LuMP

Sources can be obtained from:

https://github.com/AHorneffer/lump-l...-mpifr-pulsare

See LuMP's files and the DEKI's LuMP section for installation instructions.

Pulsar Software

For step by step instructions see https://docs.google.com/document/d/14su_NmPMbj1TDmSa7PgWCnGPivrmUzFwhzJaMIhyP24/edit?usp=sharing

To enable pulsar observations with LuMP and processing of raw voltages to folded profiles we need to have the pulsar package DSPSR installed. This in turn depends on the **development** version of PSRCHIVE. The latter depends on many packages. Here is a complete list of software that needs to be installed (in one of the possible orders in which the installation can proceed):

1. cfitsio (desirable): *ftp://heasarc.gsfc.nasa.gov/software/fitsio/c/cfitsio_latest.tar.gz* (always points to the newest version)
2. FFTW (or IPP or MKL): *http://www.fftw.org/fftw-3.3.4.tar.gz* (very rare updates)
3. git (desirable, but if you obtain sources differently then can live without this)
4. cvs (desirable, but if you obtain sources differently then can live without this)
5. Gnu Scientific Library (desirable)
6. Healpix (unlikely to be of use on recording machines)
7. pgplot (highly recommended but not mandatory) *ftp://ftp.astro.caltech.edu/pub/pgplot/pgplot5.2.tar.gz* (very unlikely to be upgraded)
8. psrcat http://www.atnf.csiro.au/research/pulsar/psrcat/downloads/psrcat_pkg.tar.gz (always points to the newest version)
9. tempo (highlly recommended but not mandatory): `git clone git://git.code.sf.net/p/tempo/tempo`
10. tempo2: cvs -z3 -d:pserver:anonymous@tempo2.cvs.sourceforge.net:/cvsroot/tempo2 co -P tempo2
11. psrchive: git clone git://git.code.sf.net/p/psrchive/code psrchive
12. dspsr: git clone git://git.code.sf.net/p/dspsr/code dspsr

In principle all the information is avaialable at http://psrchive.sourceforge.net/current/build.shtml

Here is a summary where I list deviations from the standard (bootstrap) -> configure -> make -> make install procedure.

CFITSIO:

Use *--enable-reentrant* while running configure.

FFTW:

Use *--enable-float* while running configure. This is needed for psrchive.

LuMP possibly depends on having the version of libraries for different numerical precisions, compile multiple times for all 4 possibilties (--enable-float, --enable-long-double, --enable-quad-precision, and without any of these for double precision).

If psrchive is to be compiled with python interface (desirable even on recording machines although we haven't had it so far (July 2015)) then add *--enable-shared* as well.

psrcat:
No autotools support. Simply sourcing 'makeit' should work on most systems. Remember to set PSRCAT_FILE environment variable to point to psrcat.db. You also may want to setup PSRCAT_RUNDIR where you can place additional catalogues. We use it e.g., to observe unpublished sources. At the moment we have to manually maintain the same version of catalogue on all the recording machines and LCUs

pgplot:
Follow the instructions here: http://psrchive.sourceforge.net/third/pgplot/

In particular, on newer systems one might need the patch pndriv.patch which is provided by psrchive sources in packages directory. I recommend using the XW, PS, CPS, VPS, CVPS, and PNG drivers.

If python interface for psrchive will be needed then also apply the makemake.sharedcpg.patch from the packages directory of psrchive.

One needs to setup the PGPLOT_DIR, PGPLOT_FONT variables.

tempo and tempo2:
Follow the standard procedure. Note that tempo uses a scritp called 'preparemake' rather than 'bootstrap'. One also needs to setup TEMPO and TEMPO2 variables as explained in README in the sources. May have to tweak the makefiles to get shared version to work. For tempo2 running 'make plugins' is necessary if plugins are needed (not necessary on the recording machines). I'll try to provide more detailed instructions later.

psrchive:
Standard procedure. Add *--enable-shared* to get the python interface.

Important step: after installation of psrchive do the following:

cd Util/fft

make bench

This will install FFT benchmarks which we use during data processing. Without them you need different processing scripts (simply remove -fft-bench when invoking dspsr). Make sure no one else is using the machine while running the benchmark.

dspsr:
Create and edit a backends.list file. For LOFAR recording one might populate the file with the following line:

apsr asp bcpm bpsr cpsr cpsr2 fits gmrt mark4 mark5 maxim mwa pmdaq puma2 s2 sigproc lump

Although in practice it might be enough to compile with backends.list containing only "lump" (not tested).

LOFAR Software

This is for a hopefully soon-to-be-old version of the LOFAR software. It is not yet tested on the most recent OS (installed on 24.6.2015 on lofarDN)

```
Download packages:
##################
  mkdir /opt/soft/lofar-stuff/Downloads
  cd /opt/soft/lofar-stuff/Downloads
  wget ftp://ftp.atnf.csiro.au/pub/software/asap/data/asap_data.tar.bz2

Download and Build Casacore:
############################
  cd /opt/soft/lofar-stuff/
  tar -xjvf Downloads/asap_data.tar.bz2
(This creates the "data" subdirectory)
  mkdir -p BuildDir/casacore
  cd BuildDir/casacore
  svn co http://casacore.googlecode.com/svn/trunk source
  mkdir -p build/opt
  cd build/opt
  cmake -DBUILD_TESTING=NO -DCMAKE_INSTALL_PREFIX=/opt/soft/lofar-stuff
-DUSE_FFTW3=Yes -DUSE_THREADS=YES -DDATA_DIR=/opt/soft/lofar-stuff/data
../../source
  make -j20
  make install

Download and Build pyrap
############################
  mkdir /opt/soft/lofar-stuff/BuildDir/pyrap
  cd /opt/soft/lofar-stuff/BuildDir/pyrap
  svn co http://pyrap.googlecode.com/svn/trunk dev-source
  export PATH=$PATH:/opt/soft/lofar-stuff/bin
  export LD_LIBRARY_PATH=/opt/soft/lofar-stuff/lib
  export PYTHONPATH=/opt/soft/lofar-stuff/lib/python
  ln -s /opt/soft/lofar-stuff/lib /opt/soft/lofar-stuff/lib64
  cd dev-source/
  ./batchbuild-trunk.py --casacore-root=/opt/soft/lofar-stuff
--prefix=/opt/soft/lofar-stuff --python-prefix=/opt/soft/lofar-stuff/lib/
python

Download and Build casarest
###########################
  mkdir  /opt/soft/lofar-stuff/BuildDir/casarest
  cd /opt/soft/lofar-stuff/BuildDir/casarest
  #svn co svn://lofar9.astron.nl/var/svn/repos/trunk/casarest source
```

```
  svn co https://github.com/pkgw/casarest/trunk source
  mkdir build
  cd build
  cmake -DCASACORE_ROOT_DIR=/opt/soft/lofar-stuff -DBUILD_ALL=1
-DCMAKE_INSTALL_PREFIX:PATH=/opt/soft/lofar-stuff ../source
  make -j20
  make install

Download and Build the LOFAR Software
#####################################
  mkdir /opt/soft/lofar-stuff/BuildDir/lofarsoft
  cd /opt/soft/lofar-stuff/BuildDir/lofarsoft
  svn checkout --ignore-externals https://svn.astron.nl/LOFAR/branches/
LOFAR-Release-2_6 LOFAR
username: lofar-guest
password: lofar-guest
  mkdir -p build/gnu_opt
  cd build/gnu_opt
  cmake -DCASACORE_ROOT_DIR=/opt/soft/lofar-stuff -DBUILD_SHARED_LIBS=ON
-DUSE_OPENMP=ON -DBUILD_PACKAGES="ParmDB Calibration DP3 Pipeline MSLofar
Deployment LofarFT" -DCMAKE_INSTALL_PREFIX:PATH=/opt/soft/lofar-stuff
../../LOFAR/
  make -j20
  make install

  Copy and build the TBBpack
####################################
  cd Download
  wget ftp://ftp.mpifr-bonn.mpg.de/outgoing/horneff/TBBraw2h5-pack.v2.tgz
  cd ../BuildDir
  tar -xzf ../Download/TBBraw2h5-pack.v2.tgz
  cd TBBraw2h5-pack/
  mkdir build
  cd build
  #the old cmake file to find HDF5 doesn't work on the current lofarXN
(10.7.2015)
  mv source/cmake/FindHDF5_DAL.cmake-cmake3.0 source/cmake/
FindHDF5_DAL.cmake
  #the cmake file to find Boost has similar problems, but we can fix that
from the command-line
  cmake -DDAL_INSTALL_PREFIX:PATH=/opt/lofar-stuff/TBBpack
-DBOOST_INCLUDEDIR=/usr/include -DBOOST_LIBRARYDIR=/usr/lib/
x86_64-linux-gnu ../source
  make -j12
  make install
```