BMeasure-lib

1.1.0

Generated by Doxygen 1.9.5

Chapter 1

BMeasure-lib

Author

Dr Terry Barnaby

Version

1.0.0

Date

2020-02-09

1.1 Introduction

The Beam BMeasure-125i unit is a flexible and powerful IoT system for data capture, data logging and control in the laboratory, industrial and remote sensing arenas. It is based around an 8 channel, fully differential, synchronous sampling, 24 bit ADC that can sample at speeds up to 128 ksps. Multiple units can be connected together to provide more synchronously sampled channels.

This reference information describes the data types and functions provided by the host API library allowing programs to be written to control the operation of a BMeasure unit and acquire the data from it. The API operates over a number of different physical interfaces including: USB 2.0, Ethernet, Wifi and RS485.

In addition there is a software manual providing an overview of using this API which should be read first. This document is available at:: `https://portal.beam.ltd.uk/files/products/bmeasure-125i/doc/BMeasure-lib.pdf`

1.2 Overview

The BMeasure API library, bmeasure-lib, is implemented in the C++ computer language. It has bindings layered on top of this for Python, with Matlab due to be supported soon. The API has an object orientated architecture. It has been designed as a general purpose API library for the Beam BMeasure-125i and future BMeasure products. Currently it has ports to Linux (Redhat7, Fedora29, Debian) and Microsoft Windows 7, 8 and 10.

The API provides the following functionality:

- Find BMeasure units on the USB bus or local Ethernet and Wifi networks.
- Connect to one or more BMeasure units.
- Fetch information and configure the BMeasure units.
- Start the BMeasure unit capturing and processing the sensor inputs.
- Capture the data from all of the analogue and digital channels from one or a combined set of BMeasure units running in sync.
- Access the data log files on the unit and download them to the host.
- Configure the AWG to produce waveforms or set voltages on the analogue output channels.
- Operate relays, read switches and other auxiliary operations.

The BMeasure API is implemented using the Beam BOAP (Beam Object Access protocol) communications system. It offers an BMeasureUnit API class to access an individual BMeasure unit in a relatively low level manner and an BMeasureUnits API class to access a set of BMeasure units synchronised together to operate as a single unit and with a queued data reception system..

The API supports threaded and non-threaded operation.

The referenve information provided describes the API from a C++ programming perspective. The Python and other language bindings are very similar the differences being noted under the particular language bindings section in the software manual..

1.3 API Usage

To use the API the core procedure is:

1. Either find the available BMeasure units using: BMeasureApi::BMeasureUnit::findDevices() or use a BMeasure URL string..
2. Choose to use the simple single unit interface BMeasureApi::BMeasureUnit or the BMeasureApi::BMeasureUnits classes.
3. If using the simple single unit interface, connect to the unit using the BMeasureApi::BMeasureUnit::connect() function.
4. If using the multiple unit interface, add the units using the BMeasureApi::BMeasureUnits::unitAdd() function and connect using the BMeasureApi::BMeasureUnits::unitsConnect() function.
5. Use the interface to communicate to the unit.

See the examples below and the software manual for more details.

1.4 API Usage

There are some examples of client applications using the BMeasure API in the **examples** directory of the source code. Some simple client examples are listed below:

Simple example to access and read single sets of data samples in C++

```
/*******************************************************************************
 *      Example005-dataClient-single.cpp
 *      T.Barnaby,      BEAM Ltd,       2019-10-09
 *******************************************************************************
 */
#include <BMeasureUnit.h>
#include <unistd.h>
using namespace BMeasureApi;
// Function to read some data
BError test1(){
        BError                            err;
        BList<BMeasureUnitDevice>      devices;
        BString                          device;
        BMeasureUnit                        bmeasure;
        Information                   info;
        Configuration                      config;
        MeasurementConfig              mc;
        DataBlock                      data;
        BUInt                              c;

        printf("Start Processing Task\n");
        bmeasure.start();

        printf("Find BMeasure units\n");
        if(err = BMeasureUnit::findDevicesUsb(devices)){
                return err;
        }
        if(devices.number() == 0){
                return err.set(1, "No USB BMeasure units found");
        }
        device = devices[0].device;

        printf("Connect\n");
        if(err = bmeasure.connect(device))
                return err;

        //printf("Exit\n"); return err;
        printf("Get Info\n");
        if(err = bmeasure.getInformation(info))
                return err;

        printf("NumChannels: %d\n", info.numChannels);
        //printf("Exit\n"); return err;
        printf("Configure measurement\n");
        mc.measureMode = MeasureModeOneShot;
        mc.triggerMode = TriggerModeOff;
        mc.triggerConfig = TriggerConfigNone;
        mc.triggerChannel = 0;
        mc.triggerLevel = 0;
        mc.triggerDelay = 0;
        mc.sampleRate = 8000.0;
        mc.measurePeriod = 0;
        mc.numSamples0 = 1;
        mc.numSamples1 = 0;
        if(err = bmeasure.setMeasurementConfig(0, mc))
                return err;

        printf("Run single measurement\n");
        if(err = bmeasure.measure(DataTypeFloat32, data))
                return err;

        printf("DataBlock: from: %d numChannels: %d numSamples: %d\n", data.source, data.numChannels,
       data.numSamples);
        for(c = 0; c < data.numChannels; c++){
                printf("%f ", data.data[c]);
        }
        printf("\n");
        return err;
}
int main(){
        BError    err;
        if(err = test1()){
                printf("Error: %d %s\n", err.getErrorNo(), err.str());
                return 1;
        }
        printf("Complete\n");
        return 0;
}
```

Simple example to access and read single sets of data samples in Python

```
#!/usr/bin/python3
import sys
import time
import getopt
from threading import Thread
from bmeasure import *
# Function to read some data
def test1():
        bmeasure = BMeasureUnit(True);
        print("Find BMeasure units");
        (err, devices) = BMeasureUnit.findDevicesUsb();
        if(err):
                return err;
        if(devices.number() == 0):
                return err.set(1, "No USB BMeasure units found\n");
        print("Found", len(devices));
        device = devices[0].device;
        print("Start Processing Task");
        bmeasure.start();
        print("Connect to BMeasure");
        err = bmeasure.connect(device);
        if(err):
                return err;
        print("Get Info");
        (err, info) = bmeasure.getInformation();
        if(err):
                return err;

        print("NumChannels: ", info.numChannels);
        print("Configure measurement");
        mc = MeasurementConfig();
        mc.measureMode = MeasureModeOneShot;
        mc.triggerMode = TriggerModeOff;
        mc.triggerConfig = TriggerConfigNone;
        mc.triggerChannel = 0;
        mc.triggerLevel = 0;
        mc.triggerDelay = 0;
        mc.sampleRate = 4000;
        mc.numSamples0 = 1;
        mc.numSamples1 = 0;
        mc.measurePeriod = 0;
        err = bmeasure.setMeasurementConfig(False, mc);
        if(err):
                return err;

        print("Run single measurement");
        (err, data) = bmeasure.measure();
        if(err):
                return err;
        print("DataBlock: from: %d numChannels: %d numSamples: %d" % (data.source, data.numChannels,
       data.numSamples));
        for c in range(0, data.numChannels):
                print("Chan:", c, data.data[c]);
        return err;
def main():
        err = test1();
        if(err):
                print("Error:", err.getErrorNo(), err.getString());
                return 1;
        print("Complete");

        return 0;
if __name__ == "__main__":
        main();
```

Simple example to show operating the relays in Python

```
#!/usr/bin/python3
import sys
import time
import getopt
from threading import Thread
from bmeasure import *
# Function to set the relays on/off
def test1():
        bmeasure = BMeasureUnit(True);
        print("Find BMeasure units");
        (err, devices) = BMeasureUnit.findDevicesUsb();
        if(err):
                return err;
        if(devices.number() == 0):
                return err.set(1, "No USB BMeasure units found\n");
        print("Found", len(devices));
        device = devices[0].device;
        print("Start Communications Task");
        bmeasure.start();
        print("Connect");
        err = bmeasure.connect(device);
        if(err):
                return err;
        print("Get Info");
        (err, info) = bmeasure.getInformation();
        if(err):
                return err;

        print("NumChannels: ", info.numChannels);
        # Toggle relay1
        state = 0;
        for i in range(0, 6):
                if(state):
                        state = False;
                else:
                        state = True;
                print("Set relay 0: %d" % (state));
                err = bmeasure.setRelay(0, state);
                if(err):
                        return err;

                time.sleep(1);
        return err;
def main():
        if(0):
                err = find();
                if(err):
                        print("Error:", err.getErrorNo(), err.getString());
                        return 1;
        err = test1();
        if(err):
                print("Error:", err.getErrorNo(), err.getString());
                return 1;
        print("Complete");

        return 0;
if __name__ == "__main__":
        main();
```

Chapter 2

Namespace Index

2.1 Namespace List

Here is a list of all namespaces with brief descriptions:

Chapter 3

Hierarchical Index

3.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

Chapter 4

Class Index

4.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

Chapter 5

File Index

5.1 File List

Here is a list of all files with brief descriptions:

Chapter 6

Namespace Documentation

6.1 BMeasureApi Namespace Reference

Classes

- class AlarmConfig
- class AwgConfig

  File information.
- class BMeasure
- class BMeasureUnit
- class BMeasureUnit1
- class BMeasureUnitDevice
- class BMeasureUnits
- class BMeasureUnitsDataBlock
- class BoardConfig
- class CalibrateInfo
- class ChannelConfig

  Channel configuration.
- class CommsNet
- class CommsSerial
- class CommsUsb
- class ConfigItem
- class Configuration
- class DataBlock

  Data Block. Data in floating point format for all channels.
- class DataBlockFloat

  Processed data for a channel.
- class DataBlockProc

  Info Block.
- class DataFile
- class DataProc

  Data Proc Block. Processd Data packed into bytestream based on sampleTypes.
- class FileData
- class FileInfo

  File information.
- class FilesysInfo
- class InfoBlock

AWG Configuration.

- class Information
- class MeasurementConfig

  Measurement config Data Block. Data packed into bytestream based on sampleTypes.
- class NodeInfo
- class NodeStatus
- class Version
- class WifiAccessPoint

Typedefs

- typedef **BArray**< ChannelConfig > ChannelConfigs

Enumerations

- enum ErrorNum { ErrorNumSystem = 64 , ErrorNumDataOverrun = 65 , ErrorNumToFast = 66 }
- enum NodeType { NodeTypeNone = 0 , NodeTypeBMeasure1 = 1 }
- enum SecurityMode { SecurityModeBasic , SecurityModeConfig , SecurityModeFull }
- enum Status {
  StatusNone = 0x00 , StatusError = 0x01 , StatusWarning = 0x02 , StatusRun = 0x04 ,
  StatusTriggerWait = 0x08 , StatusEnd0 = 0x10 , StatusEnd1 = 0x20 , StatusDataOverrun = 0x40 ,
  StatusFpgaOverrun = 0x80 , StatusAlarm = 0x0100 }
- enum Mode {
  ModeIdle = 0 , ModeRun = 1 , ModeRunProgram = 2 , ModeInternal = 3 ,
  ModeSleep = 4 , ModeDemo1 = 5 }
- enum BlockTypes { BlockTypeInfo = 0x424E4531 , BlockTypeData = 0x424E4532 }
- enum ChannelType {
  ChannelTypeNone = 0 , ChannelTypeAnalogueIn = 1 , ChannelTypeAnalogueOut = 2 , ChannelTypeDigitalIn = 3 ,
  ChannelTypeDigitalOut = 4 }
- enum SampleType {
  SampleTypeNone = 0 , SampleTypeBool = 1 , SampleTypeInt8 = 2 , SampleTypeInt16 = 3 ,
  SampleTypeInt32 = 4 , SampleTypeFloat32 = 5 , SampleTypeFloat64 = 6 }
- enum SyncMode { SyncModeOff = 0 , SyncModeMaster = 1 , SyncModeSlave = 2 }
- enum MeasureMode { MeasureModeOff = 0 , MeasureModeOneShot = 1 , MeasureModeRepeat = 2 , MeasureModeContinuous = 3 }
- enum MeasureOption { MeasureOptionNone = 0 , MeasureOptionProcess = 0x01 }
- enum TriggerMode { TriggerModeOff = 0 , TriggerModePositive = 1 , TriggerModeNegative = 2 }
- enum TriggerConfig { TriggerConfigNone = 0 }
- enum DigitalMode {
  DigitalModeInput = 0 , DigitalModeOutput = 1 , DigitalModeInOut = 2 , DigitalModeSyncMaster = 3 ,
  DigitalModeSyncSlave = 4 }
- enum AwgMode {
  AwgModeNone , AwgModeDc , AwgModeSine , AwgModeSquare ,
  AwgModeTriangle , AwgModeNoise , AwgModeTrackRms , AwgModeTrackMean ,
  AwgModeArbitrary }
- enum AwgOutput { AwgOutputNone , AwgOutputAO0 , AwgOutputAO1 , AwgOutputAO01 }
- enum FileType { FileTypeNone , FileTypeFile , FileTypeDir }
- enum FilesysDeleteType { FilesysDeleteTypeNone , FilesysDeleteTypeData , FilesysDeleteTypeFormat }
- enum LogData { LogDataOff , LogDataRaw = 0x01 , LogDataProcessed = 0x02 }
- enum LogDataMode { LogDataModeNormal , LogDataModeDeleteOld }
- enum DataType { DataTypeFloat32 , DataType125i , DataTypeProc }

- enum DataSend { DataSendOff , DataSendStatus = 0x01 , DataSendRaw = 0x02 , DataSendProcessed = 0x04 }
- enum CalibrateStage {
  CalibrateStageNone = 0 , CalibrateStageClear = 1 , CalibrateStageSettle = 2 , CalibrateStageAdcOffsets = 3 ,
  CalibrateStageDacOffsets = 4 , CalibrateStageDacScaling0 = 5 , CalibrateStageDacScaling1 = 6 , CalibrateStageAdcScaling = 7 ,
  CalibrateStageAttenScaling = 8 , CalibrationStageFiveVolts = 9 , CalibrateStageAdcScalingWithAtten = 10 , CalibrateStageChanClear = 11 ,
  CalibrateStageChanOffsets = 12 , CalibrateStageChanScaling = 13 }
- enum MessageSource {
  MessageSourceGeneral = 0 , MessageSourceDebug = 1 , MessageSourceTest = 2 , MessageSourceWifi = 3 ,
  MessageSourceWifiTest = 4 }
- enum NetworkMode { NetworkModeOff = 0 , NetworkModeDhcp = 1 , NetworkModeManual = 2 }
- enum WifiMode { WifiModeOff , WifiModeClient , WifiModeAp }
- enum AlarmMode {
  AlarmModeOff , AlarmModeHigh , AlarmModeLow , AlarmModeRange ,
  AlarmModeMagnitude }
- enum AlarmOutput {
  AlarmOutputOff , AlarmOutputDioHigh , AlarmOutputDioLow , AlarmOutputRelayOn ,
  AlarmOutputRelayOff }
- enum EventMode { EventModeOff , EventModeAlarm , EventModeSecond }
- enum Rs485Mode { Rs485ModeOff , Rs485ModeBoap }
- enum BMeasFileType { BMeasFileTypeBlock512 , BMeasFileTypeStream }
- enum WifiCmd {
  WifiCmdOff , WifiCmdOn , WifiCmdScan , WifiCmdConnect ,
  WifiCmdDisconnect }
- enum WifiStatus {
  WifiStatusOff , WifiStatusOn , WifiStatusConnecting , WifiStatusConnected ,
  WifiStatusConnectedInternet , WifStatusAP }
- enum TdsDataType {
  TdsTypeVoid , TdsTypeI8 , TdsTypeI16 , TdsTypeI32 ,
  TdsTypeI64 , TdsTypeU8 , TdsTypeU16 , TdsTypeU32 ,
  TdsTypeU64 , TdsTypeSingleFloat , TdsTypeDoubleFloat , TdsTypeExtendedFloat ,
  TdsTypeSingleFloatWithUnit =0x19 , TdsTypeDoubleFloatWithUnit , TdsTypeExtendedFloatWithUnit , TdsTypeString =0x20 ,
  TdsTypeBoolean =0x21 , TdsTypeTimeStamp =0x44 , TdsTypeFixedPoint =0x4F , TdsTypeComplexSingleFloat =0x08000c ,
  TdsTypeComplexDoubleFloat =0x10000d , TdsTypeDAQmxRawData =0xFFFFFFFF }

Functions

- **BString** toBString (ErrorNum v)
- **BError** fromBString (**BString** str, ErrorNum &v)
- **BString** toBStringJson (**BString** n, ErrorNum v)
- **BString** toBString (NodeType v)
- **BError** fromBString (**BString** str, NodeType &v)
- **BString** toBStringJson (**BString** n, NodeType v)
- **BString** toBString (SecurityMode v)
- **BError** fromBString (**BString** str, SecurityMode &v)
- **BString** toBStringJson (**BString** n, SecurityMode v)
- **BString** toBString (Status v)
- **BError** fromBString (**BString** str, Status &v)
- **BString** toBStringJson (**BString** n, Status v)

- **BString** toBString (Mode v)
- **BError** fromBString (**BString** str, Mode &v)
- **BString** toBStringJson (**BString** n, Mode v)
- **BString** toBString (BlockTypes v)
- **BError** fromBString (**BString** str, BlockTypes &v)
- **BString** toBStringJson (**BString** n, BlockTypes v)
- **BString** toBString (ChannelType v)
- **BError** fromBString (**BString** str, ChannelType &v)
- **BString** toBStringJson (**BString** n, ChannelType v)
- **BString** toBString (SampleType v)
- **BError** fromBString (**BString** str, SampleType &v)
- **BString** toBStringJson (**BString** n, SampleType v)
- **BString** toBString (SyncMode v)
- **BError** fromBString (**BString** str, SyncMode &v)
- **BString** toBStringJson (**BString** n, SyncMode v)
- **BString** toBString (MeasureMode v)
- **BError** fromBString (**BString** str, MeasureMode &v)
- **BString** toBStringJson (**BString** n, MeasureMode v)
- **BString** toBString (MeasureOption v)
- **BError** fromBString (**BString** str, MeasureOption &v)
- **BString** toBStringJson (**BString** n, MeasureOption v)
- **BString** toBString (TriggerMode v)
- **BError** fromBString (**BString** str, TriggerMode &v)
- **BString** toBStringJson (**BString** n, TriggerMode v)
- **BString** toBString (TriggerConfig v)
- **BError** fromBString (**BString** str, TriggerConfig &v)
- **BString** toBStringJson (**BString** n, TriggerConfig v)
- **BString** toBString (DigitalMode v)
- **BError** fromBString (**BString** str, DigitalMode &v)
- **BString** toBStringJson (**BString** n, DigitalMode v)
- **BString** toBString (AwgMode v)
- **BError** fromBString (**BString** str, AwgMode &v)
- **BString** toBStringJson (**BString** n, AwgMode v)
- **BString** toBString (AwgOutput v)
- **BError** fromBString (**BString** str, AwgOutput &v)
- **BString** toBStringJson (**BString** n, AwgOutput v)
- **BString** toBString (FileType v)
- **BError** fromBString (**BString** str, FileType &v)
- **BString** toBStringJson (**BString** n, FileType v)
- **BString** toBString (FilesysDeleteType v)
- **BError** fromBString (**BString** str, FilesysDeleteType &v)
- **BString** toBStringJson (**BString** n, FilesysDeleteType v)
- **BString** toBString (LogData v)
- **BError** fromBString (**BString** str, LogData &v)
- **BString** toBStringJson (**BString** n, LogData v)
- **BString** toBString (LogDataMode v)
- **BError** fromBString (**BString** str, LogDataMode &v)
- **BString** toBStringJson (**BString** n, LogDataMode v)
- **BString** toBString (DataType v)
- **BError** fromBString (**BString** str, DataType &v)
- **BString** toBStringJson (**BString** n, DataType v)
- **BString** toBString (DataSend v)
- **BError** fromBString (**BString** str, DataSend &v)
- **BString** toBStringJson (**BString** n, DataSend v)
- **BString** toBString (CalibrateStage v)

- **BError** fromBString (**BString** str, CalibrateStage &v)
- **BString** toBStringJson (**BString** n, CalibrateStage v)
- **BString** toBString (MessageSource v)
- **BError** fromBString (**BString** str, MessageSource &v)
- **BString** toBStringJson (**BString** n, MessageSource v)
- **BString** toBString (NetworkMode v)
- **BError** fromBString (**BString** str, NetworkMode &v)
- **BString** toBStringJson (**BString** n, NetworkMode v)
- **BString** toBString (WifiMode v)
- **BError** fromBString (**BString** str, WifiMode &v)
- **BString** toBStringJson (**BString** n, WifiMode v)
- **BString** toBString (AlarmMode v)
- **BError** fromBString (**BString** str, AlarmMode &v)
- **BString** toBStringJson (**BString** n, AlarmMode v)
- **BString** toBString (AlarmOutput v)
- **BError** fromBString (**BString** str, AlarmOutput &v)
- **BString** toBStringJson (**BString** n, AlarmOutput v)
- **BString** toBString (EventMode v)
- **BError** fromBString (**BString** str, EventMode &v)
- **BString** toBStringJson (**BString** n, EventMode v)
- **BString** toBString (Rs485Mode v)
- **BError** fromBString (**BString** str, Rs485Mode &v)
- **BString** toBStringJson (**BString** n, Rs485Mode v)
- **BString** toBString (BMeasFileType v)
- **BError** fromBString (**BString** str, BMeasFileType &v)
- **BString** toBStringJson (**BString** n, BMeasFileType v)
- **BString** toBString (WifiCmd v)
- **BError** fromBString (**BString** str, WifiCmd &v)
- **BString** toBStringJson (**BString** n, WifiCmd v)
- **BString** toBString (WifiStatus v)
- **BError** fromBString (**BString** str, WifiStatus &v)
- **BString** toBStringJson (**BString** n, WifiStatus v)
- const char ∗ channelTypeString (ChannelType type)
- const char ∗ sampleTypeString (SampleType type)
- **BFloat32** toFloat (**BUInt32** v)
- static int unitSort (BMeasureUnit1 ∗&u1, BMeasureUnit1 ∗&u2)
- static **BUInt32** roundDown512 (**BUInt32** size)
- const **BUInt32** TocMetaData (1<< 1)
- const **BUInt32** TocNewObjList (1<< 2)
- const **BUInt32** TocRawData (1<< 3)
- const **BUInt32** TocInterleavedData (1<< 5)
- const **BUInt32** TocBigEndian (1<< 6)
- const **BUInt32** TocDaqRawData (1<< 7)
- **BUInt32** round512 (**BUInt32** s)

Variables

- const **BUInt32** apiVersion = 0

6.1.1 Typedef Documentation

6.1.1.1 ChannelConfigs

```
typedef BArray<ChannelConfig> BMeasureApi::ChannelConfigs
```

6.1.2 Enumeration Type Documentation

6.1.2.1 AlarmMode

```
enum BMeasureApi::AlarmMode
```

Enumerator

AlarmModeOff	
AlarmModeHigh	
AlarmModeLow	
AlarmModeRange	
AlarmModeMagnitude	

6.1.2.2 AlarmOutput

```
enum BMeasureApi::AlarmOutput
```

Enumerator

AlarmOutputOff	
AlarmOutputDioHigh	
AlarmOutputDioLow	
AlarmOutputRelayOn	
AlarmOutputRelayOff	

6.1.2.3 AwgMode

```
enum BMeasureApi::AwgMode
```

Enumerator

AwgModeNone	
AwgModeDc	
AwgModeSine	

Enumerator

AwgModeSquare	
AwgModeTriangle	
AwgModeNoise	
AwgModeTrackRms	
AwgModeTrackMean	
AwgModeArbitrary	

6.1.2.4 AwgOutput

```
enum BMeasureApi::AwgOutput
```

Enumerator

AwgOutputNone	
AwgOutputAO0	
AwgOutputAO1	
AwgOutputAO01	

6.1.2.5 BlockTypes

```
enum BMeasureApi::BlockTypes
```

Enumerator

BlockTypeInfo	
BlockTypeData	

6.1.2.6 BMeasFileType

```
enum BMeasureApi::BMeasFileType
```

Enumerator

BMeasFileTypeBlock512	
BMeasFileTypeStream	

6.1.2.7 CalibrateStage

```
enum BMeasureApi::CalibrateStage
```

Enumerator

CalibrateStageNone	
CalibrateStageClear	
CalibrateStageSettle	
CalibrateStageAdcOffsets	
CalibrateStageDacOffsets	
CalibrateStageDacScaling0	
CalibrateStageDacScaling1	
CalibrateStageAdcScaling	
CalibrateStageAttenScaling	
CalibrationStageFiveVolts	
CalibrateStageAdcScalingWithAtten	
CalibrateStageChanClear	
CalibrateStageChanOffsets	
CalibrateStageChanScaling	

6.1.2.8 ChannelType

```
enum BMeasureApi::ChannelType
```

Enumerator

ChannelTypeNone	
ChannelTypeAnalogueIn	
ChannelTypeAnalogueOut	
ChannelTypeDigitalIn	
ChannelTypeDigitalOut	

6.1.2.9 DataSend

```
enum BMeasureApi::DataSend
```

Enumerator

DataSendOff	
DataSendStatus	
DataSendRaw	
DataSendProcessed	

6.1.2.10 DataType

```
enum BMeasureApi::DataType
```

Enumerator

DataTypeFloat32	
DataType125i	
DataTypeProc	

6.1.2.11 DigitalMode

```
enum BMeasureApi::DigitalMode
```

Enumerator

DigitalModeInput	
DigitalModeOutput	
DigitalModeInOut	
DigitalModeSyncMaster	
DigitalModeSyncSlave	

6.1.2.12 ErrorNum

```
enum BMeasureApi::ErrorNum
```

Enumerator

ErrorNumSystem	
ErrorNumDataOverrun	
ErrorNumToFast	

6.1.2.13 EventMode

```
enum BMeasureApi::EventMode
```

Enumerator

EventModeOff	
EventModeAlarm	
EventModeSecond	

6.1.2.14 FilesysDeleteType

```
enum BMeasureApi::FilesysDeleteType
```

Enumerator

FilesysDeleteTypeNone	
FilesysDeleteTypeData	
FilesysDeleteTypeFormat	

6.1.2.15 FileType

```
enum BMeasureApi::FileType
```

Enumerator

FileTypeNone	
FileTypeFile	
FileTypeDir	

6.1.2.16 LogData

```
enum BMeasureApi::LogData
```

Enumerator

LogDataOff	
LogDataRaw	
LogDataProcessed	

6.1.2.17 LogDataMode

```
enum BMeasureApi::LogDataMode
```

Enumerator

LogDataModeNormal	
LogDataModeDeleteOld	

6.1.2.18 MeasureMode

```
enum BMeasureApi::MeasureMode
```

Enumerator

MeasureModeOff	
MeasureModeOneShot	
MeasureModeRepeat	
MeasureModeContinuous	

6.1.2.19 MeasureOption

```
enum BMeasureApi::MeasureOption
```

Enumerator

MeasureOptionNone	
MeasureOptionProcess	

6.1.2.20 MessageSource

```
enum BMeasureApi::MessageSource
```

Enumerator

MessageSourceGeneral	
MessageSourceDebug	
MessageSourceTest	
MessageSourceWifi	
MessageSourceWifiTest	

6.1.2.21 Mode

```
enum BMeasureApi::Mode
```

Enumerator

ModeIdle	
ModeRun	
ModeRunProgram	
ModeInternal	
ModeSleep	
ModeDemo1	

6.1.2.22 NetworkMode

```
enum BMeasureApi::NetworkMode
```

Enumerator

NetworkModeOff	
NetworkModeDhcp	
NetworkModeManual	

6.1.2.23 NodeType

```
enum BMeasureApi::NodeType
```

Enumerator

NodeTypeNone	
NodeTypeBMeasure1	

6.1.2.24 Rs485Mode

```
enum BMeasureApi::Rs485Mode
```

Enumerator

Rs485ModeOff	
Rs485ModeBoap	

6.1.2.25 SampleType

```
enum BMeasureApi::SampleType
```

Enumerator

SampleTypeNone	
SampleTypeBool	
SampleTypeInt8	
SampleTypeInt16	
SampleTypeInt32	
SampleTypeFloat32	
SampleTypeFloat64	

6.1.2.26 SecurityMode

```
enum BMeasureApi::SecurityMode
```

Enumerator

SecurityModeBasic	
SecurityModeConfig	
SecurityModeFull	

6.1.2.27 Status

```
enum BMeasureApi::Status
```

Enumerator

StatusNone	
StatusError	
StatusWarning	
StatusRun	
StatusTriggerWait	

Enumerator

StatusEnd0	
StatusEnd1	
StatusDataOverrun	
StatusFpgaOverrun	
StatusAlarm	

6.1.2.28 SyncMode

```
enum BMeasureApi::SyncMode
```

Enumerator

SyncModeOff	
SyncModeMaster	
SyncModeSlave	

6.1.2.29 TdsDataType

```
enum BMeasureApi::TdsDataType
```

Enumerator

TdsTypeVoid	
TdsTypeI8	
TdsTypeI16	
TdsTypeI32	
TdsTypeI64	
TdsTypeU8	
TdsTypeU16	
TdsTypeU32	
TdsTypeU64	
TdsTypeSingleFloat	
TdsTypeDoubleFloat	
TdsTypeExtendedFloat	
TdsTypeSingleFloatWithUnit	
TdsTypeDoubleFloatWithUnit	
TdsTypeExtendedFloatWithUnit	
TdsTypeString	
TdsTypeBoolean	
TdsTypeTimeStamp	
TdsTypeFixedPoint	

Enumerator

TdsTypeComplexSingleFloat	
TdsTypeComplexDoubleFloat	
TdsTypeDAQmxRawData	

6.1.2.30 TriggerConfig

```
enum BMeasureApi::TriggerConfig
```

Enumerator

TriggerConfigNone	

6.1.2.31 TriggerMode

```
enum BMeasureApi::TriggerMode
```

Enumerator

TriggerModeOff	
TriggerModePositive	
TriggerModeNegative	

6.1.2.32 WifiCmd

```
enum BMeasureApi::WifiCmd
```

Enumerator

WifiCmdOff	
WifiCmdOn	
WifiCmdScan	
WifiCmdConnect	
WifiCmdDisconnect	

6.1.2.33 WifiMode

```
enum BMeasureApi::WifiMode
```

Enumerator

WifiModeOff	
WifiModeClient	
WifiModeAp	

6.1.2.34 WifiStatus

```
enum BMeasureApi::WifiStatus
```

Enumerator

WifiStatusOff	
WifiStatusOn	
WifiStatusConnecting	
WifiStatusConnected	
WifiStatusConnectedInternet	
WifStatusAP	

6.1.3 Function Documentation

6.1.3.1 channelTypeString()

```
const char * BMeasureApi::channelTypeString (
             ChannelType type )
```

6.1.3.2 fromBString() [1/33]

```
 BError BMeasureApi::fromBString (
             BString str,
             AlarmMode & v )
```

6.1.3.3 fromBString() [2/33]

```
BError BMeasureApi::fromBString (
            BString str,
            AlarmOutput & v )
```

6.1.3.4 fromBString() [3/33]

```
BError BMeasureApi::fromBString (
            BString str,
            AwgMode & v )
```

6.1.3.5 fromBString() [4/33]

```
BError BMeasureApi::fromBString (
            BString str,
            AwgOutput & v )
```

6.1.3.6 fromBString() [5/33]

```
BError BMeasureApi::fromBString (
            BString str,
            BlockTypes & v )
```

6.1.3.7 fromBString() [6/33]

```
BError BMeasureApi::fromBString (
            BString str,
            BMeasFileType & v )
```

6.1.3.8 fromBString() [7/33]

```
BError BMeasureApi::fromBString (
            BString str,
            CalibrateStage & v )
```

6.1.3.9 fromBString() [8/33]

```
BError BMeasureApi::fromBString (
            BString str,
            ChannelType & v )
```

6.1.3.10 fromBString() [9/33]

```
BError BMeasureApi::fromBString (
            BString str,
            DataSend & v )
```

6.1.3.11 fromBString() [10/33]

```
BError BMeasureApi::fromBString (
            BString str,
            DataType & v )
```

6.1.3.12 fromBString() [11/33]

```
BError BMeasureApi::fromBString (
            BString str,
            DigitalMode & v )
```

6.1.3.13 fromBString() [12/33]

```
BError BMeasureApi::fromBString (
            BString str,
            ErrorNum & v )
```

6.1.3.14 fromBString() [13/33]

```
BError BMeasureApi::fromBString (
            BString str,
            EventMode & v )
```

6.1.3.15 fromBString() [14/33]

```
BError BMeasureApi::fromBString (
            BString str,
           FilesysDeleteType & v )
```

6.1.3.16 fromBString() [15/33]

```
BError BMeasureApi::fromBString (
            BString str,
           FileType & v )
```

6.1.3.17 fromBString() [16/33]

```
BError BMeasureApi::fromBString (
            BString str,
           LogData & v )
```

6.1.3.18 fromBString() [17/33]

```
BError BMeasureApi::fromBString (
            BString str,
           LogDataMode & v )
```

6.1.3.19 fromBString() [18/33]

```
BError BMeasureApi::fromBString (
            BString str,
           MeasureMode & v )
```

6.1.3.20 fromBString() [19/33]

```
BError BMeasureApi::fromBString (
            BString str,
           MeasureOption & v )
```

6.1.3.21 fromBString() [20/33]

```
BError BMeasureApi::fromBString (
            BString str,
            MessageSource & v )
```

6.1.3.22 fromBString() [21/33]

```
BError BMeasureApi::fromBString (
            BString str,
            Mode & v )
```

6.1.3.23 fromBString() [22/33]

```
BError BMeasureApi::fromBString (
            BString str,
            NetworkMode & v )
```

6.1.3.24 fromBString() [23/33]

```
BError BMeasureApi::fromBString (
            BString str,
            NodeType & v )
```

6.1.3.25 fromBString() [24/33]

```
BError BMeasureApi::fromBString (
            BString str,
            Rs485Mode & v )
```

6.1.3.26 fromBString() [25/33]

```
BError BMeasureApi::fromBString (
            BString str,
            SampleType & v )
```

6.1.3.27 fromBString() [26/33]

```
BError BMeasureApi::fromBString (
            BString str,
            SecurityMode & v )
```

6.1.3.28 fromBString() [27/33]

```
BError BMeasureApi::fromBString (
            BString str,
            Status & v )
```

6.1.3.29 fromBString() [28/33]

```
BError BMeasureApi::fromBString (
            BString str,
            SyncMode & v )
```

6.1.3.30 fromBString() [29/33]

```
BError BMeasureApi::fromBString (
            BString str,
            TriggerConfig & v )
```

6.1.3.31 fromBString() [30/33]

```
BError BMeasureApi::fromBString (
            BString str,
            TriggerMode & v )
```

6.1.3.32 fromBString() [31/33]

```
BError BMeasureApi::fromBString (
            BString str,
            WifiCmd & v )
```

6.1.3.33 fromBString() [32/33]

```
BError BMeasureApi::fromBString (
            BString str,
           WifiMode & v )
```

6.1.3.34 fromBString() [33/33]

```
BError BMeasureApi::fromBString (
            BString str,
           WifiStatus & v )
```

6.1.3.35 round512()

```
BUInt32 BMeasureApi::round512 (
            BUInt32 s )
```

6.1.3.36 roundDown512()

```
static  BUInt32 BMeasureApi::roundDown512 (
             BUInt32 size )  [static]
```

6.1.3.37 sampleTypeString()

```
const char * BMeasureApi::sampleTypeString (
             SampleType type )
```

6.1.3.38 toBString() [1/33]

```
BString BMeasureApi::toBString (
           AlarmMode v )
```

6.1.3.39 toBString() [2/33]

```
BString BMeasureApi::toBString (
           AlarmOutput v )
```

6.1.3.40 toBString() [3/33]

```
BString BMeasureApi::toBString (
            AwgMode v )
```

6.1.3.41 toBString() [4/33]

```
BString BMeasureApi::toBString (
            AwgOutput v )
```

6.1.3.42 toBString() [5/33]

```
BString BMeasureApi::toBString (
            BlockTypes v )
```

6.1.3.43 toBString() [6/33]

```
BString BMeasureApi::toBString (
            BMeasFileType v )
```

6.1.3.44 toBString() [7/33]

```
BString BMeasureApi::toBString (
            CalibrateStage v )
```

6.1.3.45 toBString() [8/33]

```
BString BMeasureApi::toBString (
            ChannelType v )
```

6.1.3.46 toBString() [9/33]

```
BString BMeasureApi::toBString (
            DataSend v )
```

6.1.3.47 toBString() [10/33]

```
BString BMeasureApi::toBString (
            DataType v )
```

6.1.3.48 toBString() [11/33]

```
BString BMeasureApi::toBString (
            DigitalMode v )
```

6.1.3.49 toBString() [12/33]

```
BString BMeasureApi::toBString (
            ErrorNum v )
```

6.1.3.50 toBString() [13/33]

```
BString BMeasureApi::toBString (
            EventMode v )
```

6.1.3.51 toBString() [14/33]

```
BString BMeasureApi::toBString (
            FilesysDeleteType v )
```

6.1.3.52 toBString() [15/33]

```
BString BMeasureApi::toBString (
            FileType v )
```

6.1.3.53 toBString() [16/33]

```
BString BMeasureApi::toBString (
            LogData v )
```

6.1.3.54 toBString() [17/33]

```
BString BMeasureApi::toBString (
            LogDataMode v )
```

6.1.3.55 toBString() [18/33]

```
BString BMeasureApi::toBString (
            MeasureMode v )
```

6.1.3.56 toBString() [19/33]

```
BString BMeasureApi::toBString (
            MeasureOption v )
```

6.1.3.57 toBString() [20/33]

```
BString BMeasureApi::toBString (
            MessageSource v )
```

6.1.3.58 toBString() [21/33]

```
BString BMeasureApi::toBString (
            Mode v )
```

6.1.3.59 toBString() [22/33]

```
BString BMeasureApi::toBString (
            NetworkMode v )
```

6.1.3.60 toBString() [23/33]

```
BString BMeasureApi::toBString (
            NodeType v )
```

6.1.3.61 toBString() [24/33]

```
BString BMeasureApi::toBString (
            Rs485Mode v )
```

6.1.3.62 toBString() [25/33]

```
BString BMeasureApi::toBString (
            SampleType v )
```

6.1.3.63 toBString() [26/33]

```
BString BMeasureApi::toBString (
            SecurityMode v )
```

6.1.3.64 toBString() [27/33]

```
BString BMeasureApi::toBString (
            Status v )
```

6.1.3.65 toBString() [28/33]

```
BString BMeasureApi::toBString (
            SyncMode v )
```

6.1.3.66 toBString() [29/33]

```
BString BMeasureApi::toBString (
            TriggerConfig v )
```

6.1.3.67 toBString() [30/33]

```
BString BMeasureApi::toBString (
            TriggerMode v )
```

6.1.3.68 toBString() [31/33]

```
BString BMeasureApi::toBString (
            WifiCmd v )
```

6.1.3.69 toBString() [32/33]

```
BString BMeasureApi::toBString (
            WifiMode v )
```

6.1.3.70 toBString() [33/33]

```
BString BMeasureApi::toBString (
            WifiStatus v )
```

6.1.3.71 toBStringJson() [1/33]

```
BString BMeasureApi::toBStringJson (
             BString n,
            AlarmMode v )
```

6.1.3.72 toBStringJson() [2/33]

```
BString BMeasureApi::toBStringJson (
             BString n,
            AlarmOutput v )
```

6.1.3.73 toBStringJson() [3/33]

```
BString BMeasureApi::toBStringJson (
             BString n,
            AwgMode v )
```

6.1.3.74 toBStringJson() [4/33]

```
BString BMeasureApi::toBStringJson (
            BString n,
           AwgOutput v )
```

6.1.3.75 toBStringJson() [5/33]

```
BString BMeasureApi::toBStringJson (
            BString n,
           BlockTypes v )
```

6.1.3.76 toBStringJson() [6/33]

```
BString BMeasureApi::toBStringJson (
            BString n,
           BMeasFileType v )
```

6.1.3.77 toBStringJson() [7/33]

```
BString BMeasureApi::toBStringJson (
            BString n,
           CalibrateStage v )
```

6.1.3.78 toBStringJson() [8/33]

```
BString BMeasureApi::toBStringJson (
            BString n,
           ChannelType v )
```

6.1.3.79 toBStringJson() [9/33]

```
BString BMeasureApi::toBStringJson (
            BString n,
           DataSend v )
```

6.1.3.80 toBStringJson() [10/33]

```
BString BMeasureApi::toBStringJson (
            BString n,
           DataType v )
```

6.1.3.81 toBStringJson() [11/33]

```
BString BMeasureApi::toBStringJson (
            BString n,
           DigitalMode v )
```

6.1.3.82 toBStringJson() [12/33]

```
BString BMeasureApi::toBStringJson (
            BString n,
           ErrorNum v )
```

6.1.3.83 toBStringJson() [13/33]

```
BString BMeasureApi::toBStringJson (
            BString n,
           EventMode v )
```

6.1.3.84 toBStringJson() [14/33]

```
BString BMeasureApi::toBStringJson (
            BString n,
           FilesysDeleteType v )
```

6.1.3.85 toBStringJson() [15/33]

```
BString BMeasureApi::toBStringJson (
            BString n,
           FileType v )
```

6.1.3.86 toBStringJson() [16/33]

```
BString BMeasureApi::toBStringJson (
            BString n,
           LogData v )
```

6.1.3.87 toBStringJson() [17/33]

```
BString BMeasureApi::toBStringJson (
            BString n,
           LogDataMode v )
```

6.1.3.88 toBStringJson() [18/33]

```
BString BMeasureApi::toBStringJson (
            BString n,
           MeasureMode v )
```

6.1.3.89 toBStringJson() [19/33]

```
BString BMeasureApi::toBStringJson (
            BString n,
           MeasureOption v )
```

6.1.3.90 toBStringJson() [20/33]

```
BString BMeasureApi::toBStringJson (
            BString n,
           MessageSource v )
```

6.1.3.91 toBStringJson() [21/33]

```
BString BMeasureApi::toBStringJson (
            BString n,
           Mode v )
```

6.1.3.92 toBStringJson() [22/33]

```
BString BMeasureApi::toBStringJson (
            BString n,
           NetworkMode v )
```

6.1.3.93 toBStringJson() [23/33]

```
BString BMeasureApi::toBStringJson (
            BString n,
           NodeType v )
```

6.1.3.94 toBStringJson() [24/33]

```
BString BMeasureApi::toBStringJson (
            BString n,
           Rs485Mode v )
```

6.1.3.95 toBStringJson() [25/33]

```
BString BMeasureApi::toBStringJson (
            BString n,
           SampleType v )
```

6.1.3.96 toBStringJson() [26/33]

```
BString BMeasureApi::toBStringJson (
            BString n,
           SecurityMode v )
```

6.1.3.97 toBStringJson() [27/33]

```
BString BMeasureApi::toBStringJson (
            BString n,
           Status v )
```

6.1.3.98 toBStringJson() [28/33]

```
BString BMeasureApi::toBStringJson (
            BString n,
            SyncMode v )
```

6.1.3.99 toBStringJson() [29/33]

```
BString BMeasureApi::toBStringJson (
            BString n,
            TriggerConfig v )
```

6.1.3.100 toBStringJson() [30/33]

```
BString BMeasureApi::toBStringJson (
            BString n,
            TriggerMode v )
```

6.1.3.101 toBStringJson() [31/33]

```
BString BMeasureApi::toBStringJson (
            BString n,
            WifiCmd v )
```

6.1.3.102 toBStringJson() [32/33]

```
BString BMeasureApi::toBStringJson (
            BString n,
            WifiMode v )
```

6.1.3.103 toBStringJson() [33/33]

```
BString BMeasureApi::toBStringJson (
            BString n,
            WifiStatus v )
```

6.1.3.104 TocBigEndian()

```
const  BUInt32 BMeasureApi::TocBigEndian (
            1<< 6 )
```

6.1.3.105 TocDaqRawData()

```
const  BUInt32 BMeasureApi::TocDaqRawData (
            1<< 7 )
```

6.1.3.106 TocInterleavedData()

```
const  BUInt32 BMeasureApi::TocInterleavedData (
            1<< 5 )
```

6.1.3.107 TocMetaData()

```
const  BUInt32 BMeasureApi::TocMetaData (
            1<< 1 )
```

6.1.3.108 TocNewObjList()

```
const  BUInt32 BMeasureApi::TocNewObjList (
            1<< 2 )
```

6.1.3.109 TocRawData()

```
const  BUInt32 BMeasureApi::TocRawData (
            1<< 3 )
```

6.1.3.110 toFloat()

```
 BFloat32 BMeasureApi::toFloat (
             BUInt32 v )  [inline]
```

6.1.3.111 unitSort()

```
static int BMeasureApi::unitSort (
            BMeasureUnit1 *& u1,
            BMeasureUnit1 *& u2 )  [static]
```

6.1.4 Variable Documentation

6.1.4.1 apiVersion

```
const  BUInt32 BMeasureApi::apiVersion = 0
```

Chapter 7

Class Documentation

7.1 BMeasureApi::AlarmConfig Class Reference

```
#include <BMeasureD.h>
```

Static Public Member Functions

- static const **BObjMember** ∗ getMembers ()

Public Attributes

- AlarmMode mode

    Alarm mode.
- AlarmOutput output

    Alarm output.
- **BUInt8** outputChannel

    Alarm output channel.
- **BUInt8** spare1
- **BFloat32** levelHigh

    Alarm level high.
- **BFloat32** levelLow

    Alarm level low.
- **BUInt32** spare2

7.1.1 Member Function Documentation

7.1.1.1 getMembers()

```
const  BObjMember ∗ BMeasureApi::AlarmConfig::getMembers ( )  [static]
```

7.1.2 Member Data Documentation

7.1.2.1 levelHigh

`BFloat32 BMeasureApi::AlarmConfig::levelHigh`

Alarm level high.

7.1.2.2 levelLow

`BFloat32 BMeasureApi::AlarmConfig::levelLow`

Alarm level low.

7.1.2.3 mode

`AlarmMode BMeasureApi::AlarmConfig::mode`

Alarm mode.

7.1.2.4 output

`AlarmOutput BMeasureApi::AlarmConfig::output`

Alarm output.

7.1.2.5 outputChannel

`BUInt8 BMeasureApi::AlarmConfig::outputChannel`

Alarm output channel.

7.1.2.6 spare1

`BUInt8 BMeasureApi::AlarmConfig::spare1`

7.1.2.7 spare2

```
BUInt32 BMeasureApi::AlarmConfig::spare2
```

The documentation for this class was generated from the following files:

- BMeasureD.h
- BMeasureD.cpp

7.2 BMeasureApi::AwgConfig Class Reference

File information.

```
#include <BMeasureD.h>
```

Static Public Member Functions

- static const **BObjMember** ∗ getMembers ()

Public Attributes

- AwgMode mode

  The mode including waveform type.
- AwgOutput output

  The output channels.
- **BUInt8** trackChannel

  Input channel to track.
- **BUInt8** spare
- **BFloat32** frequency

  The frequency.
- **BFloat32** amplitude

  The peak amplitude in Volts.
- **BFloat32** offset

  The DC offset in volts.
- **BFloat32** duty

  The Duty cycle in %.
- **BUInt32** numSamples

  The number of samples when using arbitary waveforms.

7.2.1 Detailed Description

File information.

7.2.2 Member Function Documentation

7.2.2.1 getMembers()

```
const BObjMember * BMeasureApi::AwgConfig::getMembers ( ) [static]
```

7.2.3 Member Data Documentation

7.2.3.1 amplitude

```
BFloat32 BMeasureApi::AwgConfig::amplitude
```

The peak amplitude in Volts.

7.2.3.2 duty

```
BFloat32 BMeasureApi::AwgConfig::duty
```

The Duty cycle in %.

7.2.3.3 frequency

```
BFloat32 BMeasureApi::AwgConfig::frequency
```

The frequency.

7.2.3.4 mode

```
AwgMode BMeasureApi::AwgConfig::mode
```

The mode including waveform type.

7.2.3.5 numSamples

```
BUInt32 BMeasureApi::AwgConfig::numSamples
```

The number of samples when using arbitary waveforms.

7.2.3.6 offset

```
BFloat32 BMeasureApi::AwgConfig::offset
```

The DC offset in volts.

7.2.3.7 output

```
AwgOutput BMeasureApi::AwgConfig::output
```

The output channels.

7.2.3.8 spare

```
BUInt8 BMeasureApi::AwgConfig::spare
```

7.2.3.9 trackChannel

```
BUInt8 BMeasureApi::AwgConfig::trackChannel
```

Input channel to track.

The documentation for this class was generated from the following files:

- BMeasureD.h
- BMeasureD.cpp

7.3 BFirmwareInfo Struct Reference

Public Attributes

- **BUInt32** magic
- **BUInt32** length
- **BUInt32** checksum
- **BUInt8** type
- **BUInt8** ver0
- **BUInt8** ver1
- **BUInt8** ver2

7.3.1 Member Data Documentation

7.3.1.1 checksum

```
BUInt32 BFirmwareInfo::checksum
```

7.3.1.2 length

```
BUInt32 BFirmwareInfo::length
```

7.3.1.3 magic

```
BUInt32 BFirmwareInfo::magic
```

7.3.1.4 type

```
BUInt8 BFirmwareInfo::type
```

7.3.1.5 ver0

```
BUInt8 BFirmwareInfo::ver0
```

7.3.1.6 ver1

```
BUInt8 BFirmwareInfo::ver1
```

7.3.1.7 ver2

```
BUInt8 BFirmwareInfo::ver2
```

The documentation for this struct was generated from the following file:

- Dfu.cpp

7.4 BMdns Class Reference

```
#include <BMdns.h>
```

Public Member Functions

- BMdns ()
- ∼BMdns ()
- **BError** init ()
- **BError** findServices (**BString** service, **BUInt32** timeoutMs, **BList**< BMdnsService > &services)

Private Attributes

- **BSocket** osocket
- **BUInt32** otransactionId

7.4.1 Constructor & Destructor Documentation

7.4.1.1 BMdns()

```
BMdns::BMdns ( )
```

7.4.1.2 ∼BMdns()

```
BMdns::∼BMdns ( )
```

7.4.2 Member Function Documentation

7.4.2.1 findServices()

```
 BError BMdns::findServices (
              BString service,
              BUInt32 timeoutMs,
              BList< BMdnsService > & services )
```

Unicast response, class IN

7.4.2.2 init()

```
BError BMdns::init ( )
```

7.4.3 Member Data Documentation

7.4.3.1 osocket

```
BSocket BMdns::osocket [private]
```

7.4.3.2 otransactionId

```
BUInt32 BMdns::otransactionId [private]
```

The documentation for this class was generated from the following files:

- BMdns.h
- BMdns.cpp

7.5 BMdnsService Class Reference

```
#include <BMdns.h>
```

Public Attributes

- **BString** name
- **BSocketAddressINET** address
- **BString** hostname
- BStringList extra

7.5.1 Member Data Documentation

7.5.1.1 address

```
BSocketAddressINET BMdnsService::address
```

7.5.1.2 extra

```
BStringList BMdnsService::extra
```

7.5.1.3 hostname

```
BString BMdnsService::hostname
```

7.5.1.4 name

```
BString BMdnsService::name
```

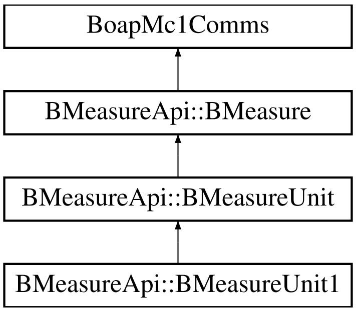
The documentation for this class was generated from the following file:

- BMdns.h

7.6 BMeasureApi::BMeasure Class Reference

```
#include <BMeasureB.h>
```

Inheritance diagram for BMeasureApi::BMeasure:



Public Member Functions

- BMeasure (**Bool** threaded=0, **BUInt** reqSize=512)
- **BError** getNodeInfo (NodeInfo &nodeInfo)

  Get node information.
- **BError** login (const **BString** &userid, const **BString** &password)

  Provides user/password information for secure connection.
- **BError** logout ()

  Logs out.
- **BError** changePassword (const **BString** &userid, const **BString** &oldPassword, const **BString** &newPassword)

  Changes the users password.
- void factoryReset (const **Bool** &bootLoader, const **Bool** &resetConfig)

  Factory reset.
- void sendTime (const **BTimeUs** &time)

  Sends the current time.
- **BError** setMode (const Mode &mode)

  Set the current operational mode.
- **BError** getStatus (NodeStatus &nodeStatus)

  Get the node status.
- void sendStatus (const NodeStatus &nodeStatus)

  Sends the current status.
- **BError** getInformation (Information &info)
- **BError** getInfoBlock (InfoBlock &infoBlock)
- **BError** getChannelConfig (const **BUInt32** &channelNumber, ChannelConfig &channelConfig)
- **BError** setChannelConfig (const **BUInt32** &channelNumber, const ChannelConfig &channelConfig)
- **BError** setChannelConfigFull (const **BUInt64** &key, const **BUInt32** &channelNumber, const ChannelConfig &channelConfig)
- **BError** getConfig (Configuration &config)

  Return units configuration.
- **BError** setConfig (const Configuration &config)

  Set units configuration.
- **BError** getMeasurementConfig (const **Bool** &saved, MeasurementConfig &measurementConfig)

  Get measurement config.
- **BError** setMeasurementConfig (const **Bool** &save, const MeasurementConfig &measurementConfig)

  Set measurement config.
- **BError** sendDataEnable (const **BUInt8** &sendSet)

  Enable the sending of different data streams.
- **BError** measure (const DataType &dataType, DataBlock &dataBlock)

  Performs a single sample measurement.
- void sendInfo (const InfoBlock &infoBlock)

  Sends an info block.
- void sendData (const DataBlock &dataBlock)

  Sends a data block.
- void sendChannelConfig (const ChannelConfig &channelConfig)

  Sends a ChannelConfig block.
- **BError** getAwgConfig (const **Bool** &saved, AwgConfig &awgConfig)

  Get AWG Configuration.
- **BError** setAwgConfig (const **Bool** &save, const AwgConfig &awgConfig)

  Configure AWG.
- **BError** setAwgWaveform (const **BUInt32** &chan, const **BUInt32** &pos, const FileData &dataBlock)

  Configure AWG Arbitrary waveform.

- **BError** setAnalogueOut (const **BUInt32** &chan, const **BFloat32** &value)

  Set analogue output value.
- **BError** setDigital (const **BUInt32** &bits)

  Set digital bits.
- **BError** getDigital (**BUInt32** &bits)

  Get digital bits.
- **BError** setRelay (const **BUInt32** &relayNum, const **Bool** &state)

  Set relay.
- **BError** getSwitch (const **BUInt32** &switchNum, **Bool** &state)

  Get digital bits.
- **BError** alarmsClear (const **BUInt32** &bits)

  Clear alarms.
- **BError** filesysInfo (const **BString** &path, FilesysInfo &filesysInfo)
- **BError** filesysDelete (const **BString** &path, const FilesysDeleteType &deleteType)
- **BError** fileList (const **BString** &path, const **BUInt32** &pos, FileInfo &fileInfo)
- **BError** fileOpen (const **BString** &name, const **BString** &mode, **BUInt32** &handle)
- **BError** fileRead (const **BUInt32** &handle, const **BUInt32** &pos, const **BUInt32** &len, FileData & **data**)
- **BError** fileWrite (const **BUInt32** &handle, const **BUInt32** &pos, const FileData & **data**)
- **BError** fileClose (const **BUInt32** &handle)
- **BError** fileDelete (const **BString** &name)
- **BError** wifiAccesspointNum (**BUInt32** &numFound)

  Return how many access points found.
- **BError** wifiAccesspointInfo (const **BUInt32** &id, WifiAccessPoint &accessPoint)

  Return information on Wifi accesspoint.
- **BError** wifiCommand (const WifiCmd &wifiCmd)

  Perform Wifi command.
- **BError** functionUnLock (const **BUInt32** &unlocks, const **BString** &key)

  UnLock/Lock special functions.
- **BError** getBoardConfig (BoardConfig &config)

  Get the boards configuration.
- **BError** setBoardConfig (const BoardConfig &config)

  Sets the boards configuration, requires key.
- **BError** runBoardTest (const **BString** &test)

  Runs the given board test.
- **BError** calibrate (const CalibrateInfo &calibInfo)

  Calibrate system.
- void sendMessage (const **BUInt32** &source, const **BString** &message)

  Send text messages.
- **BError** processRequest ()
- virtual **BError** getNodeInfoServe (NodeInfo &nodeInfo)
- virtual **BError** loginServe (const **BString** &userid, const **BString** &password)
- virtual **BError** logoutServe ()
- virtual **BError** changePasswordServe (const **BString** &userid, const **BString** &oldPassword, const **BString** &newPassword)
- virtual void factoryResetServe (const **Bool** &bootLoader, const **Bool** &resetConfig)
- virtual void sendTimeServe (const **BTimeUs** &time)
- virtual **BError** setModeServe (const Mode &mode)
- virtual **BError** getStatusServe (NodeStatus &nodeStatus)
- virtual void sendStatusServe (const NodeStatus &nodeStatus)
- virtual **BError** getInformationServe (Information &info)
- virtual **BError** getInfoBlockServe (InfoBlock &infoBlock)
- virtual **BError** getChannelConfigServe (const **BUInt32** &channelNumber, ChannelConfig &channelConfig)

- virtual **BError** setChannelConfigServe (const **BUInt32** &channelNumber, const ChannelConfig &channel↩ Config)
- virtual **BError** setChannelConfigFullServe (const **BUInt64** &key, const **BUInt32** &channelNumber, const ChannelConfig &channelConfig)
- virtual **BError** getConfigServe (Configuration &config)
- virtual **BError** setConfigServe (const Configuration &config)
- virtual **BError** getMeasurementConfigServe (const **Bool** &saved, MeasurementConfig &measurement↩ Config)
- virtual **BError** setMeasurementConfigServe (const **Bool** &save, const MeasurementConfig &measurement↩ Config)
- virtual **BError** sendDataEnableServe (const **BUInt8** &sendSet)
- virtual **BError** measureServe (const DataType &dataType, DataBlock &dataBlock)
- virtual void sendInfoServe (const InfoBlock &infoBlock)
- virtual void sendDataServe (const DataBlock &dataBlock)
- virtual void sendChannelConfigServe (const ChannelConfig &channelConfig)
- virtual **BError** getAwgConfigServe (const **Bool** &saved, AwgConfig &awgConfig)
- virtual **BError** setAwgConfigServe (const **Bool** &save, const AwgConfig &awgConfig)
- virtual **BError** setAwgWaveformServe (const **BUInt32** &chan, const **BUInt32** &pos, const FileData &data↩ Block)
- virtual **BError** setAnalogueOutServe (const **BUInt32** &chan, const **BFloat32** &value)
- virtual **BError** setDigitalServe (const **BUInt32** &bits)
- virtual **BError** getDigitalServe (**BUInt32** &bits)
- virtual **BError** setRelayServe (const **BUInt32** &relayNum, const **Bool** &state)
- virtual **BError** getSwitchServe (const **BUInt32** &switchNum, **Bool** &state)
- virtual **BError** alarmsClearServe (const **BUInt32** &bits)
- virtual **BError** filesysInfoServe (const **BString** &path, FilesysInfo &filesysInfo)
- virtual **BError** filesysDeleteServe (const **BString** &path, const FilesysDeleteType &deleteType)
- virtual **BError** fileListServe (const **BString** &path, const **BUInt32** &pos, FileInfo &fileInfo)
- virtual **BError** fileOpenServe (const **BString** &name, const **BString** &mode, **BUInt32** &handle)
- virtual **BError** fileReadServe (const **BUInt32** &handle, const **BUInt32** &pos, const **BUInt32** &len, FileData & **data**)
- virtual **BError** fileWriteServe (const **BUInt32** &handle, const **BUInt32** &pos, const FileData & **data**)
- virtual **BError** fileCloseServe (const **BUInt32** &handle)
- virtual **BError** fileDeleteServe (const **BString** &name)
- virtual **BError** wifiAccesspointNumServe (**BUInt32** &numFound)
- virtual **BError** wifiAccesspointInfoServe (const **BUInt32** &id, WifiAccessPoint &accessPoint)
- virtual **BError** wifiCommandServe (const WifiCmd &wifiCmd)
- virtual **BError** functionUnLockServe (const **BUInt32** &unlocks, const **BString** &key)
- virtual **BError** getBoardConfigServe (BoardConfig &config)
- virtual **BError** setBoardConfigServe (const BoardConfig &config)
- virtual **BError** runBoardTestServe (const **BString** &test)
- virtual **BError** calibrateServe (const CalibrateInfo &calibInfo)
- virtual void sendMessageServe (const **BUInt32** &source, const **BString** &message)

Additional Inherited Members

7.6.1 Detailed Description

Channel configuration Measurement config Data Block. Data packed into bytestream based on sampleTypes. Data Block. Data in floating point format for all channels Processed data for a channel Data Proc Block. Processd Data packed into bytestream based on sampleTypes. Info Block AWG Configuration File information File information

7.6.2 Constructor & Destructor Documentation

7.6.2.1 BMeasure()

```
BMeasureApi::BMeasure::BMeasure (
              Bool threaded = 0,
              BUInt reqSize = 512 )
```

7.6.3 Member Function Documentation

7.6.3.1 alarmsClear()

```
BError BMeasureApi::BMeasure::alarmsClear (
            const  BUInt32 & bits )
```

Clear alarms.

7.6.3.2 alarmsClearServe()

```
BError BMeasureApi::BMeasure::alarmsClearServe (
            const  BUInt32 & bits )  [virtual]
```

7.6.3.3 calibrate()

```
BError BMeasureApi::BMeasure::calibrate (
            const CalibrateInfo & calibInfo )
```

Calibrate system.

7.6.3.4 calibrateServe()

```
BError BMeasureApi::BMeasure::calibrateServe (
            const CalibrateInfo & calibInfo )  [virtual]
```

7.6.3.5 changePassword()

```
BError BMeasureApi::BMeasure::changePassword (
            const  BString & userid,
            const  BString & oldPassword,
            const  BString & newPassword )
```

Changes the users password.

7.6.3.6 changePasswordServe()

```
BError BMeasureApi::BMeasure::changePasswordServe (
            const  BString & userid,
            const  BString & oldPassword,
            const  BString & newPassword )  [virtual]
```

7.6.3.7 factoryReset()

```
void BMeasureApi::BMeasure::factoryReset (
            const  Bool & bootLoader,
            const  Bool & resetConfig )
```

Factory reset.

7.6.3.8 factoryResetServe()

```
void BMeasureApi::BMeasure::factoryResetServe (
            const  Bool & bootLoader,
            const  Bool & resetConfig )  [virtual]
```

7.6.3.9 fileClose()

```
BError BMeasureApi::BMeasure::fileClose (
            const  BUInt32 & handle )
```

7.6.3.10 fileCloseServe()

```
BError BMeasureApi::BMeasure::fileCloseServe (
            const  BUInt32 & handle )  [virtual]
```

7.6.3.11 fileDelete()

```
BError BMeasureApi::BMeasure::fileDelete (
            const  BString & name )
```

7.6.3.12 fileDeleteServe()

```
BError BMeasureApi::BMeasure::fileDeleteServe (
            const  BString & name )  [virtual]
```

7.6.3.13 fileList()

```
BError BMeasureApi::BMeasure::fileList (
            const  BString & path,
            const  BUInt32 & pos,
            FileInfo & fileInfo )
```

7.6.3.14 fileListServe()

```
BError BMeasureApi::BMeasure::fileListServe (
            const  BString & path,
            const  BUInt32 & pos,
            FileInfo & fileInfo )  [virtual]
```

7.6.3.15 fileOpen()

```
BError BMeasureApi::BMeasure::fileOpen (
            const  BString & name,
            const  BString & mode,
             BUInt32 & handle )
```

7.6.3.16 fileOpenServe()

```
BError BMeasureApi::BMeasure::fileOpenServe (
            const  BString & name,
            const  BString & mode,
             BUInt32 & handle )  [virtual]
```

7.6.3.17 fileRead()

```
BError BMeasureApi::BMeasure::fileRead (
            const  BUInt32 & handle,
            const  BUInt32 & pos,
            const  BUInt32 & len,
            FileData & data )
```

7.6.3.18 fileReadServe()

```
BError BMeasureApi::BMeasure::fileReadServe (
            const  BUInt32 & handle,
            const  BUInt32 & pos,
            const  BUInt32 & len,
            FileData & data )  [virtual]
```

7.6.3.19 filesysDelete()

```
BError BMeasureApi::BMeasure::filesysDelete (
            const  BString & path,
            const FilesysDeleteType & deleteType )
```

7.6.3.20 filesysDeleteServe()

```
BError BMeasureApi::BMeasure::filesysDeleteServe (
            const  BString & path,
            const FilesysDeleteType & deleteType )  [virtual]
```

7.6.3.21 filesysInfo()

```
BError BMeasureApi::BMeasure::filesysInfo (
            const  BString & path,
            FilesysInfo & filesysInfo )
```

7.6.3.22 filesysInfoServe()

```
BError BMeasureApi::BMeasure::filesysInfoServe (
            const  BString & path,
            FilesysInfo & filesysInfo )  [virtual]
```

7.6.3.23 fileWrite()

```
BError BMeasureApi::BMeasure::fileWrite (
            const  BUInt32 & handle,
            const  BUInt32 & pos,
            const FileData & data )
```

7.6.3.24 fileWriteServe()

```
BError BMeasureApi::BMeasure::fileWriteServe (
            const  BUInt32 & handle,
            const  BUInt32 & pos,
            const FileData & data )  [virtual]
```

7.6.3.25 functionUnLock()

```
BError BMeasureApi::BMeasure::functionUnLock (
            const  BUInt32 & unlocks,
            const  BString & key )
```

UnLock/Lock special functions.

7.6.3.26 functionUnLockServe()

```
BError BMeasureApi::BMeasure::functionUnLockServe (
            const  BUInt32 & unlocks,
            const  BString & key )  [virtual]
```

7.6.3.27 getAwgConfig()

```
BError BMeasureApi::BMeasure::getAwgConfig (
            const  Bool & saved,
            AwgConfig & awgConfig )
```

Get AWG Configuration.

7.6.3.28 getAwgConfigServe()

```
BError BMeasureApi::BMeasure::getAwgConfigServe (
            const Bool & saved,
            AwgConfig & awgConfig ) [virtual]
```

7.6.3.29 getBoardConfig()

```
BError BMeasureApi::BMeasure::getBoardConfig (
            BoardConfig & config )
```

Get the boards configuration.

7.6.3.30 getBoardConfigServe()

```
BError BMeasureApi::BMeasure::getBoardConfigServe (
            BoardConfig & config ) [virtual]
```

7.6.3.31 getChannelConfig()

```
BError BMeasureApi::BMeasure::getChannelConfig (
            const BUInt32 & channelNumber,
            ChannelConfig & channelConfig )
```

7.6.3.32 getChannelConfigServe()

```
BError BMeasureApi::BMeasure::getChannelConfigServe (
            const BUInt32 & channelNumber,
            ChannelConfig & channelConfig ) [virtual]
```

7.6.3.33 getConfig()

```
BError BMeasureApi::BMeasure::getConfig (
            Configuration & config )
```

Return units configuration.

7.6.3.34 getConfigServe()

```
BError BMeasureApi::BMeasure::getConfigServe (
            Configuration & config )  [virtual]
```

7.6.3.35 getDigital()

```
BError BMeasureApi::BMeasure::getDigital (
             BUInt32 & bits )
```

Get digital bits.

7.6.3.36 getDigitalServe()

```
BError BMeasureApi::BMeasure::getDigitalServe (
             BUInt32 & bits )  [virtual]
```

7.6.3.37 getInfoBlock()

```
BError BMeasureApi::BMeasure::getInfoBlock (
            InfoBlock & infoBlock )
```

7.6.3.38 getInfoBlockServe()

```
BError BMeasureApi::BMeasure::getInfoBlockServe (
            InfoBlock & infoBlock )  [virtual]
```

7.6.3.39 getInformation()

```
BError BMeasureApi::BMeasure::getInformation (
            Information & info )
```

7.6.3.40 getInformationServe()

```
BError BMeasureApi::BMeasure::getInformationServe (
            Information & info )  [virtual]
```

7.6.3.41 getMeasurementConfig()

```
BError BMeasureApi::BMeasure::getMeasurementConfig (
            const  Bool & saved,
            MeasurementConfig & measurementConfig )
```

Get measurement config.

7.6.3.42 getMeasurementConfigServe()

```
BError BMeasureApi::BMeasure::getMeasurementConfigServe (
            const  Bool & saved,
            MeasurementConfig & measurementConfig )  [virtual]
```

7.6.3.43 getNodeInfo()

```
BError BMeasureApi::BMeasure::getNodeInfo (
            NodeInfo & nodeInfo )
```

Get node information.

7.6.3.44 getNodeInfoServe()

```
BError BMeasureApi::BMeasure::getNodeInfoServe (
            NodeInfo & nodeInfo )  [virtual]
```

7.6.3.45 getStatus()

```
BError BMeasureApi::BMeasure::getStatus (
            NodeStatus & nodeStatus )
```

Get the node status.

7.6.3.46 getStatusServe()

```
BError BMeasureApi::BMeasure::getStatusServe (
            NodeStatus & nodeStatus )  [virtual]
```

7.6.3.47 getSwitch()

```
BError BMeasureApi::BMeasure::getSwitch (
            const  BUInt32 & switchNum,
             Bool & state )
```

Get digital bits.

7.6.3.48 getSwitchServe()

```
BError BMeasureApi::BMeasure::getSwitchServe (
            const  BUInt32 & switchNum,
             Bool & state )  [virtual]
```

7.6.3.49 login()

```
BError BMeasureApi::BMeasure::login (
            const  BString & userid,
            const  BString & password )
```

Provides user/password information for secure connection.

7.6.3.50 loginServe()

```
BError BMeasureApi::BMeasure::loginServe (
            const  BString & userid,
            const  BString & password )  [virtual]
```

7.6.3.51 logout()

```
BError BMeasureApi::BMeasure::logout ( )
```

Logs out.

7.6.3.52 logoutServe()

```
BError BMeasureApi::BMeasure::logoutServe ( )  [virtual]
```

7.6.3.53 measure()

```
BError BMeasureApi::BMeasure::measure (
            const DataType & dataType,
            DataBlock & dataBlock )
```

Performs a single sample measurement.

7.6.3.54 measureServe()

```
BError BMeasureApi::BMeasure::measureServe (
            const DataType & dataType,
            DataBlock & dataBlock )  [virtual]
```

7.6.3.55 processRequest()

```
BError BMeasureApi::BMeasure::processRequest ( )
```

7.6.3.56 runBoardTest()

```
BError BMeasureApi::BMeasure::runBoardTest (
            const  BString & test )
```

Runs the given board test.

7.6.3.57 runBoardTestServe()

```
BError BMeasureApi::BMeasure::runBoardTestServe (
            const  BString & test )  [virtual]
```

7.6.3.58 sendChannelConfig()

```
void BMeasureApi::BMeasure::sendChannelConfig (
             const ChannelConfig & channelConfig )
```

Sends a ChannelConfig block.

7.6.3.59 sendChannelConfigServe()

```
void BMeasureApi::BMeasure::sendChannelConfigServe (
             const ChannelConfig & channelConfig )  [virtual]
```

7.6.3.60 sendData()

```
void BMeasureApi::BMeasure::sendData (
             const DataBlock & dataBlock )
```

Sends a data block.

7.6.3.61 sendDataEnable()

```
 BError BMeasureApi::BMeasure::sendDataEnable (
             const  BUInt8 & sendSet )
```

Enable the sending of different data streams.

7.6.3.62 sendDataEnableServe()

```
 BError BMeasureApi::BMeasure::sendDataEnableServe (
             const  BUInt8 & sendSet )  [virtual]
```

7.6.3.63 sendDataServe()

```
void BMeasureApi::BMeasure::sendDataServe (
             const DataBlock & dataBlock )  [virtual]
```

Reimplemented in BMeasureApi::BMeasureUnit.

7.6.3.64 sendInfo()

```
void BMeasureApi::BMeasure::sendInfo (
            const InfoBlock & infoBlock )
```

Sends an info block.

7.6.3.65 sendInfoServe()

```
void BMeasureApi::BMeasure::sendInfoServe (
            const InfoBlock & infoBlock )  [virtual]
```

7.6.3.66 sendMessage()

```
void BMeasureApi::BMeasure::sendMessage (
            const  BUInt32 & source,
            const  BString & message )
```

Send text messages.

7.6.3.67 sendMessageServe()

```
void BMeasureApi::BMeasure::sendMessageServe (
            const  BUInt32 & source,
            const  BString & message )  [virtual]
```

Reimplemented in BMeasureApi::BMeasureUnit1.

7.6.3.68 sendStatus()

```
void BMeasureApi::BMeasure::sendStatus (
            const NodeStatus & nodeStatus )
```

Sends the current status.

7.6.3.69 sendStatusServe()

```
void BMeasureApi::BMeasure::sendStatusServe (
            const NodeStatus & nodeStatus )  [virtual]
```

Reimplemented in BMeasureApi::BMeasureUnit1.

7.6.3.70 sendTime()

```
void BMeasureApi::BMeasure::sendTime (
             const  BTimeUs & time )
```

Sends the current time.

7.6.3.71 sendTimeServe()

```
void BMeasureApi::BMeasure::sendTimeServe (
             const  BTimeUs & time )  [virtual]
```

7.6.3.72 setAnalogueOut()

```
 BError BMeasureApi::BMeasure::setAnalogueOut (
             const  BUInt32 & chan,
             const  BFloat32 & value )
```

Set analogue output value.

7.6.3.73 setAnalogueOutServe()

```
 BError BMeasureApi::BMeasure::setAnalogueOutServe (
             const  BUInt32 & chan,
             const  BFloat32 & value )  [virtual]
```

7.6.3.74 setAwgConfig()

```
 BError BMeasureApi::BMeasure::setAwgConfig (
             const  Bool & save,
             const AwgConfig & awgConfig )
```

Configure AWG.

7.6.3.75 setAwgConfigServe()

```
 BError BMeasureApi::BMeasure::setAwgConfigServe (
             const  Bool & save,
             const AwgConfig & awgConfig )  [virtual]
```

7.6.3.76 setAwgWaveform()

```
BError BMeasureApi::BMeasure::setAwgWaveform (
            const  BUInt32 & chan,
            const  BUInt32 & pos,
            const FileData & dataBlock )
```

Configure AWG Arbitrary waveform.

7.6.3.77 setAwgWaveformServe()

```
BError BMeasureApi::BMeasure::setAwgWaveformServe (
            const  BUInt32 & chan,
            const  BUInt32 & pos,
            const FileData & dataBlock )  [virtual]
```

7.6.3.78 setBoardConfig()

```
BError BMeasureApi::BMeasure::setBoardConfig (
            const BoardConfig & config )
```

Sets the boards configuration, requires key.

7.6.3.79 setBoardConfigServe()

```
BError BMeasureApi::BMeasure::setBoardConfigServe (
            const BoardConfig & config )  [virtual]
```

7.6.3.80 setChannelConfig()

```
BError BMeasureApi::BMeasure::setChannelConfig (
            const  BUInt32 & channelNumber,
            const ChannelConfig & channelConfig )
```

7.6.3.81 setChannelConfigFull()

```
BError BMeasureApi::BMeasure::setChannelConfigFull (
            const  BUInt64 & key,
            const  BUInt32 & channelNumber,
            const ChannelConfig & channelConfig )
```

7.6.3.82 setChannelConfigFullServe()

```
BError BMeasureApi::BMeasure::setChannelConfigFullServe (
            const  BUInt64 & key,
            const  BUInt32 & channelNumber,
            const ChannelConfig & channelConfig )  [virtual]
```

7.6.3.83 setChannelConfigServe()

```
BError BMeasureApi::BMeasure::setChannelConfigServe (
            const  BUInt32 & channelNumber,
            const ChannelConfig & channelConfig )  [virtual]
```

7.6.3.84 setConfig()

```
BError BMeasureApi::BMeasure::setConfig (
            const Configuration & config )
```

Set units configuration.

7.6.3.85 setConfigServe()

```
BError BMeasureApi::BMeasure::setConfigServe (
            const Configuration & config )  [virtual]
```

7.6.3.86 setDigital()

```
BError BMeasureApi::BMeasure::setDigital (
            const  BUInt32 & bits )
```

Set digital bits.

7.6.3.87 setDigitalServe()

```
BError BMeasureApi::BMeasure::setDigitalServe (
            const  BUInt32 & bits )  [virtual]
```

7.6.3.88 setMeasurementConfig()

```
BError BMeasureApi::BMeasure::setMeasurementConfig (
            const  Bool & save,
            const MeasurementConfig & measurementConfig )
```

Set measurement config.

7.6.3.89 setMeasurementConfigServe()

```
BError BMeasureApi::BMeasure::setMeasurementConfigServe (
            const  Bool & save,
            const MeasurementConfig & measurementConfig )  [virtual]
```

7.6.3.90 setMode()

```
BError BMeasureApi::BMeasure::setMode (
            const Mode & mode )
```

Set the current operational mode.

7.6.3.91 setModeServe()

```
BError BMeasureApi::BMeasure::setModeServe (
            const Mode & mode )  [virtual]
```

7.6.3.92 setRelay()

```
BError BMeasureApi::BMeasure::setRelay (
            const  BUInt32 & relayNum,
            const  Bool & state )
```

Set relay.

7.6.3.93 setRelayServe()

```
BError BMeasureApi::BMeasure::setRelayServe (
            const  BUInt32 & relayNum,
            const  Bool & state )  [virtual]
```

7.6.3.94 wifiAccesspointInfo()

```
BError BMeasureApi::BMeasure::wifiAccesspointInfo (
            const  BUInt32 & id,
            WifiAccessPoint & accessPoint )
```

Return information on Wifi accesspoint.

7.6.3.95 wifiAccesspointInfoServe()

```
BError BMeasureApi::BMeasure::wifiAccesspointInfoServe (
            const  BUInt32 & id,
            WifiAccessPoint & accessPoint )  [virtual]
```

7.6.3.96 wifiAccesspointNum()

```
BError BMeasureApi::BMeasure::wifiAccesspointNum (
             BUInt32 & numFound )
```

Return how many access points found.

7.6.3.97 wifiAccesspointNumServe()

```
BError BMeasureApi::BMeasure::wifiAccesspointNumServe (
             BUInt32 & numFound )  [virtual]
```

7.6.3.98 wifiCommand()

```
BError BMeasureApi::BMeasure::wifiCommand (
            const WifiCmd & wifiCmd )
```

Perform Wifi command.

7.6.3.99 wifiCommandServe()

```
BError BMeasureApi::BMeasure::wifiCommandServe (
            const WifiCmd & wifiCmd )  [virtual]
```

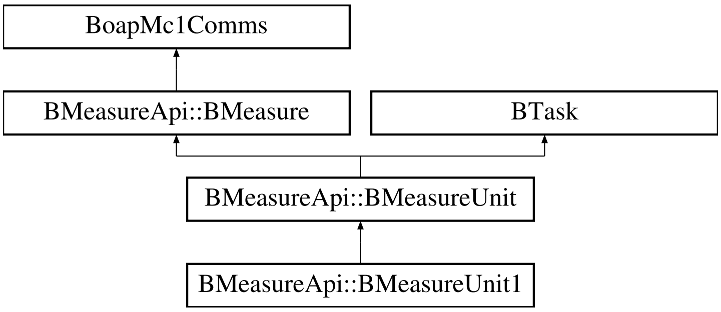
The documentation for this class was generated from the following files:

- BMeasureB.h
- BMeasureB.cpp

7.7 BMeasureApi::BMeasureUnit Class Reference

`#include <BMeasureUnit.h>`

Inheritance diagram for BMeasureApi::BMeasureUnit:



Public Member Functions

- BMeasureUnit (**Bool** threaded=0, **BUInt** reqSize=2048)
- virtual ∼BMeasureUnit ()
- **BError** connect (**BString** device)

  Connect to a device.
- void disconnect ()
- **BString** device ()
- **BString** serialNumber ()
- **BString** info ()
- **BUInt** numChannels ()

  The number of channels of data.
- void run ()

  Threaded run mode.
- virtual void disconnected ()
- virtual void sendDataServe (const DataBlock &dataBlock)
- virtual void sendDataFloatServe (const DataBlockFloat &dataBlockFloat)
- virtual void sendDataProcServe (const DataBlockProc &dataBlockProc)
- virtual **BError** setMeasurementConfig (const **Bool** &save, const MeasurementConfig &configMeasurement)
- virtual **BError** setChannelConfig (const **BUInt8** &channelNumber, const ChannelConfig &channelConfig)
- **BError** getNodeInfo (NodeInfo &nodeInfo)

  Override getNodeInfo.

Static Public Member Functions

- static **BError** findDevices (**BList**< BMeasureUnitDevice > &devices)

  Find available devices.
- static **BError** findDevicesUsb (**BList**< BMeasureUnitDevice > &devices)

  Find available devices on USB bus.
- static **BError** findDevicesNetwork (**BList**< BMeasureUnitDevice > &devices)

  Find available devices on Network.
- static void processdataBlock (const DataBlock &dataBlock, DataBlockFloat *dataBlockOut)

Static Public Attributes

- static int blockNumChannels = 16
- static int blockNumSamples = 13

Protected Attributes

- **BString** odevice
- NodeInfo onodeInfo
- Information oinfo

  Instrument info.
- MeasurementConfig oconfigMeasurement
- **BArray**< ChannelConfig > ochannels
- DataBlockFloat ∗ odataBlockFloat
- **BUInt32** osequenceNext
- **BUInt32** osampleCount
- **BUInt32** oblockCount
- **Bool** odisconnecting
- **BSemaphoreBool** oprocEnable

  Enable processing.
- **BSemaphoreBool** oprocRunning

  Processing is running.

Additional Inherited Members

7.7.1 Constructor & Destructor Documentation

7.7.1.1 BMeasureUnit()

```
BMeasureApi::BMeasureUnit::BMeasureUnit (
              Bool threaded = 0,
              BUInt reqSize = 2048 )
```

7.7.1.2 ∼BMeasureUnit()

```
BMeasureApi::BMeasureUnit::∼BMeasureUnit ( )  [virtual]
```

7.7.2 Member Function Documentation

7.7.2.1 connect()

```
BError BMeasureApi::BMeasureUnit::connect (
             BString device )
```

Connect to a device.

7.7.2.2 device()

```
BString BMeasureApi::BMeasureUnit::device ( )
```

7.7.2.3 disconnect()

```
void BMeasureApi::BMeasureUnit::disconnect ( )
```

7.7.2.4 disconnected()

```
void BMeasureApi::BMeasureUnit::disconnected ( )  [virtual]
```

Reimplemented in BMeasureApi::BMeasureUnit1.

7.7.2.5 findDevices()

```
BError BMeasureApi::BMeasureUnit::findDevices (
             BList< BMeasureUnitDevice > & devices )  [static]
```

Find available devices.

7.7.2.6 findDevicesNetwork()

```
BError BMeasureApi::BMeasureUnit::findDevicesNetwork (
             BList< BMeasureUnitDevice > & devices )  [static]
```

Find available devices on Network.

7.7.2.7 findDevicesUsb()

```
BError BMeasureApi::BMeasureUnit::findDevicesUsb (
            BList< BMeasureUnitDevice > & devices )  [static]
```

Find available devices on USB bus.

7.7.2.8 getNodeInfo()

```
BError BMeasureApi::BMeasureUnit::getNodeInfo (
            NodeInfo & nodeInfo )
```

Override getNodeInfo.

7.7.2.9 info()

```
BString BMeasureApi::BMeasureUnit::info ( )
```

7.7.2.10 numChannels()

```
BUInt BMeasureApi::BMeasureUnit::numChannels ( )
```

The number of channels of data.

7.7.2.11 processdataBlock()

```
void BMeasureApi::BMeasureUnit::processdataBlock (
            const DataBlock & dataBlock,
            DataBlockFloat * dataBlockOut )  [static]
```

7.7.2.12 run()

```
void BMeasureApi::BMeasureUnit::run ( )  [virtual]
```

Threaded run mode.

Reimplemented from **BTask**.

7.7.2.13 sendDataFloatServe()

```
void BMeasureApi::BMeasureUnit::sendDataFloatServe (
            const DataBlockFloat & dataBlockFloat )  [virtual]
```

Reimplemented in BMeasureApi::BMeasureUnit1.

7.7.2.14 sendDataProcServe()

```
void BMeasureApi::BMeasureUnit::sendDataProcServe (
            const DataBlockProc & dataBlockProc )  [virtual]
```

Reimplemented in BMeasureApi::BMeasureUnit1.

7.7.2.15 sendDataServe()

```
void BMeasureApi::BMeasureUnit::sendDataServe (
            const DataBlock & dataBlock )  [virtual]
```

Reimplemented from BMeasureApi::BMeasure.

7.7.2.16 serialNumber()

```
BString BMeasureApi::BMeasureUnit::serialNumber ( )
```

7.7.2.17 setChannelConfig()

```
BError BMeasureApi::BMeasureUnit::setChannelConfig (
            const  BUInt8 & channelNumber,
            const ChannelConfig & channelConfig )  [virtual]
```

7.7.2.18 setMeasurementConfig()

```
BError BMeasureApi::BMeasureUnit::setMeasurementConfig (
            const  Bool & save,
            const MeasurementConfig & configMeasurement )  [virtual]
```

7.7.3 Member Data Documentation

7.7.3.1 blockNumChannels

```
int BMeasureApi::BMeasureUnit::blockNumChannels = 16  [static]
```

7.7.3.2 blockNumSamples

```
int BMeasureApi::BMeasureUnit::blockNumSamples = 13  [static]
```

7.7.3.3 oblockCount

```
BUInt32 BMeasureApi::BMeasureUnit::oblockCount  [protected]
```

7.7.3.4 ochannels

```
BArray<ChannelConfig> BMeasureApi::BMeasureUnit::ochannels  [protected]
```

7.7.3.5 oconfigMeasurement

```
MeasurementConfig BMeasureApi::BMeasureUnit::oconfigMeasurement  [protected]
```

7.7.3.6 odataBlockFloat

```
DataBlockFloat* BMeasureApi::BMeasureUnit::odataBlockFloat  [protected]
```

7.7.3.7 odevice

```
BString BMeasureApi::BMeasureUnit::odevice  [protected]
```

7.7.3.8 odisconnecting

```
Bool BMeasureApi::BMeasureUnit::odisconnecting  [protected]
```

7.7.3.9 oinfo

```
Information BMeasureApi::BMeasureUnit::oinfo  [protected]
```

Instrument info.

7.7.3.10 onodeInfo

```
NodeInfo BMeasureApi::BMeasureUnit::onodeInfo  [protected]
```

7.7.3.11 oprocEnable

```
BSemaphoreBool BMeasureApi::BMeasureUnit::oprocEnable  [protected]
```

Enable processing.

7.7.3.12 oprocRunning

```
BSemaphoreBool BMeasureApi::BMeasureUnit::oprocRunning  [protected]
```

Processing is running.

7.7.3.13 osampleCount

```
BUInt32 BMeasureApi::BMeasureUnit::osampleCount  [protected]
```

7.7.3.14 osequenceNext

```
BUInt32 BMeasureApi::BMeasureUnit::osequenceNext  [protected]
```

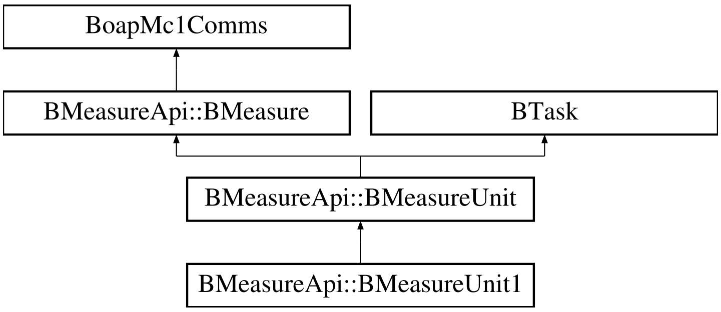
The documentation for this class was generated from the following files:

- BMeasureUnit.h
- BMeasureUnit.cpp

7.8 BMeasureApi::BMeasureUnit1 Class Reference

```
#include <BMeasureUnits.h>
```

Inheritance diagram for BMeasureApi::BMeasureUnit1:



Public Member Functions

- BMeasureUnit1 (BMeasureUnits &measureUnits, **BString** device, **Bool** threaded=0, **BUInt** reqSize=2048)
- **BString** serialNumber ()
- void setSerialNumber (**BString** serialNumber)
- void disconnected ()
- void sendStatusServe (const NodeStatus &nodeStatus)
- void sendMessageServe (const **BUInt32** &source, const **BString** &message)
- void sendDataFloatServe (const DataBlockFloat &dataBlockFloat)
- void sendDataProcServe (const DataBlockProc &dataBlockProc)

Public Attributes

- BMeasureUnits & omeasureUnits
- **Bool** oenabled
- **Bool** oconnected
- **BUInt** oorder
- **BUInt** osource
- **BString** oserialNumber

Additional Inherited Members

7.8.1 Constructor & Destructor Documentation

7.8.1.1 BMeasureUnit1()

```
BMeasureApi::BMeasureUnit1::BMeasureUnit1 (
             BMeasureUnits & measureUnits,
              BString device,
              Bool threaded = 0,
              BUInt reqSize = 2048 )
```

7.8.2 Member Function Documentation

7.8.2.1 disconnected()

```
void BMeasureApi::BMeasureUnit1::disconnected ( )  [virtual]
```

Reimplemented from BMeasureApi::BMeasureUnit.

7.8.2.2 sendDataFloatServe()

```
void BMeasureApi::BMeasureUnit1::sendDataFloatServe (
             const DataBlockFloat & dataBlockFloat )  [virtual]
```

Reimplemented from BMeasureApi::BMeasureUnit.

7.8.2.3 sendDataProcServe()

```
void BMeasureApi::BMeasureUnit1::sendDataProcServe (
             const DataBlockProc & dataBlockProc )  [virtual]
```

Reimplemented from BMeasureApi::BMeasureUnit.

7.8.2.4 sendMessageServe()

```
void BMeasureApi::BMeasureUnit1::sendMessageServe (
             const  BUInt32 & source,
             const  BString & message )  [virtual]
```

Reimplemented from BMeasureApi::BMeasure.

7.8.2.5 sendStatusServe()

```
void BMeasureApi::BMeasureUnit1::sendStatusServe (
             const NodeStatus & nodeStatus )  [virtual]
```

Reimplemented from BMeasureApi::BMeasure.

7.8.2.6 serialNumber()

```
BString BMeasureApi::BMeasureUnit1::serialNumber ( )
```

7.8.2.7 setSerialNumber()

```
void BMeasureApi::BMeasureUnit1::setSerialNumber (
              BString serialNumber )
```

7.8.3 Member Data Documentation

7.8.3.1 oconnected

```
Bool BMeasureApi::BMeasureUnit1::oconnected
```

7.8.3.2 oenabled

```
Bool BMeasureApi::BMeasureUnit1::oenabled
```

7.8.3.3 omeasureUnits

```
BMeasureUnits& BMeasureApi::BMeasureUnit1::omeasureUnits
```

7.8.3.4 oorder

```
BUInt BMeasureApi::BMeasureUnit1::oorder
```

7.8.3.5 oserialNumber

```
BString BMeasureApi::BMeasureUnit1::oserialNumber
```

7.8.3.6 osource

```
BUInt BMeasureApi::BMeasureUnit1::osource
```

The documentation for this class was generated from the following files:

- BMeasureUnits.h
- BMeasureUnits.cpp

7.9 BMeasureApi::BMeasureUnitDevice Class Reference

```
#include <BMeasureUnit.h>
```

Public Member Functions

- BMeasureUnitDevice (**BString** serialNumber="", **BString** device="")

Public Attributes

- **BString** serialNumber
- **BString** device

7.9.1 Constructor & Destructor Documentation

7.9.1.1 BMeasureUnitDevice()

```
BMeasureApi::BMeasureUnitDevice::BMeasureUnitDevice (
            BString serialNumber = "",
            BString device = "" )  [inline]
```

7.9.2 Member Data Documentation

7.9.2.1 device

```
BString BMeasureApi::BMeasureUnitDevice::device
```

7.9.2.2 serialNumber

```
BString BMeasureApi::BMeasureUnitDevice::serialNumber
```

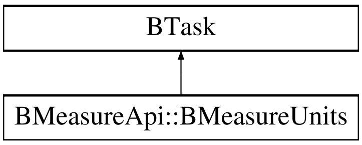
The documentation for this class was generated from the following file:

- BMeasureUnit.h

7.10 BMeasureApi::BMeasureUnits Class Reference

```
#include <BMeasureUnits.h>
```

Inheritance diagram for BMeasureApi::BMeasureUnits:



Public Member Functions

- BMeasureUnits (**Bool** threaded=0)
- virtual ∼BMeasureUnits ()
- void clear ()
- **BError** unitsFind ()
- **BError** unitAdd (**BString** serialNumber, **BString** device)
- **BError** unitDelete (**BString** device)
- void setMulti (**Bool** on)
- **BUInt32** unitsNum ()
- **BUInt32** unitsConnectedNum ()
- BMeasureUnit1 & unit (**BUInt** u)
- BMeasureUnit1 & unitMaster ()
- **BError** unitsConnect ()
- **Bool** unitsConnected ()
- **BError** unitsDisconnect ()
- virtual void disconnected ()
- **BError** unitSetOrder (**BUInt** u, **BUInt** order, **Bool** move)
- **BError** unitSetEnabled (**BUInt** u, **Bool** enable)
- **BError** dataSetNumStreams (**BUInt** num)

  Set the number of data output channels.
- void dataStreamEnable (**Bool** on)

  Enable the streaming of data.
- void dataClear ()
- **BUInt** dataAvailable (**BUInt** stream)
- **BError** dataWait (**BUInt** stream, **BTimeout** timeoutUs= **BTimeoutForever**)
- virtual void dataEvent (**BUInt** stream)
- DataBlockFloat ∗ dataRead (**BUInt** stream)
- void dataDone (**BUInt** stream)
- virtual void dataProcEvent (**BUInt** stream)

- DataBlockProc ∗ dataProcRead (**BUInt** stream)
- void dataProcDone (**BUInt** stream)
- void run ()

  Threaded run mode.
- void sendDataFloatQueue (const DataBlockFloat &dataBlock)
- void sendDataProcQueue (const DataBlockProc &dataBlock)
- void sendDataProcess ()
- void sendDataProcessTrigger ()
- void outputBlock (BMeasureUnitsDataBlock ∗block)
- virtual **BError** getNodeInfo (NodeInfo &nodeInfo)
- virtual **BError** login (const **BString** &userid, const **BString** &password)
- virtual **BError** logout ()
- virtual **BError** changePassword (const **BString** &userid, const **BString** &oldPassword, const **BString** &newPassword)
- virtual **BUInt** numChannels ()

  The number of channels of data.
- virtual **BError** setMode (const Mode &mode)

  Set the current operational mode.
- virtual **BError** getStatus (NodeStatus &nodeStatus)
- virtual void sendTime (const **BTimeUs** &time)

  Sends the current time.
- virtual **BError** getInformation (Information &info)
- virtual **BError** getInfoBlock (InfoBlock &infoBlock)
- virtual **BError** getChannelConfig (const **BUInt8** &channelNumber, ChannelConfig &channelConfig)
- virtual **BError** setChannelConfig (const **BUInt8** &channelNumber, const ChannelConfig &channelConfig)
- virtual **BError** getConfig (Configuration &config)

  Should we have this, not generic for different instruments ?
- virtual **BError** setConfig (const Configuration &config)

  Should we have this, not generic for different instruments ?
- virtual **BError** getMeasurementConfig (const **Bool** &saved, MeasurementConfig &measurement)

  Get measurement config.
- virtual **BError** setMeasurementConfig (const **Bool** &save, const MeasurementConfig &measurement)

  Set measurement config.
- virtual **BError** sendDataEnable (const DataSend &dataSend)

  Enables the sending of data.
- virtual **BError** getAwgConfig (const **Bool** &saved, AwgConfig &awgConfig)

  Get AWG Configuration.
- virtual **BError** setAwgConfig (const **Bool** &save, const AwgConfig &awgConfig)

  Configure AWG.
- virtual **BError** alarmsClear (const **BUInt32** &bits)

  Clear all alarms.
- virtual void sendStatusServe (const NodeStatus &nodeStatus)
- virtual void sendDataFloatServe (const DataBlockFloat &dataBlockFloat)
- virtual void sendDataProcServe (const DataBlockProc &dataBlockProc)
- virtual void sendMessage (**BUInt32** &source, **BString** &message)
- virtual void sendMessageServe (const **BUInt32** &source, const **BString** &message)
- void debugPrint ()

Private Member Functions

- BMeasureUnitsDataBlock ∗ getFreeBlock (**BUInt** numSamples)

Private Attributes

- **BSemaphoreBool** oprocEnable

  Enable processing.
- **BSemaphoreBool** oprocRunning

  Processing is running.
- **Bool** omulti

  Multiple BMeasures connected together.
- **BMutex** olockUnits
- **BList**< BMeasureUnit1 ∗ > ounits
- **BInt** ounitMaster
- **BUInt** onumConnected
- **BUInt** onumChannels
- **BUInt** odataStreamNum
- **BUInt32** ofill
- **BUInt** onumBlocks
- **BMutex** olockInput
- **BList**< BMeasureUnitsDataBlock ∗ > odataBlocksFree
- **BList**< BMeasureUnitsDataBlock ∗ > odataBlocksIn
- **BList**< BMeasureUnitsDataBlock ∗ > odataBlocksProcess
- **BCondInt** odataBlocksProcessNum
- **BMutex** olockOutput
- **BList**< BMeasureUnitsDataBlock ∗ > odataBlocksOut [2]
- **BCondInt** odataBlocksOutCount [2]
- **BMutex** olockProcInput
- **BList**< DataBlockProc > odataProcBlocks
- MeasurementConfig olocalTrigger
- **Bool** otriggered
- **BUInt** ostartSample

Additional Inherited Members

7.10.1 Constructor & Destructor Documentation

7.10.1.1 BMeasureUnits()

```
BMeasureApi::BMeasureUnits::BMeasureUnits (
              Bool threaded = 0 )
```

7.10.1.2 ∼BMeasureUnits()

```
BMeasureApi::BMeasureUnits::∼BMeasureUnits ( )  [virtual]
```

7.10.2 Member Function Documentation

7.10.2.1 alarmsClear()

```
 BError BMeasureApi::BMeasureUnits::alarmsClear (
             const  BUInt32 & bits )  [virtual]
```

Clear all alarms.

7.10.2.2 changePassword()

```
 BError BMeasureApi::BMeasureUnits::changePassword (
             const  BString & userid,
             const  BString & oldPassword,
             const  BString & newPassword )  [virtual]
```

7.10.2.3 clear()

```
void BMeasureApi::BMeasureUnits::clear ( )
```

7.10.2.4 dataAvailable()

```
 BUInt BMeasureApi::BMeasureUnits::dataAvailable (
              BUInt stream )
```

7.10.2.5 dataClear()

```
void BMeasureApi::BMeasureUnits::dataClear ( )
```

7.10.2.6 dataDone()

```
void BMeasureApi::BMeasureUnits::dataDone (
              BUInt stream )
```

7.10.2.7 dataEvent()

```
void BMeasureApi::BMeasureUnits::dataEvent (
              BUInt stream )  [virtual]
```

7.10.2.8 dataProcDone()

```
void BMeasureApi::BMeasureUnits::dataProcDone (
              BUInt stream )
```

7.10.2.9 dataProcEvent()

```
void BMeasureApi::BMeasureUnits::dataProcEvent (
              BUInt stream )  [virtual]
```

7.10.2.10 dataProcRead()

```
DataBlockProc * BMeasureApi::BMeasureUnits::dataProcRead (
              BUInt stream )
```

7.10.2.11 dataRead()

```
DataBlockFloat * BMeasureApi::BMeasureUnits::dataRead (
              BUInt stream )
```

7.10.2.12 dataSetNumStreams()

```
 BError BMeasureApi::BMeasureUnits::dataSetNumStreams (
              BUInt num )
```

Set the number of data output channels.

7.10.2.13 dataStreamEnable()

```
void BMeasureApi::BMeasureUnits::dataStreamEnable (
              Bool on )
```

Enable the streaming of data.

7.10.2.14 dataWait()

```
 BError BMeasureApi::BMeasureUnits::dataWait (
              BUInt stream,
              BTimeout timeoutUs =  BTimeoutForever )
```

7.10.2.15 debugPrint()

```
void BMeasureApi::BMeasureUnits::debugPrint ( )
```

7.10.2.16 disconnected()

```
void BMeasureApi::BMeasureUnits::disconnected ( )  [virtual]
```

7.10.2.17 getAwgConfig()

```
 BError BMeasureApi::BMeasureUnits::getAwgConfig (
             const  Bool & saved,
             AwgConfig & awgConfig )  [virtual]
```

Get AWG Configuration.

7.10.2.18 getChannelConfig()

```
 BError BMeasureApi::BMeasureUnits::getChannelConfig (
             const  BUInt8 & channelNumber,
             ChannelConfig & channelConfig )  [virtual]
```

7.10.2.19 getConfig()

```
BError BMeasureApi::BMeasureUnits::getConfig (
            Configuration & config )  [virtual]
```

Should we have this, not generic for different instruments ?

7.10.2.20 getFreeBlock()

```
BMeasureUnitsDataBlock * BMeasureApi::BMeasureUnits::getFreeBlock (
              BUInt numSamples )  [private]
```

7.10.2.21 getInfoBlock()

```
BError BMeasureApi::BMeasureUnits::getInfoBlock (
            InfoBlock & infoBlock )  [virtual]
```

7.10.2.22 getInformation()

```
BError BMeasureApi::BMeasureUnits::getInformation (
            Information & info )  [virtual]
```

7.10.2.23 getMeasurementConfig()

```
BError BMeasureApi::BMeasureUnits::getMeasurementConfig (
            const  Bool & saved,
            MeasurementConfig & measurement )  [virtual]
```

Get measurement config.

7.10.2.24 getNodeInfo()

```
BError BMeasureApi::BMeasureUnits::getNodeInfo (
            NodeInfo & nodeInfo )  [virtual]
```

7.10.2.25 getStatus()

```
BError BMeasureApi::BMeasureUnits::getStatus (
            NodeStatus & nodeStatus )  [virtual]
```

7.10.2.26 login()

```
BError BMeasureApi::BMeasureUnits::login (
            const  BString & userid,
            const  BString & password )  [virtual]
```

7.10.2.27 logout()

```
BError BMeasureApi::BMeasureUnits::logout ( )  [virtual]
```

7.10.2.28 numChannels()

```
BUInt BMeasureApi::BMeasureUnits::numChannels ( )  [virtual]
```

The number of channels of data.

7.10.2.29 outputBlock()

```
void BMeasureApi::BMeasureUnits::outputBlock (
            BMeasureUnitsDataBlock * block )
```

7.10.2.30 run()

```
void BMeasureApi::BMeasureUnits::run ( )  [virtual]
```

Threaded run mode.

Reimplemented from **BTask**.

7.10.2.31 sendDataEnable()

```
BError BMeasureApi::BMeasureUnits::sendDataEnable (
             const DataSend & dataSend )  [virtual]
```

Enables the sending of data.

7.10.2.32 sendDataFloatQueue()

```
void BMeasureApi::BMeasureUnits::sendDataFloatQueue (
             const DataBlockFloat & dataBlock )
```

7.10.2.33 sendDataFloatServe()

```
void BMeasureApi::BMeasureUnits::sendDataFloatServe (
             const DataBlockFloat & dataBlockFloat )  [virtual]
```

7.10.2.34 sendDataProcess()

```
void BMeasureApi::BMeasureUnits::sendDataProcess ( )
```

7.10.2.35 sendDataProcessTrigger()

```
void BMeasureApi::BMeasureUnits::sendDataProcessTrigger ( )
```

7.10.2.36 sendDataProcQueue()

```
void BMeasureApi::BMeasureUnits::sendDataProcQueue (
             const DataBlockProc & dataBlock )
```

7.10.2.37 sendDataProcServe()

```
void BMeasureApi::BMeasureUnits::sendDataProcServe (
             const DataBlockProc & dataBlockProc )  [virtual]
```

7.10.2.38 sendMessage()

```
void BMeasureApi::BMeasureUnits::sendMessage (
              BUInt32 & source,
              BString & message )  [virtual]
```

7.10.2.39 sendMessageServe()

```
void BMeasureApi::BMeasureUnits::sendMessageServe (
             const  BUInt32 & source,
             const  BString & message )  [virtual]
```

7.10.2.40 sendStatusServe()

```
void BMeasureApi::BMeasureUnits::sendStatusServe (
             const NodeStatus & nodeStatus )  [virtual]
```

7.10.2.41 sendTime()

```
void BMeasureApi::BMeasureUnits::sendTime (
             const  BTimeUs & time )  [virtual]
```

Sends the current time.

7.10.2.42 setAwgConfig()

```
 BError BMeasureApi::BMeasureUnits::setAwgConfig (
             const  Bool & save,
             const AwgConfig & awgConfig )  [virtual]
```

Configure AWG.

7.10.2.43 setChannelConfig()

```
 BError BMeasureApi::BMeasureUnits::setChannelConfig (
             const  BUInt8 & channelNumber,
             const ChannelConfig & channelConfig )  [virtual]
```

7.10.2.44 setConfig()

```
 BError BMeasureApi::BMeasureUnits::setConfig (
            const Configuration & config )  [virtual]
```

Should we have this, not generic for different instruments ?

7.10.2.45 setMeasurementConfig()

```
 BError BMeasureApi::BMeasureUnits::setMeasurementConfig (
            const  Bool & save,
            const MeasurementConfig & measurement )  [virtual]
```

Set measurement config.

7.10.2.46 setMode()

```
 BError BMeasureApi::BMeasureUnits::setMode (
            const Mode & mode )  [virtual]
```

Set the current operational mode.

7.10.2.47 setMulti()

```
void BMeasureApi::BMeasureUnits::setMulti (
              Bool on )
```

7.10.2.48 unit()

```
BMeasureUnit1 & BMeasureApi::BMeasureUnits::unit (
              BUInt u )
```

7.10.2.49 unitAdd()

```
 BError BMeasureApi::BMeasureUnits::unitAdd (
             BString serialNumber,
             BString device )
```

7.10.2.50 unitDelete()

```
BError BMeasureApi::BMeasureUnits::unitDelete (
             BString device )
```

7.10.2.51 unitMaster()

```
BMeasureUnit1 & BMeasureApi::BMeasureUnits::unitMaster ( )
```

7.10.2.52 unitsConnect()

```
BError BMeasureApi::BMeasureUnits::unitsConnect ( )
```

7.10.2.53 unitsConnected()

```
Bool BMeasureApi::BMeasureUnits::unitsConnected ( )
```

7.10.2.54 unitsConnectedNum()

```
BUInt BMeasureApi::BMeasureUnits::unitsConnectedNum ( )
```

7.10.2.55 unitsDisconnect()

```
BError BMeasureApi::BMeasureUnits::unitsDisconnect ( )
```

7.10.2.56 unitSetEnabled()

```
BError BMeasureApi::BMeasureUnits::unitSetEnabled (
             BUInt u,
             Bool enable )
```

7.10.2.57 unitSetOrder()

```
BError BMeasureApi::BMeasureUnits::unitSetOrder (
            BUInt u,
            BUInt order,
            Bool move )
```

7.10.2.58 unitsFind()

```
BError BMeasureApi::BMeasureUnits::unitsFind ( )
```

7.10.2.59 unitsNum()

```
BUInt BMeasureApi::BMeasureUnits::unitsNum ( )
```

7.10.3 Member Data Documentation

7.10.3.1 odataBlocksFree

```
BList<BMeasureUnitsDataBlock*> BMeasureApi::BMeasureUnits::odataBlocksFree  [private]
```

7.10.3.2 odataBlocksIn

```
BList<BMeasureUnitsDataBlock*> BMeasureApi::BMeasureUnits::odataBlocksIn  [private]
```

7.10.3.3 odataBlocksOut

```
BList<BMeasureUnitsDataBlock*> BMeasureApi::BMeasureUnits::odataBlocksOut[2]  [private]
```

7.10.3.4 odataBlocksOutCount

```
BCondInt BMeasureApi::BMeasureUnits::odataBlocksOutCount[2]  [private]
```

7.10.3.5 odataBlocksProcess

BList<BMeasureUnitsDataBlock*> BMeasureApi::BMeasureUnits::odataBlocksProcess [private]

7.10.3.6 odataBlocksProcessNum

BCondInt BMeasureApi::BMeasureUnits::odataBlocksProcessNum [private]

7.10.3.7 odataProcBlocks

BList<DataBlockProc> BMeasureApi::BMeasureUnits::odataProcBlocks [private]

7.10.3.8 odataStreamNum

BUInt BMeasureApi::BMeasureUnits::odataStreamNum [private]

7.10.3.9 ofill

BUInt32 BMeasureApi::BMeasureUnits::ofill [private]

7.10.3.10 olocalTrigger

MeasurementConfig BMeasureApi::BMeasureUnits::olocalTrigger [private]

7.10.3.11 olockInput

BMutex BMeasureApi::BMeasureUnits::olockInput [private]

7.10.3.12 olockOutput

BMutex BMeasureApi::BMeasureUnits::olockOutput [private]

7.10.3.13 olockProcInput

```
BMutex BMeasureApi::BMeasureUnits::olockProcInput  [private]
```

7.10.3.14 olockUnits

```
BMutex BMeasureApi::BMeasureUnits::olockUnits  [private]
```

7.10.3.15 omulti

```
Bool BMeasureApi::BMeasureUnits::omulti  [private]
```

Multiple BMeasures connected together.

7.10.3.16 onumBlocks

```
BUInt BMeasureApi::BMeasureUnits::onumBlocks  [private]
```

7.10.3.17 onumChannels

```
BUInt BMeasureApi::BMeasureUnits::onumChannels  [private]
```

7.10.3.18 onumConnected

```
BUInt BMeasureApi::BMeasureUnits::onumConnected  [private]
```

7.10.3.19 oprocEnable

```
BSemaphoreBool BMeasureApi::BMeasureUnits::oprocEnable  [private]
```

Enable processing.

7.10.3.20 oprocRunning

```
BSemaphoreBool BMeasureApi::BMeasureUnits::oprocRunning [private]
```

Processing is running.

7.10.3.21 ostartSample

```
BUInt BMeasureApi::BMeasureUnits::ostartSample [private]
```

7.10.3.22 otriggered

```
Bool BMeasureApi::BMeasureUnits::otriggered [private]
```

7.10.3.23 ounitMaster

```
BInt BMeasureApi::BMeasureUnits::ounitMaster [private]
```

7.10.3.24 ounits

```
BList<BMeasureUnit1*> BMeasureApi::BMeasureUnits::ounits [private]
```

The documentation for this class was generated from the following files:

- BMeasureUnits.h
- BMeasureUnits.cpp

7.11 BMeasureApi::BMeasureUnitsDataBlock Class Reference

```
#include <BMeasureUnits.h>
```

Public Member Functions

- BMeasureUnitsDataBlock (**BUInt** numChannels=0, **BUInt** numSamples=0)
- ∼BMeasureUnitsDataBlock ()
- void init (**BUInt** numChannels, **BUInt** numSamples)

Public Attributes

- DataBlock ∗ odataBlock
- **BUInt32** ofill
- **BUInt** oinUse

7.11.1 Constructor & Destructor Documentation

7.11.1.1 BMeasureUnitsDataBlock()

```
BMeasureApi::BMeasureUnitsDataBlock::BMeasureUnitsDataBlock (
              BUInt numChannels = 0,
              BUInt numSamples = 0 )
```

7.11.1.2 ∼BMeasureUnitsDataBlock()

```
BMeasureApi::BMeasureUnitsDataBlock::∼BMeasureUnitsDataBlock ( )
```

7.11.2 Member Function Documentation

7.11.2.1 init()

```
void BMeasureApi::BMeasureUnitsDataBlock::init (
              BUInt numChannels,
              BUInt numSamples )
```

7.11.3 Member Data Documentation

7.11.3.1 odataBlock

```
DataBlock∗ BMeasureApi::BMeasureUnitsDataBlock::odataBlock
```

7.11.3.2 ofill

```
BUInt32 BMeasureApi::BMeasureUnitsDataBlock::ofill
```

7.11.3.3 oinUse

```
BUInt BMeasureApi::BMeasureUnitsDataBlock::oinUse
```

The documentation for this class was generated from the following files:

- BMeasureUnits.h
- BMeasureUnits.cpp

7.12 BMeasureApi::BoardConfig Class Reference

```
#include <BMeasureD.h>
```

Static Public Member Functions

- static const **BObjMember** $*$ getMembers ()

Public Attributes

- **BUInt32** magic
- Version hardwareVersion
- **BChar** serialNumber [12]
- **BTime** buildTime
- **BUInt8** macAddress [6]
- **BUInt8** testMode
- **BUInt8** spare0
- **BTime** calibTime
- **BFloat32** calibTemp
- **BFloat64** calibDacOffsets [2]
- **BFloat64** calibDacScales [2]
- **BFloat64** calibAdcOffsets [8]
- **BFloat64** calibAdcScales [8]
- **BFloat64** calibAttenScales [8]
- **BFloat64** calibFiveVolts
- Version fpgaVersion
- Version wifiVersion
- **BUInt32** spare [8]

7.12.1 Member Function Documentation

7.12.1.1 getMembers()

```
const BObjMember * BMeasureApi::BoardConfig::getMembers ( ) [static]
```

7.12.2 Member Data Documentation

7.12.2.1 buildTime

```
BTime BMeasureApi::BoardConfig::buildTime
```

7.12.2.2 calibAdcOffsets

```
BFloat64 BMeasureApi::BoardConfig::calibAdcOffsets[8]
```

7.12.2.3 calibAdcScales

```
BFloat64 BMeasureApi::BoardConfig::calibAdcScales[8]
```

7.12.2.4 calibAttenScales

```
BFloat64 BMeasureApi::BoardConfig::calibAttenScales[8]
```

7.12.2.5 calibDacOffsets

```
BFloat64 BMeasureApi::BoardConfig::calibDacOffsets[2]
```

7.12.2.6 calibDacScales

```
BFloat64 BMeasureApi::BoardConfig::calibDacScales[2]
```

7.12.2.7 calibFiveVolts

```
BFloat64 BMeasureApi::BoardConfig::calibFiveVolts
```

7.12.2.8 calibTemp

```
BFloat32 BMeasureApi::BoardConfig::calibTemp
```

7.12.2.9 calibTime

```
BTime BMeasureApi::BoardConfig::calibTime
```

7.12.2.10 fpgaVersion

```
Version BMeasureApi::BoardConfig::fpgaVersion
```

7.12.2.11 hardwareVersion

```
Version BMeasureApi::BoardConfig::hardwareVersion
```

7.12.2.12 macAddress

```
BUInt8 BMeasureApi::BoardConfig::macAddress[6]
```

7.12.2.13 magic

```
BUInt32 BMeasureApi::BoardConfig::magic
```

7.12.2.14 serialNumber

```
BChar BMeasureApi::BoardConfig::serialNumber[12]
```

7.12.2.15 spare

```
BUInt32 BMeasureApi::BoardConfig::spare[8]
```

7.12.2.16 spare0

```
BUInt8 BMeasureApi::BoardConfig::spare0
```

7.12.2.17 testMode

```
BUInt8 BMeasureApi::BoardConfig::testMode
```

7.12.2.18 wifiVersion

```
Version BMeasureApi::BoardConfig::wifiVersion
```

The documentation for this class was generated from the following files:

- BMeasureD.h
- BMeasureD.cpp

7.13 BMeasureApi::CalibrateInfo Class Reference

```
#include <BMeasureD.h>
```

Static Public Member Functions

- static const **BObjMember** ∗ getMembers ()

Public Attributes

- **BUInt32** stage

  Stage to run.
- **BUInt32** spare
- **BFloat64** calibrateFrequency

  The Awg frequency for calibration.
- **BFloat64** calibrateAmplitude

  The Awg amplitude for calibration.
- **BFloat64** calibrateTime

  Number of seconds to calibrate over (synced to multiple AWG cycles)
- **BFloat64** value

  Target/Set Value.
- **BUInt32** channelMask

  Bitmask of channels to be calibrated. 0xFFFF is all channels.
- **BUInt32** numAverage

  The number of runs to average over.
- **BFloat64** sampleRate

  The data processing sample rate to use.

7.13.1 Member Function Documentation

7.13.1.1 getMembers()

```
const BObjMember * BMeasureApi::CalibrateInfo::getMembers ( ) [static]
```

7.13.2 Member Data Documentation

7.13.2.1 calibrateAmplitude

```
BFloat64 BMeasureApi::CalibrateInfo::calibrateAmplitude
```

The Awg amplitude for calibration.

7.13.2.2 calibrateFrequency

```
BFloat64 BMeasureApi::CalibrateInfo::calibrateFrequency
```

The Awg frequency for calibration.

7.13.2.3 calibrateTime

```
BFloat64 BMeasureApi::CalibrateInfo::calibrateTime
```

Number of seconds to calibrate over (synced to multiple AWG cycles)

7.13.2.4 channelMask

```
BUInt32 BMeasureApi::CalibrateInfo::channelMask
```

Bitmask of channels to be calibrated. 0xFFFF is all channels.

7.13.2.5 numAverage

```
BUInt32 BMeasureApi::CalibrateInfo::numAverage
```

The number of runs to average over.

7.13.2.6 sampleRate

```
BFloat64 BMeasureApi::CalibrateInfo::sampleRate
```

The data processing sample rate to use.

7.13.2.7 spare

```
BUInt32 BMeasureApi::CalibrateInfo::spare
```

7.13.2.8 stage

```
BUInt32 BMeasureApi::CalibrateInfo::stage
```

Stage to run.

7.13.2.9 value

```
BFloat64 BMeasureApi::CalibrateInfo::value
```

Target/Set Value.

The documentation for this class was generated from the following files:

- BMeasureD.h
- BMeasureD.cpp

7.14 BMeasureApi::ChannelConfig Class Reference

Channel configuration.

```
#include <BMeasureD.h>
```

Static Public Member Functions

- static const **BObjMember** ∗ getMembers ()

Public Attributes

- **BUInt8** number

  The channel number.
- **BUInt8** enabled

  Channel is enabled.
- **BUInt8** attenuator

  Attenuator number in use.
- ChannelType type

  The channel type.
- SampleType sampleType

  The sample type.
- **BUInt8** spare0 [3]
- **BUInt32** dataChannel

  Data channel.
- **BChar** id [16]
- **BChar** name [16]
- **BChar** siUnits [8]
- **BUInt32** spare1
- **BFloat64** calibOffset

  The calibration data offset.
- **BFloat64** calibScale

  The calibration data scale factor to volts.
- **BFloat64** calibScaleAtten1

  Attenuator 1 scaling.
- **BFloat64** pgaGain

  The PGA gain.
- **BFloat64** scale

  The user data scale factor.
- **BFloat64** offset

  The user data offset.
- **BChar** process [32]

7.14.1 Detailed Description

Channel configuration.

7.14.2 Member Function Documentation

7.14.2.1 getMembers()

```
const BObjMember * BMeasureApi::ChannelConfig::getMembers ( ) [static]
```

7.14.3 Member Data Documentation

7.14.3.1 attenuator

```
BUInt8 BMeasureApi::ChannelConfig::attenuator
```

Attenuator number in use.

7.14.3.2 calibOffset

```
BFloat64 BMeasureApi::ChannelConfig::calibOffset
```

The calibration data offset.

7.14.3.3 calibScale

```
BFloat64 BMeasureApi::ChannelConfig::calibScale
```

The calibration data scale factor to volts.

7.14.3.4 calibScaleAtten1

```
BFloat64 BMeasureApi::ChannelConfig::calibScaleAtten1
```

Attenuator 1 scaling.

7.14.3.5 dataChannel

```
BUInt32 BMeasureApi::ChannelConfig::dataChannel
```

Data channel.

7.14.3.6 enabled

```
BUInt8 BMeasureApi::ChannelConfig::enabled
```

Channel is enabled.

7.14.3.7 id

```
BChar BMeasureApi::ChannelConfig::id[16]
```

7.14.3.8 name

```
BChar BMeasureApi::ChannelConfig::name[16]
```

7.14.3.9 number

```
BUInt8 BMeasureApi::ChannelConfig::number
```

The channel number.

7.14.3.10 offset

```
BFloat64 BMeasureApi::ChannelConfig::offset
```

The user data offset.

7.14.3.11 pgaGain

```
BFloat64 BMeasureApi::ChannelConfig::pgaGain
```

The PGA gain.

7.14.3.12 process

```
BChar BMeasureApi::ChannelConfig::process[32]
```

7.14.3.13 sampleType

```
SampleType BMeasureApi::ChannelConfig::sampleType
```

The sample type.

7.14.3.14 scale

```
BFloat64 BMeasureApi::ChannelConfig::scale
```

The user data scale factor.

7.14.3.15 siUnits

```
BChar BMeasureApi::ChannelConfig::siUnits[8]
```

7.14.3.16 spare0

```
BUInt8 BMeasureApi::ChannelConfig::spare0[3]
```

7.14.3.17 spare1

```
BUInt32 BMeasureApi::ChannelConfig::spare1
```

7.14.3.18 type

```
ChannelType BMeasureApi::ChannelConfig::type
```

The channel type.

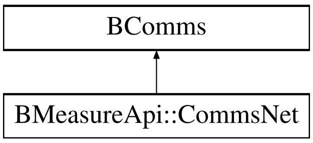
The documentation for this class was generated from the following files:

- BMeasureD.h
- BMeasureD.cpp

7.15 BMeasureApi::CommsNet Class Reference

```
#include <CommsNet.h>
```

Inheritance diagram for BMeasureApi::CommsNet:



Public Member Functions

- CommsNet (**BUInt** rxFifoSize=1024, **BUInt** txFifoSize=1024)
- ∼CommsNet ()
- **BError** init ()
- **BError** connect (**BString** host, **BUInt16** port)
- **BError** disconnect ()
- **BUInt** readAvailable ()
- **BError** wait (**BUInt32** eventSet, **BTimeout** timeout=-1, **BUInt32** num=1)
- **BError** read (void ∗ **data**, **BUInt32** num, **BUInt32** &nt)
- **BUInt** writeAvailable ()
- **BError** write (const void ∗ **data**, **BUInt32** nBytes, **BUInt32** &nt)
- **BError** writeChunks (const **BDataChunk** ∗chunks, **BUInt** nChunks, **BUInt32** &nt)

Protected Attributes

- **BSocket** osocket
- **Bool** oinWait
- **Bool** oterminating

Additional Inherited Members

7.15.1 Constructor & Destructor Documentation

7.15.1.1 CommsNet()

```
BMeasureApi::CommsNet::CommsNet (
            BUInt rxFifoSize = 1024,
            BUInt txFifoSize = 1024 )
```

7.15.1.2 ∼CommsNet()

```
BMeasureApi::CommsNet::∼CommsNet ( )
```

7.15.2 Member Function Documentation

7.15.2.1 connect()

```
BError BMeasureApi::CommsNet::connect (
            BString host,
            BUInt16 port )
```

7.15.2.2 disconnect()

```
BError BMeasureApi::CommsNet::disconnect ( )  [virtual]
```

Reimplemented from **BComms**.

7.15.2.3 init()

```
BError BMeasureApi::CommsNet::init ( )  [virtual]
```

Reimplemented from **BComms**.

7.15.2.4 read()

```
BError BMeasureApi::CommsNet::read (
           void * data,
            BUInt32 num,
            BUInt32 & nt )  [virtual]
```

Implements **BComms**.

7.15.2.5 readAvailable()

```
BUInt BMeasureApi::CommsNet::readAvailable ( )  [virtual]
```

Reimplemented from **BComms**.

7.15.2.6 wait()

```
BError BMeasureApi::CommsNet::wait (
             BUInt32 eventSet,
             BTimeout timeout = -1,
             BUInt32 num = 1 )
```

7.15.2.7 write()

```
BError BMeasureApi::CommsNet::write (
            const void * data,
             BUInt32 nBytes,
             BUInt32 & nt )  [virtual]
```

Implements **BComms**.

7.15.2.8 writeAvailable()

```
BUInt BMeasureApi::CommsNet::writeAvailable ( )  [virtual]
```

Reimplemented from **BComms**.

7.15.2.9 writeChunks()

```
BError BMeasureApi::CommsNet::writeChunks (
            const  BDataChunk * chunks,
             BUInt nChunks,
             BUInt32 & nt )  [virtual]
```

Reimplemented from **BComms**.

7.15.3 Member Data Documentation

7.15.3.1 oinWait

```
Bool BMeasureApi::CommsNet::oinWait  [protected]
```

7.15.3.2 osocket

```
BSocket BMeasureApi::CommsNet::osocket [protected]
```

7.15.3.3 oterminating

```
Bool BMeasureApi::CommsNet::oterminating [protected]
```

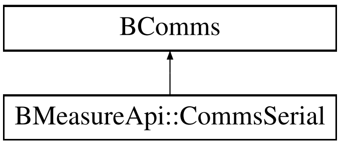
The documentation for this class was generated from the following files:

- CommsNet.h
- CommsNet.cpp

7.16 BMeasureApi::CommsSerial Class Reference

```
#include <CommsSerial.h>
```

Inheritance diagram for BMeasureApi::CommsSerial:



Public Member Functions

- CommsSerial ()
- ∼CommsSerial ()
- **BError** connect (**BString** device)
- **BError** disconnect ()
- **BUInt** readAvailable ()
- **BError** read (void ∗ **data**, **BUInt32** num, **BUInt32** &nTrans)
- **BError** write (const void ∗ **data**, **BUInt32** num, **BUInt32** &nTrans)
- **BError** wait (**BUInt32** eventSet, **BTimeout** timeout=-1, **BUInt32** num=1)

Private Attributes

- **BString** odevice
- int oserialPort

Additional Inherited Members

7.16.1 Constructor & Destructor Documentation

7.16.1.1 CommsSerial()

```
BMeasureApi::CommsSerial::CommsSerial ( )
```

7.16.1.2 ∼CommsSerial()

```
BMeasureApi::CommsSerial::∼CommsSerial ( )
```

7.16.2 Member Function Documentation

7.16.2.1 connect()

```
 BError BMeasureApi::CommsSerial::connect (
              BString device )
```

7.16.2.2 disconnect()

```
 BError BMeasureApi::CommsSerial::disconnect ( )  [virtual]
```

Reimplemented from **BComms**.

7.16.2.3 read()

```
 BError BMeasureApi::CommsSerial::read (
             void * data,
              BUInt32 num,
              BUInt32 & nTrans )  [virtual]
```

Implements **BComms**.

7.16.2.4 readAvailable()

```
 BUInt BMeasureApi::CommsSerial::readAvailable ( )  [virtual]
```

Reimplemented from **BComms**.

7.16.2.5 wait()

```
BError BMeasureApi::CommsSerial::wait (
            BUInt32 eventSet,
            BTimeout timeout = -1,
            BUInt32 num = 1 )
```

7.16.2.6 write()

```
BError BMeasureApi::CommsSerial::write (
            const void * data,
            BUInt32 num,
            BUInt32 & nTrans ) [virtual]
```

Implements **BComms**.

7.16.3 Member Data Documentation

7.16.3.1 odevice

```
BString BMeasureApi::CommsSerial::odevice [private]
```

7.16.3.2 oserialPort

```
int BMeasureApi::CommsSerial::oserialPort [private]
```

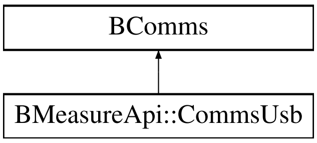
The documentation for this class was generated from the following file:

- CommsSerial.h

7.17 BMeasureApi::CommsUsb Class Reference

```
#include <CommsUsb.h>
```

Inheritance diagram for BMeasureApi::CommsUsb:



Public Member Functions

- CommsUsb ()
- ∼CommsUsb ()
- **BError** connect (**BString** device)
- **BError** disconnect ()
- **BUInt** readAvailable ()
- **BError** read (void ∗ **data**, **BUInt32** num, **BUInt32** &nTrans)
- **BError** write (const void ∗ **data**, **BUInt32** num, **BUInt32** &nTrans)
- **BError** wait (**BUInt32** eventSet, **BTimeout** timeout=-1, **BUInt32** num=1)

Private Member Functions

- **BError** readChunk (**BTimeout** timeout)

Private Attributes

- **BString** odevice
- libusb_context ∗ ocontext
- libusb_device_handle ∗ odev
- char obuffer [102400]
- **BUInt** onum
- **Bool** oterminated
- **Bool** oterminating
- **Bool** ousbDisconnected

Additional Inherited Members

7.17.1 Constructor & Destructor Documentation

7.17.1.1 CommsUsb()

```
BMeasureApi::CommsUsb::CommsUsb ( )
```

7.17.1.2 ∼CommsUsb()

```
BMeasureApi::CommsUsb::∼CommsUsb ( )
```

7.17.2 Member Function Documentation

7.17.2.1 connect()

```
BError BMeasureApi::CommsUsb::connect (
            BString device )
```

7.17.2.2 disconnect()

```
BError BMeasureApi::CommsUsb::disconnect ( )  [virtual]
```

Reimplemented from **BComms**.

7.17.2.3 read()

```
BError BMeasureApi::CommsUsb::read (
           void * data,
            BUInt32 num,
            BUInt32 & nTrans )  [virtual]
```

Implements **BComms**.

7.17.2.4 readAvailable()

```
BUInt BMeasureApi::CommsUsb::readAvailable ( )  [virtual]
```

Reimplemented from **BComms**.

7.17.2.5 readChunk()

```
BError BMeasureApi::CommsUsb::readChunk (
            BTimeout timeout )  [private]
```

7.17.2.6 wait()

```
BError BMeasureApi::CommsUsb::wait (
            BUInt32 eventSet,
            BTimeout timeout = -1,
            BUInt32 num = 1 )
```

7.17.2.7 write()

```
BError BMeasureApi::CommsUsb::write (
            const void * data,
            BUInt32 num,
            BUInt32 & nTrans )  [virtual]
```

Implements **BComms**.

7.17.3 Member Data Documentation

7.17.3.1 obuffer

```
char BMeasureApi::CommsUsb::obuffer[102400]  [private]
```

7.17.3.2 ocontext

```
libusb_context* BMeasureApi::CommsUsb::ocontext  [private]
```

7.17.3.3 odev

```
libusb_device_handle* BMeasureApi::CommsUsb::odev  [private]
```

7.17.3.4 odevice

```
BString BMeasureApi::CommsUsb::odevice  [private]
```

7.17.3.5 onum

```
BUInt BMeasureApi::CommsUsb::onum  [private]
```

7.17.3.6 oterminated

```
Bool BMeasureApi::CommsUsb::oterminated [private]
```

7.17.3.7 oterminating

```
Bool BMeasureApi::CommsUsb::oterminating [private]
```

7.17.3.8 ousbDisconnected

```
Bool BMeasureApi::CommsUsb::ousbDisconnected [private]
```

The documentation for this class was generated from the following files:

- CommsUsb.h
- CommsUsb.cpp

7.18 BMeasureApi::ConfigItem Class Reference

```
#include <BMeasureD.h>
```

Static Public Member Functions

- static const **BObjMember** ∗ getMembers ()

Public Attributes

- **BChar** name [16]
- **BUInt8** type

  The type of data.
- **BUInt8** spare [7]
- **BChar** value [16]

7.18.1 Member Function Documentation

7.18.1.1 getMembers()

```
const BObjMember * BMeasureApi::ConfigItem::getMembers ( ) [static]
```

7.18.2 Member Data Documentation

7.18.2.1 name

```
BChar BMeasureApi::ConfigItem::name[16]
```

7.18.2.2 spare

```
BUInt8 BMeasureApi::ConfigItem::spare[7]
```

7.18.2.3 type

```
BUInt8 BMeasureApi::ConfigItem::type
```

The type of data.

7.18.2.4 value

```
BChar BMeasureApi::ConfigItem::value[16]
```

The documentation for this class was generated from the following files:

- BMeasureD.h
- BMeasureD.cpp

7.19 BMeasureApi::Configuration Class Reference

```
#include <BMeasureD.h>
```

Static Public Member Functions

- static const **BObjMember** $*$ getMembers ()

Public Attributes

- **BUInt32** version

  The configuration version.
- **BChar** name [16]
- **BChar** location [16]
- Mode mode

  The boot run mode.
- SecurityMode securityMode

  The security mode.
- **BUInt8** logData

  Log the data.
- **BUInt8** logDataMode

  Log data mode.
- **BUInt8** logDataDevice

  The device to store data.
- **BUInt8** source

  The source number if multiple units are in use.
- **BUInt8** sampleFrequencyMode

  The base sample frequency mode.
- DigitalMode digitalMode

  The digital mode.
- **BUInt8** digitalPins [8]
- NetworkMode networkMode

  The network mode (0 - off, 1 - dhcp, 2 - manual)
- **BUInt8** spare1 [3]
- **BUInt32** networkAddress

  The network IP address.
- **BUInt32** networkMask

  The network netmask.
- **BUInt32** networkGateway

  The network gateway.
- **BUInt32** networkNameServer0

  The network nameserver.
- **BUInt32** networkTimeServer

  The network timeserver.
- Rs485Mode rs485Mode

  The RS485 mode.
- **BUInt8** rs485Bits

  The RS485 number of bits.
- **BUInt8** rs485StopBits

  The RS485 stop bits.
- **BUInt8** spare2
- **BUInt32** rs485BaudRate

  The RS485 baud rate.
- WifiMode wifiMode

  The wifi mode.
- **BUInt8** spare3 [3]
- **BChar** wifiAp0 [32]
- **BChar** spare4 [32]
- AlarmConfig alarms [16]
- EventMode mqttMode

MQTT mode.

- **BUInt8** spare5 [3]
- **BChar** mqttServer [32]
- **BUInt32** mqttPort

  The MQTT port.
- EventMode emailMode

  Email mode.
- **BUInt8** spare6 [3]
- **BChar** emailAddress [32]
- **BChar** program [32]

7.19.1 Member Function Documentation

7.19.1.1 getMembers()

```
const  BObjMember * BMeasureApi::Configuration::getMembers ( )  [static]
```

7.19.2 Member Data Documentation

7.19.2.1 alarms

```
AlarmConfig BMeasureApi::Configuration::alarms[16]
```

7.19.2.2 digitalMode

```
DigitalMode BMeasureApi::Configuration::digitalMode
```

The digital mode.

7.19.2.3 digitalPins

```
 BUInt8 BMeasureApi::Configuration::digitalPins[8]
```

7.19.2.4 emailAddress

```
BChar BMeasureApi::Configuration::emailAddress[32]
```

7.19.2.5 emailMode

```
EventMode BMeasureApi::Configuration::emailMode
```

Email mode.

7.19.2.6 location

```
BChar BMeasureApi::Configuration::location[16]
```

7.19.2.7 logData

```
BUInt8 BMeasureApi::Configuration::logData
```

Log the data.

7.19.2.8 logDataDevice

```
BUInt8 BMeasureApi::Configuration::logDataDevice
```

The device to store data.

7.19.2.9 logDataMode

```
BUInt8 BMeasureApi::Configuration::logDataMode
```

Log data mode.

7.19.2.10 mode

```
Mode BMeasureApi::Configuration::mode
```

The boot run mode.

7.19.2.11 mqttMode

```
EventMode BMeasureApi::Configuration::mqttMode
```

MQTT mode.

7.19.2.12 mqttPort

```
BUInt32 BMeasureApi::Configuration::mqttPort
```

The MQTT port.

7.19.2.13 mqttServer

```
BChar BMeasureApi::Configuration::mqttServer[32]
```

7.19.2.14 name

```
BChar BMeasureApi::Configuration::name[16]
```

7.19.2.15 networkAddress

```
BUInt32 BMeasureApi::Configuration::networkAddress
```

The network IP address.

7.19.2.16 networkGateway

```
BUInt32 BMeasureApi::Configuration::networkGateway
```

The network gateway.

7.19.2.17 networkMask

```
BUInt32 BMeasureApi::Configuration::networkMask
```

The network netmask.

7.19.2.18 networkMode

```
NetworkMode BMeasureApi::Configuration::networkMode
```

The network mode (0 - off, 1 - dhcp, 2 - manual)

7.19.2.19 networkNameServer0

```
BUInt32 BMeasureApi::Configuration::networkNameServer0
```

The network nameserver.

7.19.2.20 networkTimeServer

```
BUInt32 BMeasureApi::Configuration::networkTimeServer
```

The network timeserver.

7.19.2.21 program

```
BChar BMeasureApi::Configuration::program[32]
```

7.19.2.22 rs485BaudRate

BUInt32 BMeasureApi::Configuration::rs485BaudRate

The RS485 baud rate.

7.19.2.23 rs485Bits

BUInt8 BMeasureApi::Configuration::rs485Bits

The RS485 number of bits.

7.19.2.24 rs485Mode

Rs485Mode BMeasureApi::Configuration::rs485Mode

The RS485 mode.

7.19.2.25 rs485StopBits

BUInt8 BMeasureApi::Configuration::rs485StopBits

The RS485 stop bits.

7.19.2.26 sampleFrequencyMode

BUInt8 BMeasureApi::Configuration::sampleFrequencyMode

The base sample frequency mode.

7.19.2.27 securityMode

SecurityMode BMeasureApi::Configuration::securityMode

The security mode.

7.19.2.28 source

```
BUInt8 BMeasureApi::Configuration::source
```

The source number if multiple units are in use.

7.19.2.29 spare1

```
BUInt8 BMeasureApi::Configuration::spare1[3]
```

7.19.2.30 spare2

```
BUInt8 BMeasureApi::Configuration::spare2
```

7.19.2.31 spare3

```
BUInt8 BMeasureApi::Configuration::spare3[3]
```

7.19.2.32 spare4

```
BChar BMeasureApi::Configuration::spare4[32]
```

7.19.2.33 spare5

```
BUInt8 BMeasureApi::Configuration::spare5[3]
```

7.19.2.34 spare6

```
BUInt8 BMeasureApi::Configuration::spare6[3]
```

7.19.2.35 version

```
BUInt32 BMeasureApi::Configuration::version
```

The configuration version.

7.19.2.36 wifiAp0

```
BChar BMeasureApi::Configuration::wifiAp0[32]
```

7.19.2.37 wifiMode

```
WifiMode BMeasureApi::Configuration::wifiMode
```

The wifi mode.

The documentation for this class was generated from the following files:

- BMeasureD.h
- BMeasureD.cpp

7.20 BMeasureApi::DataBlock Class Reference

Data Block. Data in floating point format for all channels.

```
#include <BMeasureD.h>
```

Static Public Member Functions

- static const **BObjMember** $*$ getMembers ()

Public Attributes

- **BUInt64** time

  The time in microseconds since 1970-01-01 to TAI.
- **BUInt16** source

  The source unit,.
- **BUInt16** status
- **BUInt16** numChannels

  The number of data channels.
- **BUInt16** numSamples

  The number of samples.
- **BUInt32** sequence

  The sequence number.
- DataType type

  The type of data block.
- **BUInt8** spare [3]
- **BFloat32** data [118]

7.20.1 Detailed Description

Data Block. Data in floating point format for all channels.

7.20.2 Member Function Documentation

7.20.2.1 getMembers()

```
const  BObjMember * BMeasureApi::DataBlock::getMembers ( )  [static]
```

7.20.3 Member Data Documentation

7.20.3.1 data

```
BFloat32 BMeasureApi::DataBlock::data[118]
```

7.20.3.2 numChannels

```
BUInt16 BMeasureApi::DataBlock::numChannels
```

The number of data channels.

7.20.3.3 numSamples

```
BUInt16 BMeasureApi::DataBlock::numSamples
```

The number of samples.

7.20.3.4 sequence

```
BUInt32 BMeasureApi::DataBlock::sequence
```

The sequence number.

7.20.3.5 source

```
BUInt16 BMeasureApi::DataBlock::source
```

The source unit,.

7.20.3.6 spare

```
BUInt8 BMeasureApi::DataBlock::spare[3]
```

7.20.3.7 status

```
BUInt16 BMeasureApi::DataBlock::status
```

7.20.3.8 time

```
BUInt64 BMeasureApi::DataBlock::time
```

The time in microseconds since 1970-01-01 to TAI.

7.20.3.9 type

```
DataType BMeasureApi::DataBlock::type
```

The type of data block.

The documentation for this class was generated from the following files:

- BMeasureD.h
- BMeasureD.cpp

7.21 BMeasureApi::DataBlockFloat Class Reference

Processed data for a channel.

```
#include <BMeasureD.h>
```

Static Public Member Functions

- static const **BObjMember** ∗ getMembers ()

Public Attributes

- **BUInt64** time

  The time in microseconds since 1970-01-01 to TAI.
- **BUInt16** source

  The source unit,.
- **BUInt16** status
- **BUInt16** numChannels

  The number of data channels.
- **BUInt16** numSamples

  The number of samples.
- **BUInt32** sequence

  The sequence number.
- DataType type

  The type of data block.
- **BUInt8** spare [3]
- **BFloat32** data [208]

7.21.1 Detailed Description

Processed data for a channel.

7.21.2 Member Function Documentation

7.21.2.1 getMembers()

```
const  BObjMember ∗ BMeasureApi::DataBlockFloat::getMembers ( )  [static]
```

7.21.3 Member Data Documentation

7.21.3.1 data

```
BFloat32 BMeasureApi::DataBlockFloat::data[208]
```

7.21.3.2 numChannels

```
BUInt16 BMeasureApi::DataBlockFloat::numChannels
```

The number of data channels.

7.21.3.3 numSamples

```
BUInt16 BMeasureApi::DataBlockFloat::numSamples
```

The number of samples.

7.21.3.4 sequence

```
BUInt32 BMeasureApi::DataBlockFloat::sequence
```

The sequence number.

7.21.3.5 source

```
BUInt16 BMeasureApi::DataBlockFloat::source
```

The source unit,.

7.21.3.6 spare

```
BUInt8 BMeasureApi::DataBlockFloat::spare[3]
```

7.21.3.7 status

```
BUInt16 BMeasureApi::DataBlockFloat::status
```

7.21.3.8 time

```
BUInt64 BMeasureApi::DataBlockFloat::time
```

The time in microseconds since 1970-01-01 to TAI.

7.21.3.9 type

```
DataType BMeasureApi::DataBlockFloat::type
```

The type of data block.

The documentation for this class was generated from the following files:

- BMeasureD.h
- BMeasureD.cpp

7.22 BMeasureApi::DataBlockProc Class Reference

Info Block.

```
#include <BMeasureD.h>
```

Static Public Member Functions

- static const **BObjMember** ∗ getMembers ()

Public Attributes

- **BUInt64** time

  The time in microseconds since 1970-01-01 to TAI.
- **BUInt16** source

  The source unit,.
- **BUInt16** status
- **BUInt16** numChannels

  The number of data channels.
- **BUInt16** numSamples

  The number of samples.
- **BUInt32** sequence

  The sequence number.
- DataType type

  The type of data block.
- **BUInt8** spare [3]
- DataProc analogueData [8]
- **BUInt32** digitalData

  Digital channel data.
- **BFloat32** period

  Time over which samples were processsed.

7.22.1 Detailed Description

Info Block.

7.22.2 Member Function Documentation

7.22.2.1 getMembers()

```
const BObjMember * BMeasureApi::DataBlockProc::getMembers ( ) [static]
```

7.22.3 Member Data Documentation

7.22.3.1 analogueData

```
DataProc BMeasureApi::DataBlockProc::analogueData[8]
```

7.22.3.2 digitalData

```
BUInt32 BMeasureApi::DataBlockProc::digitalData
```

Digital channel data.

7.22.3.3 numChannels

```
BUInt16 BMeasureApi::DataBlockProc::numChannels
```

The number of data channels.

7.22.3.4 numSamples

```
BUInt16 BMeasureApi::DataBlockProc::numSamples
```

The number of samples.

7.22.3.5 period

```
BFloat32 BMeasureApi::DataBlockProc::period
```

Time over which samples were processsed.

7.22.3.6 sequence

```
BUInt32 BMeasureApi::DataBlockProc::sequence
```

The sequence number.

7.22.3.7 source

```
BUInt16 BMeasureApi::DataBlockProc::source
```

The source unit,.

7.22.3.8 spare

```
BUInt8 BMeasureApi::DataBlockProc::spare[3]
```

7.22.3.9 status

```
BUInt16 BMeasureApi::DataBlockProc::status
```

7.22.3.10 time

```
BUInt64 BMeasureApi::DataBlockProc::time
```

The time in microseconds since 1970-01-01 to TAI.

7.22.3.11 type

```
DataType BMeasureApi::DataBlockProc::type
```

The type of data block.

The documentation for this class was generated from the following files:

- BMeasureD.h
- BMeasureD.cpp

7.23 BMeasureApi::DataFile Class Reference

```
#include <DataFile.h>
```

Public Member Functions

- DataFile ()
- ∼DataFile ()
- void init ()

  Initialise.
- **BError** open (**BString** fileName, **BString** mode, **BString** format="")

  Open the file for read or write.
- **BError** close ()

  Close the file.
- **BString** getFileName ()

  Return the file name.
- **BError** writeInfo (const InfoBlock &infoBlock, const ChannelConfigs &channels)
- **BError** writeData (DataBlock ∗ **data**)

  Write a block of data.
- **BError** writeData (DataBlockFloat ∗ **data**)

  Write a block of processed data.
- **BError** writeData (DataBlockProc ∗ **data**)

  Write a block of processed data.
- **BError** writeEnd ()
- **BError** readInfo (**BString** &format, InfoBlock &infoBlock, ChannelConfigs &channels)
- **BError** readData (DataBlock ∗ **data**)

  Read a block of data.

Private Member Functions

- **BError** validateFormat (**BString** format)
- **BError** writeInfoCsv (const InfoBlock &infoBlock, const ChannelConfigs &channels)
- **BError** writeInfoTdms (const InfoBlock &infoBlock, const ChannelConfigs &channels)
- **BError** writeInfoBMeas (const InfoBlock &infoBlock, const ChannelConfigs &channels)

Private Attributes

- **BString** ofileName
- **BString** omode
- **BString** oformat
- **BUInt** ofileType
- **BFile** ofile
- **BUInt32** opacketLen
- **BoapMc1Packet** ∗ opacket

7.23.1 Constructor & Destructor Documentation

7.23.1.1 DataFile()

```
BMeasureApi::DataFile::DataFile ( )
```

7.23.1.2 ∼DataFile()

```
BMeasureApi::DataFile::∼DataFile ( )
```

7.23.2 Member Function Documentation

7.23.2.1 close()

```
BError BMeasureApi::DataFile::close ( )
```

Close the file.

7.23.2.2 getFileName()

```
BString BMeasureApi::DataFile::getFileName ( )
```

Return the file name.

7.23.2.3 init()

```
void BMeasureApi::DataFile::init ( )
```

Initialise.

7.23.2.4 open()

```
BError BMeasureApi::DataFile::open (
             BString fileName,
             BString mode,
             BString format = "" )
```

Open the file for read or write.

7.23.2.5 readData()

```
BError BMeasureApi::DataFile::readData (
            DataBlock * data )
```

Read a block of data.

7.23.2.6 readInfo()

```
BError BMeasureApi::DataFile::readInfo (
             BString & format,
            InfoBlock & infoBlock,
            ChannelConfigs & channels )
```

7.23.2.7 validateFormat()

```
BError BMeasureApi::DataFile::validateFormat (
             BString format )  [private]
```

7.23.2.8 writeData() [1/3]

```
BError BMeasureApi::DataFile::writeData (
            DataBlock * data )
```

Write a block of data.

7.23.2.9 writeData() [2/3]

```
BError BMeasureApi::DataFile::writeData (
            DataBlockFloat * data )
```

Write a block of processed data.

7.23.2.10 writeData() [3/3]

```
BError BMeasureApi::DataFile::writeData (
            DataBlockProc * data )
```

Write a block of processed data.

7.23.2.11 writeEnd()

```
BError BMeasureApi::DataFile::writeEnd ( )
```

7.23.2.12 writeInfo()

```
BError BMeasureApi::DataFile::writeInfo (
            const InfoBlock & infoBlock,
            const ChannelConfigs & channels )
```

7.23.2.13 writeInfoBMeas()

```
BError BMeasureApi::DataFile::writeInfoBMeas (
            const InfoBlock & infoBlock,
            const ChannelConfigs & channels )  [private]
```

7.23.2.14 writeInfoCsv()

```
BError BMeasureApi::DataFile::writeInfoCsv (
            const InfoBlock & infoBlock,
            const ChannelConfigs & channels )  [private]
```

7.23.2.15 writeInfoTdms()

```
BError BMeasureApi::DataFile::writeInfoTdms (
            const InfoBlock & infoBlock,
            const ChannelConfigs & channels )  [private]
```

7.23.3 Member Data Documentation

7.23.3.1 ofile

```
BFile BMeasureApi::DataFile::ofile  [private]
```

7.23.3.2 ofileName

```
BString BMeasureApi::DataFile::ofileName  [private]
```

7.23.3.3 ofileType

```
BUInt BMeasureApi::DataFile::ofileType  [private]
```

7.23.3.4 oformat

```
BString BMeasureApi::DataFile::oformat  [private]
```

7.23.3.5 omode

```
BString BMeasureApi::DataFile::omode  [private]
```

7.23.3.6 opacket

```
BoapMc1Packet* BMeasureApi::DataFile::opacket  [private]
```

7.23.3.7 opacketLen

```
BUInt32 BMeasureApi::DataFile::opacketLen  [private]
```

The documentation for this class was generated from the following files:

- DataFile.h
- DataFile.cpp

7.24 BMeasureApi::DataProc Class Reference

Data Proc Block. Processd Data packed into bytestream based on sampleTypes.

```
#include <BMeasureD.h>
```

Static Public Member Functions

- static const **BObjMember** ∗ getMembers ()

Public Attributes

- **BFloat32** mean

  Processed data mean.
- **BFloat32** rms

  Processed data RMS.
- **BFloat32** peakHigh

  Processed data highest peak.
- **BFloat32** peakLow

  Processed data lowest peak.
- **BFloat32** power

  Processed data power.
- **BFloat32** spare1 [6]
- **Bool** alarm

  An alarm condition on this channel.
- **BUInt8** spare2 [3]

7.24.1 Detailed Description

Data Proc Block. Processd Data packed into bytestream based on sampleTypes.

7.24.2 Member Function Documentation

7.24.2.1 getMembers()

```
const BObjMember * BMeasureApi::DataProc::getMembers ( ) [static]
```

7.24.3 Member Data Documentation

7.24.3.1 alarm

```
Bool BMeasureApi::DataProc::alarm
```

An alarm condition on this channel.

7.24.3.2 mean

```
BFloat32 BMeasureApi::DataProc::mean
```

Processed data mean.

7.24.3.3 peakHigh

```
BFloat32 BMeasureApi::DataProc::peakHigh
```

Processed data highest peak.

7.24.3.4 peakLow

```
BFloat32 BMeasureApi::DataProc::peakLow
```

Processed data lowest peak.

7.24.3.5 power

```
BFloat32 BMeasureApi::DataProc::power
```

Processed data power.

7.24.3.6 rms

```
BFloat32 BMeasureApi::DataProc::rms
```

Processed data RMS.

7.24.3.7 spare1

```
BFloat32 BMeasureApi::DataProc::spare1[6]
```

7.24.3.8 spare2

```
BUInt8 BMeasureApi::DataProc::spare2[3]
```

The documentation for this class was generated from the following files:

- BMeasureD.h
- BMeasureD.cpp

7.25 Dfu Class Reference

The Dfu access class.

```
#include <Dfu.h>
```

Public Member Functions

- Dfu ()
- ~Dfu ()
- **BError** init (**Bool** verbose)

  Initialise.
- **BError** detectDevice ()

  Check if DFU devuce exists.
- **BError** validateFile (**BString** filename, **BUInt** type, **BString** &version)

  Check if file is valid firmware.
- **BError** connect ()

  Connect to USB DFU device.
- **BError** disconnect ()

  Disconnect from USB DFU device.
- **BError** reset ()

  Reset.
- **BError** clearStatus ()
- **BError** getStatus (DfuStatus &status)
- **BError** upload (**BString** filename, **BUInt** type)

  Upload a file.
- **BError** upload_cmd (**BUInt8 cmd**, **BUInt32** address)

Private Attributes

- **Bool** overbose
- **Bool** oconnected
- libusb_context ∗ ocontext
- libusb_device_handle ∗ odev

7.25.1 Detailed Description

The Dfu access class.

7.25.2 Constructor & Destructor Documentation

7.25.2.1 Dfu()

```
Dfu::Dfu ( )
```

7.25.2.2 ∼Dfu()

```
Dfu::∼Dfu ( )
```

7.25.3 Member Function Documentation

7.25.3.1 clearStatus()

```
BError Dfu::clearStatus ( )
```

7.25.3.2 connect()

```
BError Dfu::connect ( )
```

Connect to USB DFU device.

7.25.3.3 detectDevice()

```
BError Dfu::detectDevice ( )
```

Check if DFU devuce exists.

7.25.3.4 disconnect()

```
BError Dfu::disconnect ( )
```

Disconnect from USB DFU device.

7.25.3.5 getStatus()

```
BError Dfu::getStatus (
            DfuStatus & status )
```

7.25.3.6 init()

```
BError Dfu::init (
             Bool verbose )
```

Initialise.

7.25.3.7 reset()

```
BError Dfu::reset ( )
```

Reset.

7.25.3.8 upload()

```
BError Dfu::upload (
             BString filename,
             BUInt type )
```

Upload a file.

7.25.3.9 upload_cmd()

```
BError Dfu::upload_cmd (
             BUInt8 cmd,
             BUInt32 address )
```

7.25.3.10 validateFile()

```
BError Dfu::validateFile (
             BString filename,
             BUInt type,
             BString & version )
```

Check if file is valid firmware.

7.25.4 Member Data Documentation

7.25.4.1 oconnected

```
Bool Dfu::oconnected [private]
```

7.25.4.2 ocontext

```
libusb_context* Dfu::ocontext [private]
```

7.25.4.3 odev

```
libusb_device_handle* Dfu::odev [private]
```

7.25.4.4 overbose

```
Bool Dfu::overbose [private]
```

The documentation for this class was generated from the following files:

- Dfu.h
- Dfu.cpp

7.26 DfuStatus Struct Reference

```
#include <Dfu.h>
```

Public Attributes

- **BUInt8** status
- **BUInt** pollTimeout
- **BUInt8** state
- **BUInt8** iString

7.26.1 Member Data Documentation

7.26.1.1 iString

```
BUInt8 DfuStatus::iString
```

7.26.1.2 pollTimeout

```
BUInt DfuStatus::pollTimeout
```

7.26.1.3 state

```
BUInt8 DfuStatus::state
```

7.26.1.4 status

```
BUInt8 DfuStatus::status
```

The documentation for this struct was generated from the following file:

- Dfu.h

7.27 BMeasureApi::FileData Class Reference

```
#include <BMeasureD.h>
```

Static Public Member Functions

- static const **BObjMember** ∗ getMembers ()

Public Attributes

- **BUInt32** length

  The data length.
- **BUInt8** data [512]

7.27.1 Member Function Documentation

7.27.1.1 getMembers()

```
const BObjMember * BMeasureApi::FileData::getMembers ( ) [static]
```

7.27.2 Member Data Documentation

7.27.2.1 data

```
BUInt8 BMeasureApi::FileData::data[512]
```

7.27.2.2 length

```
BUInt32 BMeasureApi::FileData::length
```

The data length.

The documentation for this class was generated from the following files:

- BMeasureD.h
- BMeasureD.cpp

7.28 BMeasureApi::FileInfo Class Reference

File information.

```
#include <BMeasureD.h>
```

Static Public Member Functions

- static const **BObjMember** ∗ getMembers ()

Public Attributes

- **BChar** name [128]
- **BTime** time

  The file date/time.
- FileType fileType

  The file type.
- **BUInt8** spare [3]
- **BUInt64** fileLength

  The file length.

7.28.1 Detailed Description

File information.

7.28.2 Member Function Documentation

7.28.2.1 getMembers()

```
const BObjMember * BMeasureApi::FileInfo::getMembers ( ) [static]
```

7.28.3 Member Data Documentation

7.28.3.1 fileLength

```
BUInt64 BMeasureApi::FileInfo::fileLength
```

The file length.

7.28.3.2 fileType

```
FileType BMeasureApi::FileInfo::fileType
```

The file type.

7.28.3.3 name

```
BChar BMeasureApi::FileInfo::name[128]
```

7.28.3.4 spare

```
BUInt8 BMeasureApi::FileInfo::spare[3]
```

7.28.3.5 time

```
BTime BMeasureApi::FileInfo::time
```

The file date/time.

The documentation for this class was generated from the following files:

- BMeasureD.h
- BMeasureD.cpp

7.29 BMeasureApi::FilesysInfo Class Reference

```
#include <BMeasureD.h>
```

Static Public Member Functions

- static const **BObjMember** * getMembers ()

Public Attributes

- **BChar** name [128]
- **BUInt64** size

  The store size.
- **BUInt64** free

  The store free space.

7.29.1 Member Function Documentation

7.29.1.1 getMembers()

```
const  BObjMember * BMeasureApi::FilesysInfo::getMembers ( )  [static]
```

7.29.2 Member Data Documentation

7.29.2.1 free

```
BUInt64 BMeasureApi::FilesysInfo::free
```

The store free space.

7.29.2.2 name

```
BChar BMeasureApi::FilesysInfo::name[128]
```

7.29.2.3 size

```
BUInt64 BMeasureApi::FilesysInfo::size
```

The store size.

The documentation for this class was generated from the following files:

- BMeasureD.h
- BMeasureD.cpp

7.30 BMeasureApi::InfoBlock Class Reference

AWG Configuration.

```
#include <BMeasureD.h>
```

Static Public Member Functions

- static const **BObjMember** ∗ getMembers ()

Public Attributes

- **BUInt64** time

  The time in microseconds since 1970-01-01 to TAI.
- **BUInt16** source

  The source unit.
- **BUInt16** numChannels

  The number of data channels.
- **BUInt16** version

  The info/data version.
- BMeasFileType fileType

  The file structure type.
- DataType dataType

  The data type file contents.
- **BChar** name [16]
- **BChar** location [16]
- NodeInfo nodeInfo

  Information on the unit.
- MeasurementConfig measureConfig

  The measurement configuration.

7.30.1 Detailed Description

AWG Configuration.

7.30.2 Member Function Documentation

7.30.2.1 getMembers()

```
const  BObjMember * BMeasureApi::InfoBlock::getMembers ( )  [static]
```

7.30.3 Member Data Documentation

7.30.3.1 dataType

```
DataType BMeasureApi::InfoBlock::dataType
```

The data type file contents.

7.30.3.2 fileType

```
BMeasFileType BMeasureApi::InfoBlock::fileType
```

The file structure type.

7.30.3.3 location

```
BChar BMeasureApi::InfoBlock::location[16]
```

7.30.3.4 measureConfig

```
MeasurementConfig BMeasureApi::InfoBlock::measureConfig
```

The measurement configuration.

7.30.3.5 name

```
BChar BMeasureApi::InfoBlock::name[16]
```

7.30.3.6 nodeInfo

```
NodeInfo BMeasureApi::InfoBlock::nodeInfo
```

Information on the unit.

7.30.3.7 numChannels

```
BUInt16 BMeasureApi::InfoBlock::numChannels
```

The number of data channels.

7.30.3.8 source

`BUInt16 BMeasureApi::InfoBlock::source`

The source unit.

7.30.3.9 time

`BUInt64 BMeasureApi::InfoBlock::time`

The time in microseconds since 1970-01-01 to TAI.

7.30.3.10 version

`BUInt16 BMeasureApi::InfoBlock::version`

The info/data version.

The documentation for this class was generated from the following files:

- BMeasureD.h
- BMeasureD.cpp

7.31 BMeasureApi::Information Class Reference

`#include <BMeasureD.h>`

Static Public Member Functions

- static const **BObjMember** ∗ getMembers ()

Public Attributes

- NodeInfo nodeInfo
- **BUInt8** numConfigItems

  The number of config items.
- **BUInt8** numChannels

  The number of channels.
- **BUInt8** spare0 [2]
- **BUInt32** spare1
- **BTimeUs** time

  The system time.
- **BUInt32** networkMode

  The network Mode.
- **BUInt8** networkMacAddress [6]
- **BUInt8** spare2 [2]
- **BUInt32** networkAddress

  The network IP address.
- **BUInt32** networkMask

  The network netmask.
- **BUInt32** networkGateway

  The network gateway.
- **BUInt32** networkNameServer0

  The network nameserver.
- **BUInt32** networkTimeServer

  The network time server.
- **BUInt32** wifiMode

  The Wifi mode.
- **BUInt8** wifiMacAddress [6]
- **BUInt8** spare3 [2]
- **BUInt32** wifiAddress

  The Wifi IP address.
- **BUInt32** wifiMask

  The Wifi netmask.
- **BUInt32** wifiGateway

  The Wifi gateway.
- **BTime** calibTime

  The last calibration time.
- **BUInt8** spare4 [28]

7.31.1 Member Function Documentation

7.31.1.1 getMembers()

```
const BObjMember * BMeasureApi::Information::getMembers ( )  [static]
```

7.31.2 Member Data Documentation

7.31.2.1 calibTime

BTime `BMeasureApi::Information::calibTime`

The last calibration time.

7.31.2.2 networkAddress

BUInt32 `BMeasureApi::Information::networkAddress`

The network IP address.

7.31.2.3 networkGateway

BUInt32 `BMeasureApi::Information::networkGateway`

The network gateway.

7.31.2.4 networkMacAddress

BUInt8 `BMeasureApi::Information::networkMacAddress[6]`

7.31.2.5 networkMask

BUInt32 `BMeasureApi::Information::networkMask`

The network netmask.

7.31.2.6 networkMode

BUInt32 `BMeasureApi::Information::networkMode`

The network Mode.

7.31.2.7 networkNameServer0

```
BUInt32 BMeasureApi::Information::networkNameServer0
```

The network nameserver.

7.31.2.8 networkTimeServer

```
BUInt32 BMeasureApi::Information::networkTimeServer
```

The network time server.

7.31.2.9 nodeInfo

```
NodeInfo BMeasureApi::Information::nodeInfo
```

7.31.2.10 numChannels

```
BUInt8 BMeasureApi::Information::numChannels
```

The number of channels.

7.31.2.11 numConfigItems

```
BUInt8 BMeasureApi::Information::numConfigItems
```

The number of config items.

7.31.2.12 spare0

```
BUInt8 BMeasureApi::Information::spare0[2]
```

7.31.2.13 spare1

```
BUInt32 BMeasureApi::Information::spare1
```

7.31.2.14 spare2

```
BUInt8 BMeasureApi::Information::spare2[2]
```

7.31.2.15 spare3

```
BUInt8 BMeasureApi::Information::spare3[2]
```

7.31.2.16 spare4

```
BUInt8 BMeasureApi::Information::spare4[28]
```

7.31.2.17 time

```
BTimeUs BMeasureApi::Information::time
```

The system time.

7.31.2.18 wifiAddress

```
BUInt32 BMeasureApi::Information::wifiAddress
```

The Wifi IP address.

7.31.2.19 wifiGateway

```
BUInt32 BMeasureApi::Information::wifiGateway
```

The Wifi gateway.

7.31.2.20 wifiMacAddress

```
BUInt8 BMeasureApi::Information::wifiMacAddress[6]
```

7.31.2.21 wifiMask

```
BUInt32 BMeasureApi::Information::wifiMask
```

The Wifi netmask.

7.31.2.22 wifiMode

```
BUInt32 BMeasureApi::Information::wifiMode
```

The Wifi mode.

The documentation for this class was generated from the following files:

- BMeasureD.h
- BMeasureD.cpp

7.32 BMeasureApi::MeasurementConfig Class Reference

Measurement config Data Block. Data packed into bytestream based on sampleTypes.

```
#include <BMeasureD.h>
```

Static Public Member Functions

- static const **BObjMember** ∗ getMembers ()

Public Attributes

- MeasureMode measureMode
- **BUInt8** measureOptions

  Measure option bit set.
- **BUInt8** peakFilter

  Peak filtering number of samples.
- **BUInt8** spare1
- TriggerMode triggerMode
- TriggerConfig triggerConfig

  Trigger config including direction, filters etc.
- **BUInt8** triggerChannel
- **BUInt8** spare2
- **BFloat32** triggerLevel
- **BInt32** triggerDelay

  Trigger delay in samples.
- **BFloat64** sampleRate
- **BUInt32** numSamples0

  The number of samples in a chunk for display and/or repeat.
- **BUInt32** numSamples1

  The number of samples per each data processing cycle. 0 disables this processing.
- **BUInt32** numSamples2

  The number of samples to capture. 0 is continuous.
- **BUInt32** measurePeriod

  Time in seconds between measurement sample bursts. 0 is continuous.
- **BUInt32** numSamplesBlock

  The number of samples per block. 0 is default setting.
- **BUInt32** spare3
- **BChar** description [64]

7.32.1 Detailed Description

Measurement config Data Block. Data packed into bytestream based on sampleTypes.

7.32.2 Member Function Documentation

7.32.2.1 getMembers()

```
const BObjMember * BMeasureApi::MeasurementConfig::getMembers ( ) [static]
```

7.32.3 Member Data Documentation

7.32.3.1 description

```
BChar BMeasureApi::MeasurementConfig::description[64]
```

7.32.3.2 measureMode

```
MeasureMode BMeasureApi::MeasurementConfig::measureMode
```

7.32.3.3 measureOptions

```
BUInt8 BMeasureApi::MeasurementConfig::measureOptions
```

Measure option bit set.

7.32.3.4 measurePeriod

```
BUInt32 BMeasureApi::MeasurementConfig::measurePeriod
```

Time in seconds between measurement sample bursts. 0 is continuous.

7.32.3.5 numSamples0

```
BUInt32 BMeasureApi::MeasurementConfig::numSamples0
```

The number of samples in a chunk for display and/or repeat.

7.32.3.6 numSamples1

```
BUInt32 BMeasureApi::MeasurementConfig::numSamples1
```

The number of samples per each data processing cycle. 0 disables this processing.

7.32.3.7 numSamples2

```
BUInt32 BMeasureApi::MeasurementConfig::numSamples2
```

The number of samples to capture. 0 is continuous.

7.32.3.8 numSamplesBlock

```
BUInt32 BMeasureApi::MeasurementConfig::numSamplesBlock
```

The number of samples per block. 0 is default setting.

7.32.3.9 peakFilter

```
BUInt8 BMeasureApi::MeasurementConfig::peakFilter
```

Peak filtering number of samples.

7.32.3.10 sampleRate

```
BFloat64 BMeasureApi::MeasurementConfig::sampleRate
```

7.32.3.11 spare1

```
BUInt8 BMeasureApi::MeasurementConfig::spare1
```

7.32.3.12 spare2

```
BUInt8 BMeasureApi::MeasurementConfig::spare2
```

7.32.3.13 spare3

```
BUInt32 BMeasureApi::MeasurementConfig::spare3
```

7.32.3.14 triggerChannel

```
BUInt8 BMeasureApi::MeasurementConfig::triggerChannel
```

7.32.3.15 triggerConfig

```
TriggerConfig BMeasureApi::MeasurementConfig::triggerConfig
```

Trigger config including direction, filters etc.

7.32.3.16 triggerDelay

```
BInt32 BMeasureApi::MeasurementConfig::triggerDelay
```

Trigger delay in samples.

7.32.3.17 triggerLevel

```
BFloat32 BMeasureApi::MeasurementConfig::triggerLevel
```

7.32.3.18 triggerMode

```
TriggerMode BMeasureApi::MeasurementConfig::triggerMode
```

The documentation for this class was generated from the following files:

- BMeasureD.h
- BMeasureD.cpp

7.33 BMeasureApi::NodeInfo Class Reference

```
#include <BMeasureD.h>
```

Static Public Member Functions

- static const **BObjMember** ∗ getMembers ()

Public Attributes

- **BUInt32** apiVersion
- Version hardwareVersion
- Version fpgaVersion
- Version wifiVersion
- Version softwareVersion
- **BChar** variant [12]
- **BChar** serialNumber [12]
- SecurityMode securityMode
- **BUInt8** spare1 [3]
- **BUInt32** apiSubVersion
- **BUInt32** spare2

7.33.1 Member Function Documentation

7.33.1.1 getMembers()

```
const BObjMember * BMeasureApi::NodeInfo::getMembers ( ) [static]
```

7.33.2 Member Data Documentation

7.33.2.1 apiSubVersion

```
BUInt32 BMeasureApi::NodeInfo::apiSubVersion
```

7.33.2.2 apiVersion

```
BUInt32 BMeasureApi::NodeInfo::apiVersion
```

7.33.2.3 fpgaVersion

```
Version BMeasureApi::NodeInfo::fpgaVersion
```

7.33.2.4 hardwareVersion

```
Version BMeasureApi::NodeInfo::hardwareVersion
```

7.33.2.5 securityMode

```
SecurityMode BMeasureApi::NodeInfo::securityMode
```

7.33.2.6 serialNumber

```
BChar BMeasureApi::NodeInfo::serialNumber[12]
```

7.33.2.7 softwareVersion

```
Version BMeasureApi::NodeInfo::softwareVersion
```

7.33.2.8 spare1

```
BUInt8 BMeasureApi::NodeInfo::spare1[3]
```

7.33.2.9 spare2

```
BUInt32 BMeasureApi::NodeInfo::spare2
```

7.33.2.10 variant

```
BChar BMeasureApi::NodeInfo::variant[12]
```

7.33.2.11 wifiVersion

```
Version BMeasureApi::NodeInfo::wifiVersion
```

The documentation for this class was generated from the following files:

- BMeasureD.h
- BMeasureD.cpp

7.34 BMeasureApi::NodeStatus Class Reference

```
#include <BMeasureD.h>
```

Static Public Member Functions

- static const **BObjMember** ∗ getMembers ()

Public Attributes

- **BTimeUs** time
- **BUInt32** status
- **BUInt32** error
- **BChar** errorStr [32]
- Mode mode
- **BUInt8** ethernetStatus
- **BUInt8** wifiStatus
- **BUInt8** spare [13]

7.34.1 Member Function Documentation

7.34.1.1 getMembers()

```
const  BObjMember * BMeasureApi::NodeStatus::getMembers ( )  [static]
```

7.34.2 Member Data Documentation

7.34.2.1 error

```
BUInt32 BMeasureApi::NodeStatus::error
```

7.34.2.2 errorStr

```
BChar BMeasureApi::NodeStatus::errorStr[32]
```

7.34.2.3 ethernetStatus

```
BUInt8 BMeasureApi::NodeStatus::ethernetStatus
```

7.34.2.4 mode

```
Mode BMeasureApi::NodeStatus::mode
```

7.34.2.5 spare

```
BUInt8 BMeasureApi::NodeStatus::spare[13]
```

7.34.2.6 status

```
BUInt32 BMeasureApi::NodeStatus::status
```

7.34.2.7 time

```
BTimeUs BMeasureApi::NodeStatus::time
```

7.34.2.8 wifiStatus

```
BUInt8 BMeasureApi::NodeStatus::wifiStatus
```

The documentation for this class was generated from the following files:

- BMeasureD.h
- BMeasureD.cpp

7.35 BMeasureApi::Version Class Reference

```
#include <BMeasureD.h>
```

Static Public Member Functions

- static const **BObjMember** ∗ getMembers ()

Public Attributes

- **BUInt8** type
- **BUInt8** ver0
- **BUInt8** ver1
- **BUInt8** ver2

7.35.1 Member Function Documentation

7.35.1.1 getMembers()

```
const BObjMember * BMeasureApi::Version::getMembers ( ) [static]
```

7.35.2 Member Data Documentation

7.35.2.1 type

```
BUInt8 BMeasureApi::Version::type
```

7.35.2.2 ver0

```
BUInt8 BMeasureApi::Version::ver0
```

7.35.2.3 ver1

```
BUInt8 BMeasureApi::Version::ver1
```

7.35.2.4 ver2

```
BUInt8 BMeasureApi::Version::ver2
```

The documentation for this class was generated from the following files:

- BMeasureD.h
- BMeasureD.cpp

7.36 BMeasureApi::WifiAccessPoint Class Reference

```
#include <BMeasureD.h>
```

Static Public Member Functions

- static const **BObjMember** ∗ getMembers ()

Public Attributes

- **BChar** name [36]
- **BInt32** signalLevel

  The signal level.
- **BUInt8** channel

  The Wifi channel.
- **BUInt8** auth

  The Wifi autentication.
- **BUInt8** spare [2]

7.36.1 Member Function Documentation

7.36.1.1 getMembers()

```
const BObjMember * BMeasureApi::WifiAccessPoint::getMembers ( ) [static]
```

7.36.2 Member Data Documentation

7.36.2.1 auth

```
BUInt8 BMeasureApi::WifiAccessPoint::auth
```

The Wifi autentication.

7.36.2.2 channel

```
BUInt8 BMeasureApi::WifiAccessPoint::channel
```

The Wifi channel.

7.36.2.3 name

```
BChar BMeasureApi::WifiAccessPoint::name[36]
```

7.36.2.4 signalLevel

```
BInt32 BMeasureApi::WifiAccessPoint::signalLevel
```

The signal level.

7.36.2.5 spare

```
BUInt8 BMeasureApi::WifiAccessPoint::spare[2]
```

The documentation for this class was generated from the following files:

- BMeasureD.h
- BMeasureD.cpp

Chapter 8

File Documentation

8.1 BMdns.cpp File Reference

```
#include <BMdns.h>
#include <BDebug.h>
#include <stdio.h>
#include <errno.h>
#include <sys/ioctl.h>
#include <sys/socket.h>
#include <arpa/inet.h>
#include <netdb.h>
#include <net/if.h>
```

Macros

- #define BDEBUGL1 0

Enumerations

- enum MdnsRecordType {
  MDNS_RECORDTYPE_IGNORE = 0 , MDNS_RECORDTYPE_A = 1 , MDNS_RECORDTYPE_PTR = 12 ,
  MDNS_RECORDTYPE_TXT = 16 ,
  MDNS_RECORDTYPE_AAAA = 28 , MDNS_RECORDTYPE_SRV = 33 }
- enum MdnsEntryType { MDNS_ENTRYTYPE_ANSWER = 1 , MDNS_ENTRYTYPE_AUTHORITY = 2 , MDNS_ENTRYTYPE_ADDITIONAL = 3 }
- enum MdnsClass { MDNS_CLASS_IN = 1 }

Functions

- static int mdns_write_string (**BUInt8** ∗buffer, **BUInt8** ∗p, **BString** str)
- static int mdns_read_string (void ∗buffer, **BUInt8** ∗p, **BString** &str)
- static int mdns_read_strings (void ∗buffer, **BUInt8** ∗p, **BString** &str)

8.1.1 Macro Definition Documentation

8.1.1.1 BDEBUGL1

```
#define BDEBUGL1 0
```

8.1.2 Enumeration Type Documentation

8.1.2.1 MdnsClass

```
enum MdnsClass
```

Enumerator

MDNS_CLASS_IN	

8.1.2.2 MdnsEntryType

```
enum MdnsEntryType
```

Enumerator

MDNS_ENTRYTYPE_ANSWER	
MDNS_ENTRYTYPE_AUTHORITY	
MDNS_ENTRYTYPE_ADDITIONAL	

8.1.2.3 MdnsRecordType

```
enum MdnsRecordType
```

Enumerator

MDNS_RECORDTYPE_IGNORE	
MDNS_RECORDTYPE_A	
MDNS_RECORDTYPE_PTR	
MDNS_RECORDTYPE_TXT	
MDNS_RECORDTYPE_AAAA	
MDNS_RECORDTYPE_SRV	

8.1.3 Function Documentation

8.1.3.1 mdns_read_string()

```
static int mdns_read_string (
             void * buffer,
              BUInt8 * p,
              BString & str )  [static]
```

8.1.3.2 mdns_read_strings()

```
static int mdns_read_strings (
             void * buffer,
              BUInt8 * p,
              BString & str )  [static]
```

8.1.3.3 mdns_write_string()

```
static int mdns_write_string (
              BUInt8 * buffer,
              BUInt8 * p,
              BString str )  [static]
```

8.2 BMdns.h File Reference

```
#include <BSocket.h>
```

Classes

- class BMdnsService
- class BMdns

8.3 BMeasureB.cpp File Reference

```
#include <BMeasureB.h>
#include <string.h>
```

Namespaces

- namespace BMeasureApi

8.4 BMeasureB.h File Reference

```
#include <BTypes.h>
#include <BComplex.h>
#include <BoapMc1.h>
#include <BMeasureD.h>
```

Classes

- class BMeasureApi::BMeasure

Namespaces

- namespace BMeasureApi

Variables

- const **BUInt32** BMeasureApi::apiVersion = 0

8.5 BMeasureD.cpp File Reference

```
#include <BMeasureD.h>
```

Namespaces

- namespace BMeasureApi

Macros

- #define boffsetof(T, F) ((**BUInt**)((char*)&((T*)0L)->F - (char*)0L))

Functions

- **BString** BMeasureApi::toBString (ErrorNum v)
- **BError** BMeasureApi::fromBString (**BString** str, ErrorNum &v)
- **BString** BMeasureApi::toBStringJson (**BString** n, ErrorNum v)
- **BString** BMeasureApi::toBString (NodeType v)
- **BError** BMeasureApi::fromBString (**BString** str, NodeType &v)
- **BString** BMeasureApi::toBStringJson (**BString** n, NodeType v)
- **BString** BMeasureApi::toBString (SecurityMode v)
- **BError** BMeasureApi::fromBString (**BString** str, SecurityMode &v)
- **BString** BMeasureApi::toBStringJson (**BString** n, SecurityMode v)
- **BString** BMeasureApi::toBString (Status v)
- **BError** BMeasureApi::fromBString (**BString** str, Status &v)
- **BString** BMeasureApi::toBStringJson (**BString** n, Status v)
- **BString** BMeasureApi::toBString (Mode v)
- **BError** BMeasureApi::fromBString (**BString** str, Mode &v)
- **BString** BMeasureApi::toBStringJson (**BString** n, Mode v)
- **BString** BMeasureApi::toBString (BlockTypes v)
- **BError** BMeasureApi::fromBString (**BString** str, BlockTypes &v)
- **BString** BMeasureApi::toBStringJson (**BString** n, BlockTypes v)
- **BString** BMeasureApi::toBString (ChannelType v)
- **BError** BMeasureApi::fromBString (**BString** str, ChannelType &v)
- **BString** BMeasureApi::toBStringJson (**BString** n, ChannelType v)
- **BString** BMeasureApi::toBString (SampleType v)
- **BError** BMeasureApi::fromBString (**BString** str, SampleType &v)
- **BString** BMeasureApi::toBStringJson (**BString** n, SampleType v)
- **BString** BMeasureApi::toBString (SyncMode v)
- **BError** BMeasureApi::fromBString (**BString** str, SyncMode &v)
- **BString** BMeasureApi::toBStringJson (**BString** n, SyncMode v)
- **BString** BMeasureApi::toBString (MeasureMode v)
- **BError** BMeasureApi::fromBString (**BString** str, MeasureMode &v)
- **BString** BMeasureApi::toBStringJson (**BString** n, MeasureMode v)
- **BString** BMeasureApi::toBString (MeasureOption v)
- **BError** BMeasureApi::fromBString (**BString** str, MeasureOption &v)
- **BString** BMeasureApi::toBStringJson (**BString** n, MeasureOption v)
- **BString** BMeasureApi::toBString (TriggerMode v)
- **BError** BMeasureApi::fromBString (**BString** str, TriggerMode &v)
- **BString** BMeasureApi::toBStringJson (**BString** n, TriggerMode v)
- **BString** BMeasureApi::toBString (TriggerConfig v)
- **BError** BMeasureApi::fromBString (**BString** str, TriggerConfig &v)
- **BString** BMeasureApi::toBStringJson (**BString** n, TriggerConfig v)
- **BString** BMeasureApi::toBString (DigitalMode v)
- **BError** BMeasureApi::fromBString (**BString** str, DigitalMode &v)
- **BString** BMeasureApi::toBStringJson (**BString** n, DigitalMode v)
- **BString** BMeasureApi::toBString (AwgMode v)
- **BError** BMeasureApi::fromBString (**BString** str, AwgMode &v)
- **BString** BMeasureApi::toBStringJson (**BString** n, AwgMode v)
- **BString** BMeasureApi::toBString (AwgOutput v)
- **BError** BMeasureApi::fromBString (**BString** str, AwgOutput &v)
- **BString** BMeasureApi::toBStringJson (**BString** n, AwgOutput v)
- **BString** BMeasureApi::toBString (FileType v)
- **BError** BMeasureApi::fromBString (**BString** str, FileType &v)
- **BString** BMeasureApi::toBStringJson (**BString** n, FileType v)
- **BString** BMeasureApi::toBString (FilesysDeleteType v)
- **BError** BMeasureApi::fromBString (**BString** str, FilesysDeleteType &v)

- **BString** BMeasureApi::toBStringJson (**BString** n, FilesysDeleteType v)
- **BString** BMeasureApi::toBString (LogData v)
- **BError** BMeasureApi::fromBString (**BString** str, LogData &v)
- **BString** BMeasureApi::toBStringJson (**BString** n, LogData v)
- **BString** BMeasureApi::toBString (LogDataMode v)
- **BError** BMeasureApi::fromBString (**BString** str, LogDataMode &v)
- **BString** BMeasureApi::toBStringJson (**BString** n, LogDataMode v)
- **BString** BMeasureApi::toBString (DataType v)
- **BError** BMeasureApi::fromBString (**BString** str, DataType &v)
- **BString** BMeasureApi::toBStringJson (**BString** n, DataType v)
- **BString** BMeasureApi::toBString (DataSend v)
- **BError** BMeasureApi::fromBString (**BString** str, DataSend &v)
- **BString** BMeasureApi::toBStringJson (**BString** n, DataSend v)
- **BString** BMeasureApi::toBString (CalibrateStage v)
- **BError** BMeasureApi::fromBString (**BString** str, CalibrateStage &v)
- **BString** BMeasureApi::toBStringJson (**BString** n, CalibrateStage v)
- **BString** BMeasureApi::toBString (MessageSource v)
- **BError** BMeasureApi::fromBString (**BString** str, MessageSource &v)
- **BString** BMeasureApi::toBStringJson (**BString** n, MessageSource v)
- **BString** BMeasureApi::toBString (NetworkMode v)
- **BError** BMeasureApi::fromBString (**BString** str, NetworkMode &v)
- **BString** BMeasureApi::toBStringJson (**BString** n, NetworkMode v)
- **BString** BMeasureApi::toBString (WifiMode v)
- **BError** BMeasureApi::fromBString (**BString** str, WifiMode &v)
- **BString** BMeasureApi::toBStringJson (**BString** n, WifiMode v)
- **BString** BMeasureApi::toBString (AlarmMode v)
- **BError** BMeasureApi::fromBString (**BString** str, AlarmMode &v)
- **BString** BMeasureApi::toBStringJson (**BString** n, AlarmMode v)
- **BString** BMeasureApi::toBString (AlarmOutput v)
- **BError** BMeasureApi::fromBString (**BString** str, AlarmOutput &v)
- **BString** BMeasureApi::toBStringJson (**BString** n, AlarmOutput v)
- **BString** BMeasureApi::toBString (EventMode v)
- **BError** BMeasureApi::fromBString (**BString** str, EventMode &v)
- **BString** BMeasureApi::toBStringJson (**BString** n, EventMode v)
- **BString** BMeasureApi::toBString (Rs485Mode v)
- **BError** BMeasureApi::fromBString (**BString** str, Rs485Mode &v)
- **BString** BMeasureApi::toBStringJson (**BString** n, Rs485Mode v)
- **BString** BMeasureApi::toBString (BMeasFileType v)
- **BError** BMeasureApi::fromBString (**BString** str, BMeasFileType &v)
- **BString** BMeasureApi::toBStringJson (**BString** n, BMeasFileType v)
- **BString** BMeasureApi::toBString (WifiCmd v)
- **BError** BMeasureApi::fromBString (**BString** str, WifiCmd &v)
- **BString** BMeasureApi::toBStringJson (**BString** n, WifiCmd v)
- **BString** BMeasureApi::toBString (WifiStatus v)
- **BError** BMeasureApi::fromBString (**BString** str, WifiStatus &v)
- **BString** BMeasureApi::toBStringJson (**BString** n, WifiStatus v)

8.5.1 Macro Definition Documentation

8.5.1.1 boffsetof

```
#define boffsetof(
            T,
            F ) (( BUInt)((char*)&((T*)0L)->F - (char*)0L))
```

8.6 BMeasureD.h File Reference

```
#include <BTypes.h>
#include <BObj.h>
#include <BTime.h>
#include <BTimeUs.h>
#include <BArray.h>
#include <BComplex.h>
#include <BoapMc.h>
```

Classes

- class BMeasureApi::Version
- class BMeasureApi::NodeInfo
- class BMeasureApi::NodeStatus
- class BMeasureApi::BoardConfig
- class BMeasureApi::ChannelConfig

  Channel configuration.
- class BMeasureApi::Information
- class BMeasureApi::AlarmConfig
- class BMeasureApi::Configuration
- class BMeasureApi::ConfigItem
- class BMeasureApi::MeasurementConfig

  Measurement config Data Block. Data packed into bytestream based on sampleTypes.
- class BMeasureApi::DataBlock

  Data Block. Data in floating point format for all channels.
- class BMeasureApi::DataBlockFloat

  Processed data for a channel.
- class BMeasureApi::DataProc

  Data Proc Block. Processd Data packed into bytestream based on sampleTypes.
- class BMeasureApi::DataBlockProc

  Info Block.
- class BMeasureApi::InfoBlock

  AWG Configuration.
- class BMeasureApi::AwgConfig

  File information.
- class BMeasureApi::FilesysInfo
- class BMeasureApi::FileInfo

  File information.
- class BMeasureApi::FileData
- class BMeasureApi::CalibrateInfo
- class BMeasureApi::WifiAccessPoint

Namespaces

- namespace BMeasureApi

Enumerations

- enum BMeasureApi::ErrorNum { BMeasureApi::ErrorNumSystem = 64 , BMeasureApi::ErrorNumDataOverrun = 65 , BMeasureApi::ErrorNumToFast = 66 }
- enum BMeasureApi::NodeType { BMeasureApi::NodeTypeNone = 0 , BMeasureApi::NodeTypeBMeasure1 = 1 }
- enum BMeasureApi::SecurityMode { BMeasureApi::SecurityModeBasic , BMeasureApi::SecurityModeConfig , BMeasureApi::SecurityModeFull }
- enum BMeasureApi::Status {
  BMeasureApi::StatusNone = 0x00 , BMeasureApi::StatusError = 0x01 , BMeasureApi::StatusWarning = 0x02 , BMeasureApi::StatusRun = 0x04 ,
  BMeasureApi::StatusTriggerWait = 0x08 , BMeasureApi::StatusEnd0 = 0x10 , BMeasureApi::StatusEnd1 = 0x20 , BMeasureApi::StatusDataOverrun = 0x40 ,
  BMeasureApi::StatusFpgaOverrun = 0x80 , BMeasureApi::StatusAlarm = 0x0100 }
- enum BMeasureApi::Mode {
  BMeasureApi::ModeIdle = 0 , BMeasureApi::ModeRun = 1 , BMeasureApi::ModeRunProgram = 2 , BMeasureApi::ModeInternal = 3 ,
  BMeasureApi::ModeSleep = 4 , BMeasureApi::ModeDemo1 = 5 }
- enum BMeasureApi::BlockTypes { BMeasureApi::BlockTypeInfo = 0x424E4531 , BMeasureApi::BlockTypeData = 0x424E4532 }
- enum BMeasureApi::ChannelType {
  BMeasureApi::ChannelTypeNone = 0 , BMeasureApi::ChannelTypeAnalogueIn = 1 , BMeasureApi::ChannelTypeAnalogueOut = 2 , BMeasureApi::ChannelTypeDigitalIn = 3 ,
  BMeasureApi::ChannelTypeDigitalOut = 4 }
- enum BMeasureApi::SampleType {
  BMeasureApi::SampleTypeNone = 0 , BMeasureApi::SampleTypeBool = 1 , BMeasureApi::SampleTypeInt8 = 2 , BMeasureApi::SampleTypeInt16 = 3 ,
  BMeasureApi::SampleTypeInt32 = 4 , BMeasureApi::SampleTypeFloat32 = 5 , BMeasureApi::SampleTypeFloat64 = 6 }
- enum BMeasureApi::SyncMode { BMeasureApi::SyncModeOff = 0 , BMeasureApi::SyncModeMaster = 1 , BMeasureApi::SyncModeSlave = 2 }
- enum BMeasureApi::MeasureMode { BMeasureApi::MeasureModeOff = 0 , BMeasureApi::MeasureModeOneShot = 1 , BMeasureApi::MeasureModeRepeat = 2 , BMeasureApi::MeasureModeContinuous = 3 }
- enum BMeasureApi::MeasureOption { BMeasureApi::MeasureOptionNone = 0 , BMeasureApi::MeasureOptionProcess = 0x01 }
- enum BMeasureApi::TriggerMode { BMeasureApi::TriggerModeOff = 0 , BMeasureApi::TriggerModePositive = 1 , BMeasureApi::TriggerModeNegative = 2 }
- enum BMeasureApi::TriggerConfig { BMeasureApi::TriggerConfigNone = 0 }
- enum BMeasureApi::DigitalMode {
  BMeasureApi::DigitalModeInput = 0 , BMeasureApi::DigitalModeOutput = 1 , BMeasureApi::DigitalModeInOut = 2 , BMeasureApi::DigitalModeSyncMaster = 3 ,
  BMeasureApi::DigitalModeSyncSlave = 4 }
- enum BMeasureApi::AwgMode {
  BMeasureApi::AwgModeNone , BMeasureApi::AwgModeDc , BMeasureApi::AwgModeSine , BMeasureApi::AwgModeSquare ,
  BMeasureApi::AwgModeTriangle , BMeasureApi::AwgModeNoise , BMeasureApi::AwgModeTrackRms , BMeasureApi::AwgModeTrackMean ,
  BMeasureApi::AwgModeArbitrary }
- enum BMeasureApi::AwgOutput { BMeasureApi::AwgOutputNone , BMeasureApi::AwgOutputAO0 , BMeasureApi::AwgOutputAO1 , BMeasureApi::AwgOutputAO01 }
- enum BMeasureApi::FileType { BMeasureApi::FileTypeNone , BMeasureApi::FileTypeFile , BMeasureApi::FileTypeDir }

- enum BMeasureApi::FilesysDeleteType { BMeasureApi::FilesysDeleteTypeNone , BMeasureApi::FilesysDeleteTypeData , BMeasureApi::FilesysDeleteTypeFormat }
- enum BMeasureApi::LogData { BMeasureApi::LogDataOff , BMeasureApi::LogDataRaw = 0x01 , BMeasureApi::LogDataProcessed = 0x02 }
- enum BMeasureApi::LogDataMode { BMeasureApi::LogDataModeNormal , BMeasureApi::LogDataModeDeleteOld }
- enum BMeasureApi::DataType { BMeasureApi::DataTypeFloat32 , BMeasureApi::DataType125i , BMeasureApi::DataTypeProc }
- enum BMeasureApi::DataSend { BMeasureApi::DataSendOff , BMeasureApi::DataSendStatus = 0x01 , BMeasureApi::DataSendRaw = 0x02 , BMeasureApi::DataSendProcessed = 0x04 }
- enum BMeasureApi::CalibrateStage {
  BMeasureApi::CalibrateStageNone = 0 , BMeasureApi::CalibrateStageClear = 1 , BMeasureApi::CalibrateStageSettle = 2 , BMeasureApi::CalibrateStageAdcOffsets = 3 ,
  BMeasureApi::CalibrateStageDacOffsets = 4 , BMeasureApi::CalibrateStageDacScaling0 = 5 , BMeasureApi::CalibrateStageDa = 6 , BMeasureApi::CalibrateStageAdcScaling = 7 ,
  BMeasureApi::CalibrateStageAttenScaling = 8 , BMeasureApi::CalibrationStageFiveVolts = 9 , BMeasureApi::CalibrateStageAd = 10 , BMeasureApi::CalibrateStageChanClear = 11 ,
  BMeasureApi::CalibrateStageChanOffsets = 12 , BMeasureApi::CalibrateStageChanScaling = 13 }
- enum BMeasureApi::MessageSource {
  BMeasureApi::MessageSourceGeneral = 0 , BMeasureApi::MessageSourceDebug = 1 , BMeasureApi::MessageSourceTest = 2 , BMeasureApi::MessageSourceWifi = 3 ,
  BMeasureApi::MessageSourceWifiTest = 4 }
- enum BMeasureApi::NetworkMode { BMeasureApi::NetworkModeOff = 0 , BMeasureApi::NetworkModeDhcp = 1 , BMeasureApi::NetworkModeManual = 2 }
- enum BMeasureApi::WifiMode { BMeasureApi::WifiModeOff , BMeasureApi::WifiModeClient , BMeasureApi::WifiModeAp }
- enum BMeasureApi::AlarmMode {
  BMeasureApi::AlarmModeOff , BMeasureApi::AlarmModeHigh , BMeasureApi::AlarmModeLow , BMeasureApi::AlarmModeRar ,
  BMeasureApi::AlarmModeMagnitude }
- enum BMeasureApi::AlarmOutput {
  BMeasureApi::AlarmOutputOff , BMeasureApi::AlarmOutputDioHigh , BMeasureApi::AlarmOutputDioLow , BMeasureApi::AlarmOutputRelayOn ,
  BMeasureApi::AlarmOutputRelayOff }
- enum BMeasureApi::EventMode { BMeasureApi::EventModeOff , BMeasureApi::EventModeAlarm , BMeasureApi::EventModeSecond }
- enum BMeasureApi::Rs485Mode { BMeasureApi::Rs485ModeOff , BMeasureApi::Rs485ModeBoap }
- enum BMeasureApi::BMeasFileType { BMeasureApi::BMeasFileTypeBlock512 , BMeasureApi::BMeasFileTypeStream }
- enum BMeasureApi::WifiCmd {
  BMeasureApi::WifiCmdOff , BMeasureApi::WifiCmdOn , BMeasureApi::WifiCmdScan , BMeasureApi::WifiCmdConnect ,
  BMeasureApi::WifiCmdDisconnect }
- enum BMeasureApi::WifiStatus {
  BMeasureApi::WifiStatusOff , BMeasureApi::WifiStatusOn , BMeasureApi::WifiStatusConnecting , BMeasureApi::WifiStatusConnected ,
  BMeasureApi::WifiStatusConnectedInternet , BMeasureApi::WifStatusAP }

Functions

- **BString** BMeasureApi::toBString (ErrorNum v)
- **BError** BMeasureApi::fromBString (**BString** str, ErrorNum &v)
- **BString** BMeasureApi::toBStringJson (**BString** n, ErrorNum v)
- **BString** BMeasureApi::toBString (NodeType v)
- **BError** BMeasureApi::fromBString (**BString** str, NodeType &v)
- **BString** BMeasureApi::toBStringJson (**BString** n, NodeType v)

- **BString** BMeasureApi::toBString (SecurityMode v)
- **BError** BMeasureApi::fromBString (**BString** str, SecurityMode &v)
- **BString** BMeasureApi::toBStringJson (**BString** n, SecurityMode v)
- **BString** BMeasureApi::toBString (Status v)
- **BError** BMeasureApi::fromBString (**BString** str, Status &v)
- **BString** BMeasureApi::toBStringJson (**BString** n, Status v)
- **BString** BMeasureApi::toBString (Mode v)
- **BError** BMeasureApi::fromBString (**BString** str, Mode &v)
- **BString** BMeasureApi::toBStringJson (**BString** n, Mode v)
- **BString** BMeasureApi::toBString (BlockTypes v)
- **BError** BMeasureApi::fromBString (**BString** str, BlockTypes &v)
- **BString** BMeasureApi::toBStringJson (**BString** n, BlockTypes v)
- **BString** BMeasureApi::toBString (ChannelType v)
- **BError** BMeasureApi::fromBString (**BString** str, ChannelType &v)
- **BString** BMeasureApi::toBStringJson (**BString** n, ChannelType v)
- **BString** BMeasureApi::toBString (SampleType v)
- **BError** BMeasureApi::fromBString (**BString** str, SampleType &v)
- **BString** BMeasureApi::toBStringJson (**BString** n, SampleType v)
- **BString** BMeasureApi::toBString (SyncMode v)
- **BError** BMeasureApi::fromBString (**BString** str, SyncMode &v)
- **BString** BMeasureApi::toBStringJson (**BString** n, SyncMode v)
- **BString** BMeasureApi::toBString (MeasureMode v)
- **BError** BMeasureApi::fromBString (**BString** str, MeasureMode &v)
- **BString** BMeasureApi::toBStringJson (**BString** n, MeasureMode v)
- **BString** BMeasureApi::toBString (MeasureOption v)
- **BError** BMeasureApi::fromBString (**BString** str, MeasureOption &v)
- **BString** BMeasureApi::toBStringJson (**BString** n, MeasureOption v)
- **BString** BMeasureApi::toBString (TriggerMode v)
- **BError** BMeasureApi::fromBString (**BString** str, TriggerMode &v)
- **BString** BMeasureApi::toBStringJson (**BString** n, TriggerMode v)
- **BString** BMeasureApi::toBString (TriggerConfig v)
- **BError** BMeasureApi::fromBString (**BString** str, TriggerConfig &v)
- **BString** BMeasureApi::toBStringJson (**BString** n, TriggerConfig v)
- **BString** BMeasureApi::toBString (DigitalMode v)
- **BError** BMeasureApi::fromBString (**BString** str, DigitalMode &v)
- **BString** BMeasureApi::toBStringJson (**BString** n, DigitalMode v)
- **BString** BMeasureApi::toBString (AwgMode v)
- **BError** BMeasureApi::fromBString (**BString** str, AwgMode &v)
- **BString** BMeasureApi::toBStringJson (**BString** n, AwgMode v)
- **BString** BMeasureApi::toBString (AwgOutput v)
- **BError** BMeasureApi::fromBString (**BString** str, AwgOutput &v)
- **BString** BMeasureApi::toBStringJson (**BString** n, AwgOutput v)
- **BString** BMeasureApi::toBString (FileType v)
- **BError** BMeasureApi::fromBString (**BString** str, FileType &v)
- **BString** BMeasureApi::toBStringJson (**BString** n, FileType v)
- **BString** BMeasureApi::toBString (FilesysDeleteType v)
- **BError** BMeasureApi::fromBString (**BString** str, FilesysDeleteType &v)
- **BString** BMeasureApi::toBStringJson (**BString** n, FilesysDeleteType v)
- **BString** BMeasureApi::toBString (LogData v)
- **BError** BMeasureApi::fromBString (**BString** str, LogData &v)
- **BString** BMeasureApi::toBStringJson (**BString** n, LogData v)
- **BString** BMeasureApi::toBString (LogDataMode v)
- **BError** BMeasureApi::fromBString (**BString** str, LogDataMode &v)
- **BString** BMeasureApi::toBStringJson (**BString** n, LogDataMode v)
- **BString** BMeasureApi::toBString (DataType v)

- **BError** BMeasureApi::fromBString (**BString** str, DataType &v)
- **BString** BMeasureApi::toBStringJson (**BString** n, DataType v)
- **BString** BMeasureApi::toBString (DataSend v)
- **BError** BMeasureApi::fromBString (**BString** str, DataSend &v)
- **BString** BMeasureApi::toBStringJson (**BString** n, DataSend v)
- **BString** BMeasureApi::toBString (CalibrateStage v)
- **BError** BMeasureApi::fromBString (**BString** str, CalibrateStage &v)
- **BString** BMeasureApi::toBStringJson (**BString** n, CalibrateStage v)
- **BString** BMeasureApi::toBString (MessageSource v)
- **BError** BMeasureApi::fromBString (**BString** str, MessageSource &v)
- **BString** BMeasureApi::toBStringJson (**BString** n, MessageSource v)
- **BString** BMeasureApi::toBString (NetworkMode v)
- **BError** BMeasureApi::fromBString (**BString** str, NetworkMode &v)
- **BString** BMeasureApi::toBStringJson (**BString** n, NetworkMode v)
- **BString** BMeasureApi::toBString (WifiMode v)
- **BError** BMeasureApi::fromBString (**BString** str, WifiMode &v)
- **BString** BMeasureApi::toBStringJson (**BString** n, WifiMode v)
- **BString** BMeasureApi::toBString (AlarmMode v)
- **BError** BMeasureApi::fromBString (**BString** str, AlarmMode &v)
- **BString** BMeasureApi::toBStringJson (**BString** n, AlarmMode v)
- **BString** BMeasureApi::toBString (AlarmOutput v)
- **BError** BMeasureApi::fromBString (**BString** str, AlarmOutput &v)
- **BString** BMeasureApi::toBStringJson (**BString** n, AlarmOutput v)
- **BString** BMeasureApi::toBString (EventMode v)
- **BError** BMeasureApi::fromBString (**BString** str, EventMode &v)
- **BString** BMeasureApi::toBStringJson (**BString** n, EventMode v)
- **BString** BMeasureApi::toBString (Rs485Mode v)
- **BError** BMeasureApi::fromBString (**BString** str, Rs485Mode &v)
- **BString** BMeasureApi::toBStringJson (**BString** n, Rs485Mode v)
- **BString** BMeasureApi::toBString (BMeasFileType v)
- **BError** BMeasureApi::fromBString (**BString** str, BMeasFileType &v)
- **BString** BMeasureApi::toBStringJson (**BString** n, BMeasFileType v)
- **BString** BMeasureApi::toBString (WifiCmd v)
- **BError** BMeasureApi::fromBString (**BString** str, WifiCmd &v)
- **BString** BMeasureApi::toBStringJson (**BString** n, WifiCmd v)
- **BString** BMeasureApi::toBString (WifiStatus v)
- **BError** BMeasureApi::fromBString (**BString** str, WifiStatus &v)
- **BString** BMeasureApi::toBStringJson (**BString** n, WifiStatus v)

8.7 BMeasureLib.cpp File Reference

```
#include <BMeasureLib.h>
#include <BObjStringFormat.h>
#include <BDebug.h>
```

Namespaces

- namespace BMeasureApi

Macros

- #define BDEBUGL1 0
- #define BDEBUGL2 0

Functions

- **BString** toBStringJson (**BString** n, BMeasureApi::Version v)
- **BString** toBStringJson (**BString** n, BMeasureApi::AlarmConfig v)
- **BString** toBStringJson (**BString** n, BMeasureApi::NodeInfo v)

8.7.1 Macro Definition Documentation

8.7.1.1 BDEBUGL1

```
#define BDEBUGL1 0
```

8.7.1.2 BDEBUGL2

```
#define BDEBUGL2 0
```

8.7.2 Function Documentation

8.7.2.1 toBStringJson() [1/3]

```
BString toBStringJson (
             BString n,
            BMeasureApi::AlarmConfig v )
```

8.7.2.2 toBStringJson() [2/3]

```
BString toBStringJson (
             BString n,
            BMeasureApi::NodeInfo v )
```

8.7.2.3 toBStringJson() [3/3]

```
BString toBStringJson (
            BString n,
           BMeasureApi::Version v )
```

8.8 BMeasureLib.h File Reference

```
#include <BMeasureD.h>
```

Namespaces

- namespace BMeasureApi

Typedefs

- typedef **BArray**< ChannelConfig > BMeasureApi::ChannelConfigs

Functions

- **BString** toBStringJson (**BString** n, BMeasureApi::Version v)
- **BString** toBStringJson (**BString** n, BMeasureApi::AlarmConfig v)
- **BString** toBStringJson (**BString** n, BMeasureApi::NodeInfo v)

8.8.1 Function Documentation

8.8.1.1 toBStringJson() [1/3]

```
BString toBStringJson (
            BString n,
           BMeasureApi::AlarmConfig v )
```

8.8.1.2 toBStringJson() [2/3]

```
BString toBStringJson (
            BString n,
           BMeasureApi::NodeInfo v )
```

8.8.1.3 toBStringJson() [3/3]

```
BString toBStringJson (
            BString n,
           BMeasureApi::Version v )
```

8.9 BMeasureS.cpp File Reference

```
#include <BMeasureS.h>
#include <string.h>
```

Namespaces

- namespace BMeasureApi

8.10 BMeasureUnit.cpp File Reference

```
#include <BMeasureUnit.h>
#include <CommsSerial.h>
#include <CommsNet.h>
#include <CommsUsb.h>
#include <BDir.h>
#include <BSys.h>
#include <libusb-1.0/libusb.h>
#include <BMdns.h>
#include <BDebug.h>
#include <unistd.h>
#include <errno.h>
```

Namespaces

- namespace BMeasureApi

Macros

- #define BDEBUGL1 0
- #define BDEBUGL2 0
- #define BDEBUGL3 0
- #define CONVERT_FLOAT 0

  Convert to floating point.

Functions

- const char ∗ BMeasureApi::channelTypeString (ChannelType type)
- const char ∗ BMeasureApi::sampleTypeString (SampleType type)
- **BFloat32** BMeasureApi::toFloat (**BUInt32** v)

8.10.1 Macro Definition Documentation

8.10.1.1 BDEBUGL1

```
#define BDEBUGL1 0
```

8.10.1.2 BDEBUGL2

```
#define BDEBUGL2 0
```

8.10.1.3 BDEBUGL3

```
#define BDEBUGL3 0
```

8.10.1.4 CONVERT_FLOAT

```
#define CONVERT_FLOAT 0
```

Convert to floating point.

8.11 BMeasureUnit.h File Reference

```
#include <BMeasureD.h>
#include <BMeasureB.h>
#include <BTask.h>
```

Classes

- class BMeasureApi::BMeasureUnitDevice
- class BMeasureApi::BMeasureUnit

Namespaces

- namespace BMeasureApi

Functions

- const char ∗ BMeasureApi::channelTypeString (ChannelType type)
- const char ∗ BMeasureApi::sampleTypeString (SampleType type)

8.12 BMeasureUnits.cpp File Reference

```
#include <BMeasureUnits.h>
#include <BDebug.h>
#include <unistd.h>
```

Namespaces

- namespace BMeasureApi

Macros

- #define BDEBUGL1 0
- #define BDEBUGL2 0
- #define BDEBUGL3 0

Functions

- static int BMeasureApi::unitSort (BMeasureUnit1 ∗&u1, BMeasureUnit1 ∗&u2)

8.12.1 Macro Definition Documentation

8.12.1.1 BDEBUGL1

```
#define BDEBUGL1 0
```

8.12.1.2 BDEBUGL2

```
#define BDEBUGL2 0
```

8.12.1.3 BDEBUGL3

```
#define BDEBUGL3 0
```

8.13 BMeasureUnits.h File Reference

```
#include <BMeasureUnit.h>
#include <BMutex.h>
#include <BSemaphore.h>
```

Classes

- class BMeasureApi::BMeasureUnit1
- class BMeasureApi::BMeasureUnitsDataBlock
- class BMeasureApi::BMeasureUnits

Namespaces

- namespace BMeasureApi

8.14 CommsNet.cpp File Reference

```
#include <CommsNet.h>
#include <BPoll.h>
#include <BSys.h>
#include <BDebug.h>
#include <string.h>
#include <errno.h>
```

Namespaces

- namespace BMeasureApi

Macros

- #define BDEBUGL1 0
- #define BDEBUGL2 0
- #define BDEBUGL3 0

8.14.1 Macro Definition Documentation

8.14.1.1 BDEBUGL1

```
#define BDEBUGL1 0
```

8.14.1.2 BDEBUGL2

```
#define BDEBUGL2 0
```

8.14.1.3 BDEBUGL3

```
#define BDEBUGL3 0
```

8.15 CommsNet.h File Reference

```
#include <BComms.h>
#include <BSocket.h>
```

Classes

- class BMeasureApi::CommsNet

Namespaces

- namespace BMeasureApi

8.16 CommsSerial.cpp File Reference

8.17 CommsSerial.h File Reference

```
#include <BComms.h>
```

Classes

- class BMeasureApi::CommsSerial

Namespaces

- namespace BMeasureApi

8.18 CommsUsb.cpp File Reference

```
#include <CommsUsb.h>
#include <BSys.h>
#include <libusb-1.0/libusb.h>
#include <stdio.h>
#include <stdlib.h>
#include <BDebug.h>
```

Namespaces

- namespace BMeasureApi

Macros

- #define BDEBUGL1 0
- #define BDEBUGL2 0

Functions

- static **BUInt32** BMeasureApi::roundDown512 (**BUInt32** size)

8.18.1 Macro Definition Documentation

8.18.1.1 BDEBUGL1

```
#define BDEBUGL1 0
```

8.18.1.2 BDEBUGL2

```
#define BDEBUGL2 0
```

8.19 CommsUsb.h File Reference

```
#include <BComms.h>
#include <BMutex.h>
#include <libusb-1.0/libusb.h>
```

Classes

- class BMeasureApi::CommsUsb

Namespaces

- namespace BMeasureApi

8.20 DataFile.cpp File Reference

```
#include <DataFile.h>
#include <BoapMc1.h>
#include <BBuffer.h>
#include <BDebug.h>
#include <errno.h>
```

Namespaces

- namespace BMeasureApi

Macros

- #define BDEBUGL1 0
- #define BDEBUGL2 0

Enumerations

- enum BMeasureApi::TdsDataType {
  BMeasureApi::TdsTypeVoid , BMeasureApi::TdsTypeI8 , BMeasureApi::TdsTypeI16 , BMeasureApi::TdsTypeI32
  ,
  BMeasureApi::TdsTypeI64 , BMeasureApi::TdsTypeU8 , BMeasureApi::TdsTypeU16 , BMeasureApi::TdsTypeU32
  ,
  BMeasureApi::TdsTypeU64 , BMeasureApi::TdsTypeSingleFloat , BMeasureApi::TdsTypeDoubleFloat ,
  BMeasureApi::TdsTypeExtendedFloat ,
  BMeasureApi::TdsTypeSingleFloatWithUnit =0x19 , BMeasureApi::TdsTypeDoubleFloatWithUnit , BMeasureApi::TdsTypeExten
  , BMeasureApi::TdsTypeString =0x20 ,
  BMeasureApi::TdsTypeBoolean =0x21 , BMeasureApi::TdsTypeTimeStamp =0x44 , BMeasureApi::TdsTypeFixedPoint
  =0x4F , BMeasureApi::TdsTypeComplexSingleFloat =0x08000c ,
  BMeasureApi::TdsTypeComplexDoubleFloat =0x10000d , BMeasureApi::TdsTypeDAQmxRawData =0x←
  FFFFFFFF }

Functions

- const **BUInt32** BMeasureApi::TocMetaData (1<< 1)
- const **BUInt32** BMeasureApi::TocNewObjList (1<< 2)
- const **BUInt32** BMeasureApi::TocRawData (1<< 3)
- const **BUInt32** BMeasureApi::TocInterleavedData (1<< 5)
- const **BUInt32** BMeasureApi::TocBigEndian (1<< 6)
- const **BUInt32** BMeasureApi::TocDaqRawData (1<< 7)
- **BUInt32** BMeasureApi::round512 (**BUInt32** s)

8.20.1 Macro Definition Documentation

8.20.1.1 BDEBUGL1

```
#define BDEBUGL1 0
```

8.20.1.2 BDEBUGL2

```
#define BDEBUGL2 0
```

8.21 DataFile.h File Reference

```
#include <BString.h>
#include <BFile.h>
#include <BMeasureLib.h>
#include <BoapMc1.h>
```

Classes

- class BMeasureApi::DataFile

Namespaces

- namespace BMeasureApi

8.22 Dfu.cpp File Reference

```
#include <Dfu.h>
#include <BFile.h>
#include <BDebug.h>
#include <unistd.h>
```

Classes

- struct BFirmwareInfo

Macros

- #define BDEBUGL1 0
- #define BDEBUGL2 0
- #define STATE_APP_IDLE 0x00
- #define STATE_APP_DETACH 0x01
- #define STATE_DFU_IDLE 0x02
- #define STATE_DFU_DOWNLOAD_SYNC 0x03
- #define STATE_DFU_DOWNLOAD_BUSY 0x04
- #define STATE_DFU_DOWNLOAD_IDLE 0x05
- #define STATE_DFU_MANIFEST_SYNC 0x06
- #define STATE_DFU_MANIFEST 0x07
- #define STATE_DFU_MANIFEST_WAIT_RESET 0x08
- #define STATE_DFU_UPLOAD_IDLE 0x09
- #define STATE_DFU_ERROR 0x0a
- #define DFU_STATUS_OK 0x00
- #define DFU_STATUS_ERROR_TARGET 0x01
- #define DFU_STATUS_ERROR_FILE 0x02
- #define DFU_STATUS_ERROR_WRITE 0x03
- #define DFU_STATUS_ERROR_ERASE 0x04
- #define DFU_STATUS_ERROR_CHECK_ERASED 0x05
- #define DFU_STATUS_ERROR_PROG 0x06
- #define DFU_STATUS_ERROR_VERIFY 0x07
- #define DFU_STATUS_ERROR_ADDRESS 0x08
- #define DFU_STATUS_ERROR_NOTDONE 0x09
- #define DFU_STATUS_ERROR_FIRMWARE 0x0a
- #define DFU_STATUS_ERROR_VENDOR 0x0b
- #define DFU_STATUS_ERROR_USBR 0x0c
- #define DFU_STATUS_ERROR_POR 0x0d
- #define DFU_STATUS_ERROR_UNKNOWN 0x0e
- #define DFU_STATUS_ERROR_STALLEDPKT 0x0f
- #define DFU_DETACH 0
- #define DFU_DNLOAD 1
- #define DFU_UPLOAD 2
- #define DFU_GETSTATUS 3
- #define DFU_CLRSTATUS 4
- #define DFU_GETSTATE 5
- #define DFU_ABORT 6
- #define DFU_IFF_DFU 0x0001 /∗ DFU Mode, (not Runtime) ∗/
- #define DFU_IFF_VENDOR 0x0100
- #define DFU_IFF_PRODUCT 0x0200
- #define DFU_IFF_CONFIG 0x0400
- #define DFU_IFF_IFACE 0x0800
- #define DFU_IFF_ALT 0x1000
- #define DFU_IFF_DEVNUM 0x2000
- #define DFU_IFF_PATH 0x4000

Enumerations

- enum dfuse_command { SET_ADDRESS , ERASE_PAGE , MASS_ERASE , READ_UNPROTECT }

Functions

- static **BInt32** pageNumber (**BUInt32** address)
- static **BUInt32** pageAddress (**BUInt32** page)

Variables

- const **BUInt32** BFirmwareInfoMagic = 0xBBEEAA00
- const **BUInt8** BFirmwareInfoEncrypt1 = 0x40

8.22.1 Macro Definition Documentation

8.22.1.1 BDEBUGL1

```
#define BDEBUGL1 0
```

8.22.1.2 BDEBUGL2

```
#define BDEBUGL2 0
```

8.22.1.3 DFU_ABORT

```
#define DFU_ABORT 6
```

8.22.1.4 DFU_CLRSTATUS

```
#define DFU_CLRSTATUS 4
```

8.22.1.5 DFU_DETACH

```
#define DFU_DETACH 0
```

8.22.1.6 DFU_DNLOAD

```
#define DFU_DNLOAD 1
```

8.22.1.7 DFU_GETSTATE

```
#define DFU_GETSTATE 5
```

8.22.1.8 DFU_GETSTATUS

```
#define DFU_GETSTATUS 3
```

8.22.1.9 DFU_IFF_ALT

```
#define DFU_IFF_ALT 0x1000
```

8.22.1.10 DFU_IFF_CONFIG

```
#define DFU_IFF_CONFIG 0x0400
```

8.22.1.11 DFU_IFF_DEVNUM

```
#define DFU_IFF_DEVNUM 0x2000
```

8.22.1.12 DFU_IFF_DFU

```
#define DFU_IFF_DFU 0x0001 /* DFU Mode, (not Runtime) */
```

8.22.1.13 DFU_IFF_IFACE

```
#define DFU_IFF_IFACE 0x0800
```

8.22.1.14 DFU_IFF_PATH

```
#define DFU_IFF_PATH 0x4000
```

8.22.1.15 DFU_IFF_PRODUCT

```
#define DFU_IFF_PRODUCT 0x0200
```

8.22.1.16 DFU_IFF_VENDOR

```
#define DFU_IFF_VENDOR 0x0100
```

8.22.1.17 DFU_STATUS_ERROR_ADDRESS

```
#define DFU_STATUS_ERROR_ADDRESS 0x08
```

8.22.1.18 DFU_STATUS_ERROR_CHECK_ERASED

```
#define DFU_STATUS_ERROR_CHECK_ERASED 0x05
```

8.22.1.19 DFU_STATUS_ERROR_ERASE

```
#define DFU_STATUS_ERROR_ERASE 0x04
```

8.22.1.20 DFU_STATUS_ERROR_FILE

```
#define DFU_STATUS_ERROR_FILE 0x02
```

8.22.1.21 DFU_STATUS_ERROR_FIRMWARE

```
#define DFU_STATUS_ERROR_FIRMWARE 0x0a
```

8.22.1.22 DFU_STATUS_ERROR_NOTDONE

```
#define DFU_STATUS_ERROR_NOTDONE 0x09
```

8.22.1.23 DFU_STATUS_ERROR_POR

```
#define DFU_STATUS_ERROR_POR 0x0d
```

8.22.1.24 DFU_STATUS_ERROR_PROG

```
#define DFU_STATUS_ERROR_PROG 0x06
```

8.22.1.25 DFU_STATUS_ERROR_STALLEDPKT

```
#define DFU_STATUS_ERROR_STALLEDPKT 0x0f
```

8.22.1.26 DFU_STATUS_ERROR_TARGET

```
#define DFU_STATUS_ERROR_TARGET 0x01
```

8.22.1.27 DFU_STATUS_ERROR_UNKNOWN

```
#define DFU_STATUS_ERROR_UNKNOWN 0x0e
```

8.22.1.28 DFU_STATUS_ERROR_USBR

```
#define DFU_STATUS_ERROR_USBR 0x0c
```

8.22.1.29 DFU_STATUS_ERROR_VENDOR

```
#define DFU_STATUS_ERROR_VENDOR 0x0b
```

8.22.1.30 DFU_STATUS_ERROR_VERIFY

```
#define DFU_STATUS_ERROR_VERIFY 0x07
```

8.22.1.31 DFU_STATUS_ERROR_WRITE

```
#define DFU_STATUS_ERROR_WRITE 0x03
```

8.22.1.32 DFU_STATUS_OK

```
#define DFU_STATUS_OK 0x00
```

8.22.1.33 DFU_UPLOAD

```
#define DFU_UPLOAD 2
```

8.22.1.34 STATE_APP_DETACH

```
#define STATE_APP_DETACH 0x01
```

8.22.1.35 STATE_APP_IDLE

```
#define STATE_APP_IDLE 0x00
```

8.22.1.36 STATE_DFU_DOWNLOAD_BUSY

```
#define STATE_DFU_DOWNLOAD_BUSY 0x04
```

8.22.1.37 STATE_DFU_DOWNLOAD_IDLE

```
#define STATE_DFU_DOWNLOAD_IDLE 0x05
```

8.22.1.38 STATE_DFU_DOWNLOAD_SYNC

```
#define STATE_DFU_DOWNLOAD_SYNC 0x03
```

8.22.1.39 STATE_DFU_ERROR

```
#define STATE_DFU_ERROR 0x0a
```

8.22.1.40 STATE_DFU_IDLE

```
#define STATE_DFU_IDLE 0x02
```

8.22.1.41 STATE_DFU_MANIFEST

```
#define STATE_DFU_MANIFEST 0x07
```

8.22.1.42 STATE_DFU_MANIFEST_SYNC

```
#define STATE_DFU_MANIFEST_SYNC 0x06
```

8.22.1.43 STATE_DFU_MANIFEST_WAIT_RESET

```
#define STATE_DFU_MANIFEST_WAIT_RESET 0x08
```

8.22.1.44 STATE_DFU_UPLOAD_IDLE

```
#define STATE_DFU_UPLOAD_IDLE 0x09
```

8.22.2 Enumeration Type Documentation

8.22.2.1 dfuse_command

```
enum dfuse_command
```

Enumerator

SET_ADDRESS	
ERASE_PAGE	
MASS_ERASE	
READ_UNPROTECT	

8.22.3 Function Documentation

8.22.3.1 pageAddress()

```
static  BUInt32 pageAddress (
              BUInt32 page )   [static]
```

8.22.3.2 pageNumber()

```
static  BInt32 pageNumber (
              BUInt32 address )   [static]
```

8.22.4 Variable Documentation

8.22.4.1 BFirmwareInfoEncrypt1

```
const  BUInt8 BFirmwareInfoEncrypt1 = 0x40
```

8.22.4.2 BFirmwareInfoMagic

```
const  BUInt32 BFirmwareInfoMagic = 0xBBEEAA00
```

8.23 Dfu.h File Reference

```
#include <BError.h>
#include <libusb-1.0/libusb.h>
```

Classes

- struct DfuStatus
- class Dfu

  The Dfu access class.

8.24 overview.dox File Reference

Index