SBMS: SCons for Hall-D Offline

David Lawrence JLab

Nov. 13, 2013

Build System Features

- Centralized configurations
  - *Don't want every Makefile/SConscript file to require update in order to add new compiler argument or feature*
  - *E.g. Compiler flags and libraries needed for ROOT are defined in only one file. Packages using ROOT include this file*
- Build-all-source-in -directory model
  - *For all but special cases, all source in a directory is compiled. This helps keep Makefiles/SConscript files simpler and source tree cleaner*
- Executable and library names based on directory names (mostly)
  - *The source for an program resides in a directory whose name is the filename of the program. Makes it easier to find source for a given program*
- Cleaning in a subdirectory cleans it and its subdirectories
  - *Cleaning affects all descendants but not ancestors*

Make vs. SCons

make

- Prescriptive
- Capable of Parallel Builds
- Higher learning curve for complex build systems

scons

- Prescriptive+procedural*
- Capable of Parallel Builds
- Low learning curve**

* Scons ***is*** made of Python scripts, but the scripts are primarily used to set up the dependency tree and build parameters. If you're writing a lot of Python code, you are probably not using Scons correctly.

** Simple builds are trivial in both make and scons. Being Python, scons is a little friendlier if you start to go off the trail, but still has a high learning curve for doing non-standard things within the system

Why switch?

- BMS is an implementation of “make” that used recursive builds (i.e. re-invoked *make* in subdirectories)
- This meant *make* never had the full dependency tree and so fails if the parallel build procedure is activated
- Complete builds take 15-30 minutes on a modern machine*

**This may be slower on some new machines with many cores, but slower clock speeds*

Common Makefile/SConscript files

Build a Library

Makefile 〉 No Selection

```
1
2  PACKAGES = JANA:ROOT
3
4  include $(HALLD_HOME)/src/BMS/Makefile.lib
5
```

Build a Library

SConscript 〉 No Selection

```
1
2
3  import sbms
4
5  # get env object and clone it
6  Import('*')
7  env = env.Clone()
8
9  sbms.AddDANA(env)
10 sbms.library(env)
11
```

Build a Program

Makefile 〉 No Selection

```
1
2  PACKAGES = ROOT:DANA
3
4  include $(HALLD_HOME)/src/BMS/Makefile.bin
5
```

Build a Program

SConscript 〉 No Selection

```
1
2
3  import sbms
4
5  # get env object and clone it
6  Import('*')
7  env = env.Clone()
8
9  sbms.AddDANA(env)
10 sbms.AddROOT(env)
11 sbms.executable(env)
12
```

From SBMS/sbms.py

```
16  ################################
17  # library
18  ################################
19  def library(env, libname=''):
20
21      # Library name comes from directory name
22      if libname=='':
23          libname = os.path.split(os.getcwd())[1]
24
25      env.PrependUnique(CPPPATH = ['.'])
26
27      # Add C/C++, and FORTRAN targets
28      env.AppendUnique(ALL_SOURCES = env.Glob('*.c*'))
29      env.AppendUnique(ALL_SOURCES = env.Glob('*.F'))
30
31      sources = env['ALL_SOURCES']
32
33      # Build static library from all source
34      myobjs = env.Object(sources)
35      mylib = env.Library(target = libname, source = myobjs)
36
37      # Cleaning and installation are restricted to the directory
38      # scons was launched from or its descendents
39      CurrentDir = env.Dir('.').srcnode().abspath
40      if not CurrentDir.startswith(env.GetLaunchDir()):
41          # Not in launch directory. Tell scons not to clean these targets
42          env.NoClean([myobjs, mylib])
43      else:
44          # We're in launch directory (or descendent) schedule installation
45
46          # Installation directories for library and headers
47          installdir = env.subst('$INSTALLDIR')
48          includedir = "%s/%s" %(env.subst('$INCDIR'), libname)
49          libdir = env.subst('$LIBDIR')
50
51          # Install targets
52          env.Install(libdir, mylib)
53          env.Install(includedir, env.Glob('*.h*'))
```

Directory-level (dispatchers)

Makefile 〉 No Selection

```
DIRS += include HDDM BCAL CDC DANA FCAL CCAL PID ANALYSIS FDC START_COUNTER CERE RICH
DIRS += HDGEOMETRY TAGGER TOF TRACKING TRIGGER

ifdef AMPTOOLS
ifdef CLHEP

DIRS += AMPTOOLS_MCGEN AMPTOOLS_DATAIO AMPTOOLS_AMPS

endif
endif


include $(HALLD_HOME)/src/BMS/Makefile.dirs
```

SConscript 〉 No Selection

```
import os

Import('env osname')

# Loop over libraries, building each
subdirs = ['BCAL', 'FCAL', 'DANA', 'TAGGER', 'HDGEOMETRY', 'TRACKING', 'CCAL', 'START_COUNTER']
subdirs.append(['TRIGGER', 'CDC', 'FDC', 'PID', 'ANALYSIS', 'CERE', 'HDDM', 'RICH', 'TOF'])
SConscript(dirs=subdirs, exports='env osname', duplicate=0)

```

Times to compile and install all of *sim-recon* (and clean) on gluon46

Intel(R) Xeon(R) CPU E5-2650 v2 @ 2.60GHz (16 core + 16 ht)

make install
24:15

scons -j32 install
1:39

make clean
1:52

scons -c
0:38

make -j32 clean
0:24

scons -c -j32
0:32

How to use

BUILDING Single Threaded

From *src* directory …

```
> scons
```

From *any* other directory (also works in src directory) …

```
> scons -u
```

BUILDING Multi-Threaded

From *src* directory …

```
> scons -j8
```

From *any* other directory (also works in src directory) …

```
> scons -u -j32
```

How to use

INSTALLING

From *any* directory …

```
> scons -u install
```

CLEANING

Clean only build directory

```
> scons -u -c
```

Clean build directory *and* installation directory

```
> scons -u -c install
```

Output

SBMS

```
harriet:TOF>scons -u
scons: Entering directory `/Users/davidl/HallD/builds/sim-recon/src'
scons: Reading SConscript files ...
scons: done reading SConscript files.
scons: Building targets ...
scons: building associated VariantDir targets: .Darwin_macosx10.9-x86_64-llvm5.0/libraries/TOF
Compiling  [libraries/TOF/DTOFGeometry_factory.cc]
Compiling  [libraries/TOF/DTOFHit_factory.cc]
Compiling  [libraries/TOF/DTOFPaddleHit_factory.cc]
Compiling  [libraries/TOF/DTOFPoint_factory.cc]
Compiling  [libraries/TOF/TOF_init.cc]
Archiving    [.Darwin_macosx10.9-x86_64-llvm5.0/libraries/TOF/libTOF.a]
Ranlib       [.Darwin_macosx10.9-x86_64-llvm5.0/libraries/TOF/libTOF.a]
scons: `libraries/TOF' is up to date.
scons: done building targets.
harriet:TOF>
```

Output

BMS (partial)

```
harriet:TOF>make
mkdir -p .depends/Darwin_macosx10.9-x86_64-llvm5.0
g++ -MM -MT ".lib/Darwin_macosx10.9-x86_64-llvm5.0/libTOF.a(TOF_init.o)" -I. -I.. -I../include -I/Users/davidl/
HallD/builds/sim-recon/include -D_FILE_OFFSET_BITS=64 -I/Users/davidl/12GeV/builds/jana_0.6.6p2/
Darwin_macosx10.9-x86_64-llvm5.0/include -I/usr/local/xerces/PRO/include -DUSE_EVIO -I/Users/davidl/HallD/
ONLINE/ExternalPackages/EVIO/PRO/include -I/Users/davidl/HallD/ONLINE/ExternalPackages/ET/PRO/include
-DHAVE_ET -I/Users/davidl/HallD/ONLINE/ExternalPackages/ET/PRO/include -DHAVE_ET -D_ROOT_ -
DROOT_MAJOR=5 -DROOT_MINOR=34 -pthread -stdlib=libc++ -m64 -I/usr/local/root/
root_v5.34.11.Darwin_macosx10.9-x86_64-llvm5.0/include -I/usr/local/root/root_v5.34.11.Darwin_macosx10.9-
x86_64-llvm5.0/include -O2 -DBASENAME_IN_LIBGEN -DXDR_LONGLONG_MISSING TOF_init.cc > .depends/
Darwin_macosx10.9-x86_64-llvm5.0/TOF_init.d
mkdir -p .depends/Darwin_macosx10.9-x86_64-llvm5.0
g++ -MM -MT ".lib/Darwin_macosx10.9-x86_64-llvm5.0/libTOF.a(DTOFPoint_factory.o)" -I. -I.. -I../include -I/Users/
davidl/HallD/builds/sim-recon/include -D_FILE_OFFSET_BITS=64 -I/Users/davidl/12GeV/builds/jana_0.6.6p2/
Darwin_macosx10.9-x86_64-llvm5.0/include -I/usr/local/xerces/PRO/include -DUSE_EVIO -I/Users/davidl/HallD/
ONLINE/ExternalPackages/EVIO/PRO/include -I/Users/davidl/HallD/ONLINE/ExternalPackages/ET/PRO/include
-DHAVE_ET -I/Users/davidl/HallD/ONLINE/ExternalPackages/ET/PRO/include -DHAVE_ET -D_ROOT_ -
DROOT_MAJOR=5 -DROOT_MINOR=34 -pthread -stdlib=libc++ -m64 -I/usr/local/root/
root_v5.34.11.Darwin_macosx10.9-x86_64-llvm5.0/include -I/usr/local/root/root_v5.34.11.Darwin_macosx10.9-
x86_64-llvm5.0/include -O2 -DBASENAME_IN_LIBGEN -DXDR_LONGLONG_MISSING DTOFPoint_factory.cc
> .depends/Darwin_macosx10.9-x86_64-llvm5.0/DTOFPoint_factory.d
mkdir -p .depends/Darwin_macosx10.9-x86_64-llvm5.0

…
```

With make we had to help with defining dependencies. Scons handles it internally.

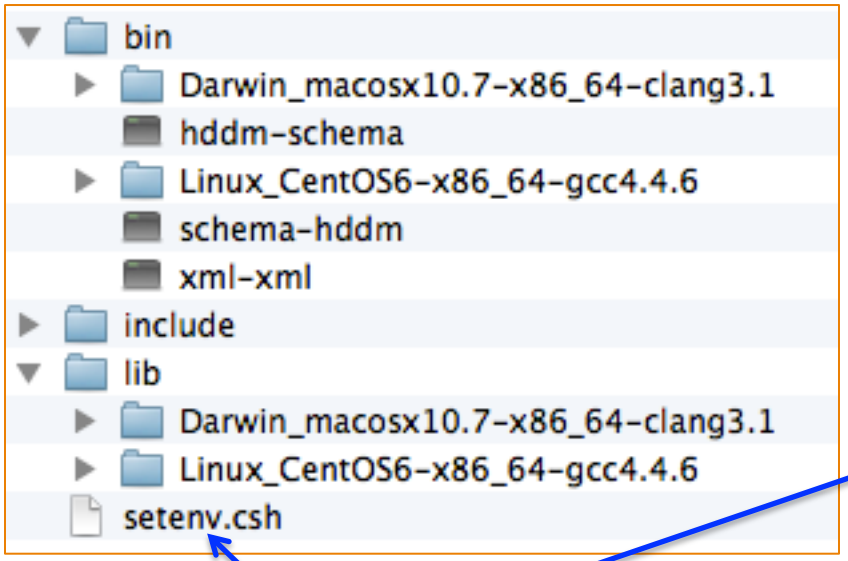
Telling scons to show you what it's doing

add the SHOWBUILD=1 argument

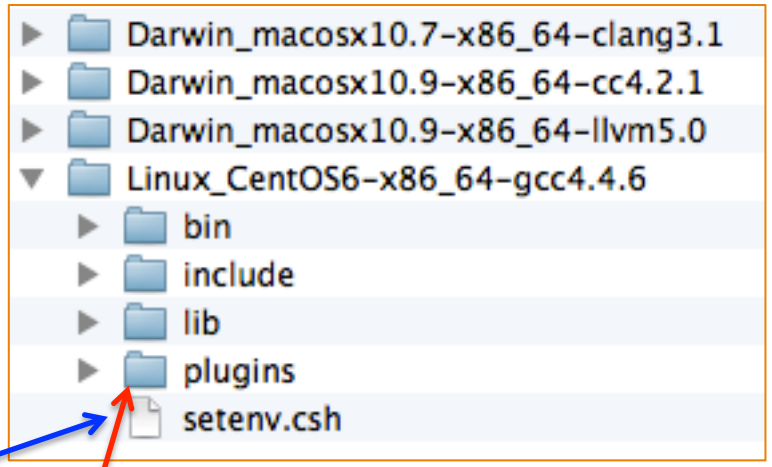
```
harriet:TOF>scons -u SHOWBUILD=1
scons: Entering directory `/Users/davidl/HallD/builds/sim-recon/src'
scons: Reading SConscript files ...

…

scons: done reading SConscript files.
scons: Building targets ...
scons: building associated VariantDir targets: .Darwin_macosx10.9-x86_64-llvm5.0/libraries/TOF
g++ -o .Darwin_macosx10.9-x86_64-llvm5.0/libraries/TOF/DTOFGeometry_factory.o -c -g -fPIC -fPIC -I/Users/
davidl/12GeV/builds/jana_0.6.6p2/Darwin_macosx10.9-x86_64-llvm5.0/include -I/usr/local/mysql/include -g -Os -
arch x86_64 -fno-common -D_P1003_1B_VISIBLE -DSIGNAL_WITH_VIO_CLOSE -
DSIGNALS_DONT_BREAK_READ -DIGNORE_SIGHUP_SIGQUIT -DDONT_DECLARE_CXA_PURE_VIRTUAL -I/
usr/local/xerces/xerces-c-3.1.1.Darwin_macosx10.9-x86_64-llvm5.0/include -I/usr/local/xerces/xerces-
c-3.1.1.Darwin_macosx10.9-x86_64-llvm5.0/include/xercesc -I/usr/local/root/PRO/include -I/usr/local/cMsg/PRO/
Darwin-x86_64/include -I/Users/davidl/HallD/builds/ccdb_latest/include -I.Darwin_macosx10.9-x86_64-llvm5.0/
libraries/TOF -Ilibraries/TOF -Iexternal/xstream/include -I/usr/local/xerces/PRO/include -I/Users/davidl/HallD/
builds/hdds/src -I/Users/davidl/HallD/builds/ccdb_latest/include -I. -Ilibraries -Ilibraries/include -I/Users/davidl/
HallD/builds/sim-recon/Darwin_macosx10.9-x86_64-llvm5.0/include libraries/TOF/DTOFGeometry_factory.cc

…
```

Change to installation directory Structure

BMS

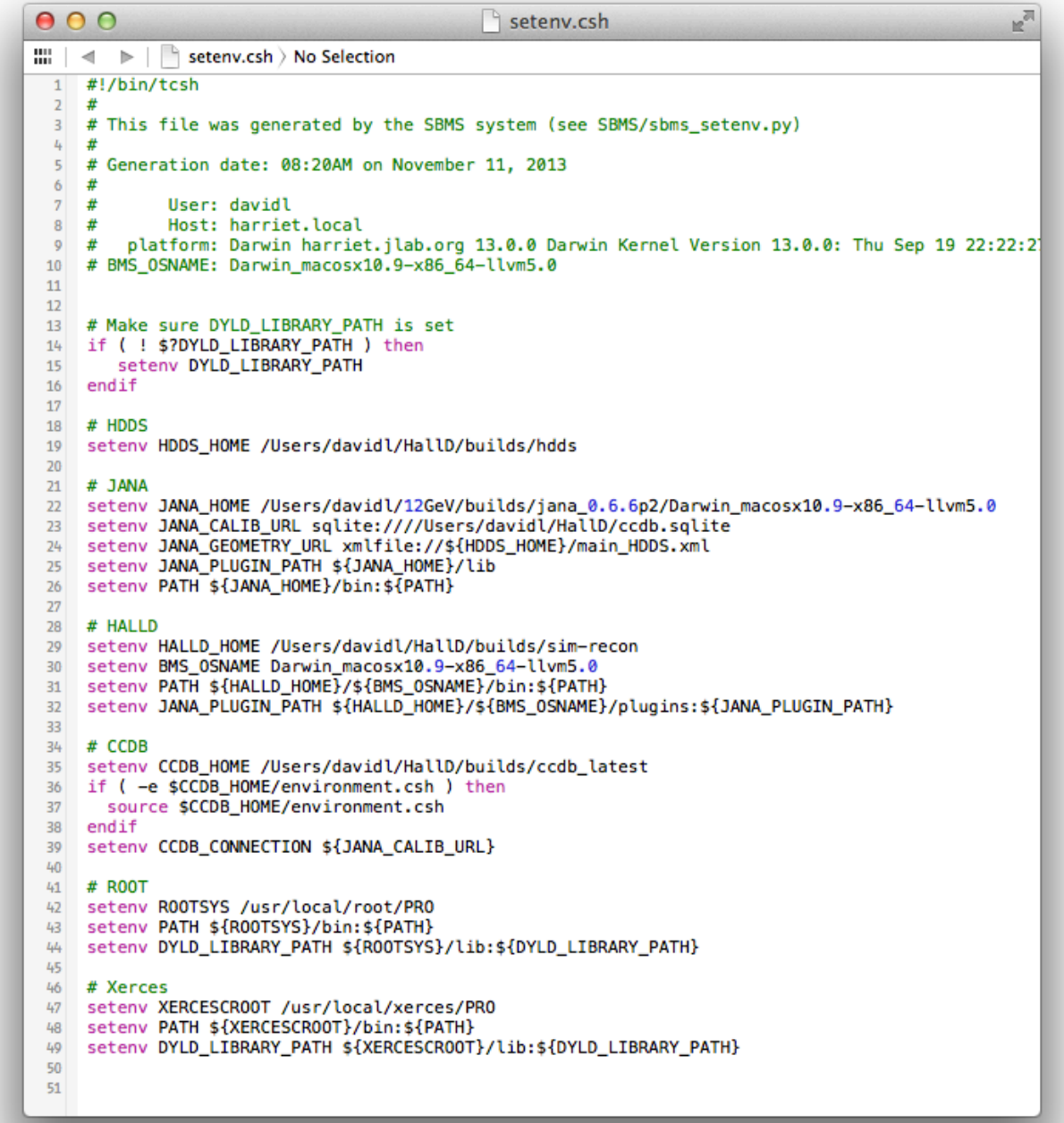


SBMS



setenv.csh generated by SBMS (previously had to run *mk_setenv.csh* script)

plugins now install in their own directory (used to install in *lib/XXX*)


setenv.csh
setenv.csh 〉 No Selection
#!/bin/tcsh
#
This file was generated by the SBMS system (see SBMS/sbms_setenv.py)
#
Generation date: 08:20AM on November 11, 2013
#
#   User: davidl
#   Host: harriet.local
#   platform: Darwin harriet.jlab.org 13.0.0 Darwin Kernel Version 13.0.0: Thu Sep 19 22:22:2
# BMS_OSNAME: Darwin_macosx10.9-x86_64-llvm5.0
# Make sure DYLD_LIBRARY_PATH is set
if (! $?DYLD_LIBRARY_PATH) then
setenv DYLD_LIBRARY_PATH
endif
# HDDS
setenv HDDS_HOME /Users/davidl/HallD/builds/hdds
JANA
setenv JANA_HOME /Users/davidl/12GeV/builds/jana_0.6.6p2/Darwin_macosx10.9-x86_64-llvm5.0
setenv JANA_CALIB_URL sqlite:////Users/davidl/HallD/ccdb.sqlite
setenv JANA_GEOMETRY_URL xmlfile://${HDDS_HOME}/main_HDDS.xml
setenv JANA_PLUGIN_PATH ${JANA_HOME}/lib
setenv PATH ${JANA_HOME}/bin:${PATH}
HALLD
setenv HALLD_HOME /Users/davidl/HallD/builds/sim-recon
setenv BMS_OSNAME Darwin_macosx10.9-x86_64-llvm5.0
setenv PATH ${HALLD_HOME}/${BMS_OSNAME}/bin:${PATH}
setenv JANA_PLUGIN_PATH ${HALLD_HOME}/${BMS_OSNAME}/plugins:${JANA_PLUGIN_PATH}
CCDB
setenv CCDB_HOME /Users/davidl/HallD/builds/ccdb_latest
if (-e $CCDB_HOME/environment.csh) then
source $CCDB_HOME/environment.csh
endif
setenv CCDB_CONNECTION ${JANA_CALIB_URL}
ROOT
setenv ROOTSYS /usr/local/root/PRO
setenv PATH ${ROOTSYS}/bin:${PATH}
setenv DYLD_LIBRARY_PATH ${ROOTSYS}/lib:${DYLD_LIBRARY_PATH}
Xerces
setenv XERCESCROOT /usr/local/xerces/PRO
setenv PATH ${XERCESCROOT}/bin:${PATH}
setenv DYLD_LIBRARY_PATH ${XERCESCROOT}/lib:${DYLD_LIBRARY_PATH}

Documentation

- https://halldweb1.jlab.org/wiki/index.php/SCons_Build_System