Improvement of calibration of the timing detectors of the Precision Proton Spectrometer of CMS

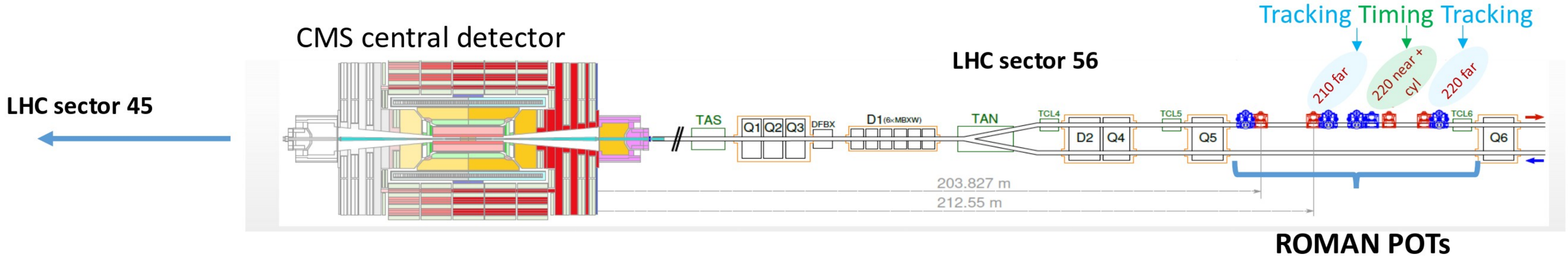
T. Ostafin[1] on behalf of the CMS Collaboration

[1]AGH University of Krakow

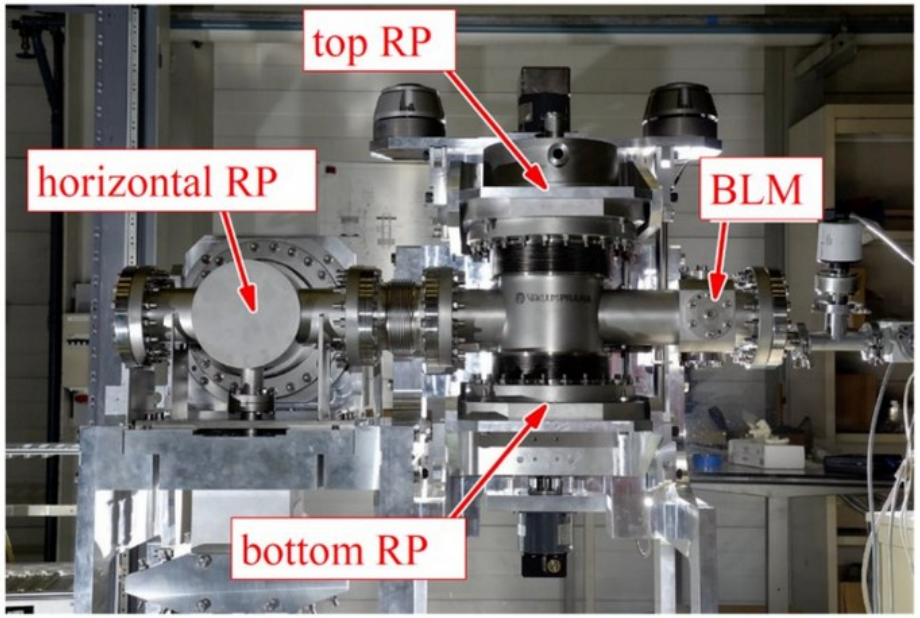
January 14, 2025

Research partially funded by the Ministry of Science and Higher Education grant 2022/WK/14

Precision Proton Spectrometer (PPS)



- PPS is a sub-detector of CMS which extends the physics program to Central Exclusive Production (CEP) processes where both protons remain intact after the interaction at IP 5 [1]
- PPS can measure the proton kinematics, which in combination with the information from the central CMS detector allows to reconstruct the full event
- Consists of tracking and timing detectors located symmetrically on both sides of IP 5 and hosted in movable devices called Roman Pots (RP) which allow to bring the detectors very close to the beam (~1.5 mm for PPS) [2]



PPS Timing Detector

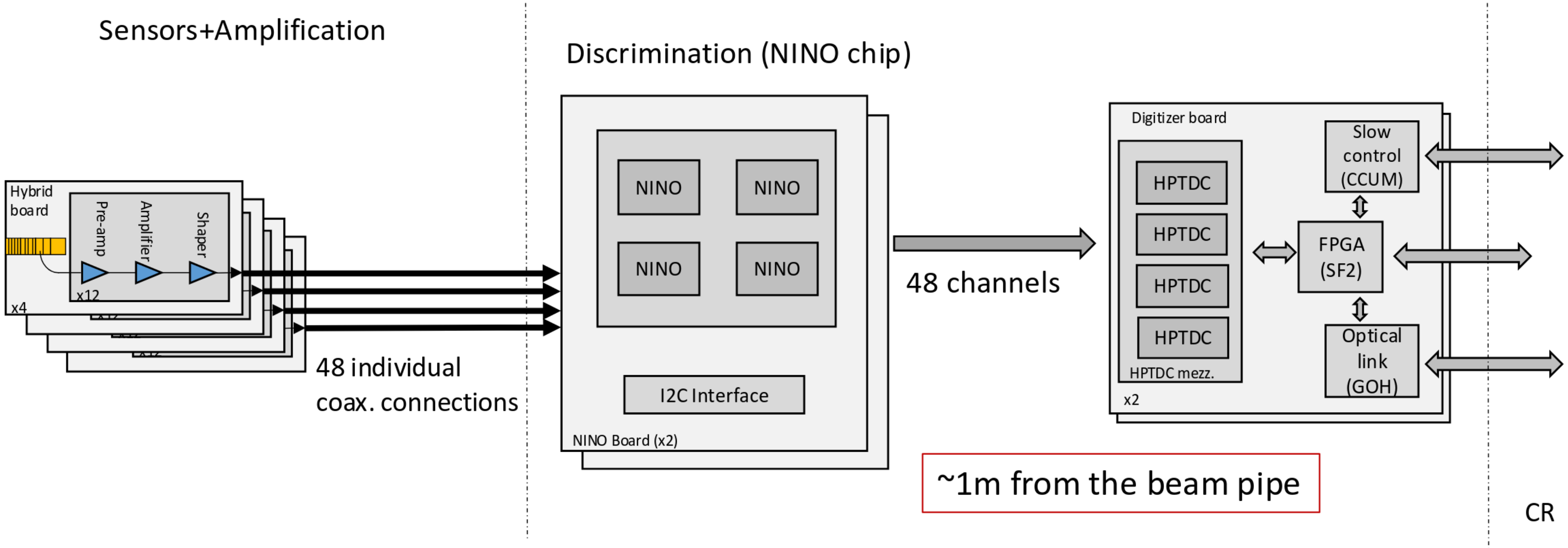
- The PPS tracking detector has no way of disentangling pileup in the CMS detector because of its placement far away from IP 5
- The timing detector is used for measuring the time of flight of the protons, which helps to reduce the pileup effect coming from other collisions
- Started collecting data in Run 2 with 1 station in each sector, with 3 planes in them and a mix of single and double diamond readout channels
- In 2024 the full setup consisted of:
  - 2 sectors (45 and 56)
  - 2 stations (cylindrical and box) per sector
  - 4 planes per station
  - 10-12 double diamond readout channels per plane

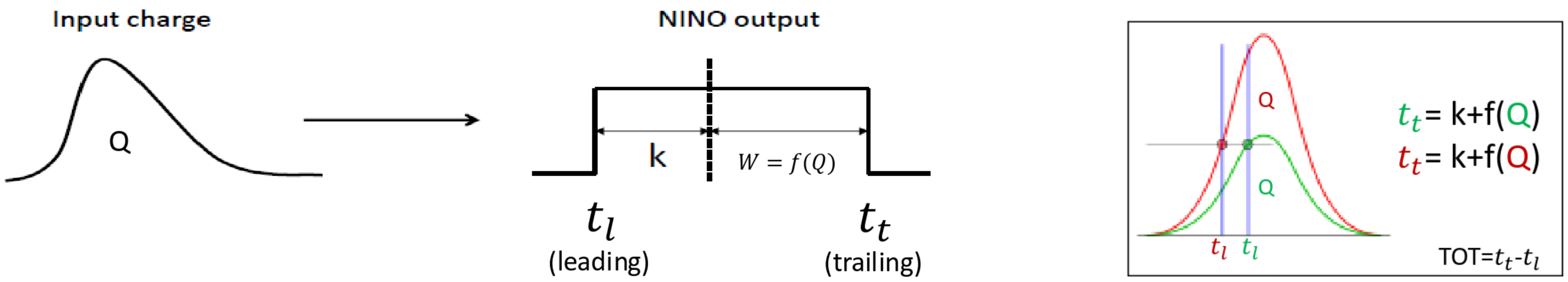
Knowing the difference between the time of flights of the protons (Δt), we can compute the *z* vertex position

$$z_{pp} = \frac{c}{2} \Delta t$$

We can correlate the *z* vertex with one of the vertices reconstructed by CMS and observe if the two protons came from the same vertex or not

PPS Timing Detector Digitization





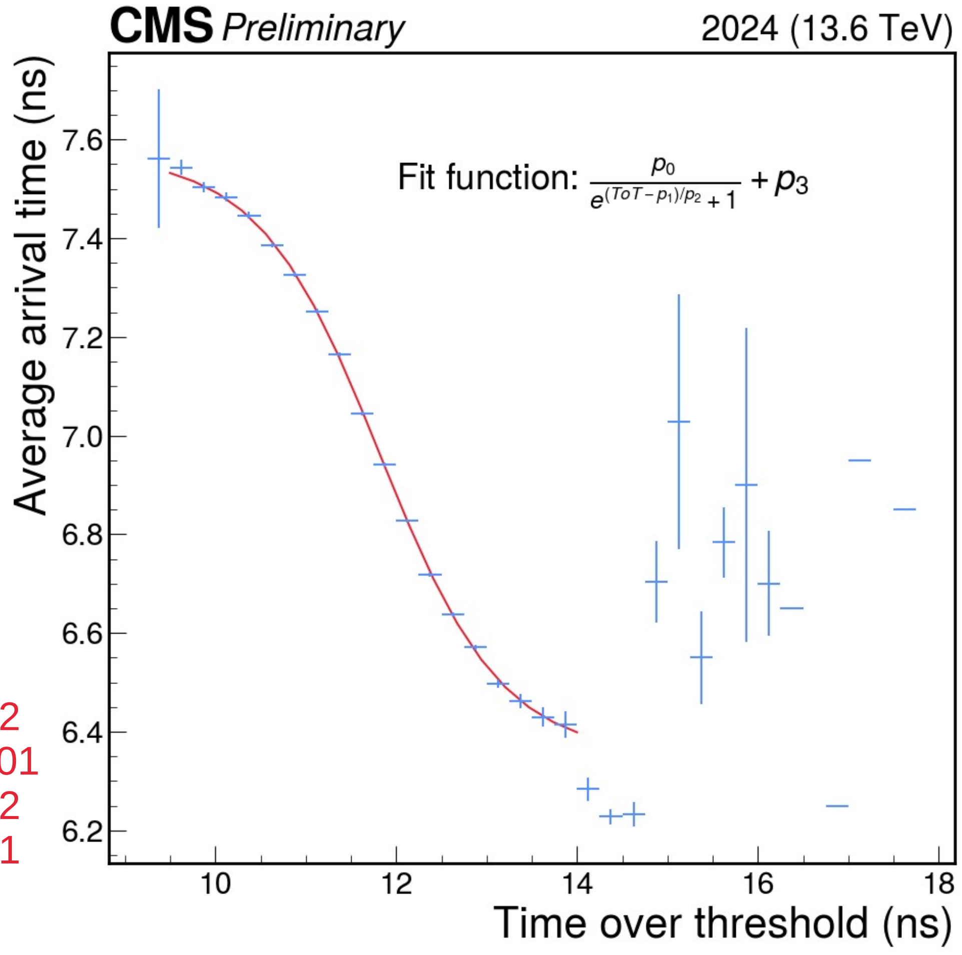
PPS Timing Calibration

- A two-step timing calibration procedure has already been established in Run 2 [3]
  1) Timing correction and alignment
  2) Timing resolution
- In Run 3 new problems have been spotted [4]
  1) Non-converging *t* vs *ToT* (time over threshold) fits for many readout channels
  2) Bad quality of the fits (wrong shape, high χ^2 / ndf etc.)
  3) Leading edge *double peak*

Timing Correction and Alignment

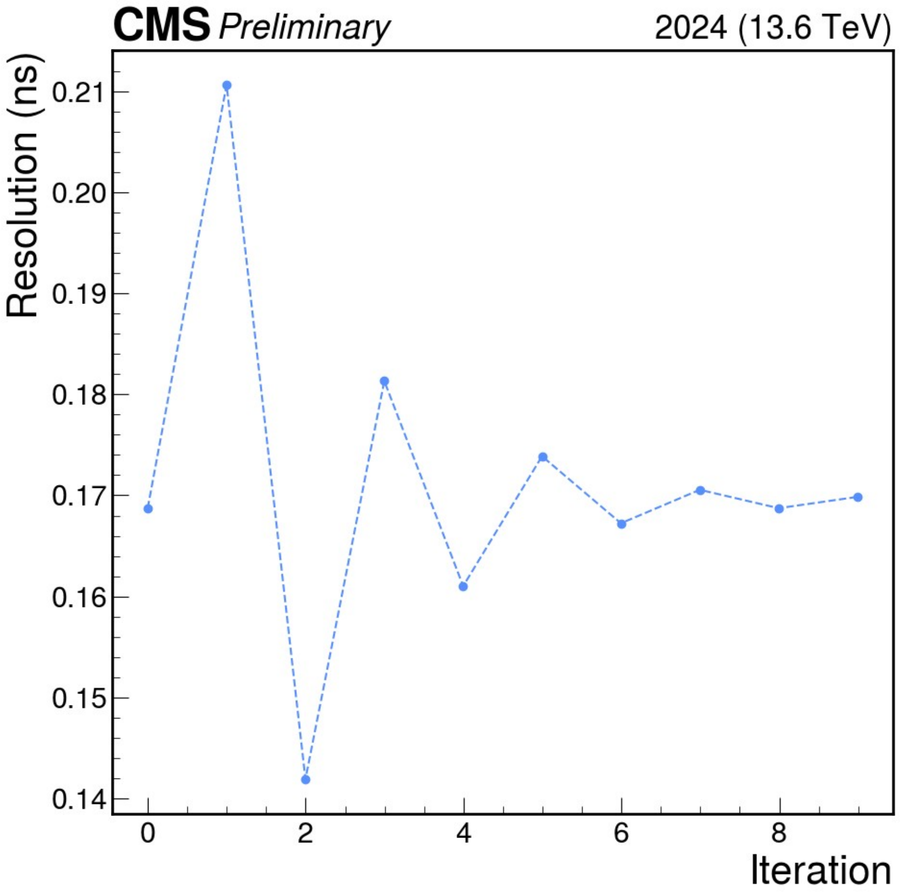
- Used for correcting the time walk effect
- Based on a fit function
- Result: 4 fit parameters
  - p_0: difference between the upper and lower asymptotes
  - p_1: center of the distribution
  - p_2: slope
  - p_3: lower asymptote
- Ideally, an S-shaped curve describing well the most populated ToT regions

$p_0 = 1.22 \pm 0.02$
$p_1 = 11.82 \pm 0.01$
$p_2 = 0.67 \pm 0.02$
$p_3 = 6.25 \pm 0.01$



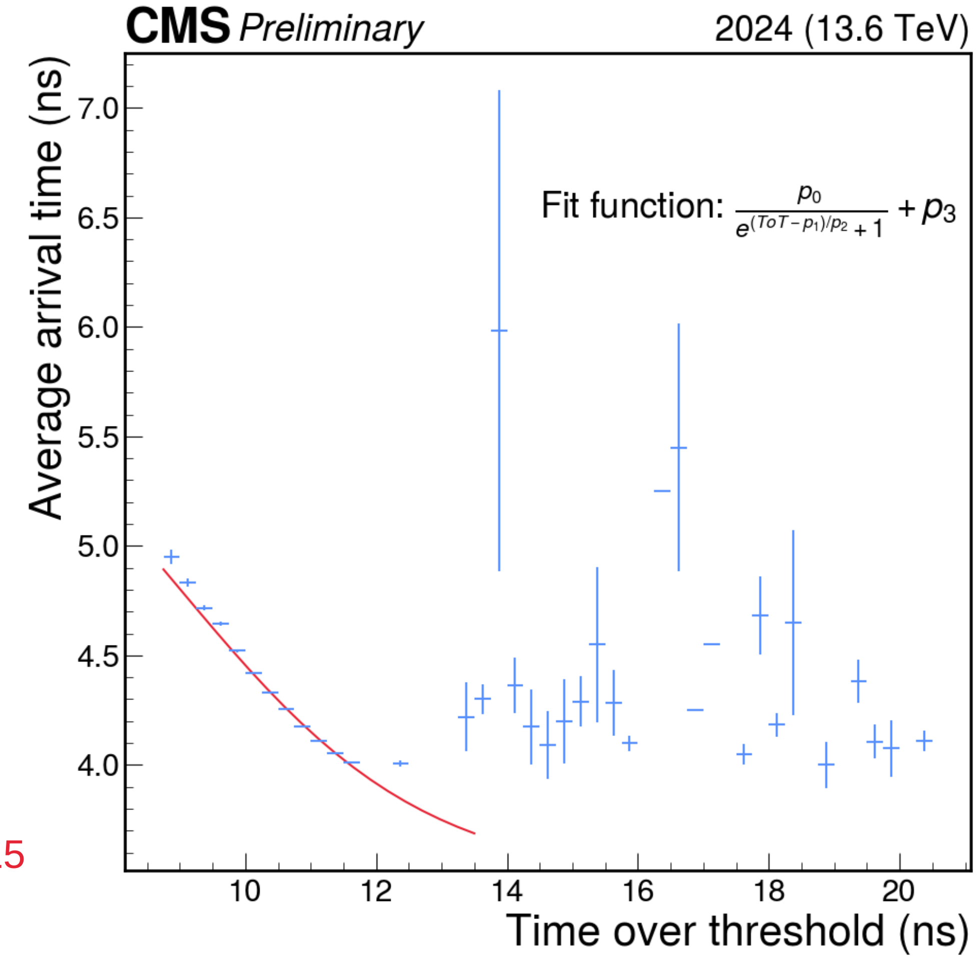
Timing Resolution

- Uses reconstructed and corrected data based on the previously computed fit parameters
- Additionally, tracks reconstructed with the PPS tracking detector are used to reduce the background
- Performed in *n* iterations
  - Throughout Run 2 and in the beginning of Run 3 the value of *n* was arbitrary, usually *n* = 4
  - In each iteration the resolution from the previous step is used as a weight in the current one
- Ideally, converging to a certain value after *n* iterations



Bad Quality Fits

- As opposed to Run 2, in Run 3 many channels have data which doesn't align into an S shape
- The fit doesn't describe the most populated ToT regions very well for both low and high ToT values
- As a result, high χ^2 / ndf
- Often results in non-converging fits

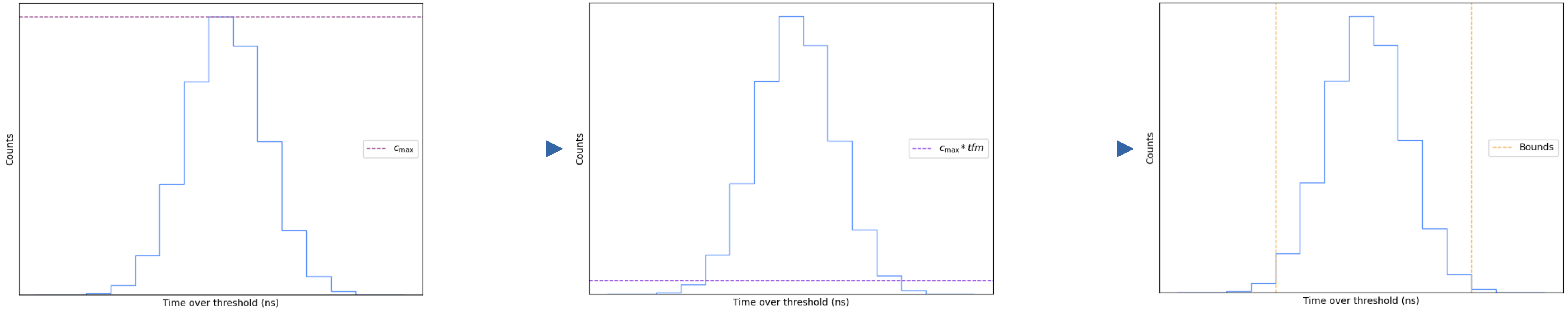


χ^2 / ndf = 1649.34 / 15

Fit Improvements

- Changing the fit parameters limits
  - Previously, only two of them were bounded
  - Old limits weren’t good enough in Run 3
- Increasing the max function call limit of the minimizer
- Introducing iterative thresholds
  - In Run 2 the fit had constant bounds
  - In the beginning of Run 3 the bounds were based on an arbitrary constant ToT fraction of its max bin
  - Now, that fraction is iterative and the bounds are set to the ones which give the best χ^2 / ndf

Constant Threshold

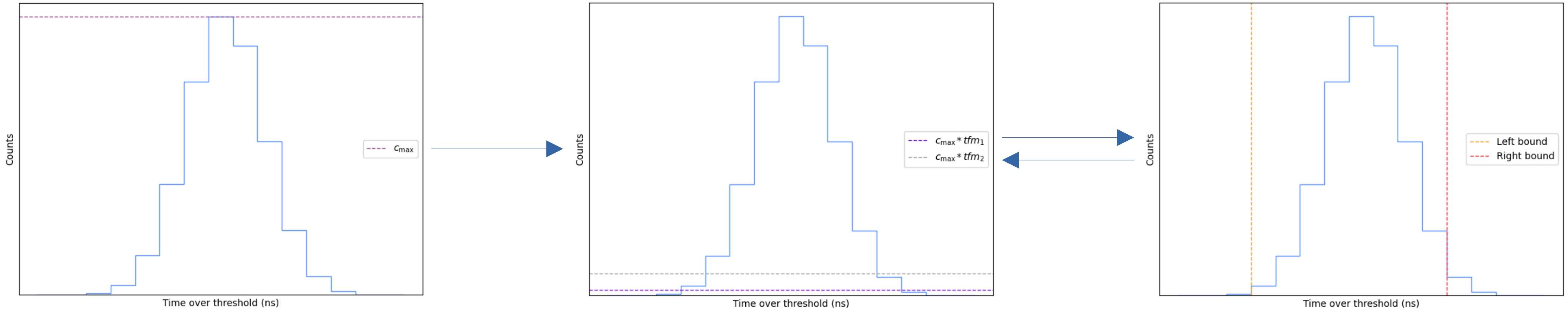


1. Find the bin with the max count (c_{max})

2. Compute the threshold by multiplying c_{max} with a constant threshold fraction of max (*tfm*) value

3. Find the bins with the min count which are still above the threshold to determine the fit bounds

Iterative Thresholds



1. Find the bin with the max count (c_{max})

2. Pick tfm_1 and tfm_2 for the left and right bounds respectively

3. Compute the thresholds by multiplying c_{max} with tfm_1 and tfm_2

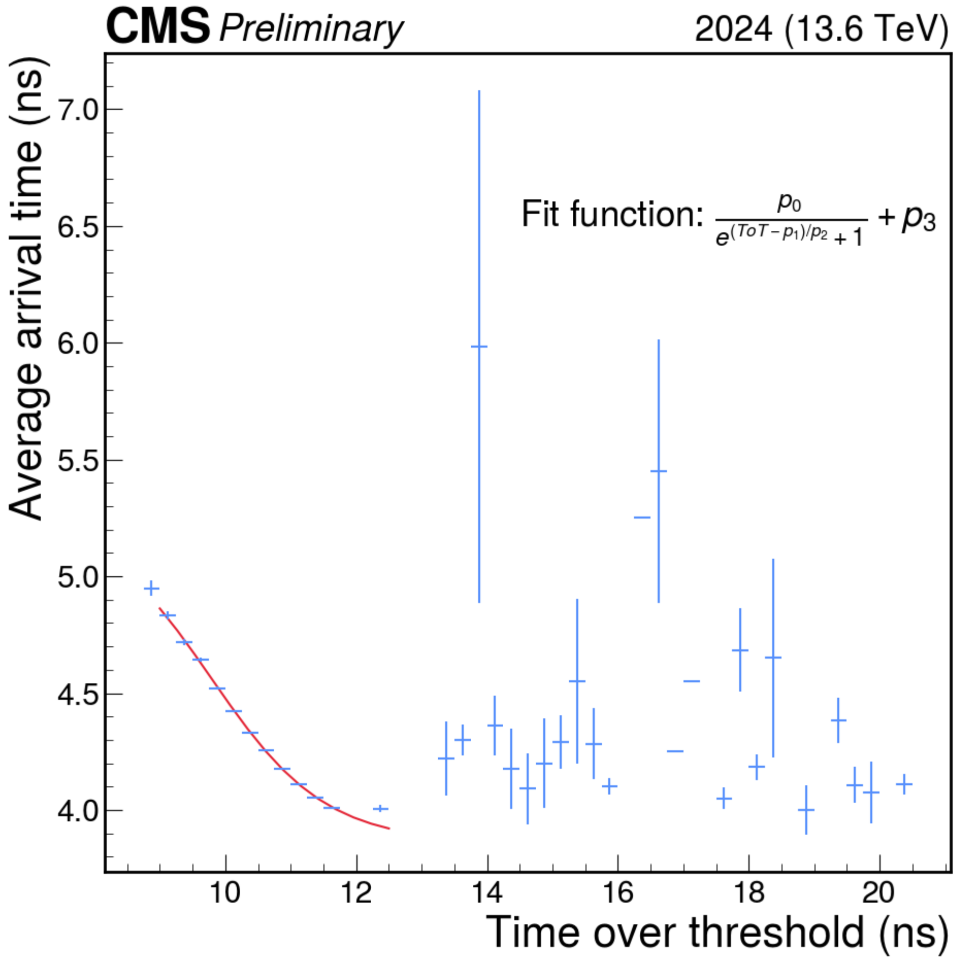
4. Find the bins with the min count which are still above the thresholds to determine the fit bounds

5. Check χ^2 / ndf, save if the best and go to 2.

Fit Improvements Results

- The most populated ToT regions well described
- Low χ^2 / ndf
- Small chance of non-converging fits
- Even though the data doesn't always align into an S-shape, it can still be fit properly most of the time

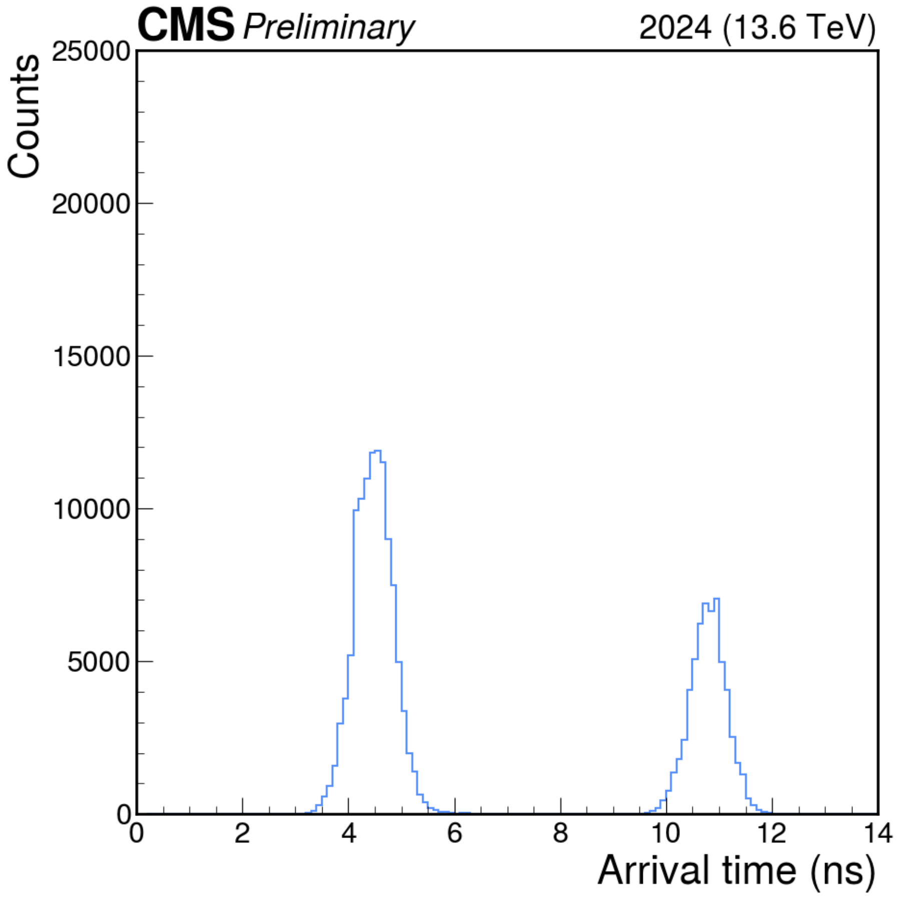
χ^2 / ndf = 67.34 / 11



Leading Edge Double Peak

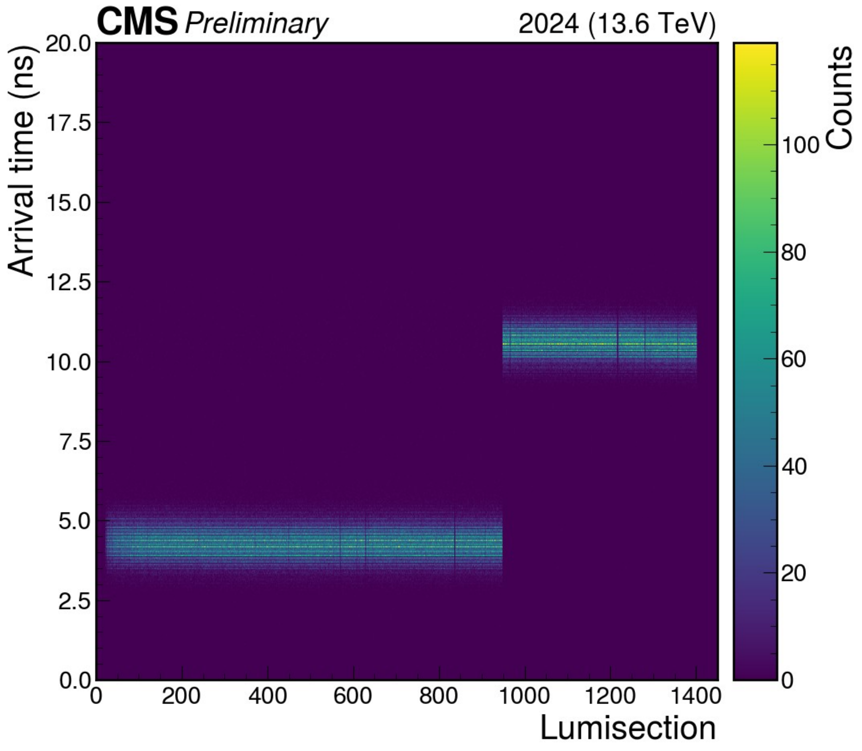
Channel distribution

- Detectors are tuned to have the signal leading edges concentrated around a certain value (~5 ns)
- This ensures that also the trailing edge of the signal (~13 ns after the leading) is registered in the 25 ns acquisition window
- Sometimes, during a data acquisition run a shift happens to either higher or lower values
- Reasons aren’t exactly known; probably a phase shift of the precision clock used for the timing measurement, possibly due to a single event upset in the clock distribution circuitry



Double Peak Correction

Plane distribution

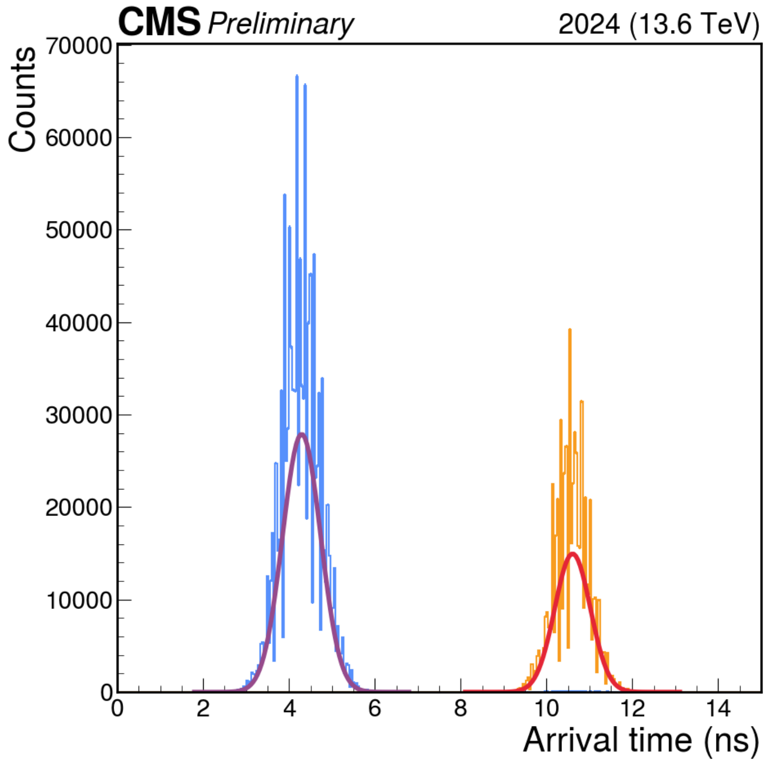


1. Detect the double peak lumisection

2. Project the Y axis and fit the Gaussians

$\mu_1 = 4.29$
$\mu_2 = 10.61$

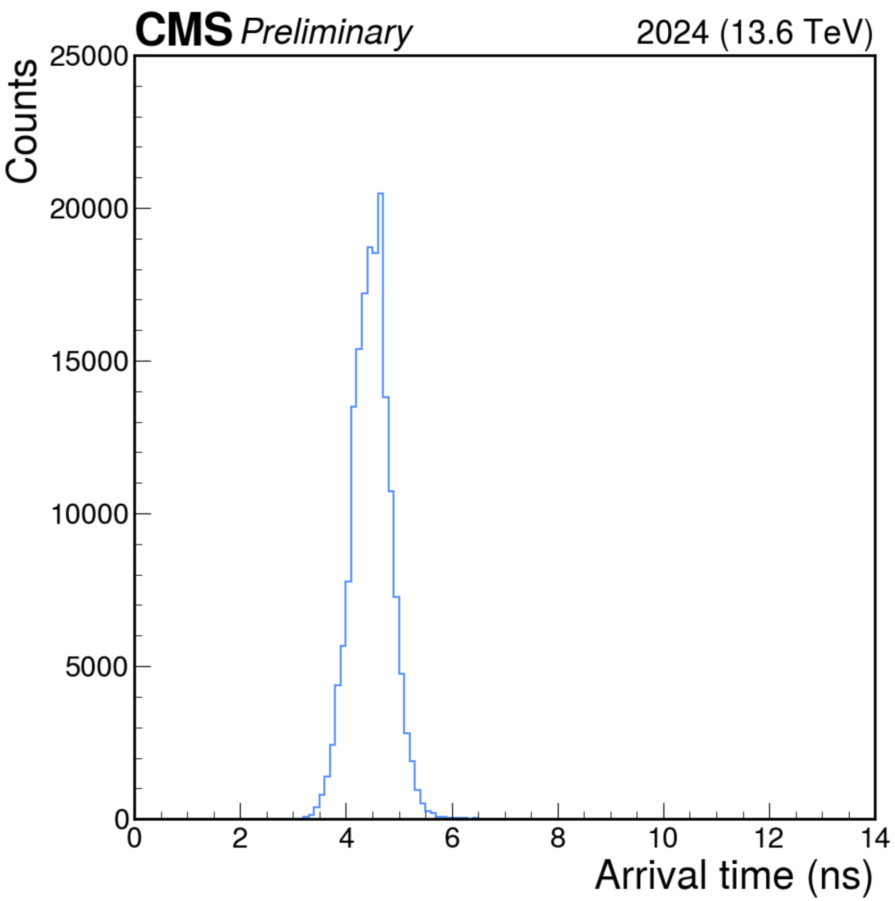
Plane distribution



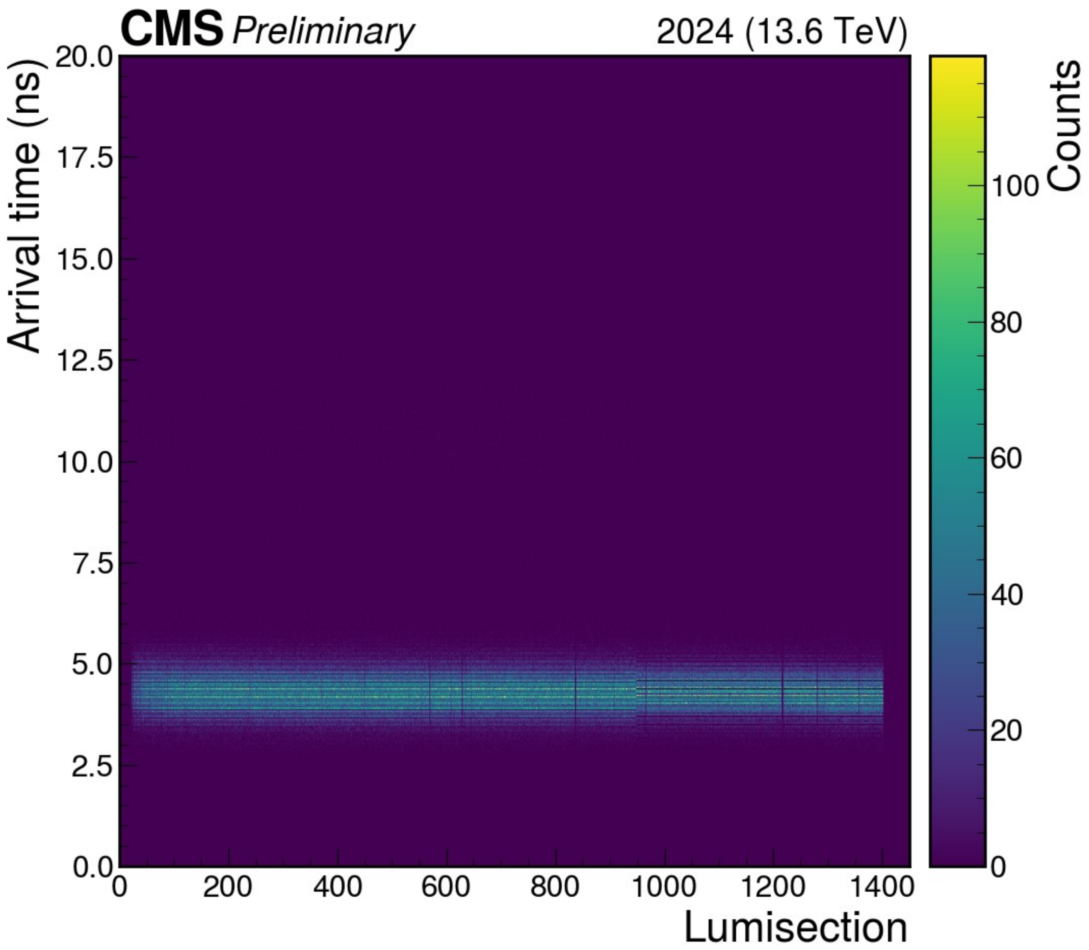
3. Compute the shift by subtracting $\mu_2 - \mu_1$

After Double Peak Correction

Channel distribution

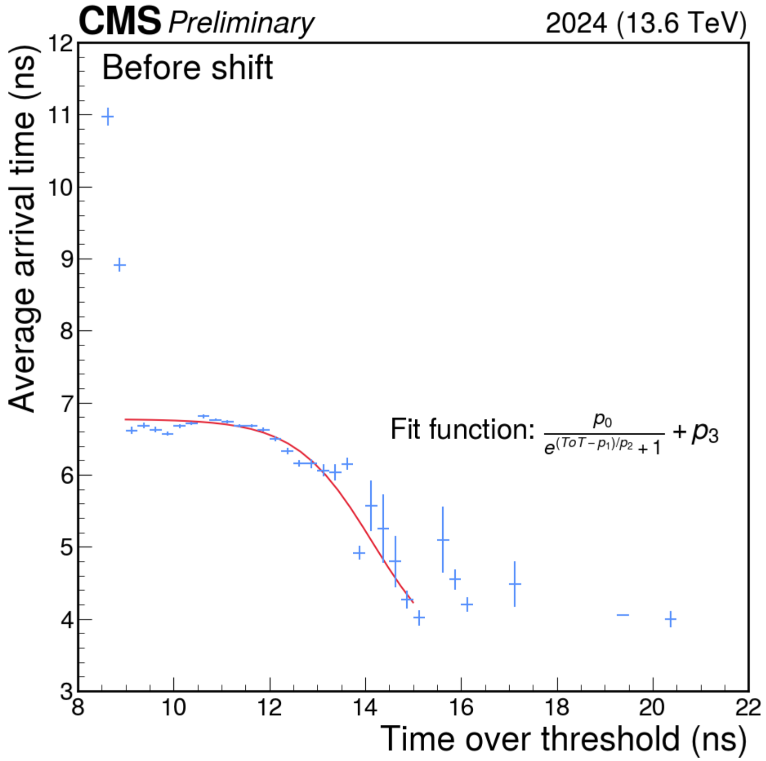


Plane distribution



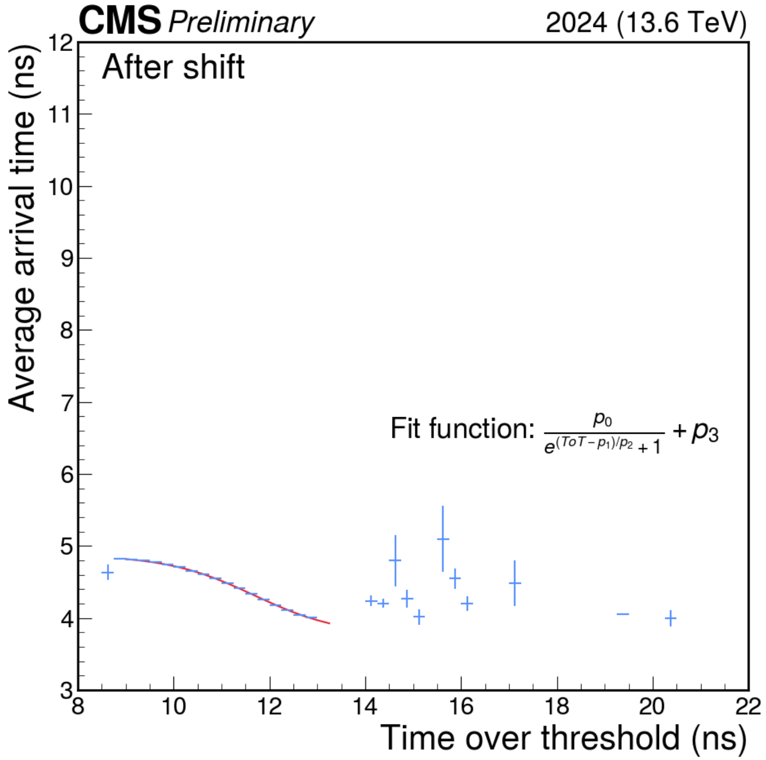
t vs *ToT* Fit

χ^2 / ndf = 645.2 / 21



Bad shape, the most populated regions aren't described very well, big χ^2 / ndf

χ^2 / ndf = 40.34 / 14



Good shape, describes the most populated regions very well, low χ^2 / ndf

Automation Workflow

- A step-by-step automatic computation per Run based on user-defined datasets (e.g. *ZeroBias*, *AlCaPPSPrompt*).
- A workflow consists of a set of **tasks** which are logically grouped**:**
  - Workers (one worker per dataset file)
  - Harvesters (grouping workers' results)
  - Others (e.g. validation, iteration)
- A task execution is represented by a **job**. One task can have many jobs (e.g. one job per input file).
- Tasks run in a pipeline, i.e. are executed one-by-one and depend on each other (e.g. a harvester task can only begin when all worker jobs have finished processing).
- Often the injection of Runs to be processed happens automatically (**prompt mode**) through periodic synchronization (e.g. with CMS Online Monitoring System (OMS)).

PPS Automation Framework

Software for running automation workflows for PPS:

tracking efficiency and timing calibrations [5]

- Originally developed by ECAL
  - Relies on their automation control library which is a steering wheel for the framework
  - We made contributions to this library as well
- Includes different modes: *dev* (manual Run injection and testing), ***prompt*** (automatic Run sync with OMS) and *repro* (allows reprocessing campaigns)
- Comes with full doumentation which is being constantly developed alongside new features
- Previously **manual** calibration performed by a person took **~1-2 months**; now reduced to **~36 h** done **fully automatically**
  - Easier testing of improved workflows due to much lower turn-around time
  - Made this analysis a lot faster!

Iteration Stop Condition

- Previously stopping the timing resolution workflow after meeting a certain number of iterations
  - Not ideal since Runs differ from each other and in some channels converge much faster than in others
  - We ended up doing either too few (bad resolution) or too many (wasting resources) iterations for some Runs
  - Required monitoring to change the iteration number if necessary
- Instead, a new multi-step condition was implemented:
  1) Perform at least iteration 4
  2) If at least 95% of the channels have a difference between the resolution from the previous step and the current one below 10 ps, stop iteration
  3) Otherwise, continue unless iteration 10 has been reached

```
[TIMING-RESOLUTION-ITERATION]: Channels below delta: 164. Working channels: 174.
[TIMING-RESOLUTION-ITERATION]: Iteration 8 for Run 383418 requested. Tasks to iterate marked for reprocessing: ['timing-resolution-worker', 'timing-resolution-harvester']
[TIMING-RESOLUTION-ITERATION]: Max iterations reached for Run 383487.
[TIMING-RESOLUTION-ITERATION]: Iteration stop condition for Run 383487 reached after iteration 10 for tasks: ['timing-resolution-worker', 'timing-resolution-harvester'].
[TIMING-RESOLUTION-ITERATION]: Channels below delta: 167. Working channels: 174.
[TIMING-RESOLUTION-ITERATION]: Iteration stop condition for Run 385127 reached after iteration 9 for tasks: ['timing-resolution-worker', 'timing-resolution-harvester'].
```

Validation Task Handler

- Some Runs require reprocessing the workflow based on the produced results (e.g. double peak correction)
- For some Runs a workflow can also produce results which seem valid from the CMSSW point of view but are useless
  - For instance, for the timing correction and alignment workflow this results in all channels having parameters equal to zero
  - It could cause the timing resolution workflow to fail since all channels are broken
- A handler has been implemented in our framework to validate results after each processing.
  - If validation detects a need for reprocessing, it marks the specified tasks as *reprocess*.
  - If validation fails, the whole task is marked as *failed* and its dependencies as *skipped,* which will cause the global status to be *failed* as well.

Timing calibration overview

Run	Fill	Global status	timing-time-walk-worker	timing-time-walk-harvester	timing-time-walk-validation	timing-resolution-worker	timing-resolution-harvester	timing-resolution-iteration
383767	9943	failed	done	done	failed	skipped	skipped	skipped
384273	9988	failed	done	done	failed	skipped	skipped	skipped
384593	10013	failed	done	done	failed	skipped	skipped	skipped

References

[1] CMS and TOTEM Collaborations, *CMS-TOTEM Precision Proton Spectrometer*, CERN-LHCC-2014-021 ; TOTEM-TDR-003 ; CMS-TDR-013

[2] E. Bossini, *Status and perspective of the CMS Precision Proton Spectrometer timing system*, FAST 2023

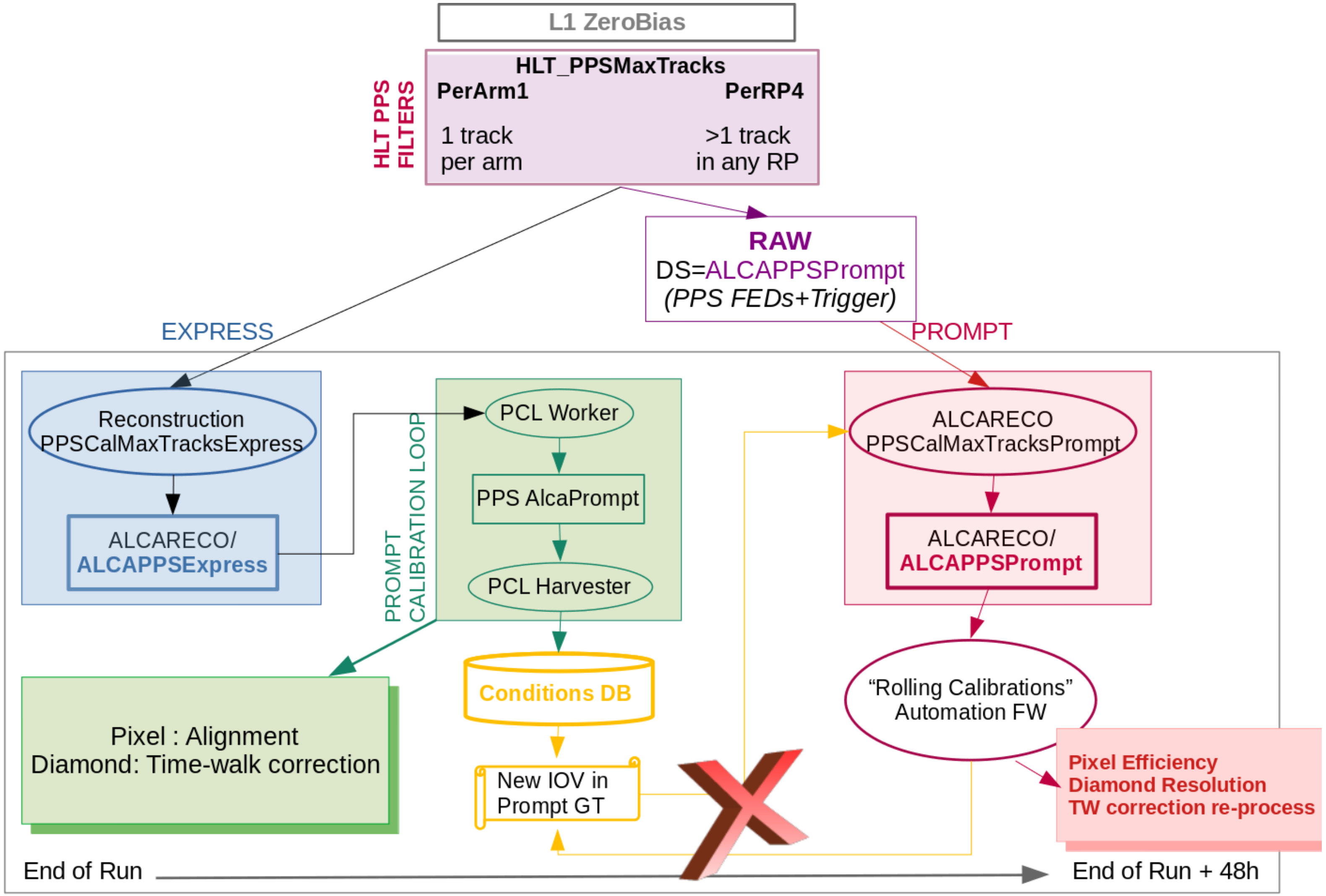
[3] CMS Collaboration, *Time resolution of the diamond sensors used in the Precision Proton Spectrometer*, CMS-DP-2019-034

[4] CMS Collaboration, *Improvement of the timing calibration in the CMS-PPS timing detectors*, CMS-DP-2025-001

[5] A. Bellora, T. Ostafin, *An automation framework for the calibration of the CMS Precision Proton Spectrometer*, CHEP 2024

Backup Slides

Framework Input



Automation Technologies

- InfluxDB
  - A time series database used for storing information about workflows, tasks and jobs
  - Access through the automation library
- HTCondor
  - A high performance framework for parallel job processing
  - Executed through CLI (*condor_**); in our case, through Python scripts in the framework
- Grafana
  - A monitoring tool for the framework
  - Integrated with the InfluxDB instance
  - Allows easy filtering, provides charts etc
- Jenkins
  - A DevOps tool for executing tasks in a pipeline

Automation Prompt Mode

Both timing and tracking efficiency workflows are working now in prompt

- This means Runs are periodically synced with OMS every 1 h and processed once their datasets are available (usually up to 48 h)
- Every Run from 2024 was processed if it passed a two-step filtering:

    1) Based on the OMS *runs* resource:
        a) *sequence = GLOBAL-RUN*
        b) *last_lumisection_number >= 100*
        c) *stable_beam = true*
    2) Our own Run filter (based for now on the *lumisections* resource):
        a) *rp_sect_45_ready >= 100*
        b) *rp_sect_56_ready >= 100*
        c) *rp_time_ready >= 100* (only for the timing workflows)