# Table of Contents

# How do I redo stripping?

The stripping is implemented in this package. Some of it can be modified from the DaVinci script, but other things require modifications in DaVinci. Therefore one has to checkout the right version of DaVinci, modify it and run the ntuple production with it.

# What DaVinci version to use

For:

- **Data**: Use the latest version. This is not relevant here because data is stored in *MDST* files which for which restripping is not possible.
- **Simulation:** Use the version of DaVinci used for processing the corresponding data.

The latest stripping version for the data is here. So, if you are using 2017 data and the latest stripping available is *29r2p1* you check here and find that the DaVinci version used was *v42r9p2*.

# Checking out DaVinci

Follow these instructions:

```
#disable default login
touch ~/.nogrouplogin

#set LHCb environment
source /cvmfs/lhcb.cern.ch/lib/LbEnv

#what platforms have a specific DaVinci version?
lb-sdb-query p DaVinci v42r9p2

#set platform
lb-set-platform x86_64-centos7-gcc62-opt

#pick a directory where all your DaVinci versions will be
cd /afs/cern.ch/work/a/acampove/RK/DaVinci/

#Create DaVinci skeleton
lb-dev DaVinci/v42r9p2
cd DaVinciDev_v42r9p2/

#Declare what packages you will be modifying
git lb-use Phys
git lb-use Stripping
```

each DaVinci version is associated with a specific version of the other packages. Here we need to know the version of `Phys` and `Stripping`, which can be found here.

```
#Place where PID cut on electrons are set, when building dielectron objects
git lb-checkout Phys/v23r10 Phys/CommonParticles
#Place where stripping lines are defined
git lb-checkout Stripping/v14r5p2 Phys/StrippingArchive

#build
make configure
make

#Edit code where PID is set for electrons
cd Phys/CommonParticles/python/CommonParticles/
vim StdLooseDiElectron.py
```

The change needed for `StdLooseDiElectron.py` is below:

```
51#dieLL.Electron.Selection = ["RequiresDet='CALO' CombDLL(e-pi)>'-2'"]
53dieLL.Electron.Selection = [""]
```

Then modify `StdLooseDiMuon.py` as indicated below:

```
30#StdLooseDiMuon.Inputs = ["Phys/StdAllLooseMuons/Particles"]
32StdLooseDiMuon.Inputs = ["Phys/StdAllNoPIDsMuons/Particles"]
```

Then implement changes in the file where the stripping line is defined:

```
cd Phys/StrippingArchive/python/StrippingArchive/Stripping29r2p1/StrippingRD/
vim StrippingB2LLXBDT.py
```

by changing:

```
229#from StandardParticles import StdLooseANNPions, StdLooseANNKaons, StdLooseANNProtons
```

```
231         from StandardParticles import StdAllNoPIDsPions, StdAllNoPIDsKaons, StdAllNoPIDsProt
```

and

```
232self.SelPions = self.createSubSel(
234             OutputList=self.name + "SelPions",
236             #InputList=StdLooseANNPions,
238             InputList=StdAllNoPIDsPions,
240             Cuts=config['PionCuts'])
242 
244         self.SelKaons = self.createSubSel(
246             OutputList=self.name + "SelKaons",
248             #InputList=StdLooseANNKaons,
250             InputList=StdAllNoPIDsKaons,
252             Cuts=config['KaonCuts'])
254 
256         self.SelProtons = self.createSubSel(
258             OutputList=self.name + "SelProtons",
260             #InputList=StdLooseANNProtons,
262             InputList=StdAllNoPIDsProtons,
264             Cuts=config['ProtonCuts'])
```

# Adding or modifying tools

Stripping `42r9p2` is meant to be used with 2017 data and `44r10p5` with 2016 data. However `TupleToolCorrectedMass` does not exist in the former version but it does in the latter. Therefore you need to add that tool by doing:

```
git lb-use Analysis
git checkout Analysis/v18r9 Phys/DecayTreeTuple
```

Now from gitlab you should be able to see the missing tool in `Analysis/v20r9p3` here. Download the `.cpp` and `.h` files and put them in the corresponding folder. Click on the `History` button and see the first commit here You see that not only the file with the class was added, but also the `CMakeLists.txt` was modified to link a library against the `Physics` library in ROOT. So, that modification is also needed. After that change do:

```
make configure
make
```

this should build the final version of the code.

# Modifying the DaVinci options

A working example of an option file that works with the software set up above is here. The relevant lines are:

```
# Imports the stripping builder and the configuration dictionary from the right stripping ve
from StrippingArchive.Stripping29r2p1.StrippingRD.StrippingB2LLXBDT import B2LLXBDTConf   as
from StrippingArchive.Stripping29r2p1.StrippingRD.StrippingB2LLXBDT import default_config as

#Picks the dictionary with the stripping cuts
conf= config['CONFIG']

print(conf)
print("---------------")
print("---------------")

#redefine stripping cuts
if RESTRIP and LINE == "bdt":
    conf['DiElectronCuts']="""
                                (HASVERTEX) & (VFASPF(VCHI2)<16) & (MM<5.0*GeV)
                               & (INTREE( (ID=='e+') & (PT>200*MeV) & (MIPCHI2DV(PRIMARY)>1.
                               & (INTREE( (ID=='e-') & (PT>200*MeV) & (MIPCHI2DV(PRIMARY)>1.
                            """
    conf['PionCuts'   ]    ="(PROBNNpi> -1000) & (PT>250*MeV) & (TRGHOSTPROB<0.4)"
    conf['KaonCuts'   ]    ="(PROBNNk > -1000) & (PT>300*MeV) & (TRGHOSTPROB<0.4)"
    conf['ProtonCuts']     ="(PROBNNp > -1000) & (PT>300*MeV) & (TRGHOSTPROB<0.4)"
elif RESTRIP and LINE == "cut":
    conf['PIDe']       = -1000

print(conf)
print("---------------")
print("---------------")

#Create a new stripping
from StrippingConf.Configuration import StrippingConf, StrippingStream
MyStream = StrippingStream("MyStream")

if LINE == "bdt":
    lb = builder('B2LLXBDT', conf)
else:
    lb = builder('Bu2LLK'  , conf)

#pick up the old line with redefined selection
for line in lb.lines():
    if line.name() == 'Stripping' + STR_LINE:
        print("Appending {}".format(STR_LINE))
        MyStream.appendLines([line])

# Configure Stripping
from Configurables import ProcStatusCheck
filterBadEvents = ProcStatusCheck()

sc = StrippingConf( Streams = [ MyStream ], MaxCandidates = 2000, AcceptBadEvents = False, B
```

and at the end of the file:

```
from Configurables import DaVinci, CheckPV, GaudiSequencer, LoKi__HDRFilter
#Need to kill old stripping
from Configurables import EventNodeKiller

eventNodeKiller = EventNodeKiller('Stripkiller')
eventNodeKiller.Nodes = [ '/Event/AllStreams', '/Event/Strip' ]

DaVinci()
```

```
388DaVinci().EvtMax = -1
390DaVinci().PrintFreq = 1000
392DaVinci().HistogramFile = 'ROOT.root'
394DaVinci().Simulation = True
396DaVinci().Lumi = False
398#Add killing stripping algorithm
400DaVinci().appendToMainSequence( [ eventNodeKiller ] )
402#Add new stripping
404DaVinci().appendToMainSequence( [ sc.sequence() ] )
406DaVinci().UserAlgorithms = [ tupleee ]
408DaVinci().VerboseMessages = True
410DaVinci().DataType = "2017"
```

# Producing the ntuple

After this, to test everything `cd` to the `DaVinciDev_v42r9p2` directory and:

```
./run gaudirun.py DV_B2Kee_2017.py
```

this should produce the ntuple.

-- AngelCampoverde - 2020-04-10

---

This topic: Sandbox > RestrippingTwiki
Topic revision: r2 - 2020-04-10 - AngelCampoverde