1. install

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.（我们的是areaDetector R2-0，因此需要安装TIFF, ZLIB, JPEG, SZIP, XML2, HDF5）

On Linux and Darwin the libtiff, libjpeg, libxml2, and libz libraries often come already installed.

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

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（未下载）

ADPilatus, etc. （未下载）

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（没有从master拷贝的）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

cp EXAMPLE\_RELEASE\_PATH.local RELEASE\_PATH.local(说明文档上没有，但是编译的时候会提示报错。而且RELEASE里有include RELEASE\_PATH.local，故自己加上了，事实上RELEASE\_PATH.local只是包含了BASE、AREADETECTOR及EV4\_BASE的地址，BASE、AREADETECTOR的地址在RELEASE\_LIBS.local里面都有定义，是否可以将RELEASE include $(TOP)/configure/RELEASE\_PATH.local注释掉？另外，RELEASE\_PRODS.local对autosave、busy、calc等等都有定义，为何在RELEASE里面没有include RELEASE\_PRODS.local??Makefile和RELEASE的关系是什么？)

附RELEASE内容：

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.

错误2：

https://github.com/areaDetector/ADSupport/blob/a6e6fa23c2478baf57ff63c122b93288447eafcb/supportApp/hdf5Src下载的hdfSrc里面居然没有hdf5.h，make的时候报错:

<https://www.hdfgroup.org/downloads/hdf5/source-code/>

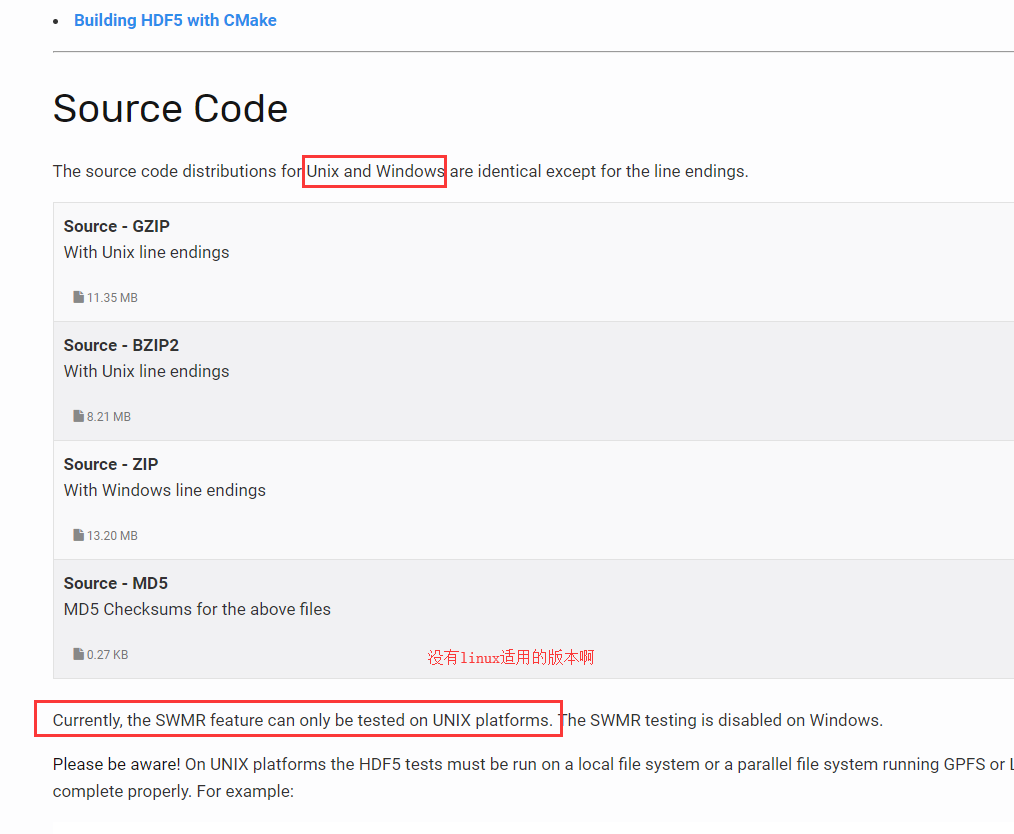


Figure ．<https://www.hdfgroup.org/downloads/hdf5/source-code/>官网

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.

照此，我的操作为，在Detector/configure下的CONFIG\_SITE.local改动如下：

原文：WITH\_HDF5 = YES

HDF5\_EXTERNAL=YES

改动后: WITH\_HDF5 = YES

HDF5\_EXTERNAL=NO

make，又报错，致命错误：szlib.h：没有那个文件或目录

再改CONFIG\_SITE.local

原文：WITH\_SZIP = YES

SZIP\_EXTERNAL = NO

make又报错，致命错误：libxml/parser.h没有那个文件或目录

百度给的报错原因如下：



我想知道，如果不安装，这个错误该怎么解决呢？（第一周的时候遇到了，没有记住。）

我是这样做的：

（1）ADCore/ADApp/ADSrc中的Makefile去掉黄色的部分（思路：既然xml没有安装，那xml的部分我给注释掉，不就解决了吗。PS:本身是仿真，没有必要装libxml2吧。但是libxml2和xml2应该是一回事吧。Orz。。。）编译依旧有错，错误提醒同上。

ADCore/ADApp/ADSrc中的Makefile内容

TOP=../..

include $(TOP)/configure/CONFIG

#----------------------------------------

# ADD MACRO DEFINITIONS AFTER THIS LINE

#=============================

# The following gets rid of the -fno-implicit-templates flag on vxWorks,

# so we get automatic template instantiation.

# This is what we want for miscellaneous/asynPortDriver.cpp

ifeq (vxWorks,$(findstring vxWorks, $(T\_A)))

CODE\_CXXFLAGS=

endif

# The following flag is need to compile/link NDArray.cpp on Solaris

ifeq ($(GNU),NO)

NDArray\_CXXFLAGS\_solaris += -features=tmplrefstatic

endif

DBD += ADSupport.dbd

INC += ADCoreVersion.h

INC += NDAttribute.h

INC += NDAttributeList.h

INC += NDArray.h

INC += PVAttribute.h

INC += paramAttribute.h

INC += functAttribute.h

INC += asynNDArrayDriver.h

INC += ADDriver.h

LIBRARY\_IOC = ADBase

LIB\_SRCS += NDAttribute.cpp

LIB\_SRCS += NDAttributeList.cpp

LIB\_SRCS += NDArrayPool.cpp

LIB\_SRCS += NDArray.cpp

LIB\_SRCS += asynNDArrayDriver.cpp

LIB\_SRCS += ADDriver.cpp

LIB\_SRCS += paramAttribute.cpp

ifeq ($(EPICS\_LIBCOM\_ONLY),YES)

USR\_CXXFLAGS += -DEPICS\_LIBCOM\_ONLY

else

LIB\_SRCS += PVAttribute.cpp

LIB\_SRCS += functAttribute.cpp

LIB\_SRCS += parseAreaPrefixes.c

LIB\_SRCS += myTimeStampSource.cpp

LIB\_SRCS += myAttributeFunctions.cpp

endif

LIB\_LIBS += asyn

ifeq ($(EPICS\_LIBCOM\_ONLY),YES)

LIB\_LIBS += Com

else

LIB\_LIBS += $(EPICS\_BASE\_IOC\_LIBS)

endif

ifeq ($(XML2\_EXTERNAL),NO)

LIB\_LIBS += xml2

else

ifdef XML2\_INCLUDE

USR\_INCLUDES += -I$(XML2\_INCLUDE)

endif

ifdef XML2\_LIB

xml2\_DIR = $(XML2\_LIB)

LIB\_LIBS += xml2

else

LIB\_SYS\_LIBS += xml2

endif

endif

# Install commonDriverMakefile and commonLibraryMakefile in top-level cfg directory.

# Drivers currently look in $(TOP)/ADApp for these files so we leave them there but

# need to install from this directory which requires using vpath.

vpath common%Makefile $(TOP)/ADApp/

CFG += commonLibraryMakefile

CFG += commonDriverMakefile

include $(TOP)/configure/RULES

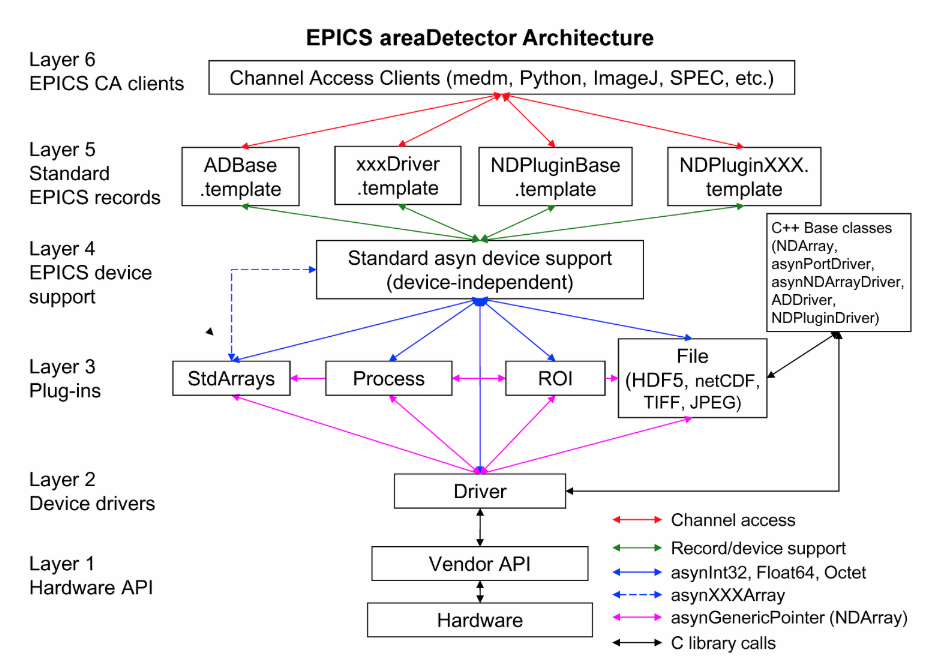
#----------------------------------------

# ADD RULES AFTER THIS LINE

（2）我又注释掉了红色标注的部分，make时依然有错，错误换了，内容为：NDluginDriver.h:10:31: 致命错误：asynNDArrayDriver.h没有那个文件或目录（当然没有了，我自己注释掉了）

（3）看来是PluginDriver处出了问题，因此，我又在/ADAPP/pluginSrc下的NDpluginDriver.h中将#include “asynNDArrayDriver.h”注释掉了（##include “asynNDArrayDriver.h”）

可以看出层层相关，由此对下图areaDetector的架构有了进一步的认识



（4）还是报错，错误入下：这次我将黄色标注的地方删了，删了，删了。它再错我要明天再弄了，orz……..心力衰竭

In file included from ../NDPluginDriver.cpp:26:0:

../NDPluginDriver.h:11:1: 错误：程序中有游离的‘##’

##include "asynNDArrayDriver.h"

../NDPluginDriver.h:11:3: 错误：‘include’不是一个类型名

##include "asynNDArrayDriver.h"

^

In file included from ../NDPluginDriver.cpp:26:0:

../NDPluginDriver.h:54:64: 错误：expected class-name before ‘,’ token

class epicsShareClass NDPluginDriver : public asynNDArrayDriver, public epicsThreadRunable {

^

../NDPluginDriver.h:76:35: 错误：‘NDArray’未声明

virtual void processCallbacks(NDArray \*pArray) = 0;

^

../NDPluginDriver.h:77:40: 错误：‘NDArray’未声明

virtual void beginProcessCallbacks(NDArray \*pArray);

^

../NDPluginDriver.h:78:44: 错误：‘NDArray’未声明

virtual asynStatus endProcessCallbacks(NDArray \*pArray, bool copyArray=false, bool readAttributes=true);

^

../NDPluginDriver.h:104:5: 错误：‘NDArray’不是一个类型名

NDArray \*pPrevInputArray\_;

^

../NDPluginDriver.h:122:5: 错误：‘asynGenericPointer’不是一个类型名

asynGenericPointer \*pasynGenericPointer\_; /\*\*< asyn interface for connecting to NDArray driver \*/

^

../NDPluginDriver.h:124:5: 错误：‘vector’不是命名空间‘std’中的一个类型名

std::vector<epicsThread\*>pThreads\_;

^

../NDPluginDriver.h:127:19: 错误：‘sortedListElement’在此作用域中尚未声明

std::multiset<sortedListElement> sortedNDArrayList\_;

^

../NDPluginDriver.h:127:36: 错误：模板第 1 个参数无效

std::multiset<sortedListElement> sortedNDArrayList\_;

^

../NDPluginDriver.h:127:36: 错误：模板第 2 个参数无效

../NDPluginDriver.h:127:36: 错误：模板第 3 个参数无效

../NDPluginDriver.h:131:19: 错误：‘ND\_ARRAY\_MAX\_DIMS’在此作用域中尚未声明

int dimsPrev\_[ND\_ARRAY\_MAX\_DIMS];

^

../NDPluginDriver.cpp:35:5: 错误：‘NDArray’不是一个类型名

NDArray \*pArray;

^

../NDPluginDriver.cpp:50:1: 错误：‘sortedListElement’不是一个类型名

sortedListElement::sortedListElement(NDArray \*pArray, epicsTimeStamp time)

^

../NDPluginDriver.cpp: 在构造函数‘NDPluginDriver::NDPluginDriver(const char\*, int, int, const char\*, int, int, int, size\_t, int, int, int, int, int, int, int)’中:

../NDPluginDriver.cpp:92:7: 错误：类‘NDPluginDriver’没有名为‘asynNDArrayDriver’的字段

: asynNDArrayDriver(portName, maxAddr, maxBuffers, maxMemory,

^

../NDPluginDriver.cpp:93:27: 错误：‘asynInt32Mask’在此作用域中尚未声明

interfaceMask | asynInt32Mask | asynFloat64Mask | asynOctetMask | asynInt32ArrayMask | asynDrvUserMask,

^

../NDPluginDriver.cpp:93:43: 错误：‘asynFloat64Mask’在此作用域中尚未声明

interfaceMask | asynInt32Mask | asynFloat64Mask | asynOctetMask | asynInt32ArrayMask | asynDrvUserMask,

^

../NDPluginDriver.cpp:93:61: 错误：‘asynOctetMask’在此作用域中尚未声明

interfaceMask | asynInt32Mask | asynFloat64Mask | asynOctetMask | asynInt32ArrayMask | asynDrvUserMask,

^

../NDPluginDriver.cpp:93:77: 错误：‘asynInt32ArrayMask’在此作用域中尚未声明

interfaceMask | asynInt32Mask | asynFloat64Mask | asynOctetMask | asynInt32ArrayMask | asynDrvUserMask,

^

../NDPluginDriver.cpp:93:98: 错误：‘asynDrvUserMask’在此作用域中尚未声明

interfaceMask | asynInt32Mask | asynFloat64Mask | asynOctetMask | asynInt32ArrayMask | asynDrvUserMask,

^

../NDPluginDriver.cpp:96:5: 错误：类‘NDPluginDriver’没有名为‘pPrevInputArray\_’的字段

pPrevInputArray\_(0),

^

../NDPluginDriver.cpp:107:10: 错误：‘lock’在此作用域中尚未声明

lock();

^

../NDPluginDriver.cpp:111:19: 错误：‘class NDPluginDriver’没有名为‘dimsPrev\_’的成员

memset(&this->dimsPrev\_, 0, sizeof(this->dimsPrev\_));

^

../NDPluginDriver.cpp:111:46: 错误：‘class NDPluginDriver’没有名为‘dimsPrev\_’的成员

memset(&this->dimsPrev\_, 0, sizeof(this->dimsPrev\_));

^

../NDPluginDriver.cpp:112:11: 错误：‘class NDPluginDriver’没有名为‘pasynGenericPointer\_’的成员

this->pasynGenericPointer\_ = NULL;

^

../NDPluginDriver.cpp:123:46: 错误：‘NDArrayData’在此作用域中尚未声明

this->pasynUserGenericPointer\_->reason = NDArrayData;

^

../NDPluginDriver.cpp:125:56: 错误：‘asynParamOctet’在此作用域中尚未声明

createParam(NDPluginDriverArrayPortString, asynParamOctet, &NDPluginDriverArrayPort);

^

../NDPluginDriver.cpp:125:96: 错误：‘createParam’在此作用域中尚未声明

createParam(NDPluginDriverArrayPortString, asynParamOctet, &NDPluginDriverArrayPort);

^

../NDPluginDriver.cpp:126:56: 错误：‘asynParamInt32’在此作用域中尚未声明

createParam(NDPluginDriverArrayAddrString, asynParamInt32, &NDPluginDriverArrayAddr);

^

../NDPluginDriver.cpp:134:56: 错误：‘asynParamFloat64’在此作用域中尚未声明

createParam(NDPluginDriverSortTimeString, asynParamFloat64, &NDPluginDriverSortTime);

^

../NDPluginDriver.cpp:151:57: 错误：‘setStringParam’在此作用域中尚未声明

setStringParam (NDPluginDriverArrayPort, NDArrayPort);

^

../NDPluginDriver.cpp:152:57: 错误：‘setIntegerParam’在此作用域中尚未声明

setIntegerParam(NDPluginDriverArrayAddr, NDArrayAddr);

^

../NDPluginDriver.cpp:168:12: 错误：‘unlock’在此作用域中尚未声明

unlock();

^

../NDPluginDriver.cpp: 在析构函数‘virtual NDPluginDriver::~NDPluginDriver()’中:

../NDPluginDriver.cpp:178:9: 错误：‘class NDPluginDriver’没有名为‘lock’的成员

this->lock();

^

../NDPluginDriver.cpp:180:9: 错误：‘class NDPluginDriver’没有名为‘unlock’的成员

this->unlock();

^

../NDPluginDriver.cpp: 在全局域：

../NDPluginDriver.cpp:190:48: 错误：变量或字段‘beginProcessCallbacks’声明为 void

void NDPluginDriver::beginProcessCallbacks(NDArray \*pArray)

^

../NDPluginDriver.cpp:190:48: 错误：‘NDArray’在此作用域中尚未声明

../NDPluginDriver.cpp:190:57: 错误：‘pArray’在此作用域中尚未声明

void NDPluginDriver::beginProcessCallbacks(NDArray \*pArray)

^

../NDPluginDriver.cpp:48:20: 警告：‘driverName’定义后未使用 [-Wunused-variable]

static const char \*driverName="NDPluginDriver";

^

../NDPluginDriver.cpp:53:13: 警告：‘void sortingTaskC(void\*)’定义后未使用 [-Wunused-function]

static void sortingTaskC(void \*drvPvt)

（5）安装上述要求修改 ##include “asynNDArrayDriver.h”直接删除，其他按照要求也进行修改和删除

#ifndef NDPluginDriver\_H

#define NDPluginDriver\_H

#include <set>

#include <epicsTypes.h>

#include <epicsMessageQueue.h>

#include <epicsThread.h>

#include <epicsTime.h>

##include "asynNDArrayDriver.h"

// This class defines the object that is contained in the std::multilist for sorting output NDArrays

// It contains a pointer to the NDArray and the time that the object was added to the list

// It defines the < operator to use the NDArray::uniqueId field as the sort key

// We would like to hide this class definition in NDPluginDriver.cpp and just forward reference it here.

// That works on Visual Studio, and on gcc if instantiating plugins as heap variables with "new", but fails on gcc

// if instantiating plugins as automatic variables.

//class sortedListElement;

class sortedListElement {

public:

sortedListElement(NDArray \*pArray, epicsTimeStamp time);

friend bool operator<(const sortedListElement& lhs, const sortedListElement& rhs) {

return (lhs.pArray\_->uniqueId < rhs.pArray\_->uniqueId);

}

NDArray \*pArray\_;

epicsTimeStamp insertionTime\_;

};

#define NDPluginDriverArrayPortString "NDARRAY\_PORT" /\*\*< (asynOctet, r/w) The port for the NDArray interface \*/

#define NDPluginDriverArrayAddrString "NDARRAY\_ADDR" /\*\*< (asynInt32, r/w) The address on the port \*/

#define NDPluginDriverPluginTypeString "PLUGIN\_TYPE" /\*\*< (asynOctet, r/o) The type of plugin \*/

#define NDPluginDriverDroppedArraysString "DROPPED\_ARRAYS" /\*\*< (asynInt32, r/w) Number of dropped input arrays \*/

#define NDPluginDriverQueueSizeString "QUEUE\_SIZE" /\*\*< (asynInt32, r/w) Total queue elements \*/

#define NDPluginDriverQueueFreeString "QUEUE\_FREE" /\*\*< (asynInt32, r/w) Free queue elements \*/

#define NDPluginDriverMaxThreadsString "MAX\_THREADS" /\*\*< (asynInt32, r/w) Maximum number of threads \*/

#define NDPluginDriverNumThreadsString "NUM\_THREADS" /\*\*< (asynInt32, r/w) Number of threads \*/

#define NDPluginDriverSortModeString "SORT\_MODE" /\*\*< (asynInt32, r/w) sorted callback mode \*/

#define NDPluginDriverSortTimeString "SORT\_TIME" /\*\*< (asynFloat64, r/w) sorted callback time \*/

#define NDPluginDriverSortSizeString "SORT\_SIZE" /\*\*< (asynInt32, r/o) std::multiset maximum # elements \*/

#define NDPluginDriverSortFreeString "SORT\_FREE" /\*\*< (asynInt32, r/o) std::multiset free elements \*/

#define NDPluginDriverDisorderedArraysString "DISORDERED\_ARRAYS" /\*\*< (asynInt32, r/o) Number of out of order output arrays \*/

#define NDPluginDriverDroppedOutputArraysString "DROPPED\_OUTPUT\_ARRAYS" /\*\*< (asynInt32, r/o) Number of dropped output arrays \*/

#define NDPluginDriverEnableCallbacksString "ENABLE\_CALLBACKS" /\*\*< (asynInt32, r/w) Enable callbacks from driver (1=Yes, 0=No) \*/

#define NDPluginDriverBlockingCallbacksString "BLOCKING\_CALLBACKS" /\*\*< (asynInt32, r/w) Callbacks block (1=Yes, 0=No) \*/

#define NDPluginDriverProcessPluginString "PROCESS\_PLUGIN" /\*\*< (asynInt32, r/w) Process plugin with last callback array \*/

#define NDPluginDriverExecutionTimeString "EXECUTION\_TIME" /\*\*< (asynFloat64, r/o) The last execution time (milliseconds) \*/

#define NDPluginDriverMinCallbackTimeString "MIN\_CALLBACK\_TIME" /\*\*< (asynFloat64, r/w) Minimum time between calling processCallbacks

\* to execute plugin code \*/

/\*\* Class from which actual plugin drivers are derived; derived from asynNDArrayDriver \*/

class epicsShareClass NDPluginDriver : public asynNDArrayDriver, public epicsThreadRunable {

public:

NDPluginDriver(const char \*portName, int queueSize, int blockingCallbacks,

const char \*NDArrayPort, int NDArrayAddr, int maxAddr,

int maxBuffers, size\_t maxMemory, int interfaceMask, int interruptMask,

int asynFlags, int autoConnect, int priority, int stackSize, int maxThreads);

~NDPluginDriver();

/\* These are the methods that we override from asynNDArrayDriver \*/

virtual asynStatus writeInt32(asynUser \*pasynUser, epicsInt32 value);

virtual asynStatus writeOctet(asynUser \*pasynUser, const char \*value, size\_t maxChars,

size\_t \*nActual);

virtual asynStatus readInt32Array(asynUser \*pasynUser, epicsInt32 \*value,

size\_t nElements, size\_t \*nIn);

/\* These are the methods that are new to this class \*/

virtual void driverCallback(asynUser \*pasynUser, void \*genericPointer);

virtual void run(void);

virtual asynStatus start(void);

void sortingTask();

protected:

virtual void processCallbacks(NDArray \*pArray) = 0;

virtual void beginProcessCallbacks(NDArray \*pArray);

virtual asynStatus endProcessCallbacks(NDArray \*pArray, bool copyArray=false, bool readAttributes=true);

virtual asynStatus connectToArrayPort(void);

virtual asynStatus setArrayInterrupt(int connect);

protected:

int NDPluginDriverArrayPort;

#define FIRST\_NDPLUGIN\_PARAM NDPluginDriverArrayPort

int NDPluginDriverArrayAddr;

int NDPluginDriverPluginType;

int NDPluginDriverDroppedArrays;

int NDPluginDriverQueueSize;

int NDPluginDriverQueueFree;

int NDPluginDriverMaxThreads;

int NDPluginDriverNumThreads;

int NDPluginDriverSortMode;

int NDPluginDriverSortTime;

int NDPluginDriverSortSize;

int NDPluginDriverSortFree;

int NDPluginDriverDisorderedArrays;

int NDPluginDriverDroppedOutputArrays;

int NDPluginDriverEnableCallbacks;

int NDPluginDriverBlockingCallbacks;

int NDPluginDriverProcessPlugin;

int NDPluginDriverExecutionTime;

int NDPluginDriverMinCallbackTime;

NDArray \*pPrevInputArray\_;

private:

void processTask();

asynStatus createCallbackThreads();

asynStatus startCallbackThreads();

asynStatus deleteCallbackThreads();

asynStatus createSortingThread();

/\* The asyn interfaces we access as a client \*/

void \*asynGenericPointerInterruptPvt\_;

/\* Our data \*/

int numThreads\_;

bool pluginStarted\_;

bool firstOutputArray\_;

asynUser \*pasynUserGenericPointer\_; /\*\*< asynUser for connecting to NDArray driver \*/

void \*asynGenericPointerPvt\_; /\*\*< Handle for connecting to NDArray driver \*/

asynGenericPointer \*pasynGenericPointer\_; /\*\*< asyn interface for connecting to NDArray driver \*/

bool connectedToArrayPort\_;

std::vector<epicsThread\*>pThreads\_;

epicsMessageQueue \*pToThreadMsgQ\_;

epicsMessageQueue \*pFromThreadMsgQ\_;

std::multiset<sortedListElement> sortedNDArrayList\_;

int prevUniqueId\_;

epicsThreadId sortingThreadId\_;

epicsTimeStamp lastProcessTime\_;

int dimsPrev\_[ND\_ARRAY\_MAX\_DIMS];

};

#*e*ndif

---------------------------------------------------------------------

(5)错误跟刚才差不多。嗯嗯…..我直接在/ADApp/pluginSrc Makefile里面把NOPluginDriver.h和NOPluginDriver.cpp去掉了。再看。这次是真的心力衰竭了。

…………嗯，果不其然，又错了，我想把以上工作全抹掉，从头再来……嗯…..怎么说呢，压抑住这种想法…..不能浅尝辄止是不是…….淡定点，让那堆错误自生自灭去吧………..明天再改错。Orz…我的PPT还没搞。