CMS DN-13-022

The Compact Muon Solenoid Experiment

Detector Note

The content of this note is intended for CMS internal use and distribution only

2016/07/15

ME1/1 Local Trigger Algorithm for Operation under High Pile-Up Running Conditions

Sven Dildick[1], Jose Roberto Dimas Valle[1], Jason Gilmore[1], Jay Hauser[2], Tao Huang[1], Vadim Khotilovich[1], Vyacheslav Krutelyov[1], Alexander Madorsky[3], Yuriy Pakhotin[1], Andrew Peck[2], Alexei Safonov[1], Aysen Tatarinov[1], and Vyacheslav Valuev[2]

[1] Texas A&M University
[2] University of California, Los Angeles
[3] University of Florida

Abstract

An important piece of the CMS muon trigger upgrade is developing and maintaining FPGA firmware for new custom electronics - optical trigger mother-board for ME1/1 muon detectors. This firmware uses massively parallel pattern recognition algorithm to find quality muon stubs. An improved local trigger algorithm to perform sophisticated background rejection is developed for operation under high pile-up running conditions. Implementation of the improvements to the algorithm is described. Configuration of the CMS software for the algorithm simulation is specified. Individual effects of the improvements on the ME1/1 local trigger efficiency are quantified and ranked using simulated samples.

Contents

1 Introduction

Muon system is hosted in the steel yokes of the CMS detector [1] and it is divided into a central part (barrel: $|\eta| < 1.2$) with Drift-Tube (DT) detectors and two forward parts (endcaps: $0.9 < |\eta| < 2.4$) with Cathode Strip Chambers (CSC) as shown in Fig. 1. Resistive Plate Chambers (RPC) are located in barrel and endcap parts ($|\eta| < 1.9$). The detectors of the muon system identify muons, provide a fast muon trigger, and give a precise measurement of the muon trajectory. Performance of the CMS muon system in LHC Run1 is described in [2].

The muon trigger is a tracking trigger that determines the momentum of muons using hits (position and angular measurements) in the muon system chambers situated in the magnetic field of the CMS detector. In this paper we describe several major improvements and upgrades to the muons system and their effect on the muon trigger performance. The focus of the muon trigger upgrade is to improve its rate reduction capability without significantly affecting the efficiency. We overview implementation of the muon trigger in endcaps in Section 2.

Installation of the outermost ring of CSCs in the fourth disk of each endcap (ME4/2) during 2013-2015 shutdown of the LHC (Long Shutdown 1 or LS1) allows to increase the number of muon hits along its trajectory. Major revision of the electronics for innermost ring of CSCs in the first disk (ME1/1) and unganging strips in the bottom part of these chambers allow to significantly enhance their performance in the trigger and in offline reconstruction. We describe details of the CSC upgrade during LS1 in Section 3.

New, robust and sophisticated, ME1/1 local trigger algorithm which is tolerant of the increased pile-up was also developed during LS1. Individual improvements of the upgraded algorithm

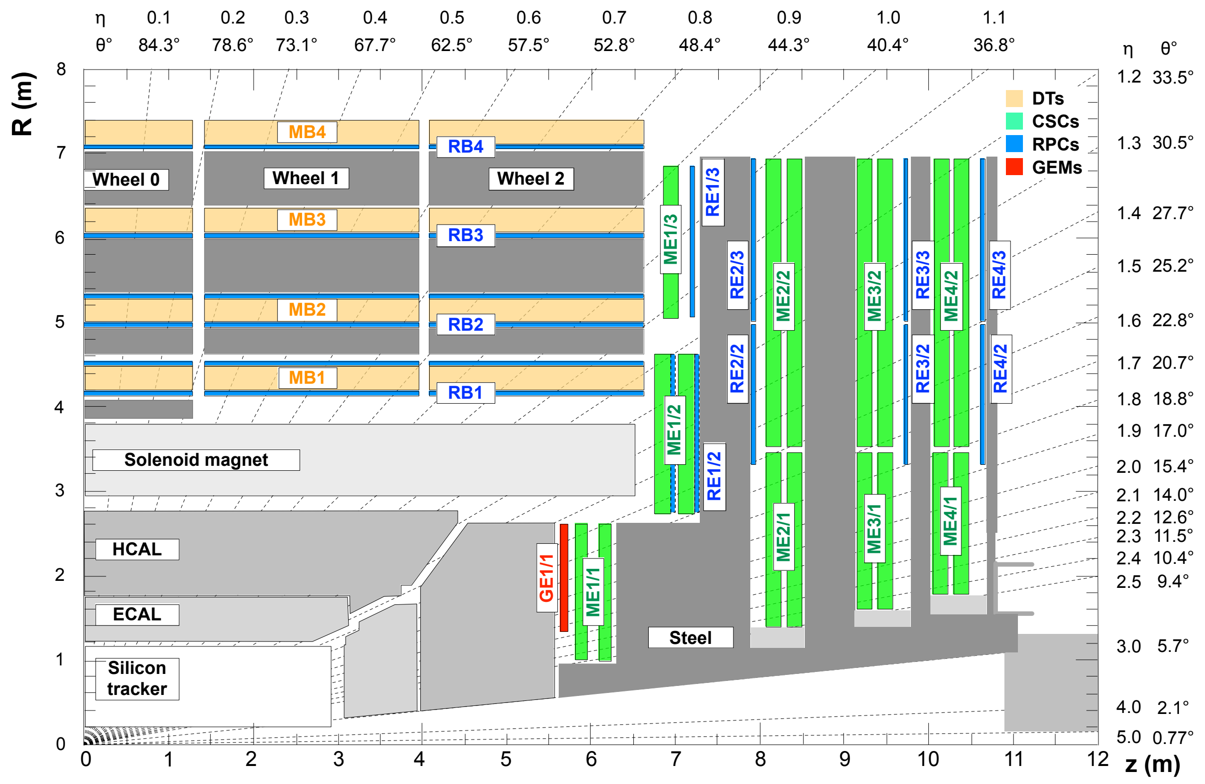


Figure 1: Quarter-view of the CMS cross section. Detectors in the muon system are highlighted: DT – orange, CSC – green, RPC – blue and future GE1/1 – red.

are studied in details using simulation, and ranked in terms of their importance. The goal is to have improved ME1/1 local trigger algorithm commissioned during the 2015-16 Year-End Technical Stop (YETS), so that the upgraded trigger will be available to operate under high pile-up running conditions in 2016. Overview of the CSC local trigger algorithm is in Section 4. Specifics of the previous and upgraded algorithms implementation in electronics firmware, emulation in CMS software, as well as results of individual improvements are described in three sections:

1. anode signal processing (see Section 5),
2. cathode signal processing (see Section 6),
3. cathode-anode correlation (see Section 7).

As a part of future upgrade for Phase II of LHC running, the CMS will be equipped with Gas Electron Multiplier (GEM) detectors in the high pseudorapidity region ($1.5 < |\eta| < 2.2$) as shown in Fig. 1. Pairs of triple-GEM chambers will be installed in the currently vacant position in front of the ME1/1 chambers and are dubbed GE1/1. The addition of such chambers allows to measure the bending angle of a track between GE1/1 and ME1/1. Usage of the bending angle at L1 can help to keep the rates down while having the efficiency high. Several implementation possibilities of the combined GEM-CSC local trigger algorithm for the high luminosity run of LHC are investigated in details in [3].

2 Muon Trigger in Endcaps

Each CSC can provide up to two local charged track (LCT) segments to the trigger logic per BX. These are formed in the trigger motherboard (TMB) combining cathode (CLCT) and anode (ALCT) segments. Present and improved logics of the algorithm which constructs LCTs are described in Section **??** and Section **??**, respectively. The CLCT data contains information on the azimuthal position of the segment (ϕ), the bend angle, and the pattern of cathode half-strips with hits in a chamber. The ALCT data contains information on the radial position from the beamline of the segment (equivalent to η), and the pattern of anode wires with hits in a chamber. The timing information from anodes is used to define the time of the combined LCT. There is one TMB per CSC, located in a crate on the periphery of the detector. The TMB sends up to two LCTs over a custom backplane to the muon port card (MPC), which is located in the same peripheral crate. One MPC can receive data from up to 9 TMBs, or equivalently, can receive up to 18 LCTs. The LCTs in a MPC are sorted by rank (see definition in MPC documentation). The best three LCTs are sent over optical fibers to the CSCTF. There are a total of sixty peripheral crates for the CSC system, each with one MPC.

The CSCTF system is partitioned into sectors, each of which corresponds to a 60^o azimuthal region of an endcap. Twelve "sector processors" are required for the entire endcap muon system, six per endcap. Each sector processor is a 9U VME card that is housed in a single crate. Three 1.6 Gb/s optical links from each of five MPCs are received by each sector processor, for a total of 180 optical links for the entire system. The CSCTF sectors are independent, since there is no sharing of data across boundaries of neighboring sectors, leading to slight inefficiencies.

There are several Field Programmable Gate Arrays (FPGAs) on each "sector processor", but the main FPGA for the track-finding algorithms is from the Xilinx Virtex-5 family. The conversion of strip and wire positions of each track segment to (η, ϕ) coordinates is accomplished via a set of cascaded SRAM look-up tables (LUTs), each $512K \times 16$ bits. These coordinates are then used for track-finding and momentum assignment.

The CSCTF track-finding logic consists of pairwise comparisons of track segments in different detector stations. These test for compatibility in ϕ and η with a muon emanating from the collision vertex within certain tolerance windows. The comparisons are then analyzed and built into tracks consisting of possibly more than two segments from different stations. Possible duplicate (?ghost?) tracks are canceled. The track-finding logic has the ability to accept segments in different assigned bunch crossings by analyzing across a sliding time window of programmable length (nominally 2 BX) every bunch crossing. Duplicate tracks found on consecutive bunch crossings are canceled. The bunch crossing of a track is given by the second arriving track segment.

The p_T of a muon candidate is calculated by using a large LUT implemented in SRAM. Information such as the track type, track η, the segment ϕ differences between a maximum of 3 stations, and the segment bend angle in the first measurement station are used to calculate the LUT address.

In addition to identifying muons from proton collisions, the CSCTF processors also simultaneously identifies any beam halo muons for monitoring and veto purposes by looking for trajectories approximately parallel to the beam line.

Each CSCTF sends up to three muon candidates per bunch crossing over a custom backplane to a muon sorter (MS). The MS then sorts the candidates by momentum and quality and selects the best 4 for the GMT. The CSCTF data are also sent to a DAQ card with SLINK interface which puts the trigger data into the event record.

3 Endcap Muon System Upgrade

All CSCs in ME4/2 rings have been installed during LS1. This results in four measurement stations for muons in the region $1.25 < |\eta| < 1.8$ providing additional redundancy in a high rate environment. This redundancy is especially important for future upgraded Global Muon Trigger (GMT) algorithms. For the CSC Track Finder (TF) additional measurement in this region will increase the efficiency and improve the rate reduction since it will be more likely to have 3 or more hits used in the p_T assignment logic. No additional hardware or reconfiguration of the present muon trigger was required after this upgrade. The muon sector receiver boards for the fourth disk already were in place and the present CSCTF already had logic to process trigger data from these chambers.

Electronics for the CSCs have been also under major revision during LS1. All CSCs in ME1/1 rings received new digital cathode front-end boards (DCFEB) as well as new optical trigger motherboards (OTMB) and optical data acquisition motherboars (ODMB) as schematically shown in Fig. 2. The new electronics will significantly enhance ME1/1 performance in the trigger and in the offline reconstruction providing a key sagitta measurement for the muon L1 trigger in the region $1.6 < |\eta| < 2.4$. The recovered old electronic boards were used to instrument the newly installed ME4/2 chambers.

The strips of the ME1/1 chambers are split into two regions at $|\eta| = 2.1$ (see details on mechanical layout and signal readout of the ME1/1 chambers in Appendix A). The bottom region ($2.1 < |\eta| < 2.4$) previously had 48 strips triple-ganged to 16 channels in the electronics for both the trigger and the readout, making hit recognition ambiguous. The ambiguity can be mitigated using measurements from the outer stations. However, the p_T resolution using only the outer stations is quite coarse, leading to a significantly increased single muon trigger rate in the forward region $2.1 < |\eta| < 2.4$. As a result, this region generated a single muon trigger rate comparable to that of the entire region $|\eta| < 2.1$. With the new seven DCFEBs per chamber, this triple-ganging is removed, leading to improved triggering performance in the forward region which allows to maintain highly efficient muon trigger coverage up to $|\eta| = 2.4$.

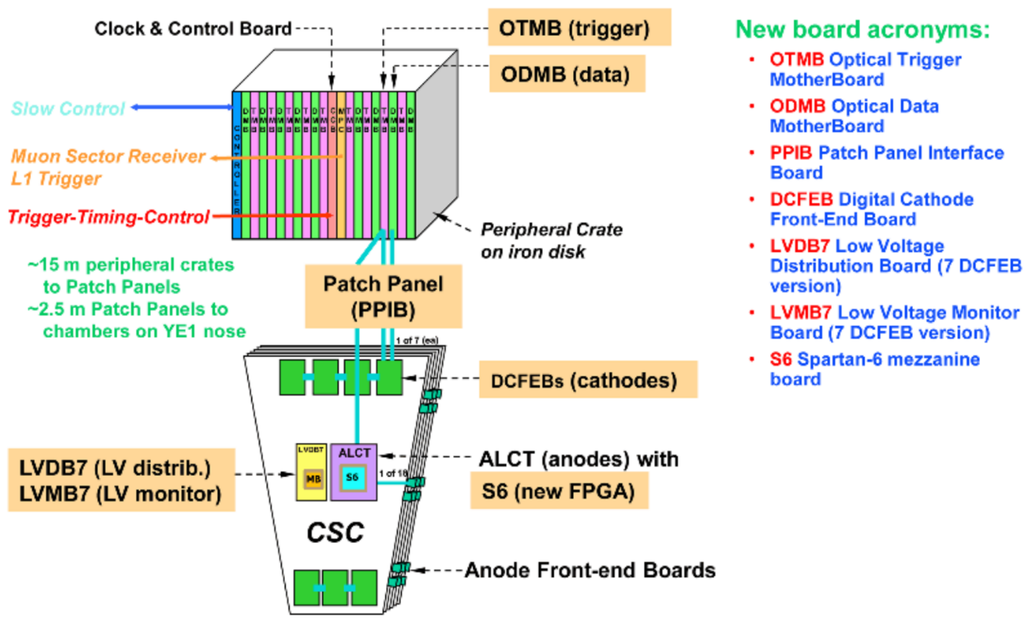


Figure 2: Schematic overview of the ME1/1 electronics upgrade after LS1.

4 Overview of the CSC Local Trigger Algorithms

An improved ME1/1 local trigger algorithm to perform sophisticated background rejection is developed for operation under high pile-up running conditions.

4.1 Results of Improvements in the ME1/1 Local Trigger Algorithm

This chapter presents results of the study of effects of individual improvements described in Sec. **??** on the ALCT, CLCT, and LCT reconstruction efficiencies.

The study is performed with Monte Carlo simulation of double muon events mixed with PU400 events, where the simulation includes GEN, SIM, DIGI, L1 steps.

Three are three baseline configurations of L1 step used in this study:

- Baseline 1: SLHC configuration, where the maximum set of improvements is turned off bringing it to 2007 configuration as close as possible. There are only two differences between Baseline 1 and 2007 configurations: separate treatments of ME1/1a and ME1/1b, and unganging cathode strips in ME1/1a;
- Baseline 2: Baseline 1 configuration with all improvements on the ALCT and CLCT processors level turned on;
- Baseline SLHC: best CSC4 configuration itself.

All algorithm improvements divided into three groups and studied with improvements in the given group turned on one by one on top of each other

- ALCT processor level
- CLCT processor level
- TMB level

In the first two groups the L1 step configuration gradually changes from Baseline 1 to Baseline 2 configuration, in the last one — from Baseline 2 to SLHC configuration.

5 Anode Signal Processing

5.1 ALCT Algorithm

Anode wires in CSC are hardwired together at the readout end in groups of 10-15 wires in order to reduce channel count. The anode wire group (WG) signals are fed into the anode front-end boards (AFEBs), each of which contains a single 16-channel amplifier/constant-fraction discriminator chip. The output signals from the AFEBs are sent into the on-chamber ALCT board, which handles triggering and readout of the CSC anode information. Due to the various sizes of CSCs, there are 3 types of ALCT boards, handling 288, 384, and 672 WG channels.

On the ALCT boards, the signals from each AFEB are first delayed by a programmable amount of time in order to perform an average time alignment of the anode signals across the chamber as well as chamber-to-chamber at a sub-bunch crossing level to about 2.2 ns precision. After the AFEB signals are received and time-aligned, then they are latched with bunch crossing frequency and fed to a FPGA (Xilinx Virtex family) mounted on a mezzanine card above the ALCT main board for pattern-finding and readout functions.

The algorithm used in the ALCT FPGA for determining muon segment position and bunch crossing is illustrated below. Since the drift time can be longer than 50 ns, the hits are first stretched by 'one-shots' to 6 BX (150 ns) length. Then, a multi-layer coincidence technique in the ALCT pattern circuitry is used to identify the bunch crossing. For each spatial pattern of anode hits, a low coincidence level, typically 2 or more layers, is used to establish timing, whereas a higher coincidence level, typically 4 layers, is used to establish the existence of a muon track. The general idea of a spatial pattern of CSC wire group hits is illustrated below in Fig.3.

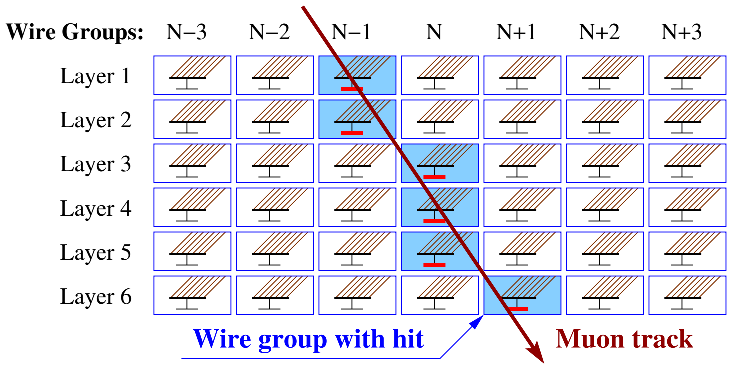


Figure 3: Illustration of CSC anode wire groups with hits from muon track

while the general idea of the time stretching of hits, and pretrigger followed by a pattern trigger is shown in Fig.4 (using an example in which one hit is actually missing due to some type of inefficiency).

Each pattern detector can detect a programmable "collision" pattern as well as a fixed "accelerator" pattern. The input data for the collision pattern detector are selected as shown below:

```
...n-2 n-1 n...........Layer 1
.......n-1 n...........Layer 2
...........n...........Layer 3
...........n n+1.......Layer 4
...........n n+1 n+2...Layer 5
```

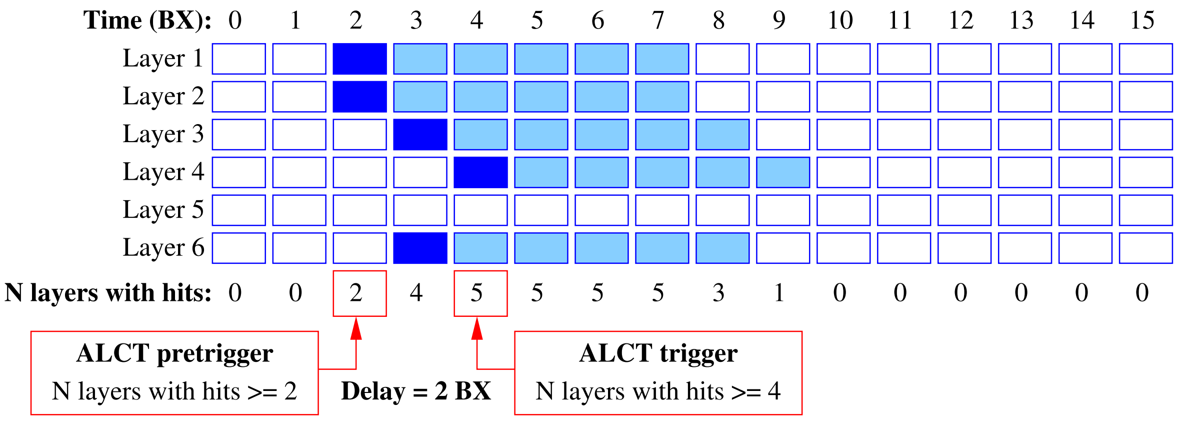


Figure 4: Illustration of anode hits time stretching (6BX) as well as determination of ALCT pretrigger and trigger.

```
..........n n+1 n+2...Layer 6
```

where n in this diagram is the key wire group number, which this particular pattern detector is searching the patterns for. The programming of the programmable collision pattern is implemented as a simple masking-out of the bits that we do not want to include in the pattern. The accelerator pattern is a vertical pattern of 6 layers all with strip n only.

Each ALCT candidate is assigned a quality equal to number of layers with hits minus 3 and passes through a ghost cancellation procedure: it is cancelled if there is another ALCT candidate at the same bunch crossing in the previous wirewith the same or better quality or in the next wire with better quality, or if there is ALCT candidate up to 4 bunch crossing clocks earlier with any quality.

In each bunch crossing two ALCTs with highest quality are sent to the TMB (Trigger MotherBoard), which requires a coincidence between anode and cathode trigger information. In the case of a Level-1 Accept signal from the Global Trigger (distributed via the TTC system to the CCB in each peripheral crate), ALCT data are sequentially transmitted to the Trigger Mother Board and hence to the DAQ Motherboard. These data frames include a few words of ALCT trigger data and a much larger amount of ALCT raw hit data consisting of a time sequence of raw CSC anode wire-group hits that have been stored at the 40 MHz bunch crossing frequence by the ALCT2001. Typically 8 to 16 bunch crossings are read out for each wire group. FIFO data can also be read out much more slowly through VME access via the TMB board using a JTAG electrical interface to the ALCT, if necessary.

For self-monitoring and also for powering and controlling the AFEB cards, the ALCT contains a Slow Control section that supplies power to the AFEBs, controls AFEB thresholds, provides and controls the amplitude of test pulses to the AFEBs, and reads back power supply voltages and currents, as well as on-board temperature.

5.2 Improvement of the ALCT Algorithm

It should be possible to improve the efficiency, rate and timing precision of ALCT stubs by, e.g.

- tuning of the ghost cancellation logic (ALCTs in neighboring wiregroups, see ”alctGhostCancellationBxDepth” and ”alctGhostCancellationSideQuality” parameters in Sec. B.2) and removing pre-trigger deadtime (see ”alctPretrigDeadtime” parameter in Sec. B.2);

- using more narrow ALCT pattern in ring 1 chambers (see ”alctNarrowMaskForR1” parameter in Sec. B.2);
- using more precise algorithm (e.g., running median or truncated average) for BX assignment (see ”alctUseCorrectedBx” parameter in Sec. B.2).

However, here we would only like to focus on how the ALCT stubs would be used by TMB.

ALCTs are reconstructed from the signals in layers of anode wires which are ganged into wiregroups and are continuously covering the whole ME1/1 chamber. Most of the wiregroups can only physically cross only strips either in ME1/1a or only in ME1/1b. A complication specific to ME1/1 is that wires here are not perpendicular to strips, but are slanted at 29 degrees from the straight angle. Thus, some wiregroups are crossing the border between ME1/1a and ME1/1b. If signal is deterced in such a wiregroup, there is an ambiguity about which part of ME1/1 it might belong to.

For the ALCTs received by TMB we propose to split the incoming stubs into two parts, ME1/1a and ME1/1b, that would be used further for LCT matching separately in ME1/1a and ME1/1b.

5.3 Emulation of ALCT Algorithm

ALCT processing includes the following five steps:

- Pulse extension;
- Pretrigger;
- Trigger;
- Ghost cancellation;
- ALCT construction.

5.3.1 Pulse Extension

Sofware emulation provides information about all wire signals in DAQ readout window (16 BXs). A search for these signals is performed in a loop over all wire groups, all layers, and all 16 BXs; found signals are stretched over 6 BXs (see Fig. 5).

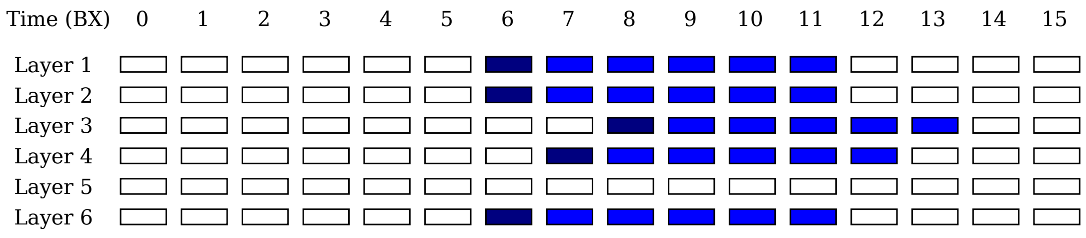


Figure 5: Illustration of ALCT pulse extension for one specific wire group.

5.3.2 Pretrigger

After all available wire signals are stretched, a search for ALCT pretriggers is performed in all wire groups and all BXs. For any given wire group and BX, we count the number of layers with hits within the pattern mask shown on Fig. 6, and if this number is greater than or equal to three, then we say that a pretrigger occured in this wire group and BX. The search for next ALCT pretrigger starts 6 BXs later.

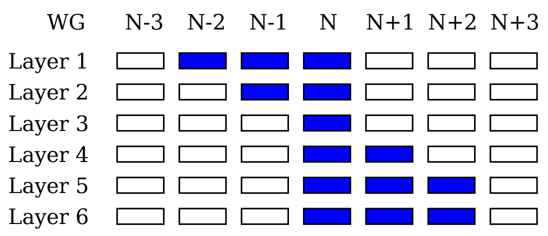


Figure 6: ALCT pattern mask for pretriggering and triggering.

5.3.3 Trigger

After all ALCT pretriggers are found, for each pretrigger in BX = B we check for a trigger in BX = B+2. For any given wire group and BX = B+2, we count the number of layers with hits within the same pattern mask used for pretriggering, and if this number is greater than or equal to four, we say that a pretrigger occured in this wire group and BX, and assign it a quality Q = number of layers with hits-3. If in some wire group more than one trigger occured, we report only the one with the highest quality. If there are two triggers with the same quality, report the earlier one.

5.3.4 Ghost Cancellation

Not all triggers found in the previous step are used to construct ALCTs: before that all of them pass through so called ghost cancellation procedure.

A trigger in wire group = N and BX = B is cancelled if there is a trigger in wire group = N-1:

- either in the same BX = B and with better or equal quality;
- or to 4 BXs earlier, with any quality.

In addition, a trigger in wire group = N and BX = B is cancelled if there is a trigger in wire group = N+1:

- either in the same BX = B and with better quality;
- or to 4 BXs earlier, with any quality.

5.3.5 ALCT Construction

Construct ALCTs from triggers survived after the ghost cancellation procedure: encode quality, WG, BX (defined by pretrigger BX). In every BX choose best two ALCTs: two ALCTs with the highest quality. If we need to choose one ALCT from two ALCTs with the same quality: choose the one with larger wire group.

5.4 Sofware Emulation of ALCT Level Improvements

5.4.1 Tuning of Ghost Cancellation Procedure

Current ghost cancellation:

- Loop over wire groups:
  - Consider WG = N
  - Cancel trigger in this wire group if there is trigger in WG = N-1 and:
    - with the same BX and with better or equal quality
    - up to 4 BXs earlier, with any quality

- Cancel trigger in this wire group if there is trigger in WG = N+1 and:
    - with the same BX and with better quality
    - up to 4 BXs earlier, with any quality

New ghost cancellation:

- Loop over wire groups:
    - Consider WG = N
    - Cancel trigger in this wire group if there is trigger in WG = N-1 and:
        - with the same BX and with better quality
        - up to 1 BX earlier, with better quality
    - Cancel trigger in this wire group if there is trigger in WG = N+1 and:
        - with the same BX and with better or equal quality
        - up to 1 BX earlier, with better and equal quality

The following modifications in configuration are related to this improvement:

- alctGhostCancellationBxDepth: 4BX to 1BX
- alctGhostCancellationSideQuality: False to True

5.4.2 Narrow ALCT Pattern Mask

Use more narrow ALCT pattern mask for stations in Ring 1 (see Fig. 7).

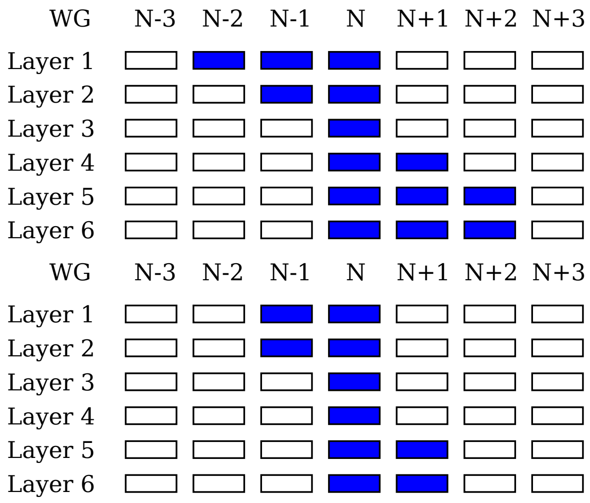


Figure 7: Top: default ALCT pattern mask, bottom: narrow ALCT pattern mask.

The following modifications in configuration are related to this improvement:

- alctNarrowMaskForR1: False to True

5.4.3 Reduced ALCT Dead Time

Currently, if there is pretrigger in BX = B (see Fig. 8):

- Check for trigger in BX = B + drift time = B+2
- Search for next pretrigger starting from BX = B + drift time + extra deadtime = B+6

Suggested improvement: decrease extra deadtime from 4 BX to 0 BX.

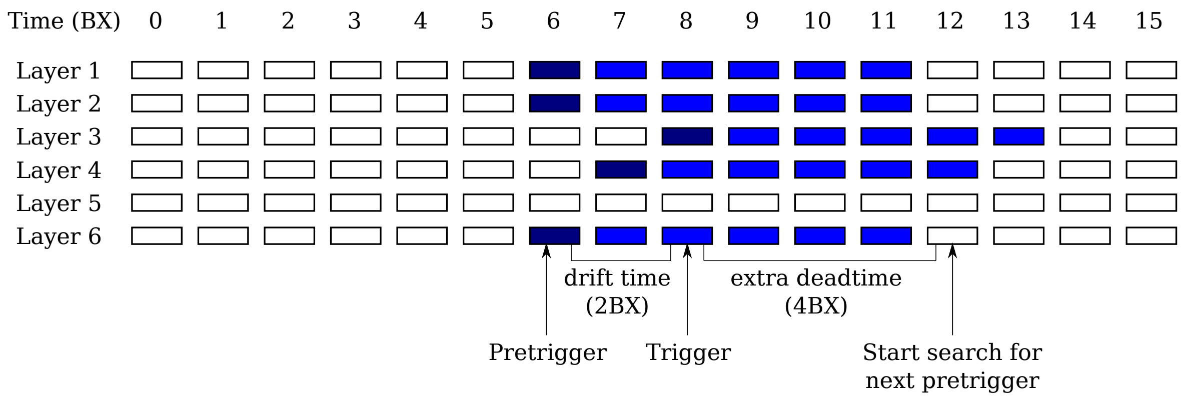


Figure 8: Top: Dead Time between ALCT pretriggers.

The following modifications in configuration are related to this improvement:

- alctPretrigDeadtime: 4BX to 0BX

5.5 Results of Improvements of the ALCT Processing

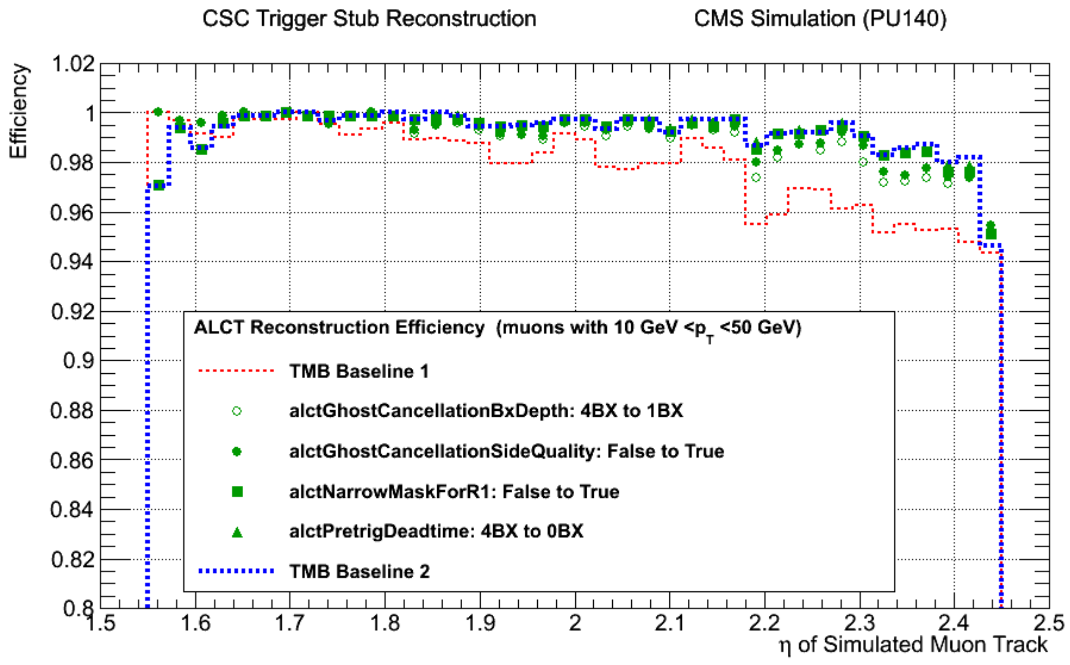
Improvements on the level of ALCT processor are related to the following configuration parameters (see Sec. B.2):

- alctGhostCancellationBxDepth: 4BX to 1BX;
- alctGhostCancellationSideQuality: False to True;
- alctNarrowMaskForR1: False to True;
- alctPretrigDeadtime: 4BX to 0BX.

Fig. 9 shows reconstruction efficiency of a good ALCT in ME1/1 station versus pseudorapidity of the simulated muon for different L1 configurations. The good ALCT is defined as ALCT:

- read out in the window of 3BX around the central BX (BX6);
- reconstructed within 2 anode wire groups from the key wire group.
- has hits at least on four layers

The major improvement in ALCT reconstruction efficiency comes from the changes in ALCT ghost cancellation procedure.



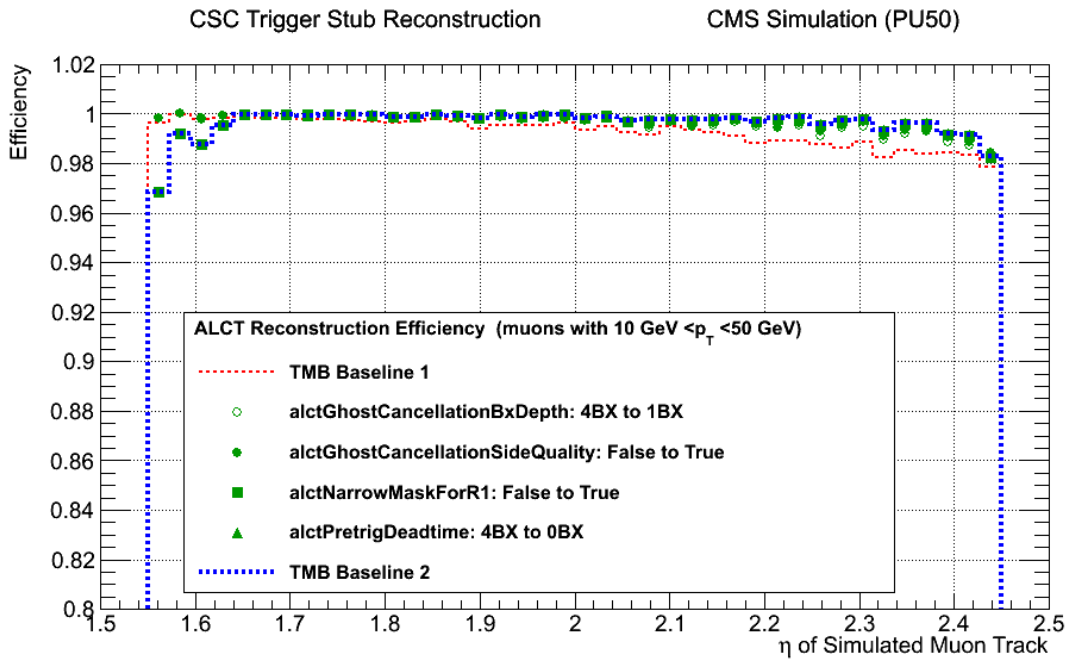


Figure 9: ALCT reconstruction efficiency in ME1/1 station for PU140 (top) and PU50 (bottom). Muons with transverse momentum 10 GeV$< p_T < 50$ GeV are used in the analysis.

6 Cathode Signal Processing

6.1 CLCT Algorithm

[We need a good picture illustrating CLCT processing process like in the case of ALCT one]

A muon passing through a CSC chamber will produce distinctive patterns of half-strip hits in the six-layer endcap muon CSC chambers. By identifying these patterns, the CSC Local Trigger provides high rejection power against backgrounds. The largest background source, neutron-induced gamma ray conversions, are generally low in energy, and produce mostly single- layer or short multi-layer hits. Other backgrounds, such as low-momentum muons or punch- through particles often do not point well enough to the primary interaction region to be considered high-momentum muon candidates.

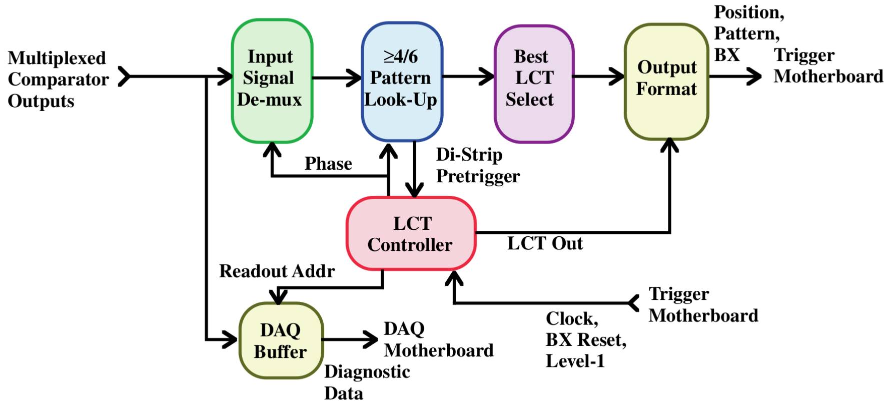


Figure 10: CLCT block diagram

[Technical description from TMB manual below]

For each of 160 key half-strips consider the 42 neighboring half-strips (i.e. on key 5 use the following half-strips):

```
hs              0123456789A
ly0[10:0]       xxxxxkxxxxx    5+1+5 =11
ly1[ 7:3]          xxkxx       2+1+2 = 5
ly2[ 5:5]            k         0+1+0 = 1
ly3[ 7:3]          xxkxx       2+1+2 = 5
ly4[ 9:1]        xxxxkxxxx     4+1+4 = 9
ly5[10:0]       xxxxxkxxxxx    5+1+5 =11
```

Figure 11: 160 key half-strips

For each of 160 key half-strips, count layers with hits matching the 9 pattern templates:

Pattern ID=1 is a layer-OR trigger, Pattern ID=0 is no-pattern-found. Result for each of 160 keys is a list of 9 pattern-ID numbers (pid) [2 to A] and corresponding number of layers [0 to 6] with matching hits (nhits). Find the best 1-of-9 pattern ID numbers for each key by comparing nhits. Ignore bend direction: left and right bends have equal priority (bit 0 of pid implies bend direction). If two pattern IDs have the same nhits, take the higher pattern ID. A key with no matching hits, would always return pid=A and nhits=0. Pre-trigger if any 1-of-160 keys have nhits $\geq$ hit_thresh_pretrig and pid $\geq$ pid_thresh_pretrig.

Construct 7-bit pattern quality pat[7:0] for sorting where pat[7:5]=nhits[2:0], pat[4:0]=pid[3:0]. Ignore the bend direction bit (pid[0]), left and right bends have equal priority. Store pat[7:0]

```
Hit pattern LUTs for 1 layer: - = don’t care, xx= one hit or the other or both
Pattern       id=2        id=3        id=4        id=5        id=6        id=7        id=8        id=9        idA
Bend dir      bd=0        bd=1        bd=0        bd=1        bd=0        bd=1        bd=0        bd=1        bd=0
              |           |           |           |           |           |           |           |           |
ly0      --------xxx xxx-------- -------xxx- -xxx------- ------xxx-- --xxx------ -----xxx--- ---xxx----- ----xxx----
ly1      ------xx--- ---xx------ ------xx--- ---xx------ -----xx---- ----xx----- -----xx---- ----xx----- -----x-----
ly2 key  -----x----- -----x----- -----x----- -----x----- -----x----- -----x----- -----x----- -----x----- -----x-----
ly3      ---xxx----- -----xxx--- ---xx------ ------xx--- ----xx----- -----xx---- ----xx----- -----xx---- -----x-----
ly4      -xxx------- -------xxx- -xxx------- -------xxx- ---xx------ ------xx--- ---xxx----- -----xxx--- ----xxx----
ly5      xxx-------- --------xxx -xxx------- -------xxx- --xxx------ ------xxx-- ---xxx----- -----xxx--- ----xxx----
              |           |           |           |           |           |           |           |           |
// Extent   0123456789A 0123456789A 0123456789A 0123456789A 0123456789A 0123456789A 0123456789A 0123456789A 0123456789A
// Avg.bend - 8.0 hs    + 8.0 hs    -6.0 hs     +6.0 hs     -4.0 hs     +4.0 hs     -2.0 hs     +2.0 hs      0.0 hs
// Min.bend -10.0 hs    + 6.0 hs    -8.0 hs     +4.0 hs     -6.0 hs     +2.0 hs     -4.0 hs      0.0 hs     -1.0 hs
// Max.bend - 6.0 hs    +10.0 hs    -4.0 hs     +8.0 hs     -2.0 hs     +6.0 hs      0.0 hs     +4.0 hs     +1.0 hs
```

Figure 12: CLCT patterns

for 160 keys for use later to find 2nd CLCT. Find the best key out 1-of-160 keys by sorting on the 6-bit number pat[7:1]. Store 1st CLCT info: key, pattern ID, and number of hits. For empty events, key=0, pid=A and nhits=0. If clct_blanking=1, then key=pid=hits=0.

Mark keys near 1st CLCT as busy from 1st key-nspan to 1st key+pspan. If clct_sep_src=1, pspan and nspan are set equal to clct_sep_vme, typically 10 half-strips. If clct_sep_src=0, pspan and nspan are read from RAM and depend on the pattern ID number, this allows two less bending tracks to be closer than more bending tracks.

Find the best key out of 1-of-160 keys by sorting on the 6-bit number pat[7:1]: skip busy keys, if two keys have the same pat[7:1] take the lower key. Store the same information for the 2nd CLCT as for the 1st one.

Wait for CSC drifting (drift delay of 2BXs) and perform matching to ALCTs.

6.2 Separation of the CLCTs in ME1/1a and ME1/1b

In the old TMB f/w, the 16 ganged strip channels from ME1/1a are appended to the 64 ME1/1b strip channels and the reconstruction of CLCTs is performed as if it’s a single regular chamber with 80 channels (is there any treatment of the boundary, so that ME1/1a strips are not combined with ME1/1b strips?). The two rather separate and distinctive areas combined and treated like one uniform unit with the maximum of two CLCT stubs on the output.

With unganged ME1/1a and higher pile-up such a simple approach becomes increasingly unnatural and ineffective. The ME1/1a would have x3 more channels now, so it would deserve to be treated like a separate chamber even more. And chances to get multiple stubs are increasing with higher luminosity, especially so in ME1/1a. With multiple stubs, the stubs in ME1/1a would start directly competing with those in ME1/1b.

With a larger size FPGA, it would be beneficial to treat ME1/1a and ME1/1b as two separate chambers for the purpose of CLCT reconstruction, with each area having their own limit on the maximum of 2 CLCTs. Thus, the whole ME1/1 would be able to have maximum 4 CLCTs available for matching with ALCTs. It’s important to keep more 2D stubs available so that at the stage of matching we can reduce the number of stubs following some better tuned criteria.

Finally, in the case if we would allow to read out the 2D CLCT stubs without an ALCT match, and we cannot read out more then 2 of them per BX per ME1/1, we can device a selection criteria when comparing stubs from ME1/1a and ME1/1b as follows: e.g., if quality of an ME1/1b stub is the same or less by one then that of an ME1/1a stub, prefer the ME1/1b one.

6.3 Localization of the TMB Dead Time

The main source of the old TMB inefficiency in high pile-up is the dead time which happens for the whole ME1/1 chamber after there was a triggering CLCT. The TMB's state-machine freezes whole TMB for several BXs after a CLCT trigger while number of coincidence layers stays over the trigger threshold. If anywhere in a chamber there was an CLCT from PU a few BX earlier before a signal muon, it would be impossible to trigger on the signal.

While it's clear that with the comparator information which we recieve in TMB, it's not really possible to distinguish close in time signals in the same strip, there seem to be no apparent reason, other then the complexity of the algorithm, for this dead time to be present in strips that had no trigger.

Thus, the algorithm approach to deal with this issue for the upgrade could be as follows:

- when a CLCT trigger happens, mark as busy only those strips within either a fixed dead time zone around CLCT (8 half-strips, see ″useDeadTimeZoning″ parameter in Sec. B.3) or within the dead time zone, the width of which depends on the CLCT pattern (from 11 half-strips for the most bent patterns to 3 half-strips for the straightest one, see ″useDynamicStateMachineZone″ parameter in Sec. B.3), and also mark the signal half-strip that specifies the triggered CLCT
- during the following bunch-crossings, check if the number of coincidence layers drops under the trigger threshold for the signal half-strip
  - if it does, remove the ″busy″ mark from the corresponding strips
- if strips are marked as busy in a BX, they are excluded from pattern recognition

6.4 Restriction on the CLCT Pattern Bend

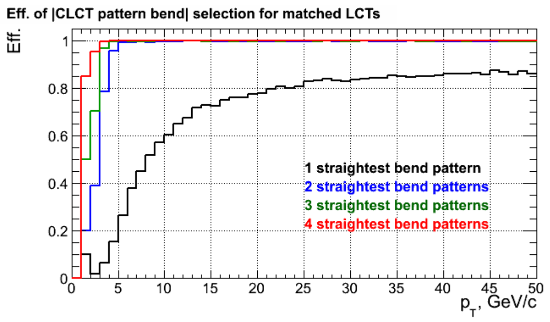
The current set of CLCT patterns (see details in Sec. [**?**]) is largely geared towards low pt tracks. For tracks with pt¿10, only the straighest and the next one bent patterns are significant. The restriction on allowed pattern bending would significantly help with many issues including multiplicity & rate, ghosting, dead-time and corresponding loss of the efficiency (see ″clctPidThreshPretrig″ parameter in Sec. B.3). Positive effect of it would increase with increasing luminosity. However, it would also somewhat reduce the efficiency because of not so good pt-threshold resolution from fairly narrow (11cm) CSC chambers, especially for medium to lower pt muons.

Plots on Fig. 13 show the efficiency vs p_T for a simulated muon track to have a matching reconstructed CLCT stub for different sets of patterns (which defines maximum thresholds for the bend). Note, that adding GEMs could be helpful for improving the the efficiency of the bend restriction.

6.5 Improvement of the CLCT Timing

Currently, the BX time of a CLCT stub is defined as the BX of its pretrigger. Or, in other words, it's first BX when at least three layers fit one of the CLCT patterns. Note that an attempt to latch a trigger pattern is performed after the number of BX after the pretrigger defined by the clctDriftDelay parameter (=2BX). And also note that a CLCT pattern during the pretrigger could be different then the one that happen to match during the trigger.

With higher luminosity there are increasingly larger chances for some strips in some of the layers to be hit by earlier background hit. And that has chances to affect the time when a pretrigger might be detected. A more robust solution would be to define stub's time using the



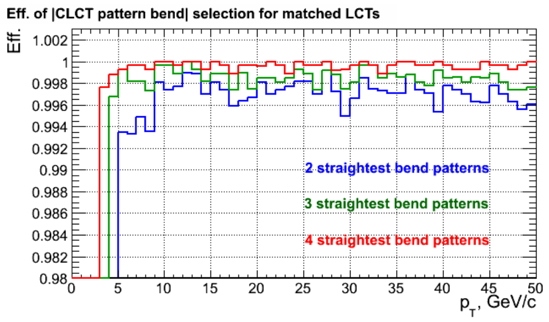


Figure 13: Efficiency vs p_T. No pile up is included in the simulated sample.

times of its strips, where stub's strips are defined as strips that were matched within a CLCT pattern during the trigger. As for a specific procedure, a median time over those strips' times or some sort of a truncated average could be used (see ”clctUseCorrectedBx” parameter in Sec. B.3, analogously for ”alctUseCorrectedBx” parameter in Sec. B.2).

6.6 Software Emulation of CLCT Processing

CLCT processing includes the following four steps:

- Pulse extension;
- Pretrigger;
- Trigger;
- CLCT construction and CLCT dead time.

In contrast to ALCT processing, where all steps are independent and performed one after another for all 16 BXs, during CLCT processing last three steps repeated in one global loop over BXs.

6.6.1 Pulse Extension

Sofware emulation provides information about all half-strip signals in DAQ readout window (16 BXs). A search for these signals is performed in a loop over all half-strips, all layers, and all 16 BXs; found signals are stretched over 6 BXs (see Fig. 14).

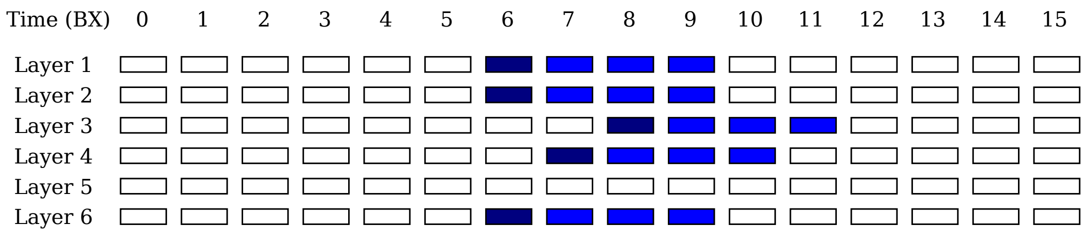


Figure 14: Illustration of ALCT pulse extension for one specific half-strip.

6.6.2 Pretrigger

After all available half-strip signals are stretched, start a global loop over all BXs from BX = 0 and search for CLCT pretriggers in all half-strips. For any given BX and half-strip, count the number of layers with hits within the patterns shown on Fig. 15, and if this number is greater than or equal to three, then we say that a pretrigger occured in this BX and this half-strip. This pretrigger is only accepted if its pattern id ≥ 2. If there were no CLCTs found in some BX = B, proceed to BX = B+1 and continue searching for pretriggers. If there are some CLCTs found in the current BX, proceed to next step.

6.6.3 Trigger

As soon as a BX = B with CLCT pretrigger(s) is found, search for triggers in BX = B+2. For each half-strip, count the number of layers with hits within the same patterns used for pretriggering, and if this number is greater than or equal to four, we say that a trigger occured in this half-strip and BX, and remember:

- pattern id with the highest number of hit layers (if there are two pattern ids with the same number of hit layers, choose smaller pattern id);
- number of hit layers in this pattern id.

Proceed to next step.

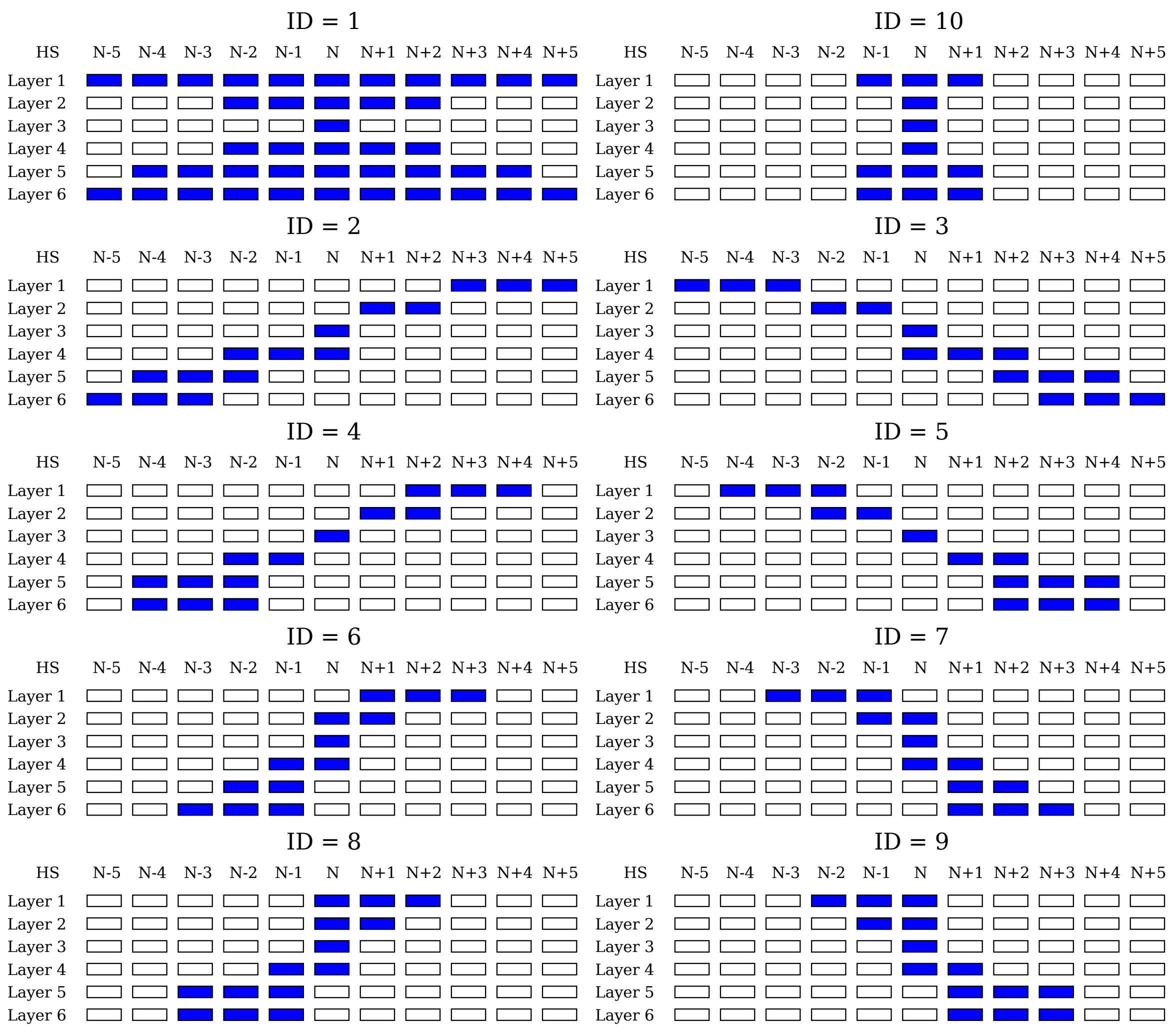


Figure 15: CLCT patterns for pretriggering and triggering.

6.6.4 CLCT Construction and CLCT Dead Time

In a BX with CLCT triggers, find up to two best triggers to be used in CLCT construction.

Find the best trigger:

- Find trigger with the highest number of hit layers;
- If there are two triggers with the same number of hit layers: choose the one with higher pattern id;
- If there are two triggers with the same number of hit layers and the same pattern id: choose the one with smaller half-strip.

Mark zone of 20 half-strips around the best trigger as used and find the second best trigger among not used half-strips.

Construct up to two best CLCTs from found best triggers: encode quality, pattern, bending direction, half-strip, cfeb, BX (defined by pretrigger BX).

After CLCT construction, keep CLCT ”dead”: continue the loop over all BXs until there is a BX with no triggers. When such a BX is found go back to pretriggering step.

6.7 Sofware Emulation of CLCT Level Improvements

6.7.1 Localizing Dead Zone

Current implementation of dead time:

- After constructing up to two CLCTs, continue the loop over all BXs until there is a BX with no triggers
  ...
- In the BX with trigger, construct up to two CLCTs from two best triggers in all half-strips

New implementation of dead time:

- After constructing up to two CLCTs, mark 16 half-strips around half-strips of these CLCTs as busy while number of hit layers in half-strips of these CLCTs ≥ 4
- After trigger in BX = B, keep searching for pretrigger in strating from BX = B+1
  ...
- In the BX with trigger, construct up to two CLCTs from two best triggers in half-strips
  - within 5 half-strips from pretrigger half-strips
  - which are not marked as busy from previous trigger

The following modifications in configuration are related to this improvement:

- useDeadTimeZoning: False to True

6.7.2 Dynamic Dead Zone Width

Fixed dead time zone:

- After constructing up to two CLCTs, mark 16 half-strips around half-strips of these CLCTs as busy while number of hit layers in half-strips of these CLCTs ≥ 4

Dynamic dead time zone:

- After constructing up to two CLCTs, mark K(pid) half-strips around half-strips of these CLCTs as busy while number of hit layers in half-strips of these CLCTs ≥ 4

K(pid) — function of pattern id:

- K(1,2,3) = 22 half-strips
- K(4,5) = 18 half-strips
- K(6,7) = 14 half-strips
- K(8,9) = 10 half-strips
- K(10) = 6 halt-strips

The following modifications in configuration are related to this improvement:

- useDynamicStateMachineZone: False to True

6.7.3 Minimal Pattern ID for Pretriggering

Current CLCT pretrigger:

- Loop over all BXs (starting from BX = 0) and all half-strips:
  - Count number of layers with hits in the following patterns
  - If this number ≥ 3: pretrigger occurs
  - Accept this pretrigger if its pattern id ≥ 2

New CLCT pretrigger:

- Loop over all BXs (starting from BX = 0) and all half-strips:
  - Count number of layers with hits in the following patterns
  - If this number ≥ 3: pretrigger occurs
  - Accept this pretrigger if its pattern id ≥ 4

The following modifications in configuration are related to this improvement:

- clctPidThreshPretrig: 2 to 4

6.7.4 Minimal Separation Between Two Best CLCTs

Current construction of up to two CLCTs:

- Search for the best trigger in this BX
- Mark 20 half-strips around the best trigger as busy
- Find the second best trigger among non-busy half-strips

New construction of up to two CLCTs:

- Search for the best trigger in this BX
- Mark 10 half-strips around the best trigger as busy
- Find the second best trigger among non-busy half-strips

The following modifications in configuration are related to this improvement:

- clctMinSeparation: 10 to 5 cathode strips

6.8 Results of Improvements of the CLCT Processing

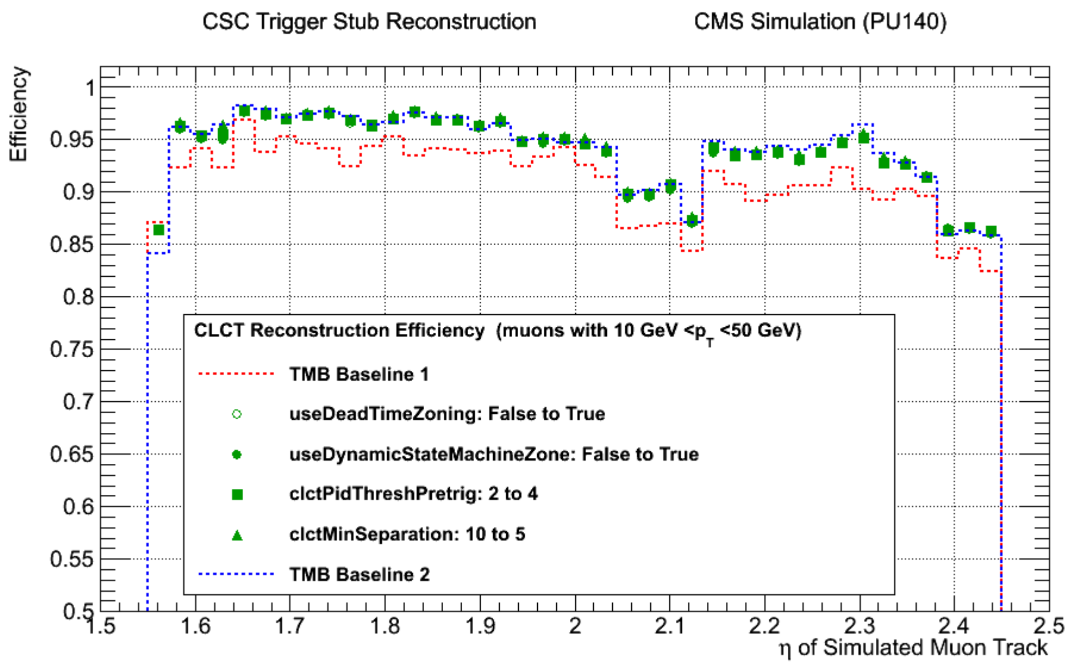
Improvements on the level of CLCT processor are related to the following configuration parameters (see Sec. B.3):

- useDeadTimeZoning: False to True;
- useDynamicStateMachineZone: False to True;
- clctPidThreshPretrig: 2 to 4;
- clctMinSeparation: 10 to 5 cathode strips.

Fig. 16 shows reconstruction efficiency of a good CLCT in ME1/1 station versus pseudorapidity of the simulated muon for different L1 configurations. The good CLCT is defined as CLCT:

- read out in the window of 3BX around the central BX (BX6);
- reconstructed within 2 cathode strips from the key strip.
- has hits at least on four layers

The major improvement in CLCT reconstruction efficiency comes from localization of the dead-time zone.



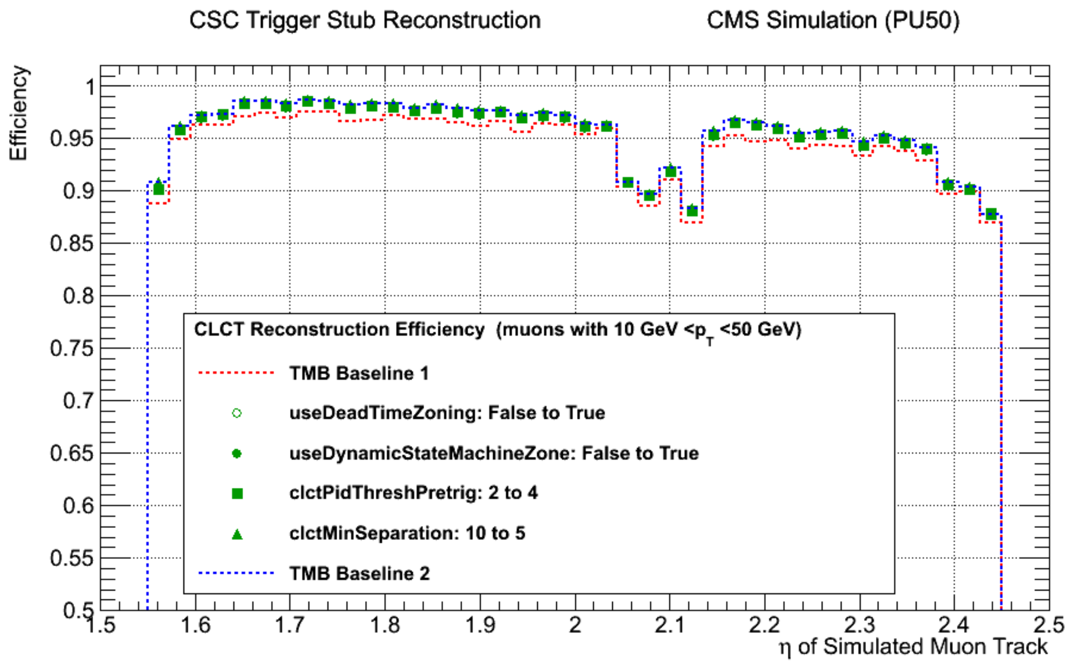


Figure 16: CLCT reconstruction efficiency in ME1/1 station for PU140 (top) and PU50 (bottom). Muons with transverse momentum 10 GeV$< p_T < 50$ GeV are used in the analysis.

7 Cathode-Anode Correlation

7.1 LCT Algorithm

The Trigger Motherboard (TMB) portion of the CLCT/TMB card receives up to two anode stubs from the ALCT board and two cathode stubs from the CLCT portion of the CLCT/ TMB card. The functions of the TMB circuitry are:

- Bunch crossing alignment of the anode and cathode tags.
- Correlation of the Anode and Cathode LCT words and construction of two combined LCTs.
- Transmission of LCT data to the Muon Port Card (MPC) for triggering, and transmission of DAQ data to the DAQ Motherboard (DAQMB).

Incoming anode and cathode LCTs are not aligned in time. Anode LCTs are created faster than cathode LCTs because of the slow development of the cathode preamp signal, and because processing inside the ALCT card is faster than processing inside the CLCT logic. The TMB contains input pipeline logic in order to delay anode LCTs for a programmable number of bunch crossings up to 10.

The anode and cathode LCTs are matched according to the more precise ALCT bunch crossing number (BXN). The Cathode LCT BXN can differ by at most ± 1 bunch crossing. For each of the selected muons the TMB outputs a 2-bit bunch crossing match word as shown in table below. These may be used by later boards in the trigger chain if additional quality information is needed. They also allow the analysis of the bunch crossing matching in the TMB, since a large number of bad matches could be an indication of a timing alignment problem.

The ideal case for a high-momentum muon is one anode and one cathode LCT pattern. However, other cases may occur, which are distinguished by a 2-bit STA (Status type A) code:

- The TMB may receive one or two anode LCTs and zero cathode LCT patterns. This happens, for example, for very low-momentum muons. Although the non-zero data is forwarded to the MPC, this case is flagged by STA=1, as is the similar case of one or two cathode LCT and zero anode LCT patterns.
- If the TMB receives two anode LCTs and one cathode LCT, the TMB outputs two LCTs, by copying the Cathode LCT bits into both muons. These, and the similar case of two cathode LCTs and one anode LCT, are flagged by STA=2.
- If there are two anode LCTs and two cathode LCTs in one chamber, they are matched according to their pattern numbers: the largest ALCT and CLCT pattern numbers are paired, and the second largest ALCT and CLCT pattern numbers are paired. These, and the ideal case of a single match, are flagged by STA=3.

TMBs maintain a local Bunch Crossing Number (BXN) using signals from the Clock and Control Board. The internal BXN is compared to the BXN received from the ALCT module, and the Sync Error bit is set if a mismatch is detected.

The TMB sends up to two anode LCT and two cathode LCT patterns for one CSC chamber to the MPC every 25 ns.

7.2 Reduction of the Matching Time Window

The current TMB uses a rather wide time matching window (7BX) that is centered on a CLCT's BX, and is used to look for an ALCT match within it. A wide matching window is not good in high pileup, as propability of incorrect matching with background 2D stubs is higher, resulting in inefficiency.

For the SLHC, we can probably assume that the system is well timed and that we can use the narrowest reasonable matching window of 3BX wide (see ”matchTrigWindowSize” parameter in Sec. B.4).

7.3 Modification of the Stub Timing Logic in Matching

In the old TMB algorithm, the stub timing logic works during the 2D stubs matching as follows:

- CLCT-centric approach: CLCTs and their BX are taken as reference points, while ALCTs are waiting in a queue
- for a BX with CLCTs we look for a first BX in the matching window that has ALCTs
- after matching is dome in this BX, the ALCTs from there are taken off the queue, and cannot be matched with any later CLCT (see ”tmbDropUsedClcts” and ”matchEarliestClctME11Only” parameters in Sec. B.4)

The main issues with this approach at high luminosity is that when there is an ALCT from a good signal muon, early CLCTs from background might steal it and form wrong match, and this correct ALCT then would not be available anymore for matching with a later correct CLCT.

Proposal to improve the situation:

- ALCT-centric approach: ALCTs and their BXs are taken as reference points, while reconstructed CLCTs are waiting in a matching window-wide queue (see ”clctToAlct” parameter in Sec. B.4)
    - ALCT’s BX and the middle BX in the matching window-wide queue are expected to be synchronized
- for an ALCT’s BX we look for CLCTs within the queue in the order of arrival try to find maximum 2 LCT matches as follows (see ”tmbCrossBxAlgorithm” parameter in Sec. B.4):
    - first look for CLCTs in the same BX 2) if we didn’t get 2 LCT matches yet, look for CLCTs in BX-1
    - if we didn’t get 2 LCT matches yet, look for CLCTs in BX+1
    - etc... depending on how wide the matching window is
- can optionally either remove CLCTs from the queue after there was an LCT match, or can keep them for reuse possibilities to be matched with later ALCTs (see ”tmbDropUsedClcts” parameter in Sec. B.4)
- NOTE: all this is supposed to be done separately in ME1/1a and in ME1/1b

7.4 Selection of the Two Best LCTs per ME1/1

With the backplane limitations, we can read out only up to two trigger stubs per BX from the whole ME1/1.

Since ME1/1a and ME1/1b now can each have up to two stubs, we need an extra step of selecting the best two ME1/1 stubs out of possible 4. If ME1/1a + ME1/1b has more then two LCTs:

- The simplest solution:
    - drop the highest eta ones until we have just two.

- Possible improvement:
  - rank stubs by special quality value which is the same as stub quality for ME1/1b and is stub quality-1 for ME1/1a stubs
  - if special quality is the same, rank by eta

7.5 Software Emulation of CLCT and ALCT Matching

Every BX OTMB receives up to 2 CLCTs and up to two ALCTs from CLCT and ALCT processors, Fig. 17 shows an example which will be used throughout the subsection.

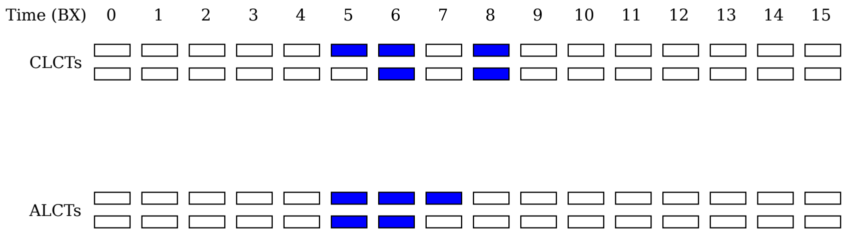


Figure 17: Example of CLCTs and ALCTs received by OTMB.

There are two approaches to CLCT and ALCT correlation: CLCT-centric and ALCT-centric. We will introduce the former below and discuss the latter later among OTMB level improvements.

CLCT-centric CLCT and ALCT correlation (see Fig. 18):

- Loop over CLCT BXs from BX = 0 to BX = 15
- For CLCT BX = B with at least one valid CLCT:
  - Loop over ALCT BXs from BX = B-3 to BX = B+3
  - Find the first ALCT BX in the matching window with at least one valid ALCT and not marked as used before
    - Correlate CLCTs and ALCTs in matching ALCT and CLCT BXs
    - Mark ALCT BX as used
    - Proceed to next CLCT BX

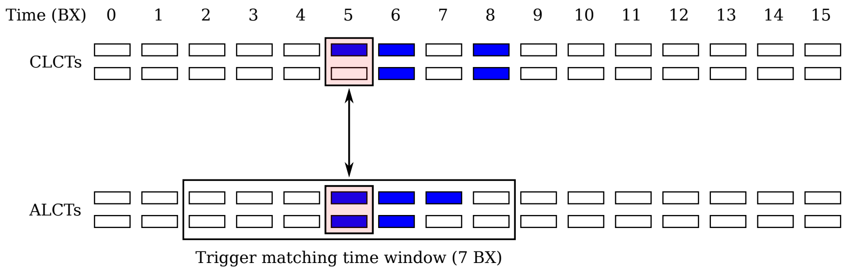


Figure 18: CLCT-centric CLCT and ALCT correlation.

The results of such a correlation are shown on Fig. 19.

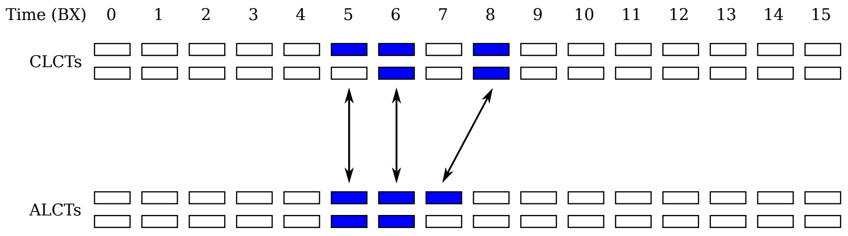


Figure 19: Result of CLCT-centric CLCT and ALCT correlation.

By default, LCTs are constructed only from valid CLCTs and ALCTs, but we may optionally allow construction of ALCT-less or CLCT-less LCTs.

If there are no ALCT BXs with at least one valid ALCT in the watching window and ALCT-less LCTs are allowed, construct LCTs from valid CLCTs in the current CLCT BX (see example on Fig. 20 with results on Fig. 21).

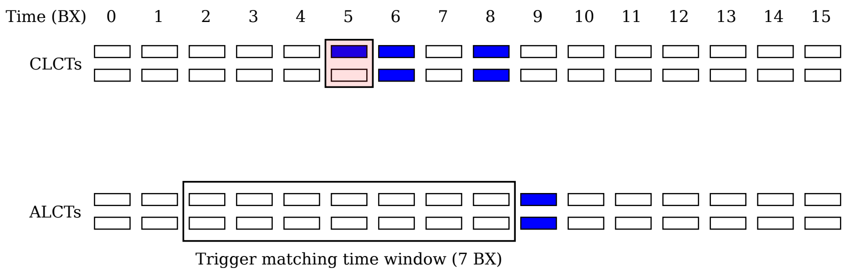

Time (BX) 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
CLCTs
ALCTs
Trigger matching time window (7 BX)

Figure 20: Example of construction of ALCT-less LCTs.

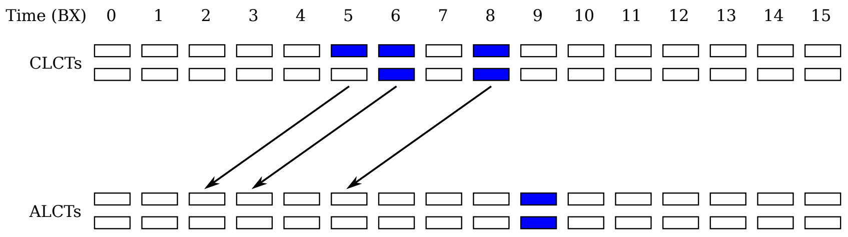


Figure 21: Results of construction of ALCT-less LCTs.

If there are no valid CLCTs in the current CLCT BX and CLCT-less LCTs are allowed (see example on Fig. 22 with results on Fig. 23):

- Find first ALCT BX in the matching window with at least one valid ALCT and not marked as used before;
- Construct LCTs from valid ALCTs in that ALCT BX.

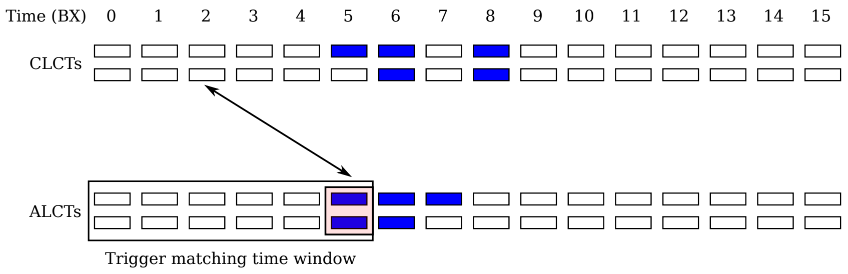

Time (BX) 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
CLCTs
ALCTs
Trigger matching time window

Figure 22: Example of construction of CLCT-less LCTs.

When we use default behavior, where LCTs are constructed only from valid CLCTs and ALCTs, the ideal case is when in matching BXs number of valid CLCTs is equal to number of valid ALCTs (see top two examples on Fig. 24).

But when, for example, there is only one valid CLCT and two valid ALCTs, we make the second valid CLCT from the first, analogously, when there are two valid CLCTs and only one valid ALCT (see bottom two examples on Fig. 24).

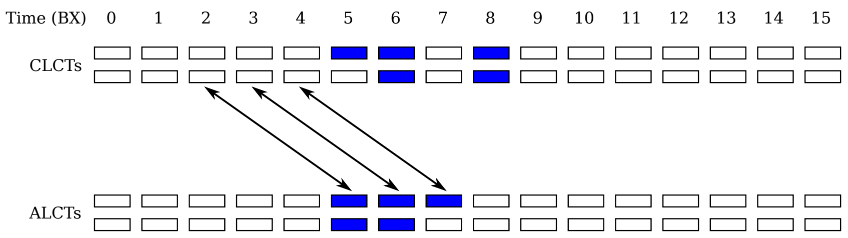


Figure 23: Results of construction of CLCT-less LCTs.

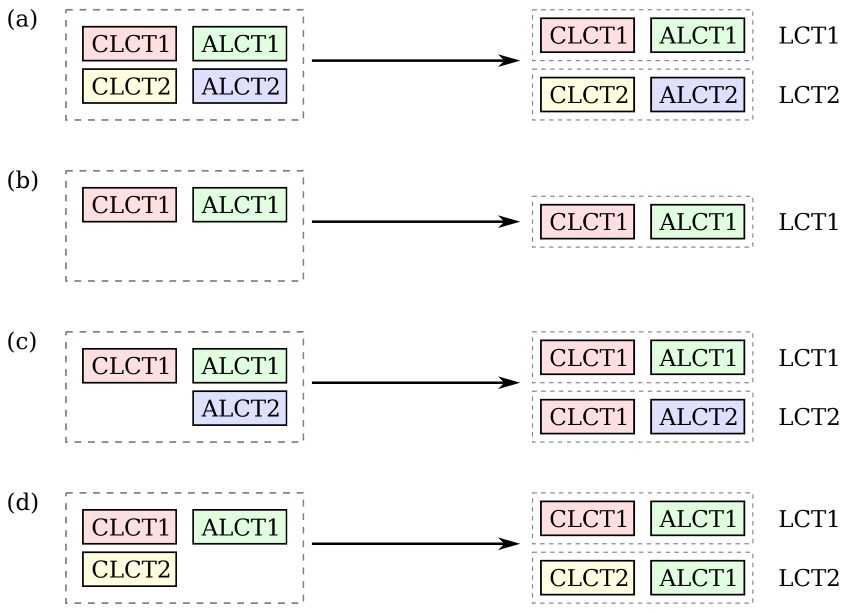


Figure 24: Construction of LCTs.

7.6 Sofware Emulation of TMB Level Improvements

7.6.1 Decreased Trigger Matching Window

Decrease size of trigger matching time window from 7 BXs to 3 BXs (see Fig. 25).

The following modifications in configuration are related to this improvement:

- matchTrigWindowSize: 7BX to 3BX

7.6.2 ALCT-centric ALCT and CLCT Correlation

Switch from CLCT-centric matching to ALCT-centric matching (see Fig. 26).

CLCT-centric matching

- Loop over CLCT BXs from BX = 0 to BX = 15
- For CLCT BX = B with at least one valid CLCT:
    - Loop over ALCT BXs from BX = B-3 to BX = B+3

ALCT-centric matching

- Loop over ALCT BXs from BX = 0 to BX = 15
- For ALCT BX = B with at least one valid ALCT:

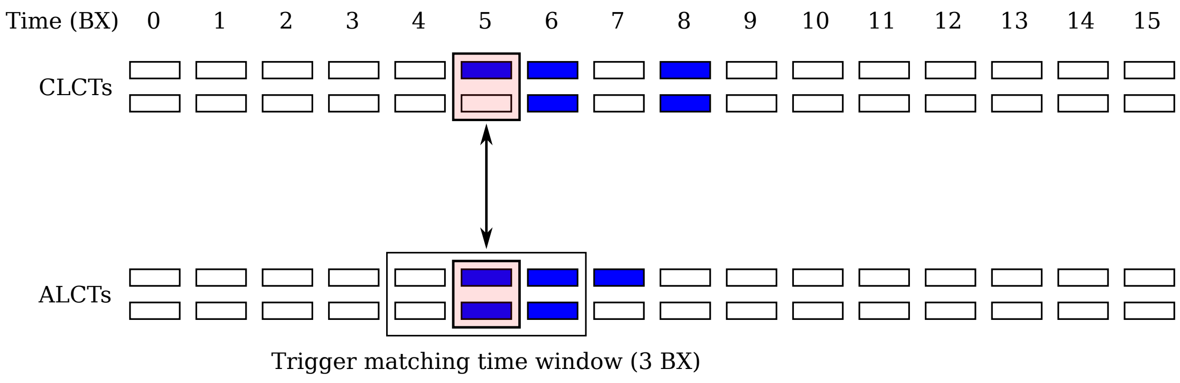


Figure 25: Decreased trigger matching window.

- Loop over CLCT BXs from BX = B-3 to BX = B+3

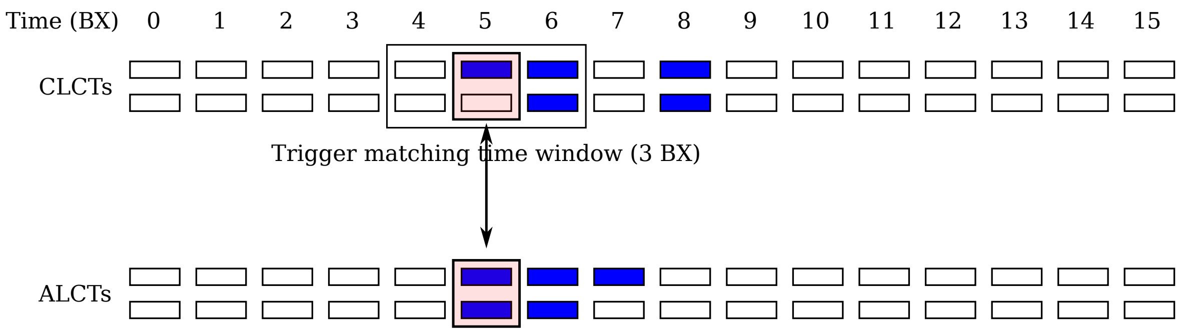


Figure 26: ALCT-centric ALCT anc CLCT correlation.

The following modifications in configuration are related to this improvement:

- clctToAlct: True to False

7.6.3 Reusage of Used ALCTs and CLCTs

Allow reusage of ALCTs and CLCTs already used during ALCT and CLCT correlation (see Fig. 27).

Current behavior:

- Drop used CLCTs: do not use them with ALCTs in other ALCT BXs;
- Proceed to next ALCT BX after matching ALCTs with earliest CLCT BX with at least one valid CLCT.

New behavior:

- Do not drop used CLCTs: reuse them with ALCTs in other ALCT BXs;
- Match ALCTs to CLCTs in all CLCT BXs within matching window.

The following modifications in configuration are related to this improvement:

- tmbDropUsedClcts: True to False
- matchEarliestClctME11Only: True to False

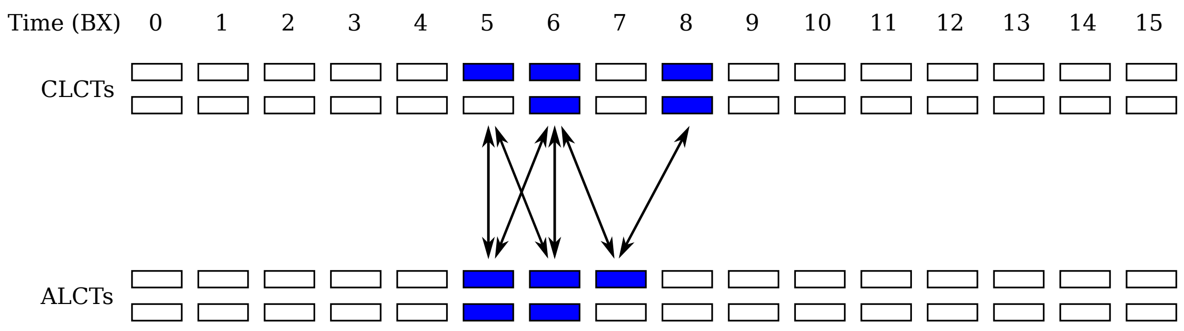


Figure 27: Reusage of already used ALCTs and CLCTs.

7.6.4 Cross BX Algorithm

In the situation shown on Fig. 27, in some given BX = B we can end up having with up to 6 LCTs (made from ALCTs in BX = B and CLCTs in BX = B-1, B, B+2. How do we choose two LCTs to be reported for BX = B?

Current behavior: ”cross bx” algorithm is turned off

- Choose two best LCTs with the highest quality

New behavior: ”cross bx algorithm”

- Take LCTs with ALCT BX = B and CLCT BX = B
- If we still don't have two LCTs, take best ones with ALCT BX = B-1
- If we still don't have two LCTs, take best ones with ALCT BX = B+1

The following modifications in configuration are related to this improvement:

- tmbCrossBxAlgorithm: 0 to 1

7.6.5 Corrected ALCT and CLCT Timing

Use more robust procedure for assignment of ALCT BX and CLCT BX.

Current behavior:

- Use ALCT and CLCT pretrigger BXs to assign ALCT BX and CLCT BX.

New behavior:

- For each hit in ALCT and CLCT trigger patterns, determine ”first BX”: BX of original hit before hit stretching over 6 BXs;
- For ALCTs, consider hits in key WG = N and two neighbouring WGs: WG = N-1 and WG = N+1;
- For CLCTs, consider all hits in the pattern;
- Store ”first BXs” of these hits in two sorted sets (one for ALCT times and another for CLCT times);
- Use median elements in these sets to assign ALCT BX and CLCT BX.

The following modifications in configuration are related to this improvement:

- alctUseCorrectedBx: False to True
- clctUseCorrectedBx: False to True

7.6.6 Reading out more LCTs

[I’m not sure I completely understand the meaning of this improvement]

Current behavior

- In digi$\rightarrow$raw step, LCTs have to be packed into the TMB header, and currently there is room just for two
- Take LCTs only from earliest BX in L1A readout with at least one LCT

New behavior

- Take LCTs from the whole L1A readout (from BX = 5 to BX = 11)

The following modifications in configuration are related to this improvement:

- tmbReadoutEarliest2: True to False

7.7 Results of Improvements of the ALCT and CLCT Matching

Improvements on the level of TMB are related to the following configuration parameters (see Sec. B.4):

- matchTrigWindowSize: 7BX to 3BX;
- tmbReadoutEarliest2: True to False;
- alctUseCorrectedBx: False to True
- clctUseCorrectedBx: False to True;
- clctToAlct: True to False;
- tmbDropUsedClcts and matchEarliestClctME11Only: True to False;
- tmbCrossBxAlgorithm: 0 to 1.

Fig. 28 shows reconstruction efficiency of a good LCT in ME1/1 station versus pseudorapidity of the simulated muon for different L1 configurations. The good LCT is defined as LCT consisted of a good ALCT and a good CLCT.

The major improvement in LCT reconstruction efficiency comes from:

- changing the size of a window where ALCTs and CLCTs are read out for further correlation between each other;
- stopping to read out only the first two CLCTs;
- stopping to drop used CLCTs;
- stopping to match only to the earliest CLCT.

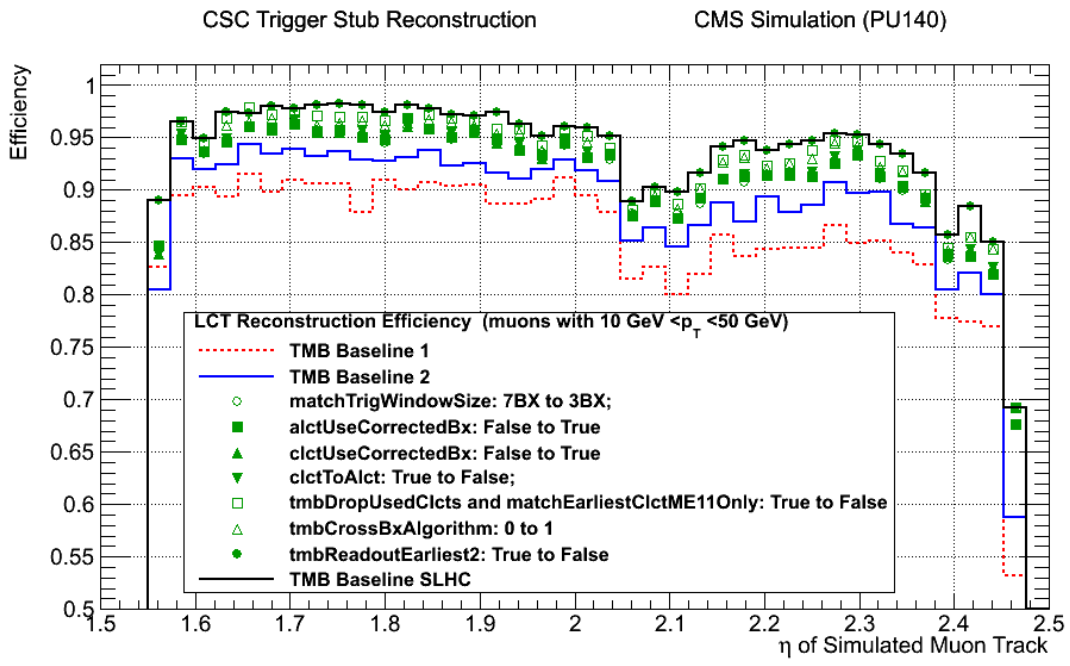
The effects on the efficiency improvements are more evident if we divide the whole ME1/1 chamber into three different sections by eta:

- from $\eta = 1.65$ to $\eta = 2.0$ to cover the ME1/1b chamber alone;
- from $\eta = 2.0$ to $\eta = 2.2$ to cover the gap between ME1/1a and ME1/1b;
- from $\eta = 2.2$ to $\eta = 2.45$ to cover the ME1/1a chamber alone;

The effects on the efficiency for each one of the TMB improvment in the main η partitions defined above are summarized in Tab. 1 for PU140 and in Tab. 2 for PU50. A more detailed study comparing each improvment over the whole η range for the ME1/1 chamber can be found in Appendix.

Improvment	ME1/1b		Gap Region		ME1/1a	
	ϵ (%)	$\delta\epsilon$	ϵ (%)	$\delta\epsilon$	ϵ (%)	$\delta\epsilon$
TMB Baseline 1	89.92	0.13	82.62	0.16	82.54	0.16
TMB Baseline 2	92.63	0.11	86.53	0.15	86.38	0.15
matchTrigWindowsSize: 7BX to 3 BX	94.38	0.09	89.27	0.13	88.97	0.13
alctUseCorrectedBX: False to True	95.00	0.09	89.36	0.13	89.35	0.13
clctUseCorrectedBX: False to True	95.01	0.09	89.37	0.13	89.36	0.13
clctToAlct: True to False	95.11	0.09	89.40	0.13	80.36	0.13
tmbDropUsedClcts and matchEarliestClctME11Only: False to True	96.27	0.08	90.54	0.12	90.59	0.12
tmbCrossBXAlgorithm: 0 to 1	95.69	0.09	90.24	0.13	90.33	0.13
tmbReadoutEarliest2: True to False	97.00	0.07	91.65	0.12	91.97	0.12
TMB Baseline SLHC	97.00	0.07	91.65	0.12	91.97	0.12

Table 1: LCT reconstruction efficiencies in ME1/1 station after individual improvements in algorithm for PU140. Muons with transverse momentum 10 GeV$< p_T <$ 50 GeV are used in the analysis.



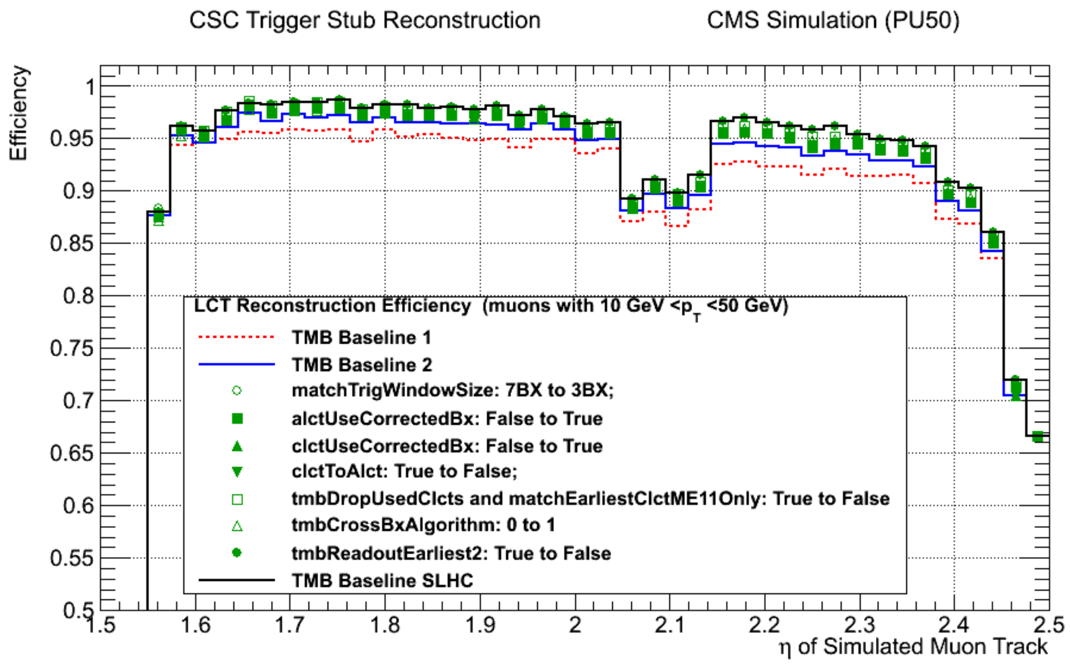


Figure 28: LCT reconstruction efficiency in ME1/1 station for PU140 (top) and PU50 (bottom). Muons with transverse momentum 10 GeV$< p_T < 50$ GeV are used in the analysis.

Improvment	ME1/1b		Gap Region		ME1/1a	
	ϵ (%)	$\delta\epsilon$	ϵ (%)	$\delta\epsilon$	ϵ (%)	$\delta\epsilon$
TMB Baseline 1	94.95	0.09	89.18	0.13	90.38	0.13
TMB Baseline 2	96.32	0.08	90.74	0.12	91.78	0.12
matchTrigWindowsSize: 7BX to 3 BX	97.13	0.07	91.72	0.12	92.91	0.11
alctUseCorrectedBX: False to True	97.13	0.07	91.80	0.12	93.00	0.11
clctUseCorrectedBX: False to True	97.03	0.07	91.80	0.12	93.06	0.11
clctToAlct: True to False	97.03	0.07	91.80	0.12	93.06	0.11
tmbDropUsedClcts and matchEarliest-ClctME11Only: False to True	97.52	0.07	92.23	0.11	93.61	0.10
tmbCrossBXAlgorithm: 0 to 1	97.03	0.07	91.86	0.12	93.39	0.11
tmbReadoutEarliest2: True to False	97.69	0.06	92.62	0.11	94.07	0.10
TMB Baseline SLHC	97.69	0.06	92.62	0.11	94.07	0.10

Table 2: LCT reconstruction efficiencies in ME1/1 station after individual improvements in algorithm for PU50. Muons with transverse momentum 10 GeV$< p_T < 50$ GeV are used in the analysis.

References

[1] CMS Collaboration, “The CMS experiment at the CERN LHC”, *JINST* **3** (2008) S08004, doi:10.1088/1748-0221/3/08/S08004.

[2] CMS Collaboration, “The performance of the CMS muon detector in proton-proton collisions at $\sqrt{s} = 7$ TeV at the LHC”, *JINST* **8** (2013) P11002, doi:10.1088/1748-0221/8/11/P11002, arXiv:1306.6905.

[3] S. Dildick, T. Huang et al., “CSC Phase-II Trigger Upgrade Studies for the Forward Endcap Muon Chambers with GEMs and RPCs”, *CMS Detector Note* **2014/018** (2014).

[4] CMS Collaboration, “CMS, the Compact Muon Solenoid. Muon technical design report”,.

[5] D. Acosta et al., “Large CMS cathode strip chambers: Design and performance”, *Nucl.Instrum.Meth.* **A453** (2000) 182–187, doi:10.1016/S0168-9002(00)00627-6.

[6] Y. Ershov et al., “Cathode strip chamber for CMS ME1/1 endcap muon station”, *Phys. Part. Nucl. Lett.* **3** (2006) 183–187, doi:10.1134/S154747710603006X.

A Mechanical Layout and Signal Readout of the ME1/1 Chambers

The ME1/1 is the first muon station of CMS Endcaps [4] in very forward region $2.43 < |\eta| < 1.55$. It is composed of 36 Cathode Strip Chambers (CSC) [5, 6], in each Endcap (see Fig. 1). The ME1/1 CSC should provide very good spatial resolution of 75 μm per station in order to achieve the required momentum resolution in the endcap muon system. This spatial resolution should be delivered in the presence of a strong axial magnetic field in excess of 3 Tesla. The chambers must provide efficient pattern recognition and matching with the inner tracker. The chambers should be very fast in order to identify the bunch-crossing. Their recovery time should be also very fast because the chambers will operate in the presence of the highest particle background rate in the CMS Muon System, up to 1 kHz/cm^2, which corresponds to a rate of 100 kHz per cathode readout channel.

The design parameters of the ME1/1 CSC are optimized to meet the specified requirements. The ME1/1 CSC layout is shown in Fig. 29 and the basic chamber parameters are presented in Tab. 3. It is a unit of 6 layers of identical proportional chambers of a trapezoidal shape with a cathode strip readout. Each layer is formed by 2 cathode electrodes: strips and continuous plane, having a gap of 7 mm. The radial strip structure of one ME1/1 CSC covers an angle of $\phi = \pm 5.42^o$ to provide an overlap with the neighboring chambers. The anode wires are placed in the middle of the gap. To compensate for the effect of the CSC spatial resolution deterioration due to the presence of the strong axial magnetic field at a nominal value of 3.8 Tesla (Lorentz effect) the anode wires are positioned at an inclination angle of 29^o with respect to and perpendicular to the central strip axis. The groups of 11 anode wires are connected to one readout channel providing radial coordinate measurements, while the interpolation of charges induced on strips gives the precise measurement of ϕ coordinate.

The background rate at the bottom of the ME1/1 CSC ($|\eta| \sim 2.4$) is expected to be significantly higher than that at the top ($|\eta| \sim 1.6$). To decrease the counting rate per strip readout channel, the cathode planes of the ME1/1 CSC are mechanically separated into two parts at $|\eta| = 2.09$. The bottom part ME1/1a ($2.09 < |\eta| < 2.43$) has 48 strips and the top part ME1/1b ($1.55 < |\eta| < 2.09$) has 64 strips.

Electronics and read out configuration of the signals from cathode strips of the ME1/1 CSC were significantly upgraded during Long Shutdown 1 (LS1) of the LHC in 2013-2015. Signals from the cathode strips are amplified and shaped by seven 96-channel Digital Cathode Front End Boards (DCFEB). Each DCFEB reads out a ”tower” of 16 strips from all 6 layers. There are 4 DCFEBs attached to the wide part of a chamber (reading out 64 strips per layer in top region) and 3 DCFEBs attached to the narrow part (reading out 48 strips per layer in bottom region). During upgrade new DCFEBs replaced old CFEBs (not digital). Also before upgrade all 48 cathode strips in bottom part of a chamber were triple-ganged to 16 channels in the electronics for both the trigger and the readout, making hit recognition ambiguous. Only one CFEB was used to read out entire bottom part of the ME1/1 CSC. After upgrade the 48 strips in bottom part of the chambers were unganged and connected to three DCFEBs, removing ambiguity in hit recognition.

Signals from the anode wires are amplified and discriminated by the Anode Front End Boards (AFEB). Each AFEB reads out 16 wire groups from 2 layers of the CSC. The discriminator bits are transmitted to the Anode Local Charge Track (ALCT) board where signals are stored as hits. An ALCT board reads out all 18 AFEBs.

Table 3: Parameters of ME1/1 CSC.

ME1/1 CSC	Inner radius	1060 mm ($\|\eta\| = 2.43$)
	Strips cut radius	1500 mm ($\|\eta\| = 2.09$)
	Outer radius	2565 mm ($\|\eta\| = 1.55$)
	Strip channels per chamber	672
	Anode channels per chamber	288
	Layers per chamber	6
Layer	Strip channels per layer	112
	– top part	64
	– bottom part	48
	Anode channels per layer	48
	Number of wires:	
	– total per anode channel (1)	37
	– total per anode channel (2 – 47)	11
	– total per anode channel (48)	44
	– total per layer	587

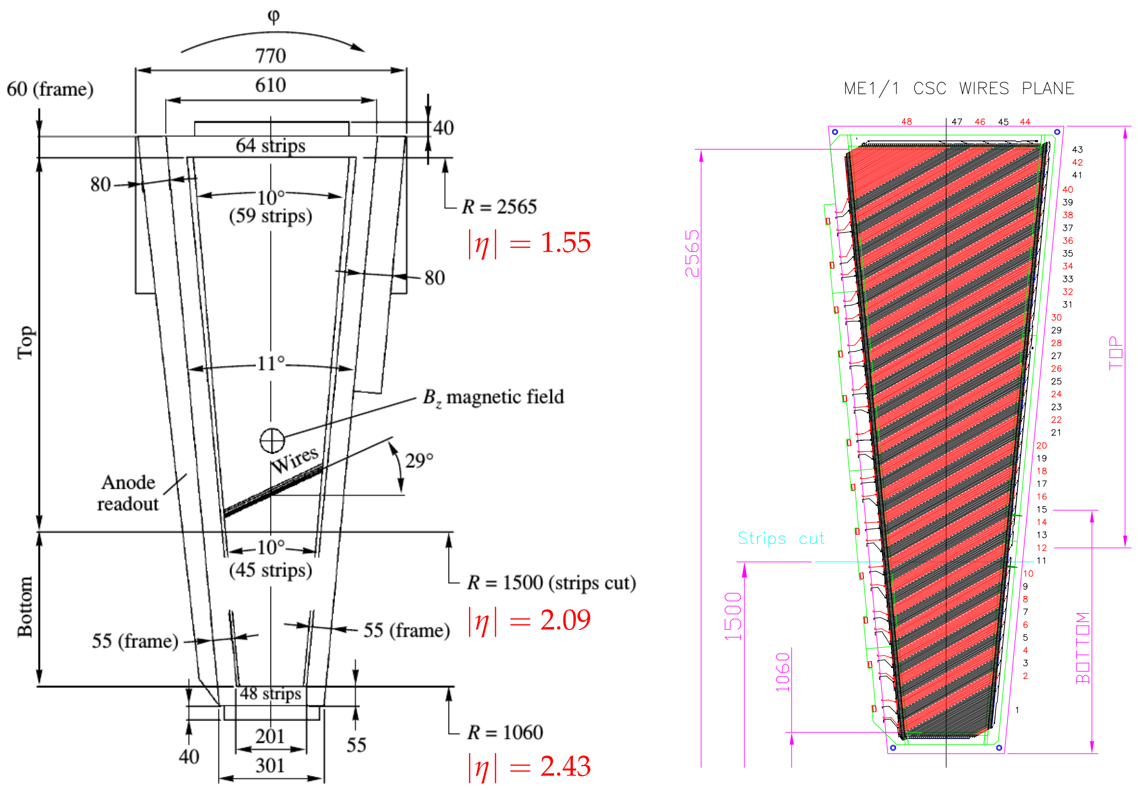


Figure 29: Overall mechanical layout of ME1/1 CSC (left) [6] and layout of 48 anode wire groups positioned at an inclination angle of 29^o (right). All dimensions are in millimeters.

B Configuration of the Local Trigger Algorithm in the CMS Software

For simulated studies the local trigger algorithm is emulated in the CMS software[1]:

- branch: `CMSSW_6_2_X_SLHC`
- package: `L1Trigger/CSCTriggerPrimitives`.

The software emulator is divided in three logical levels:

1. ALCT processing[2],
2. CLCT processing[3],
3. ALCT and CLCT matching[4].

Each part of the emulator has its own set of parameters listed in the following Sections.

B.1 Algorithm Selection and Common Configuration

There are three main local trigger algorithms emulated in the CMS software:

1. MTCC — algorithm developed for Magnet Test and Cosmic Challenge studies,
2. TMB07 — default (before upgrade) algorithm for LHC Run 1 studies,
3. SLHC — new algorithm for studies of operation under high pile-up running conditions.

Selection of the algorithm in software emulator is configured using the following parameters:

	Algorithm		
	MTCC	TMB07	SLHC
`isMTCC`	True	False	
`isTMB07`	False	True	
`isSLHC`	False	False	True

The SLHC algorithm is strongly depends on TMB07 algorithm, and, thus, requires both `isTMB07` and `isSLHC` to be set to True. The MTCC algorithm is not subject of the studies performed in this paper, therefore it is omitted from the following descriptions.

Configuration parameters common for all levels of the algorithm processing are the following:

	TMB07	SLHC
`smartME1aME1b`	False	True
`gangedME1a`	True	False
`disableME1a`	False	
`disableME42`	False	

[1]See actual code in `https://github.com/cms-sw/cmssw/tree/CMSSW_6_2_X_SLHC/L1Trigger/CSCTriggerPrimitives`

[2]ALCT processing emulator defined in `https://github.com/cms-sw/cmssw/blob/CMSSW_6_2_X_SLHC/L1Trigger/CSCTriggerPrimitives/src/CSCAnodeLCTProcessor.cc`

[3]CLCT processing emulator defined in `https://github.com/cms-sw/cmssw/blob/CMSSW_6_2_X_SLHC/L1Trigger/CSCTriggerPrimitives/src/CSCCathodeLCTProcessor.cc`

[4]ALCT and CLCT matching defined in `https://github.com/cms-sw/cmssw/blob/CMSSW_6_2_X_SLHC/L1Trigger/CSCTriggerPrimitives/src/CSCMotherboard.cc`

- smartME1aME1b — allows one ALCT finder and two CLCT finders per ME1/1 chamber with additional logic for ALCT and CLCT matching with ME1/1a unganged. It allows to set the following CLCT processing configuration parameters in SLHC algorithm:
    - useDeadTimeZoning
    - clctStateMachineZone
    - useDynamicStateMachineZone
    - clctPretriggerTriggerZone
    - use_corrected_bx
- gangedME1a — has effect only if smartME1aME1b=True and defines number of readout channels from ME1/1a:
    - all 48 cathode strips triple ganged into 16 strip groups, read out by 1 CFEB (default in TMB07)
    - all 48 cathode strips are separately read out by 3 DCFEBs (default in SLHC)
- disableME1a and disableME42 — allow to disable finding stubs in ME1/1a or ME4/2 chambers.

B.2 Configuration of the ALCT Processing Emulator

There are 18 configuration parameters for the ALCT processing emulator with 5 of them different between TMB07 and SLHC. List of all parameters is the following:

	TMB07	SLHC
alctFifoTbins	16	
alctFifoPretrig	10	
alctDriftDelay	2	
alctNplanesHitPretrig	3	
alctNplanesHitPattern	4	
alctNplanesHitAccelPretrig	3	
alctNplanesHitAccelPattern	4	
alctTrigMode	2	
alctAccelMode	0	
alctL1aWindowWidth	7	
verbosity	0	
alctEarlyTbins	4	
alctNarrowMaskForR1	False	True
alctHitPersist	6	
alctGhostCancellationBxDepth	4	1
alctGhostCancellationSideQuality	False	True
alctPretrigDeadtime	4	0
alctUseCorrectedBx	—	True

- alctFifoTbins — width of ALCT FIFO window in BX clocks[5]. Total number of time bins in ALCT DAQ readout, which thus determines the maximum length of readout time interval.

[5]Note, that 1 BX clock = 25 ns

- alctFifoPretrig — width of ALCT FIFO window in BX clocks where pretrigger can happen. Anode raw hits in DAQ readout start alctFifoPretrig−fpga_latency[6] BX clocks before L1A.The alctFifoPretrig parameter should be shorter than alctFifoTbins by the sum of alctDriftDelay and alctPretrigDeadtime.
- alctDriftDelay — drift delay in BX clocks after pre-trigger. If pretrigger happen in BX = B, then trigger is checked in BX = B + alctDriftDelay.
- alctNplanesHitPretrig — minimum number of layers with hits for pretrigger in “collision” pattern for a particular key WG.
- alctNplanesHitPattern — minimum number of layers with hits for trigger in “collision” pattern for a particular key WG.
- alctNplanesHitAccelPretrig — minimum number of layers with hits for pre-trigger in “accelerator” pattern for a particular key WG.
- alctNplanesHitAccelPattern — minimum number of layers with hits for trigger in “accelerator” pattern for a particular key WG.
- alctTrigMode — enables/disables either “collision” or “accelerator” stubs:
  - $= 0$ — enables both “collision” or “accelerator” stubs,
  - $= 1$ — disables “collision” stub,
  - $= 2$ — disables “accelerator” stub (default in TMB07 and SLHC),
  - $= 3$ — disables “collision” stub if there is an “accelerator” stub found in the same wire group at the same time.
- alctAccelMode — sets preference to either “accelerator” or “collision” stubs, current mode ignores “accelerator” patterns completely:
  - $= 0$ — ignore “accelerator” stubs (default in TMB07 and SLHC),
  - $= 1$ — prefer “collision” stubs by adding promotion bit,
  - $= 2$ — prefer “accelerator” stubs by adding promotion bit,
  - $= 3$ — ignore “collision” stubs.
- alctL1aWindowWidth — width of L1A window in BX clocks, the window spans from alctEarlyTbins to alctEarlyTbins+alctL1aWindowWidth.
- verbosity — turns on debugging messages during ALCT processing.
- alctNarrowMaskForR1 — enables/disables narrow ALCT pattern for chambers in Ring 1 (see Fig. 7).
- alctHitPersist — sets persistence of anode wire hits in BX clocks to signify the duration of a signal.
- alctGhostCancellationBxDepth — configures up to how many BX clocks in the past ghost cancellation in neighboring WGs (N+1 and N-1) for a stub in current WG=N may happen.
- alctGhostCancellationSideQuality — enebles/disables whether to compare the quality of stubs in neighboring WGs (N-1 and N+1) in the past to the quality of a stub in current WG=N when doing ghost cancellation .
- alctPretrigDeadtime — defines how soon after pretrigger and alctDriftDelay can next pretrigger happen. This is extra in addition to drift delay, so total deadtime = alctDriftDelay + alctPretrigDeadtime.

[6] fpga_latency = 6, as determined in https://github.com/cms-sw/cmssw/blob/CMSSW_6_2_X_SLHC/L1Trigger/CSCTriggerPrimitives/src/CSCAnodeLCTProcessor.cc#L229

- `alctUseCorrectedBx` — defines whether to calculate “corrected” ALCT stub time (currently it is median time of hits in a pattern) instead of pretrigger one. Implemented only in SLHC algorithm.

B.3 Configuration of the CLCT Processing Emulator

Configuration parameters for the CLCT level of the algorithm processing are the following:

	TMB07	SLHC
`clctFifoTbins`	12	
`clctFifoPretrig`	7	
`clctHitPersist`	4	
`clctDriftDelay`	2	
`clctNplanesHitPretrig`	3	
`clctNplanesHitPattern`	4	
`clctPidThreshPretrig`	2	4
`clctMinSeparation`	10	5
`verbosity`	0	
`useDeadTimeZoning`	—	True
`clctStateMachineZone`	—	8
`useDynamicStateMachineZone`	—	True
`clctPretriggerTriggerZone`	—	5
`clctUseCorrectedBx`	—	True

- `clctFifoTbins` — width of FIFO data window in bunch crossing clocks
- `clctFifoPretrig` — width of FIFO data window in bunch crossing clocks where pretrigger can occur
- `clctHitPersist` — cathode strip hits are stretched to this number of bunch crossings
- `clctDriftDelay` — if pretrigger occured in BX N, then the occurence of trigger is checked in BX = N + clctDriftDelay
- `clctNplanesHitPretrig` — number of layers with hits required for pretriggering
- `clctNplanesHitPattern` — number of layers with hits required for triggering
- `clctPidThreshPretrig`: increase pattern ID threshold from 2 to 4 to trigger on higher pt tracks
- `clctMinSeparation`: minimal allowed separation in half-strips between two CLCTs in one BX
- `verbosity` — turns on debugging messages during CLCT processing
- `useDeadTimeZoning`: in 2007 algorithm after a CLCT trigger occured the whole CLCT processor is frozen for several bunch crossings until the number of layers with hits drops below the triggering threshold, in SLHC algorithm strips within certain dead zone are marked as busy:
  - `useDeadTimeZoning` — enables usage of such a dead zone
  - `clctStateMachineZone` — width of a fixed dead zone around a key half-strip
    (in half-strips)
  - `useDynamicStateMachineZone`: use variable dead zone which depends on the width of triggered CLCT pattern (changes from 10 half-strips for the most bent patterns to 2 half-strips for the most straight pattern)

  - `clctPretriggerTriggerZone`: width of a fixed dead zone around a key pretrigger
    (in half-strips)
- `clctUseCorrectedBx`: "corrected" CLCT stub time — use median time of all hits to set CLCT time

B.4 Configuration of the ALCT and CLCT Matching Emulator

Configuration parameters for the ALCT and CLCT matching part of the algorithm processing are the following:

	TMB07	SLHC
`mpcBlockMe1a`	0	
`alctTrigEnable`	0	
`clctTrigEnable`	0	
`matchTrigEnable`	1	
`matchTrigWindowSize`	7	3
`tmbL1aWindowSize`	7	
`verbosity`	0	
`tmbEarlyTbins`	4	
`tmbReadoutEarliest2`	True	False
`tmbDropUsedAlcts`	True	False
`clctToAlct`	—	False
`tmbDropUsedClcts`	—	False
`matchEarliestAlctME11Only`	—	False
`matchEarliestClctME11Only`	—	False
`tmbCrossBxAlgorithm`	—	True
`maxME11LCTs`	—	2

- `mpcBlockMe1a` — disables reporting LCTs found in ME1a
- `alctTrigEnable` — enables ALCT-only LCTs if there are no CLCTs
- `clctTrigEnable` — enables CLCT-only LCTs if there are no ALCTs
- `matchTrigEnable` — enables LCTs constructed from valid ALCTs and CLCTs
- `matchTrigWindowSize`: rather wide time matching window (7BX) in 2007 algorithm , decrease to 3BX in SLHC algorithm
- `tmbL1aWindowWidth` — width of L1A window in bunch crossing clocks, the L1 window spans from tmbEarlyTbins to tmbEarlyTbins+alctL1aWindowWidth
- `verbosity` — turns on debugging messages during TMB emulation
- `tmbReadoutEarliest2`: read out only first 2 LCTs in L1A window
- `clctToAlct`: stub matching logic — either CLCT-centric or ALCT-centric
  - CLCT-centric matching, default non-upgrade behavior
  - ALCT-centric matching, recommended for SLHC
- `tmbDropUsedAlcts`: in CLCT-centric matching, whether to use already matched ALCTs for further matching, analogously for tmbDropUsedClcts
- `matchEarliestAlctME11Only`: in CLCT-centric matching in ME11, break after finding the first BX with matching ALCT, analogously for matchEarliestClctME11Only
- `tmbCrossBxAlgorithm`: instead of matching CLCTs to ALCTs (or vise versa) in the order of arrival time, start matching in the central BX, then in the previous and the next BX
- `maxME11LCTs`: how many maximum LCTs per whole ME11 chamber per BX to keep (ME1b and ME1a can have max 2 each)

C More Results of Improvements in the ME1/1 Local Trigger Algorithm

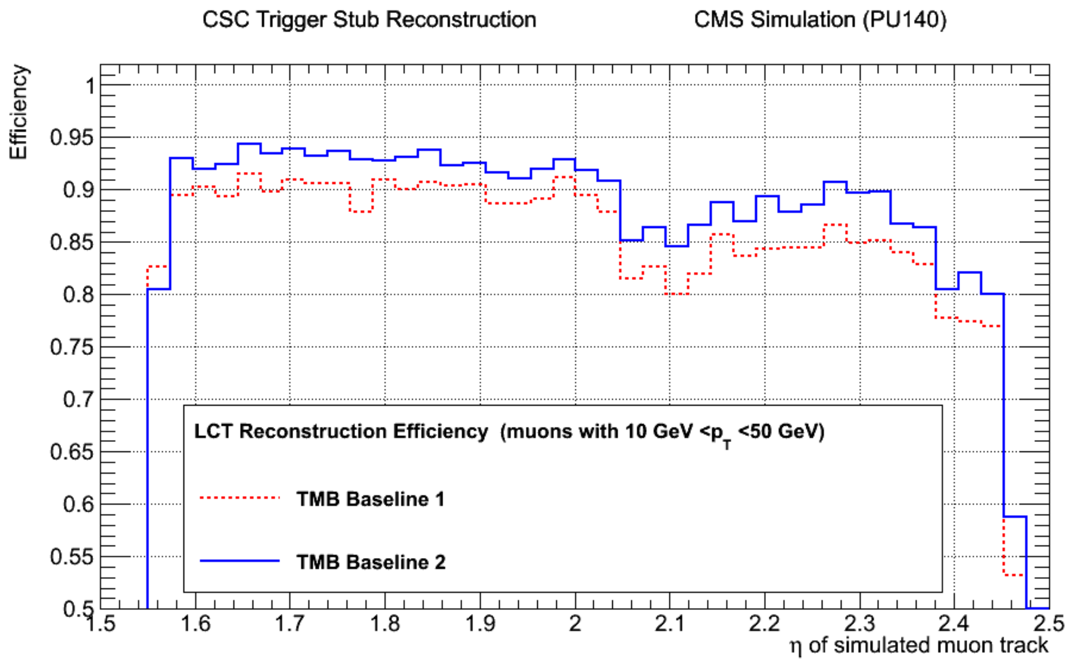
This chapter presents results of the study of effects of individual improvements described in Sec. 4.1 on the LCT reconstruction efficiencies.

Three are three baseline configurations of L1 step used in this study:

- Baseline 1: SLHC configuration, where the maximum set of improvements is turned off bringing it to 2007 configuration as close as possible. There are only two differences between Baseline 1 and 2007 configurations: separate treatments of ME1a and ME1b, and unganging cathode strips in ME1a;
- Baseline 2: Baseline 1 configuration with all improvements on the ALCT and CLCT processors level turned on;
- SLHC configuration itself.

Improvements on the level of TMB are related to the following configuration parameters (see Sec. **??**):

- matchTrigWindowSize: 7BX to 3BX;
- alctUseCorrectedBx: False to True
- clctUseCorrectedBx: False to True;
- clctToAlct: True to False;
- tmbDropUsedClcts and matchEarliestClctME11Only: True to False;
- tmbCrossBxAlgorithm: 0 to 1;
- tmbReadoutEarliest2: True to False;



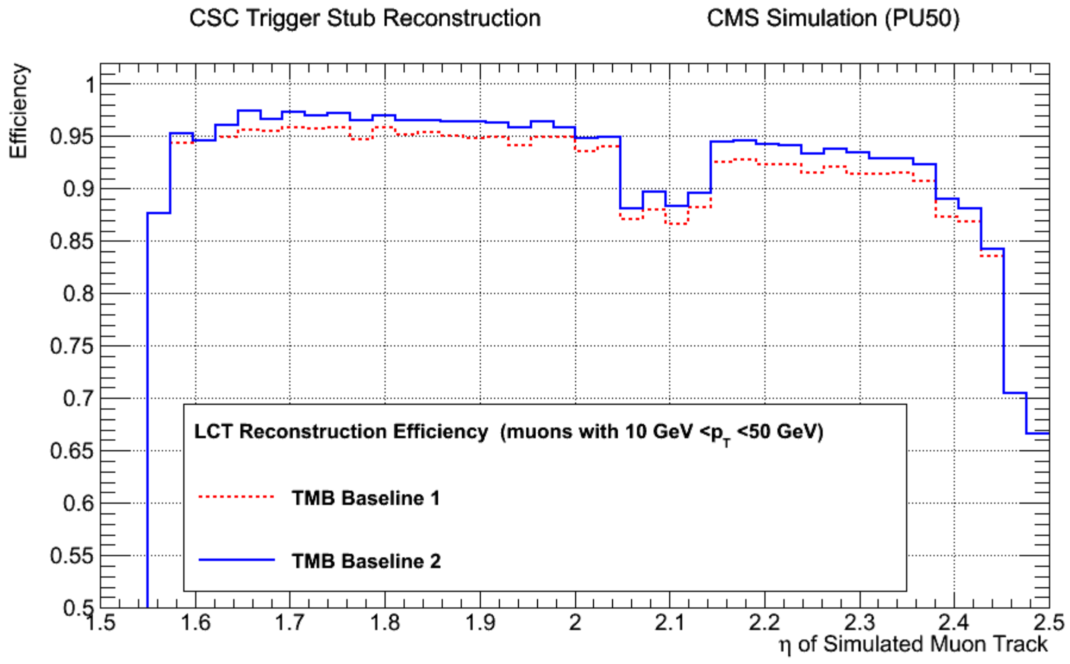
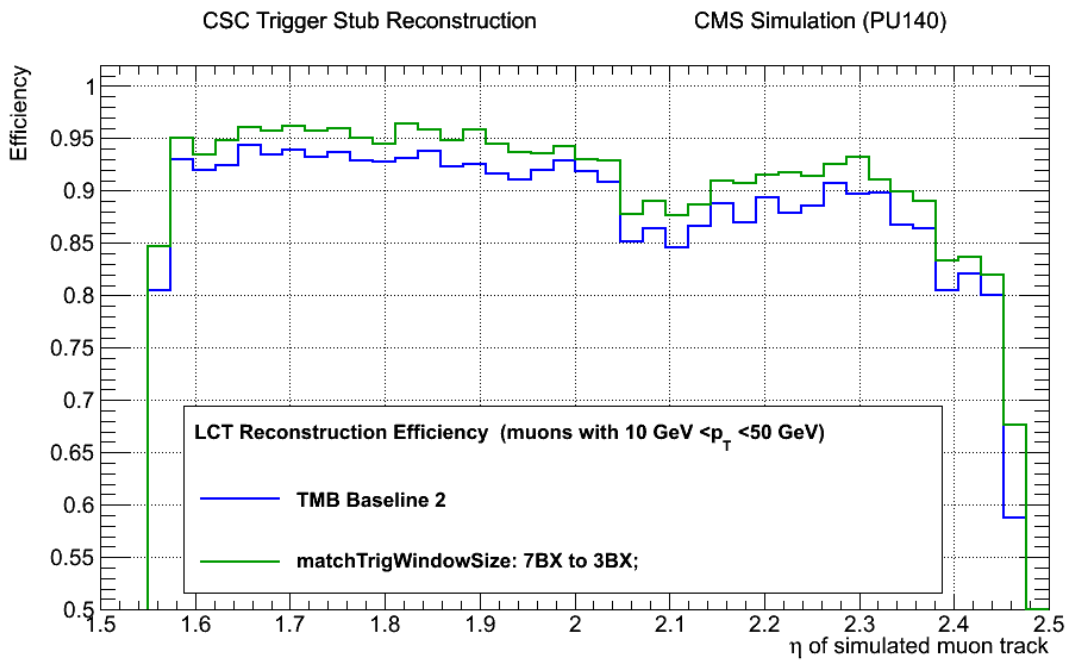


Figure 30: LCT reconstruction efficiency in ME1/1 station for PU140 (top) and PU50 (bottom). The efficiencies increase due to change from TMB Baseline 1 configuration to TMB Baseline 2 configuration. Muons with transverse momentum 10 GeV$< p_T < 50$ GeV are used in the analysis.



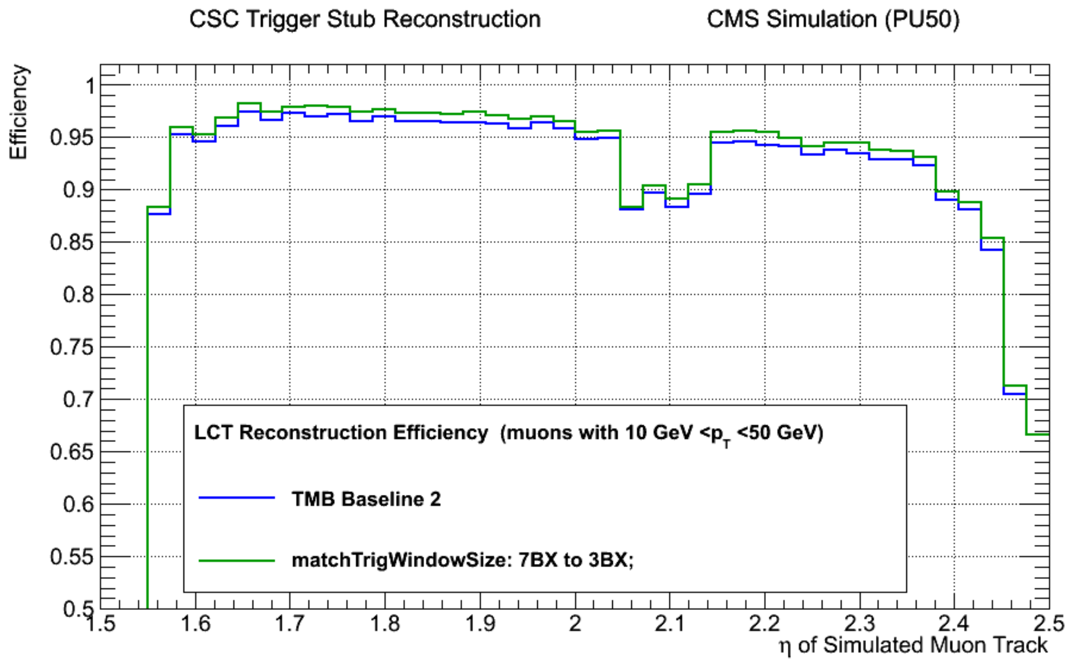
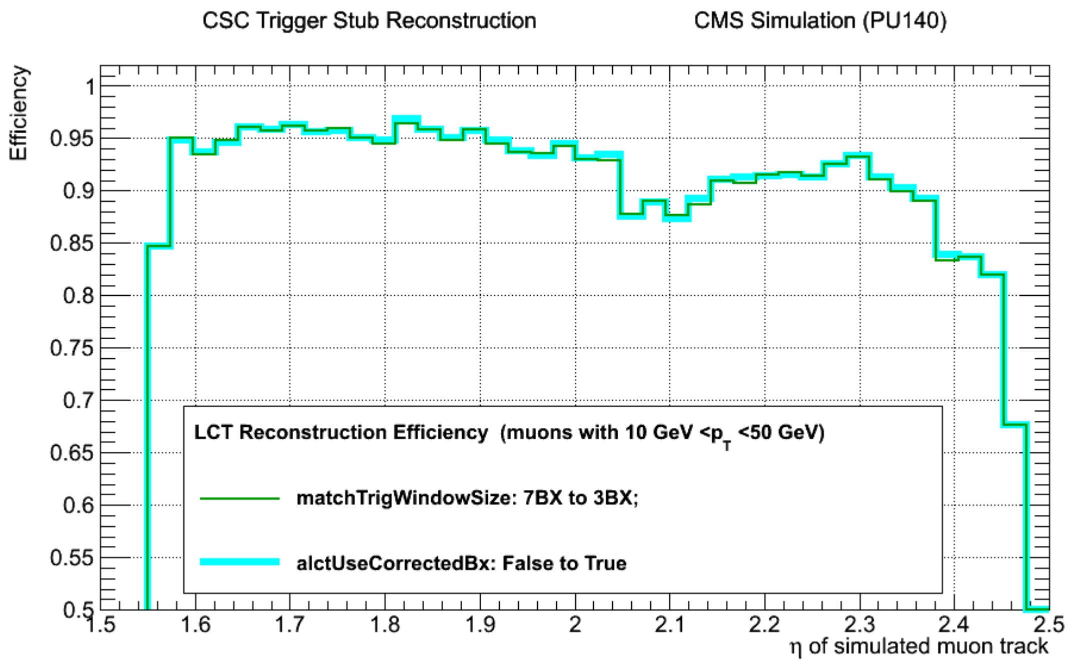


Figure 31: LCT reconstruction efficiency in ME1/1 station for PU140 (top) and PU50 (bottom). The efficiencies increase due to change of matchTrigWindowSize from 7BX to 3BX. Muons with transverse momentum 10 GeV$< p_T < 50$ GeV are used in the analysis.



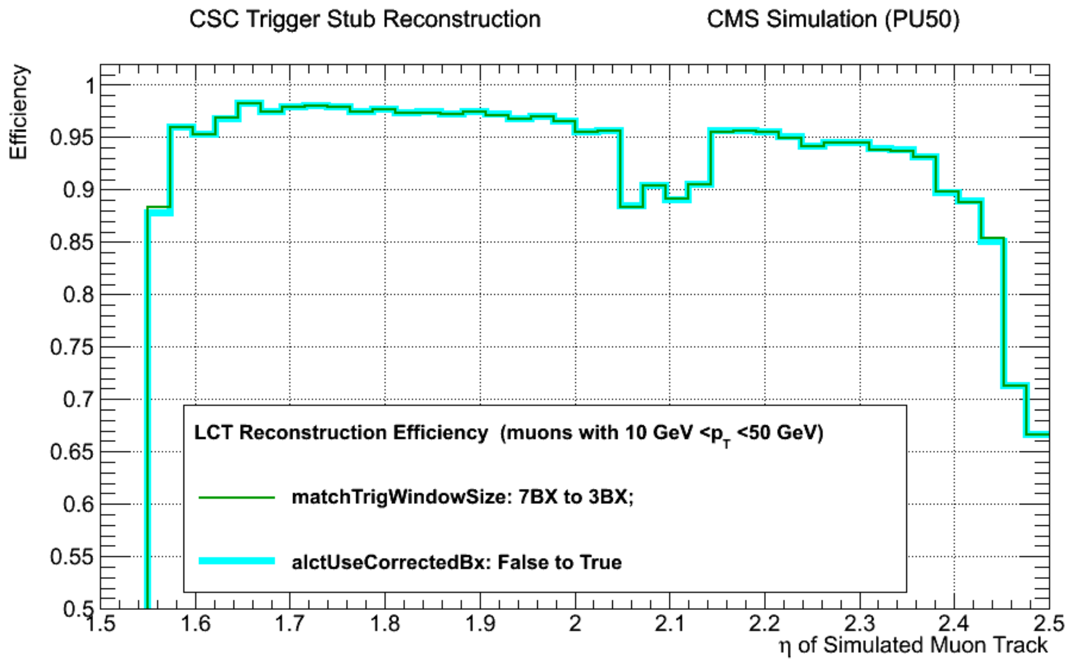
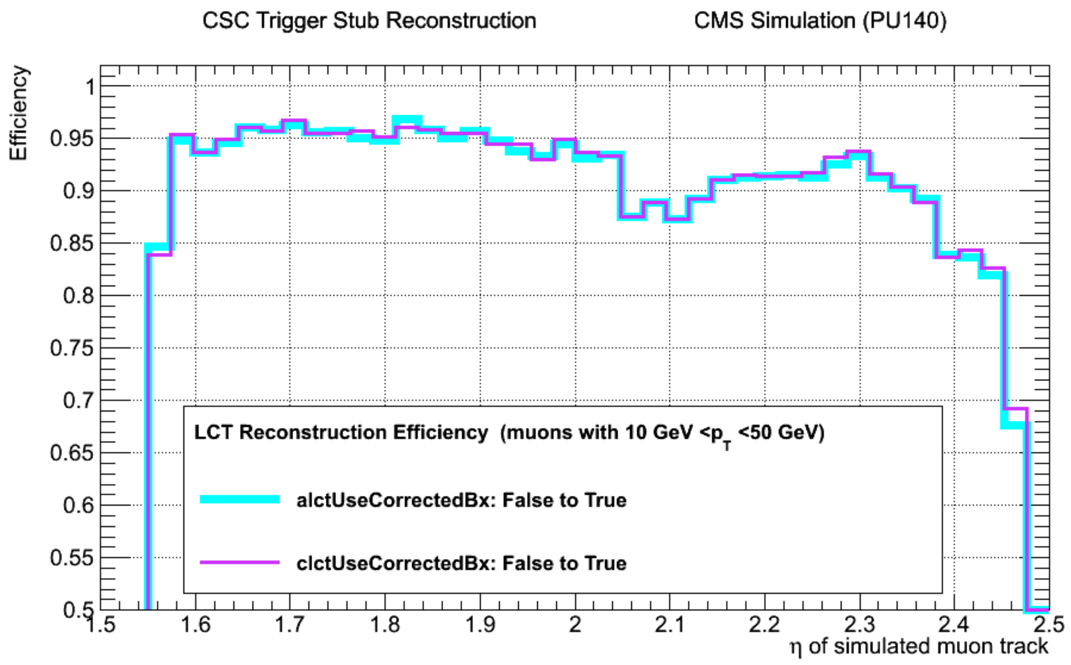


Figure 32: LCT reconstruction efficiency in ME1/1 station for PU140 (top) and PU50 (bottom). The efficiencies increase due to change of alctUseCorrectedBx from False to True. Muons with transverse momentum 10 GeV$< p_T < 50$ GeV are used in the analysis.



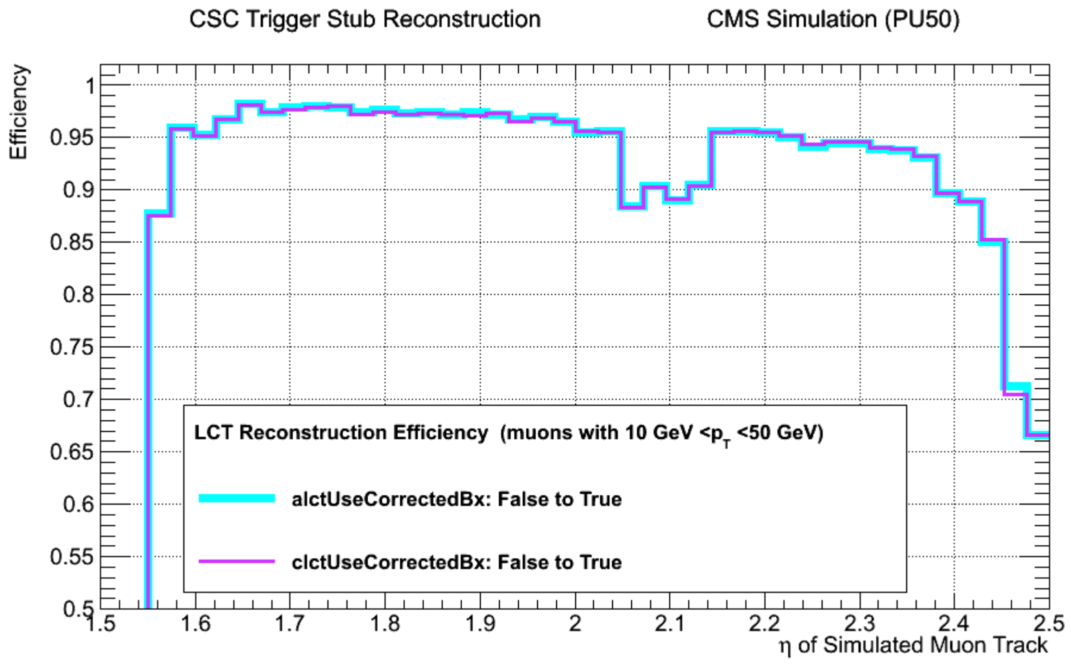
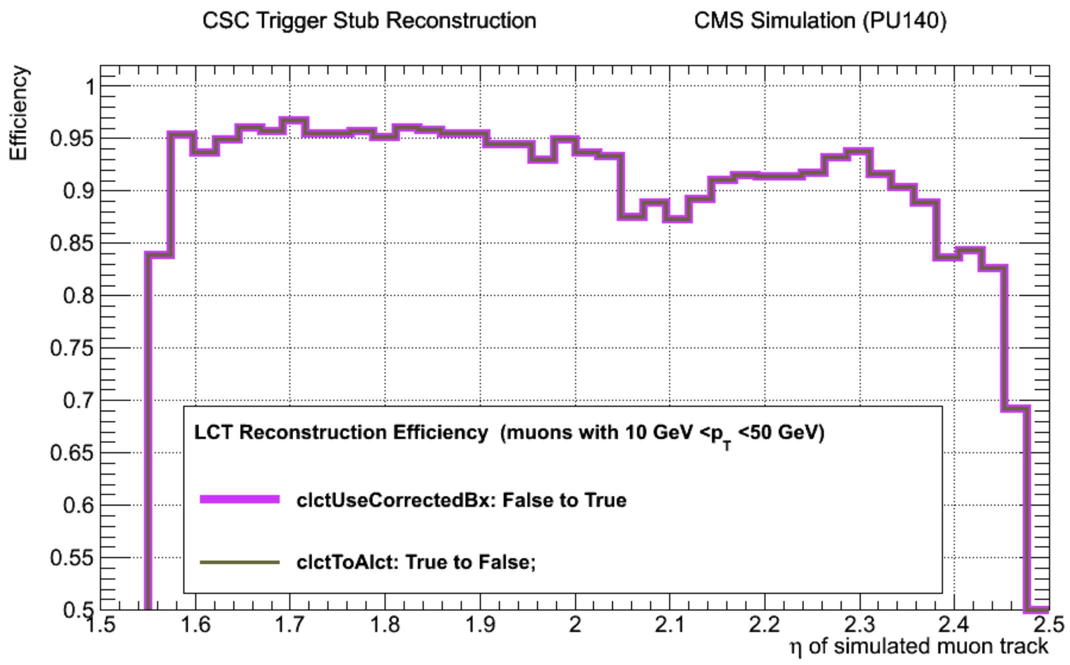


Figure 33: LCT reconstruction efficiency in ME1/1 station for PU140 (top) and PU50 (bottom). The efficiencies increase due to change of clctUseCorrectedBx from False to True. Muons with transverse momentum 10 GeV$< p_T < 50$ GeV are used in the analysis.



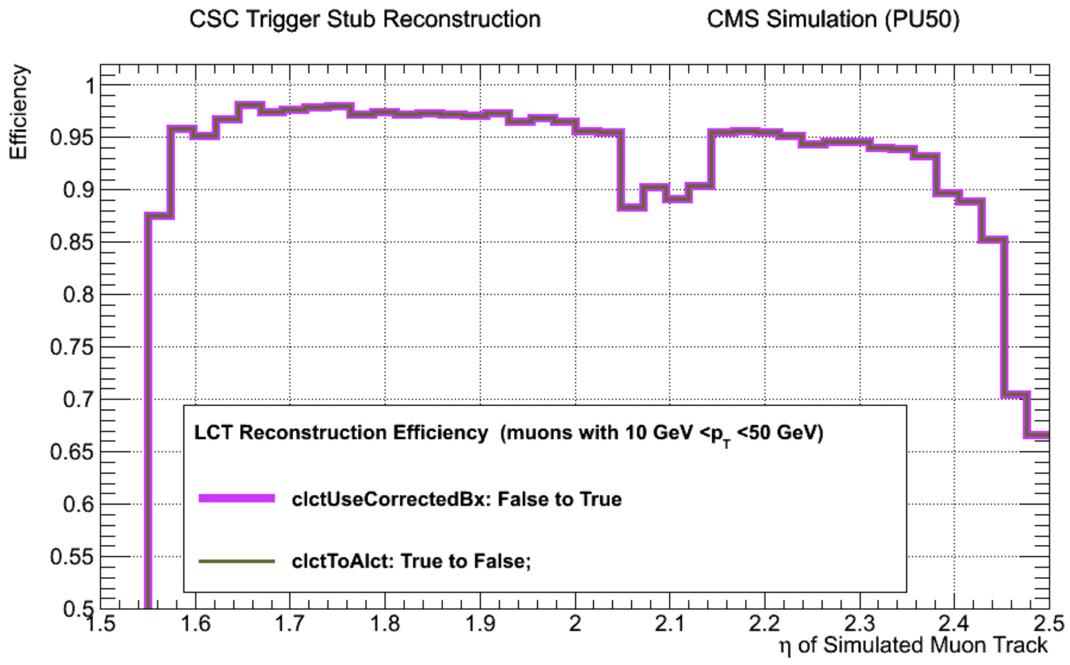
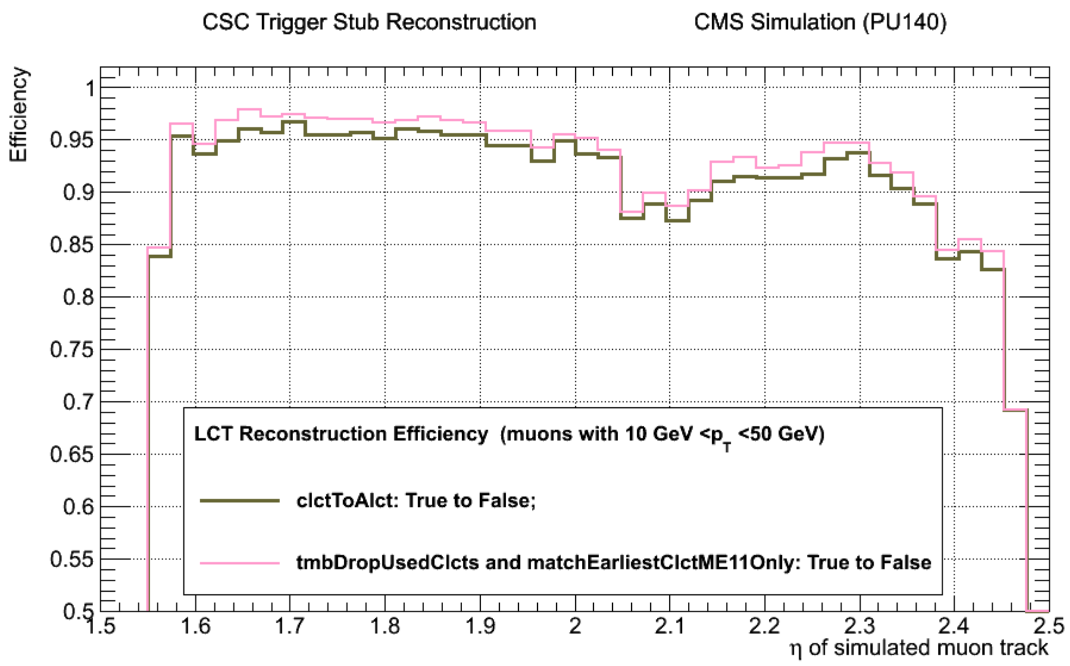


Figure 34: LCT reconstruction efficiency in ME1/1 station for PU140 (top) and PU50 (bottom). The efficiencies increase due to change of clctToAlct from True to False. Muons with transverse momentum 10 GeV$< p_T < 50$ GeV are used in the analysis.



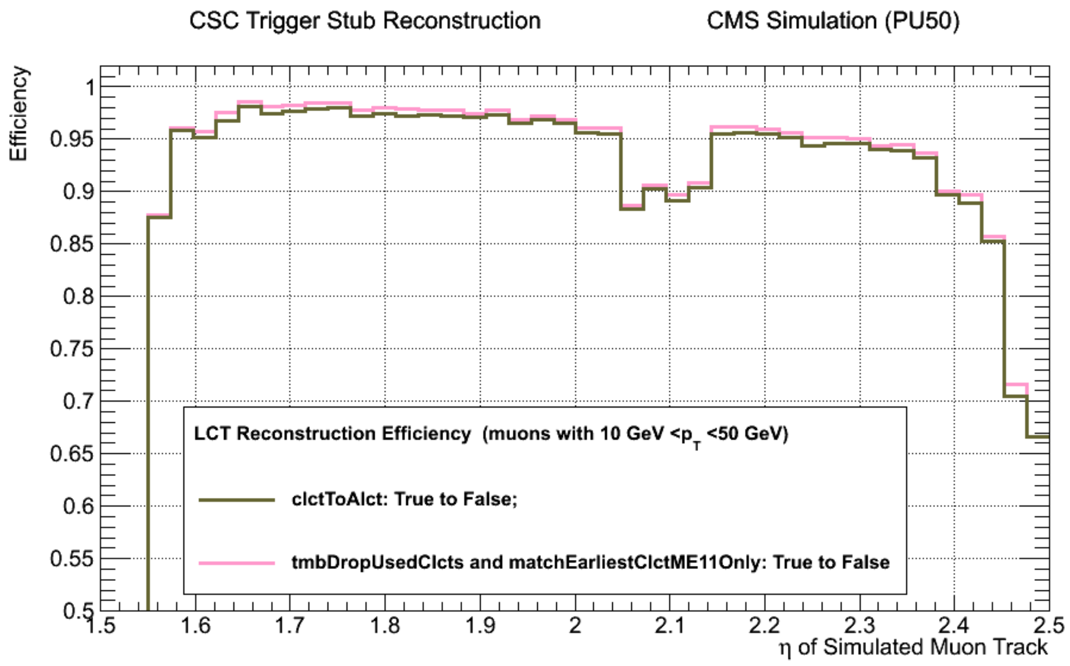
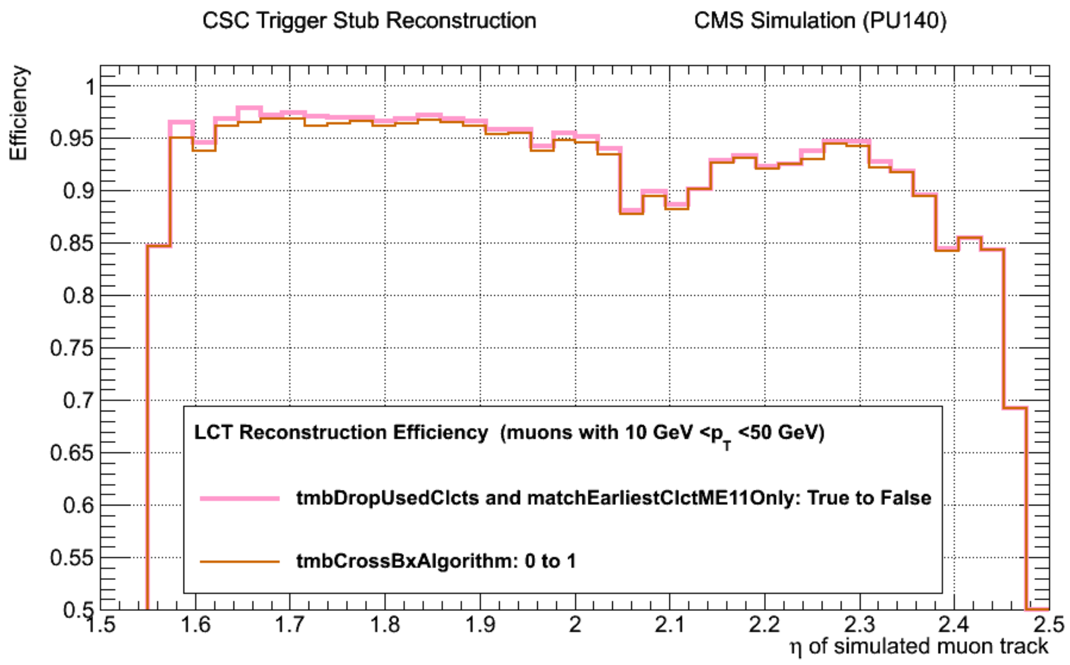


Figure 35: LCT reconstruction efficiency in ME1/1 station for PU140 (top) and PU50 (bottom). The efficiencies increase due to change of tmbDropUsedClcts and matchEarliestClctME11Only from True to False. Muons with transverse momentum 10 GeV$< p_T < 50$ GeV are used in the analysis.



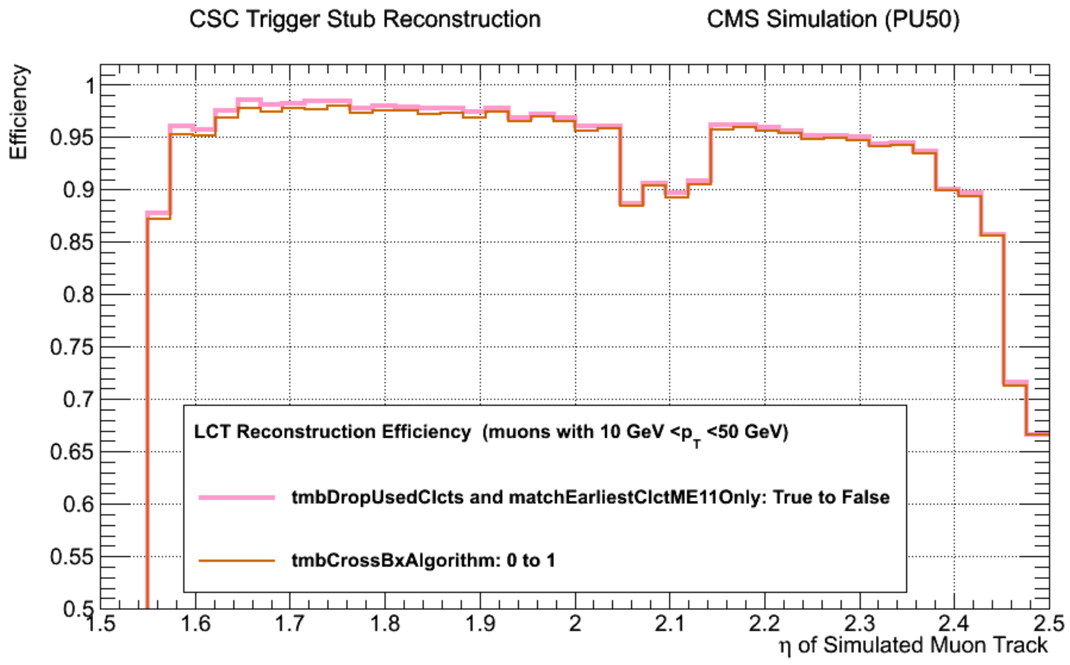
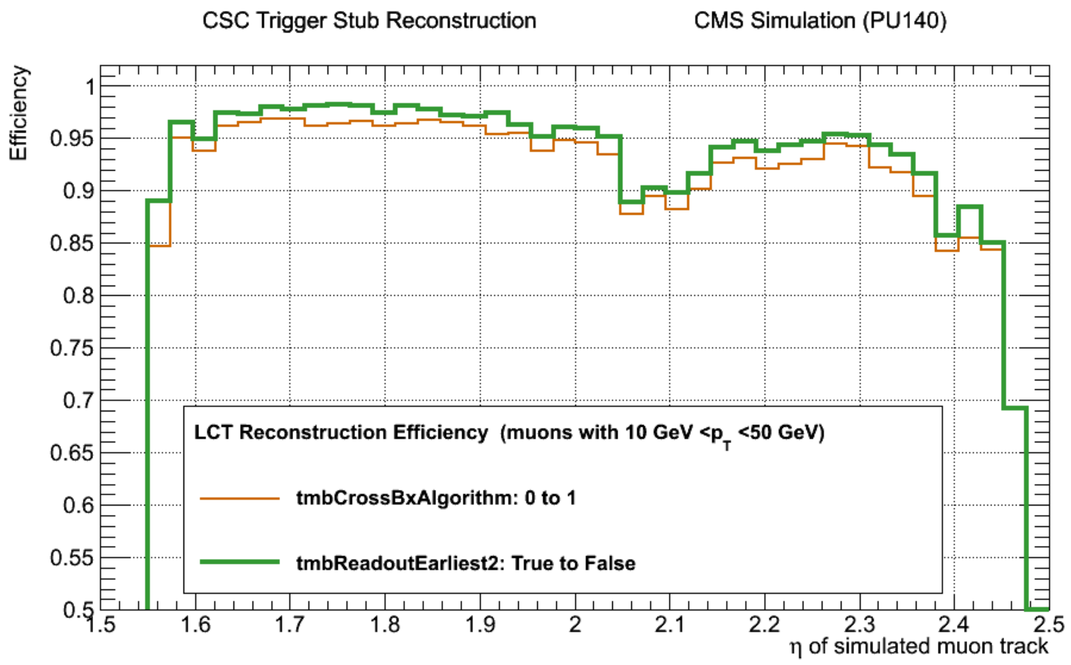


Figure 36: LCT reconstruction efficiency in ME1/1 station for PU140 (top) and PU50 (bottom). The efficiencies increase due to change of tmbCrossBxAlgorithm from 0 to 1. Muons with transverse momentum 10 GeV$< p_T < 50$ GeV are used in the analysis.



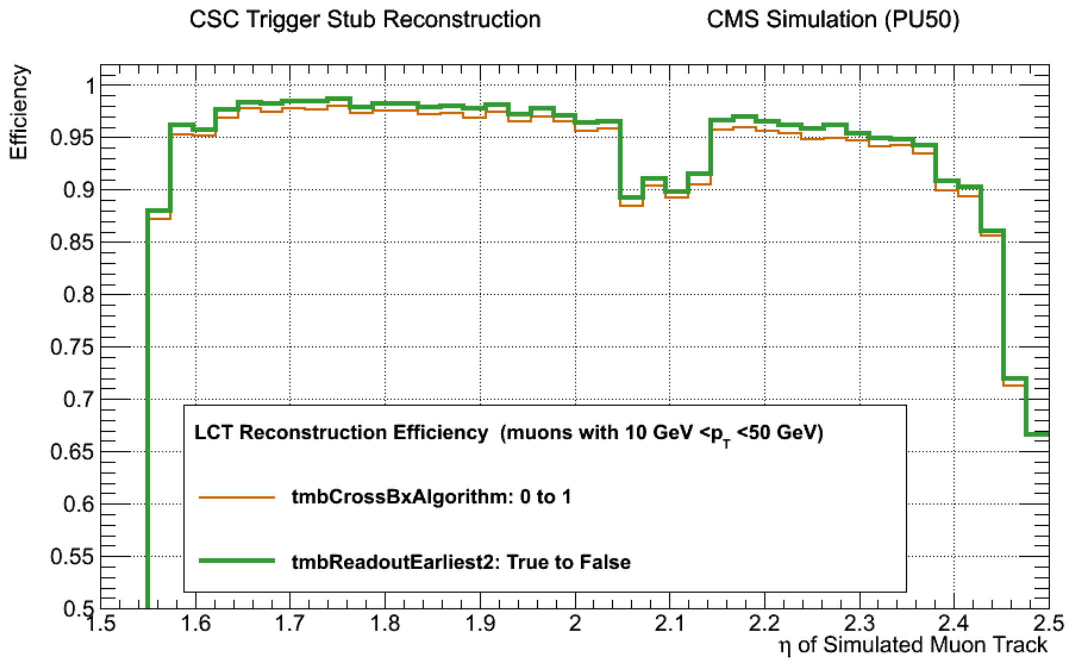
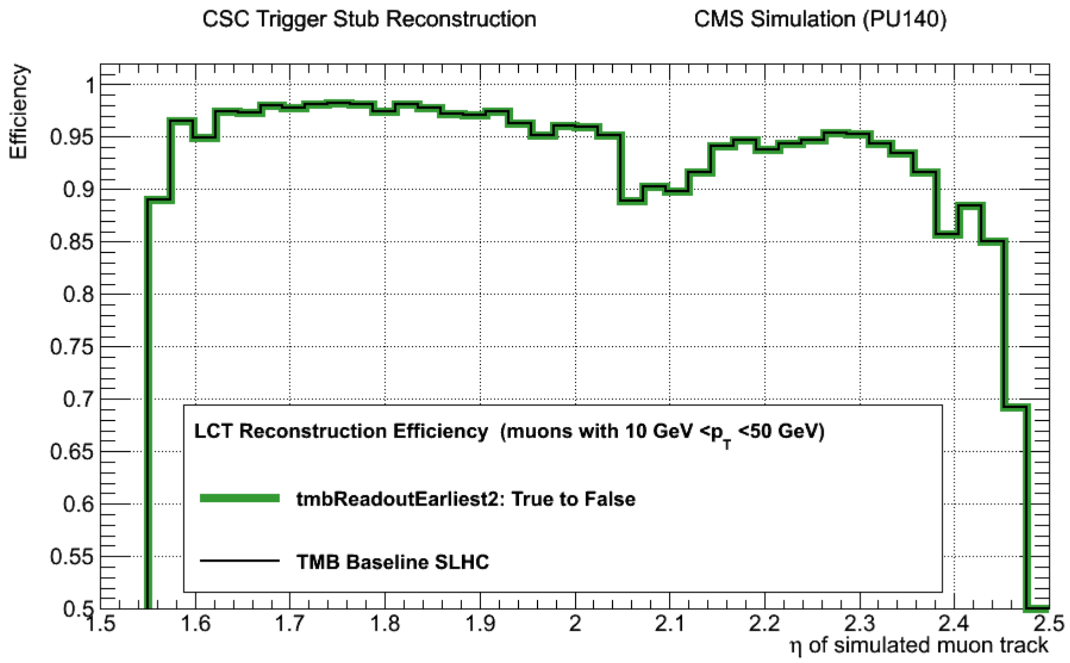


Figure 37: LCT reconstruction efficiency in ME1/1 station for PU140 (top) and PU50 (bottom). The efficiencies increase due to change of tmbReadoutEarliest2 from True to False. Muons with transverse momentum 10 GeV$< p_T < 50$ GeV are used in the analysis.



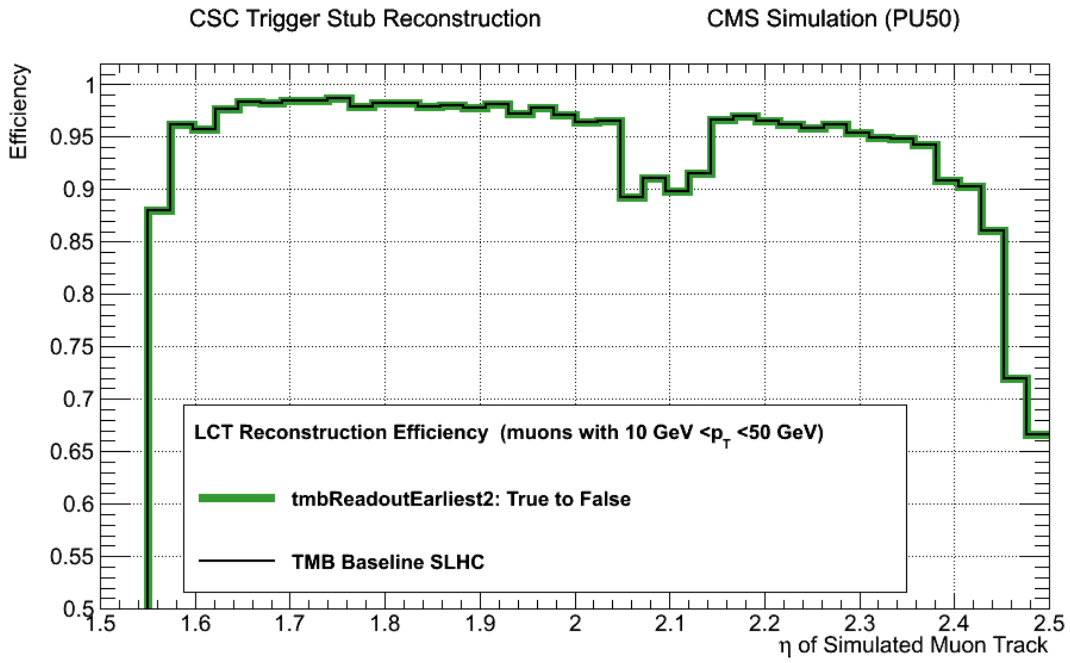


Figure 38: LCT reconstruction efficiency in ME1/1 station for PU140 (top) and PU50 (bottom). After the last improvment we are on the current TMB SLHC suggested configuration therefore it's expected that the efficiencies agree. Muons with transverse momentum 10 GeV$< p_T <$ 50 GeV are used in the analysis.