DT trigger primitives
Joint P2 Muon Upgrade + P2 L1 Muon Algorithms workshop

Camilo Carrillo

on behalf of the DT trigger primitives community

Centro de Investigaciones Energéticas, Medioambientales y Tecnológicas

29/11/18

Table of contents

Introduction

- The **DT phase 2 upgrade** will consist in substituting the on-detector and back-end electronics because the Level 1 trigger rate (500 kHz) cannot be sustained.
- Taking profit of this, we decided to build a **new trigger** system that will be **located in the back-end FPGAs** instead of on the on detector electronics ASICs.
- The new system will receive the full resolution time measurements from the chambers without any filtering. A priori, **ultimate HLT resolution could be achieved in the Level 1 trigger**.
- Building TP (Trigger Primitives) in **FPGAs** instead of in ASICS **allows us to be more flexible on the trigger primitive generation** and we are exploring different alternatives to optimize the TP production.
- The algorithm developments should improve the robustness, regarding longevity of the DT chambers.

Introduction

- For the moment we are working with two different approaches:
  - One algorithm that is based on the determination of the BX by a meantimer histogramed and then a linear fit by a **compact hough transform**
  - One algorithm that is based on a pattern classifier plus a BX determination by an **analytical method**
- The status and plans for both are being presented in these slides.
- All **developments in CMSSW are R& D oriented and firmware compatible**.

DTHits in phase-2

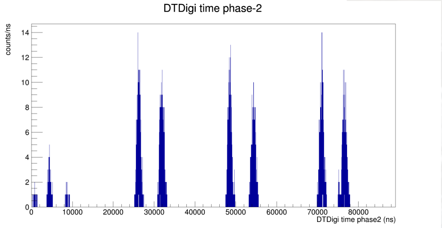
- Independently of the algorithm we will use for L1 Trigger, phase-2 DT hits will come from a TDC with approximately 1 *ns* resolution **referenced to the** $BC0$ **of the orbit** i.e:
  TDC hit = BX of the collision(TDC units) + TOF + delays + drift time
- In order to emulate these hits using phase-1 data we transform the present DT Digis in CMSSW as:

```
int wire = (*digiIt).wire();

int digiTDC = (*digiIt).countsTDC();
int digiTDCPhase2 =  (*digiIt).countsTDC()+ iEvent.eventAuxiliary().bunchCrossing()*32;

int digiTIME = (*digiIt).time();
int digiTIMEPhase2 =  (*digiIt).time()+ iEvent.eventAuxiliary().bunchCrossing()*25;

int layer = dtLId.layer();
int superlayer = dtLId.superlayer();
```



- This is the input for the following studies.

DT TPG based on **hough transform**

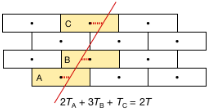
DT TPG based on **hough transform**

Overview of the MMT+CHT algorithm

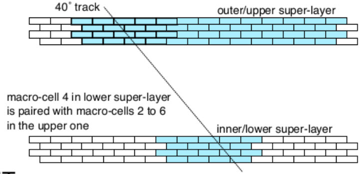
- **MMT**:Majority Mean Timer, **CHT**:Compact Hough Transform
- R& D started in Padova 4 years ago
- Two-step reconstruction of track segments at chamber level:
  - Custom implementation of Hough Transform (Compact HT); whole chamber at once, no correlation of two super-layers is needed.
  - Histogram-based BX identification exploiting the mean-timer equations for each pattern of wires and laterality hypothesis (Majority MT)
- Concept and several steps of evaluation documented available in CMS indico and/or journal articles

Overview of the MMT+CHT algorithm

- macro-cells of 18 wires (14 in new version) are processed at a time
- statistical revisitation of the Mean Timer equations for the identification of muon crossing time



  - guessed values of BX time fill a histogram and the most counted in a macro-cell is chosen, priority to 4-layers patterns
- dedicated implementation of Hough Transform working on pairs of DT anode wires at a time and processing in parallel clusters of pairs from each super-layer and pairs with one wire from each super-layer



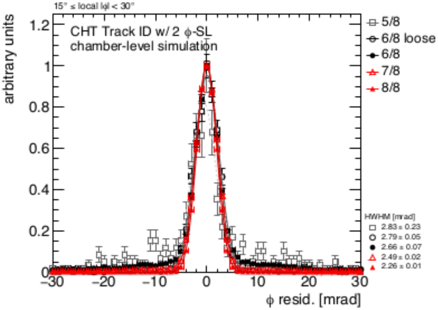
  - sampled variable: track slope
  - common subset of track slope range are used to define the candidate segment
  - left/right ambiguities naturally solved by HT
  - accounts for additional qualities such as 4+2 and 2+4

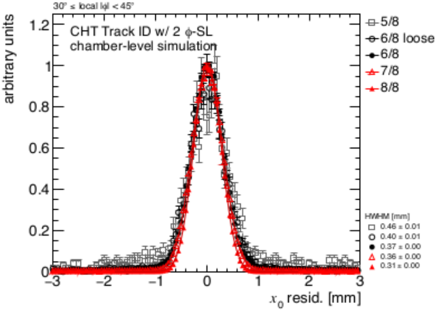
Overview of the MMT+CHT simulation software R& D oriented

- **studies started since early times with a private CMSSW producer** (let's call it DTHough for the time being): direct access to edm products, in particular DTDigis and MC-truth related collections for evaluation purposes
  - **these trigger primitives are built from DTDigis**
- some "creative" manipulation of standard DTDigis were needed at DTDigiProducer level to allow for a **"test-beam-like" simulation of muons shot right in front of a chamber**
  - **comparison with actual data from test beam of BTI and TRACO for efficiency and resolution comparison in terms of local coordinates over a wide angle range**
  - the DTHough producer could switch between the two
- next "creative" complication: random parent BX
- another "creative" manipulation was needed to f**orce CMSSW have memory of previous events in order to simulate a continuous flow of Poisson-distributed muons with realistic rate**
  - this was needed to span the whole orbit duration and allow for overlapping muons from different BX, **to estimate the input occupancy, to prepare test vectors for the hw demonstrator etc.**

Status of the MMT+CHT algorithm in CMSSW, resolution

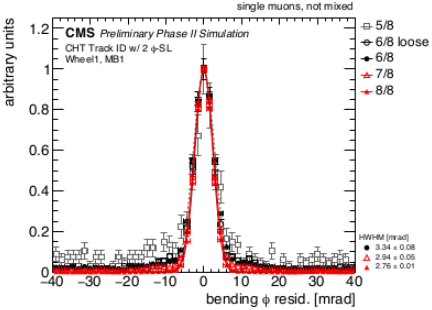
- resolution of local track parameters is compatible or better than BTI-TRACO – example shown for segments built using hits from both φ-SL
  - current TRACO resolution: ~3 mrad / ~1.4 mm
  - **our expected resolution: ~2.5 mrad / ~0.3 mm**

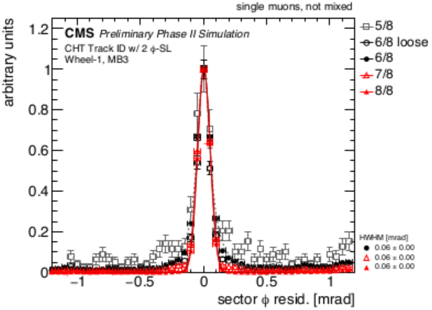




Status of the MMT+CHT algorithm in CMSSW, resolution

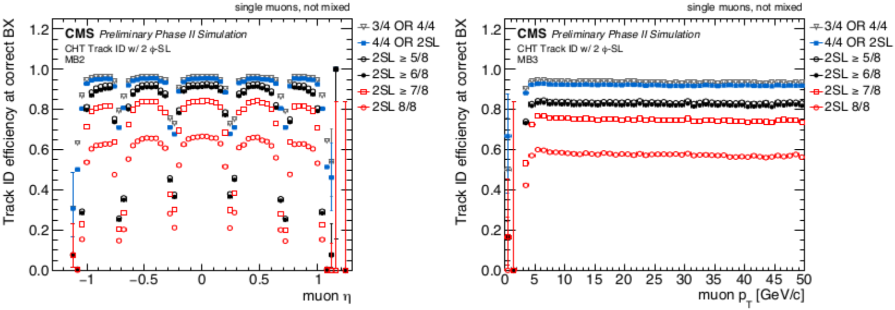
- these are examples of the resolution of bending and sector angles, measured w.r.t. the direction and coordinates of the MC-truth track in the chamber
  - **need to account for 15-16 bits for sector φ (instead of 12)**, accounting for tails which make its range larger than the nominal π/6 of a sector
  - **need to account 12-13 bits for bending φ (instead of 9)**, spanning up to ±60°





Status of the MMT+CHT algorithm in CMSSW, efficiency

- this is an example of the efficiency we have in two stations (integrated over wheels and sectors) with the "CMSSW-compliant" producer



DT TPG based on the analytical method

DT TPG based on the **analytical method**

Analytical method algebra

Basic equations (what do we want to do?)

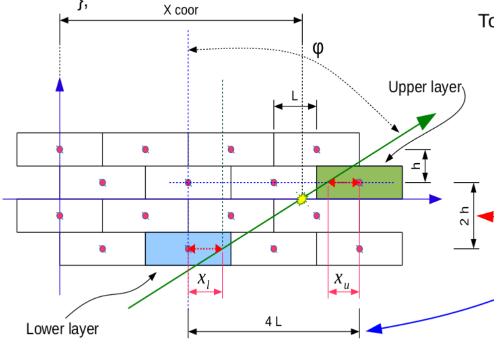
DT Hit (C-like syntax)

```
struct DT_HIT {
    uint_32 layer;
    uint_32 cell_in_layer;
    uint_32
    TDC_time_since_BX0;
};
```

Convert 3 or 4 of THIS into THAT

$$\begin{Bmatrix} tg(\varphi) \\ X_{coor} \end{Bmatrix}$$

To do it, we need THESE equations



$$L_{low} L_{up} \rightarrow H_{ul} \cdot L - x_u + x_l = V_{ul} \cdot h \cdot tg(\varphi)$$

$$R_{low} R_{up} \rightarrow H_{ul} \cdot L + x_u - x_l = V_{ul} \cdot h \cdot tg(\varphi)$$

$$L_{low} R_{up} \rightarrow H_{ul} \cdot L + x_u + x_l = V_{ul} \cdot h \cdot tg(\varphi)$$

$$R_{low} L_{up} \rightarrow H_{ul} \cdot L - x_u - x_l = V_{ul} \cdot h \cdot tg(\varphi)$$

They only differ in a few signs

Analytical method: generalized Meantimer equations

$\tau \equiv TDC_{time}$

$Bx_t \equiv$ Bx time since BX#0

$x_{cell} = v_{drift} \cdot t_{drift}$

$L = v_{drift} \cdot T_{max\ drift}$

$$V_{um} \cdot (H_{ml} \cdot L - x_m + x_l) - V_{ml} \cdot (H_{um} \cdot L - x_u + x_m) = 0$$

Mixing everything

$$V_{um} \cdot (H_{ml} \cdot T - (\tau_m - Bx_t) + (\tau_l - Bx_t)) - V_{ml} \cdot (H_{um} \cdot T - (\tau_u - Bx_t) + (\tau_m - Bx_t)) = 0$$

We can generalize the equation, using layer-dependant laterality coeficients (a, b, c, d)

$$V_{um} \cdot (H_{ml} \cdot T + a\tau_m + b\tau_l) - V_{ml} \cdot (H_{um} \cdot T + c\tau_u + d\tau_m) - (V_{um} \cdot (a+b) - V_{ml} \cdot (c+d)) \cdot Bx_t = 0$$

Laterality coeficients

Layer *l*	*m*	*u*		a	b	c	d
L	L	L	→	1	-1	1	-1
L	L	R	→	1	-1	1	1
…							
R	R	L	→	-1	1	-1	-1
R	R	R	→	-1	1	-1	1

We can get a BX, if **denominator** is **not NULL**

$$Bx_t = \frac{V_{um} \cdot (H_{ml} \cdot T + a\tau_m + b\tau_l) - V_{ml} \cdot (H_{um} \cdot T + c\tau_u + d\tau_m)}{V_{um} \cdot (a+b) - V_{ml} \cdot (c+d)}$$

Condition to be accepted as a valid BX: It must have physical sense

- 'Bx_t' must be positive, and greater or equal than every 'TDC time' value used in its calculation.
- Also each drift time has to be smaller than the maximum cell's drift time.

Only when both conditions are satisfied, the group of hits form a straight path → It is a muon candidate

a, b, c, d are constants that depend on the assumed laterality.

Once the BX is computed, x and $\tan(\phi)$ are computed to generate the TP

Offline-like equations

These equations are applied after getting the associated BX

For a valid 4-hits group

$$X_{chamber} = \frac{x_3 + x_2 + x_1 + x_0}{4}$$

$$\tan(\varphi) = \frac{3 \cdot x_3 + x_2 - x_1 - 3 \cdot x_0}{10 \cdot h}$$

$$\chi^2 = \sum_{i=0}^{3} \left(x_i - \tan(\varphi) \cdot y_{wire_i} - b\right)$$

$$b = \frac{1}{4} \sum_{i=0}^{3} x_i$$

x_i → horizontal hit's coordinate in the chamber for layer 'i'
h → cell's height
y_{wire} → cell's wire vertical coordinate

For a valid 3-hits group

$$X_{chamber} = \frac{x_i + x_j}{2}$$

$$\tan(\varphi) = \frac{\frac{1}{2} L \cdot H_{i,j} + v_{drift} \cdot (a \cdot \tau_{drift\,i} + b \cdot \tau_{drift\,j})}{V_{i,j} \cdot h}$$

i, j = 0, 3 when there are valid hits for layers 0, 3
i, j = 1, 2 otherwise

x_i → horizontal hit's coordinate in the chamber for layer 'i'
h → cell's height; L → cell's length
a, b → laterality coeficients
τ_i → signal drift time on 'i' cell
$V_{i,j}$ → vertical distance between cell's wires on layers 'i' and 'j' (in 'h' units)
$H_{i,j}$ → horizontal distance between cell's wires on layers 'i' and 'j' (in semilength units)

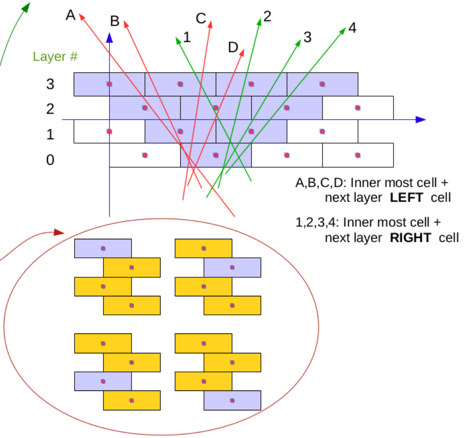
Pattern-classifier for the analytical method

- We have to resolve previous equations for every group of 4 cells in different layers, using their TDC times.
- The whole 4-cell combinational space can't be managed and, moreover, not all 4-cell combinations are good candidates.

  We can preselect and discard many combinations by geometrical criteria

- The algorithm operates only over 10 cells at a time, building reasonable 4-cell groups.
- For each 4-cell group, equations are applied up to 4 times, over subgroups of 3 cells.
- For every 3-cell subgroup, the resulting BX time must be equal to their complementary partners, within a 'Δ' margin (now ~ 30 ns).

Possible particle paths, compatible with a stright line, flowing from one given inner-most super-layer cell



Analytical Method algorithm

- A custom C/C++ program designed with an **architecture and methods firmware oriented** has been developed for a Mean Timer like algorithm (analytical method).
- Several preliminary results independent from CMSSW with n-tuple like analysis have been presented about this algorithm. `https://indico.cern.ch/event/713719/contributions/2932502/attachments/1616887/2570507/Luigi_DTUpgradeWorkshop20180314.pdf`
- The algorithm has been validated in various stand alone programs showing good performance
- The porting of all these capabilities to an L1-Phase 2 emulator in CMSSW. Tested with CMS data and extended to all chambers.

Porting the Analytical Method to CMSSW, check-list:

- ☑ build a CMSSW wrapper for any algorithm starting from DTDigis → phase-1 DTTrigger primitives
- ☑ port firmware-like-algorithm from C/C++ to cmssw wrapper
  - ☑ develop phase-2 DTDigis producer.
  - ☑ transmit phase-2 DTDigi information as input to firmware-like-algorithm
  - ☐ **debugging:**
    - ☑ **resolution measurements**
    - ☑ **extrapolate 4D-seg to corresponding SL**
  - ☑ **uniform coordinates and code-conventions**
  - ☐ code memory/cpu optimization (for the moment not optimal at all)
  - ☑ **comparison with HLT segments**
    - ☑ **time correlation and resolution**
    - ☑ **tan(ϕ) correlation and resolution**
    - ☑ **position correlation and resolution**
  - ☐ comparison with offline segments
  - ☐ TnP algorithm from DPGO, DT n-tuples
- ☐ change data-format for DTPrimitives to phase-2 format (handshake with kalman filter)
- ☐ implementation/testing of new algorithms, other superlayers, RPCs

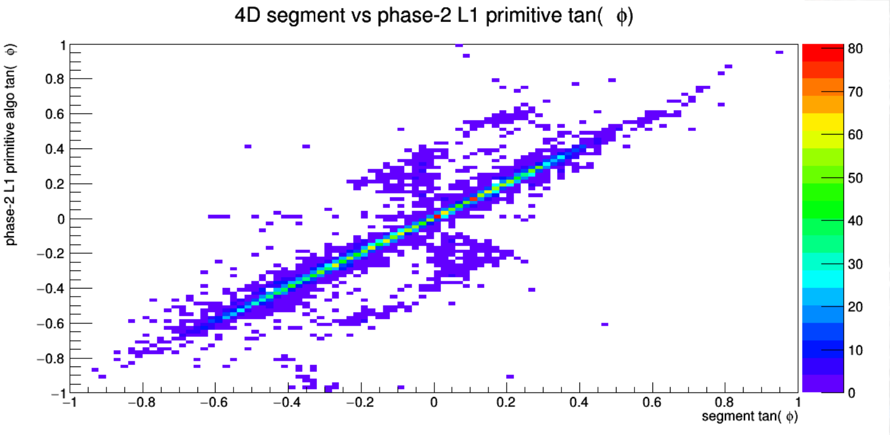
The code & dataset

- Development in *CMSSW 10 1 1*,
- The data used was from the 2017A run 295655 (LHC trains of bunches are more spaced).
- We use RPCMonitorStream data which is simply an enriched low-level muon information dataset available at **HLT**:
  - DT+RPC+CSC digis
  - local reco@**HLT** products:
    (local RecHits, 1DRecHits, 2D and 4D segments)
  - L1 and **HLT** info.
- Could be run over any other dataset.
- The firmware-like-algorithm written in C/C++ from the one results have been already shown are uploaded here for reference:

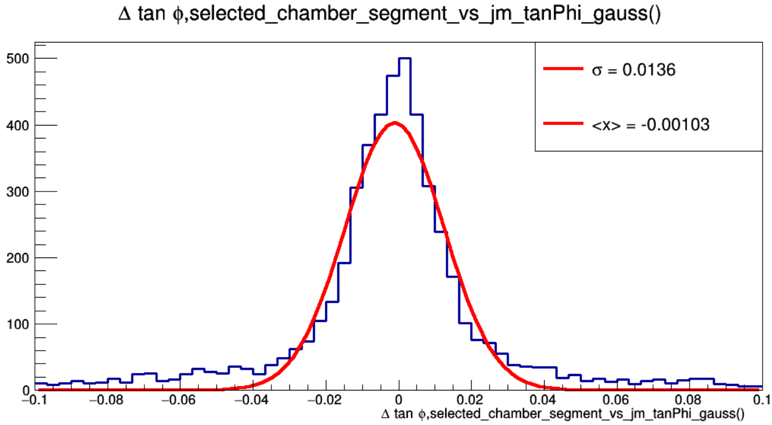
  `https://github.com/camilocarrillo/DTTrigger/tree/master/doc/referece_code`
- Out of the box instructions to test the algorithm:

  `https://github.com/camilocarrillo/DTTrigger/`

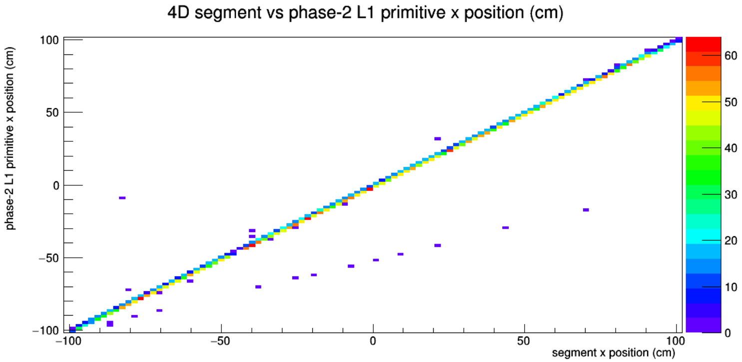
$tan(\phi)$, comparison with HLT segments



difference segment - firmware-meantimer algorithm, $tan(\phi)$

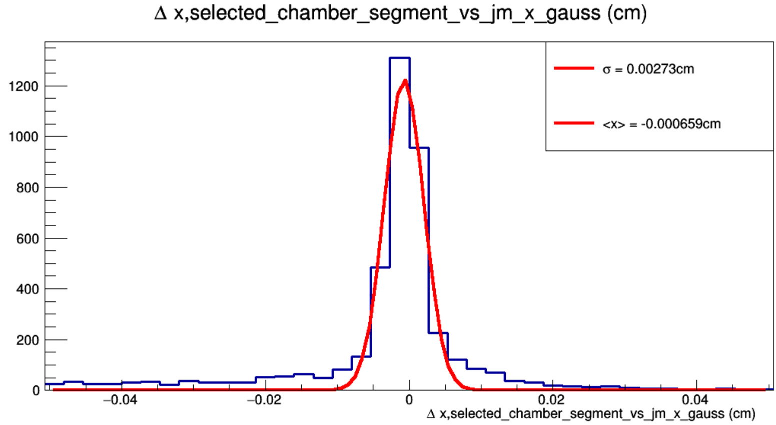


Position, comparison with HLT segments

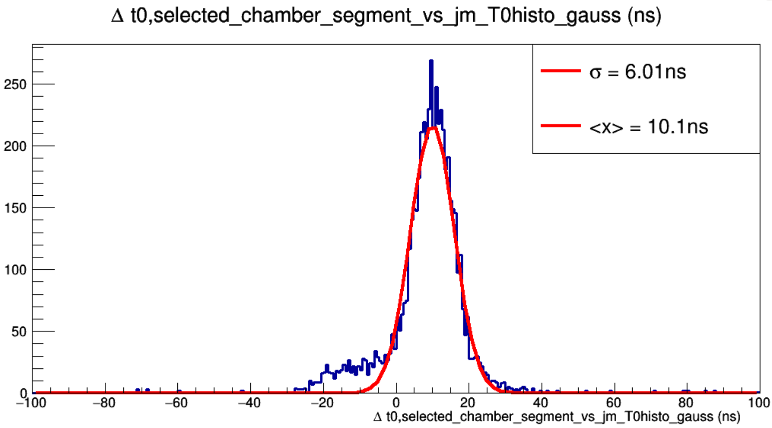



```
//using the loop to fill up map of shifts with the position of the wire 1 of the layer 2
DTWireId wireId1(segment1->chamberId(),1,2,1);
int firstWire1 = dtGeo->layer(wireId1)->specificTopology().firstChannel();
if(my_debug) std::cout<<"Dtp2: from geometry I got for SL=1 "<<wireId1<<" "  <<dtGeo->layer(
shift_jm_cmssw[wireId1]=dtGeo->layer(wireId1)->specificTopology().wirePosition(firstWire1);
```

Difference HLT segment - analytical method position



difference HLT segment - analytical method, t0



algorithm output, phase-1 primitives

```
outPhi.push_back(L1MuDTChambPhDigi(step,
                              (*it).ChamberId().wheel(),
                              sc_sector,
                              (*it).ChamberId().station(),
                              (*it).phi(),
                              (*it).phiB(),
                              (*it).code(),
                              !(*it).isFirst(),
                              0
                              ));
```

- An analogous primitive is defined for the η projection, but for the moment is filled up empty.

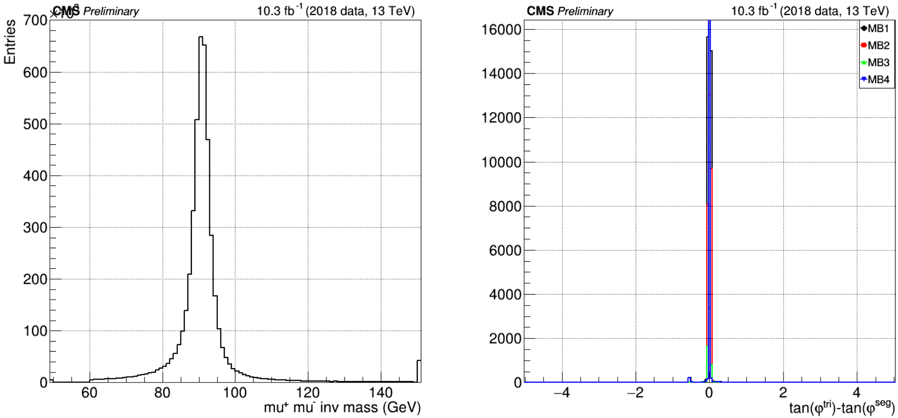
algorithm output, phase-2 primitive proposal

- The *proposed* new dataformat for phase-2 primitives. New information and station granularity.
- The η projection is included in one single instance of the format (with the z information and $z - slope$)
- The existing code to measure performance would have to be adapted as well for this new dataformat

```
// Operations
int bxNum()        const;
int whNum()        const;
int scNum()        const;
int stNum()        const;
int phi()          const;
int phiBend()      const;
int z()            const;
int zSlope()       const;
int quality()      const;
int index()        const;
int t0()           const;
int chi2()         const;
int rpcFlag()       const;

private:
 int m_bx;
 int m_wheel;
 int m_sector;
 int m_station;
 int m_phiAngle;
 int m_phiBending;
 int m_zCoordinate;
 int m_zSlope;
 int m_qualityCode;
 int m_segmentIndex;
 int m_t0Segment;
 int m_chi2Segment;
 int m_rpcFlag;
```

Preliminar results with TnP method of the generated primitives

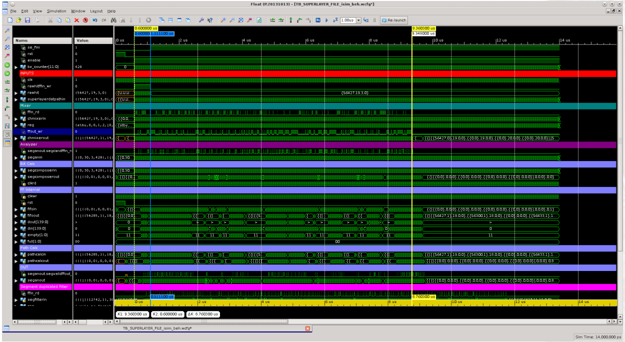


From MuonPOG: `http://www.hep.uniovi.es/vrbouza/Proyectos/trigger_primitives/results/TMuxOut/phi/`.

Firmware implementation for the analytical method

FPGA implementation for an algorithm of 2 ϕ Superlayers (not yet one chamber) in a Virtex 7 FPGA (XC7VX330t-2ffg1157), clock frequency 240 MHz.

- Slices 30%
- LUTs 21%
- DSP 30%
- FPGA not yet fully used!



Test-bench implemented with real data and simulations on going.

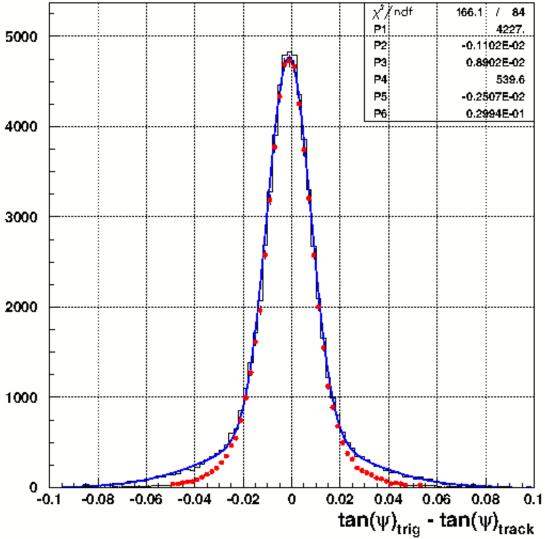
Plans and next steps / conclusions

- **Algorithm:** The goal is to evaluate the performance of both approaches in similar conditions and try to either merge or select the parts that are more appropriate. New algorithms and extensions will be also tested with all the developed tools.
- **Emulator:** The algos developed are being ported to CMSSW. The measured resolutions by comparing with HLT 4D segments, TnP and MC truth is satisfactory. Few debugging issues still pending.
  We aim for a tested/firmware oriented emulator pull request to CMSSW version by **beginning of next year** that we can validate in different sample types.
- **Firmware:** first version will be ready in the next months and will be tested in the SX5 test stand with a DT chamber. Testing and debugging will continue until the CMS Slice test in June.

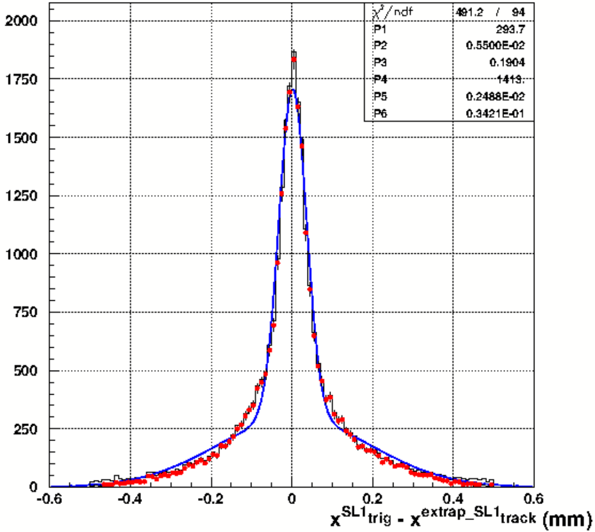
Backup

Backup

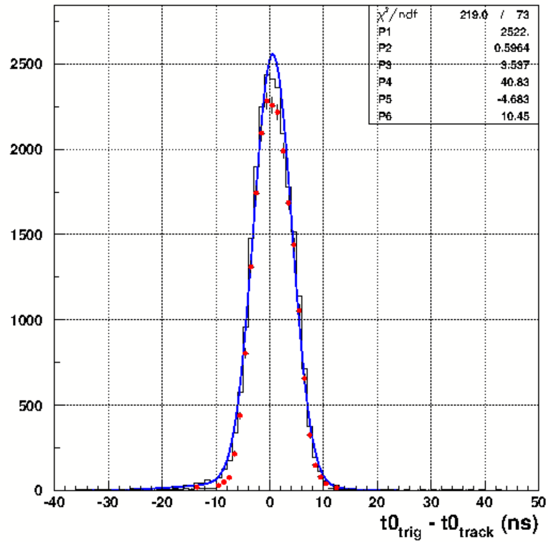
tanPhi, n-tuple study analytical method



x, n-tuple study analytical method



t0, n-tuple study analytical method



log arranged in order to facilitate debugging

- **spanish:** firmware-meantimer-algorithm
- **english:** cmssw wrapper and porting

```
lx-014
DTp2: we found a perfect segment in ciemat's chamber
DTp2: segment_x:34.1778cm segment_tanPhi:-0.339109
DTp2:    we have (DTDigis(cmssw)):  layer1:1 layer2:1 layer3:1 layer4:1
DTp2:    (savedTime): layer1:26073 layer2:26264 layer3:26237 layer4:26093
DTp2:    (savedWire(cmssw)): layer1:33 layer2:33 layer3:33 layer4:33
DTp2:            we found one hit per layer
DTp2:            Capa: 0 Canal: 33 TDCTime: 26073
DTp2:            Capa: 1 Canal: 33 TDCTime: 26264
DTp2:            Capa: 2 Canal: 33 TDCTime: 26237
DTp2:            Capa: 3 Canal: 33 TDCTime: 26093
DTp2:                    input: MuonPath' analyzable, TDC Phase2 Times are: 26073 26264 26237 26093
DTp2:                    pathId: In function compute_pathId, computing_pathId for wires: 33 33 33 33
DTp2:                    pathId: built array 0 -1 0 -1
DTp2:                     pathId: computing_pathId returning: 2
DTp2:                            In analyze! checking if mPath->isAnalyzable() 1
DTp2:                                    En evaluatePathQuality Evaluando PathQ. Celda base: 33
DTp2:                                    Total lateralidades: 4
DTp2:                                    Analizando combinacion de lateralidad: 0 1 0 0
DTp2:                                    In validate Iniciando validacion de MuonPath para capas: 0/1/2
DTp2:                                    Lateralidades parciales: 0/1/0
DTp2:                                    Valores de TDC: 26073/26264/26237.
DTp2:                                    Valid TIMES: 1/1/1.
DTp2:                                    In eqMainBXTerm EQTerm(BX): -52501
DTp2:                                    In eqMainBXTerm EQTerm(BX): 52337
DTp2:                                    In validate Iniciando validacion de MuonPath para capas: 1/2/3
DTp2:                                    Lateralidades parciales: 1/0/0
DTp2:                                    Valores de TDC: 26264/26237/26093.
DTp2:                                    Valid TIMES: 1/1/1.
DTp2:                                    In eqMainBXTerm EQTerm(BX): 144
DTp2:                                    In eqMainBXTerm EQTerm(BX): -52501
DTp2:                                    In validate Iniciando validacion de MuonPath para capas: 0/1/3
DTp2:                                    Lateralidades parciales: 0/1/0
DTp2:                                    Valores de TDC: 26073/26264/26093.
DTp2:                                    Valid TIMES: 1/1/1.
DTp2:                                    In eqMainBXTerm EQTerm(BX): -52357
DTp2:                                    In eqMainBXTerm EQTerm(BX): 52337
DTp2:                                    In validate Iniciando validacion de MuonPath para capas: 0/2/3
DTp2:                                    Lateralidades parciales: 0/0/0
DTp2:                                    Valores de TDC: 26073/26237/26093.
DTp2:                                    Valid TIMES: 1/1/1.
DTp2:                                    Imposible calcular BX. Denominador para BX = 0.
DTp2:                                    Lateralidad DESCARTADA. Tolerancia de BX excedida
DTp2:                                    Analizando combinacion de lateralidad: 0 1 0 1
DTp2:                                    In validate Iniciando validacion de MuonPath para capas: 0/1/2
DTp2:                                    Lateralidades parciales: 0/1/0
DTp2:                                    Valores de TDC: 26073/26264/26237.
DTp2:                                    Valid TIMES: 1/1/1.
DTp2:                                    In eqMainBXTerm EQTerm(BX): -52501
DTp2:                                    In eqMainBXTerm EQTerm(BX): 52337
DTp2:                                    In validate Iniciando validacion de MuonPath para capas: 1/2/3
DTp2:                                    Lateralidades parciales: 1/0/1
DTp2:                                    Valores de TDC: 26264/26237/26093.
DTp2:                                    Valid TIMES: 1/1/1.
DTp2:                                    In eqMainBXTerm EQTerm(BX): 52330
DTp2:                                    In eqMainBXTerm EQTerm(BX): -52501
DTp2:                                    In validate Iniciando validacion de MuonPath para capas: 0/1/3
DTp2:                                    Lateralidades parciales: 0/1/1
DTp2:                                    Valores de TDC: 26073/26264/26093.
DTp2:                                    Valid TIMES: 1/1/1.
DTp2:                                    In eqMainBXTerm EQTerm(BX): -171
DTp2:                                    In eqMainBXTerm EQTerm(BX): 52337
DTp2:                                    In validate Iniciando validacion de MuonPath para capas: 0/2/3
DTp2:                                    Lateralidades parciales: 0/0/1
DTp2:                                    Valores de TDC: 26073/26237/26093.
DTp2:                                    Valid TIMES: 1/1/1.
--More--
 lxplus014 |                                                        0-$ code   1$ bash
 fun    lx-014    [log] (exported)-...
```

CMS Muon system quadrant