How do I run the HLT2 without PID selections?

(NB: There may be other ways to do this. Check with your supervisor or WG MC liason if you are unsure.)

(The following method was developed using `Moore/v25r4`.)

In short, you will create a new TCK that is a copy of the original TCK, modify the properties with PID requirements, then point your code to the new TCK.

You will need to create your own `config.cdb`, the file that stores the trigger configurations for Moore. Copy the default `config.cdb` (`$HLTTCKROOT/config.cdb`) to a directory where you can make changes. (NB: **Make sure your copy is still named** `config.cdb`.) Make sure you are using this file any time you want to make changes from here on out.

Modifying trigger properties

There is already a script designed to remove PID information from trigger lines, `$TCKUTILSROOT/python/TCKUtils/removePID.py`, but it is not guaranteed to be 100% successful (and is not at the time of this writing), so you will need to be careful. Copy this file to a directory where you can make changes and use it as a template for your own PID modifications.

The first change you will want to make to your copy of `removePID.py` is to adjust the final section, under `if name = "__main__":`. Change the value of `dataTCK` to whatever TCKid you want to modify. Then make sure the value for newtck is the one you want, usually `newtck=(dataTCK | 0x0c000000)`, and verify that this TCK value does not already exist using TCKsh. (NB: **Do not change the 1st digit after the x in the TCK value**; this is a special digit and your code will not work if you change it. Similarly, **do not change the final 4 digits** as they need to match your L0 TCK.) Finally, pass the `ConfigAccessSvc` constructor the `File` keyword pointing at your copy of `config.cdb`. All together, it should look something like:

```
if __name__ == "__main__":
    dataTCK = 0x6139160F
    print "Create PID-less TCK from TCK %x" % (dataTCK)
    removePID(tck=dataTCK, newtck=(dataTCK | 0x0c000000),
                        cas=ConfigAccessSvc(Mode='ReadWrite', File='./config.cdb') )
```

Next, modify the `removePID` function itself to ensure that it makes the changes you want. `removePID` finds algorithms matching the `algname` regular expression with properties matching the `property` regular expression. It then searches the values of those properties for strings matching any regular expression in `regExp` and replaces them with the string `repString`. You will almost certainly want `property='DaughtersCuts|Combination.*Cut|MotherCut|Selection|Code'`. (Note the inclusion of `|Code`.) The `removePID` function definition in your copy of `removePID.py` should look something like:

```
def removePID(tck, newtck=None,
              algname='.*', property='DaughtersCuts|Combination.*Cut|MotherCut|Selection|Code',
              regExp=['(\()( *)(PID)(e|mu|pi|K|p)( *)(&lt;|&gt;)( *)(-|)([0-9]*\.[0-9]*|[0-9]*\.|
                      '(\()( *)(\(|)( *)(PID)(e|mu|pi|K|p)( *)(-|)( *)(PID)(e|mu|pi|K|p)( *)(\)|)
                      "(\()( *)(MINTREE)( *)(\()( *')([a-z]*[A-Z]*)([\+-]*)( *')( *)(==)( *)(ABSI
              repString='( ALL )',
              dumpFile='', cas=ConfigAccessSvc() ):
```

This version of `removePID` will probably make all the changes you want. The whole `removePID.py` file should look something like:

```
'''modified version of $TCKUTILSROOT/python/TCKUtils/removePID.py
'''
import GaudiPython
from Gaudi.Configuration import*
from Configurables import ConfigStackAccessSvc, ConfigDBAccessSvc, ConfigTarFileAccessSvc, Config

from TCKUtils.utils import *

def replaceString(regExp, oldstring, repstr):

    import re
    matches = re.findall(regExp, oldstring)
    for m in matches:
        newstr = ''
        for n in m:
            newstr += n
        print "Modifying %s to %s" % (newstr, repstr)
        oldstring = oldstring.replace(newstr, repstr)

    return oldstring

def removePID(tck, newtck=None,
              algname='.*', property='DaughtersCuts|Combination.*Cut|MotherCut|Selection|Code',
              regExp=['(\()( *)(PID)(e|mu|pi|K|p)( *)(&lt;|&gt;)( *)(-|)([0-9]*\.[0-9]*|[0-9]*\.|
                      '(\()( *)(\(|)( *)(PID)(e|mu|pi|K|p)( *)(-|)( *)(PID)(e|mu|pi|K|p)( *)(\)|)
                      "(\()( *)(MINTREE)( *)(\()( *')([a-z]*[A-Z]*)([\+-]*)( *')( *)(==)( *)(ABSI
              repString='( ALL )',
              dumpFile='', cas=ConfigAccessSvc() ):

    props = getProperties(tck, algname, property)
    newprops = dict()
    for p, d in props.iteritems():
        for k, c in d.iteritems():
            newc = c
            for rexp in regExp:
                newc = replaceString(rexp, newc, repString)
            if k == 'Selection':
                print "Modifying %s to %s" % (c, '[""]')
                newc = '[""]'
            if c != newc:
                if p not in newprops:
                    newprops[p] = {}
                    newprops[p][k] = newc
                else:
                    newprops[p][k] = newc

    from pprint import pprint
    print "==================&gt; dump update &lt;===================="
    pprint(newprops)
    label = None
    for (i, j) in getConfigurations().iteritems():
        if tck in j['TCK']:
            label = j['label']
    id = updateProperties(resolveTCK(tck), newprops,
                          label='%s - Same as 0x%08x without PID cuts' % (label, tck),
                          cas=ConfigAccessSvc(Mode='ReadWrite'))
    createTCKEntries({newtck: id}, cas=ConfigAccessSvc(Mode='ReadWrite'))

if __name__ == "__main__":
    dataTCK = 0x6139160F
    print "Create PID-less TCK from TCK %x" % (dataTCK)
    removePID(tck=dataTCK, newtck=(dataTCK | 0x0c000000),
              cas=ConfigAccessSvc(Mode='ReadWrite', File='./config.cdb') )
```

You can now run the file from the command prompt with `python removePID.py | tee removePID.log` and it will update your `config.cdb`.

It is a good idea, however, to check that all the changes you wanted were actually made. To do this, you will need to use TCKsh to explore the original TCKid to find the instances of PID you wanted to remove, then look through your `removePID.log` file to ensure those changes were made. What follows is an example.

First, look at the HLT2 lines you are using with your TCK using TCKsh: `listProperties(TCKid, '.*/<HLT2linename>')`, then repeat with its Input (usually under `Inputs` or `InputSelection`), and so on until you have thoroughly explored the line. You should look for anything that refers to PID, usually in the `Code` or `Selection` property. For example, `listProperties(0x6139160F, '.*/Hlt2CharmHadLcpToPpKmPipTurboDecision')` should give you an output that looks like:

```
Requested Properties for Hlt2CharmHadLcpToPpKmPipTurboDecision
      'ExtraOutputs':[]
      'StatTableHeader': |    Counter                                      |     #     |    sum
      'MonitorHistograms':True
      'AuditExecute':True
      'HistoCountersPrint':False
      'HistoTopDir':
      'AuditReinitialize':True
      'HistoPrint':False
      'VetoObjects':[  ]
      'AuditRestart':True
      'ShortFormatFor1DHistoTable': | %1$-25.25s %2%
      'MonitorService':MonitorSvc
      'Enable':True
      'OutputSelection':'Hlt2CharmHadLcpToPpKmPipTurboDecision'
      'FullDetail':False
      'HistoSplitDir':False
      'AutoStringIDPurgeMap':{ '/' : '=SLASH=' }
      'Timeline':False
      'HistoCheckForNaN':True
      'RootInTES':
      'AuditFinalize':True
      'HistoProduce':@OnlineEnv.Monitor@False
      'AuditEndRun':True
      'ErrorsPrint':False
      'AuditBeginRun':True
      'Cardinality':1
      'ErrorMax':1
      'InputSelection':'TES:/Event/Hlt2/Hlt2CharmHadLc2HHH_LcpToKmPpPipMassFilter/Particles'
      'PropertiesPrint':False
      'StatEntityList':[  ]
      'AuditInitialize':True
      'RegularRowFormat': | %|-48.48s|%|50t||%|10d| |%|11.7g| |%|#11.5g| |%|#11.5g| |%|#12.5g| |%
      'HeaderFor1DHistoTable':|   Title                                       |    #    |     Mea
      'TypePrint':True
      'StatPrint':False
      'ExtraInputs':[]
      'HistoDescriptor':{  }
      'AuditStop':True
      'Context':HLT
      'NeededResources':[  ]
      'HistoOffSet':0
      'UseSequencialNumericAutoIDs':False
      'HistoDir':Hlt2CharmHadLcpToPpKmPipTurboDecision
      'UseEfficiencyRowFormat':True
      'OutputLevel':4
      'FormatFor1DHistoTable':| %2$-45.45s | %3$=7d |%8$11.5g | %10$-11.5g|%12$11.5g |%14$11.5g |
      'AuditStart':True
      'RegisterForContextService':True
      'AuditAlgorithms':True
      'HistogramUpdatePeriod':0
      'MinCandidates':1
      'IsClonable':False
      'RequireObjects':[  ]
      'RequirePositiveInputs':True
      'EfficiencyRowFormat': |*%|-48.48s|%|50t||%|10d| |%|11.5g| |(%|#9.6g| +- %|-#9.6g|)%%|   --
      'ErrorCounter':0
      'CounterList':[ '.*' ]
```

Looking through the output, we do not find any explicit references to PID, but we do find `'InputSelection':'TES:/Event/Hlt2/Hlt2CharmHadLc2HHH_LcpToKmPpPipMassFilter/Particles'`, so we repeat the process with `listProperties(0x6139160F, '.*/Hlt2CharmHadLc2HHH_LcpToKmPpPipMassFilter')` and find `'Inputs':[ 'Hlt2/Hlt2CharmHadLcXic2HHH_LcpToKmPpPipTisTosTagger' ]`. We continue to repeat the process, looking for PID references and checking inputs, until we find, e.g., `Hlt2CharmHadSharedDetachedLcChild_pFilter`, which has no Inputs, but does have PID selections in its `Code`:

```
>>> listProperties(0x6139160F, '.*/Hlt2CharmHadSharedDetachedLcChild_pFilter', 'Code')

Requested Properties for Hlt2CharmHadSharedDetachedLcChild_pFilter
      'Code':  (TRCHI2DOF < 3.0 )& (PT > 200.0)& (P > 1000.0)& (MIPCHI2DV(PRIMARY) > 4.0)& (PIDp
```

We repeat this process until we have explored all the Inputs and found all instances of PID information related to our trigger lines. This can be tedious; be sure to keep careful notes of what you have explored and found, and remember that exploring the GitLab repo of trigger lines may help get a handle on the underlying architecture.

Once you have done this, look through your `removePID.log` file to find, e.g.

```
'Hlt2CharmHadSharedDetachedLcChild_pFilter': {'Code': ' (TRCHI2DOF < 3.0 )& (PT > 200.0)& (P > 10
```

If the PID selection has been replaced with `( ALL )`, you know `removePID` successfully modified the property in question.

Using your new TCK

When you run Moore, you will need to tell it to use your new config.cdb and your new TCKid. Do that by including the following code snippet in your options file:

```
Moore().TCKData = '<directory containing your config.cdb>'
Moore().InitialTCK = '<your new TCKid>'
HltConfigSvc ().initialTCK = '<your new TCKid>'
```

Make sure not to include, e.g.,

```
importOptions("$APPCONFIGOPTS/Conditions/TCK-<your old TCKid>.py")
```

in your options file or to call it before the above snippet as it sets some of the same properties.

For subsequent stages, e.g., Brunel, use the following snippet in your options file:

```
from Configurables import ConfigCDBAccessSvc
ConfigCDBAccessSvc ().File = '&lt;path/to/your/config.cdb&gt;'
```

-- MichaelWilkinson - 2018-08-22

How to run from TCK?

How to check which Hlt lines fired in raw files?

-- MikaVesterinen - 2016-05-14

This topic: LHCb > HltFAQ
Topic revision: r3 - 2018-08-22 - MichaelWilkinson

Copyright &© 2008-2020 by the contributing authors. All material on this collaboration platform is the property of the contributing authors.
Ideas, requests, problems regarding TWiki? Send feedback