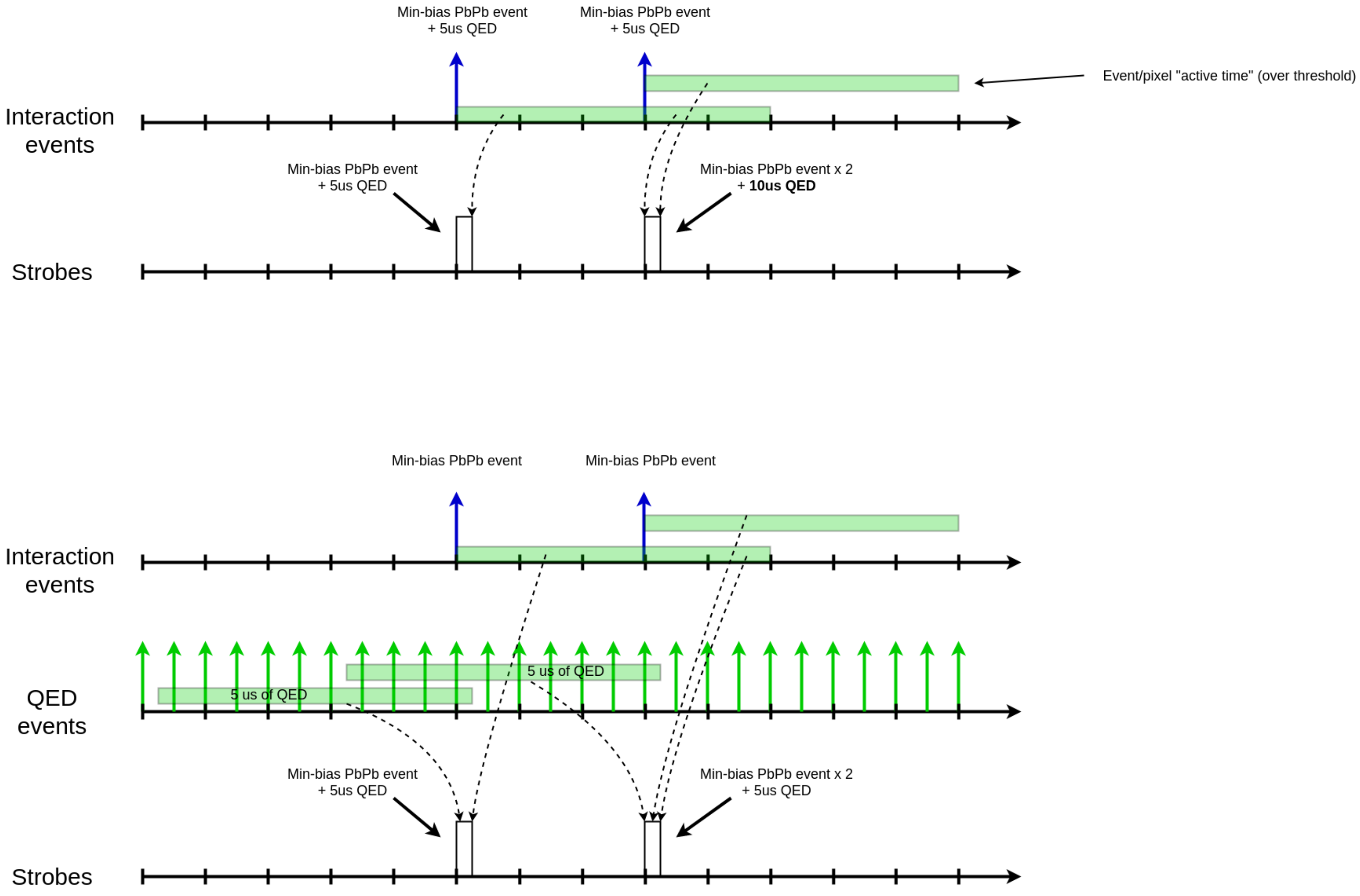
PbPb simulations – QED hits

Triggered mode

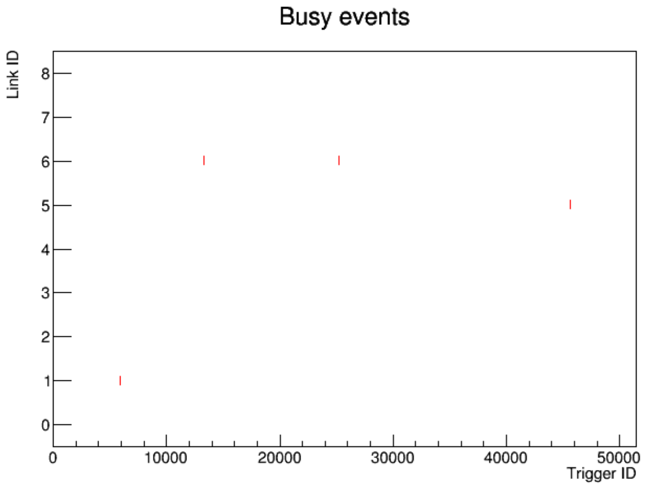
Pulse shaping time (before threshold) and trigger delay left out for simplicity

Bundling 5 us of QED with each PbPb event leads to overestimation of QED background?

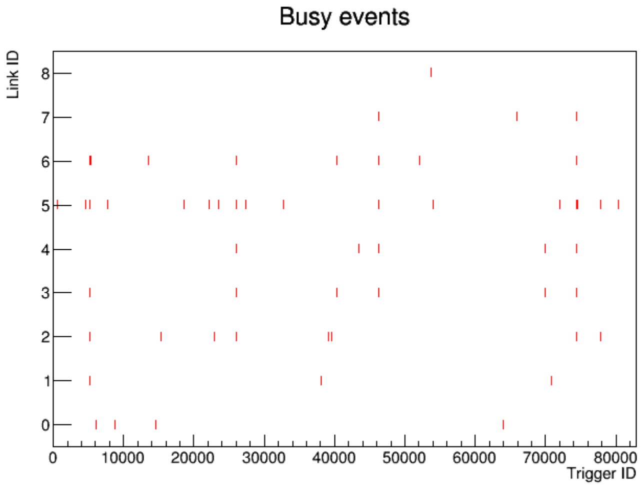
Separating them underestimates it at higher event rates (QED depends on luminosity?)



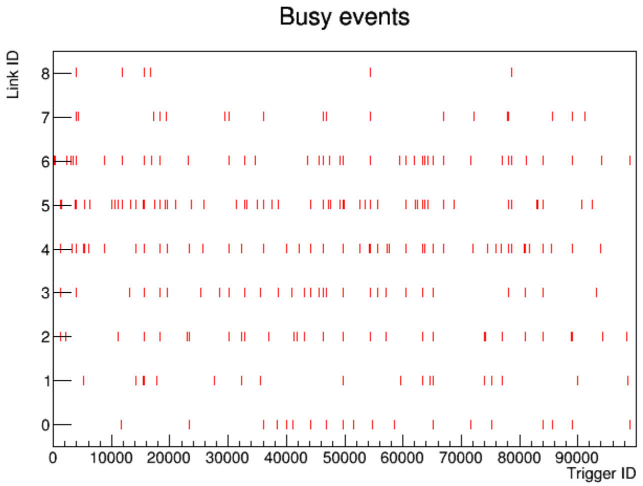
PbPb simulations (triggered) – Results



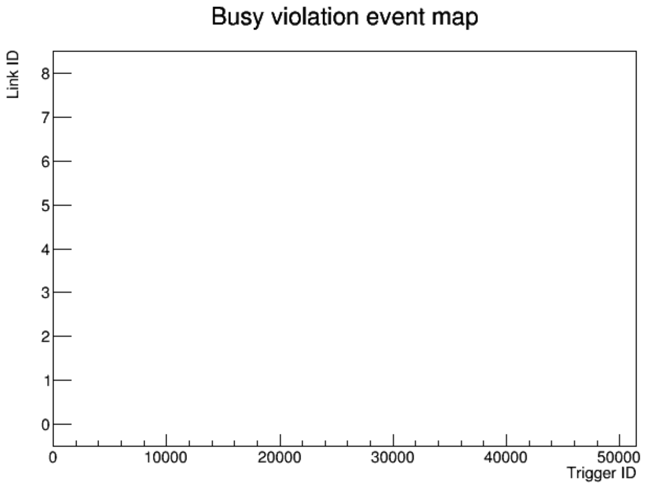
Busy map, RU 0 in layer 0, 50kHz triggered



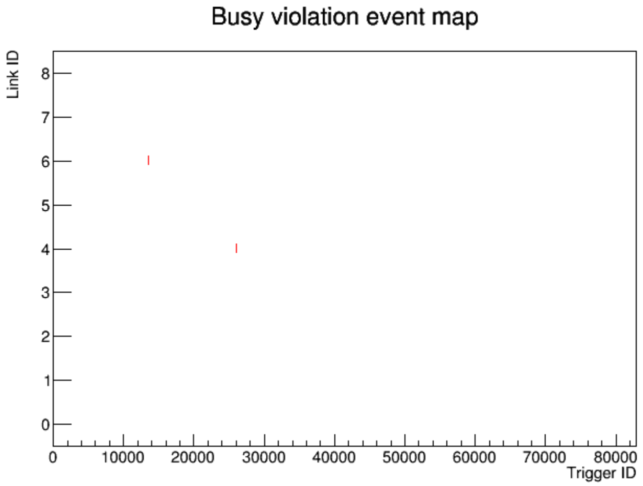
Busy map, RU 0 in layer 0, 100kHz triggered



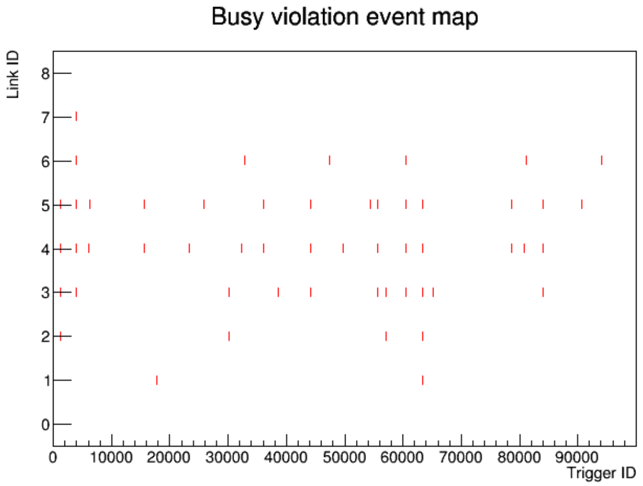
Busy map, RU 0 in layer 0, 150kHz triggered



Busy violation map, RU 0 in layer 0, 50kHz triggered



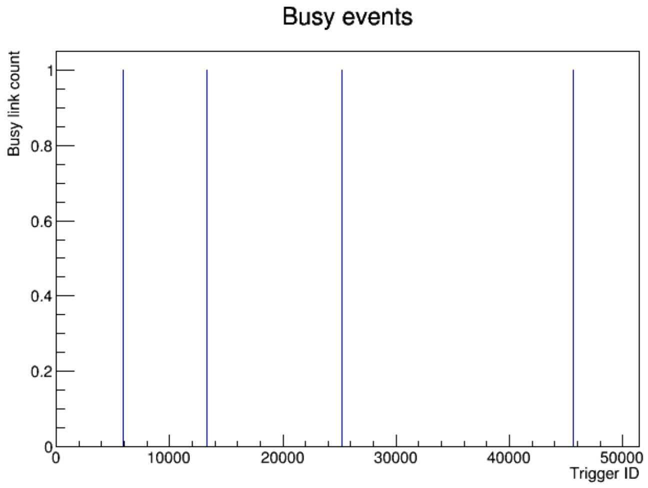
Busy violation map, RU 0 in layer 0, 100kHz triggered



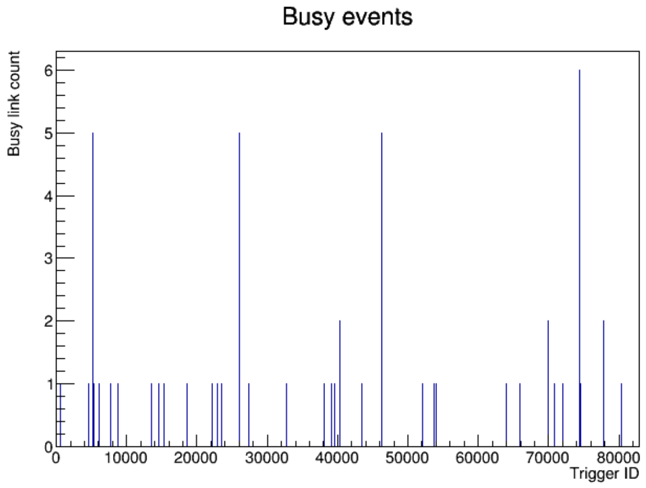
Busy violation map, RU 0 in layer 0, 150kHz triggered

PbPb simulations (triggered) – Results

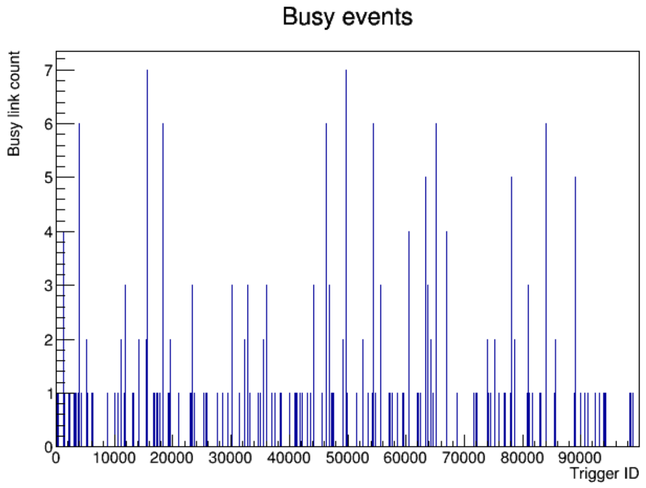
Projections of plots on the previous page, showing total counts instead of mapped per link



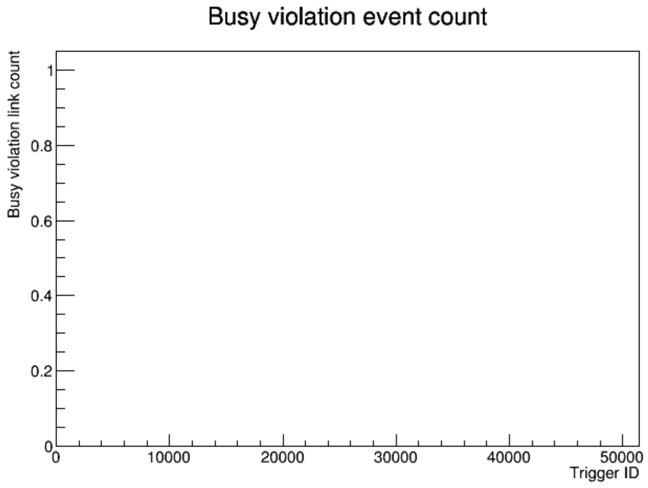
Busy count per trigger, RU 0 in layer 0, 50kHz triggered



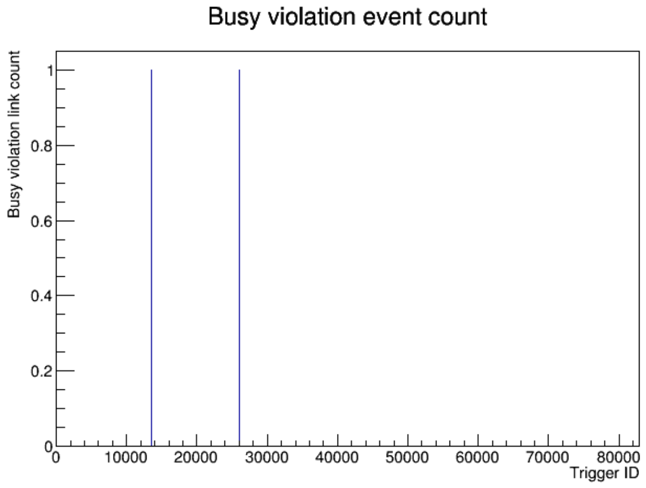
Busy count per trigger, RU 0 in layer 0, 100kHz triggered



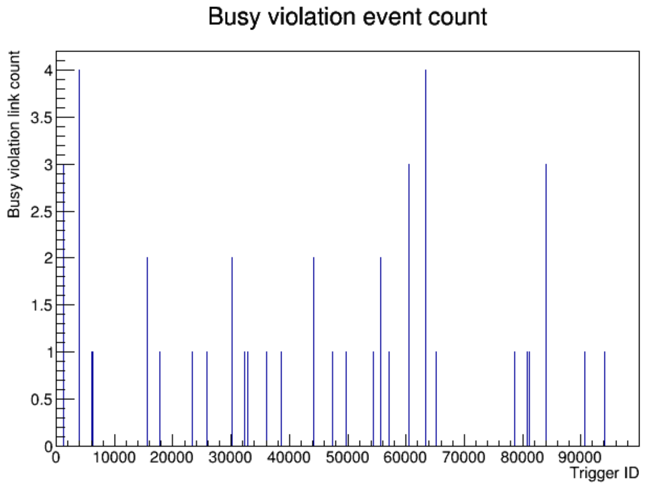
Busy count per trigger, RU 0 in layer 0, 150kHz triggered



Busy violation count per trigger, RU 0 in layer 0, 50kHz triggered



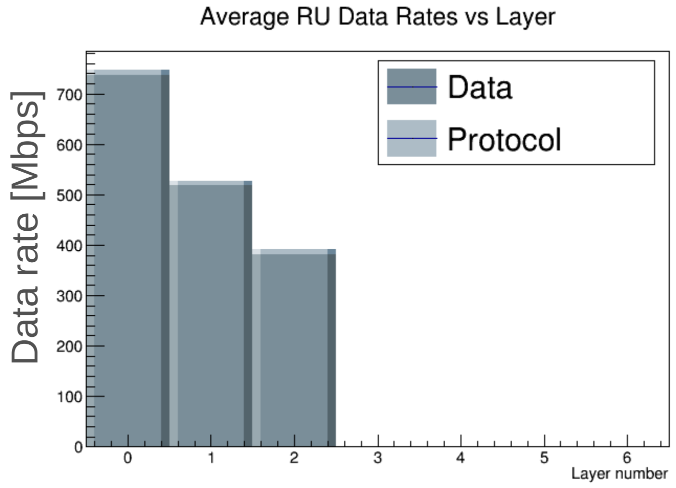
Busy violation count per trigger, RU 0 in layer 0, 100kHz triggered



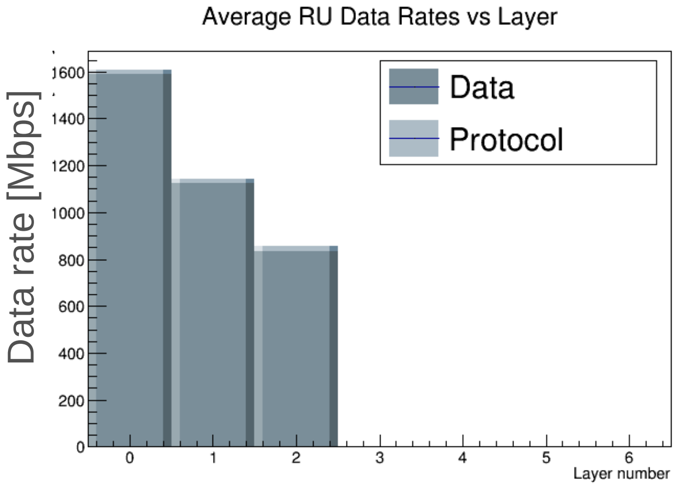
Busy violation count per trigger, RU 0 in layer 0, 150kHz triggered

PbPb simulations (triggered) – Results

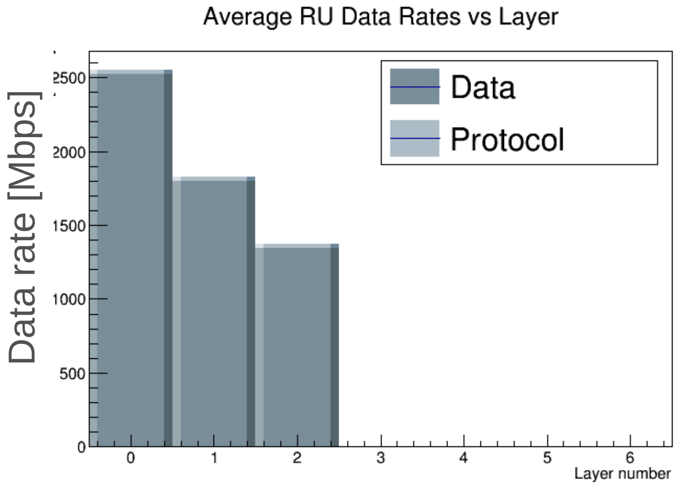
Note to bottom plots:
Not directly comparable,
different amount of events simulated!



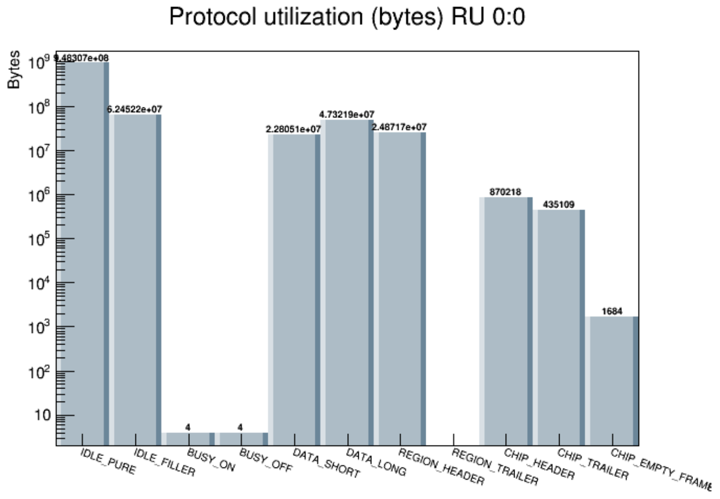
Average RU data rate per layer, 50kHz triggered



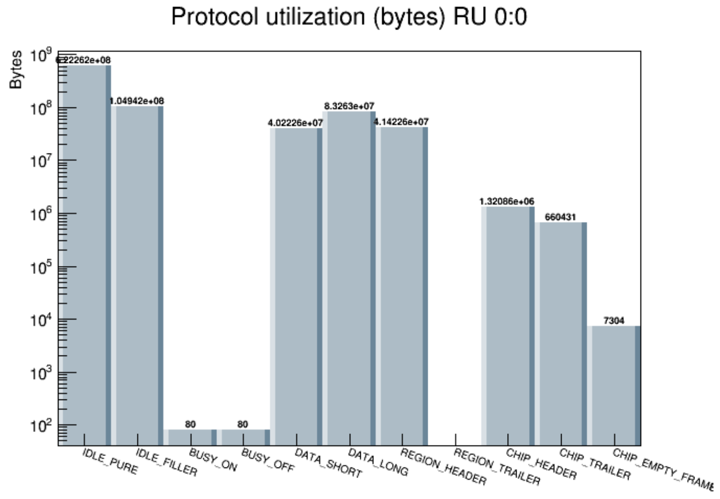
Average RU data rate per layer, 100kHz triggered



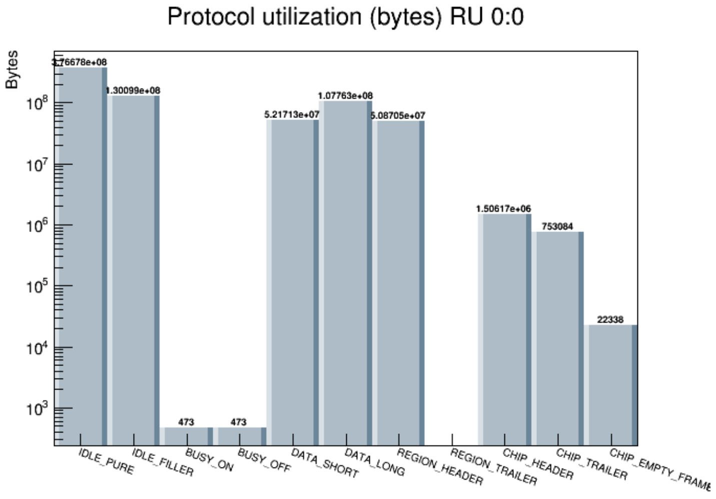
Average RU data rate per layer, 150kHz triggered



Protocol utilization in RU 0, layer 0, 50kHz triggered



Protocol utilization in RU 0, layer 0, 100kHz triggered



Protocol utilization in RU 0, layer 0, 150kHz triggered

PbPb simulations – Previous simulations

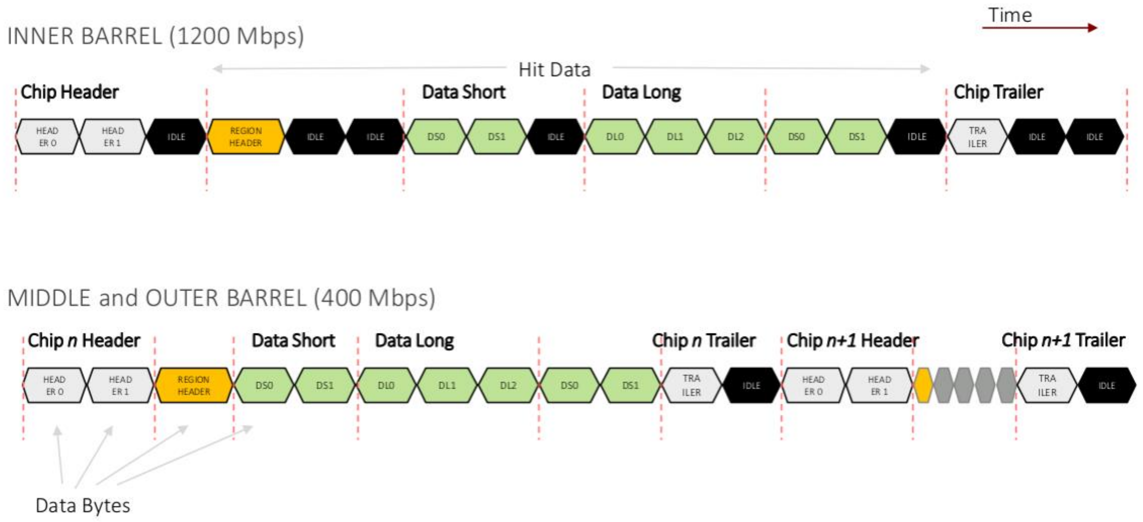
- Data rate obtained @ 100 kHz now is around 1600 Mbps, with 6 us pixel active time
- Data rate in old simulations @ 100 kHz, 5 us pixel active time, is around 2500 Mbps

- Possible reasons for discrepancy
  - Differences in simulation model
  - Differences in MC events
  - Trigger filtering (enabled at 1230 ns in my simulations)
  - Definition of what is protocol and data
    - I excluded IDLE «filler bytes»
  - Pixel noise not included
  - QED events might have been underestimated (too low luminosity for 100 kHz?)

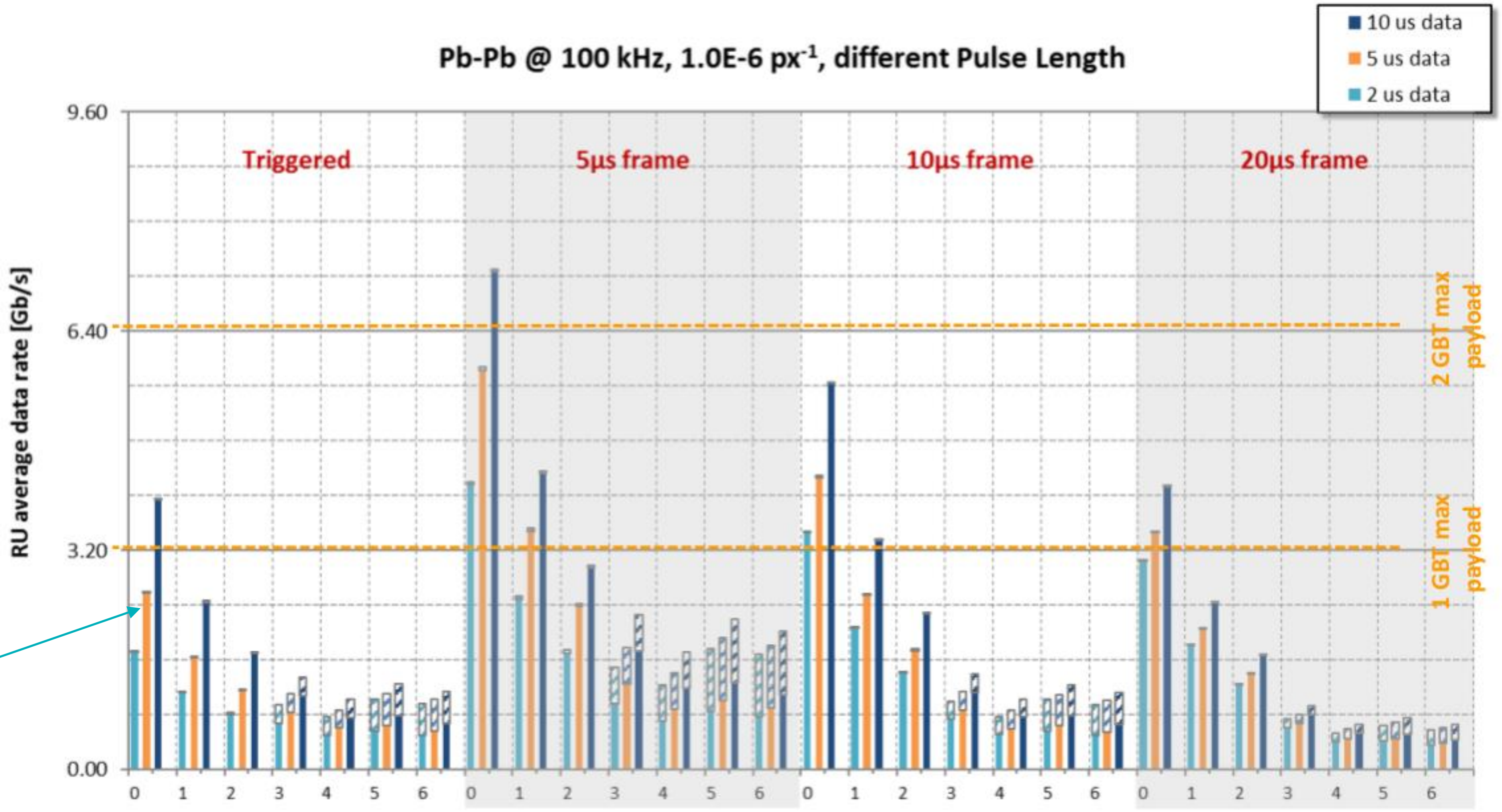
PbPb simulations – Data/Protocol definition

- Data defined as: number of bytes of REGION_HEADER, DATA_LONG, DATA_SHORT
  - IDLE «filler» bytes not included
- Protocol defined as: number of bytes of CHIP_HEADER, CHIP_TRAILER, CHIP_EMPTY_FRAME, BUSY_ON, BUSY_OFF
  - IDLE «filler» bytes not included

- Right definition?
- Should the IDLE «filler» bytes be included?

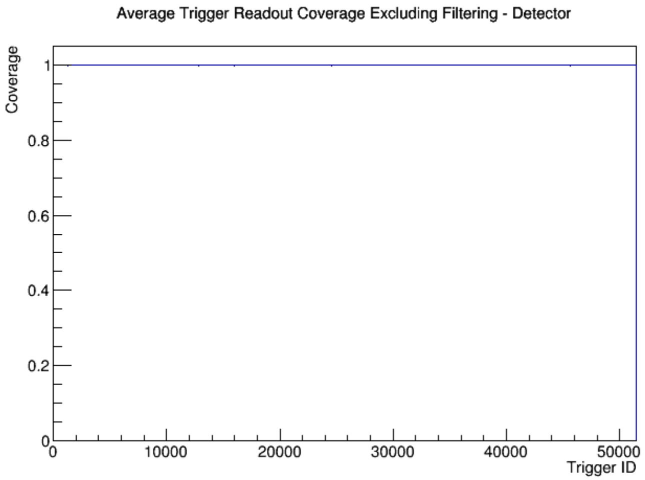


PbPb simulations – Previous simulations

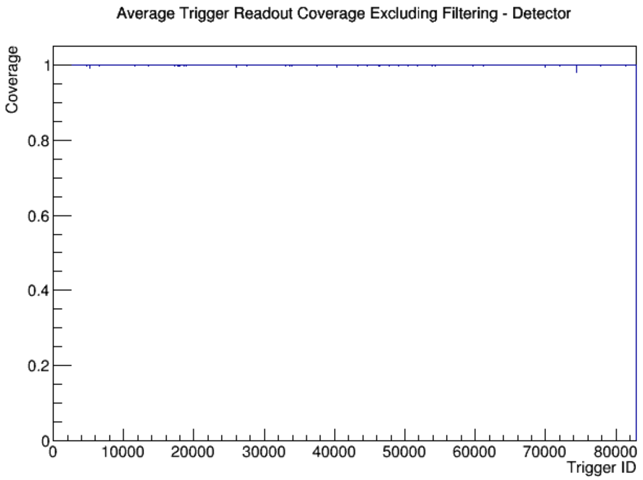


Comparable simulation

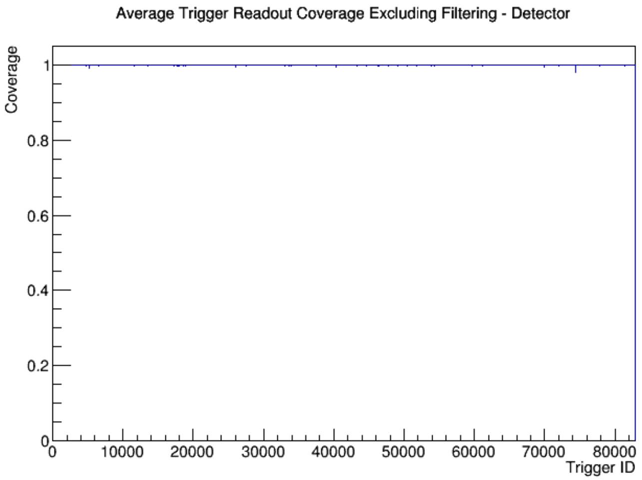
PbPb simulations (triggered) – Results



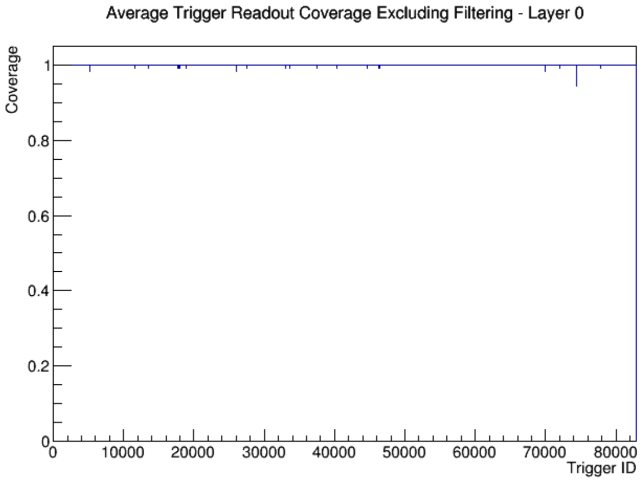
Average readout efficiency, all layers, 50kHz triggered



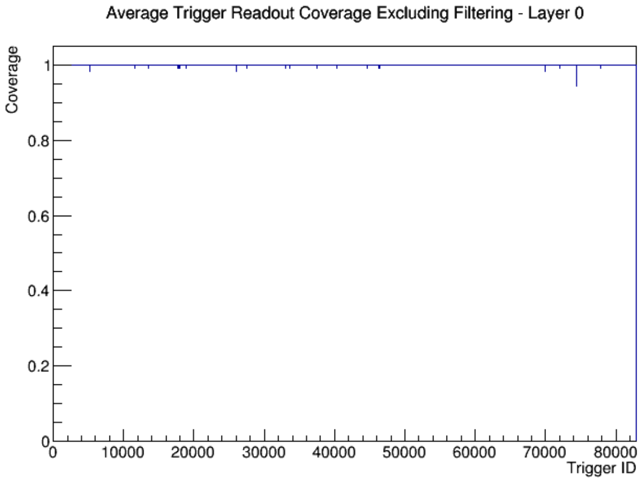
Average readout efficiency, all layers, 100kHz triggered



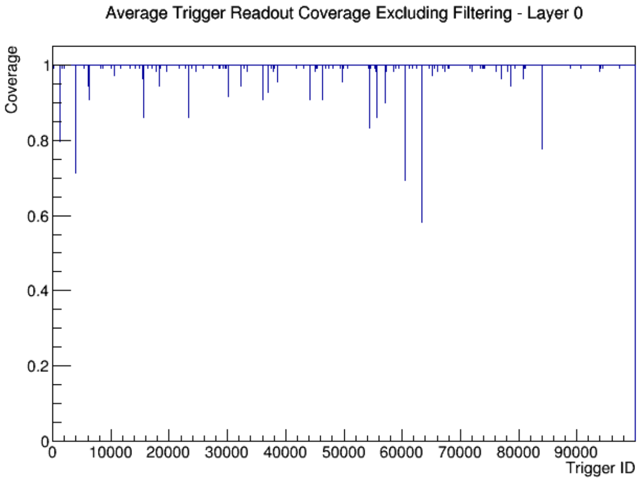
Average readout efficiency, all layers, 150kHz triggered



Average readout efficiency, layer 0, 50kHz triggered

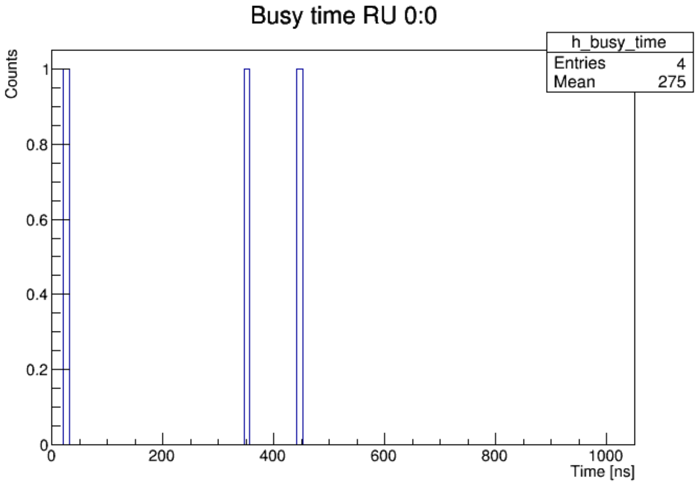


Average readout efficiency, layer 0, 100kHz triggered

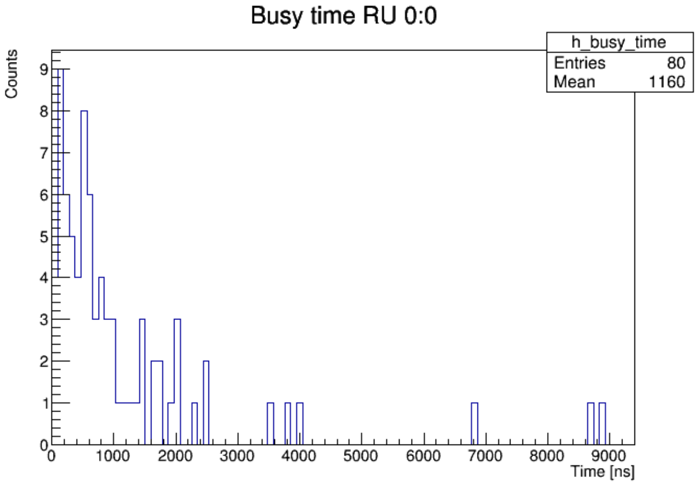


Average readout efficiency, layer 0, 150kHz triggered

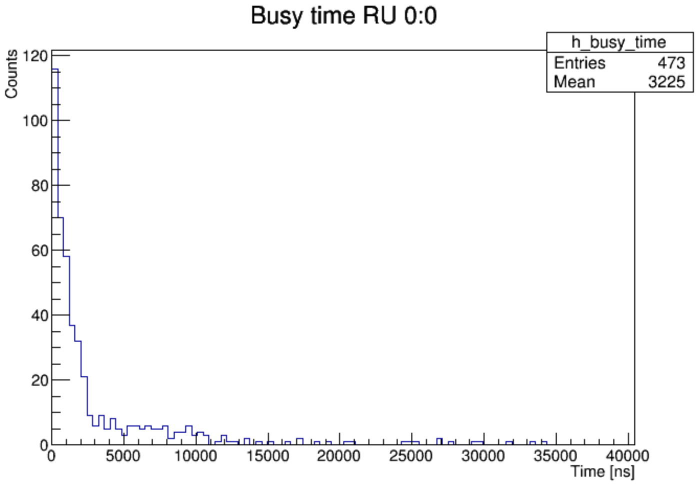
PbPb simulations (triggered) – Results



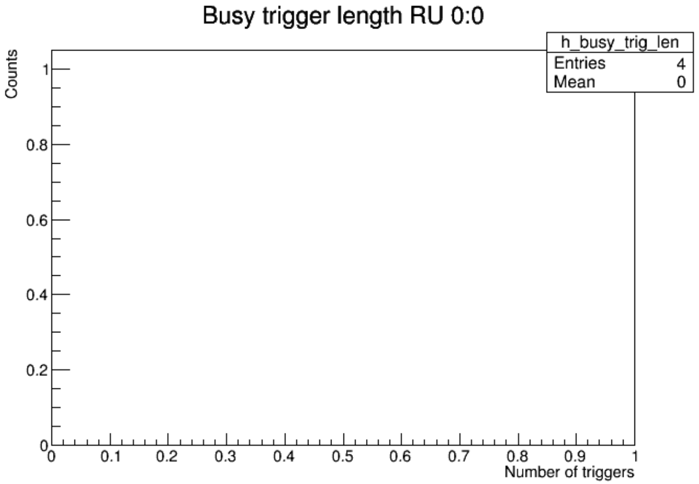
Busy durations, in time, RU 0 in layer 0, 50kHz triggered



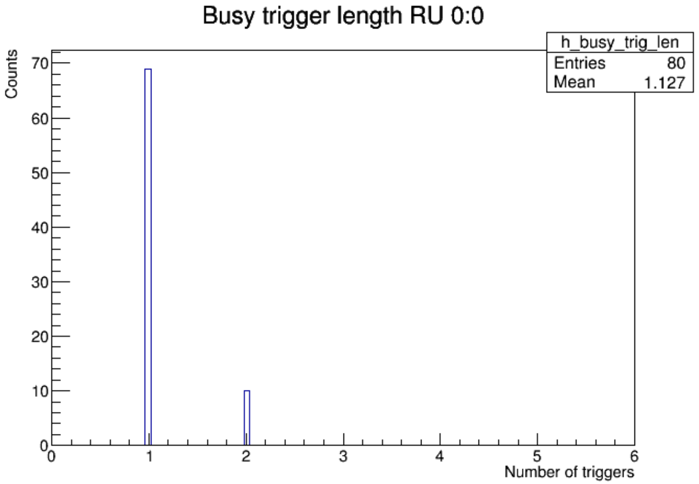
Busy durations, in time, RU 0 in layer 0, 100kHz triggered



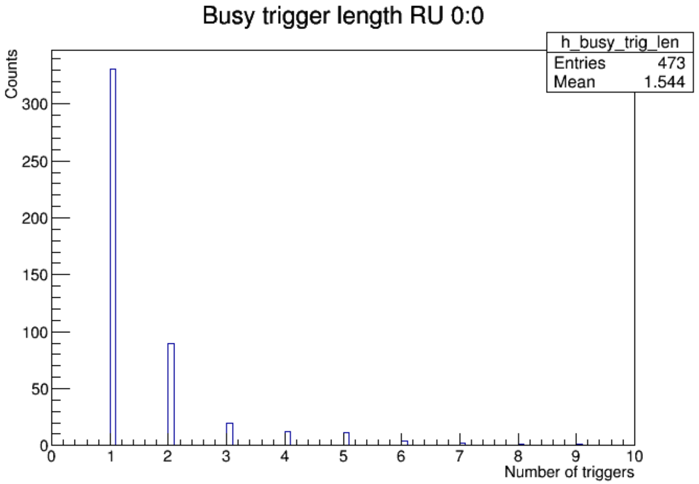
Busy durations, in time, RU 0 in layer 0, 150kHz triggered



Busy durations, in term of triggers, RU 0 in layer 0, 50kHz triggered



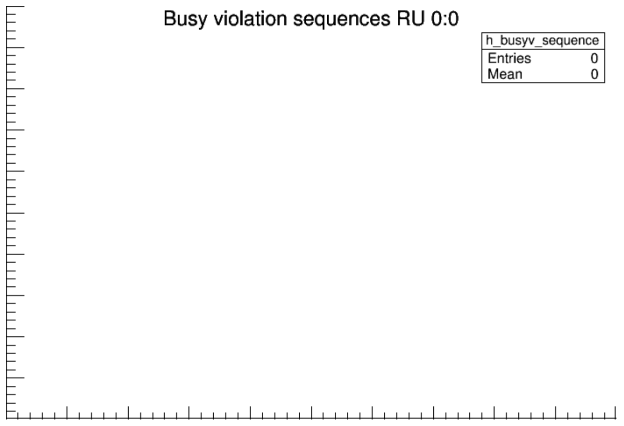
Busy durations, in term of triggers, RU 0 in layer 0, 100kHz triggered



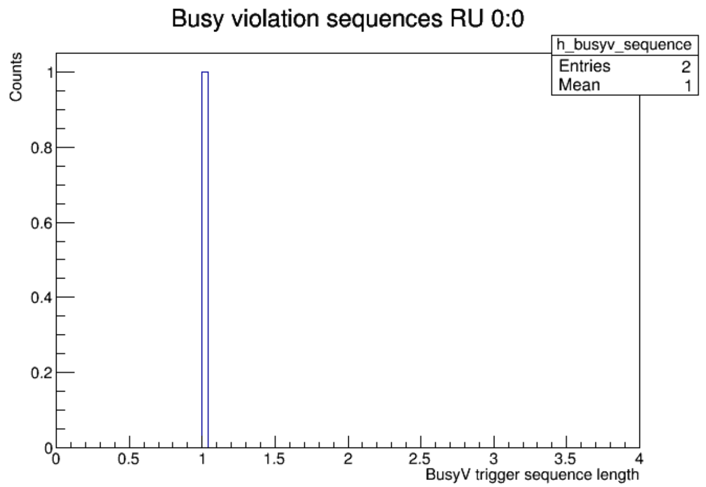
Busy durations, in term of triggers, RU 0 in layer 0, 150kHz triggered

PbPb simulations (triggered) – Results

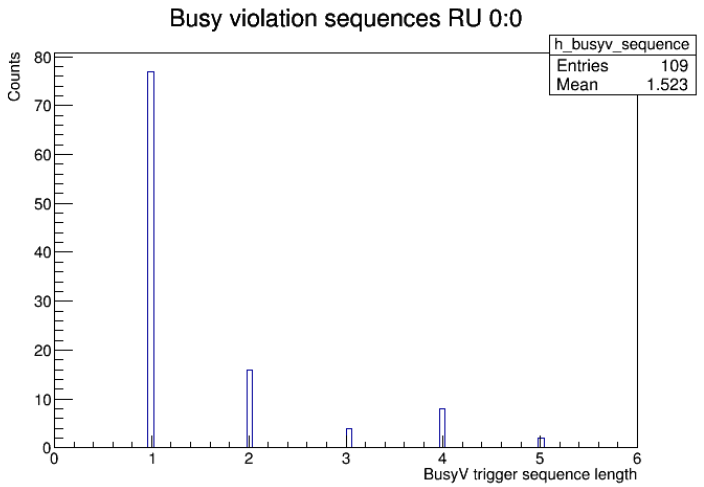
Distributions of BUSYV sequence/distance that **individual** links see, with contributions for all links. So RU would see shorter distance/longer sequence for **any** link



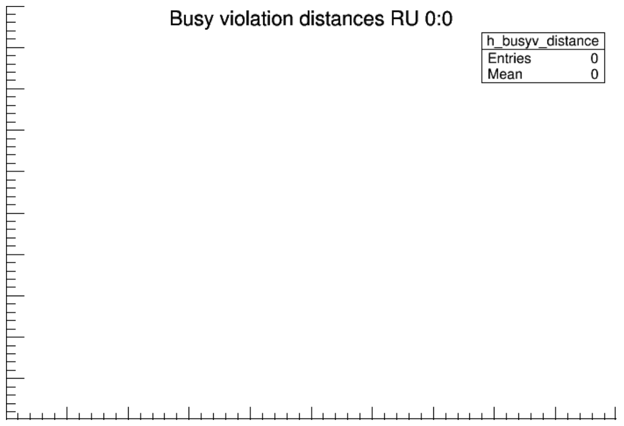
“Sequences” of busy violations, in terms of triggers, summed for links in RU 0 in layer 0, 50kHz triggered



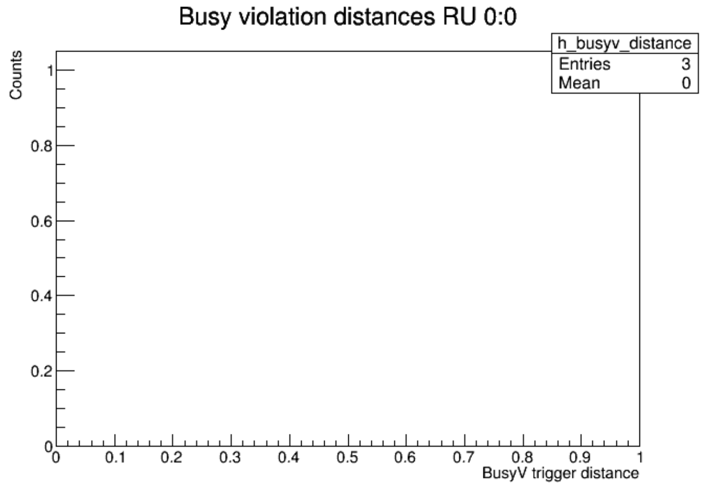
“Sequences” of busy violations, in terms of triggers, summed for links in RU 0 in layer 0, 100kHz triggered



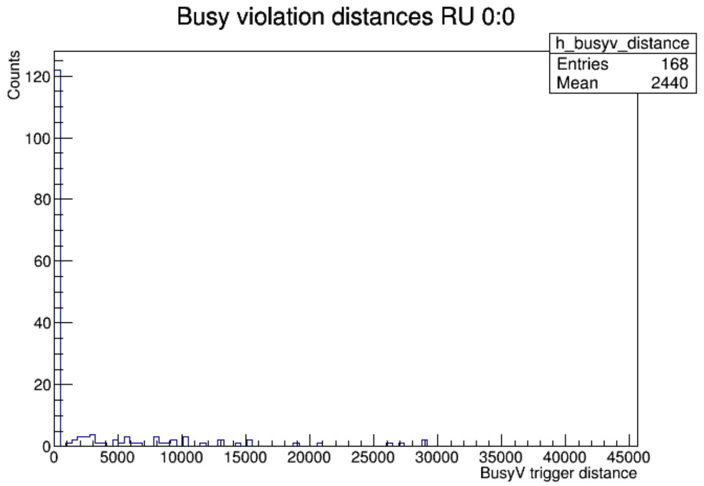
“Sequences” of busy violations, in terms of triggers, summed for links in RU 0 in layer 0, 150kHz triggered



“Distance” between busy violations for a link, in term of triggers, summed for links in RU 0 in layer 0, RU 0 in layer 0, 50kHz triggered



“Distance” between busy violations for a link, in term of triggers, summed for links in RU 0 in layer 0, RU 0 in layer 0, 100kHz triggered



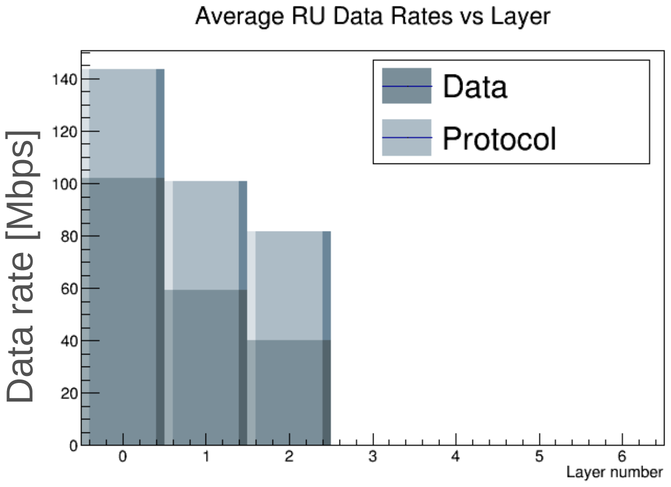
“Distance” between busy violations for a link, in term of triggers, summed for links in RU 0 in layer 0, 150kHz triggered

High-rate pp simulations – Event generation

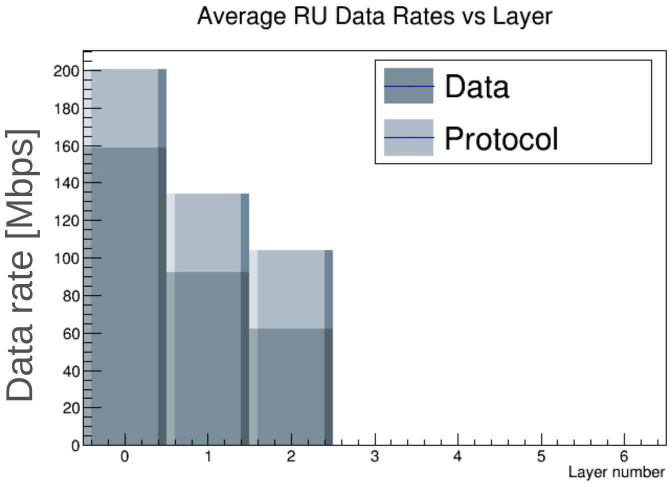
A pool of discrete MC events for proton-proton were generated using the itsuTestBench in AliRoot:

- Using Pythia for random particle generation, and GEANT4 for tracking and detector response
  - Process: kPyMb
  - Tune: 14 (kPythia8Tune_Monash2013)
  - EnergyCMS: 5500
  - Momentum range: 0 to 999999
  - pT range: 0 to 1000
  - Theta range: 0 to 180 degrees
  - Y (rapidity) range: -2.5 to 2.5
- Random events were picked from the pool using a uniform distribution, and fed to the detector in SystemC
- Random time between events is picked from an exponential distribution, with lambda = 1 / avg_event_rate (time)

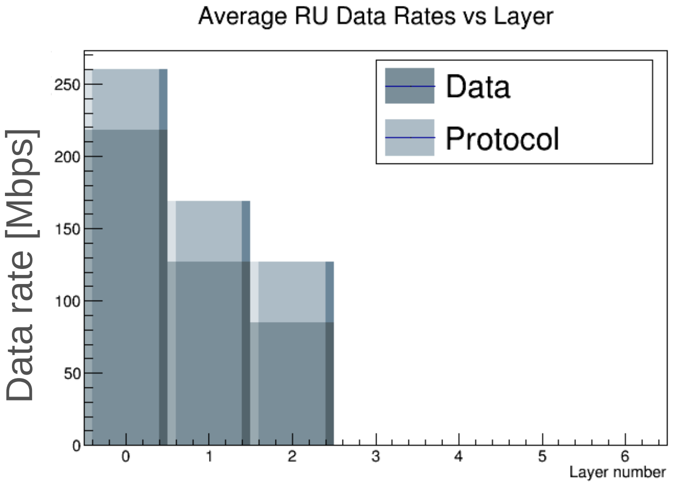
High event-rate continuous pp simulations – Results



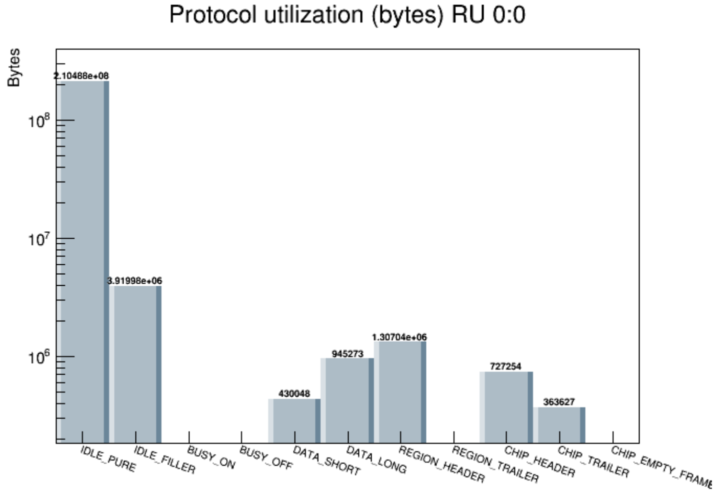
Average RU data rate per layer,
500kHz continuous



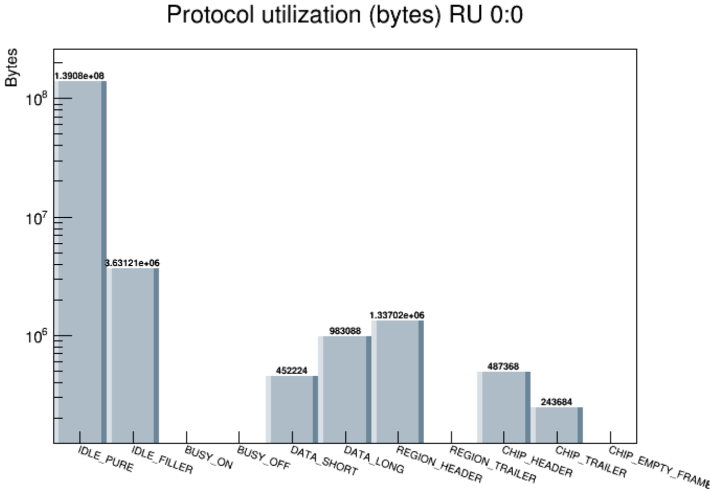
Average RU data rate per layer,
750kHz continuous



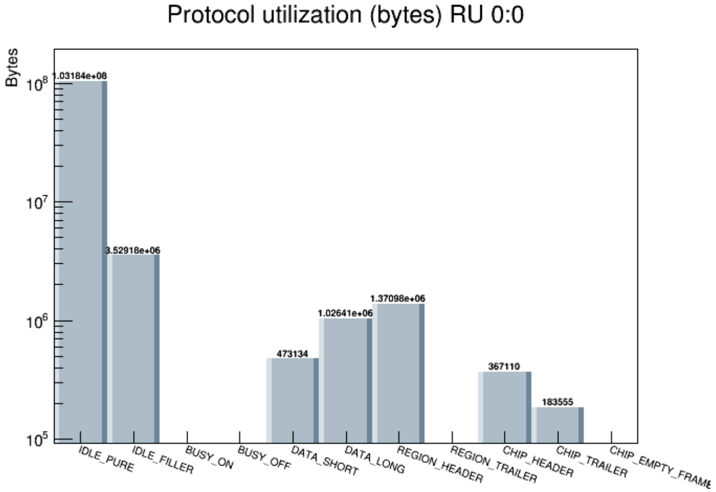
Average RU data rate per layer,
1000kHz continuous



Protocol utilization in RU 0, layer 0,
500kHz continuous

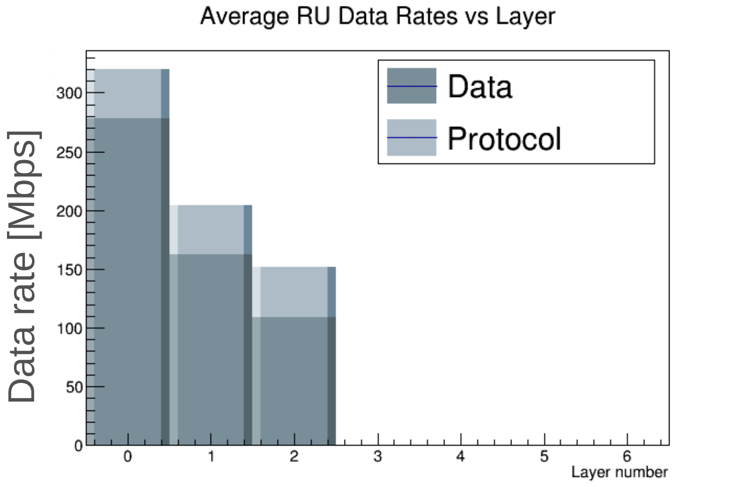


Protocol utilization in RU 0, layer 0,
750kHz continuous

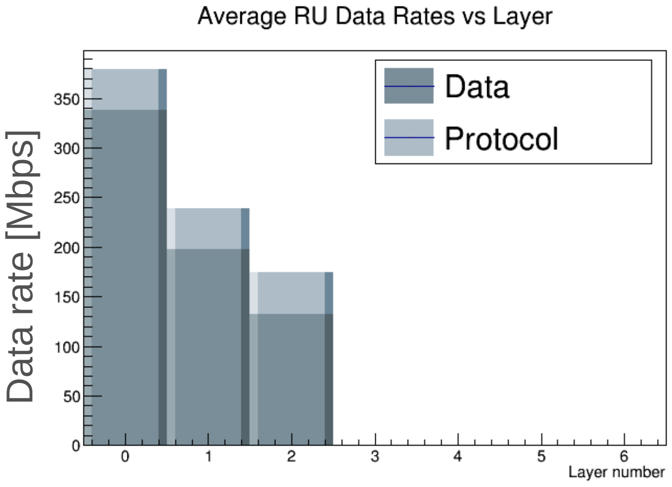


Protocol utilization in RU 0, layer 0,
1000kHz continuous

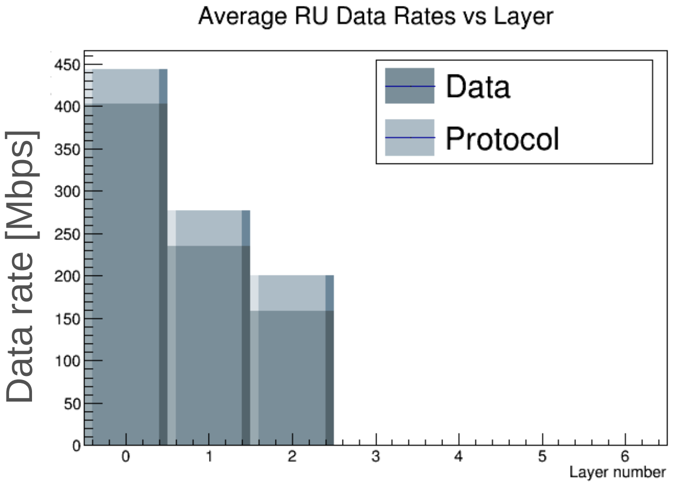
High event-rate continuous pp simulations – Results



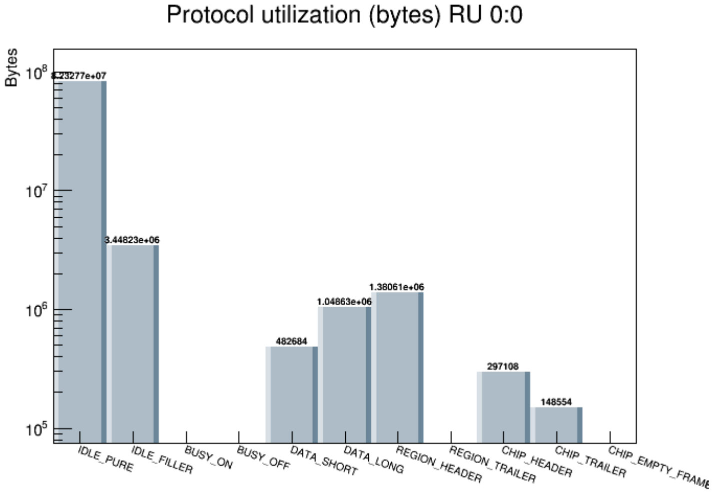
Average RU data rate per layer, 1250kHz continuous



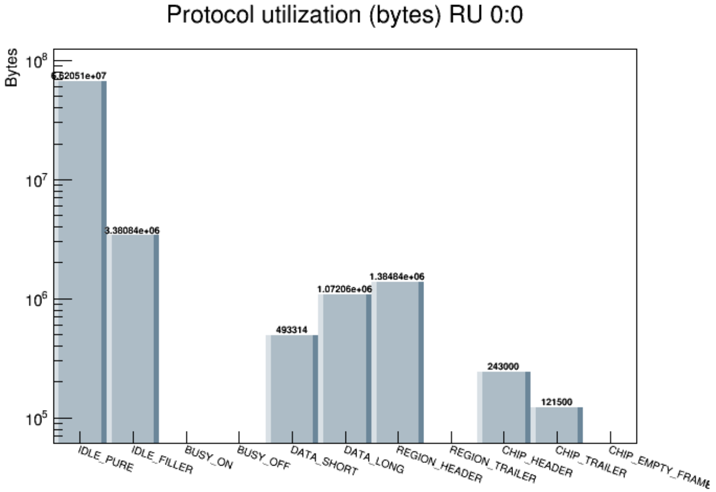
Average RU data rate per layer, 1500kHz continuous



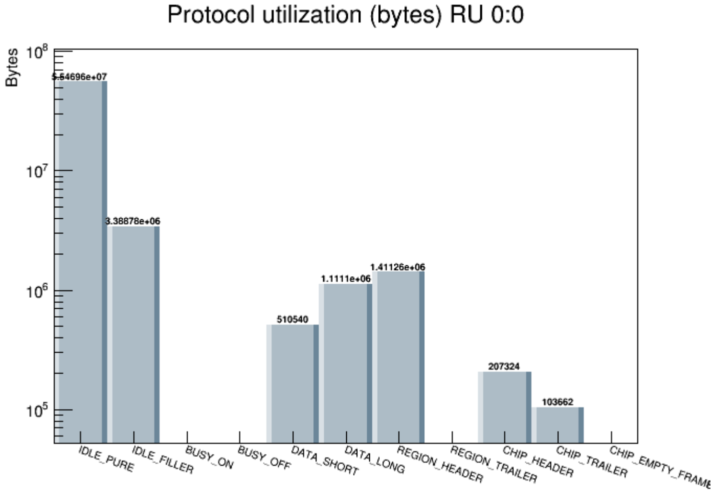
Average RU data rate per layer, 1750kHz continuous



Protocol utilization in RU 0, layer 0, 1250kHz continuous

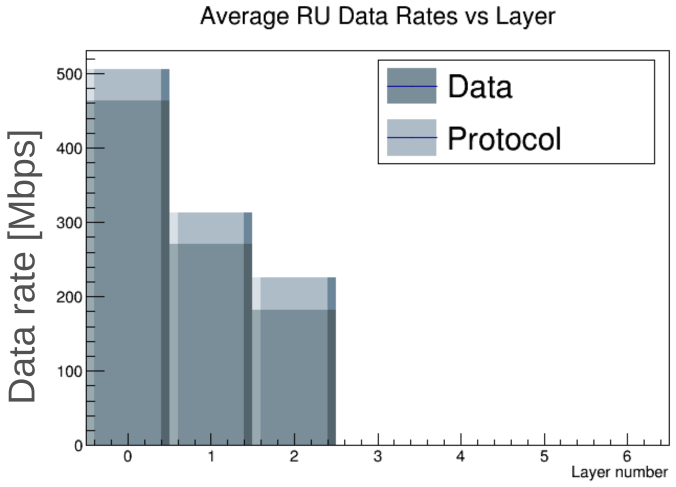


Protocol utilization in RU 0, layer 0, 1500kHz continuous

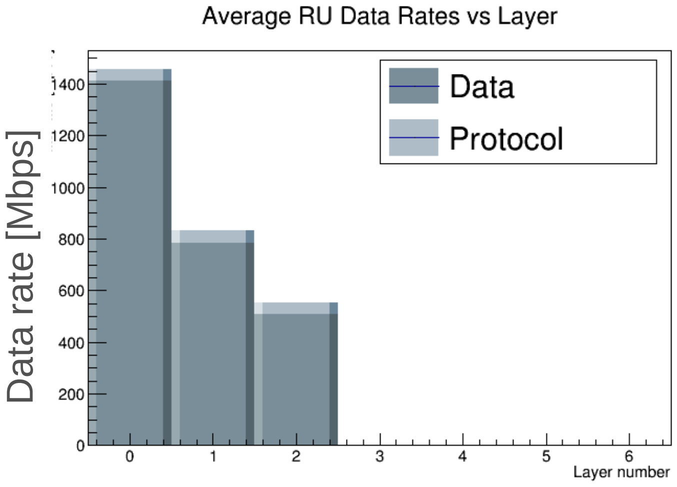


Protocol utilization in RU 0, layer 0, 1750kHz continuous

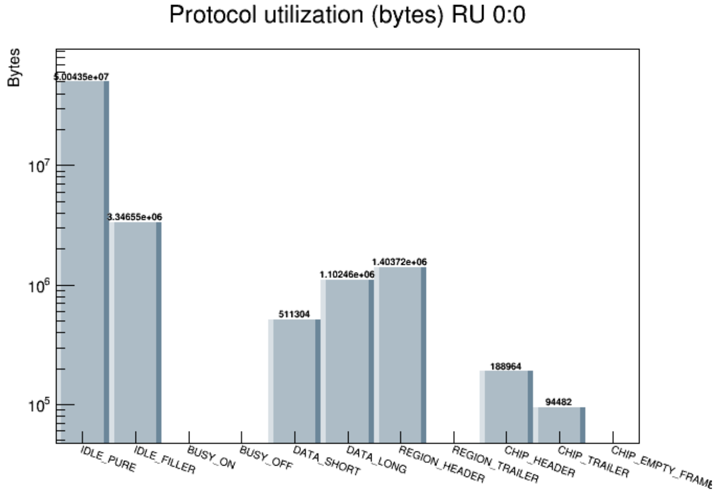
High event-rate continuous pp simulations – Results



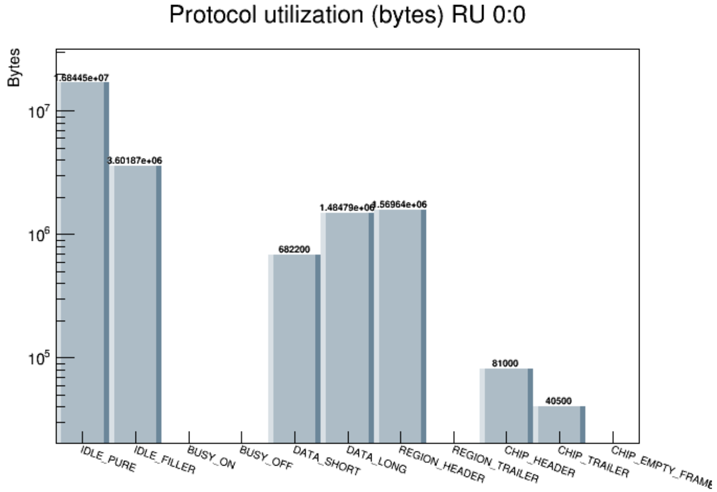
Average RU data rate per layer,
2000kHz continuous



Average RU data rate per layer,
5000kHz continuous



Protocol utilization in RU 0, layer 0,
2000kHz continuous



Protocol utilization in RU 0, layer 0,
5000kHz continuous