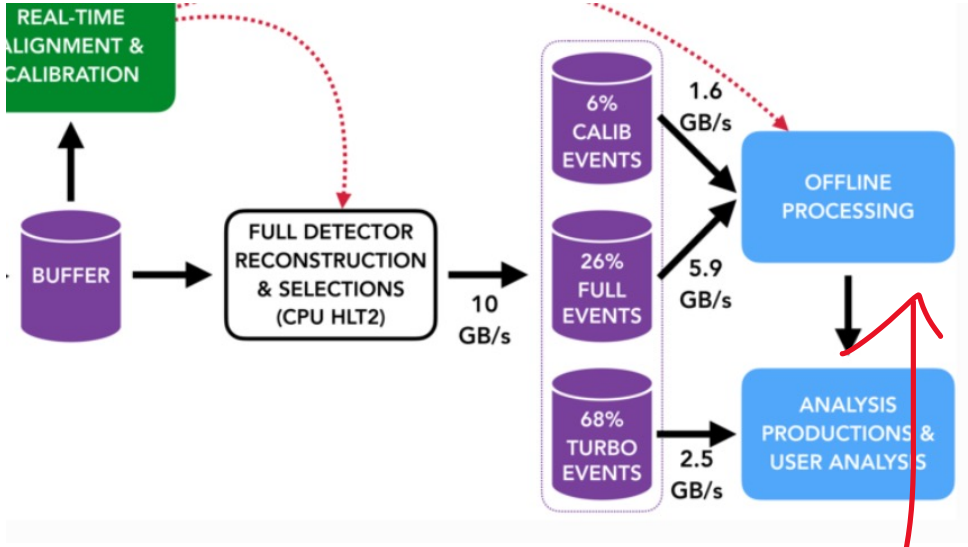
Moore Run3:
HLT2 usage + line dev

Luke Grazette

Moore and Hlt2

"HLT2, which performs a high-fidelity reconstruction and makes a decision based on the full detector read-out information."



Docs:
https://lhcbdoc.web.cern.ch/lhcbdoc/moore/master/index.html
MM Channel :
`Upgrade Hlt2` for help/advice/discussion. (SEARCH before asking!)

Sprucing, also in Moore

2500 lines

2490	Hlt2IFT_Femtoscopy_LcXiDecision
2491	Hlt2IFT_Femtoscopy_XicpLambdaDecision
2492	Hlt2IFT_Femtoscopy_XicpPDecision
2493	Hlt2IFT_Femtoscopy_XicpXiDecision
2494	Hlt2TrackEff_ZToMuMu_VeloMuon_mup_TagDecision

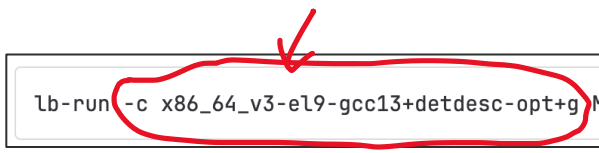
Running Moore

For purposes other than development, can use the lhcb releases mounted on CVMFS.
lb-run Moore/<v> gaudirun.py <options>

```
source /cvmfs/lhcb.cern.ch/lib/LbEnv
```

For development, see: lb-stack-setup.
Moore/run gaudirun.py <options>

Specifies appropriate binary tag.
For MC: currently requires `+detdesc` builds
For Data: dd4hep (default) builds



```
lb-run -c x86_64_v3-el9-gcc13+detdesc-opt+g Moore/v55r0 gaudirun.py options/hlt2/v0.py 2>&1 | tee hlt2_v0.log
```

Option Anatomy

[gitlab repo]

Option Anatomy v0: [gitlab repo]

v0.py 1

options > hlt2 > v0.py

```python
from Moore import options, run_moore

run_moore(options)

```

```
source /cvmfs/lhcb.cern.ch/lib/LbEnv
```

```
lb-run -c x86_64_v3-el9-gcc13+detdesc-opt+g Moore/v55r0 gaudirun.py options/hlt2/v0.py 2>&1 | tee hlt2_v0.log
```

Option Anatomy v0: [gitlab repo]

For computers using operating systems that are not comparible with el9, lb-run will try to run in a container instead.

comparison_logs > hlt2_v0.log

```
WARNING:lb-run:Decided best platform to use is x86_64_v3-el9-gcc13-opt+g
WARNING:lb-run:Decided best container to use is None
# setting LC_ALL to "C"
# --> Including file '/storage/epp2/phrhbm/lhcbsk_feb24/options/hlt2/v0.py'
Traceback (most recent call last):
  File "/cvmfs/lhcb.cern.ch/lib/lhcb/GAUDI/GAUDI_v37r2/InstallArea/x86_64_v3-el9-gcc13-opt+g/bin/gaudirun.py", line 586, in <module>
    exec(o, g, l)
  File "<string>", line 1, in <module>
  File "/cvmfs/lhcb.cern.ch/lib/lhcb/GAUDI/GAUDI_v37r2/InstallArea/x86_64_v3-el9-gcc13-opt+g/bin/gaudirun.py", line 545, in __call__
    importOptions(arg)
  File "/cvmfs/lhcb.cern.ch/lib/lhcb/GAUDI/GAUDI_v37r2/InstallArea/x86_64_v3-el9-gcc13-opt+g/python/GaudiKernel/ProcessJobOptions.py", line 552, in importOptions
    _import_function_mapping[ext](optsfile)
  File "/cvmfs/lhcb.cern.ch/lib/lhcb/GAUDI/GAUDI_v37r2/InstallArea/x86_64_v3-el9-gcc13-opt+g/python/GaudiKernel/ProcessJobOptions.py", line 486, in _import_python
    exec(code, {"__file__": file})
  File "/storage/epp2/phrhbm/lhcbsk_feb24/options/hlt2/v0.py", line 3, in <module>
    run_moore(options)
  File "/cvmfs/lhcb.cern.ch/lib/lhcb/MOORE/MOORE_v55r0/InstallArea/x86_64_v3-el9-gcc13-opt+g/python/Moore/config.py", line 261, in run_moore
    config = configure_input(options)
  File "/cvmfs/lhcb.cern.ch/lib/lhcb/LHCB/LHCB_v55r0/InstallArea/x86_64_v3-el9-gcc13-opt+g/python/PyConf/application.py", line 802, in configure_input
    options.finalize()
  File "/cvmfs/lhcb.cern.ch/lib/lhcb/LHCB/LHCB_v55r0/InstallArea/x86_64_v3-el9-gcc13-opt+g/python/PyConf/application.py", line 481, in finalize
    self._validate()
  File "/cvmfs/lhcb.cern.ch/lib/lhcb/LHCB/LHCB_v55r0/InstallArea/x86_64_v3-el9-gcc13-opt+g/python/PyConf/application.py", line 405, in _validate
    raise ConfigurationError(
PyConf.utilities.ConfigurationError: Required option simulation must be set to True or False as appropriate

```

We are trying to configure without input_file information, a reasonable complaint from Moore

Option Anatomy v1: [gitlab repo]

PRConfig: DataBase of TestFiles

v1.py 1 ×

options > hlt2 > v1.py

```python
from Moore import options, run_moore

options.set_input_and_conds_from_testfiledb("exp_24_minbias_Sim10c_magdown")

options.evt_max = 100

options.output_type = 'MDF'
options.output_file = "hlt2_v1_output.mdf"
options.output_manifest_file = "hlt2_v1_output.tck.json"

run_moore(options)
```

Moore docs:
samples with different attributes

type = 'ROOT'
for *.dst*, *.root*, *.digi*
type = 'MDF'
for *.mdf, .mep, .raw*

Stores encoding keys for the specific triggers you used.
This is to prevent storing expensive strings and instead store digits.
`3071: "/Event/Hlt2B2OC_BdToDmPi_DmToPimPimKp/Particles/" `
Without encoding keys being made **globally** available or locally passed between jobs, your output is **useless.**

```
source /cvmfs/lhcb.cern.ch/lib/LbEnv
```

```
lb-run -c x86_64_v3-el9-gcc13+detdesc-opt+g Moore/v55r0 gaudirun.py options/hlt2/v0.py 2>&1 | tee hlt2_v0.log
```

Option Anatomy v1_alt: [gitlab repo]

```python
from Moore import options, run_moore

# options.set_input_and_conds_from_testfiledb("exp_24_minbias_Sim10c_magdown")
options.input_files = [
    "root://gridproxy@eoslhcb.cern.ch//eos/lhcb/grid/prod/lhcb/MC/Dev/DIGI/00204940/0000/00204940_00001476_1.digi",
    "root://gridproxy@eoslhcb.cern.ch//eos/lhcb/grid/prod/lhcb/MC/Dev/DIGI/00204940/0000/00204940_00001481_1.digi",
    "root://gridproxy@eoslhcb.cern.ch//eos/lhcb/grid/prod/lhcb/MC/Dev/DIGI/00204940/0000/00204940_00003710_1.digi",
    "root://gridproxy@eoslhcb.cern.ch//eos/lhcb/grid/prod/lhcb/MC/Dev/DIGI/00204940/0000/00204940_00003721_1.digi",
    "root://gridproxy@eoslhcb.cern.ch//eos/lhcb/grid/prod/lhcb/MC/Dev/DIGI/00204940/0000/00204940_00005093_1.digi",
    "root://gridproxy@eoslhcb.cern.ch//eos/lhcb/grid/prod/lhcb/MC/Dev/DIGI/00204940/0000/00204940_00005160_1.digi",
    "root://gridproxy@eoslhcb.cern.ch//eos/lhcb/grid/prod/lhcb/MC/Dev/DIGI/00204940/0000/00204940_00005219_1.digi",
    "root://gridproxy@eoslhcb.cern.ch//eos/lhcb/grid/prod/lhcb/MC/Dev/DIGI/00204940/0000/00204940_00006206_1.digi",
    "root://gridproxy@eoslhcb.cern.ch//eos/lhcb/grid/prod/lhcb/MC/Dev/DIGI/00204940/0000/00204940_00006666_1.digi",
    "root://gridproxy@eoslhcb.cern.ch//eos/lhcb/grid/prod/lhcb/MC/Dev/DIGI/00204940/0000/00204940_00006855_1.digi",
    "root://gridproxy@eoslhcb.cern.ch//eos/lhcb/grid/prod/lhcb/MC/Dev/DIGI/00204940/0000/00204940_00006995_1.digi",
    "root://gridproxy@eoslhcb.cern.ch//eos/lhcb/grid/prod/lhcb/MC/Dev/DIGI/00204940/0000/00204940_00007281_1.digi",
    "root://gridproxy@eoslhcb.cern.ch//eos/lhcb/grid/prod/lhcb/MC/Dev/DIGI/00204940/0000/00204940_00007962_1.digi",
    "root://gridproxy@eoslhcb.cern.ch//eos/lhcb/grid/prod/lhcb/MC/Dev/DIGI/00204940/0000/00204940_00008009_1.digi",
    "root://gridproxy@eoslhcb.cern.ch//eos/lhcb/grid/prod/lhcb/MC/Dev/DIGI/00204940/0000/00204940_00008538_1.digi",
    "root://gridproxy@eoslhcb.cern.ch//eos/lhcb/grid/prod/lhcb/MC/Dev/DIGI/00204940/0000/00204940_00009297_1.digi",
    "root://gridproxy@eoslhcb.cern.ch//eos/lhcb/grid/prod/lhcb/MC/Dev/DIGI/00204940/0000/00204940_00009449_1.digi",
]
options.simulation = True
options.input_type = "ROOT"
options.input_raw_format = 0.5 # https://lhcbdoc.web.cern.ch/lhcbdoc/moore/master/tutorials/different_samples.html#input-samples-with-di
options.data_type = "Upgrade"
options.dddb_tag = "dddb-20231017"
options.conddb_tag = "sim-20231017-vc-md100"
```

Option Anatomy v1 : example decoding key:

[gitlab repo]

hlt2_v2_alt_1e4_output.tck.json

output > {} hlt2_v2_alt_1e4_output.tck.json > ...


```
"0x37470d61": {
    "Hlt2SelectionID": {
        "1": "Hlt2QEE_DiMuonNoIP_massRange1Decision",
        "2": "Hlt2QEE_DiMuonNoIP_massRange1_ssDecision",
        "3": "Hlt2QEE_DiMuonNoIP_massRange2Decision",
        "4": "Hlt2QEE_DiMuonNoIP_massRange2_ssDecision",
        "5": "Hlt2QEE_DiMuonNoIP_massRange3Decision",
        "6": "Hlt2QEE_DiMuonNoIP_massRange3_ssDecision",
        "7": "Hlt2QEE_DiMuonNoIP_massRange4Decision",
        "8": "Hlt2QEE_DiMuonNoIP_massRange4_ssDecision",
        "9": "Hlt2QEE_DiMuonNoIP_massRange5Decision",
        "10": "Hlt2QEE_DiMuonNoIP_massRange5_ssDecision",
        "11": "Hlt2QEE_DiMuonNoIP_massRange6Decision",
        "12": "Hlt2QEE_DiMuonNoIP_massRange6_ssDecision",
        "13": "Hlt2QEE_DiMuonNoIP_prescaledAllMassDecision",
        "14": "Hlt2QEE_DiMuonNoIP_prescaledFullEventDecision",
        "15": "Hlt2QEE_DiMuonNoIP_prescaledFullEvent_ssDecision",
        "16": "Hlt2QEE_DisplacedDiMuonDecision",
        "17": "Hlt2QEE_DisplacedDiMuon_ssDecision"
    },
    "version": "0"
},
"0xd1b25a47": {
    "PackedObjectLocations": {
        "1": "/Event/HLT2/Hlt2QEE_DiMuonNoIP_massRange1/Particle2VertexRelations",
        "2": "/Event/HLT2/Hlt2QEE_DiMuonNoIP_massRange1/Particles",
        "3": "/Event/HLT2/Hlt2QEE_DiMuonNoIP_massRange1_ss/Particle2VertexRelations",
        "4": "/Event/HLT2/Hlt2QEE_DiMuonNoIP_massRange1_ss/Particles",
        "5": "/Event/HLT2/Hlt2QEE_DiMuonNoIP_massRange2/Particle2VertexRelations",
        "6": "/Event/HLT2/Hlt2QEE_DiMuonNoIP_massRange2/Particles",
```


Local output manifest file

```
|-write_decoding_keys_to_git                = True  (default: True)
\----- (End of User ApplicationOptions/ApplicationOptions) -------
# adding key 37470d61 to registry
# adding key d1b25a47 to registry
```

Metainfo Repo: 48d5b9ef


```
{"Hlt2SelectionID": {"1": "Hlt2B2CC_BToJpsiEtaGGDecision", "2": "Hlt2B2CC_BdToJpsiKshort_JpsiToEE_DD_DetachedDecision", "3": "Hlt2B2CC_BdToJpsiKshort_JpsiToEE_DD_PromptDecision",
"4": "Hlt2B2CC_BdToJpsiKshort_JpsiToEE_LL_DetachedDecision", "5": "Hlt2B2CC_BdToJpsiKshort_JpsiToEE_LL_PromptDecision", "6":
"Hlt2B2CC_BdToJpsiKshort_JpsiToMuMu_DD_DetachedDecision", "7": "Hlt2B2CC_BdToJpsiKshort_JpsiToMuMu_DD_PromptDecision", "8": "Hlt2B2CC_BdToJpsiKshort_JpsiToMuMu_LD_DetachedDecision",
"9": "Hlt2B2CC_BdToJpsiKshort_JpsiToMuMu_LD_PromptDecision", "10": "Hlt2B2CC_BdToJpsiKshort_JpsiToMuMu_LL_DetachedDecision", "11":
"Hlt2B2CC_BdToJpsiKshort_JpsiToMuMu_LL_PromptDecision", "12": "Hlt2B2CC_BdToJpsiKshort_JpsiToMuMu_UL_DetachedDecision", "13": "Hlt2B2CC_BdToJpsiKshort_JpsiToMuMu_UL_PromptDecision",
"14": "Hlt2B2CC_BdToJpsiKstar_JpsiToEEDecision", "15": "Hlt2B2CC_BdToJpsiKstar_JpsiToEE_DetachedDecision", "16": "Hlt2B2CC_BdToJpsiKstar_JpsiToMuMu_DetachedDecision", "17":
"Hlt2B2CC_BdToJpsiKstar_JpsiToMuMu_PromptDecision", "18": "Hlt2B2CC_BdToPsitwosKshort_PsitwosToMuMu_DD_DetachedDecision", "19":
"Hlt2B2CC_BdToPsitwosKshort_PsitwosToMuMu_DD_PromptDecision", "20": "Hlt2B2CC_BdToPsitwosKshort_PsitwosToMuMu_LL_DetachedDecision", "21":
"Hlt2B2CC_BdToPsitwosKshort_PsitwosToMuMu_LL_PromptDecision", "22": "Hlt2B2CC_BsToJpsiKstarDecision", "23": "Hlt2B2CC_BsToJpsiPhiDecision", "24":
"Hlt2B2CC_BsToJpsiPhi_DetachedDecision", "25": "Hlt2B2CC_BsToJpsiPhi_ExtraMuonDecision", "26": "Hlt2B2CC_BsToJpsiPhi_JpsiToEEDecision", "27":
"Hlt2B2CC_BsToJpsiPhi_JpsiToEE_DetachedDecision", "28": "Hlt2B2CC_BsToJpsif0Decision", "29": "Hlt2B2CC_BsToJpsif0KaonDecision", "30": "Hlt2B2CC_BsToJpsif0PrescaledDecision", "31":
"Hlt2B2CC_BsToJpsif0wsDecision", "32": "Hlt2B2CC_BuToChicKPlusDecision", "33": "Hlt2B2CC_BuToJpsiKplus_JpsiToEE_DetachedDecision", "34":
"Hlt2B2CC_BuToJpsiKplus_JpsiToMuMu_DetachedDecision", "35": "Hlt2B2CC_BuToJpsiKplus_JpsiToMuMu_PromptDecision", "36": "Hlt2B2CC_BuToJpsiKstPlusDecision", "37":
"Hlt2B2CC_LbToJpsipHDecision", "38": "Hlt2B2OC_BcToD0Dp_D0ToHHHH_DpToHHHDecision", "39": "Hlt2B2OC_BcToD0Dp_D0ToHH_DpToHHHDecision", "40":
"Hlt2B2OC_BcToD0Dp_D0ToKsLLHH_DpToHHHDecision", "41": "Hlt2B2OC_BcToD0Dsp_D0ToHHHH_DspToHHHDecision", "42": "Hlt2B2OC_BcToD0Dsp_D0ToHH_DspToHHHDecision", "43":
"Hlt2B2OC_BcToD0Dsp_D0ToKsLLHH_DspToHHHDecision", "44": "Hlt2B2OC_BcToDstD0_DstToD0Pi_D0ToHHHH_D0ToHHDecision", "45": "Hlt2B2OC_BcToDstD0_DstToD0Pi_D0ToHHHH_D0ToHHHHDecision", "46":
"Hlt2B2OC_BcToDstD0_DstToD0Pi_D0ToHHHH_D0ToKsLLHHDecision", "47": "Hlt2B2OC_BcToDstD0_DstToD0Pi_D0ToHH_D0ToHHDecision", "48": "Hlt2B2OC_BcToDstD0_DstToD0Pi_D0ToHH_D0ToHHHHDecision",
"49": "Hlt2B2OC_BcToDstD0_DstToD0Pi_D0ToHH_D0ToKsLLHHDecision", "50": "Hlt2B2OC_BdToD0D0KsLL_D0ToKPiOrKPiPiPiDecision", "51": "Hlt2B2OC_BdToD0D0Kst_D0ToHHOrHHHHDecision", "52":
"Hlt2B2OC_BdToD0KKWS_D0ToHHDecision", "53": "Hlt2B2OC_BdToD0KKWS_D0ToHHHHDecision", "54": "Hlt2B2OC_BdToD0KKWS_D0ToKsLLHHDecision", "55": "Hlt2B2OC_BdToD0KK_D0ToHHDecision", "56":
"Hlt2B2OC_BdToD0KK_D0ToHHHHDecision", "57": "Hlt2B2OC_BdToD0KK_D0ToKsLLHHDecision", "58": "Hlt2B2OC_BdToD0KPiWS_D0ToHHDecision", "59": "Hlt2B2OC_BdToD0KPiWS_D0ToHHHHDecision", "60":
"Hlt2B2OC_BdToD0KPiWS_D0ToKsLLHHDecision", "61": "Hlt2B2OC_BdToD0KPi_D0ToHHDecision", "62": "Hlt2B2OC_BdToD0KPi_D0ToHHHHDecision", "63": "Hlt2B2OC_BdToD0KPi_D0ToKsLLHHDecision",
"64": "Hlt2B2OC_BdToD0PbarPWS_D0ToHHHHDecision", "65": "Hlt2B2OC_BdToD0PbarP_D0ToHHHHDecision", "66": "Hlt2B2OC_BdToD0PiPiWS_D0ToHHDecision", "67":
"Hlt2B2OC_BdToD0PiPiWS_D0ToHHHHDecision", "68": "Hlt2B2OC_BdToD0PiPiWS_D0ToKsLLHHDecision", "69": "Hlt2B2OC_BdToD0PiPi_D0ToHHDecision", "70": "Hlt2B2OC_BdToD0PiPi_D0ToHHHHDecision",
"71": "Hlt2B2OC_BdToD0PiPi_D0ToKsLLHHDecision", "72": "Hlt2B2OC_BdToDmK_DmToPimPimKpDecision", "73": "Hlt2B2OC_BdToDmPi_DmToPimPimKpDecision", "74":
"Hlt2B2OC_BdToDpDmKsLL_DpToHHHDecision", "75": "Hlt2B2OC_BdToDpDmKst_DpToHHHDecision", "76": "Hlt2B2OC_BdToDpDmPhi_DpToHHHDecision", "77":
"Hlt2B2OC_BdToDsmKPi_DsmToKmKpPimDecision", "78": "Hlt2B2OC_BdToDsmK_DsmToKpKmPimDecision", "79": "Hlt2B2OC_BdToDsmK_DsmToKpKmPim_LTUDecision", "80":
"Hlt2B2OC_BdToDsmPi_DsmToKpKmPimDecision", "81": "Hlt2B2OC_BdToDspDmRho0_DspToKpKmPip_DmToHHHDecision", "82": "Hlt2B2OC_BdToDspDsmPhi_DspToHHHDecision", "83":
"Hlt2B2OC_BdToDstD0K_DstToD0Pi_D0ToHH_D0ToHHDecision", "84": "Hlt2B2OC_BdToDstD0K_DstToD0Pi_D0ToKPiOrKPiPiPi_D0ToKPiOrKPiPiPiDecision", "85":
"Hlt2B2OC_BdToDstD0Pi_DstToD0Pi_D0ToKPiOrKPiPiPi_D0ToKPiOrKPiPiPiDecision", "86": "Hlt2B2OC_BdToDstDmKSLL_DstToD0Pi_D0ToKPiOrKPiPiPi_DmToHHHDecision", "87":
"Hlt2B2OC_BdToDstDsPhi_DstToD0Pi_D0ToKPiOrKPiPiPi_DsToHHHDecision", "88": "Hlt2B2OC_BdToDstmDpKst_DstmToD0Pi_D0ToKPiOrKPiPiPi_DpToHHHDecision", "89":
"Hlt2B2OC_BdToDstmDspRho0_DstmToD0Pi_D0ToKPiOrKPiPiPi_DspToKpKmPipDecision", "90": "Hlt2B2OC_BdToDstmMuNu_DstmToD0Pi_D0ToHHHHDecision", "91":
"Hlt2B2OC_BdToDstmMuNu_DstmToD0Pi_D0ToKsLLHHDecision", "92": "Hlt2B2OC_BdToDstpDmKst_DstpToD0Pi_D0ToKPiOrKPiPiPi_DmToHHHDecision", "93":
"Hlt2B2OC_BdToDstpDstmKsLL_DstpToD0Pi_D0ToKPiorKPiPiPiDecision", "94": "Hlt2B2OC_BdToDstpDstmPhi_DstpToD0Pi_D0ToKPiOrKPiPiPiDecision", "95":
"Hlt2B2OC_BdToScmmLcpK_ScmmToLcmPi_LcmToPKPi_LcpToPKPiDecision", "96": "Hlt2B2OC_BdToXic0PK_Xic0ToPKKPiDecision", "97": "Hlt2B2OC_BdToXic0PPi_Xic0ToPKKPiDecision", "98":
"Hlt2B2OC_BuToD0DpKsLL_D0ToKPiOrKPiPiPi_DpToHHHDecision", "99": "Hlt2B2OC_BuToD0Dp_D0ToHHHH_DpToHHHDecision", "100": "Hlt2B2OC_BuToD0Dp_D0ToHH_DpToHHHDecision", "101":
"Hlt2B2OC_BuToD0Dsp_D0ToHHHH_DspToHHHDecision", "102": "Hlt2B2OC_BuToD0Dsp_D0ToHH_DspToHHHDecision", "103": "Hlt2B2OC_BuToD0K_D0ToHHDecision", "104":
"Hlt2B2OC_BuToD0K_D0ToHHHHDecision", "105": "Hlt2B2OC_BuToD0K_D0ToHHHHWSDecision", "106": "Hlt2B2OC_BuToD0K_D0ToHHWSDecision", "107": "Hlt2B2OC_BuToD0K_D0ToKsLLHHDecision", "108":
"Hlt2B2OC_BuToD0K_D0ToKsLLHHPi0ResolvedDecision", "109": "Hlt2B2OC_BuToD0K_D0ToKsLLHHWSDecision", "110": "Hlt2B2OC_BuToD0Pi_D0ToHHDecision", "111":
"Hlt2B2OC_BuToD0Pi_D0ToHHHHDecision", "112": "Hlt2B2OC_BuToD0Pi_D0ToHHHHWSDecision", "113": "Hlt2B2OC_BuToD0Pi_D0ToHHWSDecision", "114": "Hlt2B2OC_BuToD0Pi_D0ToKsLLHHDecision",
"115": "Hlt2B2OC_BuToD0Pi_D0ToKsLLHHPi0ResolvedDecision", "116": "Hlt2B2OC_BuToD0Pi_D0ToKsLLHHWSDecision", "117": "Hlt2B2OC_BuToDmKK_DmToHHHDecision", "118":
"Hlt2B2OC_BuToDmPiPi_DmToHHHDecision", "119": "Hlt2B2OC_BuToDpDmK_DpToHHHDecision", "120": "Hlt2B2OC_BuToDpDmPi_DpToHHHDecision", "121": "Hlt2B2OC_BuToDpPiPi_DpToHHHDecision",
"122": "Hlt2B2OC_BuToDsD0Rho0_DsToKKPi_D0ToKPiDecision", "123": "Hlt2B2OC_BuToDsD0Rho0_DsToKKPi_D0ToKPiPiPiDecision", "124": "Hlt2B2OC_BuToDsmKK_DsmToHHHDecision", "125":
"Hlt2B2OC_BuToDsmPiPi_DsmToHHHDecision", "126": "Hlt2B2OC_BuToDstD0_DstToD0Pi_D0ToHHHH_D0ToHHDecision", "127": "Hlt2B2OC_BuToDstD0_DstToD0Pi_D0ToHHHH_D0ToHHHHDecision", "128":
"Hlt2B2OC_BuToDstD0_DstToD0Pi_D0ToHH_D0ToHHDecision", "129": "Hlt2B2OC_BuToDstD0_DstToD0Pi_D0ToHH_D0ToHHHHDecision", "130":
"Hlt2B2OC_BuToDstDK_DstToD0Pi_D0ToKPiOrKPiPiPi_DToHHHDecision", "131": "Hlt2B2OC_BuToDstDPi_DstToD0Pi_D0ToKPiOrKPiPiPi_DToHHHDecision", "132":
"Hlt2B2OC_BuToDstmDspPi_DstmToD0Pi_D0ToKPiOrKPiPiPi_DspToHHHDecision", "133": "Hlt2B2OC_BuToDstmKK_DstmToD0Pi_D0ToHHDecision", "134":
"Hlt2B2OC_BuToDstmKPi_DstmToD0Pi_D0ToHHDecision", "135": "Hlt2B2OC_BuToDstmPiPi_DstmToD0Pi_D0ToHHDecision", "136":
```

Option Anatomy v2: [gitlab repo]

```python
from Moore import options, run_moore

options.set_input_and_conds_from_testfiledb("exp_24_minbias_Sim10c_magdown")

options.evt_max = 100

options.output_type = 'MDF'
options.output_file = "hlt2_v2_output.mdf"
options.output_manifest_file = "hlt2_v2_output.tck.json"

from Hlt2Conf.lines.qee.dimuon_no_ip import dimuonnoip_massrange3_line


def make_lines():
    return [dimuonnoip_massrange3_line()]


from RecoConf.reconstruction_objects import reconstruction
with reconstruction.bind(from_file=False):
    run_moore(options, make_lines)
```

```
source /cvmfs/lhcb.cern.ch/lib/LbEnv
```

```
lb-run -c x86_64_v3-el9-gcc13+detdesc-opt+g Moore/v55r0 gaudirun.py options/hlt2/v0.py 2>&1 | tee hlt2_v0.log
```

This imports a specific line, and pases it to the run_moore job.

Solving the previous error.

As we want to perform reconstruction ourselves, we communicate this to the Moore job via `*binding*` the reconstruction configurable.

Sprucing would want `*from_file=True*` as it uses Hlt2 reconstructed objects.

Option Anatomy v2_alt: [gitlab repo]

```python
from Moore import options, run_moore
from RecoConf.reconstruction_objects import reconstruction
from Hlt2Conf.lines.qee.dimuon_no_ip import turbo_lines, full_lines

options.set_input_and_conds_from_testfiledb("exp_24_minbias_Sim10c_magdown")
options.evt_max = 1e4
options.output_type = 'MDF'
options.output_file = "hlt2_v2_alt_output.mdf"
options.output_manifest_file = "hlt2_v2_alt_output.tck.json"


def make_lines():
    return [
        builder() for line_dict in [turbo_lines, full_lines]
        for builder in line_dict.values()
    ]


with reconstruction.bind(from_file=False):
    run_moore(options, make_lines)
```

Now want to run over several groups of lines.

Just some list comprehension making `*[line1(), line2(), line3()]*` for all lines.

Break?

Line Development and Functors

[gitlab repo]

Parameters of Interest

Physics use-case -> mode/topology:
e.g. precision electroweak analyses in Run3 -> high pt single muon and dimuon triggers

Rate [kHz]:
*On modern Hlt1 minbias, (# pass * input_rate (~1MHz) / #events)*

Bandwidth [MB/s]:
*On modern Hlt1 minbias, Rate * avg. Event Size. ~ Rate * filesize / #events*

Purity [%]:
Rate estimated from cross-section and luminosity / Rate of line acceptance

Signal Efficiency [%]:
On modern Hlt1 signal MC,
Retention (#pass/#events) or `CanRecoChildren` efficiency or …
(#pass / # events with long-charged children particles within LHCb Acceptance)

Instructions for evaluating/tuning lines

Two Snippets:
HLT2 bandwidth testing example
Sprucing Bandwidth Example

Two RTADPA BW Presentations:
https://indico.cern.ch/event/1365217/ - 3-status-of-the-sprucing-bandw

https://indico.cern.ch/event/1365216/#6-hlt2-ratebandwidth-testing-g

Existing lines (and therefore many many examples per WG of how to do things) exist in
`*Moore/Hlt/Hlt2Conf/python/Hlt2Conf/lines/*`

Backup

Sprucing?

Hlt2 and Sprucing both occupy `*Moore*` and share many similarities in how to configure an options file, how to write a line, how to evaluate key parameters etc…

If you can run Hlt2, looking at existing (continuously tested) options should be enough to run Spruce...🤞

https://gitlab.cern.ch/lhcb/Moore/-/blob/master/Hlt/Hlt2Conf/options/sprucing/spruce_example_fromfile.py?ref_type=heads