Table of Contents

Stripping21 FAQ

Which version of DaVinci do I have to use?

In order to perform any test for S21 you should use the DaVinci HEAD present in the nightlies build. In order to setup correctly the environment you have to do

```
SetupProject DaVinci HEAD --nightly lhcb-no-reflex
```

The branch "lhcb-no-reflex" is important in order to use the libraries built against ROOT5 (that will be the ROOT version to be used in S21). NOTE: Since the nightlies are places where the code is developed, sometimes the latest nightly build could be corrupted. Before deciding which nightly build to use, please check in the web page that Phys, DaVinci, and Stripping slots for the lhcb-no-reflex branch have green light (also yellow can be fine). If the newest correctly built nightly is from the last Monday, then you have to setup the environment in this way:

```
SetupProject DaVinci HEAD --nightly lhcb-no-reflex Mon
```

How do I check if I need RawEvents

If you don't know if your stripping line need or not the Raw part of event, just add this code to your options for testing stripping

```
sc = StrippingConf(....) # this is the StrippingConf configured to run your line

from Configurables import  EventNodeKiller, StoreExplorerAlg, DataObjectVersionFilter, GaudiSeque

# you need first to load the raw banks
loadCalo = DataObjectVersionFilter( "LoadCaloRawEvent",
                                    DataObjectLocation = "/Event/Calo/RawEvent",
                                    MaxVersion = 999 )
loadMuon = DataObjectVersionFilter( "LoadMuonRawEvent",
                                    DataObjectLocation = "/Event/Muon/RawEvent",
                                    MaxVersion = 999 )
loadRich = DataObjectVersionFilter( "LoadRichRawEvent",
                                    DataObjectLocation = "/Event/Rich/RawEvent",
                                    MaxVersion = 999 )
loadOther = DataObjectVersionFilter( "LoadOtherRawEvent",
                                    DataObjectLocation = "/Event/Other/RawEvent",
                                    MaxVersion = 999 )
loadSeq = GaudiSequencer("LoadSeq")
loadSeq.Members += [loadOther,loadMuon,loadRich,loadCalo]

# once the raw banks are loaded you can kill them

killer = EventNodeKiller( name = "KillDAQBanks",
                                        Nodes = [ "/Event/DAQ/RawEvent",
                                                  "/Event/Calo/RawEvent",
                                                  "/Event/Muon/RawEvent",
                                                  "/Event/Rich/RawEvent",
                                                  "/Event/Other/RawEvent" ]
                                    )

# now you can run the whole thing inside the UserAlgorithms of DaVinci
# Note the order of algorithms, it is IMPORTANT

seq = GaudiSequencer("MySeq")
seq.Members += [ loadCalo,loadMuon,loadRich,loadOther,killer,sc.sequence() ]

.....
```

```
DaVinci().UserAlgorithms = [seq]
```

NOTE: this code can be used also to check if you need raw events in your final analysis. Just substitute "sc.sequence()" with any algorithm you run in your analysis.

How do I check if I need the RawEvent in my analysis?

Using the output of Stripping 20 we generated pairs 100K event DSTs with and without the RAW event (TES locations removed: "/Event/Rich/RawEvent", "/Event/Calo/RawEvent", "/Event/Muon/RawEvent", "/Event/Other/RawEvent"). You can run your ntuple maiking code on both DSTs and compare results.

Below is a list of the corresponding LFN (similar DSTs for other streams or strippings can be generated under request)

```
#RADIATIVE STREAM
       /lhcb/user/a/acontu/644/2014_06/80153/80153598/radiativeS20_WITHRAW.dst
       /lhcb/user/a/acontu/644/2014_06/80153/80153598/radiativeS20_NoRAW.dst

       #CHARMCOMPLETEEVENT STREAM
       /lhcb/user/a/acontu/648/2014_06/80153/80153692/charmcompleteeventS20_WITHRAW.dst
       /lhcb/user/a/acontu/648/2014_06/80153/80153692/charmcompleteeventS20_NoRAW.dst

       #BHADRONCOMPLETEEVENT STREAM
       /lhcb/user/a/acontu/647/2014_06/80153/80153691/bhadroncompleteeventS20_WITHRAW.dst
       /lhcb/user/a/acontu/647/2014_06/80153/80153691/bhadroncompleteeventS20_NoRAW.dst

       #EW STREAM
       /lhcb/user/a/acontu/645/2014_06/80153/80153618/ewS20_WITHRAW.dst
       /lhcb/user/a/acontu/645/2014_06/80153/80153618/ewS20_NoRAW.dst

       #SEMILEPTONIC STREAM
       /lhcb/user/a/acontu/643/2014_06/80153/80153594/semileptonicS20_WITHRAW.dst
       /lhcb/user/a/acontu/643/2014_06/80153/80153594/semileptonicS20_NoRAW.dst

       #DIMUON STREAM
       /lhcb/user/a/acontu/646/2014_06/80153/80153690/dimuonS20_WITHRAW.dst
       /lhcb/user/a/acontu/646/2014_06/80153/80153690/dimuonS20_NoRAW.dst
```

How can I ask to store raw banks in the output of my stripping line?

Thanks to Chris Jones it is now possible to specify the raw banks you want to store in the output events selected by your stripping line. This is possible simply configuring the StrippingLine Class as follow:

```
self.lineB2Charged2Body = StrippingLine( B2Charged2BodyName+"Line",
                                                 prescale  = config['PrescaleB2Charged2Body'],
                                                 selection = self.B2Charged2Body,
                                                 EnableFlavourTagging = True,
                                                 RequiredRawEvents = ["Trigger","Muon","Calo","Ri
```

In this way the Calo and Muon raw banks will be stored in the events selected by your line.

Valid RawBanks are "Trigger", "Muon", "Calo", "Rich", "Velo" and "Tracker" (names are case-sensitive).

How can I have FlavourTagging saved in microDST?

In order to run FlavourTagging and have it saved in microDST you just have to set a flag when creating your StrippingLine. For example, in StrippingB2HHBDT.py you can see

```
self.lineB2HHBDT = StrippingLine( B2HHBDTName+"Line",
                                        prescale  = config['PrescaleB2HHBDT'],
                                        selection = self.CutBDT,
                                        EnableFlavourTagging = True )
```

NOTE 1: The code works also for full DST, storing FlavourTagging objects in the same TES location of your line. NOTE 2:TupleToolTagging is able to understand if FlavourTagging object are already present in the data. In this case it will read what is already present.

Can I have re-fitted primary vertex in microDST?

In previous strippings this was automatically done, but from stripping21 onward it will be disabled. If your line run in microDST stream and you whant to have a re-fitted PV you have to specify it. . For example in StrippingB2JpsiXforBeta_s.py you can find

```
combiner = CombineParticles( DecayDescriptor = DecayDescriptor,
                                    DaughtersCuts = DaughterCuts,
                                    MotherCut = PostVertexCuts,
                                    CombinationCut = PreVertexCuts,
                                    ReFitPVs = ReFitPVs)
```

Can I store or variables in microDST?

In order to do this follow the receipt in the release.notes of StrippingConf package. As example you can look in StrippingBeauty2XGamma.py or StrippingBu2LLK.py

TWiki: StrippingIsolationTools

How can I run a MVA-based stripping line?

Sebastian Neubert developed a tool to run MVA-based stripping lines. Details can be found in this presentation

What is the MDST.DST stream? How can I have my line stored also in MDST.DST stream?

The MDST.DST stream is devoted to the use case where the line in question is going to uDST. In MDST.DST will be stored the full reconstructed event and not only the candidate selected by the stripping line. MDST.DST stream will allow fast regeneration of certain things, such as new Flavour tagging, or isolation etc., for those selected candidates, that requires the full reconstructed event information. The regeneration of microDST will have to be coordinated according with the computing team (MDST.DST will not be accessible to normal users). If you want to specify that you want also the MDST.DST for your line you have to:

```
self.lineB2HHBDT = StrippingLine( B2HHBDTName+"Line",
                                        prescale  = config['PrescaleB2HHBDT'],
                                        selection = self.CutBDT,
                                        EnableFlavourTagging = True,
                                        MDSTFlag = True )
```

Can I reduce the time consumed by my multi-body stripping line?

Vanya Beylaev developed the NBodyDecay tool. It proved to give an important gain in time consumption. Details can be found in this presentation. IMPORTANT: the use of NBodyDecay will become mandatory for multi-body lines in Stripping21.

Change the default vertex fitter in DaVinci

With OfflineVertexFitter being no longer maintained, the default vertex fitter has become LoKi:VertexFitter. Although the vertex fitter can be specified in the combine particles used in the stripping line in the instances of CombineParticles

```
#force OfflineVertexFitter usage
<YOURCOMBINEPARTICLE>.ParticleCombiners.update( { "" : "OfflineVertexFitter"} )

#force Loki:VertexFitter usage
<YOURCOMBINEPARTICLE>.ParticleCombiners.update( { "" : "LoKi:VertexFitter"} )
```

IMPORTANT: the default vertex fitter in the CommonParticles will be the LoKi one. Since CommonParticles are algorithms common for all the stripping lines they will use the default code, without the possibility for the users to change it.

However, if users want to deeply test the effects of migrating completely from one vertex fitter to the other they will have to follow this recipes:

- If you want to use the LoKi::VertexFitter as default, use the DaVinci HEAD from the nightlies. Be sure to use the ROOT5 based binaries (lhcb-no-reflex branch of the nightlies):

```
SetupProject DaVinci HEAD --nightly lhcb-no-reflex
```

- If you want to use OfflineVertexFitter as default:

```
SetupProject DaVinci HEAD --nightly lhcb-no-reflex --build-env
      getpack Phys/DaVinciKernel r178222
      SetupProject DaVinci HEAD --nightly lhcb-no-reflex
      cd Phys/DaVinciKernel/cmt
      cmt make clean; cmt make
      cd ../../..
      SetupProject DaVinci HEAD --nightly lhcb-no-reflex
      getpack Phys/DaVinciPVTools head
      cd Phys/DaVinciPVTools/cmt
      cmt make clean; cmt make
      cd ../../..
      SetupProject DaVinci HEAD --nightly lhcb-no-reflex
      getpack Phys/JetAccessories head
      cd Phys/JetAccessories/cmt
      cmt make clean; cmt make
      SetupProject DaVinci HEAD --nightly lhcb-no-reflex
```
