

time_measurement_audit_summary

Analysis of time measurement in `algo`. Prepared with assistance of OpenCode agent with `DeepSeek-V4.1-Flash` model.

Not every line was verified by human, so take it with a grain of salt.

Timing APIs

- xpu timers:
 - per-thread stack; parent time **includes** children (no subtraction)
 - `scoped_timer` is RAI; manual `push` / `pop` leaves a stale frame on exceptions/early returns
 - `t_add_bytes` counts only in the innermost timer
 - Monitor timers:
 - plain start/stop accumulation; restarting a running timer silently overwrites the start
 - a missing `StopTimer` loses the interval without any error
 - multi-thread merge (`AddTimer(..., parallel=true)`) stores **max** total but **summed** call counts
 - `ca::Timer::AddTimer` bug: `fMaxCallIndex` computed from `other.fMinCallIndex` (`CaTimer.h:128`)
-

Usage

- xpu timer usages: **17 files**, ~70 instances
 - production: unpack, event building, triggers, BMON/TOF/TRD/STS, Reco pipeline, CA chain
 - GPU tracking: large instrumentation exists but is compiled out (`GpuTimeMonitoring=false`)
- Monitor usages: **7 files**
 - CA: `TrackingChain`, `CaTrackFinder`, `CaTrackFinderWindow`
 - event reco: `EventReconstruction`; event selection: `Reco`
 - KFP: `KfpV0Finder`
- Monitor specializations: CA, evreco, evselect, KFP V0

- Also found ad-hoc `std::chrono` / `ca::Timer` measurements (QaManager, CaTrackFinder, CbmRecoDevice)
-

QA and monitoring inside timers

- `Reco.STS` / `Reco.TOF` / `Reco.TRD` measure hit finding **plus**:
 - `Queue*Metrics` calls and `f*HitFinderQa->Exec` (`Reco.cxx:635-679`)
 - `Reco.Unpack` includes `QueueUnpackerMetricsDet` and digi QA execution (`Reco.cxx:592-631`)
 - `Reco.CA` includes track sorting (V0-trigger-only) and `QueueTrackingMetrics` (`Reco.cxx:688-707`)
 - Outer `Reco` includes `fQaManager->SendHistograms()` (`Reco.cxx:834`)
 - `TrackingChain.CA` includes CA QA, debug monitor file dump, monitor merge (`TrackingChain.cxx:114-129,245-258`)
 - `V0Trigger` fills QA histograms in the pair loop (`V0Trigger.cxx:38-67`)
 - `KfpSelection` includes KFP-selector QA (already flagged by a comment in `Reco.cxx:751`)
-

Measurements ending too early

- TOF hit finding: timer stops before the output `PartitionedVector` is built (`tof/Hitfind.cxx:177` vs `182`)
 - TRD hit finding: timer stops before output assembly and per-partition time sort (`trd/Hitfind.cxx:304` vs `309-316` ; same in module-parallel path)
 - STS CPU `FlattenHits`: stops before offsets and `PartitionedSpan` construction (`HitfinderChainCpu.cxx:497-516`)
 - CA `PrepareTimeslice` stops before XPU initialization (`CaTrackFinder.cxx:347-348`)
 - `TrackingChain` monitor timer stops before monitor merge/copy (`TrackingChain.cxx:243` vs `257-258`)
 - KFP: early return for events without T0 skips both `StopTimer` calls (`KfpV0Finder.cxx:284-288`)
 - also returns an **uninitialized** `bool` on that path
-

Not measured sections

- Event-building orchestration and digi-time extraction (`EventbuildChain.cxx:92-93, 148-217`)

- `PidParamBuilder::Update` inside event reconstruction (`EventReconstruction.cxx:263-265`)
 - `KfpSelectorChain::ProcessEvent` has no internal timers (only thread-0 `KfpSelection`)
 - `TrackingChain::PrepareOutput` , `InitializeXpuForGpuTrackingOnce`
 - STS core pipeline (GPU and CPU): `SortDigis` , cluster finding, `SortClusters` , `FindHits`
 - only the umbrella `Reco.STS` exists; GPU D2H flatten copy is untimed
 - `FlattenClusters` has no timer at all
 - GPU multi-window: `PrepareWindows` , `SuppressNeighbourWindows` , `AppendWindowOutput`
-

API misuse and bugs

- `PrepareGrid` in GPU single-window path: `StartTimer` called **twice**, never stopped
 - `CaTrackFinderWindow.cxx:322-324` ; second call must be `StopTimer` (batch path is correct, 472-474)
 - TRD `monitor.sortTime` is measured then explicitly zeroed on the active path (`trd/Hitfind.cxx:178-179`)
 - `HitMultTrigger` uses `t_add_bytes(1)` → meaningless throughput (`HitMultTrigger.cxx:25`)
 - Manual `push/pop` (unpack, event builder, triggers, TOF/TRD/BMON) is not exception-safe
 - `TimeClusterTrigger` shows the correct pattern: validate before `push_timer`
 - Dead GPU code: enabling `GpuTimeMonitoring_tracking` leaks xpu timer frames on `continue` (`CaGpuTrackFinderSetup.cxx:1466-1487`)
-

Disabled and dead measurement code

- Entire GPU xpu instrumentation compiled out:
 - `GpuTimeMonitoring=false` , `GpuTimeMonitoring_tracking=false` (`CaDefs.h:134,140`)
 - ~50 timers in `CaGpuTrackFinderSetup.cxx` + `CaTrackFinderWindow.cxx` never execute
 - `GpuTimeMonitor` class unused; `GpuTrackFinderSetup::GetTimings()` never read
 - per-iteration results go to local floats + `printf` instead of monitors
- Never-used timer keys: `ETimer::PrepareContainers` (KFP)
- Abandoned local timers in `CaTrackFinder::FindTracksThread` (stop commented out)

- Misleading leftovers: impossible condition in `TrackingChain.cxx:246`
(`GetNofThreads() == 1 && GetNofThreads() == 10`), commented-out debug timers
-

Cross-cutting semantics

- Nested timers (xpu) and nested monitor scopes overlap; summing parent + child overestimates
 - Same work measured by different systems with different boundaries:
 - `Reco.CA` (xpu) vs CA monitor `Tracking` vs `fCaRecoTime` (`std::chrono`)
 - `Reco.EventReco` (xpu) vs per-step evreco monitors
 - Multi-thread merge: totals are maxima, call counts are sums → derived averages/ratios are inconsistent
 - CA (`CaTrackFinder.cxx:391`) and event reco (`Reco.cxx:772-785`)
 - Same timer key, different meaning per backend:
 - GPU `CreateTracks` = D2H copy; CPU = candidate construction
 - GPU `FindTracks` includes fitting/merging; CPU does not
 - `SubTS` counter differs between window loop and multi-window GPU
 - `Reco` / `TrackingChain.CA` include logging, allocations and I/O
-

cbmreco

- `online/device/CbmRecoDevice.cxx` and `reco/app/cbmreco/main.cxx`
 - `CbmRecoDevice.WriteArchive` / `cbmreco.WriteArchive` timer includes `makeStorableRecoResults` : event selection + full copies of all digis/hits/tracks + `SizeBytes` + archive write (`:110-114`)
 - `timeIdle` includes the timeslice deep copy in ReleaseMode (`:96,103-105`)
 - total wall time uses yet another clock than the in-chain `Reco` timer (`xpu::timings` wrapper)
-

Recommendations

1. Move QA execution and metric sending **outside** the scoped timers (or add separate QA timers)

2. Extend timer ranges to include output assembly, or add dedicated “store/flatten” timers
 3. Fix the double `StartTimer` and stop discarding `sortTime` ; make `pop_timer` RAI everywhere
 4. Add timers for STS core kernels, GPU multi-window prep, PID update, KFP selector internals
 5. Decide on one measurement source per algorithm; document parent/child relationships
 6. Either enable and maintain the GPU timing instrumentation or remove it
 7. Consider a lint/CI check: push/pop balance, start/stop pairing, no QA inside timers
-