Using Standard Tools to Package and Distribute Scientific Software C and Fortran Libraries: a Demonstration with the General Purpose Timing Library (GPTL)

Edward Hartnett [1,2]

1 CIRES, University of Colorado, Boulder, CO 80309, USA
2 NOAA/ESRL/GSD, Boulder, CO 80305, USA

Summary

- GPTL C/Fortran libraries have been converted to an autoconf/automake/libtool based build system.
- Use of these standard tools is simple and improves portability.
- Build system complexity is substantially reduced.
- This poster presents the annotated build system for the C, Fortran, and combined C/Fortran projects.

How It Was Done

1. Created autotools build for C library, alongside existing build system.
2. Created new repository, copied all Fortran code and tests.
3. Created autotools build for Fortran library.
4. Once both new builds are demonstrated to work, deleted old build system and removed Fortran code and tests from original repository.
5. Created combined distribution, containing both C and Fortran libraries.

Results

- Build is more portable.
- Build system is less complex.
- C, Fortran, and a combined C/Fortran library have been released on GitHub.

Build System	Files	Lines of Code
Legacy	30	4807
Autotools (combined C/Fortran)	14	593

C Library Build

configure.ac

```
# Specify minimum autoconf version.
AC_PREREQ([2.59])

# Initialize autoconf.
AC_INIT([GPTL], [5.6.0], [james.rosinski@noaa.gov])

# Find out about the host we're building on.
AC_CANONICAL_HOST

# Find out about the target we're building for.
AC_CANONICAL_TARGET

# Initialize automake.
AM_INIT_AUTOMAKE([foreign subdir-objects])

# Keep libtool macros in an m4 directory.
AC_CONFIG_MACRO_DIR([m4])

# Set up libtool.
LT_PREREQ([2.4])
LT_INIT()

# The config.h file will be created when configure script is run.
AC_CONFIG_HEADERS([config.h])

# Find the C compiler.
AC_PROG_CC()

# These ensure proper handling of const and inline.
AC_C_CONST
AC_C_INLINE

# Set HAVE_NANOTIME on x86 systems only.
AC_MSG_CHECKING([whether x86 nanotime is available])
AS_CASE([$host], [*86*], [have_nanotime=yes], [have_nanotime=no])
if test "x$have_nanotime" = xyes; then
   AC_DEFINE([HAVE_NANOTIME], [1], [x86 nanotime capability is present])
fi
AC_MSG_RESULT($have_nanotime)

# Does the user want to turn on nested OMP?
AC_MSG_CHECKING([whether nested OMP is to be enabled])
AC_ARG_ENABLE([nestedomp], [AS_HELP_STRING([--enable-nestedomp],
              [build with nested OMP capability])])
test "x$enable_nestedomp" = xyes || enable_nestedomp=no
AM_CONDITIONAL(ENABLE_NESTEDOMP, [test x$enable_nestedomp = xyes])
AC_MSG_RESULT($enable_nestedomp)

# Does the user want to turn on PMPI?
AC_MSG_CHECKING([whether PMPI is to be enabled])
AC_ARG_ENABLE([pmpi], [AS_HELP_STRING([--enable-pmpi],
              [build with PMPI capability])])
test "x$enable_pmpi" = xyes || enable_pmpi=no
AM_CONDITIONAL(ENABLE_PMPI, [test x$enable_pmpi = xyes])
if test $enable_pmpi = yes; then
   AC_DEFINE([ENABLE_PMPI], [1], [enable pmpi])
fi
AC_MSG_RESULT($enable_pmpi)
AC_DEFINE([MPI_STATUS_SIZE_IN_INTS], [5], [size of status in MPI])

# Does the user want to test autoprofiling?
AC_MSG_CHECKING([whether autoprofile testing is to be enabled])
AC_ARG_ENABLE([autoprofile-testing], [AS_HELP_STRING([--enable-autoprofile-testing],
              [test autoprofileing])])
test "x$enable_autoprofile_testing" = xyes || enable_autoprofile_testing=no
AM_CONDITIONAL(TEST_AUTOPROFILE, [test $enable_autoprofile_testing = yes])
AC_MSG_RESULT($enable_autoprofile_testing)

# Does the user want to use double underscores for fortran wrappers?
AC_MSG_CHECKING([whether double underscore for Fortran wrappers should be enabled])
AC_ARG_ENABLE([double-underscore], [AS_HELP_STRING([--enable-double-underscore],
              [use double underscore for fortran wrappers])])
test "x$enable_double_underscore" = xyes || enable_double_underscore=no
AC_MSG_RESULT($enable_pmpi)
if test $enable_double_underscore = yes; then
   AC_DEFINE([FORTRANDOUBLEUNDERSCORE], [1], [use double underscore for Fortran])
else
   AC_DEFINE([FORTRANUNDERSCORE], [1], [use single underscore for Fortran wrappers]
fi

# Check for papi library.
AC_CHECK_LIB([papi], [PAPI_library_init])
AC_MSG_CHECKING([whether system can support PAPI])
if test "x$ac_cv_lib_papi_PAPI_library_init" = xyes; then
   # If we have PAPI library, check /proc/sys/kernel/perf_event_paranoid
   # to see if we have permissions.
   if test -f /proc/sys/kernel/perf_event_paranoid; then
      if test `cat /proc/sys/kernel/perf_event_paranoid` != 1; then
         AC_MSG_ERROR([PAPI found, but /proc/sys/kernel/perf_event_paranoid != 1
                       try sudo sh -c 'echo 1 >/proc/sys/kernel/perf_event_paranoid'])
      fi
   fi
   AC_DEFINE([HAVE_PAPI], [1], [PAPI library is present and usable])
   have_papi=yes
fi
AC_MSG_RESULT($have_papi)
AM_CONDITIONAL([HAVE_PAPI], [test "x$have_papi" = xyes])

# Check for rt library.
AC_CHECK_LIB([rt], [clock_gettime])

# Check for existence of procfs, the proc file system.
AC_CHECK_FILE([/proc],
              [AC_DEFINE([HAVE_SLASHPROC], [1], [some comment])])

# Check for pthread library.
AC_CHECK_LIB([pthread], [pthread_mutex_init])
if test "x$ac_cv_lib_pthread_pthread_mutex_init" = xyes; then
   AC_DEFINE([PTHREADS], [1], [pthreads library is present])
fi

# We need the math library for some tests.
AC_CHECK_LIB([m], [floor], [],
             [AC_MSG_ERROR([Can't find or link to the math library.])])

# Check for function backtrace_symbols.
AC_CHECK_FUNC([backtrace_symbols],
              [AC_DEFINE([HAVE_BACKTRACE], [1], [backtrace_symbols function is present])])

# Check for times.
AC_CHECK_FUNC([times],
              [AC_DEFINE([HAVE_TIMES], [1], [vfprint function is available])])

# Check for gettimeofday.
AC_CHECK_FUNC([gettimeofday],
              [AC_DEFINE([HAVE_GETTIMEOFDAY], [1], [gettimeofday function is available])])

# Do we have MPI?
AC_CHECK_FUNC([MPI_Init], [have_mpi=yes])
AM_CONDITIONAL([HAVE_MPI], [test "x$have_mpi" = xyes])

# Do we have function MPI_Comm_f2c?
AC_CHECK_FUNC([MPI_Comm_f2c], [have_mpi_comm_f2c=yes])
if test "x$have_mpi_comm_f2c" = xyes; then
   AC_DEFINE([HAVE_COMM_F2C], [1], [MPI_Comm_f2c is present])
fi

# Check for function iargc, which may be part of MPI.
AC_CHECK_FUNC([iargc],
              [AC_DEFINE([HAVE_IARGCGETARG], [1], [backtrace_symbols function present])])

# Check the size of a void pointer.
AC_CHECK_SIZEOF([void *])
if test "x$ac_cv_sizeof_void_p" = x8; then
   AC_DEFINE([BIT64], [1], [void pointer is 8 bytes])
fi

# This is a list of files to be built.
AC_CONFIG_FILES([Makefile
                 include/Makefile
                 test/Makefile
                 src/Makefile
                 bin/Makefile
                ])

# Build the files listed above.
AC_OUTPUT()
```

Initialize Autotools

Setup Config.h
Send all C pre-processor macros to special include file config.h, which must be included by all code in the package.

Find C Compiler

Check for x86
The GPTL has a special feature that is only available on x86 systems.

Configure Options
The user can pass options to configure, like --enable-pmpi. This causes pre-processor macro ENABLE_PMPI to be set in config.h, also automake conditional ENABLE_PMPI is set to true, which will be used when the Makefiles are built.

Find Libraries and Functions
Check whether some libraries and functions can be found on the build system. In some cases (like the math library) configure will error out if it's not found. In other cases, pre-processor macros will be set in config.h, so that the C code can be written to handle the presence/absence of the library.

Configure Outputs
After learning about the build system, the configure script will use the results, and automake, to build the five Makefiles that are needed.

Makefile.am

```
# This directory stores libtool macros, put there by aclocal.
ACLOCAL_AMFLAGS = -I m4

# These files get added to the distribution.
EXTRA_DIST = COPYING README

# This is the list of subdirs for which Makefiles will be constructed
# and run.
SUBDIRS = include src test bin
```

Main Makefile
SUBDIRS are processed in order.

include/Makefile.am

```
include_HEADERS = gptl.h
noinst_HEADERS = private.h
```

Include Files
Headers are used for the build, and installed in the include directory, unless the noinst_ prefix is used.

bin/Makefile.am

```
# Install script in $(bindir) and distribute it.
dist_bin_SCRIPTS = parsegptlout.pl
```

Scripts
Scripts are installed in the bin directory.

src/Makefile.am

```
libgptl_la_CPPFLAGS = -I$(top_srcdir)/include

# This is our output. The GPTL library.
lib_LTLIBRARIES = libgptl.la

# These are the source files.
libgptl_la_SOURCES = f_wrappers.c getoverhead.c gptl.c gptl_papi.c   \
hashstats.c memstats.c memusage.c pmpi.c print_rusage.c pr_summary.c \
util.c
```

Build the Library
Builds static/shared libraries. Note that all library source files are listed.

test/Makefile.am

```
# Find include directory.
AM_CPPFLAGS = -I${top_srcdir}/include

# Link to GPTL library.
AM_LDFLAGS = ${top_builddir}/src/libgptl.la

# These programs will be built but not installed.
noinst_PROGRAMS = gran_overhead printwhileon

# Test programs that will be built for all configurations.
check_PROGRAMS = tst_simple global
TESTS = tst_simple global

# Build these tests if PAPI is present.
if HAVE_PAPI
check_PROGRAMS += avail testpapi
TESTS += avail testpapi
noinst_PROGRAMS += papiomptest knownflopcount
endif

# Build these tests if MPI is present.
if HAVE_MPI
check_PROGRAMS += summary
TESTS += run_par_test.sh
if ENABLE_PMPI
check_PROGRAMS += pmpi
TESTS += run_par_pmpi_test.sh
endif
endif

# Build these if the user selected --enable-autoprofile-tests during
configure.
if TEST_AUTOPROFILE
check_PROGRAMS += cygprofile cygprofilesubs
TESTS += cygprofile cygprofilesubs
endif

# Build these if the user selected --enable-nestedomp during
configure.
if ENABLE_NESTEDOMP
check_PROGRAMS += nestedomp
TESTS += nestedomp
endif

# Test output to be deleted.
CLEANFILES = timing.*
```

Set Up Testing

Build, Don't Install

Tests
These tests will be built and run for all builds.

Optional Tests
These tests will be built and run for some builds, depending on the values of automake conditionals like HAVE_MPI.

Clean Up
These files are created during testing, and removed by the clean target.

Fortran Library Build

configure.ac

```
# Specify minimum autoconf version.
AC_PREREQ([2.59])

# Initialize autoconf.
AC_INIT([GPTL-fortran], [5.6.0], [james.rosinski@noaa.gov])

# Keep libtool macros in an m4 directory.
AC_CONFIG_MACRO_DIR([m4])

# Find out about the host we're building on.
AC_CANONICAL_HOST

# Find out about the target we're building for.
AC_CANONICAL_TARGET

# Initialize automake.
AM_INIT_AUTOMAKE([foreign subdir-objects])

# Set up libtool.
LT_PREREQ([2.4])
LT_INIT()

# Find the fortran compiler.
AC_PROG_FC()

# Set HAVE_NANOTIME on x86 systems only.
AC_MSG_CHECKING([whether x86 nanotime is available])
AS_CASE([$host], [*86*], [have_nanotime=yes], [have_nanotime=no])
if test "x$have_nanotime" = xyes; then
   AC_DEFINE([HAVE_NANOTIME], [1], [x86 nanotime capability is present])
fi
AC_MSG_RESULT($have_nanotime)

# Check for papi library.
AC_CHECK_LIB([papi], [PAPI_library_init], [], [have_papi=no])
test "x$have_papi" = xno || have_papi=yes

# When built as part of the combined C/Fortran library distribution,
# the fortran library needs to be built with
# --enable-package-build. This tells the fortran library where to find
# the C library.
AC_ARG_ENABLE([package-build],
  [AS_HELP_STRING([--enable-package-build],
    [Set internally for package builds, should not be used by user.])])
test "x$enable_package_build" = xyes || enable_package_build=no
AM_CONDITIONAL([BUILD_PACKAGE], [test "x$enable_package_build" = xyes])

# Find the GPTL C library, unless this is a combined C/Fortran library
# build.
if test $enable_package_build = no; then
   AC_CHECK_LIB([gptl], [GPTLinitialize], [],
                [AC_MSG_ERROR([Can't find or link to the GPTL C library.])])
fi

# Do we have MPI?
AC_CHECK_FUNC([MPI_Init],
              [AC_DEFINE([HAVE_MPI], [1], [MPI is present])])
AM_CONDITIONAL([HAVE_MPI], [test "x$ac_cv_func_MPI_Init" = xyes])

# See if the C GPTL was built with PAPI.
AC_CHECK_FUNC([gptl_papilibraryinit], [c_has_papi=yes], [c_has_papi=no])
if test $have_papi = yes -a $c_has_papi = no; then
   AC_MSG_CHECKING([PAPI library is present, but was not built. Turning PAPI off.])
   have_papi=no
fi

# Determine the have_papi settings for this build.
AC_MSG_CHECKING([whether PAPI library is present and should be used])
AM_CONDITIONAL([HAVE_PAPI], [test "x$have_papi" = xyes])
if test "x$have_papi" = xyes; then
   AC_DEFINE([HAVE_PAPI], [1], [PAPI library is present])
fi
AC_MSG_RESULT($have_papi)

# Check for function backtrace_symbols.
AC_CHECK_FUNC([backtrace_symbols], [have_backtrace=yes], [have_backtrace=no])
if test $have_backtrace = yes; then
   AC_DEFINE([HAVE_BACKTRACE], [1], [backtrace_symbols function is present])
fi
AM_CONDITIONAL([HAVE_BACKTRACE], [test "x$have_backtrace" = xyes])

# Make sure this file is copied to build directories for tests to
# work.
AC_CONFIG_LINKS([test/gptlnl:test/gptlnl])

# This is a list of files to be built.
AC_CONFIG_FILES([Makefile
                 include/Makefile
                 src/Makefile
                 test/Makefile
                ])

# Build the files listed above.
AC_OUTPUT()
```

Initialize Autotools

Find Fortran Compiler

Check for x86

Find Library

Find GPTL C Library
There is special handling for combined builds.

Check C Library
Check for various features in the C library. Set pre-processor macros for the Fortran code.

Configure Outputs
These four makefiles will be built by configure.

Makefile.am

```
# This directory stores libtool macros, put there by aclocal.
ACLOCAL_AMFLAGS = -I m4

# These files get added to the distribution.
EXTRA_DIST = COPYING

# This is the list of subdirs for which Makefiles will be
# constructed and run.
SUBDIRS = src test include
```

Build Subdirs
Run make in subdirectories src, test, and include.

include/Makefile.am

```
include_HEADERS = gptl.inc
```

Install gptl.inc

src/Makefile.am

```
libgptlf_la_FCFLAGS = -I$(top_srcdir)/include

if HAVE_PAPI
libgptlf_la_FCFLAGS += -DHAVE_PAPI
endif

if HAVE_MPI
libgptlf_la_FCFLAGS += -DHAVE_MPI
endif

# This is our output. The GPTL-fortran library.
lib_LTLIBRARIES = libgptlf.la
libgptlf_la_SOURCES = gptlf.F90 printmpistatussize.F90      \
process_namelist.F90

# Install these in the include directory.
include_HEADERS = gptl.mod
```

Set FCFLAGS

Build Library

Install gptl.mod

test/Makefile.am

```
# Find include directory and src directory (for .mod file).
AM_CPPFLAGS = -I${top_srcdir}/include -I${top_srcdir}/src

# Link to GPTL fortran library.
AM_LDFLAGS = ${top_builddir}/src/libgptlf.la

# For combined C/Fortran builds, find the C library.
if BUILD_PACKAGE
LDADD = ${top_builddir}/../GPTL/src/libgptl.la
endif

MODFILE = ${top_builddir}/src/gptl.mod

# Test programs that will be built for all configurations.
check_PROGRAMS = testbasics errtest handle  outoforder      \
overhead testbacktrace nlreader
TESTS = testbasics errtest handle outoforder testbacktrace nlreader

# Test dependencies needed.
testbasics_SOURCES = testbasics.F90 ${MODFILE}
errtest_SOURCES = errtest.F90 ${MODFILE}
handle_SOURCES = handle.F90 ${MODFILE}
nlreader_SOURCES = nlreader.F90 ${MODFILE}
outoforder_SOURCES = outoforder.F90 ${MODFILE}

# Extra test if we have backtrace.
if HAVE_BACKTRACE
check_PROGRAMS += testbacktrace
TESTS += testbacktrace
testbacktrace_SOURCES = testbacktrace.F90 ${MODFILE}
endif

# Test PAPI functionality if libpapi was found.
if HAVE_PAPI
AM_CPPFLAGS += -DHAVE_PAPI
check_PROGRAMS += testinit testpapi
TESTS += testinit testpapi
testpapi_SOURCES = testpapi.F90 ${MODFILE}
testinit_SOURCES = testinit.F90 ${MODFILE}
endif

# Test output to be deleted.
CLEANFILES = timing.*

# This file is required for the nlreader test.
EXTRA_DIST = gptlnl
```

Set Flags
Set up the compile flags for the test code.

Build Tests

Optional Tests
These Fortran tests may be built, depending on what was learned during configure.

Clean Up

Distribute Test File

Combined C/Fortran Build

configure.ac

```
AC_PREREQ([2.69])
AC_INIT([GPTL-all], [1.0.0], [edward.hartnett@noaa.gov])

# Find out about the host we're building on.
AC_CANONICAL_HOST

# Find out about the target we're building for.
AC_CANONICAL_TARGET

# Initialize automake.
AM_INIT_AUTOMAKE([foreign subdir-objects])

# Keep macros in an m4 directory.
AC_CONFIG_MACRO_DIR([m4])

# Set up libtool.
LT_PREREQ([2.4])
LT_INIT()

# Build the GPTL C library.
AC_CONFIG_SUBDIRS([GPTL])

# Add this arg for the fortran build, to tell it to
# use the C library we just built.
ac_configure_args="$ac_configure_args --enable-package-build"

# Build the GPTL Fortran library.
AC_CONFIG_SUBDIRS([GPTL-fortran])

AC_CONFIG_FILES([Makefile])

AC_OUTPUT
```

Initialize Autotools

Build C Library

Build Fortran Library

Configure Output
Cause configure to build the Makefile which will launch the build of the C and Fortran GPTL libraries.

Makefile.am

```
# This directory stores libtool macros, put there by aclocal.
ACLOCAL_AMFLAGS = -I m4

SUBDIRS = GPTL GPTL-fortranw
```

Build Subdirectories

Extended Abstract: https://ams.confex.com/ams/2019Annual/mediafile/Manuscript/Paper350095/AutotoolGPTL.pdf