prjemian fix typo and add convenience commands 3899818 on Feb 6

5 contributors

Executable File 735 lines (560 sloc) 33.2 KB

areaDetector: Installation Guide

Marty Kraimer, Brookhaven National Laboratory

Mark Rivers, University of Chicago

Introduction

This is a guide for installing and building R3-0 and later of the EPICS areaDetector module. This guide is intended for both areaDetector users and developers. areaDetector can be obtained as a release or by cloning from the github repository.

This guide provides instructions for:

- Installing, building, and running from source code. These instructions should work on any supported EPICS host architecture, e.g. Linux, Windows, Mac OSX, vxWorks. This document assumes that the reader has already installed an EPICS development environment, and has built EPICS base, the EPICS asyn module, and the required EPICS synApps modules.
- Installing and running a pre-built binary distribution.

Installing and Building from Source Code

The build process attempts to make the build process easy for typical cases but allow site specific overrides. areaDetector uses many other products.

EPICS Products Required for Building areaDetector

areaDetector requires EPICS base. R3.14.12.4 or higher, any 3.15 release should work.

areaDetector also requires asyn. The most recent release of asyn is recommended.

Each areaDetector detector module builds both a library and an EPICS IOC application. To build the library only EPICS base and asynDriver are required. To build the IOC application the synApps modules AUTOSAVE, BUSY, CALC, and SSCAN are required. If the CALC module is built with SNCSEQ support then SNCSEQ is also required. The most recent release of the synApps modules is recommended.

The DEVIOCSTATS and ALIVE modules are optional.

EPICS base, asyn and the synApps modules must be built before building areaDetector.

External Products Required for Building areaDetector

areaDetector optionally uses the NETCDF, TIFF, ZLIB, JPEG, SZIP, HDF5, NEXUS, GRAPHICSMAGIC, OPENCV, and EPICS PVA (formerly V4) libraries. These are used for plugins and drivers and are not required. The XML2 library is required.

Prior to areaDetector R2-5 the TIFF, ZLIB, JPEG, SZIP, XML2, HDF5 libraries needed to be installed on Linux. On Windows they were provided as pre-built libraries in the ADBinaries module in areaDetector. NETCDF and NEXUS were built from source in ADCore.

Beginning with R2-5 the ADSupport module was added to areaDetector. This module builds the GRAPHICSMAGICK, NETCDF, TIFF, ZLIB, JPEG, SZIP, XML2, HDF5, BLOSC, and NEXUS libraries from source code. ADSupport on Darwin may be added in a future release. ADSupport will almost always be used to build the libraries on Windows and vxWorks.
On Linux each library can either be built in ADSupport or installed in a location external to areaDetector.

For each library XXX (XXX=TIFF, NETCDF, etc.) there are 4 Makefile variables that can be defined in CONFIG_SITE.local.

- WITH_XXX
  - If WITH_XXX=YES then build the plugins and drivers that require this library.
  - If XXX_EXTERNAL=NO then also build the source code for this library in ADSupport.
- XXX_EXTERNAL
  - If NO then build the source code for this library in ADSupport.
  - If YES then this library is installed external to areaDetector
- XXX_INCLUDE
  - If XXX_EXTERNAL=YES then this is the path to the include files for XXX. However, if XXX is a system library whose include files are in a standard include search path then do not define XXX_INCLUDE.
- XXX_LIB
  - If XXX_EXTERNAL=YES then this is the path to the library files for XXX. However, if XXX is a system library whose library files in a standard library search path then do not define XXX_LIB.

XML2 is an exception. It is required, so WITH_XML2 is not supported, but XML2_EXTERNAL, XML2_INCLUDE, and XML2_LIB are supported.

Note that there are some library interdependencies.

- If WITH_TIFF=YES then WITH_ZLIB must also be YES.
- If WITH_HDF5=YES then WITH_ZLIB and WITH_SZIP must also be YES.
- If WITH_NEXUS=YES then WITH_HDF5 must also be YES.

ADSupport can be used to build the libraries for the following operating systems.

- Linux (32/64 bit, Intel architectures only).
- Windows (Visual Studio or mingw, 32/64-bit, static or dynamic)
- vxWorks
  - On vxWorks 6.x all libraries are supported.
  - On vxWorks 5.x only XML2, TIFF, ZLIB, JPEG, and NETCDF are supported.
    WITH_SZIP, WITH_HDF5, and WITH_NEXUS must be set to NO
- Darwin. This is not currently supported but will be added in a future release.

The following instructions can be used for installing these libraries externally to areaDetector if that is desired.

TIFF, JPEG, XML2, and Z

On Linux and Darwin the libtiff, libjpeg, libxml2, and libz libraries often come already installed. If they are not already installed then they are normally available for installation via the standard package installation tools, e.g. "yum install" on Redhat systems, "apt get" for Debian systems, etc.

If for some reason you cannot install these libraries using the standard package installation tools, they can be installed from source code distributions as follows.

For each product download the source code from a product download site and then build.

Each site provides some combination of .zip, .tar, .tar.gz, etc. If the file is a zip file then just execute:

```
unzip file.zip
```

If the file is any flavor of tar just execute:

```
tar xf file
```

All of the following products put the result of unzip or tar into a sub-directory. cd to that directory and follow the instructions for the appropriate sub-section.

TIFF

This can be downloaded from libTIFF. Then click on the Master Download Site and download the latest stable release. After the latest release is unziped cd to the release and execute the commands:

```
./configure
make
sudo make install
```

The include and library files are installed by default into **/usr/local**

ZLIB

This can be downloaded from zlib.

Then look for latest release download.

After untaring the release

```
./configure
make
sudo make install
```

The include and library files are installed by default into **/usr/local**

JPEG

This can be downloaded from libjpeg.

Read the instructions from the web site about installing. I suggest that it be installed into **/usr/local** instead of **/usr**. Thus to install:

```
./configure --prefix=/usr/local  \
     --mandir=/usr/share/man \
     --with-jpeg8  \
     --disable-static \
     &&; \
sed -i -e '/^docdir/ s/$/\/libjpeg-turbo-1.3.0/'   \
     -e '/^exampledir/ s/$/\/libjpeg-turbo-1.3.0/' Makefile &amp;amp;&amp;amp;
make
make test
sudo make install
```

XML2

This can be downloaded from libxml2.

Then look for latest release download.

After untaring the release

```
./configure
make
sudo make install
```

HDF5, SZIP, and GRAPHICS_MAGICK

On Linux and Darwin these libraries may not be installed, and may not be available via the standard package installation tools. Follow these steps to install them from source.

SZIP

This can be downloaded from SZIP.

Click on **SZIP Source** and download the release that appears.

After the latest release is untared, cd to the release and execute the commands:

```
./configure --prefix=/usr/local
make
make check
sudo make install
```

HDF5

This can be downloaded from HDF Group.

Click on Downloads, then Current Release, then HDF5 Software, then Source Code, then latest release.

After the latest release is untared, cd to the release and execute the commands:

```
./configure --prefix=/usr/local/hdf5 --with-szlib=/usr/local
make
make check
sudo make install
make check-install
```

GRAPHICSMAGICK

This can be downloaded from Sourceforge

After the latest release is untared, cd to the release and execute the commands:

```
./configure
make
sudo make install
```

Downloading and Installing areaDetector Source Code

The areaDetector source code is kept on gitub.com/areaDetector.

It can be downloaded 2 ways:

1. Via the "git clone" command:

```
git clone --recursive https://github.com/areaDetector/areaDetector.git
```

After downloading with git clone --recursive each submodule will be in a "detached HEAD" state. This means that its state will be that of the last time that module was committed to the top-level areaDetector repository. This is normally not the desired state for each submodule. Rather, one should cd to each submodule and type either `git checkout master` to work on the master branch, or `git checkout RX-Y` to use release RX-Y of that submodule.

2. By downloading tar.gz or zip files for a specific release of each module through a Web browser or by the wget command:

```
wget https://github.com/areaDetector/areaDetector/archive/R2-6.tar.gz
```

If downloading tar files then each repository must be downloaded separately. To build the "core" of areaDetector the following repositories must be downloaded:

- areaDetector/areaDetector
- areaDetector/ADSupport
- areaDetector/ADCore

To build the simulation detector, which is very useful for learning areaDetector and for testing, also download

- areaDetector/ADSimDetector

To also build a specific detector, for example the ADProsilica, also download

- areaDetector/ADProsilica

The areaDetector software is designed to be installed in the following tree structure, though this is not required. If it is installed this way then only the top-level areaDetector/configure directory needs to be edited for site-specific configuration.

```
areaDetector
  ADSupport
  ADCore
```

```
ADSimDetector
ADCSmDetector
pvaDriver
ADPilatus, etc.
```

RELEASE* and CONFIG* files

areaDetector RELEASE* and CONFIG* files are a little more complex than those in a typical EPICS modules. This is because they are designed to meet the following requirements:

- If using the top-level areaDetector repository then it is only necessary to edit the CONFIG* and RELEASE* files in the areaDetector/configure directory, and not in each of the submodules, of which there are now about 40.
- Allows building multiple architectures in the same tree, including Linux and Windows. This means that SUPPORT and EPICS_BASE may be defined differently for different architectures, since the path syntax is different for Linux and Windows.
- Allows using the top-level synApps/support and synApps/support/configure directories. If these are used then one can edit synApps/support/configure/RELEASE to set the locations of EPICS_BASE and the versions of asyn, calc, etc. Typing `make release` in the top-level synApps/support directory will update the RELEASE* files in all modules defined in that RELEASE file, including those in areaDetector/configure.
- Allows using the Debian EPICS package for EPICS_BASE and the support modules (asyn, calc, etc.). It is also possible to use the Debian package for some of the modules, but use more recent versions of some modules (e.g. asyn) that are built from source.

After all the required products have been installed and a release of areaDetector has been downloaded then do the following in the areaDetector/configure directory:

```
cp EXAMPLE_RELEASE.local         RELEASE.local
cp EXAMPLE_RELEASE_SUPPORT.local RELEASE_SUPPORT.local
cp EXAMPLE_RELEASE_LIBS.local    RELEASE_LIBS.local
cp EXAMPLE_RELEASE_PRODS.local   RELEASE_PRODS.local
cp EXAMPLE_CONFIG_SITE.local     CONFIG_SITE.local
```

You may also want to copy the architecture dependent example files if you are building for multiple architectures in a single build tree, for example:

```
cp EXAMPLE_CONFIG_SITE.local.WIN32                   CONFIG_SITE.local.WIN32
cp EXAMPLE_CONFIG_SITE.local.linux-x86.vxWorks-ppc32 CONFIG_SITE.local.linux-x86.vxWorks-ppc32
```

You can copy all of the EXAMPLE_* files to the files actually used with the copyFromExample script in the areaDetector/configure directory. If you do this then be sure to edit the CONFIG_SITE.local.$(EPICS_HOST_ARCH) for your EPICS_HOST_ARCH as well. For example CONFIG_SITE.local.linux-x86_64 defines WITH_BOOST=YES and this may need to be changed if you do not have the boost-devel package installed. You can see your local modifications with the diffFromExample script.

Edit RELEASE_SUPPORT.local

The definition for SUPPORT normally points to the directory where the areaDetector, asyn, and the synApps modules (autosave, busy, calc, etc.) are located.

If using the EPICS Debian package:

- SUPPORT should be defined to be the root location of any modules which should **not** come from the Debian package. Do not define SUPPORT to be the location of the Debian packages.
- In the SUPPORT tree for those modules not from the Debian distribution (e.g. asyn) their configure/RELEASE files should also not define anything except EPICS_BASE to point to the location of the Debian package. For example if building asyn from source in the SUPPORT tree and IPAC and SNCSEQ from the Debian package then comment out the lines for IPAC and SNCSEQ in asyn/configure/RELEASE.

Optionally create RELEASE_SUPPORT.local.$(EPICS_HOST_ARCH)

Some installations chose to build for multiple target architectures using different development machines in the same directory tree on a file server. In this case the path to SUPPORT may be different for each architecture. For example SUPPORT on Linux might be `/home/epics/epics/support`, while on a Windows machine using the same copy of support the path might be `J:/epics/support`. In this case RELEASE_SUPPORT.local could specify the path for Linux while RELEASE_SUPPORT.local.win32-x86 could specify the path for the win32-x86 build host. RELEASE_SUPPORT.local is read first, and then any definitions there will be replaced by RELEASE_SUPPORT.local.$(EPICS_HOST_ARCH) if it exists.

Optionally create RELEASE_BASE.local.$(EPICS_HOST_ARCH)

If the path to EPICS_BASE is different for a specific EPICS_HOST_ARCH from the one defined in RELEASE_LIBS.local and RELEASE_PRODS.local then it can be defined in this file. This is typically used only if building Windows and Linux in the same directory tree.

Edit RELEASE_LIBS.local

The location of ASYN, AREA_DETECTOR and EPICS_BASE must be specified.
asyn and areaDetector are normally placed in the SUPPORT directory defined in RELEASE_SUPPORT.local.

As described above RELEASE_LIBS.local.$(EPICS_HOST_ARCH) can be used if the ASYN version or path is different for a specific target architecture. This is usually not necessary even for building Linux and Windows in the same tree, because only the definitions of SUPPORT in RELEASE_SUPPORT.local.$(EPICS_HOST_ARCH) and EPICS_BASE in RELEASE_BASE.local.$(EPICS_HOST_ARCH) need to be changed.

If WITH_PVA=YES is defined in CONFIG_SITE.local and EPICS_BASE version is prior to 7.0 then PVA must define the location of the EPICS PVA (formerly EPICS V4) libraries. Beginning with EPICS base 7.0 the PVA files are in EPICS base and PVA should not be defined.

If using Debian packages then the following must be done:

- SUPPORT should be defined to be the root location of any modules which should **not** come from the Debian package.
- Any modules which should come from the Debian package should be commented out, except for EPICS_BASE.
- For example to use a newer version of asyn and areaDetector then define ASYN, AREA_DETECTOR, ADCORE, and ADSUPPORT here. To use the Debian version of asyn then comment out ASYN here.

Edit RELEASE_PRODS.local

See the notes for RELEASE_LIBS.local above.

The definitions for AUTOSAVE, BUSY, CALC, and SSCAN must be specified. If the CALC module is built with SNCSEQ support then SNCSEQ must also be specified. If DEVIOCSTATS or ALIVE are defined in RELEASE_PRODS.local then IOC applications will be built with these modules as well.

If WITH_PVA=YES is defined in CONFIG_SITE.local and EPICS_BASE version is prior to 7.0 then PVA must define the location of the EPICS PVA (formerly EPICS V4) libraries. Beginning with EPICS base 7.0 the PVA files are in EPICS base and PVA should not be defined.

If using Debian packages then the following must be done:

- SUPPORT should be defined to be the root location of any modules which should **not** come from the Debian package.
- Any modules which should come from the Debian package should be commented out, except for EPICS_BASE.
- For example to use a newer version of asyn and areaDetector then define ASYN, AREA_DETECTOR, ADCORE, and ADSUPPORT here, but comment out AUTOSAVE, BUSY, etc. because they come from the Debian package.

Edit CONFIG_SITE.local and optionally CONFIG_SITE.local.$(EPICS_HOST_ARCH)

This file is used to define which support libraries are to be used, and if a library is to be used then where it should be found. The following definitions are needed.

```
WITH_BOOST=YES or NO
```

boost is needed only to built the test programs in ADCore/ADApp/pluginTests. If WITH_BOOST=YES then BOOST_INCLUDE and BOOST_LIB can be defined to point to the locations of the boost library. If the boost include and library file are located in default system locations then BOOST_INCLUDE and BOOST_LIB should not be defined.

```
WITH_PVA = YES or NO
```

EPICS PVA (formerly V4) libraries are needed for the NDPluginPVA in ADCore and the pvaDriver repository. To build these components set WITH_PVA=YES. If using a version of EPICS_BASE prior to 7.0 then define the location of the PVA libraries in RELEASE_LIBS.local and RELEASE_PRODS.local. If using EPICS_BASE 7.0 or later it is not necessary to define PVA in these files because the PVA files are located in EPICS_BASE.

- NETCDF JPEG, TIFF, ZLIB, SZIP, XML2, HDF5, NEXUS, GRAPHICSMAGICK, OPENCV
  - NETCDF is required for the NDFileNetCDF plugin
  - JPEG is required for the NDFileJPEG and GraphicsMagick

    - TIFF is required for the NDFileTIFF and GraphicsMagick
    - ZLIB is required for the NDFileTIFF and NDFileHDF5 plugins
    - XML2 is required for ADCore
    - HDF5 is required for the NDFileHDF5 and NDFileNexus plugins
    - NEXUS is required for the NDFileNexus plugin
    - GRAPHICSMAGICK is required for the NDFileMagick plugin and the ADURL driver
    - OPENCV is required for the ADPluginEdge plugin

For each library XXX (XXX=TIFF, NETCDF, etc.) there are 4 Makefile variables that can be defined in CONFIG_SITE.local.

- WITH_XXX
    - If WITH_XXX=YES then build the plugins and drivers that require this library.
    - If XXX_EXTERNAL=NO then also build the source code for this library in ADSupport.
- XXX_EXTERNAL
    - If NO then build the source code for this library in ADSupport.
    - If YES then this library is installed external to areaDetector
- XXX_INCLUDE
    - If XXX_EXTERNAL=YES then this is the path to the include files for XXX. However, if XXX is a system library whose include files are in a standard include search path then do not define XXX_INCLUDE.
- XXX_LIB
    - If XXX_EXTERNAL=YES then this is the path to the library files for XXX. However, if XXX is a system library whose library files in a standard library search path then do not define XXX_LIB.

XML2 is an exception. It is required, so WITH_XML2 is not supported, but XML2_EXTERNAL, XML2_INCLUDE, and XML2_LIB are supported.

Note that there are some library interdependencies.

- If WITH_TIFF=YES then WITH_ZLIB must also be YES.
- If WITH_HDF5=YES then WITH_ZLIB and WITH_SZIP must also be YES.
- If WITH_NEXUS=YES then WITH_HDF5 must also be YES.

CONFIG_SITE.local contains the following lines at the end:

```
# The definitions above can be overridden in the following files.
# The files are searched in this order, with the last definition being used.
#    CONFIG_SITE.local.$(OS_CLASS)
#    CONFIG_SITE.local.$(EPICS_HOST_ARCH)
#    CONFIG_SITE.local.$(EPICS_HOST_ARCH).$(T_A)
```

Thus if multiple architectures are being built in the same tree the settings can be different for each OS_CLASS, EPICS_HOST_ARCH, or EPICS_HOST_ARCH.T_A.

Edit RELEASE.local

Uncomment the lines for the drivers that should be built. None of the detector drivers are included by default. Some detectors cannot be built on all systems. For example the Roper driver can only be built on Windows systems with the Princeton Instruments WinView or WinSpec programs installed, and the Point Grey driver can currently only be built on Linux systems if the version of libc.so is 2.14 or greater.

RELEASE.local.$(EPICS_HOST_ARCH) can be used to build different drivers on different EPICS_HOST_ARCH builds in the same tree.

make

Just type:

```
make
```

If this fails then some required products have probably not been installed.

Example files in ADCore/iocBoot

Copy EXAMPLE_commonPlugins.cmd to commonPlugins.cmd and EXAMPLE_commonPlugin_settings.req to commonPlugin_settings.req.

```
cp EXAMPLE_commonPlugins.cmd           commonPlugins.cmd
cp EXAMPLE_commonPlugin_settings.req   commonPlugin_settings.req
```

Edit commonPlugins.cmd and commonPlugin_settings.req. Change whether or not the lines for optional modules (e.g. DEVIOCSTATS, ALIVE) are commented out depending on whether these modules were defined in RELEASE_PRODS.local.

Run SimDetector

```
cd ADSimDetector/iocs/simDetectorIOC/iocBoot/iocSimDetector
### Edit Makefile to set ARCH to your $(EPICS_HOST_ARCH) architecture
make
../../bin/linux-x86_64/simDetectorApp st.cmd

### If you want to be able to easily run Linux and Windows in the same tree do the following:
###   Set ARCH in Makefile for Linux, run make on the Linux machine, and copy envPaths to envPaths.linux
###   Set ARCH in Makefile for Windows, run make on the Windows machine, and copy envPaths to envPaths.windows
### Start the IOC for Linux:
../../bin/linux-x86_64/simDetectorApp st.linux
### Start the IOC for Windows:
../../bin/windows-x64/simDetectorApp st.windows
```

Installing Pre-Built Binary Versions of areaDetector

Pre-built binary versions of areaDetector can be provided for each detector. This is provided as a convenience so that it is not necessary to set up a build system to run areaDetector on a specific detector.

The instructions here use the Pilatus (ADPilatus) module as an example.
Substitute Pilatus with the name of the detector you are working with.

The pre-built binaries can be found on the CARS Web site. There is a subdirectory there for each detector (e.g. ADPilatus) that contains releases for that detector. If you don't see a pre-built package for the detector you are looking for send an e-mail to Mark Rivers and I can create one for you.

The pre-built binaries contain executables for one or more of the following architectures:

- linux-x86 (32-bit Linux built on Centos7, gcc 4.8.5, libc 2.17)
- linux-x86_64 (64-bit Linux built on Centos7, gcc 4.8.5, libc 2.17)
- linux-x86_rhel6 (32-bit Linux build on RHEL6, gcc 4.4.7, libc 2.12)
- linux-x86_64-gcc42 (64-bit Linux built on SUSE, gcc 4.2.1, libc 2.6.1)
- darwin-x86 (64-bit Mac OSX built on Darwin 11.4.2, ??, clang 4.2)
- win32-x86-static (32-bit Windows, VS2010 compiler, statically linked)
- win32-x86 (32-bit Windows, VX2010 compiler, dyanamically linked)
- windows-x64-static (64-bit Windows, VS2010 compiler, statically linked)
- windows-x64 (64-bit Windows, VX2010 compiler, dyanamically linked)

Note that the linux-x86 and linux-x86_64 builds are done a relatively new Linux system and will not run on RHEL 6, for example. The linux-x86-rhel6 build will run on RHEL 6. The linux-x86_64-gcc42 build uses a very old version of libc and should run on most Linux systems.

Follow these steps to use the prebuilt version.

- Create an installation directory for the module. On Windows I typically use C:\EPICS\support. On Linux I typically use /home/ACCOUNT/epics/support, where ACCOUNT is the name of the account that is normally used to run the detector software, e.g. marccd on a marCCD detector, mar345 on a mar345 detector, det on a Pilatus detector, etc.
- Place the distribution file in this directory. Then issue the commands (Unix style)

  ```
  tar xvzf ADPilatus_RX-Y.tgz
  ```

  On Windows it is more convenient to download the zip file and extract it using Windows Explorer.
- In the ADPilatus/iocs/pilatusIOC/iocBoot/ directory make a *copy* of the example iocPilatus directory and give it a new local name, e.g. ioc13Pilatus1. By doing this you will be able to update to later versions of areaDetector without overwriting modifications you make in the ioc13Pilatus1 directory.

- In the new io13Pilatus1 directory you just created edit st.cmd to change the PV prefix $(PREFIX) to one that is unique to your site. PV prefixes must be unique on the subnet, and if you use the default prefix there could be a conflict with other detectors of the same type.
- In the same ioc13Pilatus1 directory edit the file envPaths to point to the locations of all of the support modules on your system. Normally this is handled by the EPICS build system, but when using the prebuilt version this must be manually edited. Do not worry about the path to EPICS_BASE, it is not required.

Display Managers

A display manager is needed to view the areaDetector control screens. Control screens are provided for the following display managers: MEDM, EDM, CSS, and caQtDM. The native screens are created manually using MEDM. The EDM, CSS and caQtDM screens are converted from the MEDM screens using conversion utilities. These are discussed in a later section.

In order to control the detectors and the plugins you should install one or more of MEDM, EDM, CSS, caQtDM.

MEDM

The source code for medm can be downloaded from: medm

This requires Motif. medm can be built from source on Linux if the Motif library is available (which it is not for some new releases, such as Fedora 20).

It is available for Windows as via an EPICS Windows Tools MSI installer package.

EDM

This can be downloaded through links on the EDM home page.

CSS

This can be downloaded through links on the CSS home page.

caQtDM

This can be downloaded through links on the caQtDM home page.

Configuration

Before running an areaDetector application it is usually necessary to configure a number of items.

- EPICS environment variables. There are several environment variables that EPICS uses. I suggest setting these in the .cshrc (or .bashrc) file for the account that will be used to run the detector.
  - EPICS_CA_AUTO_ADDR_LIST and EPICS_CA_ADDR_LIST. These variables control the IP addresses that EPICS clients use when searching for EPICS PVs. The default is EPICS_CA_AUTO_ADDR_LIST=YES and EPICS_CA_ADDR_LIST to be the broadcast address of all networks connected to the host. Some detectors, for example the marCCD and mar345, come with 2 network cards, which are on 2 different subnets, typically a private one connected to the detector and a public one connected to the local LAN. If the default value of these variables is used then EPICS clients (e.g. medm) running on the detector host computer will generate many errors because each EPICS PV will appear to be coming from both networks. The solution is to set these variables as follows:

    ```
    setenv EPICS_CA_AUTO_ADDR_LIST NO

    setenv EPICS_CA_ADDR_LIST localhost:XX.YY.ZZ.255
    ```

    where XX.YY.ZZ.255 should be replaced with the broadcast address for the public network on this computer.
  - EPICS_CA_MAX_ARRAY_BYTES. This variable controls the maximum array size that EPICS can transmit with Channel Access. The default is only 16kB, which is much too small for most detector data. This value must be set to a large enough value on both the EPICS server computer (e.g. the one running the areaDetector IOC) and client computer (e.g. the one running medm, ImageJ, IDL, etc.). This should be set to a value that is larger than the largest waveform record that will be used for the detector. For example if using a detector with 1024x1024 pixels and 4-bytes per pixel (waveform record FTVL=LONG) then EPICS_CA_MAX_ARRAY_BYTES would need to be at least 1024 * 1024 * 4 = 4153344.
    In practice it should be set at least 100 bytes larger than this because there is some overhead. For example:

```
setenv EPICS_CA_MAX_ARRAY_BYTES 4154000
```

Do not simply set EPICS_CA_MAX_ARRAY_BYTES to a very large number like 100MB or 1GB. EPICS Channel Access allocates buffers of exactly EPICS_CA_MAX_ARRAY bytes whenever the required buffer size exceeds 16 kB, and one does not want unnecessarily large buffers to be allocated.

- EPICS_DISPLAY_PATH. This variable controls where medm looks for .adl display files. If the recommendation below is followed to copy all adl files to a single directory, then this environment variable should be defined to point to that directory. For example:

```
setenv EPICS_DISPLAY_PATH /home/det/epics/adls
```

- medm display files. It is convenient to copy all medm .adl files to a single directory and then point the environment variable EPICS_DISPLAY_PATH to this directory. The alternative is to point EPICS_DISPLAY_PATH to a long list of directories where the adl files are located in the distributions, which is harder to maintain. On the Pilatus, for example, create a directory called /home/det/epics/adls, and put all of the adl files there. To simplify copying the adl files to that location use the following one-line script, which can placed in /home/det/bin/sync_adls.

```
find /home/det/epics/support -name '*.adl' -exec cp -fv {} /home/det/epics/adls ;
```

This script finds all adl files in the epics/support tree and copies them to /home/det/epics/adls. That directory must be created before running this script. Similar scripts can be used for other Linux detectors (marCCD, mar345, etc.) and can be used on Windows as well if Cygwin is installed. Each time a new release of areaDetector is installed remove the old versions of each support module (areaDetector, asyn, autosave, etc.) and then run this script to install the latest medm files.

Running the IOC Application

Each example IOC directory comes with a Linux script (start_epics) or a Windows batch file (start_epics.bat) or both depending on the architectures that the detector runs on. These scripts provide simple examples of how to start medm and the EPICS IOC. For example, for the mar345 iocBoot/iocMAR345/start_epics contains the following:

```
medm -x -macro "P=13MAR345_1:, R=cam1:" mar345.adl &
../../bin/linux-x86/mar345App st.cmd
```

This script starts medm in execute mode with the appropriate medm display file and macro parameters, running it in the background. It then runs the IOC application. This script assumes that iocBoot/iocMAR345 is the default directory when it is run, which could be added to the command or set in the configuration if this script is set as the target of a desktop shortcut, etc. The script assumes that EPICS_DISPLAY_PATH has been defined to be a location where the mar345.adl and related displays that it loads can be found. You will need to edit the script in your copy of the iocXXX directory to change the prefix (P) from 13MAR345_1: to whatever prefix you chose for your IOC. The start_epics script could also be copied to a location in your PATH (e.g. /home/mar345/bin/start_epics). Add a command like

```
cd /home/mar345/epics/support/areaDetector-2-0/ADmar345/iocs/mar345IOC/iocBoot/iocMAR345
```

at the beginning of the script and then type

```
start_epics
```

from any directory to start the EPICS IOC.

Image Viewers

The areaDetector/ADViewers repository comes with tools to display images over EPICS Channel Access using ImageJ and IDL. These viewers are described in areaDetectorViewers.html.

HDFView can be used to view files saved with the HDF5 file writing plugins. ImageJ can also be used to view files saved with the TIFF, JPEG, and netCDF plugins.