CERN - European Organization for Nuclear Research

LCD-Note-2011-025

Software and Parameters for Detailed TPC Studies in the CLIC CDR

M. Killenberg*

* *CERN, Switzerland*

February 29, 2012

Abstract

For the TPC occupancy and time stamping studies in the CLIC CDR the MarlinTPC software package has been used in combination with Mokka for the full detector simulation. This document describes the working principle of the Marlin processors used for digitisation and reconstruction, and lists the parameters for reference.

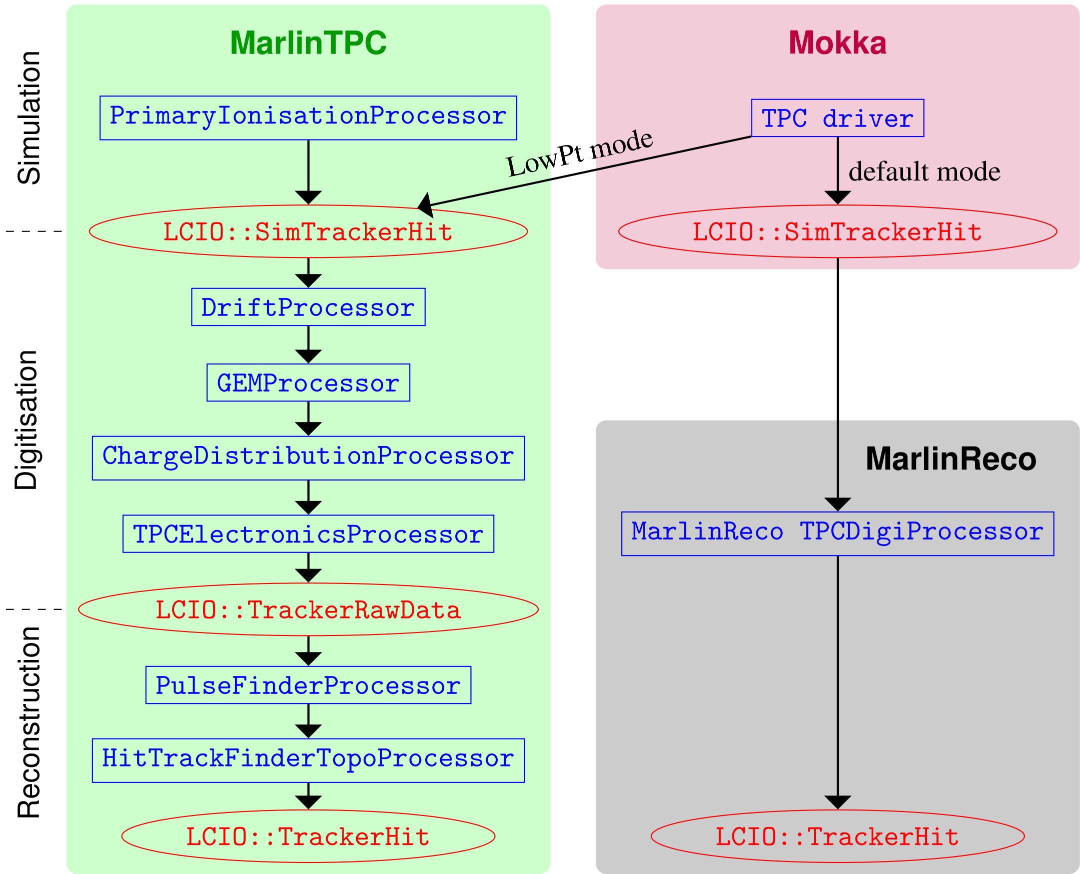

Figure 1: Data flow for simulation, digitisation and reconstruction up to the reconstructed hit level in MarlinTPC and the software used for the CLIC CDR mass production (Mokka and MarlinReco).

1 Introduction

For the dedicated TPC studies which have been performed for the CLIC CDR a very high level of simulation detail is required. For the occupancy study [1] the raw data output of the readout electronics is needed. For a time stamping study [2], where the reconstructed TPC track is matched with a hit in the Silicon External Tracker to determine the correct bunch crossing within a bunch train, the pad angle effect and the shaping of the electronics should be fully simulated and not be put in as a parametrisation.

The MarlinTPC package [3] provides this detailed digitisation and reconstruction chain. Figure 1 shows the data flow for simulation, digitisation and reconstruction up to the level of reconstructed hits for MarlinTPC and for the software used for the CLIC_CDR mass production (Mokka and MarlinReco). The MarlinTPC digitisation simulates the drift and diffusion as well as the gas amplification in a GEM stack and the influence of the electronics step by step, allowing to control and monitor the various parameters. The reconstruction is the same as the one used for prototype studies, allowing for a maximum of realism.

In this note the individual MarlinTPC processors are discussed, as well as the simulation in Mokka. For reference the parameters used in the studies are listed in section 5, and the processor lists are given in sections 6 and 7.

1.1 Wording

The words *simulation, digitisation* and *reconstruction* are used as follows throughout this note:

Simulation: The interaction of primary particles with the detector material and the generation of charge/energy depositions.

Digitisation: The calculation of the ADC response from the primary charge depositions. It includes simulating the effects of the drift in the gas, the gas amplification and the digitisation proper, i.e. the shaping of the electronics and the ADC response.

Reconstruction: All the steps from the ADC response to the reconstructed track, including pulse finding, hit finding, pattern recognition and calculation of the track parameters.

The verb *to simulate* is used both in the *simulation* and *digitisation* context.

2 Simulation

For the simulation of charge depositions in the detector the Geant4 [4] based full detector implementation Mokka [5] has been used, as well as a parametrisation based TPC simulation embedded within MarlinTPC. The simulation in MarlinTPC is fast and can directly be used in the Marlin processor chain, but only simulates the fiducial volume of the TPC. Especially for background studies this is not sufficient because backscattering particles have a significant impact on the charge deposited in the TPC (for instance pair background from beamstrahlung [1]). Thus the Mokka full simulation has been used for the studies which include background.

2.1 Full Detector Simulation with Mokka

The Mokka TPC driver in its default mode places hits on the middle of each pad row, where the reconstruction of a real TPC calculates the hits. During digitisation in MarlinReco [6] the simulated hit is smeared with the known detector resolution, and neighbouring hits are merged if they are closer than the double hit resolution. This is a good model for mass production because it keeps the digitisation fast and simple, and provides correct values for the covariance matrices of the hits since the applied errors are known. It is the default simulation and reconstruction in ILCSoft and is used for all studies presented in the CLIC CDR, except for the TPC occupancy and time stamping studies.

For these detailed studies the default MarlinReco digitisation has the disadvantage that it skips the signal generation on the pads, and the following part of the reconstruction. This cannot easily be calculated from the hits placed in the middle of a pad row, especially for tracks with a low angle to the pad row (see figure 2). At CLIC the high $\gamma\gamma \rightarrow$hadrons background produces many low energetic particles which curl and have such low angles [7]. Especially for the occupancy calculation it is important to have the exact number of pads, the correct time and also stubs from

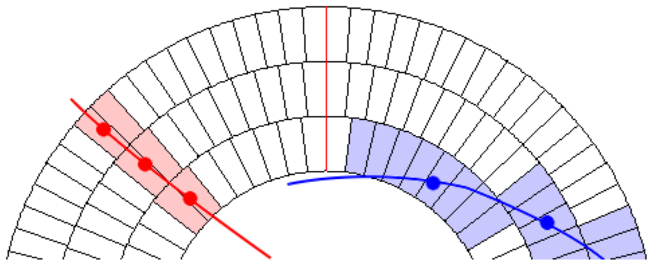

Figure 2: For tracks with a low angle to the pad row the number of occupied pads cannot easily be determined from a hit in the centre of the pad row.

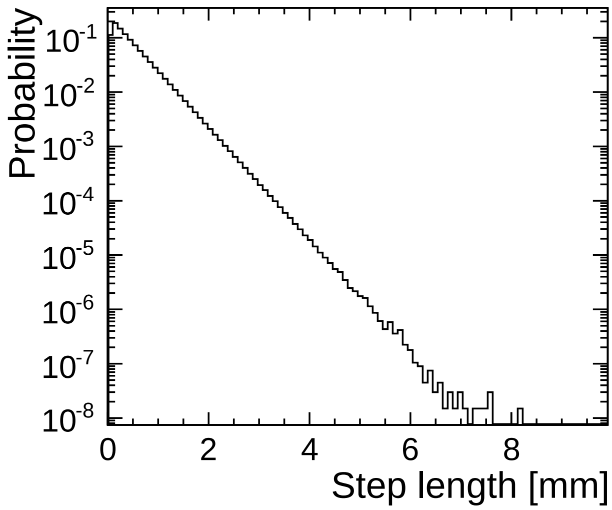

Figure 3: The distance between non-empty Geant4 steps in $Ar/CH_4/CO_2$ 93/5/2 gas, simulated with a step length limit of 50 µm.

delta electrons which occupy additional voxels and should not be absorbed in a hit in the middle of the pad row.

The TPC driver in Mokka also has a mode which is not based on pad rows, but limits the simulation step length to a fixed value. The mode with step length limit is called *lowPt* mode because it has been introduced for very low energetic particles which curl within one pad row and would not produce any hits with the normal driver, as they never cross the middle of a pad row. For the detailed simulation the `TPCLowPtCut` value was set to 3 TeV, forcing all particles to be treated with the step limit method. Figure 3 shows the distance between non-empty Geant4 steps for muons with a momentum of 200 GeV/*c* in $Ar/CH_4/CO_2$ 93/5/2 gas, simulated with a step length limit of 50 µm. One can see that the exponential behaviour, which is expected for the free path length between two collisions, is very well reproduced.

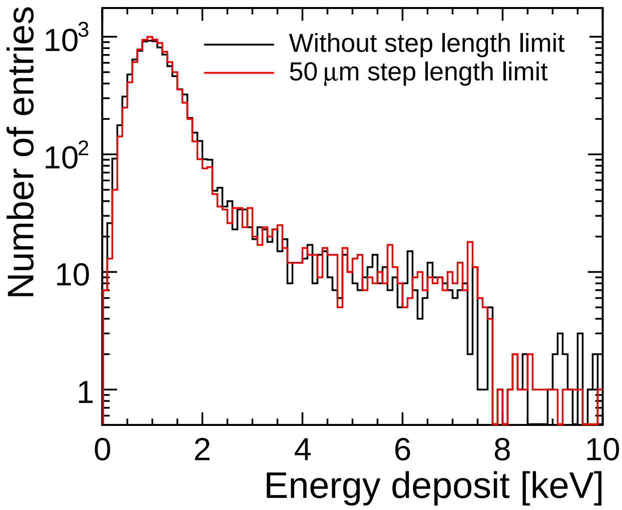

Figure 4: The energy deposited by 200 GeV muons in a 5 mm Ar/CH_4/CO_2 93/5/2 gas layer in Geant4, simulated with and without step length limit.

To demonstrate that the *lowPt* mode works reliably up to high energies, the energy deposit in a 5 mm gas layer has been simulated without step length limit and with a 50 µm step length limit. Without step length limit the step is ended at the layer boundary. The result for 200 GeV muons in Ar/CH_4/CO_2 93/5/2 is shown in figure 4: Both distributions nicely match and show, that the *lowPt* mode can also be used for high energies if this high level of detail is required.

2.2 Simulation with MarlinTPC

HEED is a dedicated software for the simulation of ionisation in gases [8]. To implement an ionisation simulation for TPC stand alone studies, the `PrimaryIonisationProcessor` is based on a parametrisation of HEED output data for several typical TPC gas mixtures [9]. This allows for realistic cluster sizes and distances between the energy deposits while being fast thanks to the use of parametrisations. The simulated hits are only placed in space, but unlike in the full Mokka simulation, the time of flight is not taken into account. This behaviour is intentional, because the reconstruction in MarlinTPC does not have a time of flight correction yet.

3 Digitisation with MarlinTPC

3.1 DriftProcessor

The first step in the digitisation calculates the effects of the drift. Each electron from the primary ionisation is displaced individually, according to Gaussian distributions determined from the diffusion constants of the gas. As primary clusters usually consist of only few electrons, it is

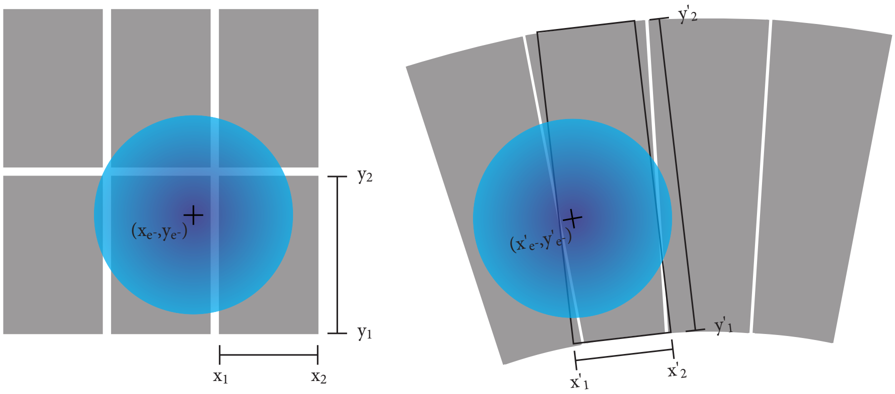

Figure 5: Visualisation of the charge distribution onto the pads [11]. For keystone shaped pads the pad shape is approximated by a rectangle to allow using an analytical solution.

more realistic to displace the individual electrons than applying a Gaussian smearing for the overall charge of the cluster.

3.2 GEMProcessor

The `GEMProcessor` simulates the gas gain in a stack of three gas electron multiplier foils (GEMs) [10]. It applies the collection into the GEM (binomial statistics for each electron), gas amplification (exponential distribution) and extraction from the GEM (binomial statistics) for each of the tree GEMs. In addition, electron attachment in the transfer regions and the induction gap is taken into account (negative exponential distribution). All coefficients are applied using a random number generator with the corresponding distributions, which realistically reproduces the gas gain fluctuations.

3.3 ChargeDistributionProcessor

The signal arriving on each pad is calculated using the `ChargeDistributionProcessor`. The charge signal leaving the GEM stack is almost Gaussian shaped for a single incoming electron. The width of the Gaussian is determined by the diffusion between the GEMs. The Gaussian charge is projected onto the pads (see figure 5) and the charge on each pad is calculated using the analytical solution of the two-dimensional integral of the Gaussian on the area of the pad [11]. Afterwards the charge is filled into a 3D map. The size of a voxel (3D space bucket) in the xy plane corresponds to a pad, and in z direction a time binning is used which for this study has been set to a third of an ADC sample. Due to the long readout time the TPC integrates over many bunch crossings, a full bunch train in the CLIC case. The digitisation has to take this into account and overlay data from several bunch crossings. The `ChargeDistributionProcessor` performs overlaying in memory. It writes data from several events into the same voxel map, applying the correct shift in time according to the bunch crossing.[1)] To avoid a huge memory consumption, in this study the time shift for the bunch crossing has been applied, but the map has been written out after every event.

3.4 TPCElectronicsProcessor

The `TPCElectronicsProcessor` calculates the ADC response of each pad. It takes the charge of each input voxel (in units of electrons) and applies a Gaussian shaping. The Gaussian is then sampled with the given readout frequency. The sampled signal of all incoming charges is added and in a final step converted into ADC counts, using the ADC resolution (usually 10 bits) and the dynamic range (usually 1 million electrons). Adding the shaped signals simulates the behaviour of a charge sensitive ADC where a capacitor is charged by the incoming electrons and discharged using an operational amplifier. The use of a Gaussian shape is an approximation. Modern TPC readout electronics like the ALTRO uses tail cancellation filters to produce a symmetric signal [12]. As the input signal from micro pattern gas amplification does not have long tails, no tail cancellation is done. The width of the Gaussian is calculated as $\sigma = risetime/3$.

The `TPCElectronicsProcessor` simulates a zero suppressing ADC with two threshold levels: The threshold itself performs the zero suppression. All samples below this value are discarded. For a pulse to be accepted its maximum has to be above the minimum pulse height value, which is higher than the threshold (see figure 6 for visualisation). This allows an effective noise suppression without cutting into the tails of the pulse.

3.5 Overlaying of Raw Data

In case of simulating a full bunch train with several background components it is preferable to do the overlaying of the digitised raw data, and not of the primary charge depositions before digitisation. The latter uses a lot of memory since all primary electrons have to be held in memory during the digitisation. Processing each background component, and the physics signal, separately significantly reduces the memory consumption. The overlaying now is done in three steps:

1. Overlaying of one bunch crossing for each component:
   To allow the assigning of the time shift due to the bunch crossing in the `ChargeDistributionProcessor` to work correctly, all data from one bunch crossing has to be in one event. This is done by overlaying one bunch crossing of background for each component, using the normal `OverlayProcessor` [13].

2. Overlaying the different components:
   The `OverlayRawDataProcessor` has been introduced to overlay the data from different LCIO files per event. It just merges the two `lcio::TrackerRawData` collections. This may result in two raw data entries which overlap. This is not resolved in this processor but in the next step.

[1] In LCIO the data is stored in events. For this study the data of one bunch crossing is stored in one LCIO event, one LCIO run corresponding to one bunch train.

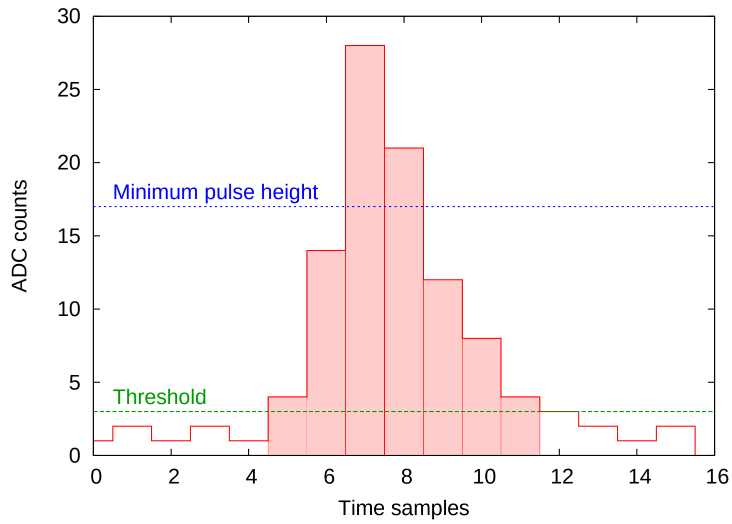

Figure 6: Visualisation of the threshold and the minimum pulse height level. A pulse is accepted if at least one sample is above the higher level. All samples above the threshold are part of the pulse (red shaded).

3. Composing one bunch train:
   After digitisation the data is still stored as one bunch crossing per event. The TPC however integrates the full bunch train, so all events have to be merged to one "bunch train event". This is done in the `MergeRawDataProcessor`. All `TPCRawData` pulses are read into a memory map. When adding to this map, the new pulse is tested for overlap. If it overlaps with an already existing one, both pulses are merged, the signals in the overlapping time samples are added. At the end of the run, when all pulses have been read into memory, a threshold and a minimal pulse height cut are applied and the data is written out. The threshold and pulse height cut at this stage are necessary since they have not been applied in the electronics processor to avoid threshold effects.

4 Reconstruction with MarlinTPC

4.1 Pulse Reconstruction

The `PulseFinderProcessor` scans the raw data for pulses, and calculates the charge sum and the time of the pulse. With a zero suppressing ADC, like it is simulated, the main functionality is not the finding but the splitting of pulses from nearby tracks and the calculation of the pulse parameters. The splitting is done by searching for local minima in the ADC signal and separating the pulses at this position.

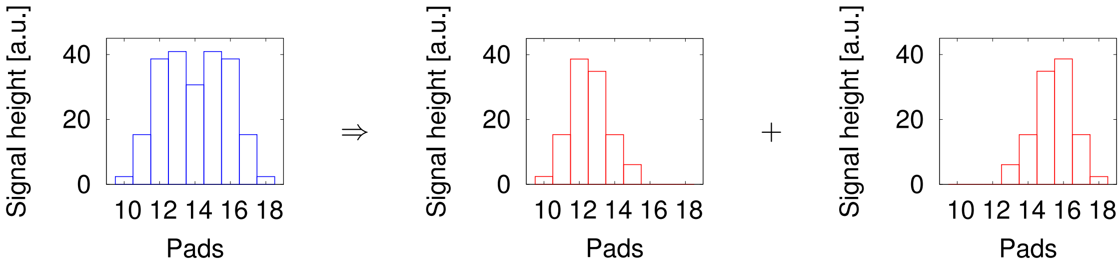

Figure 7: Visualisation of the signal splitting algorithm. The input signal (blue) is the sum of two identical Gaussian shaped signals. The two red figures are the response of the splitting algorithm.

4.2 Hit and Track Finding

For the time stamping study the full reconstruction chain is only run for single muons from the vertex.[2] In this case a combined three-dimensional hit and track finding can be used (`HitTrackFinderTopoProcessor`). All contiguous voxels in the TPC are grouped and hits are reconstructed as the centre of gravity of the charge in one pad row. All hits from each contiguous region (usually there is only one for the single muons) are stored as a track candidate. It is not a full track yet as the track parameters are only calculated in the next step. The track candidate is more a container to store the hits which belong together.

4.3 Track Parameters

The track parameters are calculated in the `TrackSeederProcessor` using an analytical method. The parameters in the xy projection (d_0, ϕ_0 and the curvature) are calculated independently from the parameters in the sz projection (z_0 and the dip angle). For all parameters the errors of the reconstructed hits are not taken into account to be able to calculate the solution analytically. For single muons, which do not have a significant change of their absolute momentum, this method calculates the track parameters with sufficient precision.

4.4 Hit Splitting

Even with single tracks like in the time stamping study, delta electrons cause badly reconstructed hits which have a centre of gravity far away from the particle trajectory, or even multiple separable hits per pads row.

The hit splitting algorithm searches for minima in the charge distribution on the pads of a reconstructed hit. For each minimum the pulse is split. In order not to simply cut the signal at the minimum, the signal splitter assumes a symmetric shape of the contributing hits. It uses the shape of the outer part to the pulse (from the left or right edge of the signal to nearest maximum) to estimate the contribution in the overlapping region and to disentangle the hits (see figure 7). This simple algorithm only works on the data of a single hit. It does not use additional

[2] The occupancy study is performed on the `TPCRawData`, so no reconstruction is needed here.

information like the inclination angle of the track or the drift distance to do an estimate of the pulse shape. A more sophisticated algorithm using this information could achieve a better hit separation, even for pulses where no explicit minimum is visible.

The `HitsInTracksSplitterProcessor` performs the hit splitting for all hits stored in a track, and writes the split hits into a new track with the same track parameters, so the association of hits to tracks from the original pattern recognition is preserved.

4.5 Outlier Rejection

As the calculation of the track parameters does not use the errors of the individual hits, it is important to cut away outliers which would bias the track parameters. The rejection of outliers is based on fixed distance cuts in the xy and z plane of the reconstructed hit to the reconstructed track. The current version of the code does not take into account the changing of the resolution in dependence on the drift distance. This is foreseen in a future version. The `OutlierRejectionProcessor` uses the expected average widths of the residual distributions σ_{xy} and σ_z and removes hits from the track which are further away than a configurable multiple of this with. For the time stamping study a very loose cut of 5 σ has been applied both in the xy and in the z direction.

5 Parameters

5.1 Mokka

The TPC subdetector in the `CLIC_ILD_CDR` detector model uses a TPC driver (`tpc06`) which does not support the *lowPt* mode. For the detailed TPC simulation this subdetector has been replaced with an identical subdetector which uses the driver `tpc09`, which does support step length limiting.

As the pair background is very sensitive to the magnetic field, the homogeneous field `SField01` has been replaced with the more realistic `FieldMap4TNoQuad`.

Parameter	Detailed TPC simulation	Mass production
Detector model	CLIC_ILD_CDR	CLIC_ILD_CDR
TPC subdetector	clictpc01swl	clictpc01
Magnetic field	FieldMap4TNoQuad	SField01
TPCLowPtStepLimit	true	–
TPCLowPtCut	3 TeV	–
TPCLowPtMaxStepLength	0.2 mm	–
TPCLowPtMaxHitSeparation	0.01 mm	–
TPCCut	0 MeV	10 MeV
rangeCut	0.005 mm	0.1 mm
Physics list	QGSP_BERT_HP	QGSP_BERT

5.2 Geometry

The geometry is identical to the CLIC_ILD_CDR detector model. The following table summarises the TPC parameters.

Parameter	Value
Inner field cage radius	329 mm
Inner field cage thickness	1.16 mm
Outer field cage radius	1808 mm
Outer field cage thickness	1.51 mm
Cathode thickness	120 µm
Drift length	2120 mm
Pad layout type	FixedPadSizeDiskLayout
Pad size (pitch)	$1 \times 6\ \text{mm}^2$
Number of pad rows	224
Inner radius pad plane	395 mm
Outer radius pad plane	1739 mm

5.3 Gas

In this study different gases have been used in the different steps. This is not fully consistent. In terms of drift velocity and diffusion the mixture used by the T2K experiment ($Ar/CF_4/iC_4H_{10}$ 95/3/2, referred to as T2K gas) is favourable to the mixture proposed in the TESLA TDR ($Ar/CH_4/CO_2$ 93/5/2, referred to as TDR gas). However, for T2K gas not all the parametrisations which are used in the digitisation are available. The following gases have been used:

- Mokka: TDR gas
  For a charged particle, the ionisation in the gas is mainly determined by the argon component. The choice of quenchers only has little impact [9]. For consistency with the mass production the TDR gas has been used.

- `DriftProcessor`: T2K gas
  Due to diffusion and drift velocity, the `DriftProcessor` has the largest impact on resolution and occupancy. Here the T2K gas has been used because it has lower diffusion and higher drift velocity compared to TDR gas and therefore is the more likely choice for a future linear collider TPC.

- `GEMProcessor`: TDR gas
  The transfer coefficients used in this processor are parameterised according to measurements [14]. As the parametrisations are not available for T2K gas, TDR gas has been used. The charge transfer in a GEM stack is mainly determined by the electrostatics, i.e. the voltages and distances in a GEM stack [15]. The voltages in this processor have been set to achieve a certain gas gain (optimal use of the ADC's dynamic range). For the T2K gas these voltages will be different, and probably the ion backdrift as well. The results of this study should not be affected by the choice of gas at this stage.

5.4 DriftProcessor

Parameter	Value
Gas	T2K gas
Drift field	300 V/cm

5.5 GEMProcessor

Parameter	Value
Number of GEMs	3
Drift field	300 V/cm
Transfer fields	2500 V/cm
Induction field	5000 V/cm
Mean gas ionisation energy	26 eV
Gas	TDR gas
Overall gas gain	2733

5.6 ChargeDistributionProcessor

The z length of a voxel has been chosen to be a third of the 25 ns sampling frequency to reduce binning effects in the following `TPCElectronicsProcessor`, compared to choosing the same frequency. All other parameters are identical to the `GEMProcessor`.

Parameter	Value
Number of GEMs	3
Drift field	300 V/cm
Transfer fields	2500 V/cm
Induction field	5000 V/cm
Gas	TDR gas
zBinLength	8.33 ns

5.7 TPCElectonicsProcessor

The ADC is supposed to be a 10 bit ADC (0 – 1023 ADC counts). Due to memory limitations in the occupancy study, the overlay could only be performed on the digitised raw data. To avoid threshold effects the resolution has been chosen a factor 10 better, and reduced after the overlaying. The result is a maximal value of 10234 (10235 would be rounded up to 1024 when scaling back). The thresholds are also set to 1 (only zero suppression) and proper thresholds will be applied during raw data overlay.

Parameter	Value
ADC resolution	10 bit
ADC max value	1023 ADC counts (10234 for occupancy study)
ADC threshold	1 ADC count (just 0 suppression)
Minimum pulse height	1 ADC count
Dynamic range	1e6 electrons
Readout frequency	40 MHz
Rise time	60 ns

5.8 PulseFinderProcessor

Parameter	Value
DoMultiplePulseSplitting	true
ForceSpectrumSave	true
Maximum ADC value	1023
Minimum pulse height	5 ADC counts
Minimum pulse length	3 ADC samples
Noise width (override parameter)	1 ADC count
Pulse end threshold	2
Pulse start threshold	3
UseChargeAveragedPulseTime	true

5.9 HitTrackFinderTopoProcessor

Parameter	Value
Maximum number of empty consecutive pads in a hit	1
Maximum number of subsequently missing hits (rows)	0
Minimum number of pads per hit	1
Minimum number of hits per track	25

5.10 OutlierRejectionProcessor

Parameter	Value
Width of the residual distribution in the *xy* plane (σ_{xy})	0.11 mm
Width of the residual distribution in the *z* direction (σ_z)	0.3 mm
Rejection cut in the *xy* plane	5.0 σ_{xy}
Rejection cut in the *z* plane	5.0 σ_z

6 Software for the TPC occupancy study

6.1 Simulation

Mokka with the `CLIC_ILD_CDR` detector model, the `clictpc01swl` TPC subdetector and the `FieldMap4TNoQuad` magnetic field.

6.2 Digitisation

- OverlayProcessor or OverlayIncoherentPairsProcessor[3)]
- DriftProcessor
- GEMProcessor
- ChargeDistributionProcessor
- TPCElectronicsProcessor
- OverlayRawDataProcessor
- MergeRawDataProcessor

6.3 Reconstruction

These data have not been reconstructed. The occupancy calculation directly runs on the digitisation output.

7 Software for the TPC time stamping study

7.1 Simulation

- PrimaryIonisationProcessor

7.2 Digitisation

- DriftProcessor
- GEMProcessor
- ChargeDistributionProcessor
- TPCElectronicsProcessor

7.3 Reconstruction

- PulseFinderProcessor
- HitTrackFinderTopoProcessor
- TrackSeederProcessor
  First run to calculate the initial track parameters before outlier rejection.

[3] The incoherent pairs consist of 300,000 Mokka events per bunch crossing, which result in three input LCIO input files per bunch crossing. It was impracticable to use the OverlayProcessor with this. The OverlayIncoherentPairsProcessor reads in three files per bunch crossing and puts all events in them into one LCIO event. The overlaying functionality is the same as the OverlayProcessor.

- `HitsInTracksSplitterProcessor`
- `OutlierRejectionProcessor`
- `TrackSeederProcessor`
  Second run to calculate the final parameters after outlier rejection.

8 Acknowledgements

Many thanks to Steve Aplin for his help extending the Mokka driver.

References

[1] M. Killenberg. Occupancy in the CLIC_ILD Time Projection Chamber. CERN LCD-Note-2011-029, 2011.

[2] M. Killenberg. Time stamping of TPC track in the CLIC_ILD detector. CERN LCD-Note-2011-030, 2011.

[3] MarlinTPC, Marlin based simulation, digitisation, reconstruction and analysis code for the TPC. Website: `http://ilcsoft.desy.de/portal/software_packages/marlintpc/`.

[4] S. Agostinelli et al. Geant4 – A Simulation Toolkit. *Nucl. Instrum. Methods Phys. Res., Sect. A*, vol. 506(3) pp. 250–303, 2003.

[5] Mokka, a detailed Geant4 simulation for the International Linear Collider detectors. Website: `http://ilcsoft.desy.de/portal/software_packages/mokka/`.

[6] MarlinReco, a Marlin based reconstruction software. Website: `http://ilcsoft.desy.de/portal/software_packages/marlinreco/`.

[7] D. Dannheim and A. Sailer. Beam-induced backgrounds in the CLIC detectors. LCD-Note-2011-021, 2011.

[8] I. B. Smirnov. Modeling of ionization produced by fast charged particles in gases. *Nucl. Instrum. Meth.*, vol. A554 pp. 474–493, 2005.

[9] A. Münnich. *Simulation Studies for a High Resolution Time Projection Chamber at the International Linear Collider*. Ph.D. thesis, RWTH Aachen University, Germany, 2007. urn:nbn:de:hbz:82-opus-18933.

[10] F. Sauli. GEM: A new concept for electron amplification in gas detectors. *Nucl. Instrum. Meth.*, vol. A386 pp. 531–534, 1997.

[11] T. Krautscheid. *Simulationen zur Ionenrückdrift in einer Zeit-Projektionskammer am ILC*. Diploma thesis, Bonn University, Germany, 2008.

[12] L. Musa et al. The ALICE TPC front end electronics. *Proc. of the IEEE Nuclear Science Symposium*, 2003.

[13] The Marlin Overlay processor. Website: `http://ilcsoft.desy.de/portal/software_packages/overlay/`.

[14] S. Lotze. *Ion Backdrift Minimisation in a GEM-Based TPC Readout*. Ph.D. thesis, RWTH Aachen University, Germany, 2006. urn:nbn:de:hbz:82-opus-14999.

[15] B. Sobloher. *Simulationsstudien zu GEM-Folien für die Auslese einer TPC*. Diploma thesis, RWTH Aachen University, Germany, 2002.