CSC Trigger Primitives Efficiency Studies

Vadim Khotilovich

Alexei Safonov, Jason Gilmore

Texas A&M University

CMS Upgrade Workshop
October 28-30, 2009

Outline

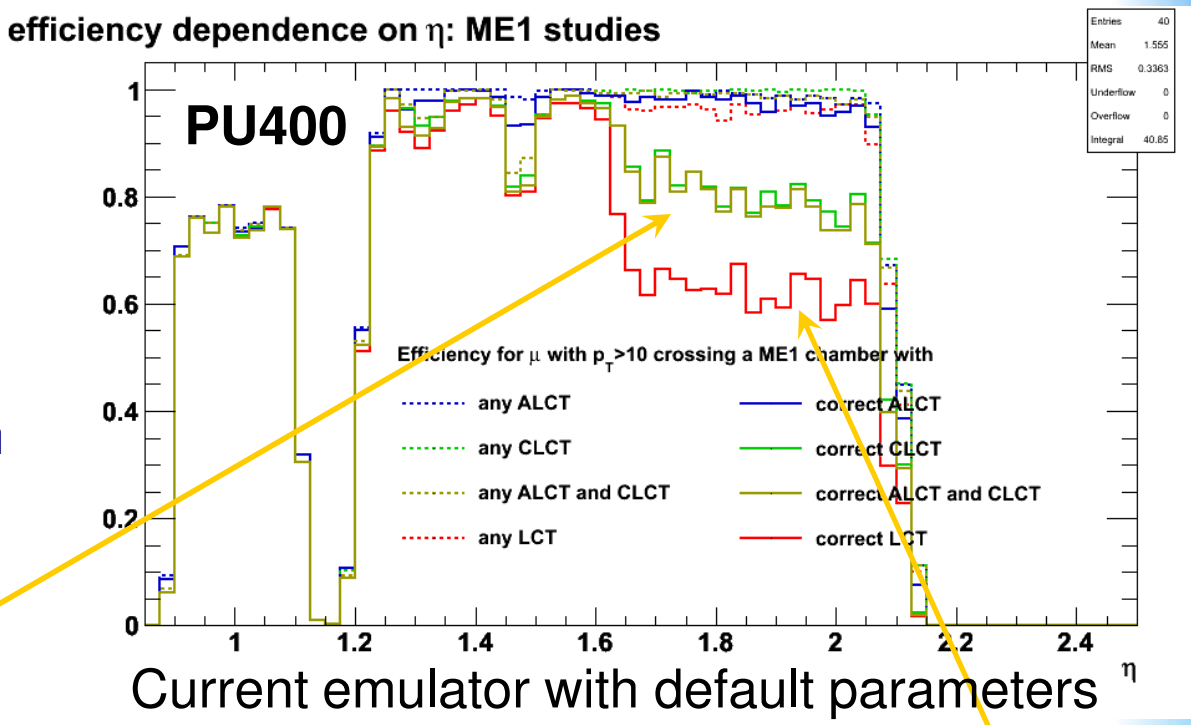
- Motivations for upgrade
- Default emulator efficiency in high pileup
- Restoring triggering in ME1/a
- CLCT improvements
- Improving ALCT efficiency at higher η
- Stronger Pre-trigger requirement as a simple improvement of time resolution
- Efficiency loss in LCT finding
  - Multiple ALCTs in match window
  - ALCT time resolution deterioration in ME1/a
  - Maximum # of LCTs per ME1/1
- Trigger primitives timing synchronization
- Plans
- Summary

Motivation for CSC L1 Trigger Upgrade

- Insufficient redundancy at around η~1.5-1.8 requires addition of ME4/2
  - **Done in simulation**
- Reinstate trigger in η~2.1-2.4 (ME1/a)
  - Requires upgrade of TMB and CSCTF
    - **We propose an algorithm for TMB**
- High ME1/1 occupancies/rates
  - Require upgrade of CFEBs to DCFEBs
    - That forces changes in TMB, DMB etc.
  - Lead to deterioration of trigger performance above η~1.6
    - Solution requires upgrade of TMB
  - Substantial increase in stub occupancies forces upgrade of CSCTF
  - **This talk:** full simulation studies to restore trigger primitives (TP) finding efficiency in ME1/1

Efficiency for Default CSC TP Emulator

- **Focus of this talk: ALCT, CLCT, LCT**
- L1 may perform only as good as its low level primitives
  - Finding correct TPs in ME1 greatly affects performance of track finder
  - CLCT finding efficiency takes serious dip in ME1/b region at high PU
    - "Dead-time" in TMB algorithm is the main reason
    - Upgraded TMB would allow improved algorithm



Current emulator with default parameters

- ME1/a signal is handled ineffectively and is not further used in Track Finder at all
  - ME1a strips from are appended to ME1b strips
  - ME1a strips and wires create "dead weight" that decreases efficiency & often leads to finding wrong LCT
  - Note: ALCTs **are** reconstructed in ME1a but not drawn

Matching TPs to Muon (Details)

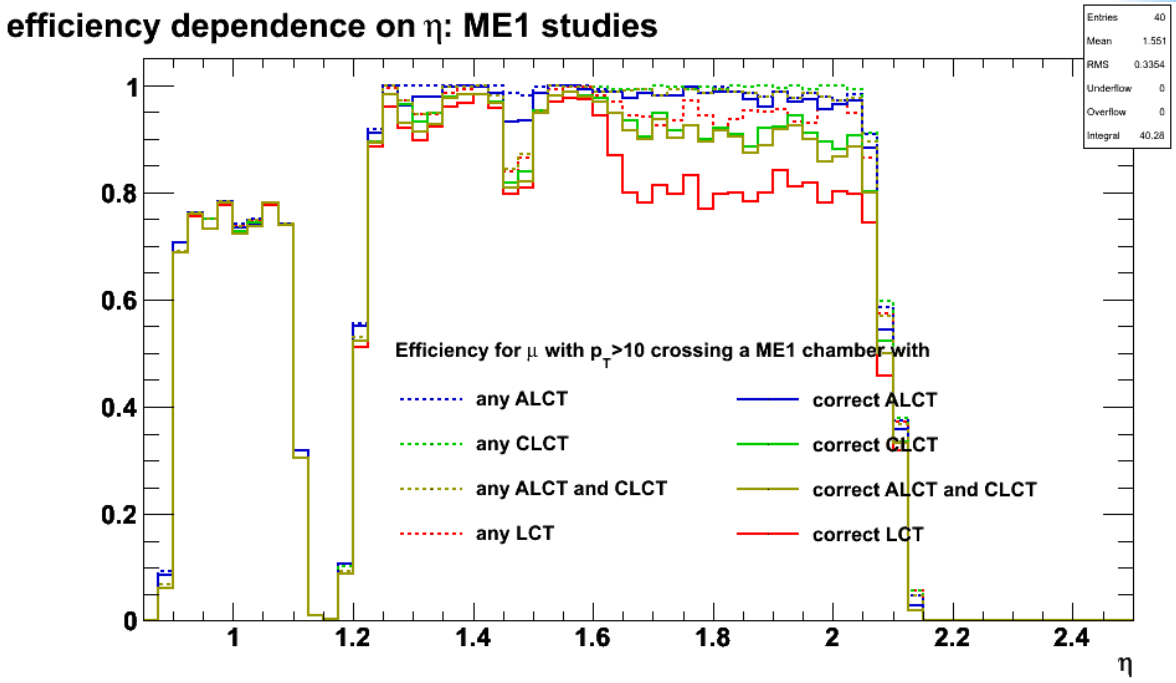
- We consider two matching options:
  - "**Any**" in chamber TP
    - TP is found in a chamber where muon SimTrack left some SimHits
  - "**Correct**" TP is defined as "Any" in chamber plus
    - ALCT: $-2 \leq WG(\text{of muons simhits}) - WG(\text{ALCT key layer}) \leq 0$
    - CLCT: $|Strip(\text{of muons simhits}) - Strip(\text{CLCT key layer})| \leq 1$
    - LCT: made by matching "correct" ALCT and CLCT
    - Inefficiency < 1%
- Note: "Correct" does not absolutely imply that a TP was definitely caused by a signal muon
  - It still could be caused by PU on the same wire/strip but in a different BX

Default Emulator: Parameter Adjusting

- Before big changes, let's adjust some default emulator parameters to get closer to SLHC conditions
  - Get rid of "dead weight" from ME1/a
    - Turn off ME1/a digis on input
    - We'll restore ME1/a later
  - Use more narrow ALCT-to-CLCT BX matching window
    - Better chance that resulting LCT is not fake at high PU conditions
    - We expect good synchronization of TPs at SLHC
  - Use more narrow TPs readout window widths
    - R/O window is centered in expected signal BX for all primitives
    - For high PU we want as narrow r/o window as possible (to reduce rates)

Default Emulator: ME1a off, 3BX Match

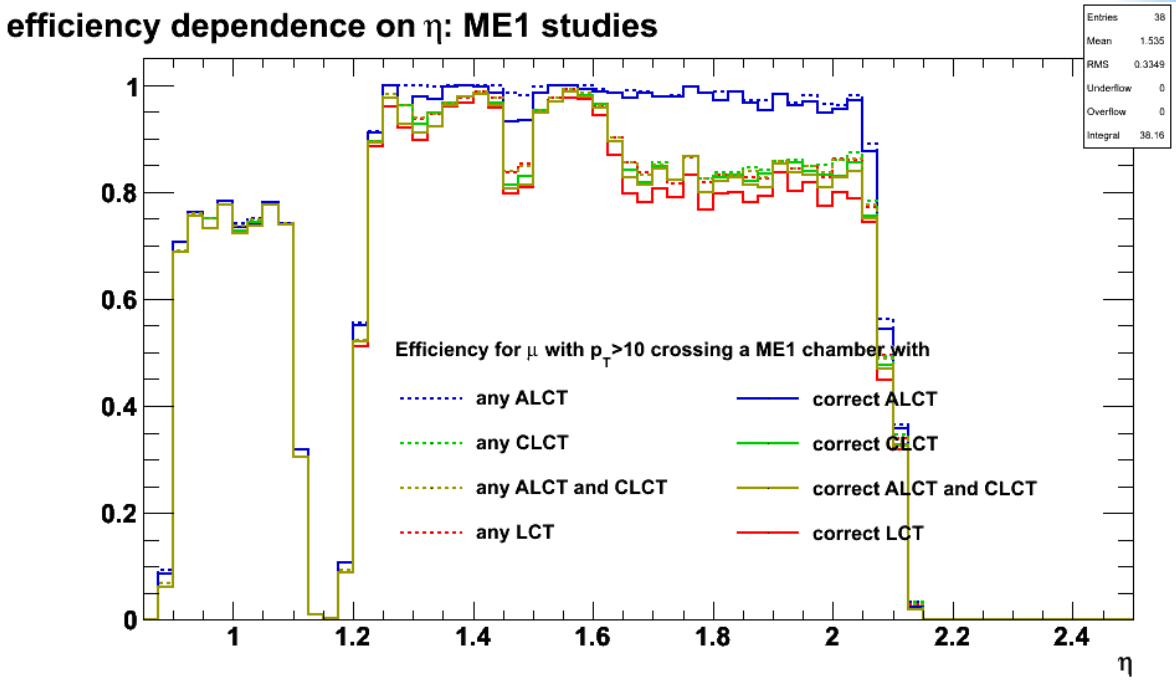
- Turn off digis from ME1a on input
- ALCT-to-CLCT matching window: 3BX instead of 7BX



- Notes:
  - Turning off “dead weight” from ME1a increases efficiency
  - More narrow matching window leads to
    - some decrease in efficiency due to time resolution
    - Improved “correctness” of LCTs

Default Emulator: Narrow R/O Window

- R/O Window width 3BX
- Notes:
  - All similar efficiency plots in the talk include all previous modifications
  - Noticeable drop of CLCT and “any” LCT finding efficiency
    - Caused by rejecting out of r/o window primitives from PU
  - “Correct” LCT efficiency stays almost the same
    - Ensured by ALCT-CLCT matching window of 3BX

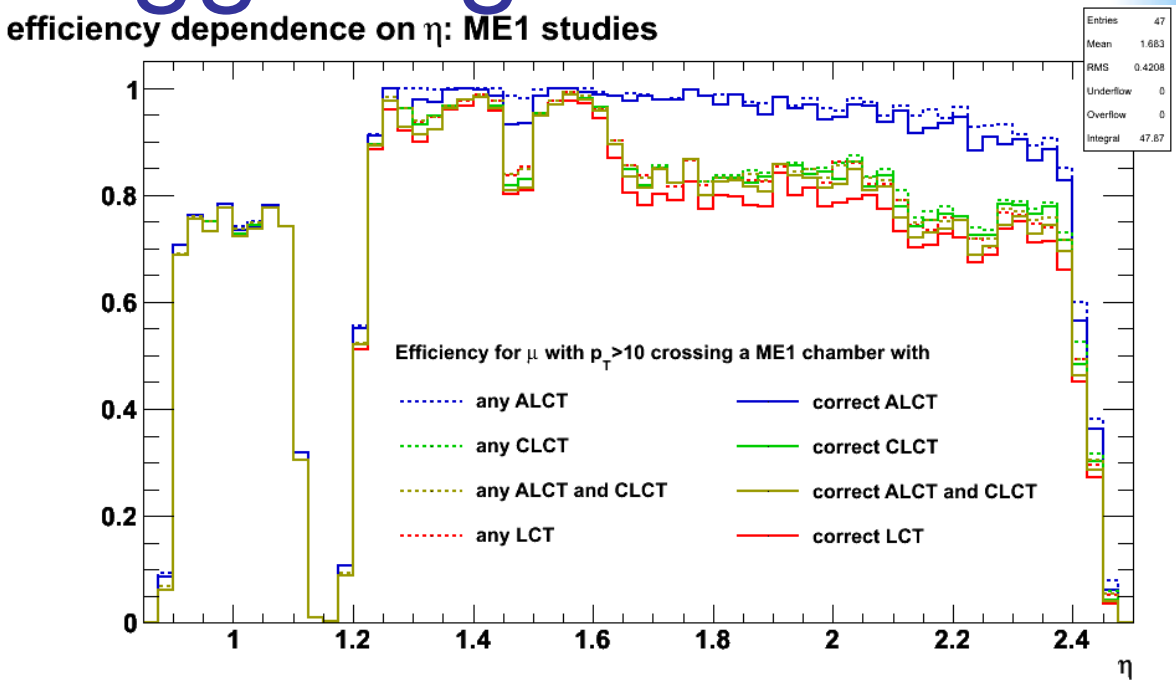


Restoring Triggering in ME1/a

- ME1/1 = ME1/a + ME1/b
  - Separate sets of strips in 1/b and 1/a
    - ME1/a strips had to be un-ganged in simulation
  - Continuous wire groups coverage
    - WG #11-16 cross the border

- Algorithm in a nutshell (see backup for more):
  - Find ALCTs in ME1/1
    - Assign them to ME1/a or ME1/b (with some overlap)
  - Find CLCTs in ME1/a and ME1/b
  - Match ALCTs to CLCTs separately in ME1/a and ME1/b
    - Properly handle possible case of ALCT and CLCT being not able to physically cross
  - Choose the best two LCTs in each, ME1/a and ME1/b
  - One more sorting step is necessary if we are allowed max 2 per whole ME1/1

Restoring Triggering in ME1/a



- Both, A&CLCT efficiencies drop slightly more in ME1/a comparing to ME1/b

CLCT Improvements

- Reducing TMB “dead-time”, main inefficiency source
- Optimizing pattern bend
  - Reducing rate while not sacrificing the efficiency

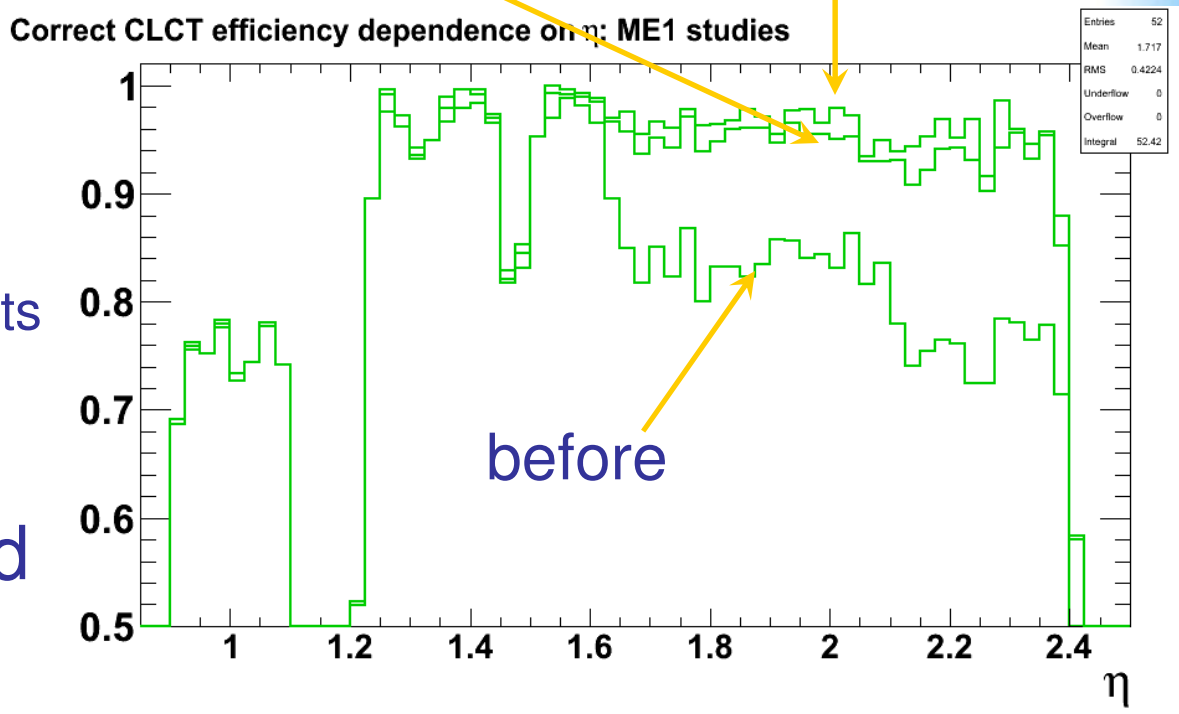
Origin of CLCT/TMB Dead-time

- (from Jay Hauser) TMB's state-machine freezes whole TMB for several BXs after a CLCT trigger while number of layers stays over trigger threshold
  - If an early CLCT comes from PU, it would be impossible to trigger on signal in a whole chamber during that time

Solution: Zoned Dead-Time

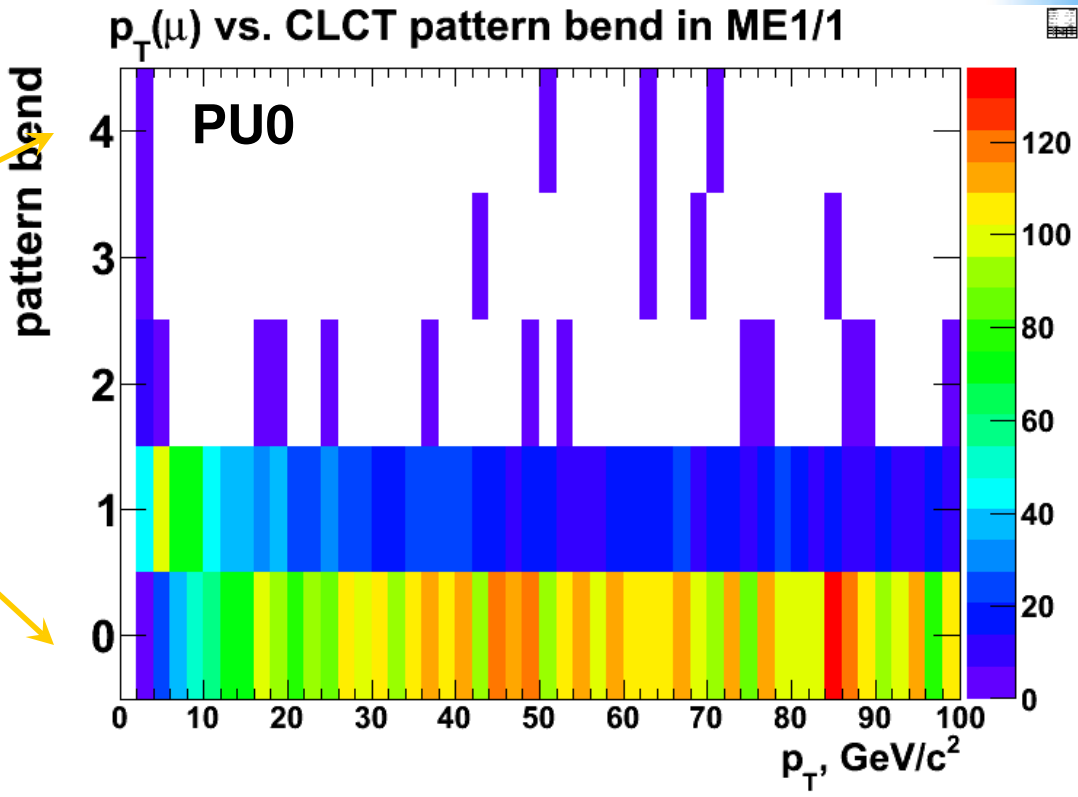
- "Zoned" dead-time algorithm:
  - After pre-trigger on some half-strip, TMB waits for 2BX drift time and checks trigger condition in half-strips window of ±5 around pre-trigger HS
  - If CLCT triggers, TMB can't trigger again only in ±8 HS zone around its triggered key HS
- Bigger FPGA would allow it

- "Dynamic zone" dead-time algorithm:
  - Instead of fixed dead-time zone width of 8 HS, make it = CLCT pattern width – 1
  - Slightly higher efficiency



CLCT Patterns Bending

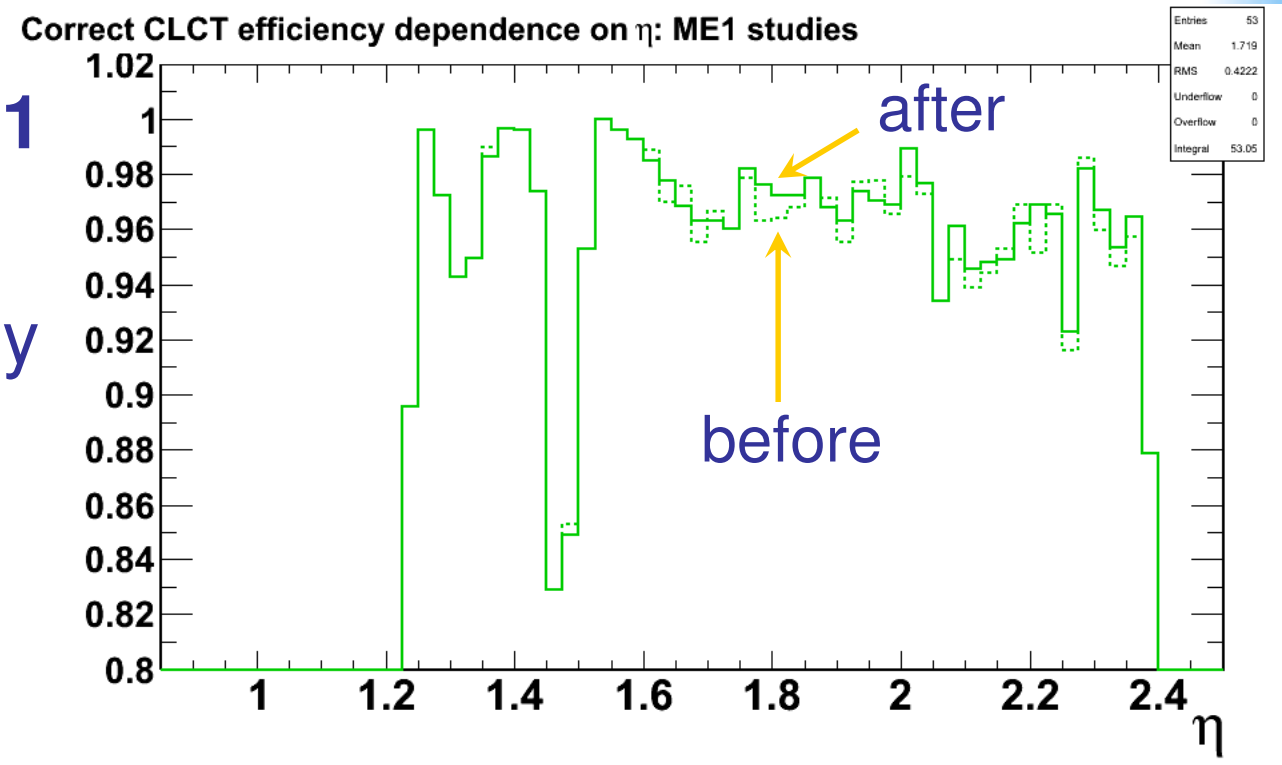
- 5 possible CLCT bend patterns
  - most bent
  - almost straight



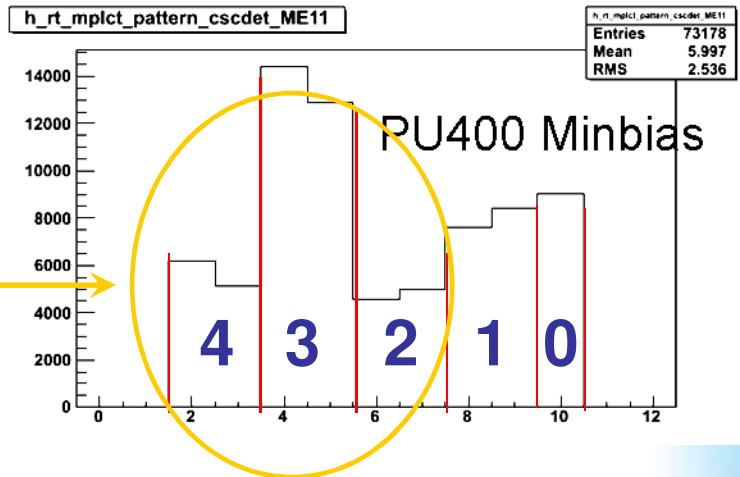
- Only the first two most narrow patterns really matter for $P_T > 4$ GeV

Optimization of CLCT Bending

- **CLCT bend ≤1**
- Does not really reduce or increase the efficiency



- **Very important for rate reduction!**
  - Most of CLCTs from PU have high bend



ALCT Improvements

- Reducing efficiency drop at higher η:
  - Ghost cancellation logic adjustment
  - Removing pre-trigger dead-time

ALCT Ghost Cancellation

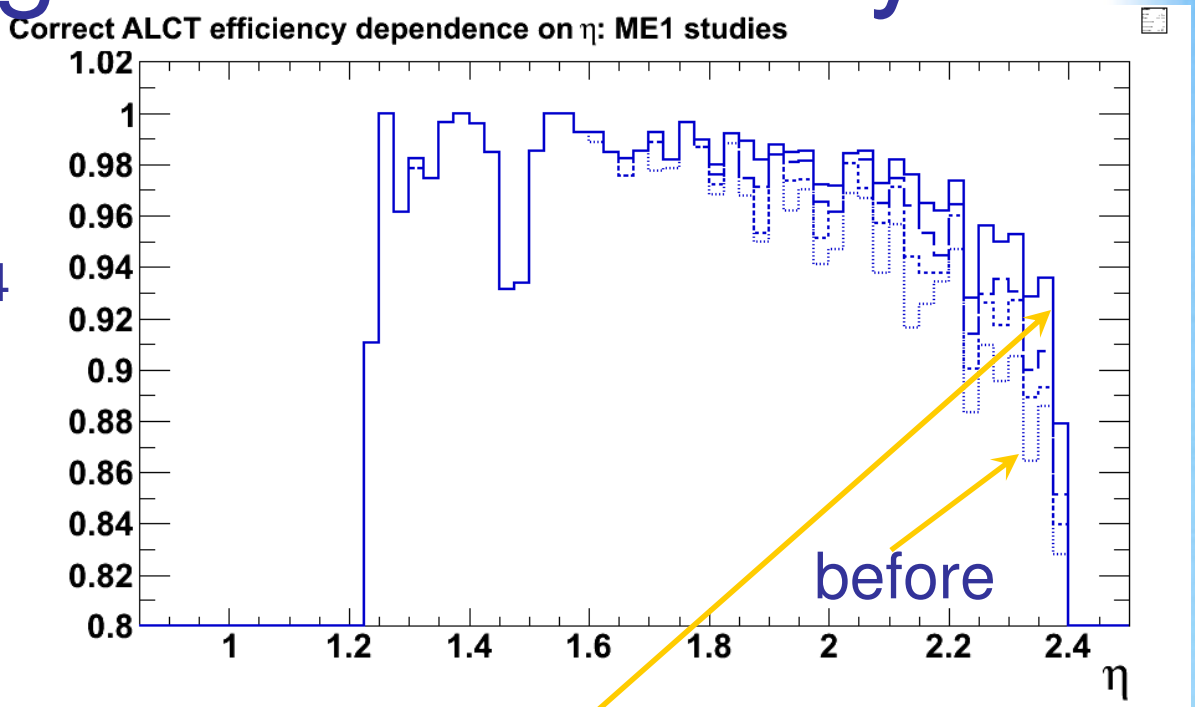
- Current algorithm:

```
If( WG-1 has ALCT of higher Quality in the same BX ||
    WG+1 has ALCT of at least the same Q in the same BX )
        ALCT on wire group WG is canceled
else (if WG+1 or WG-1 have ALCT from 1 to 4 BX ago)
        ALCT on wire group WG is canceled
```

- Is run in parallel for all Wire Groups
- Has little effect at low PU
- At high PU efficiency suffers because of the strict requirements in the '**else**' part
  - "from 1 to 4 BX ago" is a lot for time-precise ALCTs
  - No requirement on Quality of WG+1 or WG-1
    - Often leads to killing of a good signal ALCT by some low quality neighboring ALCT from early PU

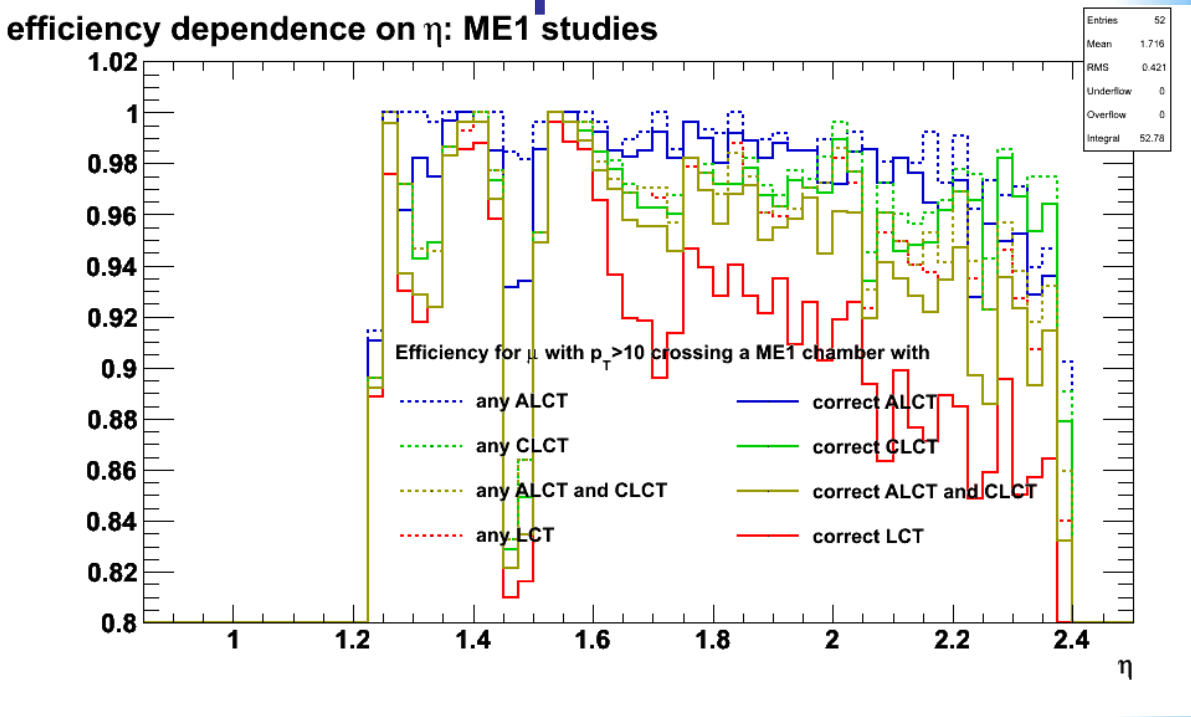
Improving ALCT Efficiency

- Ghost Cancellation:
  - Changing the WG±1 look-back ΔBX from 4 to 1 (dotted line)
  - Adding higher quality requirement for past BXs (dashed)



- Removing pre-trigger dead-time:
  - Currently, if pre-trigger happens on some WG but no trigger 2BX later, ALCT stops this WG for total 6 BX after pre-trigger
  - No reason not to reduce it to 2 BX (drift time only)

After CLCT & ALCT Improvements



- ~ 10% - 15% increase in correct LCT finding efficiency comparing to the “Restoring triggering in ME1/a” plot

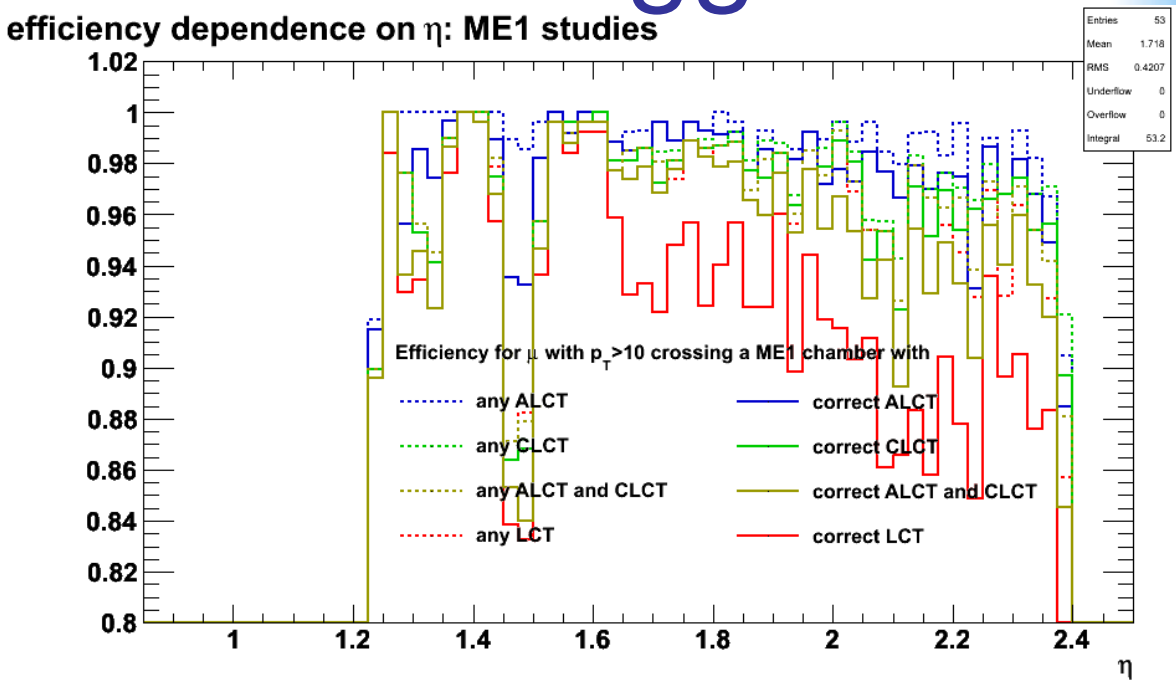
Timing Resolution Improvement

- The better timing resolution:
  - the better LCT matching
  - the more of signal's TPs fall into r/o window
  - the higher efficiency

- Simple first step: increasing min # layers in time for pre-trigger requirement
  - So far we used min 2 layers for A&CLCTs
  - As TP time is defined by its pre-trigger time, more layers in pre-trigger means more timing precision

3 Layers for Pre-Trigger

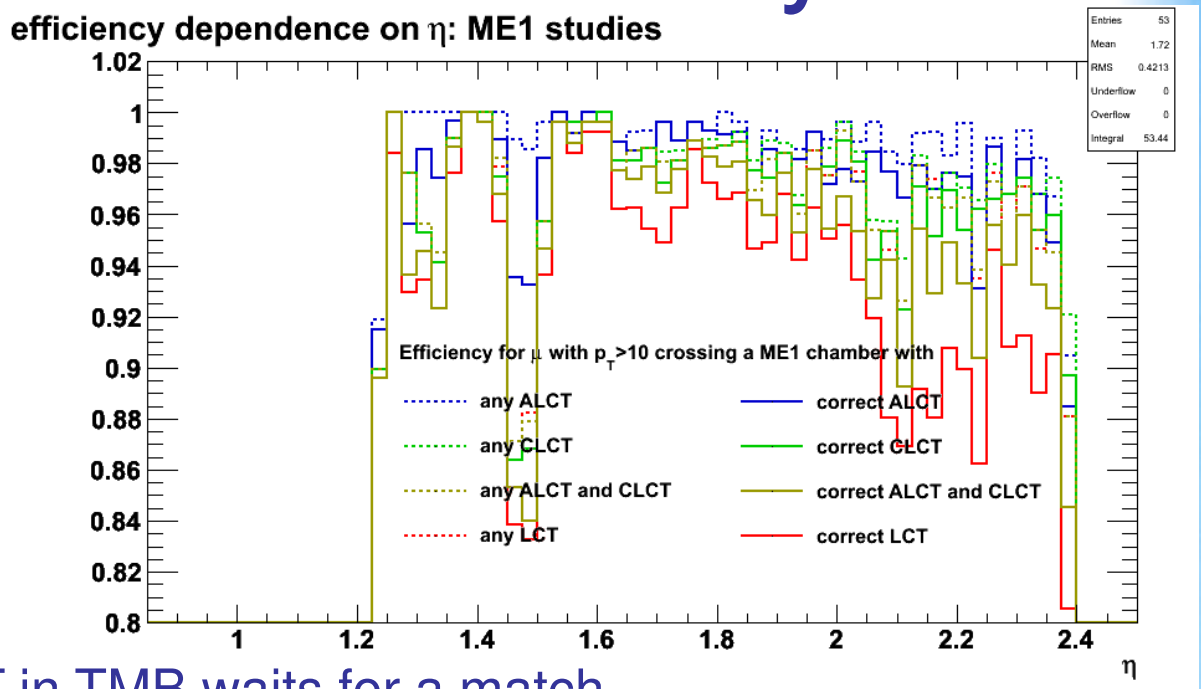
- Pre-trigger requirement: min 3 layers in time



- ~1.5% increase in ME1/1 LCT efficiencies
  - Mostly b/c of improved time resolution of TP
    - E.g. less likely to get ALCT timing off by a couple of early hits from PU (or neutrons)
  - Decreased pre-trigger rate leads to more dead-times reduction

LCT match inefficiency

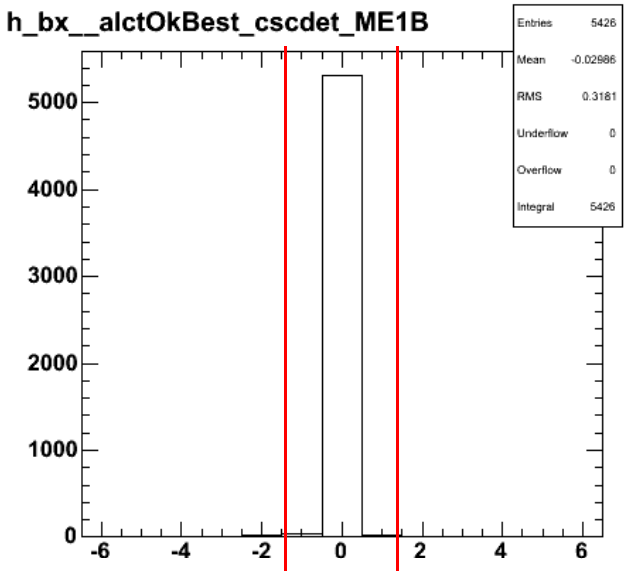
- Why on the previous page there was inefficiency between finding correct A&CLCTs and correct LCT?

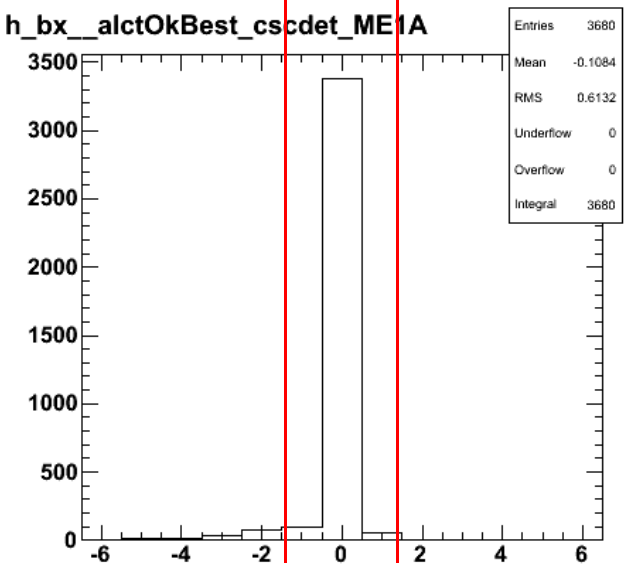


  - When a correct CLCT in TMB waits for a match, it only picks up **the first** ALCT in 3BX match window
    - If correct ALCT happens later in match window, it would not be picked up
  - **Plot shows efficiency increase ~2%-3% when all ALCTs in match window are picked up**
    - **Our current "high efficiency" plot!**
    - Implementation of such algorithm in hardware might be non-trivial
      - More than 2 LCTs/BX would have to be sent

ALCT Time Resolution in ME11

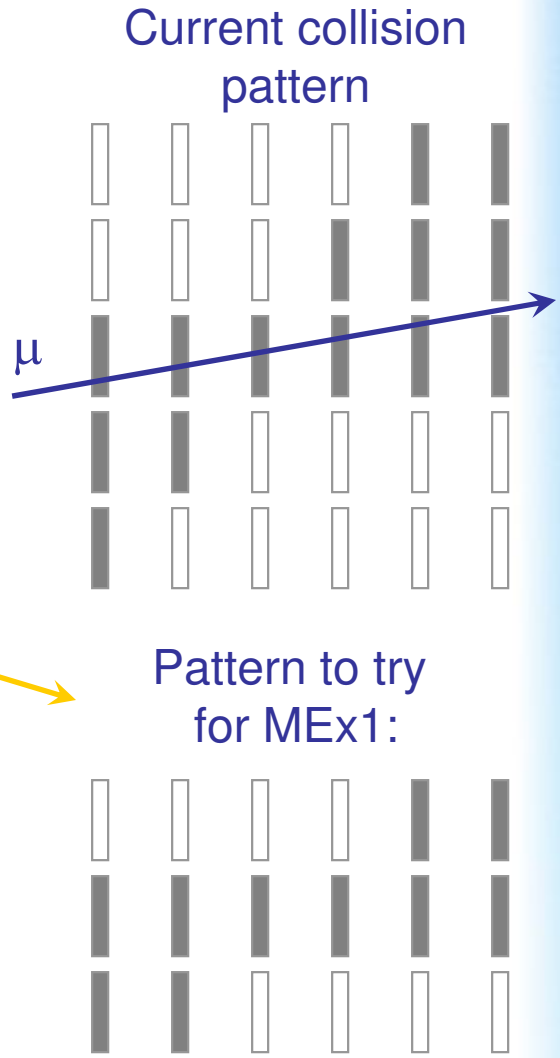
- Another reason for LCT match inefficiency is deterioration of ALCT timing in ME1a
  - ALCTs out of 3BX r/o window:
    - **ME1/b: 1.0%**
    - **ME1/a: 4.1%**
      - About ½ of it comes from the first two wire groups
  - For comparison: CLCT out of r/o window
    - ME1/b: 0.9%
    - ME1/a: 2.1%
    - Opposite of low PU case, when ALCTs have much better time resolution than CLCTs
  - **ALCTs time resolution really needs improving in ME1/a**





Plans for Improving ALCT Timing

- ME1/a: the highest occupancy and only ~13 WGs
  - Signal muon is very likely to have some earlier soft PU tracks nearby
  - Current collision pattern
    - the same for all chamber types
    - is **very wide** for ME1/a
    - Timing and spatial resolution can be easily affected by neighboring soft hits
- Plan:
  - **Use narrower pattern** for R=1 chambers
    - It should be wide enough in order to keep efficiency
  - Instead of defining ALCT time by its pre-trigger time, **calculate median time**
    - There still are some limited FPGA resources available (from Alex Madorsky)
    - Median is more robust to outliers than mean



MAX # LCT per ME11

- In previous plots we allowed max 2+2 LCTs per ME1/1 per BX (from ME1/a+ME1/b)
- If we are allowed max 2 LCTs per whole ME11
  - Correct LCT efficiency drops just by ~0.4% in ME1/a only
    - b/c my current algorithm prefers LCTs from ME1/b
  - That's a rather small number to bother

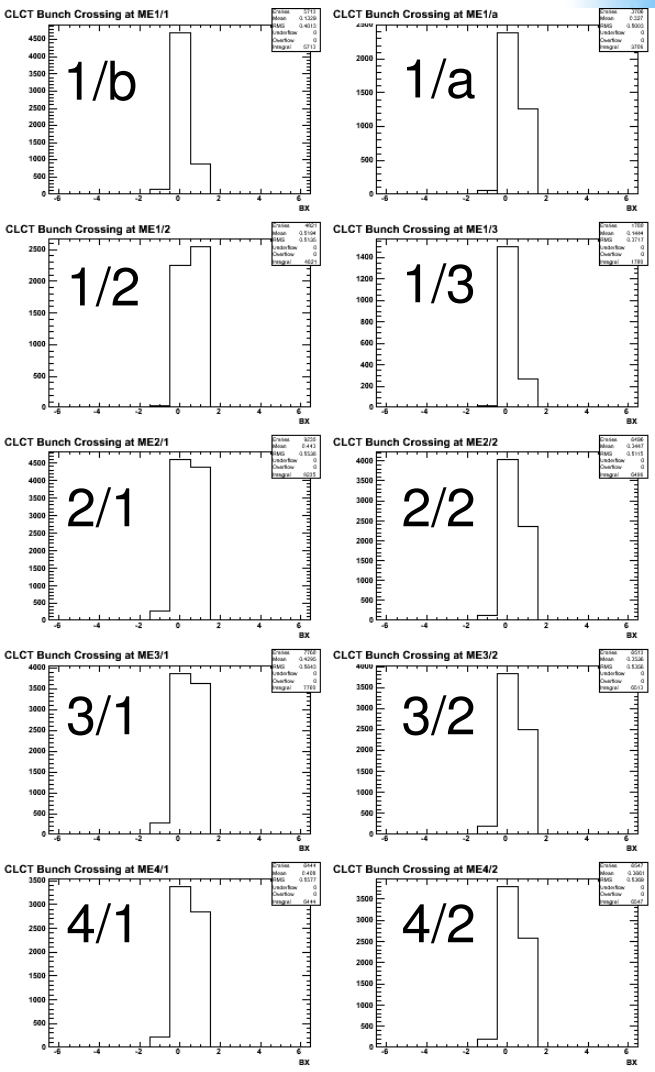
- An important case when more than 2 LCTs/BX would be needed:
  - CLCT can match not only the first ALCT in match window
    - Worth exploring hardware possibilities, as it brings considerable efficiency improvement
    - Still need to estimate probability of > 2 LCTs/BX in this case

A/CLCT Time Synchronization

- In default simulation CLCT timing is not synchronized enough for different chamber types

- Need for good time synchronization:
  - To start any work on CLCT time resolution improvement
  - To effectively match ALCT to CLCT
  - We expect that at SLHC detector will be well synchronized

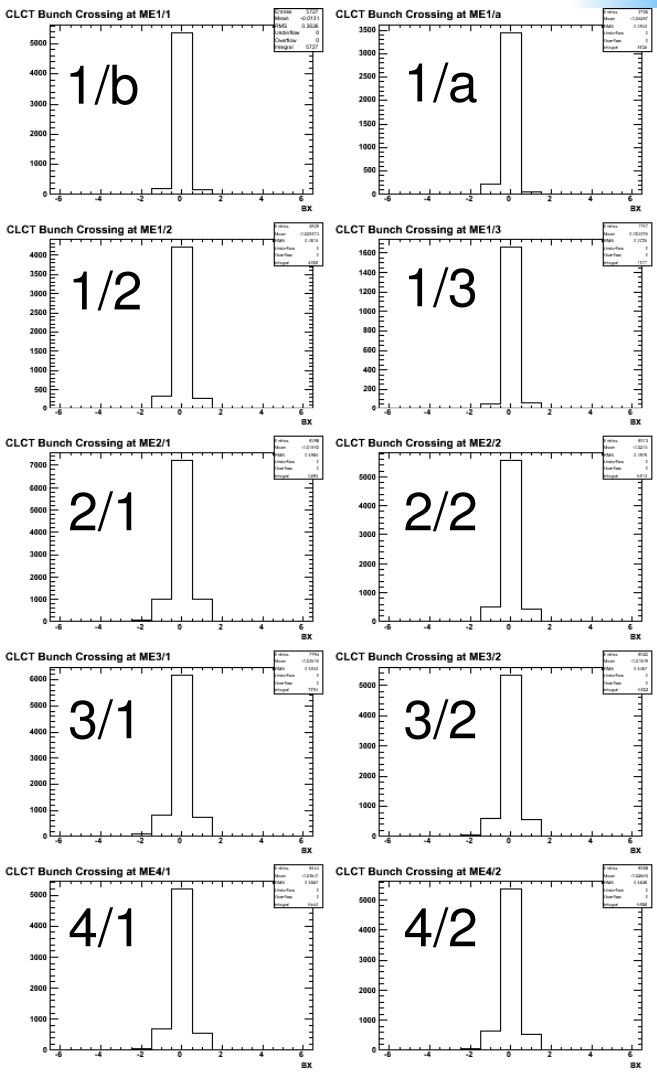
PU0 sample:
CLCT BX



A/CLCT Time Tuning in Simulation

- CSCDigitizer:
  - Has “offset” parameters to tune timing
  - Used 0PU sample to do tuning
- **All plots in this talk were made with synchronized samples**
  - Non-synchronized sample shows ~1% smaller efficiency in LCT matching
- Note: any change in how time of TP is defined requires separate tuning
  - E.g., when we use different pre-trigger requirements

CLCT BX After synchronization



Planned Work on TPs

- Improvement of time resolution in ME11
  - More resolution and pattern studies for ALCTs
  - Better CLCT time resolution will also be beneficial
- Realistic LCT finding algorithm that would allow picking not only the earliest ALCT in matching window
  - Need accurate estimates and consideration of hardware constraints
- Looking into the possibility of simulating several consecutive ALCTs on the same WG
  - Currently, only the earliest one is simulated
- Detailed trigger rate studies
- max # MPCs optimization for increased number of LCTs
- Adding neutron BG
  - Rick's simulation will be included into 3_4_X
  - MixingModule is not quite ready for an additional neutron input yet
  - A lot of things will need to be re-considered and re-optimized with neutrons

Conclusions

- Using custom full simulation framework we did detailed studies of sources of CSC trigger primitives inefficiency in high PU
- Proposed solutions to recover the efficiency and to suppress the rate
- Current estimates for possible stub finding efficiency in PU400 are
  - ME1/b: ~95%
  - ME1/a: ~90%
  - Will probably go down when neutron BG will be added
  - There is still room for improvement
    - Timing resolution studies are ongoing

- Thanks to Rick Wilkinson and Alex Madorsky for answering our questions about CSC simulation and ALCT finding

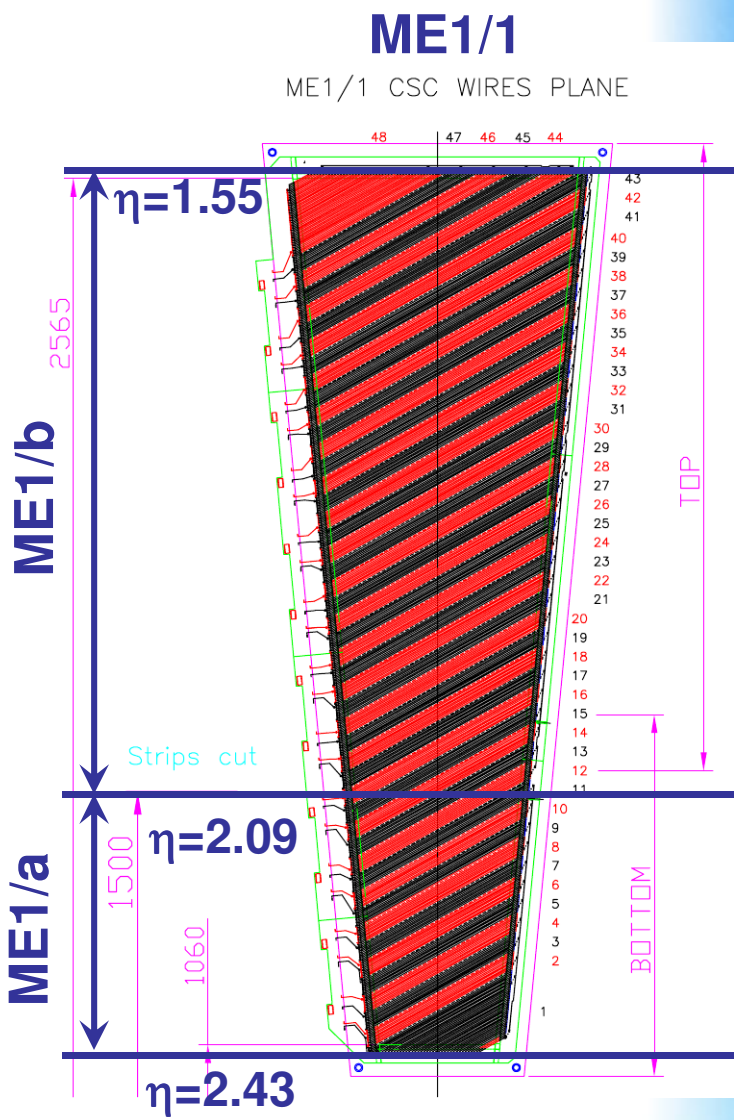
BACKUP

Restoring triggering in ME1/a

- Details…

ME1/1 Reminder

- **Crucial for Track Finder PT resolution!**
- ME1/1 = ME1/a + ME1/b
- Continuous wire groups coverage
  - WG #11-16 cross the border
- Separate sets of strips in 1/b and 1/a
  - ME1/a strips ganged into sets of 3
    - **is not useful for triggering**
- **Most affected at high luminosities**
  - The highest occupancies (esp.ME1/a)
  - **Significant efficiency drop**



ME1/a & 1/b Separation in Emulator

- We implemented two separation algorithms:
  - "Naïve" ME1/a and ME1/b separation:
    - Two separate CLCT & Two separate ALCT finders
      - Use digi MC info (some cheating)
    - Not a realistic algorithm
    - But can be used for various special studies and cross-checks
  - "Smart" separation:
    - Two separate CLCT finders for 1/a and 1/b
    - One ALCT finder for whole ME11
    - Use intelligent guess on whether LCT belongs to 1/a or 1/b
    - Implemented in the emulator as CSCMotherboardME11 class (inherits from CSCMotherboard)
    - **Need to make sure that algorithm can be implemented in hardware**
  - First working version of code is committed to CVS

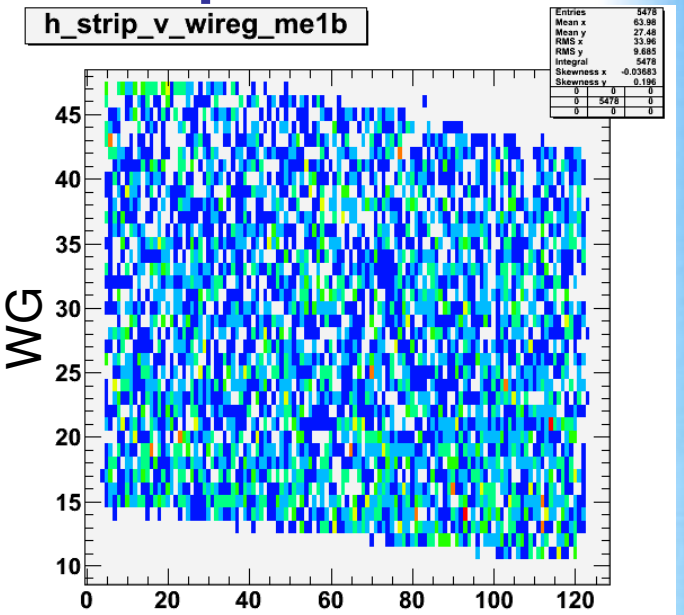
```
cvs co -r slhc_branch_2_2_X L1Trigger/CSCTriggerPrimitives
```

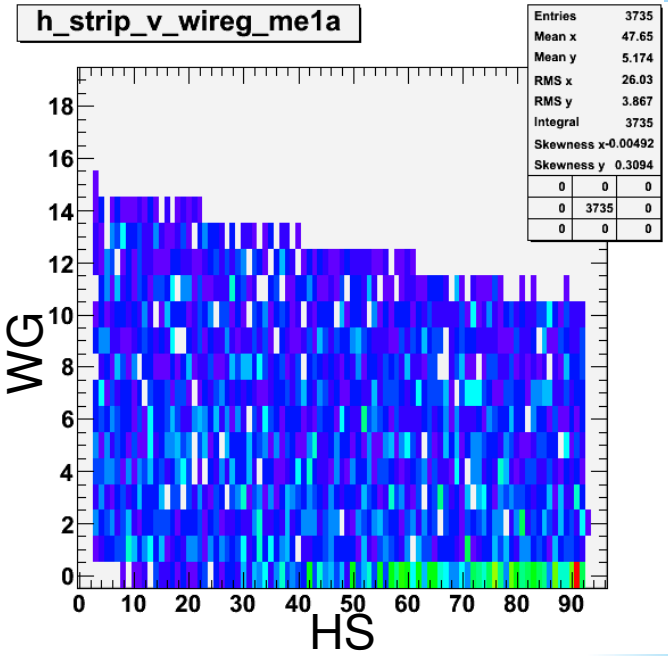
“Smart” Separation in a Nutshell

- Find ALCTs and CLCTs in ME1/a and ME1/b
  - ALCTs need special treatment
- Match ALCTs to CLCTs separately in ME1/a and ME1/b
  - Properly handle possible case of ALCT and CLCT being not able to physically cross
  - Choose the best two LCTs using special sorting algorithm
- If we are allowed max 2 per whole ME1/1
  - One more sorting step is necessary

”Smart” ME1/a & ME1/b Separation

- LCT = ALCT & CLCT match
- CLCTs:
  - unambiguous separation
- ALCTs:
  - angled at 29^{o}
  - some of them cross 1/a & 1/b border
  - Need to define how we assign an ALCT to 1/a or 1/b
- Some half-strips and some wire groups in ME1/1 don’t ever cross
  - Plots: key HS (x) vs. key WG (y) for LCTs matched to SimTrack
    - Create look-up tables
    - Note: resolution dependency





LCT Matching in ME1/a or 1/b

- **Separate** for each ME1/a and ME1/b
- Matcher input:
  - max 2 best ALCTs + max 2 best CLCTs + whether four possible combinations of ALCT+CLCT may cross
  - Use LUTs to answer if a certain ALCT+CLCT combination may physically cross
    - Two LUTs: for ME1/a & for ME1/b
    - Each LUT is 48 x 2:
      - 48 WG numbers vs. inclusive range of half-strip numbers
- Matcher output:
  - max 2 best LCTs
  - The additional “crossing” conditions make sorting algorithm to choose the best LCTs more complex…

LCT Sorting Algorithm for ME1/a or b

- Input
  - bestALCT, secondALCT & bestCLCT, secondCLCT
  - whether four possible combinations of them may cross
- Form “crossing” conditions code:
  - Bit 4: bestALCT crosses bestCLCT?
    Bit 3: bestALCT crosses secondCLCT?
    Bit 2: secondALCT crosses bestCLCT?
    Bit 1: second ALCT crosses secondCLCT?
- Use 16 x 2 LUT for choosing max 2 best LCTs
  - 16 rows correspond to possible values of “crossing” conditions code
  - 2 values in row define which ALCT+CLCT combination to use to construct **output: bestLCT and secondLCT**

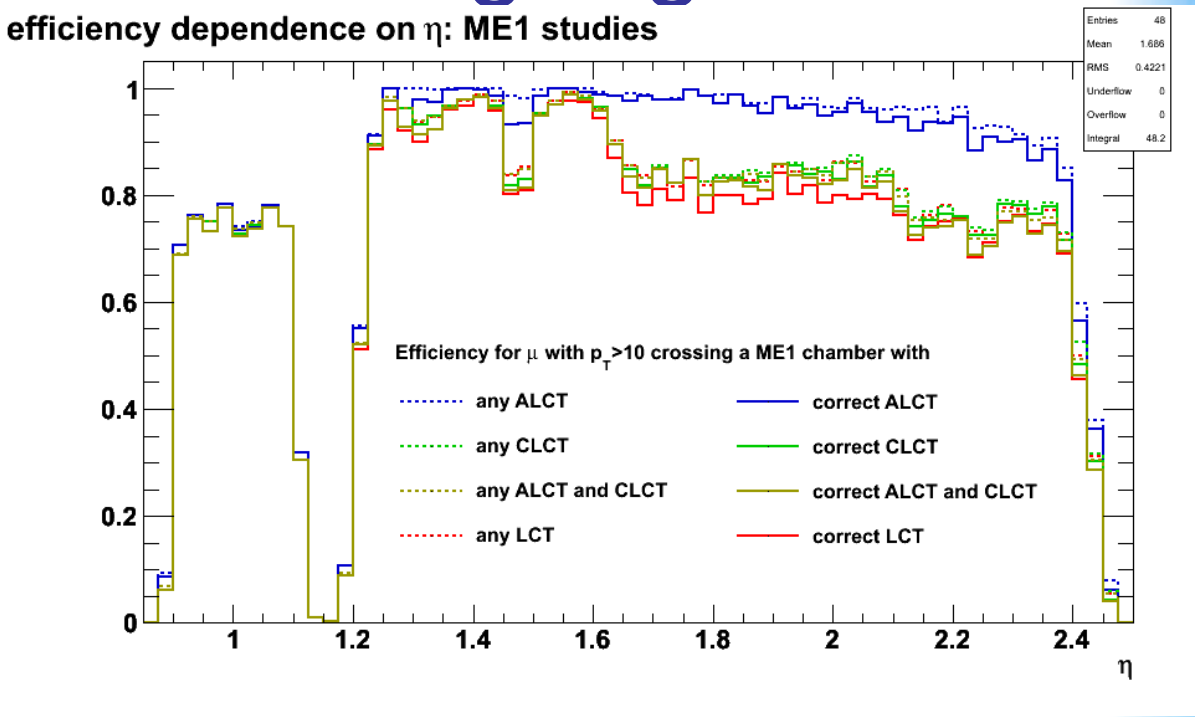
LCT Sorting in Whole ME1/1

- Things are simple if we are allowed max 2+2 LCTs per ME1/a+ME1/b

- **If we are allowed max 2 LCTs per ME11, one more sorting is needed**
  - Choose best max 2 LCTs out of (max 2 from ME1/a) + (max 2 from ME1/b)
    - Keeping it simple: prefer higher eta LCTs from ME1/b
    - May any other options be needed?

Modified Emul.: Add Unganged ME1a

- 'Naive' ME1a addition:
  - Separate ALCT & CLCT finders for ME1a and ME1b



Notes on Hit Persistency Length

- Current default hit persistency interval is 6BX
- Effects of changing it to 4BX still need more studying, but the current understanding is
  - Some ~1% efficiency increase is observed only before dead-time reduction steps
  - When most of dead-time sources are taken care of, efficiency shows either very little change or a small drop
    - Small drop may happen b/c of higher probability to have less layers in time than is required for trigger threshold
  - Studies for ALCTs are problematic b/c currently only the earliest ALCT can be reconstructed on a wire group