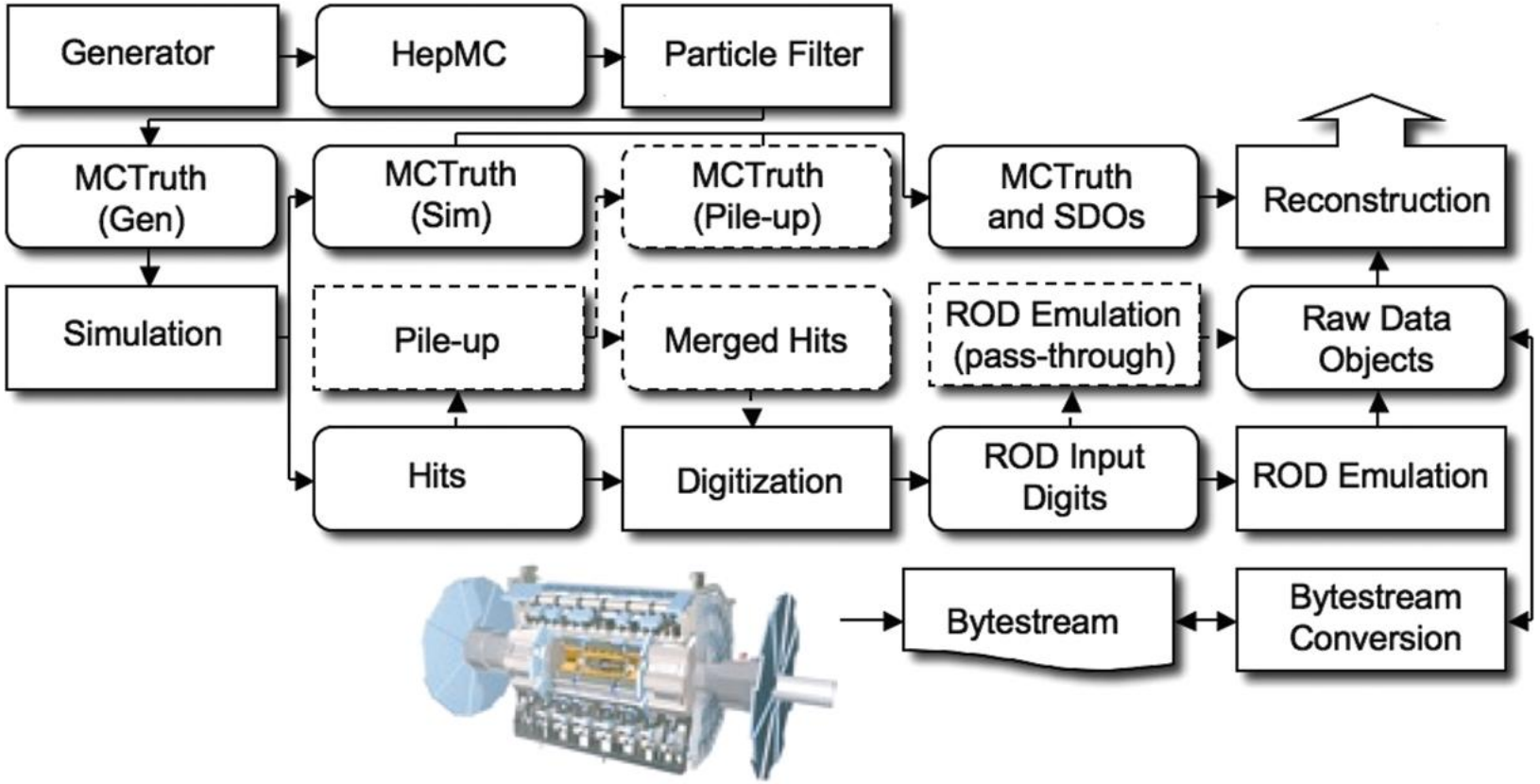
Digitization in ATLAS

"Monte Carlo in ATLAS" Tutorial
CERN, 28th September 2015

Digitization in ATLAS

- **Simulating events in ATLAS is performed as a three step process within the Athena framework:**
  - **Event Generation**
  - **Sensitive Detector Simulation**
  - **Detector Electronics Simulation (Digitization)**



What is Digitization?

- **The ATLAS Digitization software is part of the ATLAS software chain used to produce Monte Carlo events. Digitization runs after the Geant4 simulation.**
- **Digitization Input:**
  - **The energy deposits in sensitive detector volumes (HITS) from the (GEANT4) Simulation step.**

What is Digitization?

- **The ATLAS Digitization software is part of the ATLAS software chain used to produce Monte Carlo events. Digitization runs after the Geant4 simulation.**
- **Digitization Process:**
  - **The digitization software then converts these energy deposits into detector responses ("Digits"), typically voltages or times on pre-amplifier outputs.**
  - **Hence the peculiarities of each detector's charge collection, including electronics noise, and channel-dependent variations in detector response are modelled in the digitization software.**
  - **Digit creation is followed by a simulation of the RODs (Read Out Drivers) to produce RDOs (Raw Data Outputs).**

What is Digitization?

- **The ATLAS Digitization software is part of the ATLAS software chain used to produce Monte Carlo events. Digitization runs after the Geant4 simulation.**
- **Digitization Output:**
  - **RDO pool root files, containing:**
    - **RDOs**
    - **SDOs (Simulated Data Objects) are also written out to the pool file. SDOs provide a link between the RDOs and the Truth information.**

Simulated Data Objects

- **Simulated Data Objects (SDOs) are written out to the Raw Data Object (RDO) pool files (output of the Digitization step).**
- **SDOs are created by digitization algorithms for Inner Detector and Muon Spectrometer sub-detectors.**
  - **One SDO per hit channel.**
  - **Provide a link between the RDOs and the Truth information.**
  - **Contain a list of all the contributions to the response of that channel.**
- **CalibrationHits perform a similar function to the SDOs for the Calorimeters. They are only written out for certain samples.**
  - **Only produced for certain samples due to the huge size on disk.**

How do we model noise?

- **Typically noise rates are first measured in data for a particular readout technology.**
- **The average amount of noise is stored in the database. This may be an average for the whole sub-detector or for a single module/channel.**
- **In order to determine the level of noise to add to a particular channel during the digitization, the noise constant for that channel is multiplied by a Gaussian random number.**
- **In detectors with a binary read-out there is a shortcut in the case that there were no energy deposits in the module in question.**
  - **In that case the noise rate can be used to give the number of strips in a module above threshold from noise alone. This is done by throwing a Poisson number about that expected number average number of strips above threshold from noise alone. Then picking that number of channels at random from the module to be above threshold.**

How do we model Electronics?

- **Clearly this will be different for each sub-system, but the main parts are as follows:**
  - **Convert energy deposited into charge.**
  - **Drift that charge to the readout electrode.**
  - **Model any amplification of that charge.**
  - **Model any cross-talk between nearby channels.**
  - **Model any pedestals in the electronics.**
  - **Apply any thresholds (particularly in the case of binary read-outs).**
- **After this process the RDOs are written out.**

What is pile-up? (I)

- **The LHC beams are comprised of bunches of protons separated in time by at least 25ns.**
- **There are 3564 potential bunch-crossings in each orbit of the LHC beams, assuming a 25ns spacing between bunches.**
- **The exact pattern depends on how the LHC is filled. Typically the beams feature "trains" of filled bunches with larger gaps between the trains.**
  - **A "50ns run" just means that the spacing between filled bunches in each train is 50ns.**

What is pile-up? (II)

- **There are multiple pp-interactions within ATLAS in each filled LHC bunch-crossing. The average number of collisions in each filled-bunch is denoted by "mu".**
- **When talking about the level of pile-up in a run we typically talk about <mu> which is the above quantity averaged over all filled bunch-crossings in an orbit.**
- **Both mu and <mu> decay over the course of a run.**

What is pile-up? (III)

- **The prompt signal from pp collisions in the ATLAS detector is collected over only a few hundred nanoseconds.**
- **However, long after the collisions, a gas of low energy neutrons and photons is still present in the cavern. This gas is generally referred to as "cavern background."**

How do we model pile-up? (I)

- **ATLAS divides the particles from background pp-collisions into two parts:**
  - **The prompt signal from single background pp collision is simulated as a "minimum bias" event.**
    - **Minimum bias events are classified as either high pT or low pT based on the highest truth jet pT in the event. The pT cut for classification as high pT is 35 GeV.**
    - **This is to allow us to artificially boost the statistics of the high pT minimum bias sample to avoid high pT jets repeating in the dataset.**
  - **The low energy/long lived particles from this sample are dropped from the minimum bias sample simulation and simulated in a separate "cavern background" sample.**
    - **Assumed to be asynchronous, so the times of simulated hits are wrapped around modulo the mean spacing between filled bunches.**
    - **Muon detectors are most affected by high cavern-background rates.**
    - **This type of background is notoriously difficult to properly simulate, mostly due to the difficulties in correctly describing low energy neutron physics.**

How do we model pile-up? (II)

- **Modelling the Athena framework proceeds as follows:**
  - **Run the event generation and (GEANT4) simulation steps for single pp interactions.**
  - **Large low pT and high pT minimum bias samples are simulated and then used for multiple signal samples.**
  - **Multiple simulated pp interactions are combined during the digitization step ("Pile-up Digitization").**
    - **Use the HITS from many simulated pp interactions and digitizing them all together.**
    - **This includes both in-time and out-of-time pp interactions within a [-800,800] ns window.**

How do we model Pile-Up?

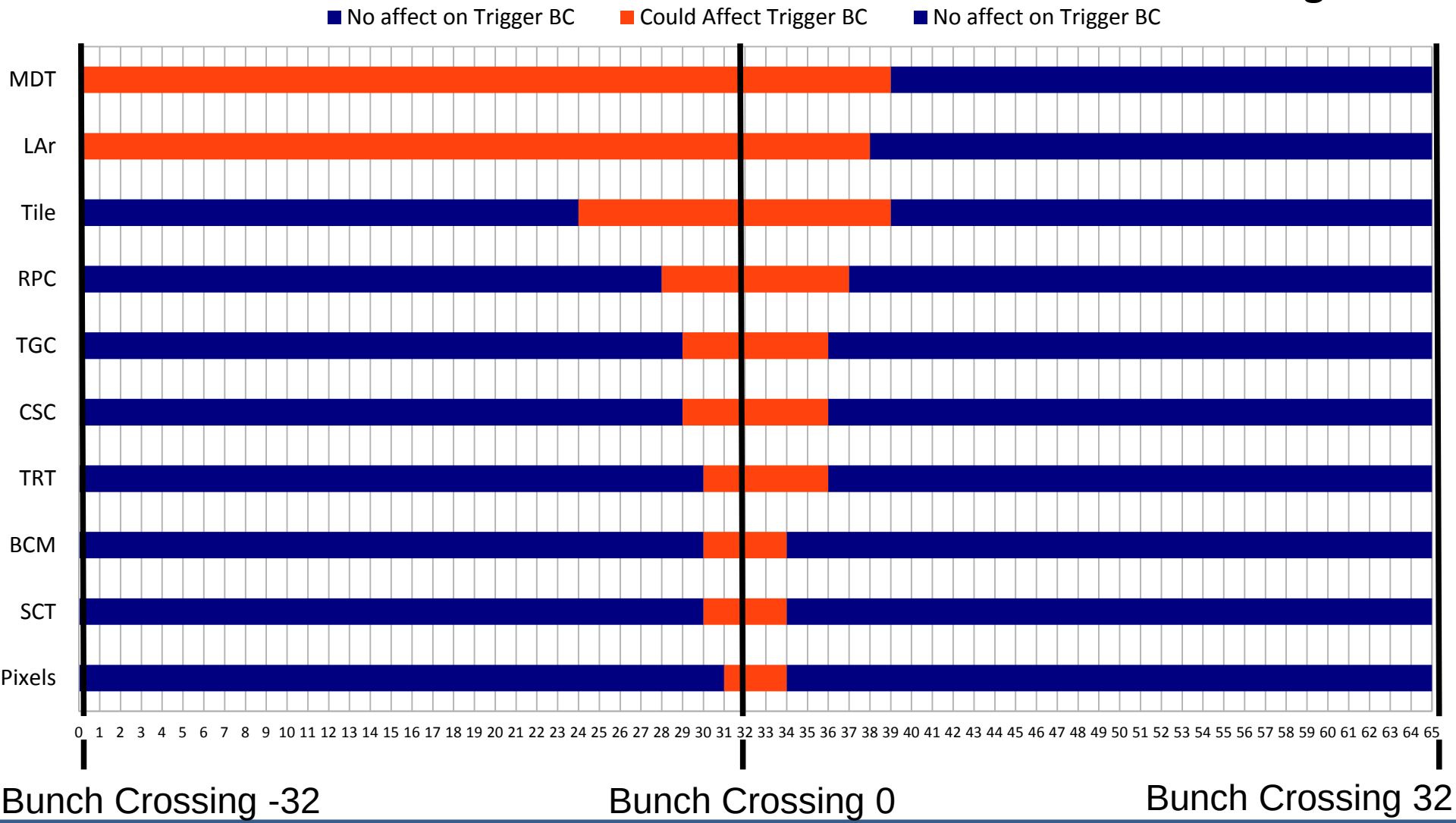
- **How background events are added to the signal event depends on the sample type:**
  - **Minimum Bias:**
    - **Specify the Poisson mean number of events to be added per filled-bunch crossing (mu).**
    - **Add a random number picked from that distribution to each filled bunch-crossing.**
  - **Cavern Background:**
    - **Constant background, therefore add a constant number to each bunch-crossing.**
    - **Separate samples for each bunch spacing to be simulated.**
- **In both cases events are then offset in time depending on the particular bunch crossing they are being used for.**

How do we model pile-up? (III)

- **Actually, it's a little bit more complicated than that…**
- **There are a few other things that we need to model:**
  - **Some sub-detectors have much shorter sensitive time windows than others. (see following slides)**
  - **Not every bunch-crossing is filled. (see following slides)**
  - **The value of <mu> is not constant over a run and certainly not over a whole data-taking period. (see following slides)**
  - **We need to know the beamspot in order to simulate pile-up correctly.**
    - **We typically simulate the signal and background samples to use a slightly larger beamspot than expected in data, this allows us to reweight the events to match data.**
    - **This means we have to be careful that the signal and background samples used as inputs to a job are simulated with the same conditions.**

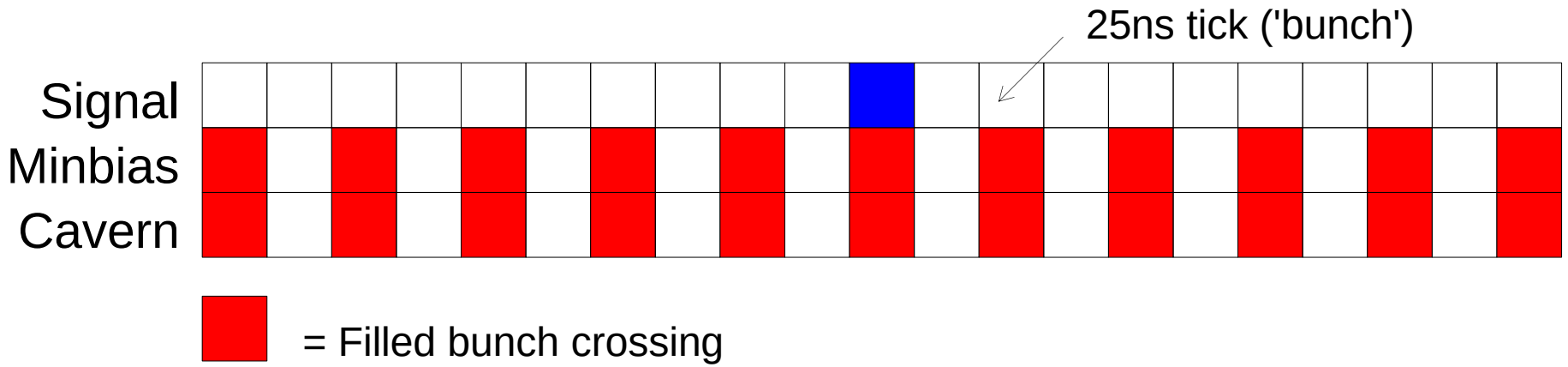
Pile-up Modelling: Time Windows

- **Each sub-system has a different sensitive time window in which interactions could affect the read-out for the current bunch-crossing.**



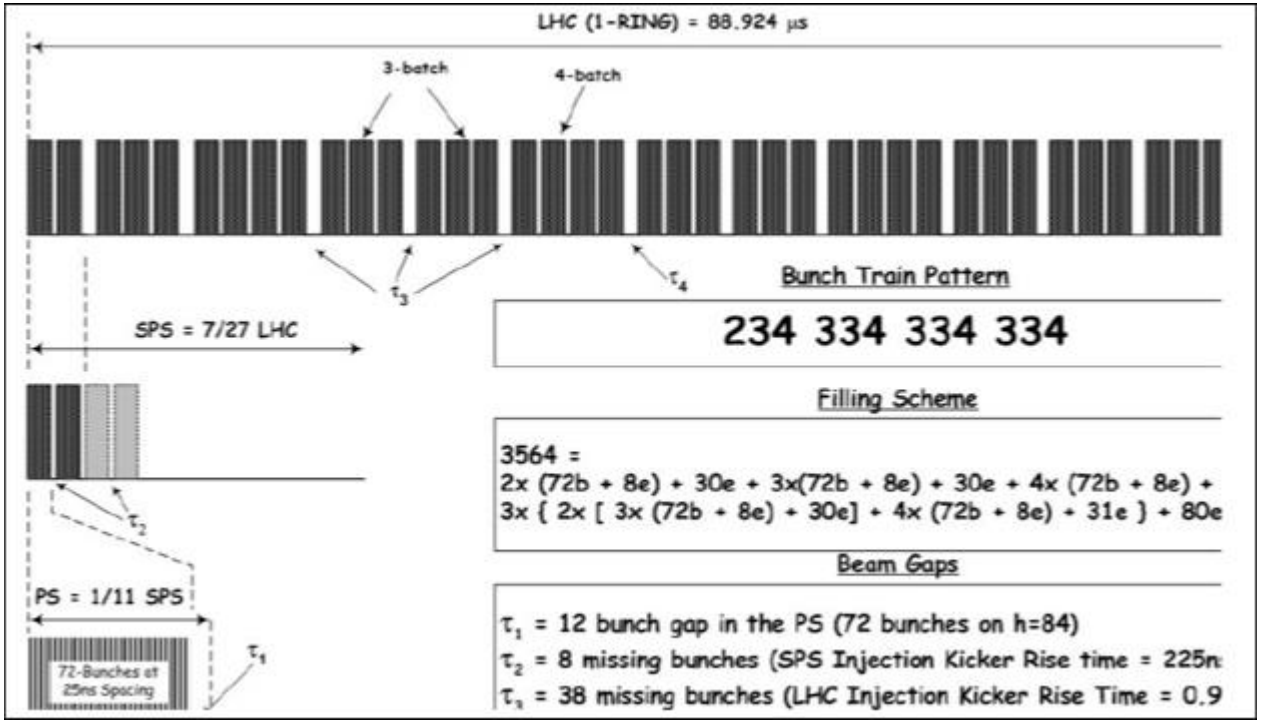
Pile-up Modelling: Bunch Structure (I)

Example of a pile-up model with fixed 50ns spacing between filled bunches:



In reality the structure of filled and empty bunch crossings can be more complicated.

Pile-up Modelling: Bunch Structure (II)



2015 LHC fill pattern.

- **ATLAS includes the modelling of more complicated bunch crossing patterns in the simulation. There are clear effects on the pile-up and hence detector response depending on where in the bunch train the triggering signal event occurs, so this needs to be taken into account.**
- **Patterns can be up to 3564 elements in length and can loop around between the beginning and end of the pattern if required.**
- **For each signal event, the triggering bunch crossing is picked from the filled bunch crossings in the pattern, with a probability proportional to the relative luminosities of each bunch crossing.**

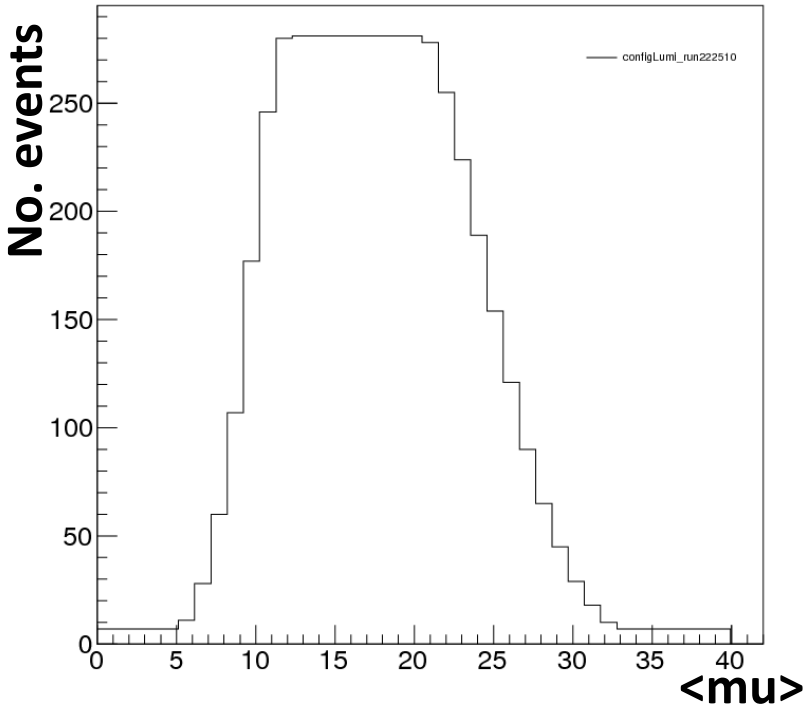
Pile-up Modelling: Variable Luminosity (I)

- **Clearly, the bunch luminosity of the LHC varies over time.**
- **Both in-time and out-of-time pile-up effects are important.**
- **Simulating samples at a fixed <mu> value makes it difficult to re-weight MC to data.**
- **You *could* reweight according to the in-time mu value, but the mu values used in out-of-time bunch-crossings will still be drawn from a distribution with a mean of <mu>.**

Pile-up Modelling: Variable Luminosity (II)

➔**Use a range of <mu> values within each simulated sample.**

- **The <mu> value used is recorded for each event.**
- **This can then be to re-weight the MC sample to match a given set of data periods.**
- **Figure shows an example 5000 event pattern for the MC15 50ns run.**



- **Machinery exists to model bunch-crossing dependent mu variations (which exist in data).**
  - **so far it has not been necessary to include this in the MC.**

PileUpEventLoopMgr

- **Pile-up Digitization jobs use the PileUpEventLoopMgr which extends the standard AthenaEventLoopMgr.**
- **The PileUpEventLoopMgr sets up the caches of background events based on the <mu> value and the relative proportions of low and high pT minimum bias to be used.**
- **At the start of each event the PileUpEventLoopMgr picks the number of minimum bias events to be used for each bunch-crossing from a Poisson distribution with mean <mu>.**
- **Minimum bias events are allocated to either the high pT or low pT samples and a matching event is assigned from the cache.**

PileUpTools: BC by BC Pile-Up

- **The original pile-up approach (AKA the "Algorithm" approach) :**
  - **digitized the information from all required bunch crossings for a given sub-detector before moving on to the next sub-detector.**
  - **Background event info was cached to allow re-use.**
  - **This design minimised I/O at the expense of memory.**
- **The "PileUpTools" approach (current default):**
  - **provides one filled bunch crossing at a time to all sensitive sub-detectors.**
  - **Background events are read as required and discarded from memory after each filled bunch crossing is processed.**
    - **Sacrifice caching of background to save memory.**
  - **A single pile-up Algorithm calls an AlgTool for each sub-detector. The AlgTools know the time window for which they are sensitive to bunch crossings.**
  - **Digits/RDOs are produced from intermediate information cached locally by the sub-detector tools, after all filled bunch-crossings have been processed.**

PileUpTool Interface

```
#include "EventInfo/PileUpEventInfo.h"  /*SubEvent*/
#include "GaudiKernel/IAlgTool.h"
class IPileUpTool : virtual public IAlgTool{
public:
  ///called before the bunchXing loop
  virtual StatusCode prepareEvent(unsigned int /*nInputEvents*/) { return StatusCode::SUCCESS; }
  ///called for each active bunch-crossing (time in ns)
  virtual StatusCode
    processBunchXing(int bunchXing,
                     PileUpEventInfo::SubEvent::const_iterator bSubEvents,
                     PileUpEventInfo::SubEvent::const_iterator eSubEvents) = 0;
  ///flags whether this tool is "live" for bunchXing (time in ns)
  /// implemented by default in PileUpToolBase as FirstXing<=bunchXing<=LastXing
  virtual bool toProcess(int bunchXing) const =0;
  ///called at the end of the bunchXing loop
  virtual StatusCode mergeEvent() { return StatusCode::SUCCESS; }
  ///alternative interface which uses the PileUpMergeSvc to obtain all
  ///the required SubEvents.
  virtual StatusCode processAllSubEvents() = 0;

  static const InterfaceID& interfaceID() {
    static const InterfaceID _IID( "IPileUpTool", 1, 0 );
    return _IID;
  }
};
```

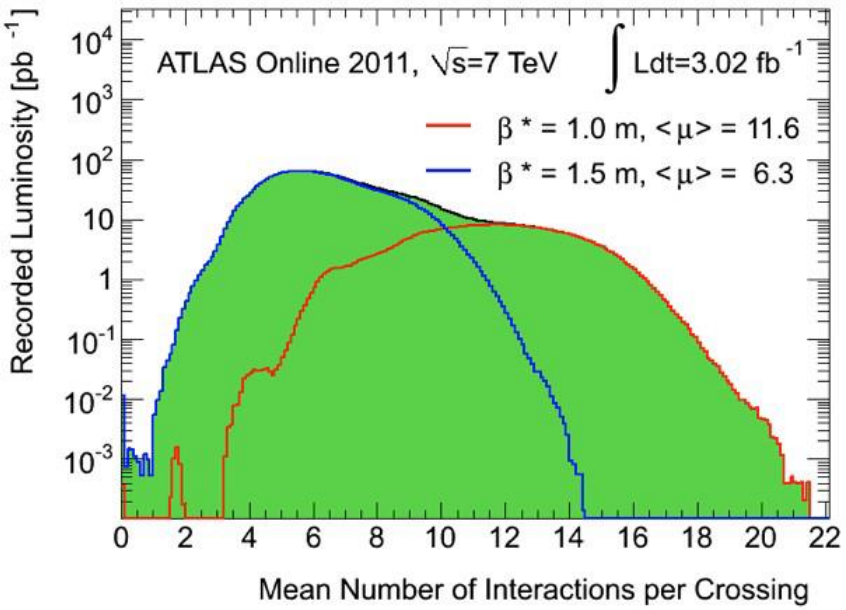
PileUpTools Implementation

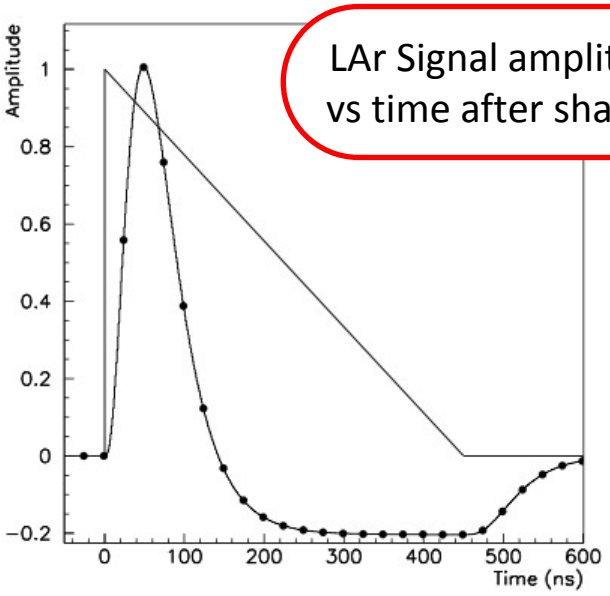
- **All PileUpTools implement the IPileUpTool interface and inherit from PileUpToolBase.**
- **The implementation details are sub-detector-specific. Multiple-implementations per sub-detector possible.**

PileUpTools Configuration

- **Configuration is done using CfgGetter (as in the ISF) wherever possible**
  - **https://twiki.cern.ch/twiki/bin/viewauth/Atlas/ConfiguredFactory**
  - **Allows for easy switching between Standard and Fast Digitization versions for each sub-detector.**
  - **As digitization makes use of a lot of sub-detector conditions code and many sub-system offline software groups do not use CfgGetter, then some parts of the configuration have to be wrapped.**

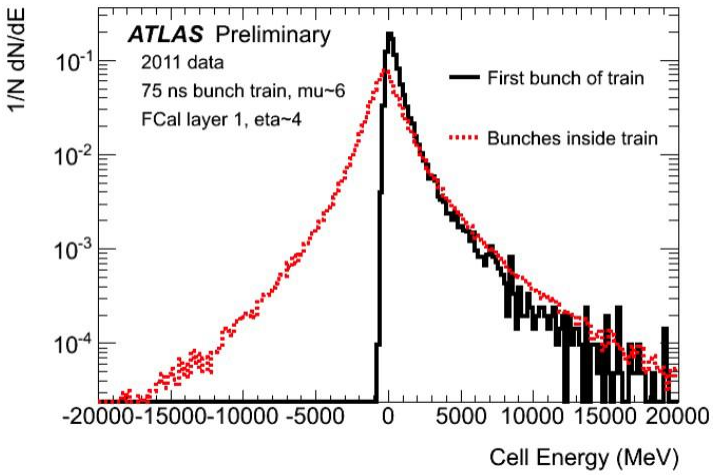
Impact of pile-up on Reconstruction (III)





LAr Signal amplitude vs time after shaping

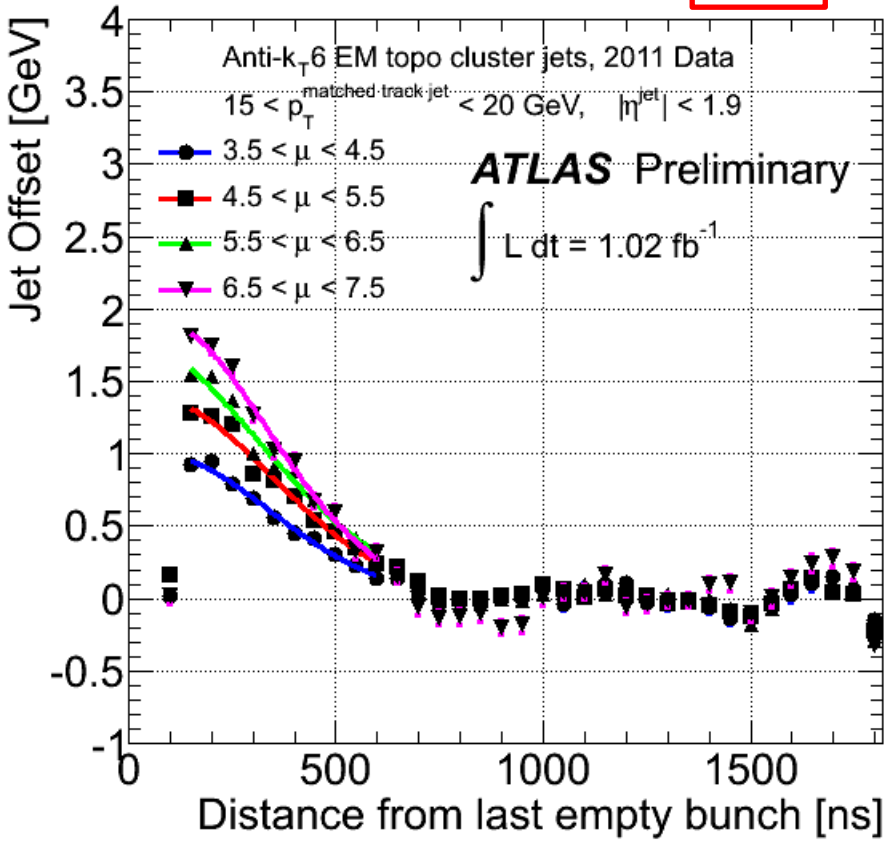
- **Do not expect a significant impact on tracking, nor muons, nor even electrons and photons.**
- **But sizeable impact on jets (+ETmiss) and t.**
- **LAr drift-time is ~ 500 ns and out-of-time bunches have impact on measurement.**
- **Bipolar pulse shaping designed so that <ET> ~ 0 for 25 ns bunch-spacing and uniform intensity per BX.**
- **Optimal performance will require correction per cell type in h-bins and as a function of luminosity to set average measured ET to ~ 0.**



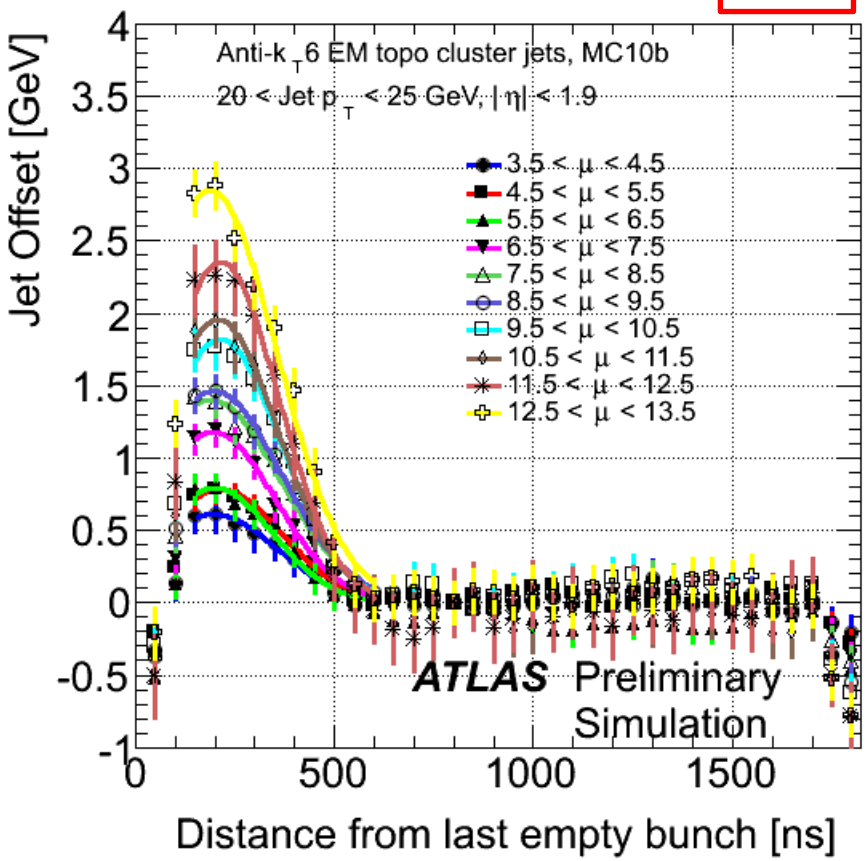
Impact of pile-up on Reconstruction (IV)

- **Jet offsets from pile-up are modelled to <50%**
- **Remaining differences from BCID-to-BCID beam current variation were not modelled in MC10b.**

Data



MC



Other backgrounds

- Currently the minimum bias background is the only one which we actively model.
- Cavern background (mentioned earlier)
  - has been investigated by the muon and cavern background groups.
  - Modelling continues to improve at the simulation level.
  - Muon Spectrometer Digitization code would need to be extended in order to properly treat it at the digitization level.
  - So far it has not been necessary to include this in the MC.
- Beam Halo interactions are the interaction of protons with material limiting apertures near the experiment (typically the tertiary collimator located at z = 150 m from the interaction point).
- Beam Gas interactions are the inelastic interactions of protons with the residual gas inside the beam-pipe.
  - Beam gas and beam halo have been investigated by the non-collisions background group.
  - They can be distinguished from pp interactions as they typically start from one side or other of the detector rather than at the interaction point.
  - So far it has not been necessary to include them in the MC.
- Cosmic ray muons entering the detector can be easily vetoed in data and so are not included in the simulation.

Example MC15 Digitization Command

```
xrdcp root://eosatlas//eos/atlas/atlascerngroupdisk/proj-sit/digitization/RTT/mc15a/valid1.110401.PowhegPythia_P2012_ttbar_nonallhad.simul.HITS.e3099_s2578_tid04919495_00/HITS.04919495._001041.pool.root.1 HITS.04919495._001041.pool.root.1; LowPtMinbiasHitsFile="/eos/atlas/atlascerngroupdisk/proj-sit/digitization/RTT/mc15a/mc15_valid.361034.Pythia8EvtGen_A2MSTW2008LO_minbias_inelastic_low.merge.HITS.e3581_s2578_s2169_tid05098374_00/"; for file in `eos ls $LowPtMinbiasHitsFile`; do xrdcp root://eosatlas/$LowPtMinbiasHitsFile/$file $file ; done; HighPtMinbiasHitsFile="/eos/atlas/atlascerngroupdisk/proj-sit/digitization/RTT/mc15a/mc15_valid.361035.Pythia8EvtGen_A2MSTW2008LO_minbias_inelastic_high.merge.HITS.e3581_s2578_s2169_tid05098387_00/"; for file in `eos ls $HighPtMinbiasHitsFile | grep 00067`; do xrdcp root://eosatlas/$HighPtMinbiasHitsFile/$file $file ;done;

Digi_tf.py --maxEvents 25 --skipEvents 0 '--inputHITSFile HITS.04919495.*.pool.root.?' \
--outputRDOFile mc15_2015_ttbar.RDO.pool.root --conditionsTag default:OFLCOND-RUN12-SDR-25 \
--geometryVersion ATLAS-R2-2015-03-01-00 --digiSeedOffset1 170 --digiSeedOffset2 170 --preInclude 'HITtoRDO:Digitization/ForceUseOfPileUpTools.py,SimulationJobOptions/preInclude.PileUpBunchTrainsMC15_2015_50ns_Config1.py,RunDependentSimData/configLumi_run222510.py' \
--postInclude 'default:PyJobTransforms/UseFrontier.py' --pileupFinalBunch 6 \
'--inputHighPtMinbiasHitsFile HITS.05098387.*.pool.root.?' \
'--inputLowPtMinbiasHitsFile HITS.05098374.*.pool.root.?' --jobNumber 1 \
--numberOfHighPtMinBias 0.12268057 --numberOfLowPtMinBias 39.8773194 \
--postExec '"all:CfgMgr.MessageSvc().setError+=[\"HepMcParticleLink\"]" "HITtoRDO:job.StandardPileUpToolsAlg.PileUpTools[\"MergeMcEventCollTool\"].OnlySaveSignalTruth=True;ToolSvc.LArAutoCorrTotalToolDefault.deltaBunch=1"' \
--preExec 'all:from AthenaCommon.BeamFlags import jobproperties;jobproperties.Beam.numberOfCollisions.set_Value_and_Lock(20.0);from LArROD.LArRODFlags import larRODFlags;larRODFlags.NumberOfCollisions.set_Value_and_Lock(20);larRODFlags.nSamples.set_Value_and_Lock(4);larRODFlags.doOFCPileupOptimization.set_Value_and_Lock(True);larRODFlags.firstSample.set_Value_and_Lock(0);larRODFlags.useHighestGainAutoCorr.set_Value_and_Lock(True)'
```

Digi_tf.py argument break-down

- **Standard arguments:**

**--maxEvents 25 --skipEvents 0 '--inputHITSFile HITS.04919495.*.pool.root.?' **
**--outputRDOFile mc15_2015_ttbar.RDO.pool.root --conditionsTag default:OFLCOND-RUN12-SDR-25 **
--geometryVersion ATLAS-R2-2015-03-01-00

- **preIncludes:**
  - **Digitization/ForceUseOfPileUpTools.py,**
    - **Does what you'd expect**
  - **SimulationJobOptions/preInclude.PileUpBunchTrainsMC15_2015_50ns_Config1.py,**
    - **Contains the bunch pattern which will be used for this job**
  - **RunDependentSimData/configLumi_run222510.py**
    - **Contains the <mu> distribution to be simulated for this dataset. Also sets the run-number and timestamp.**
- **Other arguments:**
  - **--numberOfHighPtMinBias 0.12268057 --numberOfLowPtMinBias 39.8773194**
    - **These arguments set the initial size of the cache of background events and the ratio of low to high pT minimum bias. If a configLumi file is being used then they do not indicate the <mu> value for the job.**
  - **--pileupFinalBunch 6**
    - **This argument sets the number of bunch-crossings after the interaction that will be simulated.**
  - **--jobNumber 1**
    - **The pattern in the configLumi file contains more events than are processed in a single job. This argument allows code to calculate the correct point in the pattern for this particular job**
  - **--digiSeedOffset1 170 --digiSeedOffset2 170**
    - **These arguments provide an offset for the random number streams used in this job.**

Backup

Background Simulation: Pythia8 + Geant4