Active injection veto from BGO in MC run-dependent

Giacomo De Pietro

24/02/2025 – TRG parallel session

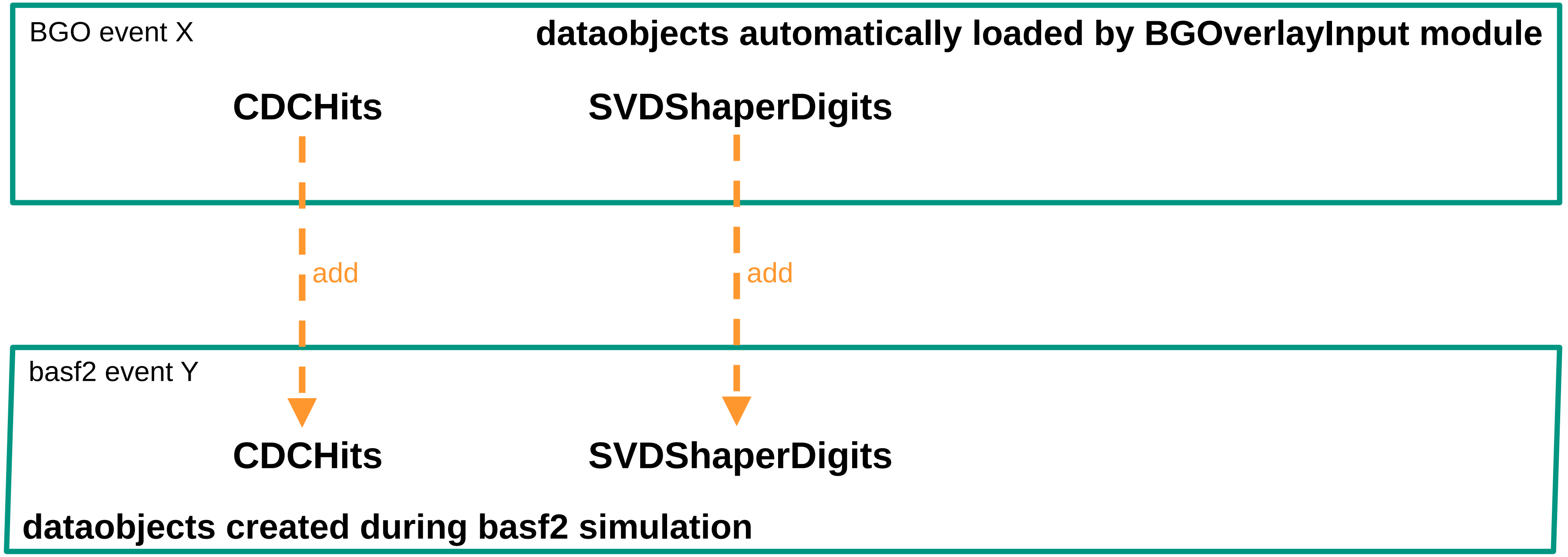
Active injection veto

- Active injection veto has been introduced in Run2
- It allows triggering events close to the injection if backgrounds are low
- Preliminary studies from performance group reported that these events are still good enough to be used in physics analyses
- But… **what about MC run dependent?**

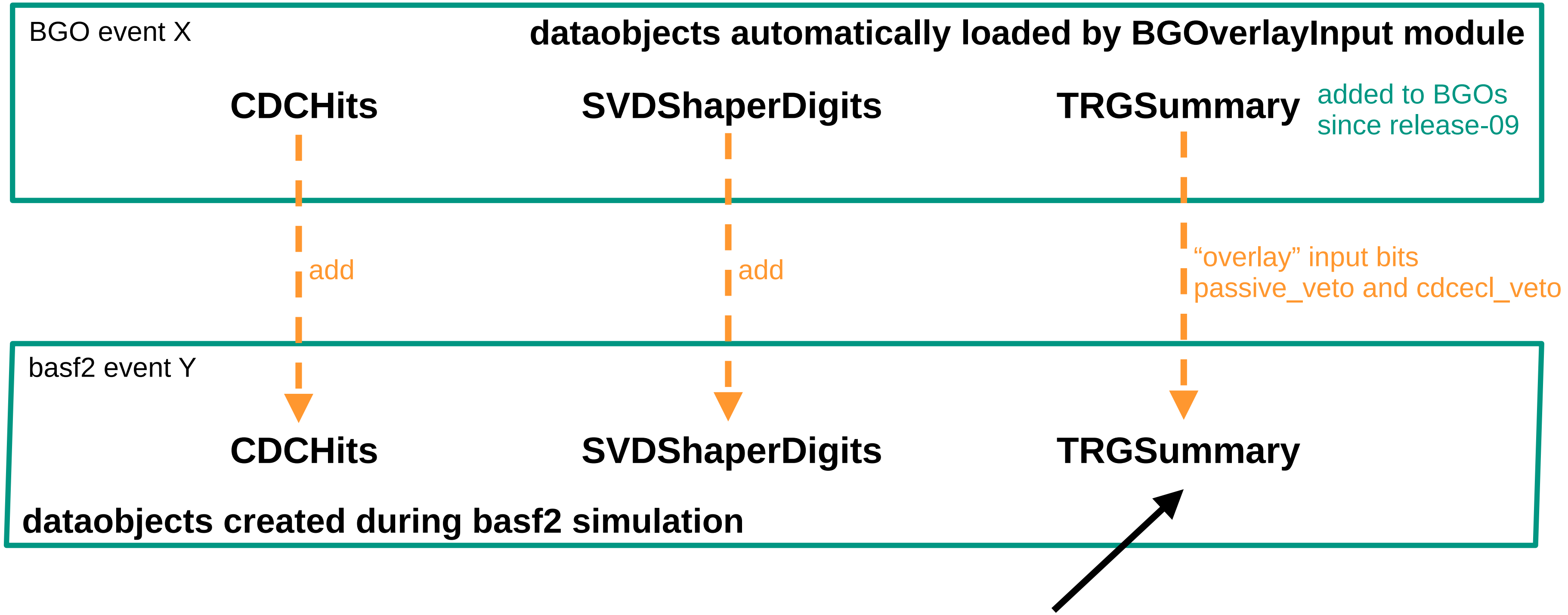
Beam background overlay

- MC run dependent uses the beam background overlay (BGO) events recorded during data taking instead of simulated beam background
- BGOs are produced by:
  - Selecting events triggered as delayed bhabha
  - Unpack such events and store in output the necessary dataobjects (mostly digits)
- Whenever a physics event is simulated with basf2, we take a random event from a BGO file and we (literally) overlay the digits from the BGO event to the basf2 one
  - So… **what about overlaying to the physics event the information about the BGO event being (or not being) triggered using the new active injection veto scheme?**

Beam background overlay


BGO event X
dataobjects automatically loaded by BGOverlayInput module
CDCHits
SVDShaperDigits
add
add
basf2 event Y
CDCHits
SVDShaperDigits
dataobjects created during basf2 simulation

Beam background overlay


BGO event X
dataobjects automatically loaded by BGOverlayInput module
CDCHits
SVDShaperDigits
TRGSummary
added to BGOs since release-09
add
add
"overlay" input bits passive_veto and cdcecl_veto
basf2 event Y
CDCHits
SVDShaperDigits
TRGSummary
dataobjects created during basf2 simulation

Similar approach as for EventLevelTriggerTimeInfo,
which contains infos from RawFTSWs like time since last injection

When doing the overlay of input bits in TSIM

- The easiest way I found to “overlay” the passive_veto and cdcecl_veto input bits is between the TRGGRLProjects (which fills the input bits) and the TRGGDL module (which fills TRGSummary) by adding a new basf2 module (TRGGRLInjectionVetoFromOverlay):

```
add_simulation(path)
basf2.print_path(path)
…
24. TRGGRLProjects
25. TRGGRLInjectionVetoFromOverlay
26. TRGGDL
```

How doing the overlay of input bits in TSIM

- The easiest way I found to “overlay” the passive_veto and cdcecl_veto input bits is between the TRGGRLProjects (which fills the input bits) and the TRGGDL module (which fills TRGSummary) by adding a new basf2 module (TRGGRLInjectionVetoFromOverlay):

class members of TRGGRLInjection… module

```
/** TRGGDLInputBits database object */
DBObjPtr<TRGGDLDBInputBits> m_TRGInputBits;

/** TRGGRLInfo object from MC simulation */
StoreObjPtr<TRGGRLInfo> m_TRGGRLInfoFromSimulation;

/** TRGSummary object from beam background overlay */
StoreObjPtr<TRGSummary> m_TRGSummaryFromOverlay;

/** Name of TRGGRLInfo object */
std::string m_TRGGRLInfoName;
```

← output of TRGGRLProjects

← TRGGRLInfo is actually named TRGGRLObjects

How doing the overlay of input bits in TSIM

- The easiest way I found to “overlay” the passive_veto and cdcecl_veto input bits is between the TRGGRLProjects (which fills the input bits) and the TRGGDL module (which fills TRGSummary) by adding a new basf2 module (TRGGRLInjectionVetoFromOverlay):

event method of TRGGRLInjection… module

```
void TRGGRLInjectionVetoFromOverlayModule::event()
{
  if (!m_TRGGRLInfoFromSimulation.isValid() or !m_TRGSummaryFromOverlay.isValid())
    return;
  try {
    // Set the passive_veto and cdcecl_veto input lines according to what is written
in the BGO event
    const unsigned int passive_vetoBit = m_TRGInputBits->getinbitnum("passive_veto");
    const bool passive_vetoAnswer = m_TRGSummaryFromOverlay->testInput("passive_veto");
    m_TRGGRLInfoFromSimulation->setInputBits(passive_vetoBit, passive_vetoAnswer);
    const unsigned int cdcecl_vetoBit = m_TRGInputBits->getinbitnum("cdcecl_veto");
    const bool cdcecl_vetoAnswer = m_TRGSummaryFromOverlay->testInput("cdcecl_veto");
    m_TRGGRLInfoFromSimulation->setInputBits(cdcecl_vetoBit, cdcecl_vetoAnswer);
  } catch (const std::exception&) {
```

“overlay” passive_veto

“overlay” cdcecl_veto

How doing the overlay of input bits in TSIM

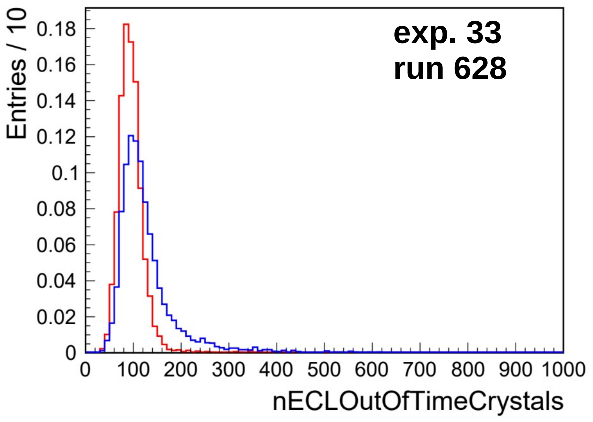
- The easiest way I found to “overlay” the passive_veto and cdcecl_veto input bits is between the TRGGRLProjects (which fills the input bits) and the TRGGDL module (which fills TRGSummary) by adding a new basf2 module (TRGGRLInjectionVetoFromOverlay):
- In this way, TRGGDL will fill the TRGSummary object according to what’s written in the TRGGRLObjects object
- The only requirement is that TRGGDLDBInputBits payload is up-to-date at the time MC run dependent is produced, which should always be true

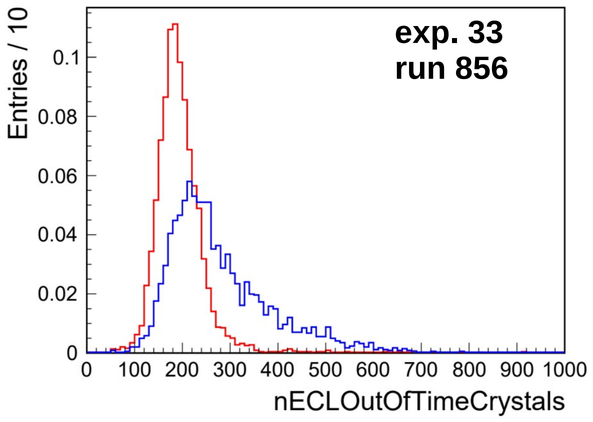
Does this work?

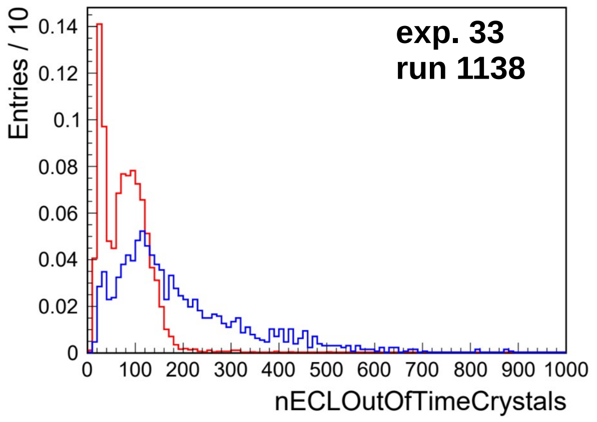
- I produced some BGOs and then some MCrd for few exp. 33 runs using the basf2 branch `feature/overlay-injection-veto-bits-in-mcrd`
  - The same branch used for developing this feature
- I first verified that BGOs have some events with passive_veto==1 and cdcecl_veto==0
  - This is independent from my feature
- I then verified that, using my branch, some signal events have passive_veto==1 and cdcecl_veto==0
  - This shows that my feature works

Dimuon events

- Comparison of MCrd, exp. 33, dimuon events privately produced:
  - In **red**: using BGOs enriched with **passive_veto==0**
  - In **blue**: using BGOs enriched with **passive_veto==1 and cdc_eclveto==0**



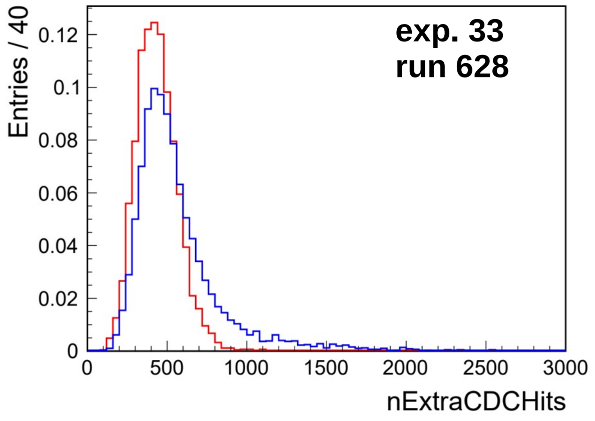


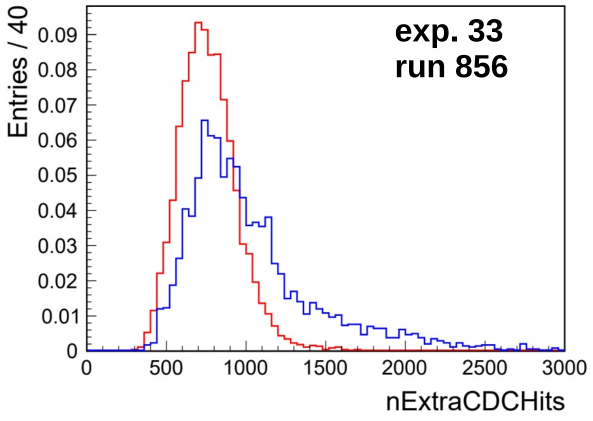


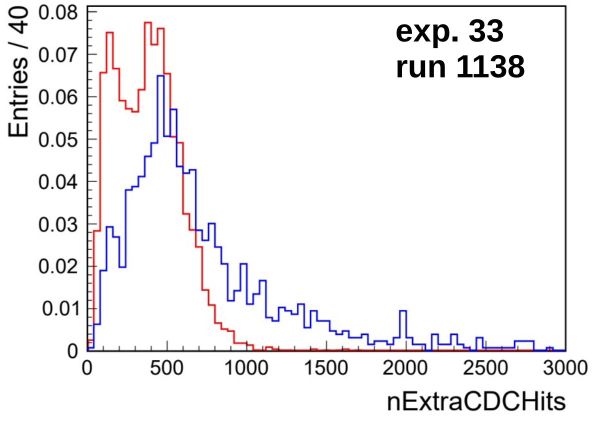
histograms normalized to 1

Dimuon events

- Comparison of MCrd, exp. 33, dimuon events privately produced:
  - In **red**: using BGOs enriched with **passive_veto==0**
  - In **blue**: using BGOs enriched with **passive_veto==1 and cdc_eclveto==0**







histograms normalized to 1

Dimuon events

- Comparison of MCrd, exp. 33, dimuon events privately produced:
  - In **red**: using BGOs enriched with **passive_veto==0**
  - In **blue**: using BGOs enriched with **passive_veto==1 and cdc_eclveto==0**
- Other distributions I checked do not show signification differences
  - Overlay of BGO reproduces the occupancy, but if, for example, the CDC gain is affected by this, it's not properly simulated

Summary

- A new basf2 module will overlay to MC run dependent the relevant input bits related to the active injection veto
  - MR to basf2: #3665
  - Target release: release-10
- Additional studies (not from me…) will be needed to see if the same features observed in data will be reproduced by MCrd
  - Including eventual performance loss
- Besides for physics analyses, this feature can be useful also for SW development (e.g. HLT speed up)
- Question for TRG group: did I overlook anything in my implementation?

Backup