Environment setting in KEKCC

Rev. 2023.06.01
2020.09.06

The new KEKCC is CentOS Linux 7.7.1908. The default compiler for CentOS 7 is gcc 4.8.5. This file shows the environment setting in the KEKCC using gcc 8.3.0 which is the default compiler for RHEL 8.

module

The following modules are available. Load gcc/830, git/2260, and python/3.7.

```
$ module av

-------------------- /opt/Modules/modulefiles ---------------------
aocl/aocl-2.1         intel32/2020          openmpi/2.1.6-gcc750
binutil/234           openmpi/1.10.7-gcc    openmpi/2.1.6-gcc830
gcc/750               openmpi/1.10.7-gcc750 openmpi/2.1.6-gcc930
gcc/830               openmpi/1.10.7-gcc830 openmpi/2.1.6-intel
gcc/930               openmpi/1.10.7-gcc930 openmpi/2.1.6-pgi
git/2260              openmpi/1.10.7-intel  pgi/20.1(default)
gsl/26                openmpi/1.10.7-pgi    pgi/2020
intel/2020            openmpi/2.1.6-gcc     python/3.7
```

/sw/packages

You can still use the software installed in /sw/packages. However, these libraries may not work if the compiler is not the same. For example PyROOT doesn't work.

package	gcc version
/sw/packages/root/5.34.38	4.8.5
/sw/packages/root/6.22.02	4.8.5
/sw/packages/geant4/9.6—10.5	4.8.5

/sw/packages/geant4/10.6—	8.3.0

Since there is no root library compiled with gcc 8.3.0, the compiled one is placed on the group disk, /group/had/sks/software/root/6.22.02 with C++11, /group/had/sks/software/root/6.22.08 with C++17.

If you really want to use the old version of root or geant4, use gcc 4.8.5 or install it yourself.

unpacker

The unpacker compiled with gcc 8.3.0 is also placed in the group directory, /group/had/sks/software/unpacker/unpacker.gcc830. This is updated constantly.

See below for local installation.

```
$ git clone ssh://sks@www-online.kek.jp:8022/~/public_html/git/unpacker.git
$ cd unpacker/src
$ cp Makefile.org Makefile
$ make
```

Check if the `unpacker-config` command is available.

```
$ unpacker-config --version
2020-01-21
```

Environment variables

The following is an example of environment setting in .bashrc.

```
module load gcc/830
module load git/2260
. /opt/python-3.7/etc/profile.d/conda.sh
. /group/had/sks/software/root/6.22.08/bin/thisroot.sh
. /sw/packages/geant4/11.0.2/bin/geant4.sh
. /sw/packages/geant4/11.0.2/share/Geant4-11.0.2/geant4make/geant4make.sh
```

```
export G4WORKDIR=$HOME/work/geant4 # set as you like
export PATH=/group/had/sks/software/unpacker/s2s/bin:$PATH
export PATH=$G4WORKDIR/bin/Linux-g++:$PATH
```

Anaconda setting

To use Python, it is necessary to build the Anaconda local environment once using the `conda` command as follows. Note that it is recommended to use `conda install` instead of `pip install` in the anaconda environment.

```
$ conda create -n py37 python=3.7 # py37 is an example name
$ conda activate py37
$ conda install numpy psutil pyyaml rich
```

Add the following line in .bashrc to activate your environment.

```
conda activate py37
```

If the prompt header of conda is annoying, add the following line in .condarc.

```
changeps1: False
```

K1.8 analyzer

Install the K1.8 analyzer as follows.

```
$ git clone \
ssh://sks@www-online.kek.jp:8022/~/public_html/git/k18-analyzer.git
$ cd k18-analyzer
$ git checkout e70 # choose branch as you like
$ cp Makefile.org Makefile
$ make
```

See README for more information on how to use the analyzer.