2015-2016 Test beam Run Control

Generated by Doxygen 1.8.10

Contents

Chapter 1

Module Index

1.1 Modules

Here is a list of all modules:

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

Data Structure Index

4.1 Data Structures

Here are the data structures with brief descriptions:

Chapter 5

Module Documentation

5.1 Socket communication objects

Data Structures

- class Client

    Base client object for the socket.
- class Messenger

    Base master object for the socket.
- class Socket

    Base socket object from which clients/master from a socket inherit.
- class SocketMessage

    Socket-passed message type.

5.1.1 Detailed Description

5.2 FPGA board control

Data Structures

- class DAQ::FPGAHandler

  Driver for timing detectors' FPGA readout.
- struct DAQ::QuickUSBHandler::Version
- class DAQ::QuickUSBHandler

  Generic QuickUSB communication handler.

5.2.1 Detailed Description

5.3 HPTDC chip control

Data Structures

- class TDCErrorFlag

  Error flags handler.
- class TDCEvent

  HPTDC event parser.
- class DAQ::TDC

  HPTDC object.
- class TDCBoundaryScan
- class TDCControl

  Control word to be sent to the HPTDC chip.
- class TDCRegister

  General register object to interact with a HPTDC chip.
- class TDCSetup

  Setup word to be sent to the HPTDC chip.
- class TDCStatus

Enumerations

- enum AcquisitionMode { CONT_STORAGE, TRIG_MATCH }

  TDC acquisition mode.
- enum DAQ::TDC::AcquisitionMode { DAQ::TDC::CONT_STORAGE, DAQ::TDC::TRIG_MATCH }

  TDC acquisition mode.

5.3.1 Detailed Description

5.3.2 Enumeration Type Documentation

5.3.2.1 enum AcquisitionMode

TDC acquisition mode.

Author

Laurent Forthomme `laurent.forthomme@cern.ch`

Enumerator

CONT_STORAGE

TRIG_MATCH

5.3.2.2 enum DAQ::TDC::AcquisitionMode

TDC acquisition mode.

Enumerator

CONT_STORAGE

TRIG_MATCH

Chapter 6

Namespace Documentation

6.1 DAQ Namespace Reference

Data Structures

- class FPGAHandler

  Driver for timing detectors' FPGA readout.
- class QuickUSBHandler

  Generic QuickUSB communication handler.
- class TDC

  HPTDC object.

6.2 DQM Namespace Reference

Data Structures

- class DQMProcess

  Handler for a common DQM process to run on the socket.
- class GastofCanvas
- class PPSCanvas
- class QuarticCanvas

Chapter 7

Data Structure Documentation

7.1 OnlineDBHandler::BurstInfo Struct Reference

```
#include <OnlineDBHandler.h>
```

Data Fields

- unsigned int burst_id
- unsigned int time_start

7.1.1 Field Documentation

7.1.1.1 unsigned int OnlineDBHandler::BurstInfo::burst_id

7.1.1.2 unsigned int OnlineDBHandler::BurstInfo::time_start

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

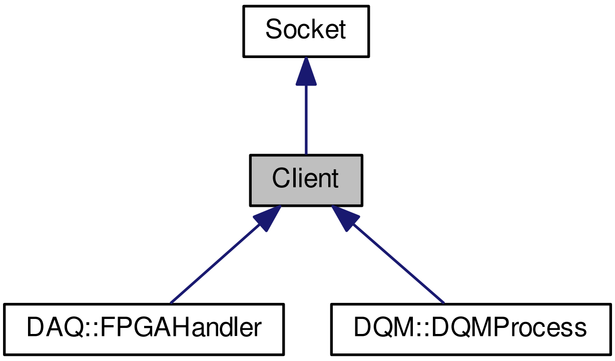
- include/OnlineDBHandler.h

7.2 Client Class Reference

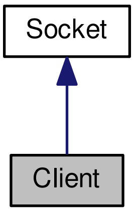
Base client object for the socket.

```
#include <Client.h>
```

Inheritance diagram for Client:



Collaboration diagram for Client:



Public Member Functions

- Client ()

  General void client constructor.
- Client (int port)

  Bind a socket client to a given port.
- virtual ∼Client ()
- bool Connect (const SocketType &type=CLIENT)

  Bind this client to the socket.
- void Disconnect ()

  Unbind this client from the socket.
- void Send (const Message &m) const

  Send a message to the master through the socket.
- void Send (const Exception &e) const
- SocketMessage SendAndReceive (const SocketMessage &m, const MessageKey &a) const
- void Receive ()

Receive a socket message from the master.

- SocketMessage Receive (const MessageKey &key)
- virtual void ParseMessage (const SocketMessage &m)

  Parse a SocketMessage received from the master.
- virtual SocketType GetType () const

  Socket actor type retrieval method.

Private Member Functions

- void Announce ()

  Announce our entry on the socket to its master.

Private Attributes

- int fClientId
- bool fIsConnected
- SocketType fType

Additional Inherited Members

7.2.1 Detailed Description

Base client object for the socket.

Client object used by the server to send/receive commands from the messenger/broadcaster.

Author

Laurent Forthomme `laurent.forthomme@cern.ch`

Date

24 Mar 2015

7.2.2 Constructor & Destructor Documentation

7.2.2.1 Client::Client ( ) `[inline]`

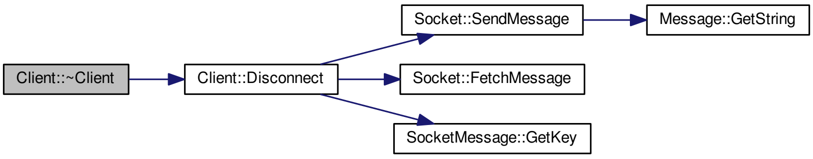
General void client constructor.

7.2.2.2 Client::Client (int *port*)

Bind a socket client to a given port.

7.2.2.3 Client::∼Client ( ) [virtual]

Here is the call graph for this function:

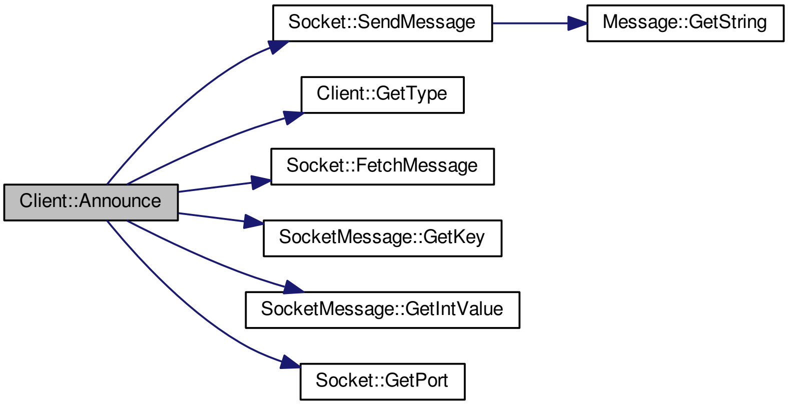


7.2.3 Member Function Documentation

7.2.3.1 void Client::Announce ( ) [private]

Announce our entry on the socket to its master.

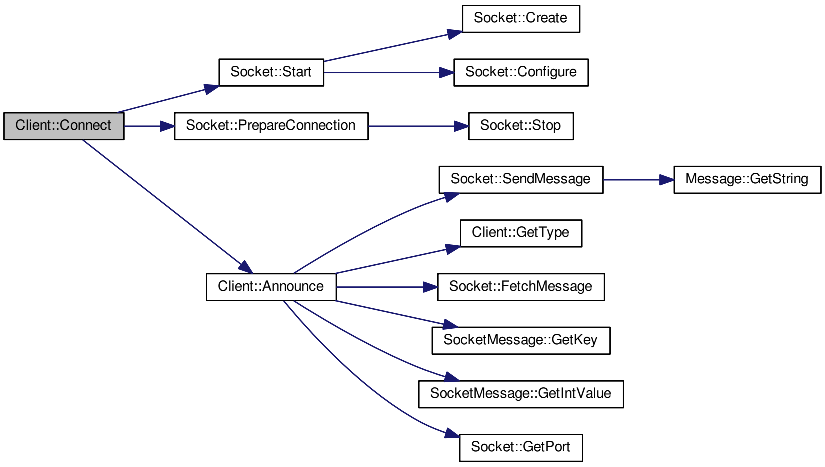
Here is the call graph for this function:



7.2.3.2 bool Client::Connect (const SocketType & *type* = CLIENT)

Bind this client to the socket.

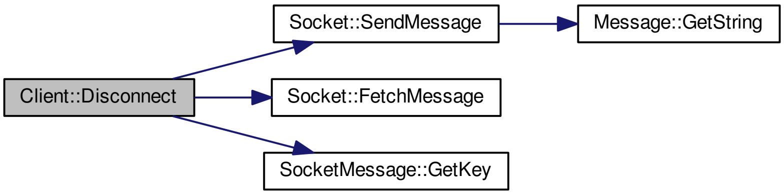
Here is the call graph for this function:



7.2.3.3 void Client::Disconnect ( )

Unbind this client from the socket.

Here is the call graph for this function:



7.2.3.4 virtual SocketType Client::GetType ( ) const [inline], [virtual]

Socket actor type retrieval method.

Reimplemented in DAQ::FPGAHandler.

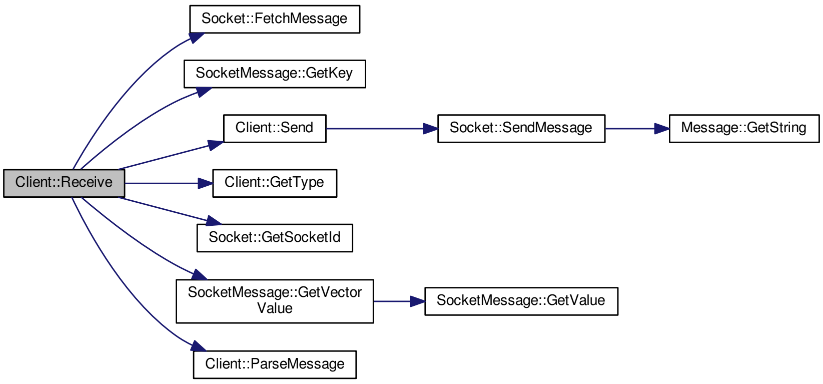
7.2.3.5 virtual void Client::ParseMessage (const SocketMessage & *m*) [inline], [virtual]

Parse a SocketMessage received from the master.

7.2.3.6 void Client::Receive ( )

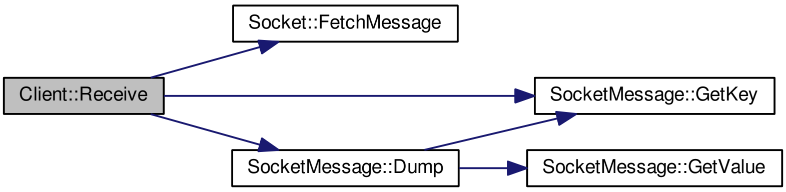
Receive a socket message from the master.

Here is the call graph for this function:



7.2.3.7 SocketMessage Client::Receive (const MessageKey & *key*)

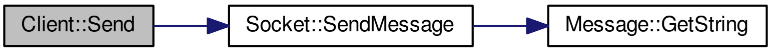
Here is the call graph for this function:



7.2.3.8 void Client::Send (const Message & *m*) const [inline]

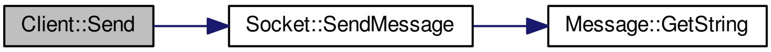
Send a message to the master through the socket.

Here is the call graph for this function:



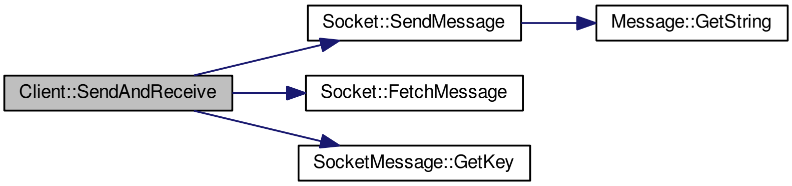
7.2.3.9 void Client::Send (const Exception & *e*) const [inline]

Here is the call graph for this function:



7.2.3.10 SocketMessage Client::SendAndReceive (const SocketMessage & *m,* const MessageKey & *a*) const [inline]

Here is the call graph for this function:



7.2.4 Field Documentation

7.2.4.1 int Client::fClientId [private]

7.2.4.2 bool Client::fIsConnected [private]

7.2.4.3 SocketType Client::fType [private]

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

- include/Client.h
- src/Client.cpp

7.3 DQM::GastofCanvas::Coord Struct Reference

Data Fields

- unsigned int x
- unsigned int y

7.3.1 Field Documentation

7.3.1.1 unsigned int DQM::GastofCanvas::Coord::x

7.3.1.2 unsigned int DQM::GastofCanvas::Coord::y

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

- include/GastofCanvas.h

7.4 DQM::QuarticCanvas::Coord Struct Reference

Data Fields

- unsigned int x
- unsigned int y

7.4.1 Field Documentation

7.4.1.1 unsigned int DQM::QuarticCanvas::Coord::x

7.4.1.2 unsigned int DQM::QuarticCanvas::Coord::y

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

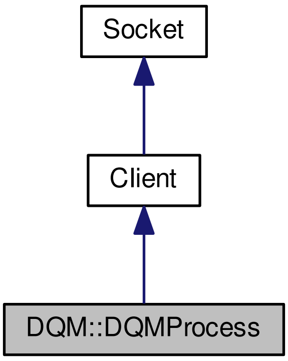
- include/QuarticCanvas.h

7.5 DQM::DQMProcess Class Reference

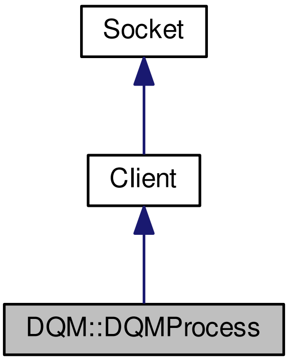
Handler for a common DQM process to run on the socket.

```
#include <DQMProcess.h>
```

Inheritance diagram for DQM::DQMProcess:



Collaboration diagram for DQM::DQMProcess:



Public Types

- enum Action { NewPlot = 0x0, UpdatedPlot = 0x1 }

Public Member Functions

- DQMProcess (int port, unsigned short order=0, const char ∗det_type="")
- ∼DQMProcess ()
- void Run (bool(∗fcn)(unsigned int addr, std::string filename, std::vector< std::string > ∗outputs), const Action &act=NewPlot)

  Run a DQM plotter making use of the board/output filename information.
- void Run (bool(∗fcn)(std::vector< std::string > ∗outputs), const Action &act=NewPlot)

  Run a DQM plotter without any information on the board/output filename.

Private Member Functions

- int ParseMessage (uint32_t ∗board_address, std::string ∗filename)
- bool IsInRun ()

Private Attributes

- unsigned short fOrder
- unsigned int fRunNumber
- std::string fDetectorType
- std::map< unsigned long, std::string > fAddressesCanProcess

Additional Inherited Members

7.5.1 Detailed Description

Handler for a common DQM process to run on the socket.

Author

Laurent Forthomme laurent.forthomme@cern.ch

Date

3 Aug 2015

7.5.2 Member Enumeration Documentation

7.5.2.1 enum DQM::DQMProcess::Action

Enumerator

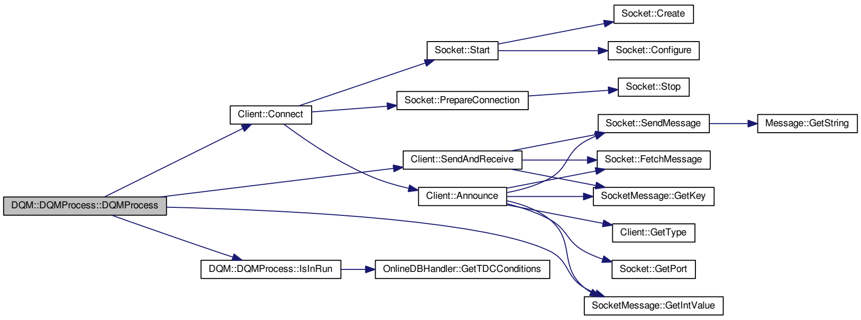
NewPlot

UpdatedPlot

7.5.3 Constructor & Destructor Documentation

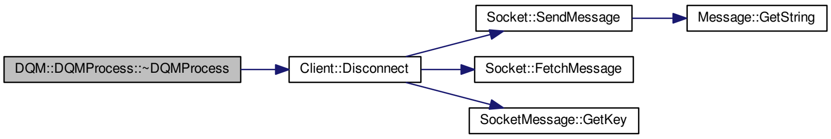
7.5.3.1 DQM::DQMProcess::DQMProcess (int *port*, unsigned short *order* = `0`, const char ∗ *det_type* = " ") `[inline]`

Here is the call graph for this function:



7.5.3.2 DQM::DQMProcess::∼DQMProcess ( ) `[inline]`

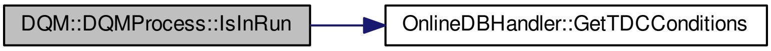
Here is the call graph for this function:



7.5.4 Member Function Documentation

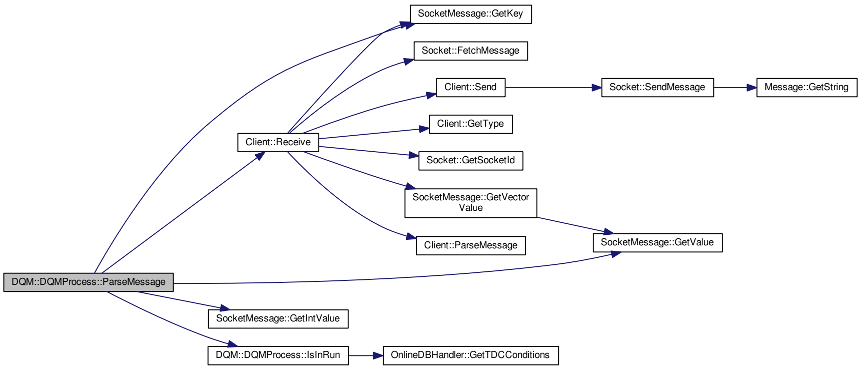
7.5.4.1 bool DQM::DQMProcess::IsInRun ( ) [inline], [private]

Here is the call graph for this function:



7.5.4.2 int DQM::DQMProcess::ParseMessage (uint32_t ∗ *board_address,* std::string ∗ *filename*) [inline], [private]

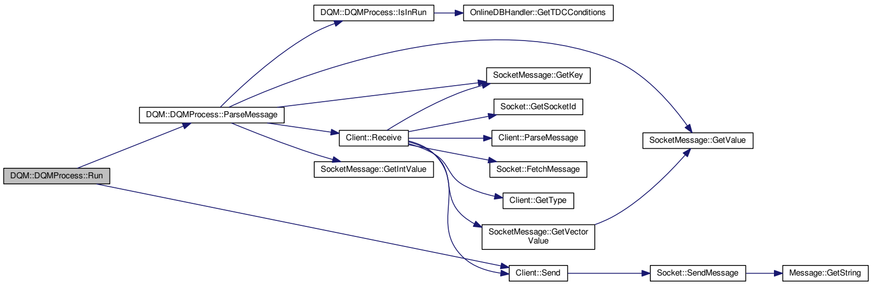
Here is the call graph for this function:



7.5.4.3 void DQM::DQMProcess::Run (bool(∗)(unsigned int addr, std::string filename, std::vector< std::string > ∗outputs) *fcn,* const Action & *act* = NewPlot) [inline]

Run a DQM plotter making use of the board/output filename information.

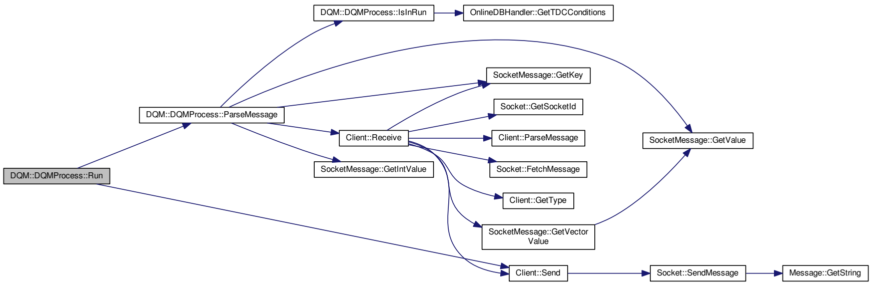
Here is the call graph for this function:



7.5.4.4 void DQM::DQMProcess::Run (bool(∗)(std::vector< std::string > ∗outputs) *fcn*, const Action & *act* = NewPlot) [inline]

Run a DQM plotter without any information on the board/output filename.

Here is the call graph for this function:



7.5.5 Field Documentation

7.5.5.1 std::map<unsigned long, std::string> DQM::DQMProcess::fAddressesCanProcess [private]

7.5.5.2 std::string DQM::DQMProcess::fDetectorType [private]

7.5.5.3 unsigned short DQM::DQMProcess::fOrder [private]

7.5.5.4 unsigned int DQM::DQMProcess::fRunNumber [private]

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

- include/DQMProcess.h

7.6 file_header_t Struct Reference

Header to the output files.

```
#include <FileConstants.h>
```

Data Fields

- uint32_t magic
- uint32_t run_id
- uint32_t spill_id
- uint8_t num_hptdc
- AcquisitionMode acq_mode
- DetectionMode det_mode

7.6.1 Detailed Description

Header to the output files.

General header to store in each collected data file for offline readout. It enable any reader to retrieve the run/spill number, as well as the HPTDC configuration during data collection.

Author

Laurent Forthomme laurent.forthomme@cern.ch

Date

14 Apr 2015

7.6.2 Field Documentation

7.6.2.1 AcquisitionMode file_header_t::acq_mode

7.6.2.2 DetectionMode file_header_t::det_mode

7.6.2.3 uint32_t file_header_t::magic

7.6.2.4 uint8_t file_header_t::num_hptdc

7.6.2.5 uint32_t file_header_t::run_id

7.6.2.6 uint32_t file_header_t::spill_id

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

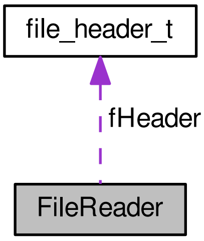
- include/FileConstants.h

7.7 FileReader Class Reference

Handler for a TDC output file readout.

```
#include <FileReader.h>
```

Collaboration diagram for FileReader:



Public Member Functions

- FileReader ()
- FileReader (std::string name)

  Class constructor.
- ∼FileReader ()

- void Open (std::string name)
- bool IsOpen () const
- void Clear ()
- void Dump () const
- unsigned int GetNumTDCs () const
- unsigned int GetRunId () const
- unsigned int GetBurstId () const
- unsigned int GetAcquisitionMode () const
- unsigned int GetDetectionMode () const
- unsigned long GetNumEvents () const
- bool GetNextEvent (TDCEvent ∗)
- bool GetNextMeasurement (unsigned int channel_id, TDCMeasurement ∗mc)

  Fetch the next full measurement on a given channel.

Private Attributes

- std::ifstream fFile
- file_header_t fHeader
- AcquisitionMode fReadoutMode
- time_t fWriteTime
- unsigned long fNumEvents

7.7.1 Detailed Description

Handler for a TDC output file readout.

Author

Laurent Forthomme `laurent.forthomme@cern.ch`

Date

Jun 2015

7.7.2 Constructor & Destructor Documentation

7.7.2.1 FileReader::FileReader ( ) `[inline]`

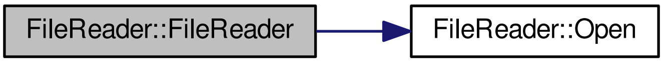
7.7.2.2 FileReader::FileReader (std::string *name*)

Class constructor.

Parameters

`in`	*name*	Path to the file to read
`in`	*ro*	Data readout mode (continuous storage or trigger matching)

Here is the call graph for this function:



7.7.2.3 FileReader::∼FileReader ( )

7.7.3 Member Function Documentation

7.7.3.1 void FileReader::Clear ( ) [inline]

7.7.3.2 void FileReader::Dump ( ) const

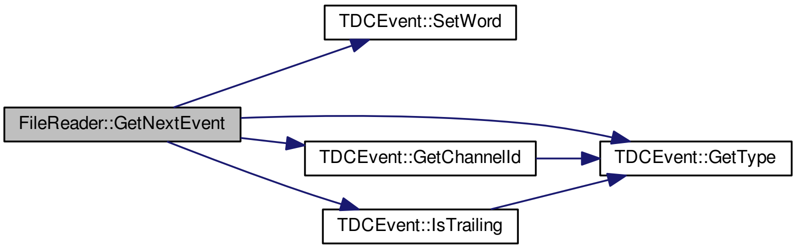
7.7.3.3 unsigned int FileReader::GetAcquisitionMode ( ) const [inline]

7.7.3.4 unsigned int FileReader::GetBurstId ( ) const [inline]

7.7.3.5 unsigned int FileReader::GetDetectionMode ( ) const [inline]

7.7.3.6 bool FileReader::GetNextEvent (TDCEvent ∗ *ev*)

Here is the call graph for this function:



7.7.3.7 bool FileReader::GetNextMeasurement (unsigned int *channel_id,* TDCMeasurement ∗ *mc*)

Fetch the next full measurement on a given channel.

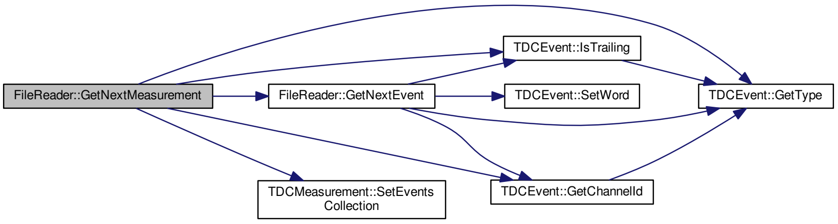
Parameters

in	*channel_id*	Unique identifier of the channel number to retrieve
out	*m*	A full measurement with leading, trailing times, ...

Returns

A boolean stating the success of retrieval operation

Here is the call graph for this function:



7.7.3.8 unsigned long FileReader::GetNumEvents ( ) const [inline]

7.7.3.9 unsigned int FileReader::GetNumTDCs ( ) const [inline]

7.7.3.10 unsigned int FileReader::GetRunId ( ) const [inline]

7.7.3.11 bool FileReader::IsOpen ( ) const [inline]

7.7.3.12 void FileReader::Open (std::string *name*)

7.7.4 Field Documentation

7.7.4.1 std::ifstream FileReader::fFile [private]

7.7.4.2 file_header_t FileReader::fHeader [private]

7.7.4.3 unsigned long FileReader::fNumEvents [private]

7.7.4.4 AcquisitionMode FileReader::fReadoutMode [private]

7.7.4.5 time_t FileReader::fWriteTime [private]

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

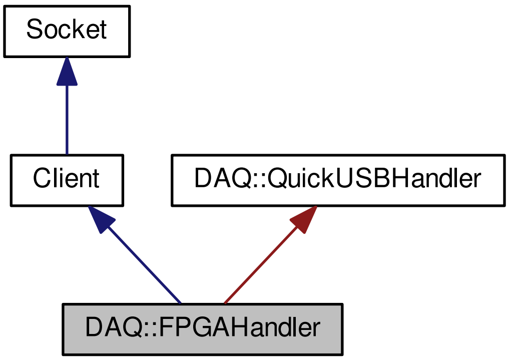
- include/FileReader.h
- src/FileReader.cpp

7.8 DAQ::FPGAHandler Class Reference

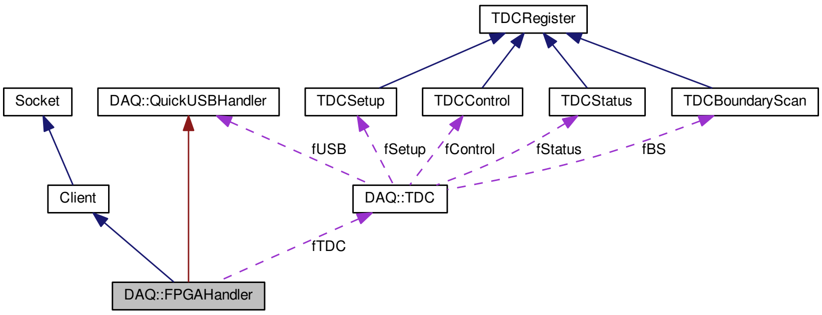
Driver for timing detectors' FPGA readout.

`#include <FPGAHandler.h>`

Inheritance diagram for DAQ::FPGAHandler:



Collaboration diagram for DAQ::FPGAHandler:



Public Member Functions

- FPGAHandler (int port, const char ∗dev)

  Bind to a FPGA through the USB protocol, and to the socket.
- ∼FPGAHandler ()
- void Stop ()
- void OpenFile ()

  Open an output file to store header/HPTDC events.
- void CloseFile ()

  Close a previously opened output file used to store header/HPTDC events.
- std::string GetFilename () const

  Retrieve the file name used to store data collected from the FPGA.
- TDC ∗ GetTDC (unsigned int i=0)
- void SetTDCSetup (const TDCSetup &s)
- bool ErrorState ()
- void StartAcquisition ()
- void StopAcquisition ()
- SocketType GetType () const

  Socket actor type retrieval method.
- TDCControl GetTDCControl () const
- TDCStatus GetTDCStatus () const

Private Member Functions

- void RegisterTest () const
- void SendSetupWord () const
- void RetrieveSetupWord () const

Private Attributes

- std::string fFilename
- std::ofstream fOutput
- bool fIsFileOpen
- TDC ∗ fTDC [NUM_HPTDC]
- bool fIsTDCInReadout

Additional Inherited Members

7.8.1 Detailed Description

Driver for timing detectors' FPGA readout.

Main driver for a homebrew FPGA designed for the timing detectors' HPTDC chip readout.

Author

Laurent Forthomme laurent.forthomme@cern.ch

Date

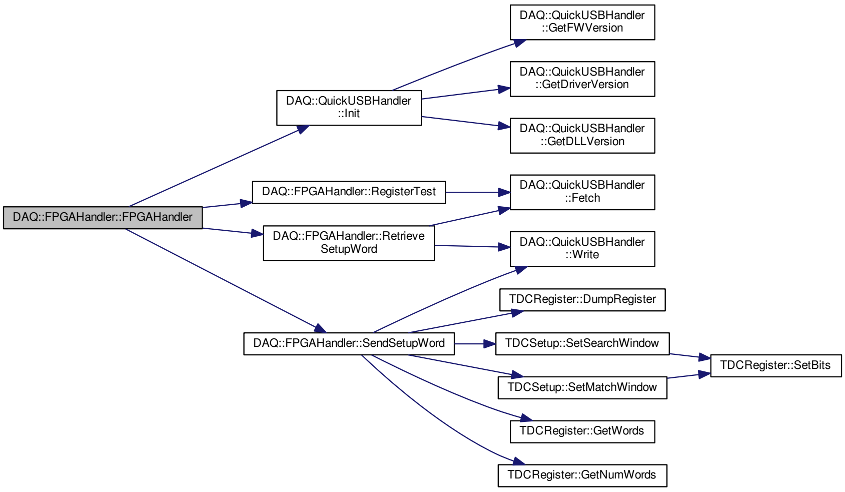
14 Apr 2015

7.8.2 Constructor & Destructor Documentation

7.8.2.1 DAQ::FPGAHandler::FPGAHandler (int *port,* const char ∗ *dev*)

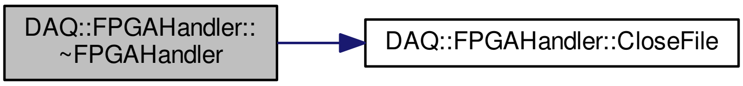
Bind to a FPGA through the USB protocol, and to the socket.

Here is the call graph for this function:


DAQ::QuickUSBHandler ::GetFWVersion
DAQ::QuickUSBHandler ::GetDriverVersion
DAQ::QuickUSBHandler ::Init
DAQ::QuickUSBHandler ::GetDLLVersion
DAQ::FPGAHandler::RegisterTest
DAQ::QuickUSBHandler ::Fetch
DAQ::FPGAHandler::FPGAHandler
DAQ::FPGAHandler::Retrieve SetupWord
DAQ::QuickUSBHandler ::Write
TDCRegister::DumpRegister
DAQ::FPGAHandler::SendSetupWord
TDCSetup::SetSearchWindow
TDCRegister::SetBits
TDCSetup::SetMatchWindow
TDCRegister::GetWords
TDCRegister::GetNumWords

7.8.2.2 DAQ::FPGAHandler::∼FPGAHandler ( )

Here is the call graph for this function:



7.8.3 Member Function Documentation

7.8.3.1 void DAQ::FPGAHandler::CloseFile ( )

Close a previously opened output file used to store header/HPTDC events.

7.8.3.2 bool DAQ::FPGAHandler::ErrorState ( )

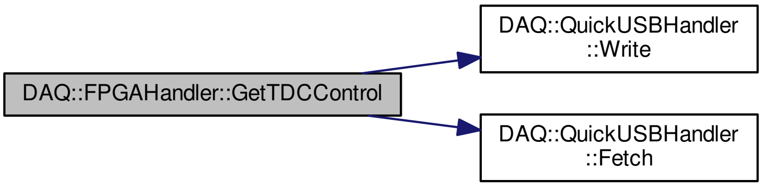
7.8.3.3 std::string DAQ::FPGAHandler::GetFilename ( ) const [inline]

Retrieve the file name used to store data collected from the FPGA.

7.8.3.4 TDC* DAQ::FPGAHandler::GetTDC (unsigned int *i* = 0) [inline]

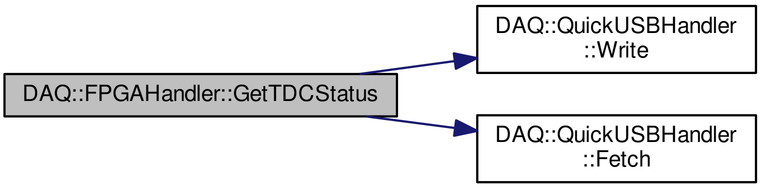
7.8.3.5 TDCControl DAQ::FPGAHandler::GetTDCControl ( ) const

Here is the call graph for this function:


DAQ::FPGAHandler::GetTDCControl
DAQ::QuickUSBHandler ::Write
DAQ::QuickUSBHandler ::Fetch

7.8.3.6 TDCStatus DAQ::FPGAHandler::GetTDCStatus ( ) const

Here is the call graph for this function:


DAQ::FPGAHandler::GetTDCStatus
DAQ::QuickUSBHandler ::Write
DAQ::QuickUSBHandler ::Fetch

7.8.3.7 SocketType DAQ::FPGAHandler::GetType ( ) const [inline], [virtual]

Socket actor type retrieval method.

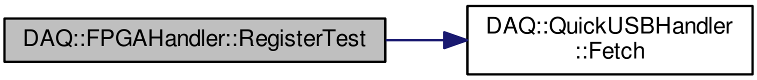
Reimplemented from Client.

7.8.3.8 void DAQ::FPGAHandler::OpenFile ( )

Open an output file to store header/HPTDC events.

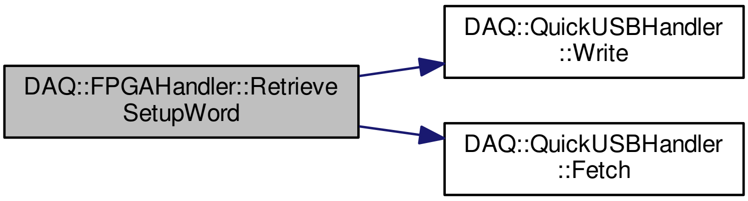
7.8.3.9 void DAQ::FPGAHandler::RegisterTest ( ) const [private]

Here is the call graph for this function:



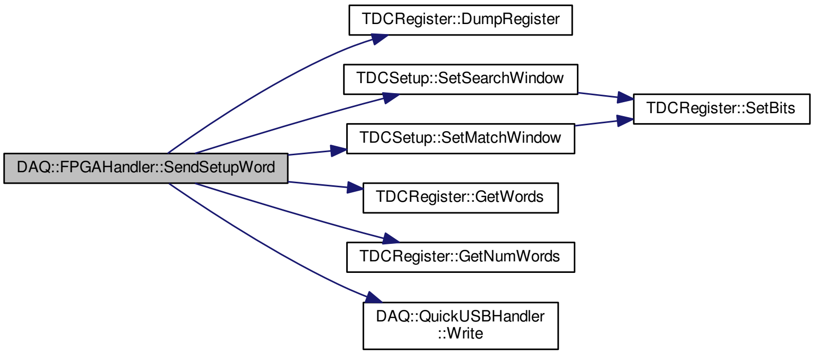
7.8.3.10 void DAQ::FPGAHandler::RetrieveSetupWord ( ) const [private]

Here is the call graph for this function:


DAQ::FPGAHandler::Retrieve SetupWord
DAQ::QuickUSBHandler ::Write
DAQ::QuickUSBHandler ::Fetch

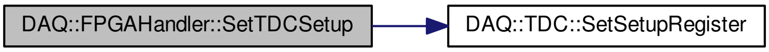
7.8.3.11 void DAQ::FPGAHandler::SendSetupWord ( ) const [private]

Here is the call graph for this function:


TDCRegister::DumpRegister
TDCSetup::SetSearchWindow
TDCRegister::SetBits
TDCSetup::SetMatchWindow
DAQ::FPGAHandler::SendSetupWord
TDCRegister::GetWords
TDCRegister::GetNumWords
DAQ::QuickUSBHandler ::Write

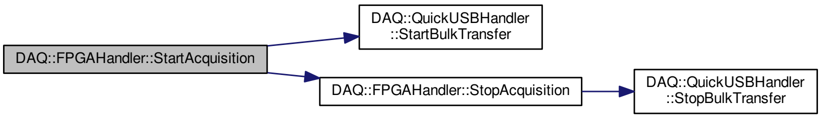
7.8.3.12 void DAQ::FPGAHandler::SetTDCSetup (const TDCSetup & *s*) [inline]

Here is the call graph for this function:



7.8.3.13 void DAQ::FPGAHandler::StartAcquisition ( )

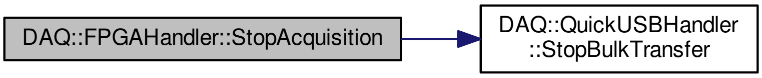
Here is the call graph for this function:


DAQ::QuickUSBHandler ::StartBulkTransfer
DAQ::FPGAHandler::StartAcquisition
DAQ::FPGAHandler::StopAcquisition
DAQ::QuickUSBHandler ::StopBulkTransfer

7.8.3.14 void DAQ::FPGAHandler::Stop ( ) [inline]

7.8.3.15 void DAQ::FPGAHandler::StopAcquisition ( )

Here is the call graph for this function:


DAQ::FPGAHandler::StopAcquisition
DAQ::QuickUSBHandler ::StopBulkTransfer

7.8.4 Field Documentation

7.8.4.1 std::string DAQ::FPGAHandler::fFilename `[private]`

7.8.4.2 bool DAQ::FPGAHandler::fIsFileOpen `[private]`

7.8.4.3 bool DAQ::FPGAHandler::fIsTDCInReadout `[private]`

7.8.4.4 std::ofstream DAQ::FPGAHandler::fOutput `[private]`

7.8.4.5 TDC∗ DAQ::FPGAHandler::fTDC[NUM_HPTDC] `[private]`

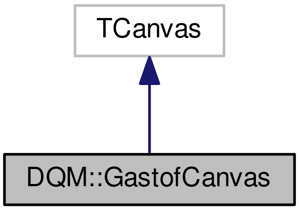
The documentation for this class was generated from the following files:

- daq/include/FPGAHandler.h
- daq/src/FPGAHandler.cpp

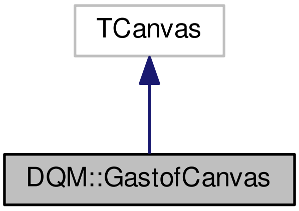
7.9 DQM::GastofCanvas Class Reference

`#include <GastofCanvas.h>`

Inheritance diagram for DQM::GastofCanvas:



Collaboration diagram for DQM::GastofCanvas:



Data Structures

- struct Coord

Public Member Functions

- GastofCanvas ()
- GastofCanvas (TString name, unsigned int width=500, unsigned int height=500, TString upper_label="")
- GastofCanvas (TString name, TString upper_label)
- virtual ∼GastofCanvas ()
- void SetRunInfo (unsigned int board_id, unsigned int run_id, unsigned int spill_id, TString date)
- void SetUpperLabel (TString text)
- void FillChannel (unsigned short nino_id, unsigned short channel_id, double content)
- TH2D ∗ Grid ()
- void Save (TString ext="png", TString path=".")

Private Member Functions

- void Build ()
- void DrawGrid ()
- Coord GetCoordinates (unsigned short nino_id, unsigned short channel_id) const

Private Attributes

- TPad ∗ c1
- TPad ∗ c2
- TH2D ∗ fHist
- double fWidth
- double fHeight
- TLegend ∗ fLegend
- double fLegendX
- double fLegendY
- unsigned int fLegendNumEntries
- TPaveText ∗ fLabel1
- TPaveText ∗ fLabel2
- TPaveText ∗ fLabel3

- TPaveText ∗ fLabel4
- TString fUpperLabelText
- TPaveText ∗ fUpperLabel
- bool fLabelsDrawn
- unsigned int fBoardId
- unsigned int fRunId
- unsigned int fSpillId
- TString fRunDate

7.9.1 Detailed Description

Author

Laurent Forthomme `laurent.forthomme@cern.ch`

Date

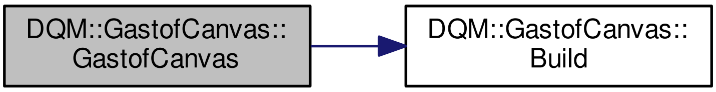
25 Jul 2015

7.9.2 Constructor & Destructor Documentation

7.9.2.1 DQM::GastofCanvas::GastofCanvas ( ) `[inline]`

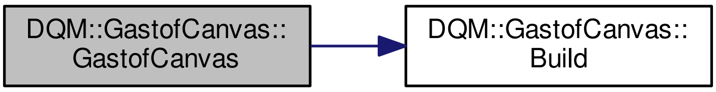
7.9.2.2 DQM::GastofCanvas::GastofCanvas (TString *name*, unsigned int *width* = `500`, unsigned int *height* = `500`, TString *upper_label* = `""`) `[inline]`

Here is the call graph for this function:



7.9.2.3 DQM::GastofCanvas::GastofCanvas (TString *name*, TString *upper_label*) `[inline]`

Here is the call graph for this function:



7.9.2.4 virtual DQM::GastofCanvas::∼GastofCanvas ( ) [inline], [virtual]

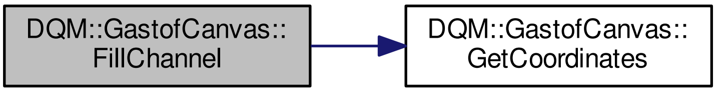
7.9.3 Member Function Documentation

7.9.3.1 void DQM::GastofCanvas::Build ( ) [inline], [private]

7.9.3.2 void DQM::GastofCanvas::DrawGrid ( ) [inline], [private]

7.9.3.3 void DQM::GastofCanvas::FillChannel (unsigned short *nino_id*, unsigned short *channel_id*, double *content*) [inline]

Here is the call graph for this function:

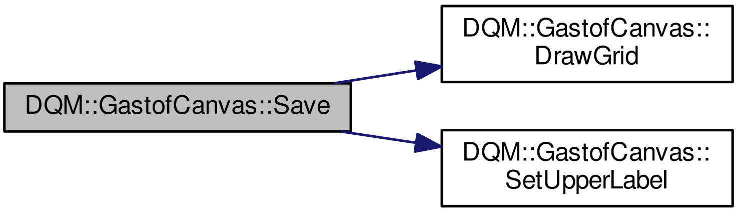


7.9.3.4 Coord DQM::GastofCanvas::GetCoordinates (unsigned short *nino_id*, unsigned short *channel_id*) const [inline], [private]

7.9.3.5 TH2D∗ DQM::GastofCanvas::Grid ( ) [inline]

7.9.3.6 void DQM::GastofCanvas::Save (TString *ext* = "png", TString *path* = ".") [inline]

Here is the call graph for this function:



7.9.3.7 void DQM::GastofCanvas::SetRunInfo (unsigned int *board_id*, unsigned int *run_id*, unsigned int *spill_id*, TString *date*) [inline]

7.9.3.8 void DQM::GastofCanvas::SetUpperLabel (TString *text*) [inline]

7.9.4 Field Documentation

7.9.4.1 TPad∗ DQM::GastofCanvas::c1 [private]

7.9.4.2 TPad ∗ DQM::GastofCanvas::c2 [private]

7.9.4.3 unsigned int DQM::GastofCanvas::fBoardId [private]

7.9.4.4 double DQM::GastofCanvas::fHeight [private]

7.9.4.5 TH2D∗ DQM::GastofCanvas::fHist [private]

7.9.4.6 TPaveText∗ DQM::GastofCanvas::fLabel1 [private]

7.9.4.7 TPaveText ∗ DQM::GastofCanvas::fLabel2 [private]

7.9.4.8 TPaveText ∗ DQM::GastofCanvas::fLabel3 [private]

7.9.4.9 TPaveText ∗ DQM::GastofCanvas::fLabel4 [private]

7.9.4.10 bool DQM::GastofCanvas::fLabelsDrawn [private]

7.9.4.11 TLegend∗ DQM::GastofCanvas::fLegend [private]

7.9.4.12 unsigned int DQM::GastofCanvas::fLegendNumEntries [private]

7.9.4.13 double DQM::GastofCanvas::fLegendX [private]

7.9.4.14 double DQM::GastofCanvas::fLegendY [private]

7.9.4.15 TString DQM::GastofCanvas::fRunDate [private]

7.9.4.16 unsigned int DQM::GastofCanvas::fRunId [private]

7.9.4.17 unsigned int DQM::GastofCanvas::fSpillId [private]

7.9.4.18 TPaveText∗ DQM::GastofCanvas::fUpperLabel [private]

7.9.4.19 TString DQM::GastofCanvas::fUpperLabelText [private]

7.9.4.20 double DQM::GastofCanvas::fWidth [private]

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

- include/GastofCanvas.h

7.10 Logger Class Reference

Redirect outputs to another output stream.

`#include <FileConstants.h>`

Public Member Functions

- Logger (std::ostream &lhs, std::ostream &rhs=std::cout)
- ∼Logger ()

Private Attributes

- std::ostream & fStream
- std::streambuf ∗const fBuffer

7.10.1 Detailed Description

Redirect outputs to another output stream.

7.10.2 Constructor & Destructor Documentation

7.10.2.1 Logger::Logger (std::ostream & *lhs*, std::ostream & *rhs* = `std::cout`) `[inline]`

7.10.2.2 Logger::∼Logger ( ) `[inline]`

7.10.3 Field Documentation

7.10.3.1 std::streambuf∗ const Logger::fBuffer `[private]`

7.10.3.2 std::ostream& Logger::fStream `[private]`

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

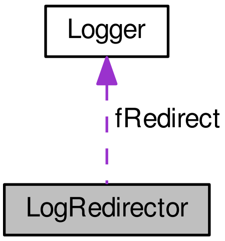
- include/FileConstants.h

7.11 LogRedirector Class Reference

Redirect output stream to a string.

`#include <FileConstants.h>`

Collaboration diagram for LogRedirector:



Public Member Functions

- LogRedirector (std::ostream &stm=std::cout)
- std::string contents () const

Private Attributes

- std::ostringstream fSS
- const Logger fRedirect

7.11.1 Detailed Description

Redirect output stream to a string.

Author

Laurent Forthomme `laurent.forthomme@cern.ch`

Date

3 Aug 2015

7.11.2 Constructor & Destructor Documentation

7.11.2.1 LogRedirector::LogRedirector (std::ostream & *stm* = `std::cout`) `[inline]`

7.11.3 Member Function Documentation

7.11.3.1 std::string LogRedirector::contents ( ) const `[inline]`

7.11.4 Field Documentation

7.11.4.1 const Logger LogRedirector::fRedirect `[private]`

7.11.4.2 std::ostringstream LogRedirector::fSS `[private]`

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

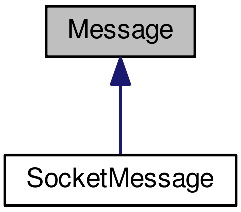
- include/FileConstants.h

7.12 Message Class Reference

Base socket message type.

`#include <Message.h>`

Inheritance diagram for Message:



Public Member Functions

- Message ()

  Void message constructor.
- Message (const char ∗msg)

  Construct a message from a string.
- Message (std::string msg)

  Construct a message from a string.
- virtual ∼Message ()
- MessageKey GetKey () const

  Placeholder for the MessageKey retrieval method.
- std::string GetString () const

  Retrieve the string carried by this message as a whole.
- bool IsFromWeb () const

  Extract from any message its potential arrival from a WebSocket protocol.
- void Dump (std::ostream &os=std::cout) const

Protected Attributes

- std::string fString

7.12.1 Detailed Description

Base socket message type.

Base handler for messages to be transmitted through the socket

Author

Laurent Forthomme `laurent.forthomme@cern.ch`

Date

6 Apr 2015

7.12.2 Constructor & Destructor Documentation

7.12.2.1 Message::Message ( ) `[inline]`

Void message constructor.

7.12.2.2 Message::Message (const char ∗ *msg*) `[inline]`

Construct a message from a string.

7.12.2.3 Message::Message (std::string *msg*) `[inline]`

Construct a message from a string.

7.12.2.4 virtual Message::∼Message ( ) [inline], [virtual]

7.12.3 Member Function Documentation

7.12.3.1 void Message::Dump (std::ostream & *os* = std::cout) const [inline]

7.12.3.2 MessageKey Message::GetKey ( ) const [inline]

Placeholder for the MessageKey retrieval method.

7.12.3.3 std::string Message::GetString ( ) const [inline]

Retrieve the string carried by this message as a whole.

7.12.3.4 bool Message::IsFromWeb ( ) const [inline]

Extract from any message its potential arrival from a WebSocket protocol.

7.12.4 Field Documentation

7.12.4.1 std::string Message::fString [protected]

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

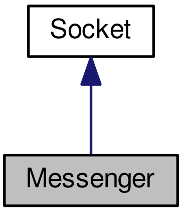
- include/Message.h

7.13 Messenger Class Reference

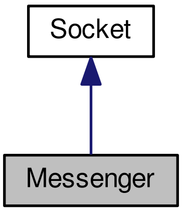
Base master object for the socket.

`#include <Messenger.h>`

Inheritance diagram for Messenger:



Collaboration diagram for Messenger:



Public Member Functions

- Messenger ()

  Build a void master object or socket actor.
- Messenger (int port)

  Build a master object to control the socket.
- ∼Messenger ()
- bool Connect ()

  Connect the master to the socket.
- void Disconnect ()

  Remove the master and destroy the socket.
- void Send (const Message &m, int sid) const

  Send any type of message to any client.
- void SendAll (const Socket::SocketType &type, const Message &m) const

  Send any type of message to all clients of one type.
- void SendAll (const Socket::SocketType &type, const Exception &e) const
- void Receive ()

  Handle a message reception from a client.
- void Broadcast (const Message &m) const

  Emit a message to all clients connected through the socket.
- void StartAcquisition ()

  Start the data acquisition.
- void StopAcquisition ()
- SocketType GetType () const

  Socket actor type retrieval method.

Private Member Functions

- void AddClient ()

  Add a client to listen to.
- void DisconnectClient (int sid, MessageKey key, bool force=false)

  Disconnect a client.
- void SwitchClientType (int sid, Socket::SocketType type)
- void ProcessMessage (SocketMessage m, int sid)

  Process a message received from the socket.

Private Attributes

- int fNumAttempts
- pid_t fPID
- int fStdoutPipe [2]
- int fStderrPipe [2]

Additional Inherited Members

7.13.1 Detailed Description

Base master object for the socket.

Messenger/broadcaster object used by the server to send/receive commands from the clients/listeners.

Author

Laurent Forthomme `laurent.forthomme@cern.ch`

Date

23 Mar 2015

7.13.2 Constructor & Destructor Documentation

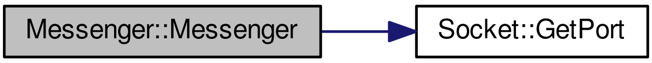
7.13.2.1 Messenger::Messenger ( )

Build a void master object or socket actor.

7.13.2.2 Messenger::Messenger (int *port*)

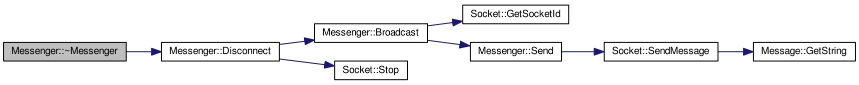
Build a master object to control the socket.

Here is the call graph for this function:



7.13.2.3 Messenger::∼Messenger ( )

Here is the call graph for this function:



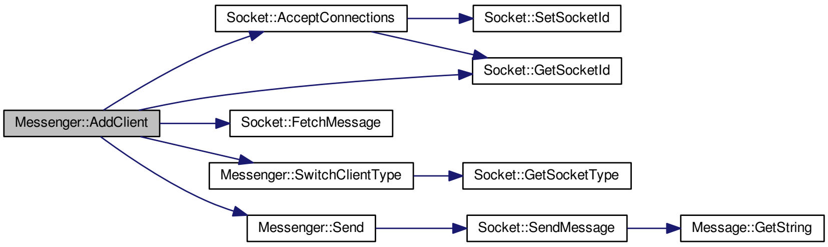
7.13.3 Member Function Documentation

7.13.3.1 void Messenger::AddClient ( ) [private]

Add a client to listen to.

Add one client to the list of socket actors to monitor for message retrieval/submission.

Here is the call graph for this function:



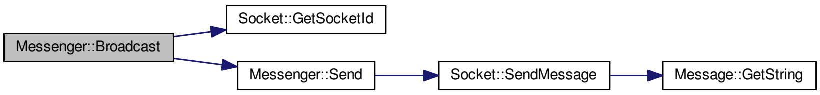
7.13.3.2 void Messenger::Broadcast (const Message & *m*) const

Emit a message to all clients connected through the socket.

Parameters

in	*m*	Message to transmit

Here is the call graph for this function:

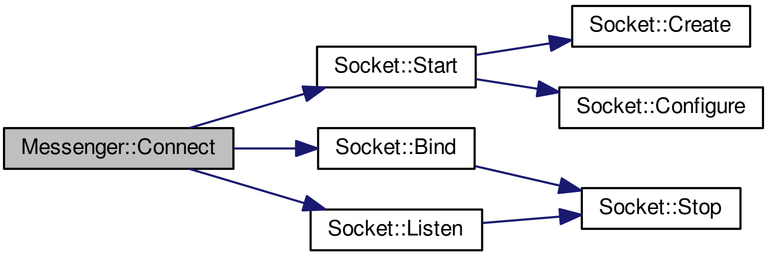


7.13.3.3 bool Messenger::Connect ( )

Connect the master to the socket.

Connect this master to the socket for clients to be able to bind.

Here is the call graph for this function:

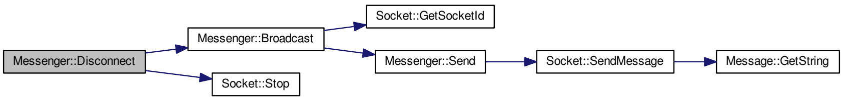


7.13.3.4 void Messenger::Disconnect ( )

Remove the master and destroy the socket.

Remove this master from the socket, thus disconnecting automatically the clients connected.

Here is the call graph for this function:



7.13.3.5 void Messenger::DisconnectClient (int *sid,* MessageKey *key,* bool *force* = `false`) `[private]`

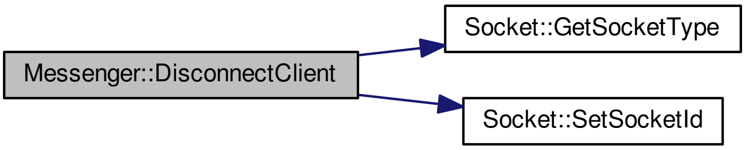
Disconnect a client.

Ask to a client to disconnect from this socket.

Parameters

`in`	*sid*	Unique identifier of the client to disconnect
`in`	*key*	Key to the message to transmit for disconnection
`in`	*force*	Do we need to force the client out of this socket ?

Here is the call graph for this function:



7.13.3.6 SocketType Messenger::GetType ( ) const [inline]

Socket actor type retrieval method.

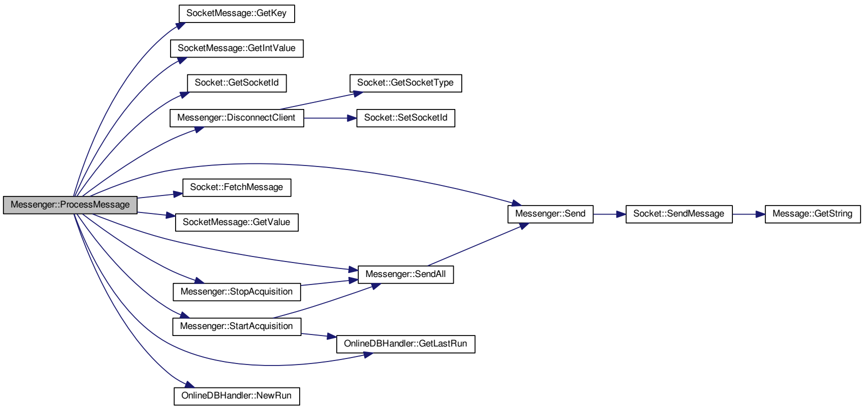
7.13.3.7 void Messenger::ProcessMessage (SocketMessage *m,* int *sid*) [private]

Process a message received from the socket.

Parameters

in	*Unique*	identifier of the client sending the message

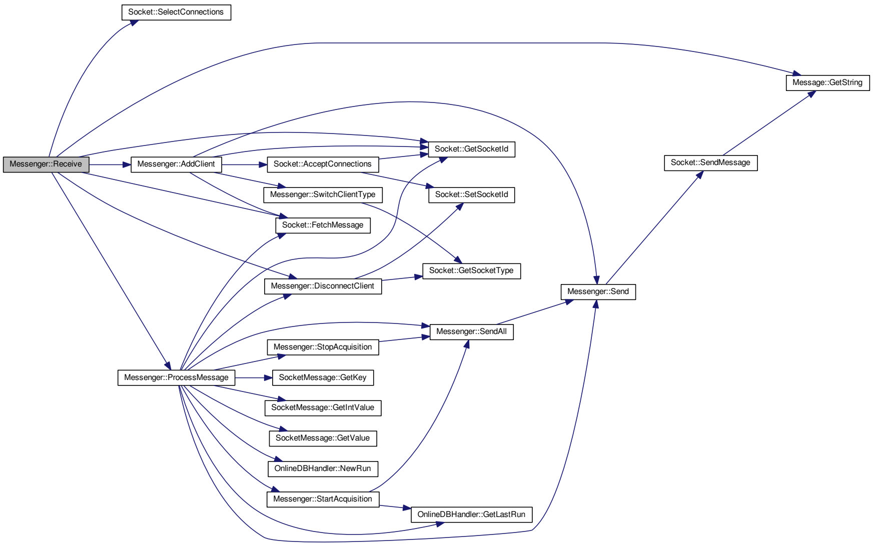
Here is the call graph for this function:



7.13.3.8 void Messenger::Receive ( )

Handle a message reception from a client.

Here is the call graph for this function:



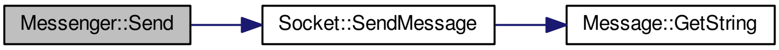
7.13.3.9 void Messenger::Send (const Message & *m,* int *sid*) const

Send any type of message to any client.

Parameters

in	*m*	Message to transmit
in	*sid*	Unique identifier of the client on this socket

Here is the call graph for this function:



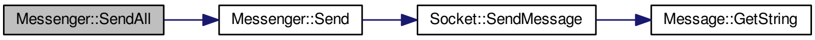
7.13.3.10 void Messenger::SendAll (const Socket::SocketType & *type,* const Message & *m*) const [inline]

Send any type of message to all clients of one type.

Parameters

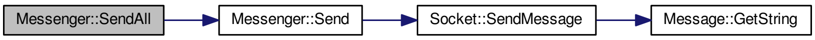
in	*type*	Client type
in	*m*	Message to transmit

Here is the call graph for this function:



7.13.3.11 void Messenger::SendAll (const Socket::SocketType & *type,* const Exception & *e*) const [inline]

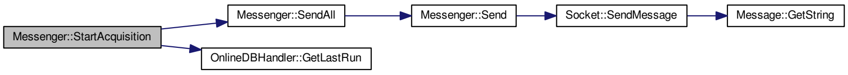
Here is the call graph for this function:



7.13.3.12 void Messenger::StartAcquisition ( )

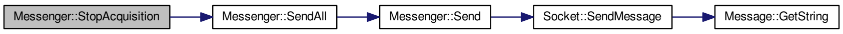
Start the data acquisition.

Here is the call graph for this function:



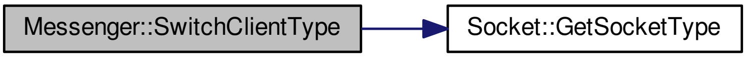
7.13.3.13 void Messenger::StopAcquisition ( )

Here is the call graph for this function:



7.13.3.14 void Messenger::SwitchClientType (int *sid,* Socket::SocketType *type*) `[private]`

Here is the call graph for this function:



7.13.4 Field Documentation

7.13.4.1 int Messenger::fNumAttempts `[private]`

7.13.4.2 pid_t Messenger::fPID `[private]`

7.13.4.3 int Messenger::fStderrPipe[2] `[private]`

7.13.4.4 int Messenger::fStdoutPipe[2] `[private]`

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

- include/Messenger.h
- src/Messenger.cpp

7.14 OnlineDBHandler Class Reference

Handler for the run information online database.

`#include <OnlineDBHandler.h>`

Data Structures

- struct BurstInfo
- struct TDCConditions

Public Types

- typedef std::map< unsigned int, unsigned int > RunCollection
- typedef std::vector< BurstInfo > BurstInfos
- typedef std::vector< TDCConditions > TDCConditionsCollection

Public Member Functions

- OnlineDBHandler (std::string path=std::string(std::getenv("PPS_PATH"))+"/run_infos.db")
- ∼OnlineDBHandler ()
- void NewRun ()
- void NewBurst ()
- RunCollection GetRuns () const
- unsigned int GetLastRun () const

  Retrieve the last run acquired.
- int GetLastBurst (unsigned int run) const
- BurstInfos GetRunInfo (unsigned int run) const

  Retrieve information on a given run (spill IDs / timestamp)
- void SetTDCConditions (unsigned short tdc_id, unsigned long tdc_address, unsigned short tdc_acq_mode, unsigned short tdc_det_mode, std::string detector)
- TDCConditionsCollection GetTDCConditions (unsigned int run_id) const
- void SetHVConditions (unsigned short channel_id, unsigned int vmax, unsigned imax)

Private Member Functions

- void BuildTables ()
- template<class T >
  std::vector< std::vector< T > > Select (std::string req, int num_fields=-1) const

Private Attributes

- sqlite3 ∗ fDB

7.14.1 Detailed Description

Handler for the run information online database.

Author

Laurent Forthomme `laurent.forthomme@cern.ch`

Date

3 Aug 2015

7.14.2 Member Typedef Documentation

7.14.2.1 typedef std::vector< BurstInfo > OnlineDBHandler::BurstInfos

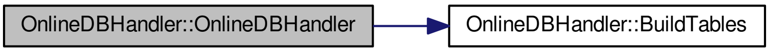
7.14.2.2 typedef std::map< unsigned int, unsigned int > OnlineDBHandler::RunCollection

7.14.2.3 typedef std::vector< TDCConditions > OnlineDBHandler::TDCConditionsCollection

7.14.3 Constructor & Destructor Documentation

7.14.3.1 OnlineDBHandler::OnlineDBHandler (std::string *path* = `std::string(std::getenv("PPS_PATH"))+"/run_infos.db"`) `[inline]`

Here is the call graph for this function:


OnlineDBHandler::OnlineDBHandler
OnlineDBHandler::BuildTables

7.14.3.2 OnlineDBHandler::∼OnlineDBHandler ( ) `[inline]`

7.14.4 Member Function Documentation

7.14.4.1 void OnlineDBHandler::BuildTables ( ) `[inline]`, `[private]`

7.14.4.2 int OnlineDBHandler::GetLastBurst (unsigned int *run*) const `[inline]`

7.14.4.3 unsigned int OnlineDBHandler::GetLastRun ( ) const `[inline]`

Retrieve the last run acquired.

7.14.4.4 BurstInfos OnlineDBHandler::GetRunInfo (unsigned int *run*) const `[inline]`

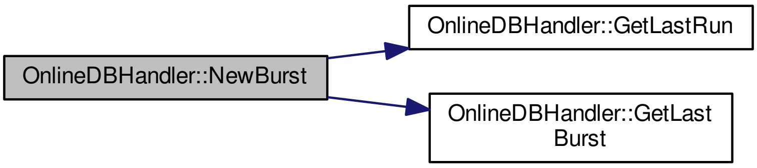
Retrieve information on a given run (spill IDs / timestamp)

7.14.4.5 RunCollection OnlineDBHandler::GetRuns ( ) const `[inline]`

7.14.4.6 TDCConditionsCollection OnlineDBHandler::GetTDCConditions (unsigned int *run_id*) const `[inline]`

7.14.4.7 void OnlineDBHandler::NewBurst ( ) [inline]

Here is the call graph for this function:

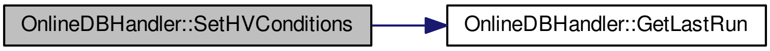

OnlineDBHandler::NewBurst
OnlineDBHandler::GetLastRun
OnlineDBHandler::GetLast Burst

7.14.4.8 void OnlineDBHandler::NewRun ( ) [inline]

7.14.4.9 template< class T > std::vector< std::vector< T > > OnlineDBHandler::Select (std::string *req*, int *num_fields* = -1) const [inline], [private]

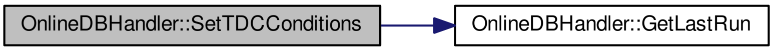
7.14.4.10 void OnlineDBHandler::SetHVConditions (unsigned short *channel_id*, unsigned int *vmax*, unsigned *imax*) [inline]

Here is the call graph for this function:



7.14.4.11 void OnlineDBHandler::SetTDCConditions (unsigned short *tdc_id*, unsigned long *tdc_address*, unsigned short *tdc_acq_mode*, unsigned short *tdc_det_mode*, std::string *detector*) [inline]

Here is the call graph for this function:


OnlineDBHandler::SetTDCConditions
OnlineDBHandler::GetLastRun

7.14.5 Field Documentation

7.14.5.1 sqlite3* OnlineDBHandler::fDB [private]

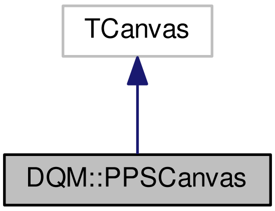
The documentation for this class was generated from the following file:

- include/OnlineDBHandler.h

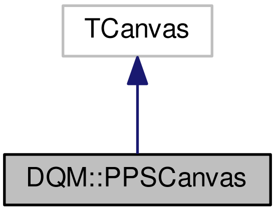
7.15 DQM::PPSCanvas Class Reference

```
#include <PPSCanvas.h>
```

Inheritance diagram for DQM::PPSCanvas:



Collaboration diagram for DQM::PPSCanvas:



Public Member Functions

- PPSCanvas ()
- PPSCanvas (TString name, unsigned int width=500, unsigned int height=500, TString upper_label="")
- PPSCanvas (TString name, TString upper_label)
- virtual ∼PPSCanvas ()
- void SetRunInfo (unsigned int run_id, TString date)
- void SetUpperLabel (TString text)
- TPad ∗ Grid ()
- void Save (TString ext="png", TString path=".")

Private Member Functions

- void Build ()
- void DrawGrid ()

Private Attributes

- TPad ∗ c1
- TPad ∗ c2
- double fWidth
- double fHeight
- TLegend ∗ fLegend
- double fLegendX
- double fLegendY
- unsigned int fLegendNumEntries
- TPaveText ∗ fLabel1
- TPaveText ∗ fLabel2
- TPaveText ∗ fLabel3
- TString fUpperLabelText
- TPaveText ∗ fUpperLabel
- bool fLabelsDrawn
- unsigned int fRunId
- TString fRunDate

7.15.1 Detailed Description

Author

Laurent Forthomme `laurent.forthomme@cern.ch`

Date

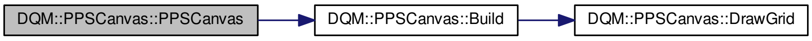
3 Aug 2015

7.15.2 Constructor & Destructor Documentation

7.15.2.1 DQM::PPSCanvas::PPSCanvas ( ) `[inline]`

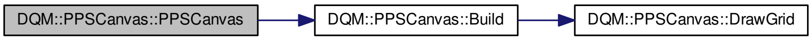
7.15.2.2 DQM::PPSCanvas::PPSCanvas (TString *name*, unsigned int *width* = `500`, unsigned int *height* = `500`, TString *upper_label* = "") `[inline]`

Here is the call graph for this function:



7.15.2.3 DQM::PPSCanvas::PPSCanvas (TString *name*, TString *upper_label*) `[inline]`

Here is the call graph for this function:

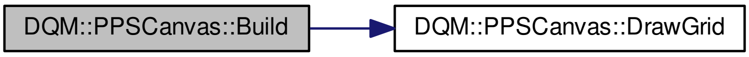


7.15.2.4 virtual DQM::PPSCanvas::∼PPSCanvas ( ) [inline], [virtual]

7.15.3 Member Function Documentation

7.15.3.1 void DQM::PPSCanvas::Build ( ) [inline], [private]

Here is the call graph for this function:

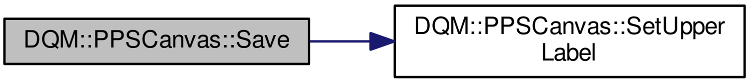


7.15.3.2 void DQM::PPSCanvas::DrawGrid ( ) [inline], [private]

7.15.3.3 TPad∗ DQM::PPSCanvas::Grid ( ) [inline]

7.15.3.4 void DQM::PPSCanvas::Save (TString *ext* = "png", TString *path* = ".") [inline]

Here is the call graph for this function:



7.15.3.5 void DQM::PPSCanvas::SetRunInfo (unsigned int *run_id*, TString *date*) [inline]

7.15.3.6 void DQM::PPSCanvas::SetUpperLabel (TString *text*) [inline]

7.15.4 Field Documentation

7.15.4.1 TPad∗ DQM::PPSCanvas::c1 [private]

7.15.4.2 TPad ∗ DQM::PPSCanvas::c2 [private]

7.15.4.3 double DQM::PPSCanvas::fHeight [private]

7.15.4.4 TPaveText∗ DQM::PPSCanvas::fLabel1 [private]

7.15.4.5 TPaveText ∗ DQM::PPSCanvas::fLabel2 [private]

7.15.4.6 TPaveText ∗ DQM::PPSCanvas::fLabel3 [private]

7.15.4.7 bool DQM::PPSCanvas::fLabelsDrawn [private]

7.15.4.8 TLegend* DQM::PPSCanvas::fLegend [private]

7.15.4.9 unsigned int DQM::PPSCanvas::fLegendNumEntries [private]

7.15.4.10 double DQM::PPSCanvas::fLegendX [private]

7.15.4.11 double DQM::PPSCanvas::fLegendY [private]

7.15.4.12 TString DQM::PPSCanvas::fRunDate [private]

7.15.4.13 unsigned int DQM::PPSCanvas::fRunId [private]

7.15.4.14 TPaveText* DQM::PPSCanvas::fUpperLabel [private]

7.15.4.15 TString DQM::PPSCanvas::fUpperLabelText [private]

7.15.4.16 double DQM::PPSCanvas::fWidth [private]

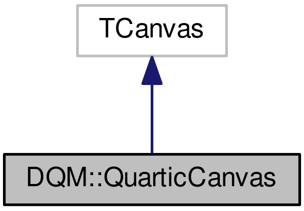
The documentation for this class was generated from the following file:

- include/PPSCanvas.h

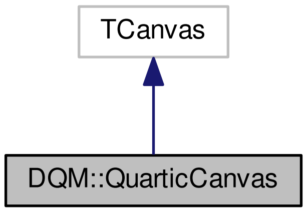
7.16 DQM::QuarticCanvas Class Reference

`#include <QuarticCanvas.h>`

Inheritance diagram for DQM::QuarticCanvas:



Collaboration diagram for DQM::QuarticCanvas:



Data Structures

- struct Coord

Public Member Functions

- QuarticCanvas ()
- QuarticCanvas (TString name, unsigned int width=500, unsigned int height=500, TString upper_label="")
- QuarticCanvas (TString name, TString upper_label)
- virtual ∼QuarticCanvas ()
- void SetRunInfo (unsigned int board_id, unsigned int run_id, unsigned int spill_id, TString date)
- void SetUpperLabel (TString text)
- void FillChannel (unsigned short channel_id, double content)
- TH2D ∗ Grid ()
- void Save (TString ext="png", TString path=".")

Private Member Functions

- void Build ()
- void DrawGrid ()
- Coord GetCoordinates (unsigned short channel_id) const

Private Attributes

- TPad ∗ c1
- TPad ∗ c2
- TH2D ∗ fHist
- double fWidth
- double fHeight
- TLegend ∗ fLegend
- double fLegendX
- double fLegendY
- unsigned int fLegendNumEntries
- TPaveText ∗ fLabel1
- TPaveText ∗ fLabel2
- TPaveText ∗ fLabel3

- TPaveText ∗ fLabel4
- TString fUpperLabelText
- TPaveText ∗ fUpperLabel
- bool fLabelsDrawn
- unsigned int fBoardId
- unsigned int fRunId
- unsigned int fSpillId
- TString fRunDate

7.16.1 Detailed Description

Author

Laurent Forthomme laurent.forthomme@cern.ch

Date

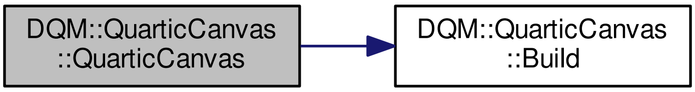
3 Aug 2015

7.16.2 Constructor & Destructor Documentation

7.16.2.1 DQM::QuarticCanvas::QuarticCanvas ( ) [inline]

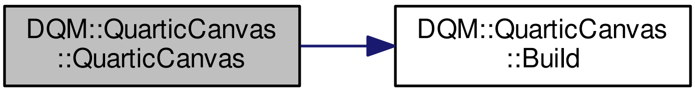
7.16.2.2 DQM::QuarticCanvas::QuarticCanvas (TString *name*, unsigned int *width* = 500, unsigned int *height* = 500, TString *upper_label* = "") [inline]

Here is the call graph for this function:


DQM::QuarticCanvas ::QuarticCanvas
DQM::QuarticCanvas ::Build

7.16.2.3 DQM::QuarticCanvas::QuarticCanvas (TString *name*, TString *upper_label*) [inline]

Here is the call graph for this function:


DQM::QuarticCanvas ::QuarticCanvas
DQM::QuarticCanvas ::Build

7.16.2.4 virtual DQM::QuarticCanvas::∼QuarticCanvas ( ) [inline], [virtual]

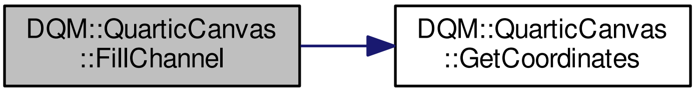
7.16.3 Member Function Documentation

7.16.3.1 void DQM::QuarticCanvas::Build ( ) [inline], [private]

7.16.3.2 void DQM::QuarticCanvas::DrawGrid ( ) [inline], [private]

7.16.3.3 void DQM::QuarticCanvas::FillChannel (unsigned short *channel_id,* double *content*) [inline]

Here is the call graph for this function:

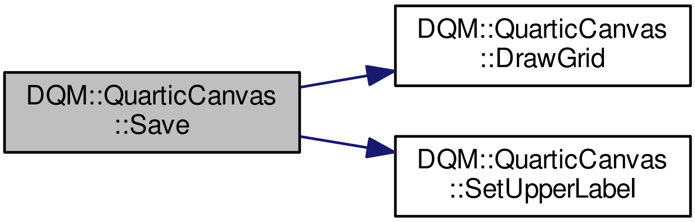

DQM::QuarticCanvas ::FillChannel
DQM::QuarticCanvas ::GetCoordinates

7.16.3.4 Coord DQM::QuarticCanvas::GetCoordinates (unsigned short *channel_id*) const [inline], [private]

7.16.3.5 TH2D∗ DQM::QuarticCanvas::Grid ( ) [inline]

7.16.3.6 void DQM::QuarticCanvas::Save (TString *ext* = "png", TString *path* = ".") [inline]

Here is the call graph for this function:


DQM::QuarticCanvas ::Save
DQM::QuarticCanvas ::DrawGrid
DQM::QuarticCanvas ::SetUpperLabel

7.16.3.7 void DQM::QuarticCanvas::SetRunInfo (unsigned int *board_id,* unsigned int *run_id,* unsigned int *spill_id,* TString *date*) [inline]

7.16.3.8 void DQM::QuarticCanvas::SetUpperLabel (TString *text*) [inline]

7.16.4 Field Documentation

7.16.4.1 TPad∗ DQM::QuarticCanvas::c1 [private]

7.16.4.2 TPad ∗ DQM::QuarticCanvas::c2 [private]

7.16.4.3 unsigned int DQM::QuarticCanvas::fBoardId [private]

7.16.4.4 double DQM::QuarticCanvas::fHeight [private]

7.16.4.5 TH2D∗ DQM::QuarticCanvas::fHist [private]

7.16.4.6 TPaveText∗ DQM::QuarticCanvas::fLabel1 [private]

7.16.4.7 TPaveText ∗ DQM::QuarticCanvas::fLabel2 [private]

7.16.4.8 TPaveText ∗ DQM::QuarticCanvas::fLabel3 [private]

7.16.4.9 TPaveText ∗ DQM::QuarticCanvas::fLabel4 [private]

7.16.4.10 bool DQM::QuarticCanvas::fLabelsDrawn [private]

7.16.4.11 TLegend∗ DQM::QuarticCanvas::fLegend [private]

7.16.4.12 unsigned int DQM::QuarticCanvas::fLegendNumEntries [private]

7.16.4.13 double DQM::QuarticCanvas::fLegendX [private]

7.16.4.14 double DQM::QuarticCanvas::fLegendY [private]

7.16.4.15 TString DQM::QuarticCanvas::fRunDate [private]

7.16.4.16 unsigned int DQM::QuarticCanvas::fRunId [private]

7.16.4.17 unsigned int DQM::QuarticCanvas::fSpillId [private]

7.16.4.18 TPaveText∗ DQM::QuarticCanvas::fUpperLabel [private]

7.16.4.19 TString DQM::QuarticCanvas::fUpperLabelText [private]

7.16.4.20 double DQM::QuarticCanvas::fWidth [private]

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

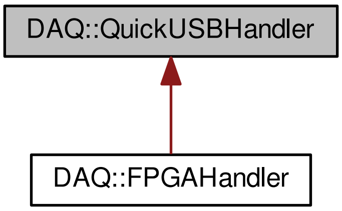
- include/QuarticCanvas.h

7.17 DAQ::QuickUSBHandler Class Reference

Generic QuickUSB communication handler.

```
#include <QuickUSBHandler.h>
```

Inheritance diagram for DAQ::QuickUSBHandler:



Data Structures

- struct Version

Public Member Functions

- QuickUSBHandler ()
- virtual ∼QuickUSBHandler ()
- void Init ()
- void Reset () const
- Version GetFWVersion () const
- Version GetDriverVersion () const
- Version GetDLLVersion () const
- void Write (uint16_t addr, uint8_t word) const

  Write a single word to the QuickUSB device.
- void Write (uint16_t addr, std::vector< uint8_t > &words, uint16_t size) const

  Write a set of words to the QuickUSB device.
- std::vector< uint8_t > Fetch (uint16_t addr, uint16_t size) const

  Receive a set of words from the QuickUSB device.
- void StartBulkTransfer (QVOIDRETURN callback(PQBULKSTREAM))
- void StopBulkTransfer ()

Protected Attributes

- bool fIsStopping

Private Attributes

- std::string fDevice
- QHANDLE fHandle
- uint8_t fStreamId

7.17.1 Detailed Description

Generic QuickUSB communication handler.

Date

17 May 2016

Author

Laurent Forthomme laurent.forthomme@cern.ch

7.17.2 Constructor & Destructor Documentation

7.17.2.1 DAQ::QuickUSBHandler::QuickUSBHandler ( )

7.17.2.2 DAQ::QuickUSBHandler::∼QuickUSBHandler ( ) [virtual]

7.17.3 Member Function Documentation

7.17.3.1 std::vector< uint8_t > DAQ::QuickUSBHandler::Fetch (uint16_t *addr,* uint16_t *size*) const

Receive a set of words from the QuickUSB device.

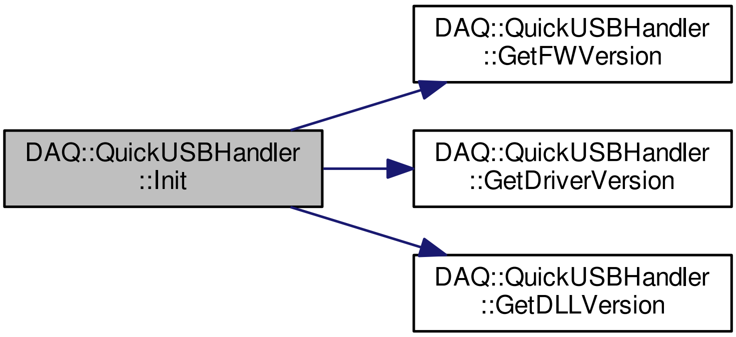
7.17.3.2 QuickUSBHandler::Version DAQ::QuickUSBHandler::GetDLLVersion ( ) const

7.17.3.3 QuickUSBHandler::Version DAQ::QuickUSBHandler::GetDriverVersion ( ) const

7.17.3.4 QuickUSBHandler::Version DAQ::QuickUSBHandler::GetFWVersion ( ) const

7.17.3.5 void DAQ::QuickUSBHandler::Init ( )

Here is the call graph for this function:


DAQ::QuickUSBHandler ::GetFWVersion
DAQ::QuickUSBHandler ::Init
DAQ::QuickUSBHandler ::GetDriverVersion
DAQ::QuickUSBHandler ::GetDLLVersion

7.17.3.6 void DAQ::QuickUSBHandler::Reset ( ) const

7.17.3.7 void DAQ::QuickUSBHandler::StartBulkTransfer (QVOIDRETURN *callbackPQBULKSTREAM*)

7.17.3.8 void DAQ::QuickUSBHandler::StopBulkTransfer ( )

7.17.3.9 void DAQ::QuickUSBHandler::Write (uint16_t *addr,* uint8_t *word*) const `[inline]`

Write a single word to the QuickUSB device.

7.17.3.10 void DAQ::QuickUSBHandler::Write (uint16_t *addr,* std::vector< uint8_t > & *words,* uint16_t *size*) const

Write a set of words to the QuickUSB device.

7.17.4 Field Documentation

7.17.4.1 std::string DAQ::QuickUSBHandler::fDevice `[private]`

7.17.4.2 QHANDLE DAQ::QuickUSBHandler::fHandle `[private]`

7.17.4.3 bool DAQ::QuickUSBHandler::fIsStopping `[protected]`

7.17.4.4 uint8_t DAQ::QuickUSBHandler::fStreamId `[private]`

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

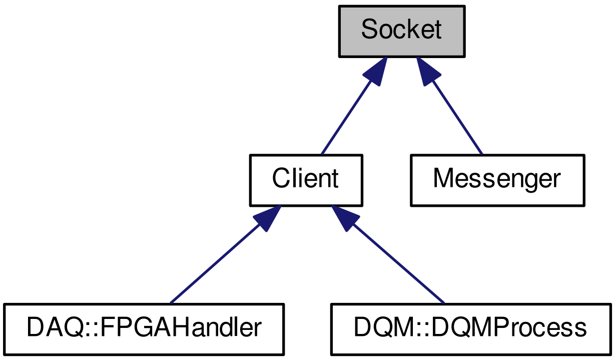
- daq/include/QuickUSBHandler.h
- daq/src/QuickUSBHandler.cpp

7.18 Socket Class Reference

Base socket object from which clients/master from a socket inherit.

`#include <Socket.h>`

Inheritance diagram for Socket:



Public Types

- enum SocketType {
  INVALID =-1, MASTER =0, WEBSOCKET_CLIENT, CLIENT,
  DETECTOR, DQM, DAQ }

    Type of actor playing a role on the socket.
- typedef std::set< std::pair< int, SocketType > > SocketCollection

Public Member Functions

- Socket ()
- Socket (int port)
- virtual ~Socket ()
- void Stop ()

    Terminates the socket and all attached communications.
- void SetPort (int port)
- int GetPort () const

    Retrieve the port used for this socket.
- void AcceptConnections (Socket &socket)

    Accept connection from a client.
- void SelectConnections ()
- void SetSocketId (int sid)
- int GetSocketId () const
- SocketType GetSocketType (int sid) const
- bool IsWebSocket (int sid) const
- void DumpConnected () const

Protected Member Functions

- bool Start ()

    Start the socket.
- void Bind ()

    Bind a name to a socket.
- void PrepareConnection ()
- void Listen (int maxconn)

    Listen to incoming messages.
- void SendMessage (Message message, int id=-1) const

    Send a message on a socket.
- Message FetchMessage (int id=-1) const

    Receive a message from a socket.

Protected Attributes

- int fPort
- char fBuffer [MAX_WORD_LENGTH]
- SocketCollection fSocketsConnected
- fd_set fMaster

    Master file descriptor list.
- fd_set fReadFds

    Temp file descriptor list for select()

Private Member Functions

- void Create ()

  Create an endpoint for communication.
- void Configure ()

  Configure the socket object for communication.

Private Attributes

- int fSocketId
- struct sockaddr_in fAddress

7.18.1 Detailed Description

Base socket object from which clients/master from a socket inherit.

General object providing all useful method to connect/bind/send/receive information through system sockets.

Author

Laurent Forthomme `laurent.forthomme@cern.ch`

Date

23 Mar 2015

7.18.2 Member Typedef Documentation

7.18.2.1 typedef std::set< std::pair<int,SocketType> > **Socket::SocketCollection**

7.18.3 Member Enumeration Documentation

7.18.3.1 enum **Socket::SocketType**

Type of actor playing a role on the socket.

Enumerator

INVALID

MASTER

WEBSOCKET_CLIENT

CLIENT

DETECTOR

DQM

DAQ

7.18.4 Constructor & Destructor Documentation

7.18.4.1 Socket::Socket ( ) `[inline]`

7.18.4.2 Socket::Socket (int *port*)

7.18.4.3 Socket::∼Socket ( ) `[virtual]`

7.18.5 Member Function Documentation

7.18.5.1 void Socket::AcceptConnections (Socket & *socket*)

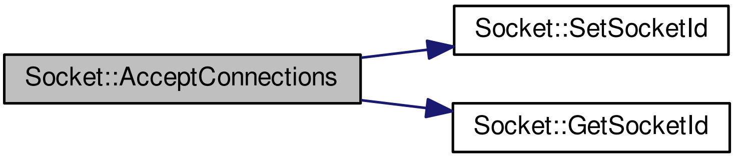
Accept connection from a client.

Set the socket to accept connections any client transmitting through the socket

Parameters

`in,out`	*socket*	Master/client object to enable on the socket

Here is the call graph for this function:



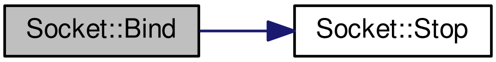
7.18.5.2 void Socket::Bind ( ) `[protected]`

Bind a name to a socket.

Returns

Success of the operation

Here is the call graph for this function:



7.18.5.3 void Socket::Configure ( ) `[private]`

Configure the socket object for communication.

7.18.5.4 void Socket::Create ( ) `[private]`

Create an endpoint for communication.

7.18.5.5 void Socket::DumpConnected ( ) const

7.18.5.6 Message Socket::FetchMessage (int *id* = `-1`) const `[protected]`

Receive a message from a socket.

Returns

Received message as a std::string

7.18.5.7 int Socket::GetPort ( ) const `[inline]`

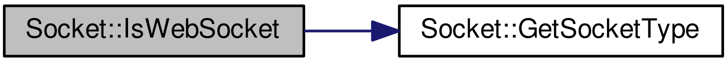
Retrieve the port used for this socket.

7.18.5.8 int Socket::GetSocketId ( ) const `[inline]`

7.18.5.9 SocketType Socket::GetSocketType (int *sid*) const `[inline]`

7.18.5.10 bool Socket::IsWebSocket (int *sid*) const `[inline]`

Here is the call graph for this function:

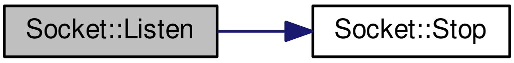


7.18.5.11 void Socket::Listen (int *maxconn*) `[protected]`

Listen to incoming messages.

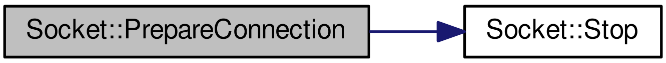
Set the socket to listen to any message coming from outside

Here is the call graph for this function:



7.18.5.12 void Socket::PrepareConnection ( ) [protected]

Here is the call graph for this function:



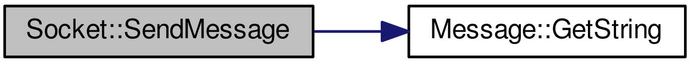
7.18.5.13 void Socket::SelectConnections ( )

Register all open file descriptors to read their communication through the socket

7.18.5.14 void Socket::SendMessage (Message *message*, int *id* = -1) const [protected]

Send a message on a socket.

Here is the call graph for this function:



7.18.5.15 void Socket::SetPort (int *port*) [inline]

7.18.5.16 void Socket::SetSocketId (int *sid*) [inline]

7.18.5.17 bool Socket::Start ( ) [protected]

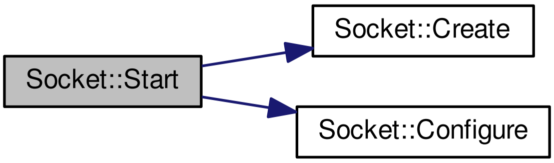
Start the socket.

Launch all mandatory operations to set the socket to be used

Returns

Success of the operation

Here is the call graph for this function:



7.18.5.18 void Socket::Stop ( )

Terminates the socket and all attached communications.

7.18.6 Field Documentation

7.18.6.1 struct sockaddr_in Socket::fAddress [private]

7.18.6.2 char Socket::fBuffer[MAX_WORD_LENGTH] [protected]

7.18.6.3 fd_set Socket::fMaster [protected]

Master file descriptor list.

7.18.6.4 int Socket::fPort [protected]

7.18.6.5 fd_set Socket::fReadFds [protected]

Temp file descriptor list for select()

7.18.6.6 int Socket::fSocketId [private]

A file descriptor for this socket, if *Create* was performed beforehand.

7.18.6.7 SocketCollection Socket::fSocketsConnected [protected]

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

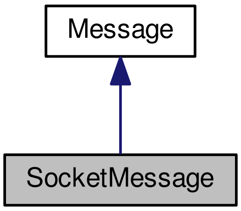
- include/Socket.h
- src/Socket.cpp

7.19 SocketMessage Class Reference

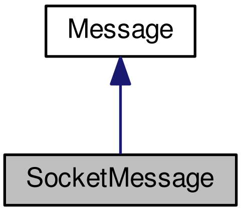
Socket-passed message type.

`#include <SocketMessage.h>`

Inheritance diagram for SocketMessage:



Collaboration diagram for SocketMessage:



Public Member Functions

- SocketMessage ()
- SocketMessage (const Message &msg)
- SocketMessage (const char ∗msg_s)
- SocketMessage (std::string msg_s)
- SocketMessage (const MessageKey &key)

  Construct a socket message out of a key.
- SocketMessage (const MessageKey &key, const char ∗value)

  Construct a socket message out of a key and a string-type value.
- SocketMessage (const MessageKey &key, std::string value)

  Construct a socket message out of a key and a string-type value.
- SocketMessage (const MessageKey &key, const short value)

  Construct a socket message out of a key and a short integer-type value.
- SocketMessage (const MessageKey &key, const int value)

  Construct a socket message out of a key and an integer-type value.

- SocketMessage (const MessageKey &key, const long value)

  Construct a socket message out of a key and a long integer-type value.
- SocketMessage (const MessageKey &key, const float value)

  Construct a socket message out of a key and a float-type value.
- SocketMessage (const MessageKey &key, const double value)

  Construct a socket message out of a key and a double precision-type value.
- SocketMessage (MessageMap msg_m)

  Construct a socket message out of a map of key/string-type value.
- ∼SocketMessage ()
- void SetKeyValue (const MessageKey &key, const char ∗value)

  String-valued message.
- void SetKeyValue (const MessageKey &key, short int_value)

  Send a short integer-valued message.
- void SetKeyValue (const MessageKey &key, int int_value)

  Send an integer-valued message.
- void SetKeyValue (const MessageKey &key, long int_value)

  Send a long integer-valued message.
- void SetKeyValue (const MessageKey &key, float float_value)

  Float-valued message.
- void SetKeyValue (const MessageKey &key, double double_value)

  Double-valued message.
- std::string GetString () const

  Extract the whole key:value message.
- MessageKey GetKey () const

  Extract the message's key.
- std::string GetValue () const

  Extract the message's string value.
- std::string GetCleanedValue () const

  Extract the message's string value (without the trailing endlines)
- int GetIntValue () const

  Extract the message's integer value.
- VectorValue GetVectorValue () const

  Extract the message's vector of string value.
- void Dump (std::ostream &os=std::cout) const

Private Member Functions

- MessageMap Object () const
- std::string String () const

Private Attributes

- MessageMap fMessage

Additional Inherited Members

7.19.1 Detailed Description

Socket-passed message type.

Author

Laurent Forthomme laurent.forthomme@cern.ch

Date

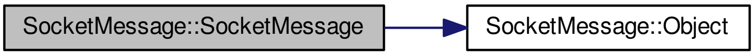
26 Mar 2015

7.19.2 Constructor & Destructor Documentation

7.19.2.1 SocketMessage::SocketMessage ( ) [inline]

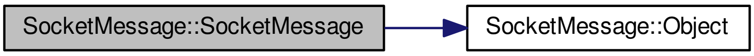
7.19.2.2 SocketMessage::SocketMessage (const Message & *msg*) [inline]

Here is the call graph for this function:



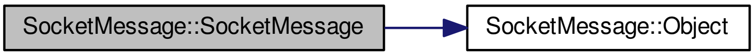
7.19.2.3 SocketMessage::SocketMessage (const char ∗ *msg_s*) [inline]

Here is the call graph for this function:



7.19.2.4 SocketMessage::SocketMessage (std::string *msg_s*) [inline]

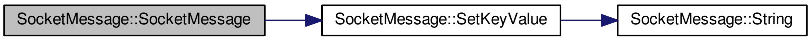
Here is the call graph for this function:



7.19.2.5 SocketMessage::SocketMessage (const MessageKey & *key*) [inline]

Construct a socket message out of a key.

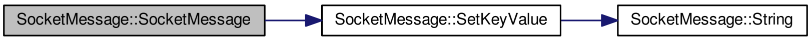
Here is the call graph for this function:



7.19.2.6 SocketMessage::SocketMessage (const MessageKey & *key*, const char ∗ *value*) [inline]

Construct a socket message out of a key and a string-type value.

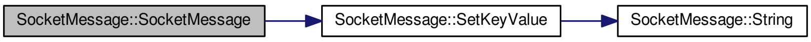
Here is the call graph for this function:



7.19.2.7 SocketMessage::SocketMessage (const MessageKey & *key*, std::string *value*) [inline]

Construct a socket message out of a key and a string-type value.

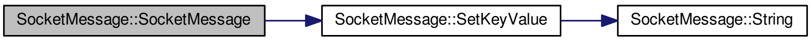
Here is the call graph for this function:



7.19.2.8 SocketMessage::SocketMessage (const MessageKey & *key*, const short *value*) [inline]

Construct a socket message out of a key and a short integer-type value.

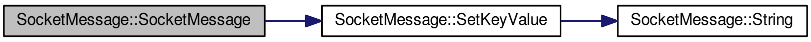
Here is the call graph for this function:



7.19.2.9 SocketMessage::SocketMessage (const MessageKey & *key*, const int *value*) [inline]

Construct a socket message out of a key and an integer-type value.

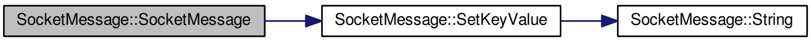
Here is the call graph for this function:



7.19.2.10 SocketMessage::SocketMessage (const MessageKey & *key*, const long *value*) [inline]

Construct a socket message out of a key and a long integer-type value.

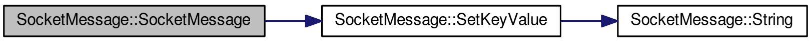
Here is the call graph for this function:



7.19.2.11 SocketMessage::SocketMessage (const MessageKey & *key*, const float *value*) [inline]

Construct a socket message out of a key and a float-type value.

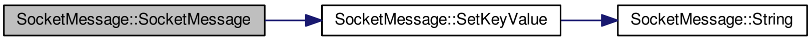
Here is the call graph for this function:



7.19.2.12 SocketMessage::SocketMessage (const MessageKey & *key*, const double *value*) [inline]

Construct a socket message out of a key and a double precision-type value.

Here is the call graph for this function:



7.19.2.13 SocketMessage::SocketMessage (MessageMap *msg_m*) [inline]

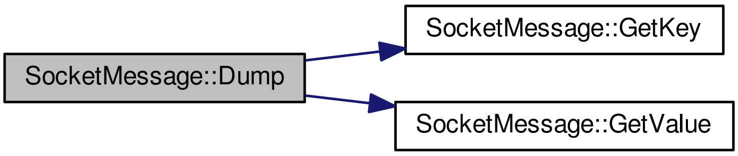
Construct a socket message out of a map of key/string-type value.

7.19.2.14 SocketMessage::∼SocketMessage ( ) [inline]

7.19.3 Member Function Documentation

7.19.3.1 void SocketMessage::Dump (std::ostream & *os* = `std::cout`) const [inline]

Here is the call graph for this function:



7.19.3.2 std::string SocketMessage::GetCleanedValue ( ) const [inline]

Extract the message's string value (without the trailing endlines)

7.19.3.3 int SocketMessage::GetIntValue ( ) const [inline]

Extract the message's integer value.

7.19.3.4 MessageKey SocketMessage::GetKey ( ) const [inline]

Extract the message's key.

7.19.3.5 std::string SocketMessage::GetString ( ) const [inline]

Extract the whole key:value message.

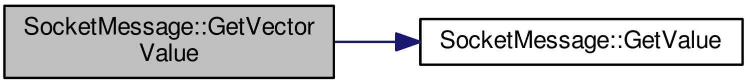
7.19.3.6 std::string SocketMessage::GetValue ( ) const [inline]

Extract the message's string value.

7.19.3.7 VectorValue SocketMessage::GetVectorValue ( ) const [inline]

Extract the message's vector of string value.

Here is the call graph for this function:

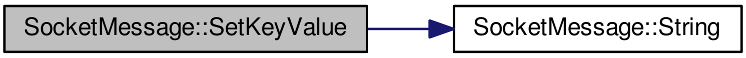


7.19.3.8 MessageMap SocketMessage::Object ( ) const [inline], [private]

7.19.3.9 void SocketMessage::SetKeyValue (const MessageKey & *key,* const char ∗ *value*) [inline]

String-valued message.

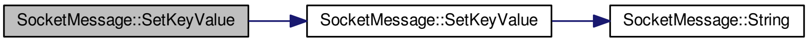
Here is the call graph for this function:



7.19.3.10 void SocketMessage::SetKeyValue (const MessageKey & *key,* short *int_value*) [inline]

Send a short integer-valued message.

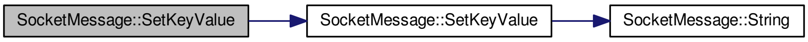
Here is the call graph for this function:



7.19.3.11 void SocketMessage::SetKeyValue (const MessageKey & *key,* int *int_value*) [inline]

Send an integer-valued message.

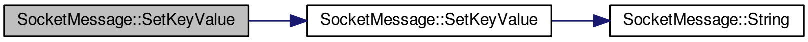
Here is the call graph for this function:



7.19.3.12 void SocketMessage::SetKeyValue (const MessageKey & *key,* long *int_value*) [inline]

Send a long integer-valued message.

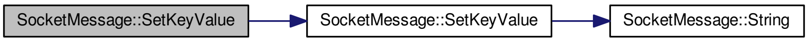
Here is the call graph for this function:



7.19.3.13 void SocketMessage::SetKeyValue (const MessageKey & *key,* float *float_value*) [inline]

Float-valued message.

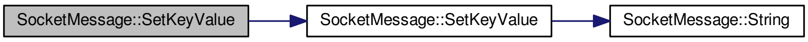
Here is the call graph for this function:



7.19.3.14 void SocketMessage::SetKeyValue (const MessageKey & *key,* double *double_value*) [inline]

Double-valued message.

Here is the call graph for this function:



7.19.3.15 std::string SocketMessage::String ( ) const [inline], [private]

7.19.4 Field Documentation

7.19.4.1 MessageMap SocketMessage::fMessage `[private]`

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

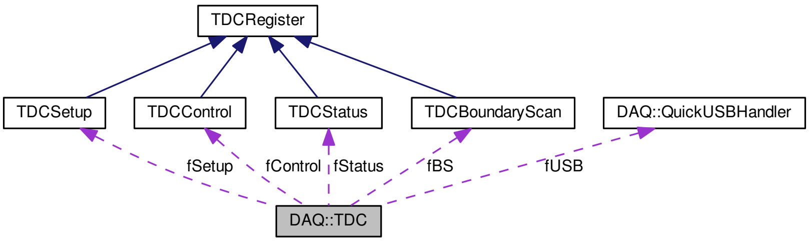
- include/SocketMessage.h

7.20 DAQ::TDC Class Reference

HPTDC object.

`#include <TDC.h>`

Collaboration diagram for DAQ::TDC:



Public Types

- enum AcquisitionMode { CONT_STORAGE, TRIG_MATCH }

  TDC acquisition mode.
- enum DetectionMode { PAIR = 0x0, OTRAILING = 0x1, OLEADING = 0x2, TRAILEAD = 0x3 }

Public Member Functions

- TDC (unsigned int id, QuickUSBHandler ∗h)
- ∼TDC ()
- void SetSetupRegister (const TDCSetup &c)

  Submit the HPTDC setup word as a TDCSetup object.
- TDCSetup GetSetupRegister ()

  Retrieve the HPTDC setup word as a TDCSetup object.
- bool CheckFirmwareVersion () const
- void SoftReset ()
- TDCEventCollection FetchEvents ()
- void ReadStatus ()

Private Member Functions

- void SendConfiguration ()

  Set the setup word to the HPTDC internal setup register.
- void ReadConfiguration ()

  Read the setup word from the HPTDC internal setup register.

- template<class T >
  void WriteRegister (unsigned int r, const T &v)

  Write one register content on the HPTDC inner memory.
- template<class T >
  T ReadRegister (unsigned int r)

  Retrieve one register content from the HPTDC inner memory.

Private Attributes

- unsigned int fId
- QuickUSBHandler ∗ fUSB
- TDCSetup fSetup
- TDCControl fControl
- TDCBoundaryScan fBS
- TDCStatus fStatus

7.20.1 Detailed Description

HPTDC object.

Author

Laurent Forthomme `laurent.forthomme@cern.ch`

Date

27 Apr 2015

7.20.2 Member Enumeration Documentation

7.20.2.1 enum DAQ::TDC::DetectionMode

Enumerator

PAIR

OTRAILING

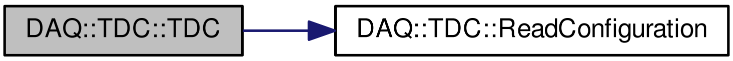
OLEADING

TRAILEAD

7.20.3 Constructor & Destructor Documentation

7.20.3.1 DAQ::TDC::TDC (unsigned int *id*, QuickUSBHandler ∗ *h*)

Here is the call graph for this function:



7.20.3.2 DAQ::TDC::∼TDC ( ) [inline]

7.20.4 Member Function Documentation

7.20.4.1 bool DAQ::TDC::CheckFirmwareVersion ( ) const

7.20.4.2 TDCEventCollection DAQ::TDC::FetchEvents ( )

7.20.4.3 TDCSetup DAQ::TDC::GetSetupRegister ( ) [inline]

Retrieve the HPTDC setup word as a TDCSetup object.

7.20.4.4 void DAQ::TDC::ReadConfiguration ( ) [private]

Read the setup word from the HPTDC internal setup register.

7.20.4.5 template<class T > T DAQ::TDC::ReadRegister (unsigned int *r*) [private]

Retrieve one register content from the HPTDC inner memory.

7.20.4.6 void DAQ::TDC::ReadStatus ( ) [inline]

7.20.4.7 void DAQ::TDC::SendConfiguration ( ) [private]

Set the setup word to the HPTDC internal setup register.

7.20.4.8 void DAQ::TDC::SetSetupRegister (const TDCSetup & *c*) [inline]

Submit the HPTDC setup word as a TDCSetup object.

7.20.4.9 void DAQ::TDC::SoftReset ( )

7.20.4.10 template<class T > void DAQ::TDC::WriteRegister (unsigned int *r*, const T & *v*) [private]

Write one register content on the HPTDC inner memory.

7.20.5 Field Documentation

7.20.5.1 TDCBoundaryScan DAQ::TDC::fBS [private]

7.20.5.2 TDCControl DAQ::TDC::fControl [private]

7.20.5.3 unsigned int DAQ::TDC::fId [private]

7.20.5.4 TDCSetup DAQ::TDC::fSetup [private]

7.20.5.5 TDCStatus DAQ::TDC::fStatus [private]

7.20.5.6 QuickUSBHandler∗ DAQ::TDC::fUSB [private]

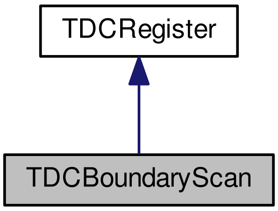
The documentation for this class was generated from the following files:

- daq/include/TDC.h
- daq/src/TDC.cpp

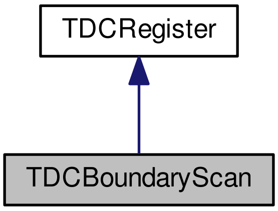
7.21 TDCBoundaryScan Class Reference

```
#include <TDCBoundaryScan.h>
```

Inheritance diagram for TDCBoundaryScan:



Collaboration diagram for TDCBoundaryScan:



Public Member Functions

- TDCBoundaryScan ()
- TDCBoundaryScan (const TDCBoundaryScan &bs)
- void SetConstantValues ()

Static Private Attributes

- static const bit kTokenOut = 0
- static const bit kStrobeOut = 1
- static const bit kSerialOut = 2
- static const bit kTest = 3
- static const bit kError = 4

- static const bit kDataReady = 5
- static const bit kParallelEnable = 6
- static const bit kParallelDataOut = 7
- static const bit kEncodedControl = 39
- static const bit kTrigger = 40
- static const bit kEventReset = 41
- static const bit kBunchReset = 42
- static const bit kGetData = 43
- static const bit kSerialBypassIn = 44
- static const bit kSerialIn = 45
- static const bit kTokenBypassIn = 46
- static const bit kTokenIn = 47
- static const bit kReset = 48
- static const bit kAuxClock = 49
- static const bit kClk = 50
- static const bit kHit = 51

Additional Inherited Members

7.21.1 Detailed Description

Author

Laurent Forthomme `laurent.forthomme@cern.ch`

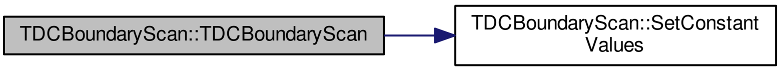
Date

24 Apr 2015

7.21.2 Constructor & Destructor Documentation

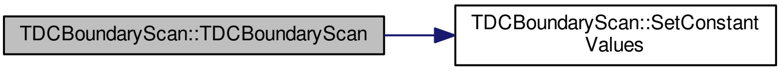
7.21.2.1 TDCBoundaryScan::TDCBoundaryScan ( ) [inline]

Here is the call graph for this function:


TDCBoundaryScan::TDCBoundaryScan
TDCBoundaryScan::SetConstant Values

7.21.2.2 TDCBoundaryScan::TDCBoundaryScan (const TDCBoundaryScan & *bs*) [inline]

Here is the call graph for this function:


TDCBoundaryScan::TDCBoundaryScan
TDCBoundaryScan::SetConstant Values

7.21.3 Member Function Documentation

7.21.3.1 void TDCBoundaryScan::SetConstantValues ( ) [inline], [virtual]

Ensure that the critical constant values are properly set in the register word

Implements TDCRegister.

7.21.4 Field Documentation

7.21.4.1 const bit TDCBoundaryScan::kAuxClock = 49 [static], [private]

7.21.4.2 const bit TDCBoundaryScan::kBunchReset = 42 [static], [private]

7.21.4.3 const bit TDCBoundaryScan::kClk = 50 [static], [private]

7.21.4.4 const bit TDCBoundaryScan::kDataReady = 5 [static], [private]

7.21.4.5 const bit TDCBoundaryScan::kEncodedControl = 39 [static], [private]

7.21.4.6 const bit TDCBoundaryScan::kError = 4 [static], [private]

7.21.4.7 const bit TDCBoundaryScan::kEventReset = 41 [static], [private]

7.21.4.8 const bit TDCBoundaryScan::kGetData = 43 [static], [private]

7.21.4.9 const bit TDCBoundaryScan::kHit = 51 [static], [private]

7.21.4.10 const bit TDCBoundaryScan::kParallelDataOut = 7 [static], [private]

7.21.4.11 const bit TDCBoundaryScan::kParallelEnable = 6 [static], [private]

7.21.4.12 const bit TDCBoundaryScan::kReset = 48 [static], [private]

7.21.4.13 const bit TDCBoundaryScan::kSerialBypassIn = 44 [static], [private]

7.21.4.14 const bit TDCBoundaryScan::kSerialIn = 45 [static], [private]

7.21.4.15 const bit TDCBoundaryScan::kSerialOut = 2 [static], [private]

7.21.4.16 const bit TDCBoundaryScan::kStrobeOut = 1 [static], [private]

7.21.4.17 const bit TDCBoundaryScan::kTest = 3 [static], [private]

7.21.4.18 const bit TDCBoundaryScan::kTokenBypassIn = 46 [static], [private]

7.21.4.19 const bit TDCBoundaryScan::kTokenIn = 47 [static], [private]

7.21.4.20 const bit TDCBoundaryScan::kTokenOut = 0 [static], [private]

7.21.4.21 const bit TDCBoundaryScan::kTrigger = 40 [static], [private]

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

- daq/include/TDCBoundaryScan.h

7.22 OnlineDBHandler::TDCConditions Struct Reference

```
#include <OnlineDBHandler.h>
```

Public Member Functions

- bool operator== (const TDCConditions &rhs) const
- TDCConditions & operator= (const TDCConditions &rhs)

Data Fields

- unsigned int run_id
- unsigned short tdc_id
- unsigned long tdc_address
- unsigned short tdc_acq_mode
- unsigned short tdc_det_mode
- std::string detector

7.22.1 Member Function Documentation

7.22.1.1 TDCConditions& OnlineDBHandler::TDCConditions::operator= (const TDCConditions & *rhs*) [inline]

7.22.1.2 bool OnlineDBHandler::TDCConditions::operator== (const TDCConditions & *rhs*) const [inline]

7.22.2 Field Documentation

7.22.2.1 std::string OnlineDBHandler::TDCConditions::detector

7.22.2.2 unsigned int OnlineDBHandler::TDCConditions::run_id

7.22.2.3 unsigned short OnlineDBHandler::TDCConditions::tdc_acq_mode

7.22.2.4 unsigned long OnlineDBHandler::TDCConditions::tdc_address

7.22.2.5 unsigned short OnlineDBHandler::TDCConditions::tdc_det_mode

7.22.2.6 unsigned short OnlineDBHandler::TDCConditions::tdc_id

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

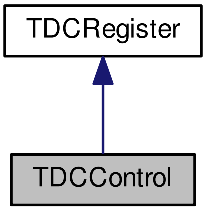
- include/OnlineDBHandler.h

7.23 TDCControl Class Reference

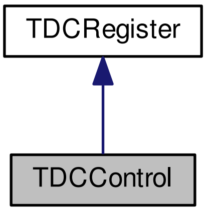
Control word to be sent to the HPTDC chip.

```
#include <TDCControl.h>
```

Inheritance diagram for TDCControl:



Collaboration diagram for TDCControl:



Public Types

- enum EnablePattern
- enum RegisterName { R_EnablePattern, R_GlobalReset, R_DLLReset, R_PLLReset }

Public Member Functions

- TDCControl ()
- TDCControl (const TDCControl &c)
- TDCControl (const std::vector< uint8_t > &words)
- void SetEnablePattern (const EnablePattern &ep)
- EnablePattern GetEnablePattern () const
- void SetGlobalReset (const bool gr=true)
- bool GetGlobalReset () const
- void SetDLLReset (const bool dr=true)
- bool GetDLLReset () const
- void SetPLLReset (const bool pr=true)
- bool GetPLLReset () const
- void EnableChannel (unsigned int id)
- void EnableAllChannels ()
- void DisableChannel (unsigned int id)

- void DisableAllChannels ()
- void Dump (int verb=1, std::ostream &os=std::cout) const
- void SetConstantValues ()

Private Member Functions

- void SetControlParity (const bool cp=true)

Static Private Attributes

- static const bit kEnablePattern = 0
- static const bit kGlobalReset = 4
- static const bit kEnableChannel = 5
- static const bit kDLLReset = 37
- static const bit kPLLReset = 38
- static const bit kControlParity = 39

Additional Inherited Members

7.23.1 Detailed Description

Control word to be sent to the HPTDC chip.

Object handling the control word provided by/to the HPTDC chip

Author

Laurent Forthomme laurent.forthomme@cern.ch

Date

24 Apr 2015

7.23.2 Member Enumeration Documentation

7.23.2.1 enum TDCControl::EnablePattern

7.23.2.2 enum TDCControl::RegisterName

Enumerator

R_EnablePattern

R_GlobalReset

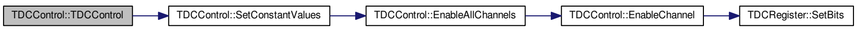
R_DLLReset

R_PLLReset

7.23.3 Constructor & Destructor Documentation

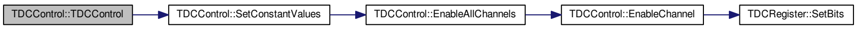
7.23.3.1 TDCControl::TDCControl ( ) [inline]

Here is the call graph for this function:


TDCControl::TDCControl
TDCControl::SetConstantValues
TDCControl::EnableAllChannels
TDCControl::EnableChannel
TDCRegister::SetBits

7.23.3.2 TDCControl::TDCControl (const **TDCControl** & *c*) [inline]

Here is the call graph for this function:

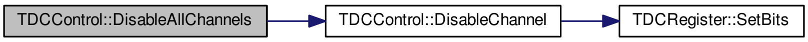

TDCControl::TDCControl
TDCControl::SetConstantValues
TDCControl::EnableAllChannels
TDCControl::EnableChannel
TDCRegister::SetBits

7.23.3.3 TDCControl::TDCControl (const std::vector< uint8_t > & *words*) [inline]

7.23.4 Member Function Documentation

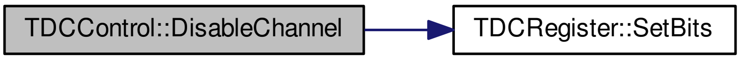
7.23.4.1 void TDCControl::DisableAllChannels ( ) [inline]

Here is the call graph for this function:



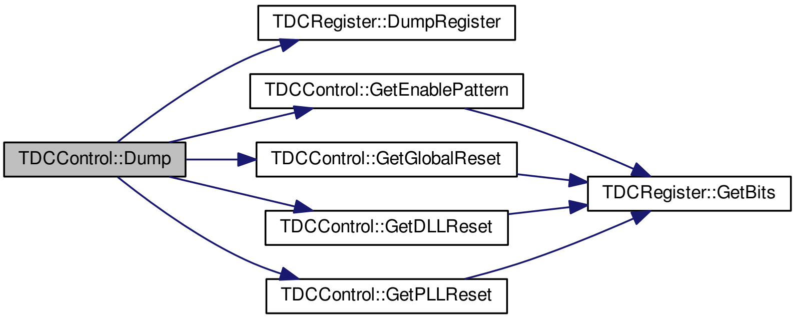
7.23.4.2 void TDCControl::DisableChannel (unsigned int *id*) [inline]

Here is the call graph for this function:



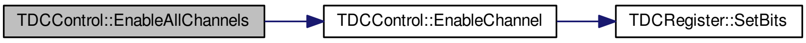
7.23.4.3 void TDCControl::Dump (int *verb* = 1, std::ostream & *os* = std::cout) const [inline]

Here is the call graph for this function:



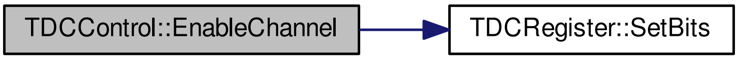
7.23.4.4 void TDCControl::EnableAllChannels ( ) [inline]

Here is the call graph for this function:



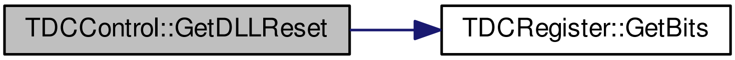
7.23.4.5 void TDCControl::EnableChannel (unsigned int *id*) [inline]

Here is the call graph for this function:



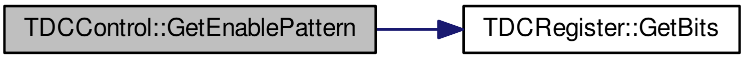
7.23.4.6 bool TDCControl::GetDLLReset ( ) const [inline]

Here is the call graph for this function:



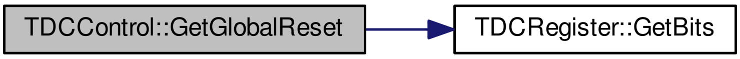
7.23.4.7 EnablePattern TDCControl::GetEnablePattern ( ) const [inline]

Here is the call graph for this function:



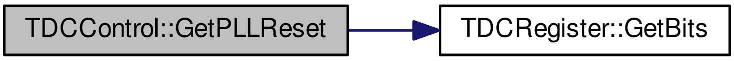
7.23.4.8 bool TDCControl::GetGlobalReset ( ) const [inline]

Here is the call graph for this function:



7.23.4.9 bool TDCControl::GetPLLReset ( ) const [inline]

Here is the call graph for this function:

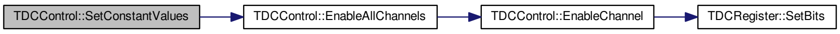


7.23.4.10 void TDCControl::SetConstantValues ( ) [inline], [virtual]

Ensure that the critical constant values are properly set in the register word

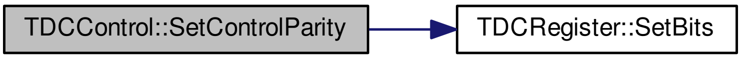
Implements TDCRegister.

Here is the call graph for this function:


TDCControl::SetConstantValues
TDCControl::EnableAllChannels
TDCControl::EnableChannel
TDCRegister::SetBits

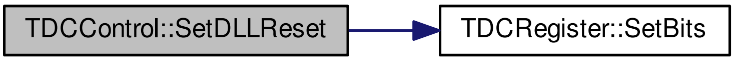
7.23.4.11 void TDCControl::SetControlParity (const bool *cp* = true) [inline], [private]

Here is the call graph for this function:



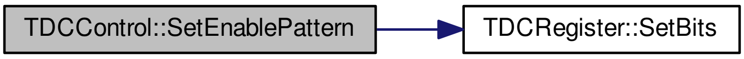
7.23.4.12 void TDCControl::SetDLLReset (const bool *dr* = true) [inline]

Here is the call graph for this function:



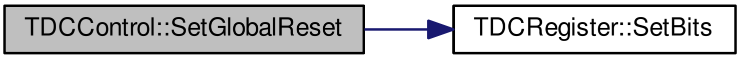
7.23.4.13 void TDCControl::SetEnablePattern (const EnablePattern & *ep*) [inline]

Here is the call graph for this function:



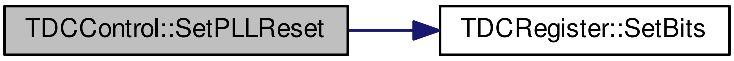
7.23.4.14 void TDCControl::SetGlobalReset (const bool *gr* = true) [inline]

Here is the call graph for this function:



7.23.4.15 void TDCControl::SetPLLReset (const bool *pr* = true) [inline]

Here is the call graph for this function:



7.23.5 Field Documentation

7.23.5.1 const bit TDCControl::kControlParity = 39 [static], [private]

7.23.5.2 const bit TDCControl::kDLLReset = 37 [static], [private]

7.23.5.3 const bit TDCControl::kEnableChannel = 5 [static], [private]

7.23.5.4 const bit TDCControl::kEnablePattern = 0 [static], [private]

7.23.5.5 const bit TDCControl::kGlobalReset = 4 [static], [private]

7.23.5.6 const bit TDCControl::kPLLReset = 38 [static], [private]

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

- daq/include/TDCControl.h

7.24 TDCErrorFlag Class Reference

Error flags handler.

`#include <TDCEvent.h>`

Public Member Functions

- TDCErrorFlag (uint16_t ef)
- virtual ∼TDCErrorFlag ()
- uint16_t GetWord () const
- void Dump () const
- bool HasReadoutFIFOOverflow (unsigned int group_id) const

  Check whether hits have been lost from read-out FIFO overflow in a given group.
- bool HasL1BufferOverflow (unsigned int group_id) const

  Check whether hits have been lost from L1 buffer overflow in a given group.
- bool HasGroupError (unsigned int group_id) const

  Check whether hits have been lost due to error in a given group.
- bool HasReachedEventSizeLimit () const

  Hits rejected because of programmed event size limit.
- bool HasTriggerFIFOOverflow () const

  Event lost (trigger FIFO overflow)
- bool HasInternalChipError () const

  Internal fatal chip error has been detected.

Private Attributes

- uint16_t fWord

Friends

- std::ostream & operator<< (std::ostream &os, const TDCErrorFlag &ef)

7.24.1 Detailed Description

Error flags handler.

Author

Laurent Forthomme `laurent.forthomme@cern.ch`

Date

22 Jun 2015

7.24.2 Constructor & Destructor Documentation

7.24.2.1 TDCErrorFlag::TDCErrorFlag (uint16_t *ef*) [inline]

7.24.2.2 virtual TDCErrorFlag::∼TDCErrorFlag ( ) [inline], [virtual]

7.24.3 Member Function Documentation

7.24.3.1 void TDCErrorFlag::Dump ( ) const [inline]

7.24.3.2 uint16_t TDCErrorFlag::GetWord ( ) const [inline]

7.24.3.3 bool TDCErrorFlag::HasGroupError (unsigned int *group_id*) const [inline]

Check whether hits have been lost due to error in a given group.

7.24.3.4 bool TDCErrorFlag::HasInternalChipError ( ) const [inline]

Internal fatal chip error has been detected.

7.24.3.5 bool TDCErrorFlag::HasL1BufferOverflow (unsigned int *group_id*) const [inline]

Check whether hits have been lost from L1 buffer overflow in a given group.

7.24.3.6 bool TDCErrorFlag::HasReachedEventSizeLimit ( ) const [inline]

Hits rejected because of programmed event size limit.

7.24.3.7 bool TDCErrorFlag::HasReadoutFIFOOverflow (unsigned int *group_id*) const [inline]

Check whether hits have been lost from read-out FIFO overflow in a given group.

7.24.3.8 bool TDCErrorFlag::HasTriggerFIFOOverflow ( ) const [inline]

Event lost (trigger FIFO overflow)

7.24.4 Friends And Related Function Documentation

7.24.4.1 std::ostream& operator<< (std::ostream & *os,* const TDCErrorFlag & *ef*) [friend]

7.24.5 Field Documentation

7.24.5.1 uint16_t TDCErrorFlag::fWord [private]

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

- include/TDCEvent.h

7.25 TDCEvent Class Reference

HPTDC event parser.

```
#include <TDCEvent.h>
```

Public Types

- enum EventType {
  TDCMeasurement = 0x0, TDCHeader = 0x1, TDCTrailer = 0x3, TDCError = 0x4,
  GlobalHeader = 0x8, GlobalTrailer = 0x10, ETTT = 0x11, Filler = 0x18,
  Trigger = 0x1f }

Public Member Functions

- TDCEvent ()
- TDCEvent (const TDCEvent &ev)
- TDCEvent (const uint32_t &word)
- TDCEvent (const EventType &ev)
- virtual ~TDCEvent ()
- void Dump () const
- void SetWord (const uint32_t &word)
- uint32_t GetWord () const
- EventType GetType () const

  Type of packet read out from the TDC.
- unsigned int GetTDCId () const

  Programmed identifier of master TDC providing the event.
- uint16_t GetEventId () const

  Event identifier from event counter.
- uint16_t GetWordCount () const

  Total number of words in event (including headers and trailers)
- unsigned int GetGeo () const
- unsigned int GetChannelId () const

  Channel number for.
- uint32_t GetEventCount () const

  Total number of events.
- uint16_t GetBunchId () const

  Bunch identifier of trigger (or trigger time tag)
- bool IsTrailing () const

  Are we dealing with a trailing or a leading measurement?
- uint32_t GetETTT () const

  Extended trigger time tag.
- uint32_t GetTime (bool pair=false) const

  Edge measurement in programmed time resolution.
- unsigned int GetWidth () const

  Width of pulse in programmed time resolution.
- unsigned int GetStatus () const
- TDCErrorFlag GetErrorFlags () const

  Return error flags if an error condition has been detected.

Private Attributes

- uint32_t fWord

7.25.1 Detailed Description

HPTDC event parser.

Object enabling to decipher any measurement/error/debug event returned by the HPTDC chip

Author

Laurent Forthomme `laurent.forthomme@cern.ch`

Date

4 May 2015

7.25.2 Member Enumeration Documentation

7.25.2.1 enum TDCEvent::EventType

Enumerator

TDCMeasurement

TDCHeader

TDCTrailer

TDCError

GlobalHeader

GlobalTrailer

ETTT

Filler

Trigger

7.25.3 Constructor & Destructor Documentation

7.25.3.1 TDCEvent::TDCEvent ( ) `[inline]`

7.25.3.2 TDCEvent::TDCEvent (const TDCEvent & *ev*) `[inline]`

7.25.3.3 TDCEvent::TDCEvent (const uint32_t & *word*) `[inline]`

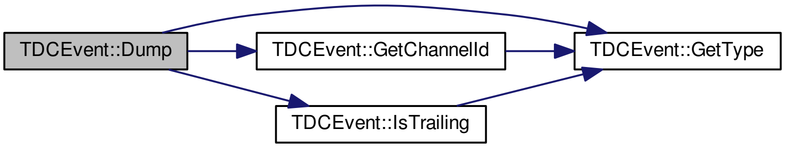
7.25.3.4 TDCEvent::TDCEvent (const EventType & *ev*) `[inline]`

7.25.3.5 virtual TDCEvent::∼TDCEvent ( ) `[inline]`, `[virtual]`

7.25.4 Member Function Documentation

7.25.4.1 void TDCEvent::Dump ( ) const [inline]

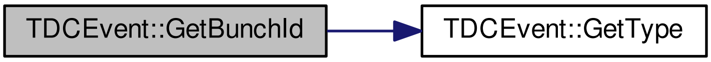
Here is the call graph for this function:



7.25.4.2 uint16_t TDCEvent::GetBunchId ( ) const [inline]

Bunch identifier of trigger (or trigger time tag)

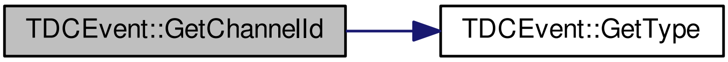
Here is the call graph for this function:



7.25.4.3 unsigned int TDCEvent::GetChannelId ( ) const [inline]

Channel number for.

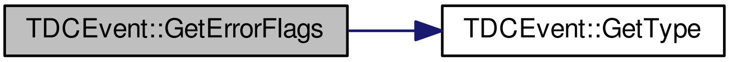
Here is the call graph for this function:



7.25.4.4 TDCErrorFlag TDCEvent::GetErrorFlags ( ) const [inline]

Return error flags if an error condition has been detected.

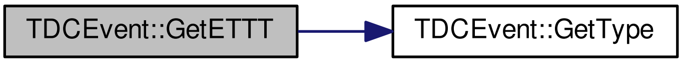
Here is the call graph for this function:



7.25.4.5 uint32_t TDCEvent::GetETTT ( ) const [inline]

Extended trigger time tag.

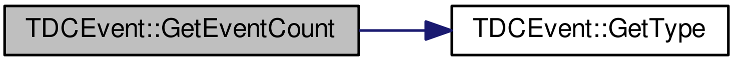
Here is the call graph for this function:



7.25.4.6 uint32_t TDCEvent::GetEventCount ( ) const [inline]

Total number of events.

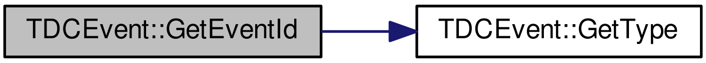
Here is the call graph for this function:



7.25.4.7 uint16_t TDCEvent::GetEventId ( ) const [inline]

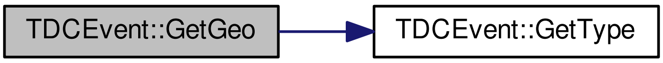
Event identifier from event counter.

Here is the call graph for this function:



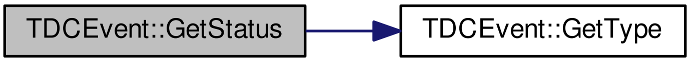
7.25.4.8 unsigned int TDCEvent::GetGeo ( ) const [inline]

Here is the call graph for this function:



7.25.4.9 unsigned int TDCEvent::GetStatus ( ) const [inline]

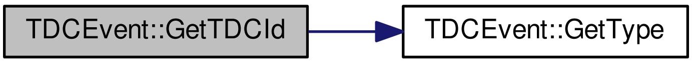
Here is the call graph for this function:



7.25.4.10 unsigned int TDCEvent::GetTDCId ( ) const [inline]

Programmed identifier of master TDC providing the event.

Here is the call graph for this function:



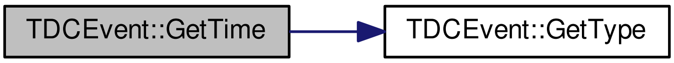
7.25.4.11 uint32_t TDCEvent::GetTime (bool *pair* = false) const [inline]

Edge measurement in programmed time resolution.

Parameters

in	*pair*	Are we dealing with a pair measurement? (only for leading time word)

Here is the call graph for this function:



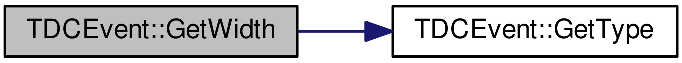
7.25.4.12 EventType TDCEvent::GetType ( ) const [inline]

Type of packet read out from the TDC.

7.25.4.13 unsigned int TDCEvent::GetWidth ( ) const [inline]

Width of pulse in programmed time resolution.

Here is the call graph for this function:

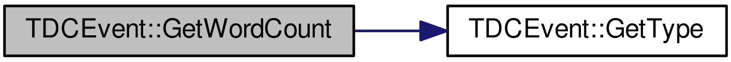


7.25.4.14 uint32_t TDCEvent::GetWord ( ) const [inline]

7.25.4.15 uint16_t TDCEvent::GetWordCount ( ) const [inline]

Total number of words in event (including headers and trailers)

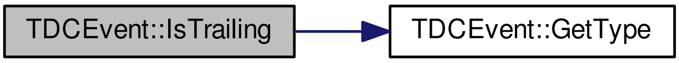
Here is the call graph for this function:



7.25.4.16 bool TDCEvent::IsTrailing ( ) const [inline]

Are we dealing with a trailing or a leading measurement?

Here is the call graph for this function:



7.25.4.17 void TDCEvent::SetWord (const uint32_t & *word*) [inline]

7.25.5 Field Documentation

7.25.5.1 uint32_t TDCEvent::fWord [private]

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

- include/TDCEvent.h

7.26 TDCMeasurement Class Reference

```
#include <TDCMeasurement.h>
```

Public Member Functions

- TDCMeasurement ()
- TDCMeasurement (const std::vector< TDCEvent > &v)

- ∼TDCMeasurement ()
- void Dump ()
- void SetEventsCollection (const std::vector< TDCEvent > &v)
- uint32_t GetLeadingTime (unsigned short event_id=0)
- uint32_t GetTrailingTime (unsigned short event_id=0)
- uint16_t GetToT (unsigned short event_id=0)
- uint16_t GetChannelId (unsigned short event_id=0)
- uint16_t GetTDCId ()
- uint16_t GetEventId ()
- uint16_t GetBunchId ()
- uint32_t GetETTT ()
- size_t NumEvents () const
- size_t NumErrors () const

Private Attributes

- std::map< TDCEvent::EventType, TDCEvent > fMap
- std::vector< std::pair< TDCEvent, TDCEvent > > fEvents

7.26.1 Detailed Description

Author

Laurent Forthomme `laurent.forthomme@cern.ch`

Date

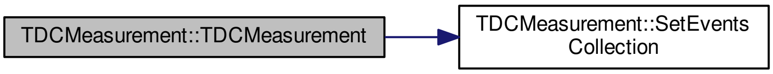
Jun 2015

7.26.2 Constructor & Destructor Documentation

7.26.2.1 TDCMeasurement::TDCMeasurement ( ) [inline]

7.26.2.2 TDCMeasurement::TDCMeasurement (const std::vector< TDCEvent > & v) [inline]

Here is the call graph for this function:

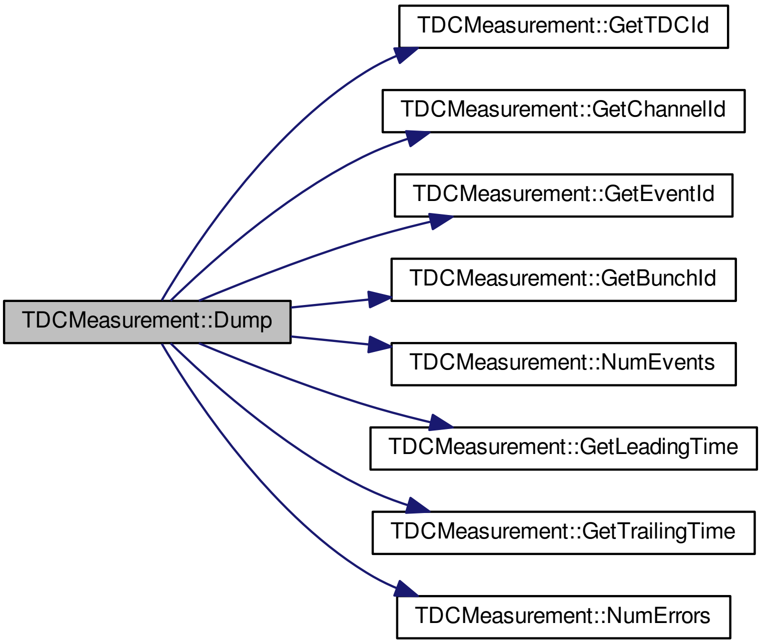

TDCMeasurement::TDCMeasurement
TDCMeasurement::SetEvents Collection

7.26.2.3 TDCMeasurement::∼TDCMeasurement ( ) [inline]

7.26.3 Member Function Documentation

7.26.3.1 void TDCMeasurement::Dump ( ) [inline]

Here is the call graph for this function:


TDCMeasurement::Dump
TDCMeasurement::GetTDCId
TDCMeasurement::GetChannelId
TDCMeasurement::GetEventId
TDCMeasurement::GetBunchId
TDCMeasurement::NumEvents
TDCMeasurement::GetLeadingTime
TDCMeasurement::GetTrailingTime
TDCMeasurement::NumErrors

7.26.3.2 uint16_t TDCMeasurement::GetBunchId ( ) [inline]

7.26.3.3 uint16_t TDCMeasurement::GetChannelId (unsigned short *event_id* = 0) [inline]

7.26.3.4 uint32_t TDCMeasurement::GetETTT ( ) [inline]

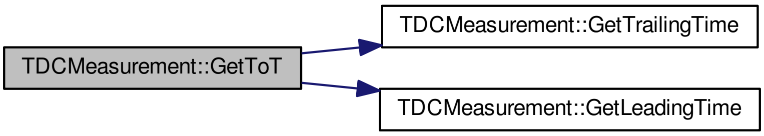
7.26.3.5 uint16_t TDCMeasurement::GetEventId ( ) [inline]

7.26.3.6 uint32_t TDCMeasurement::GetLeadingTime (unsigned short *event_id* = 0) [inline]

7.26.3.7 uint16_t TDCMeasurement::GetTDCId ( ) [inline]

7.26.3.8 uint16_t TDCMeasurement::GetToT (unsigned short *event_id* = 0) [inline]

Here is the call graph for this function:


TDCMeasurement::GetToT
TDCMeasurement::GetTrailingTime
TDCMeasurement::GetLeadingTime

7.26.3.9 uint32_t TDCMeasurement::GetTrailingTime (unsigned short *event_id* = 0) [inline]

7.26.3.10 size_t TDCMeasurement::NumErrors ( ) const [inline]

7.26.3.11 size_t TDCMeasurement::NumEvents ( ) const [inline]

7.26.3.12 void TDCMeasurement::SetEventsCollection (const std::vector< **TDCEvent** > & *v*) [inline]

7.26.4 Field Documentation

7.26.4.1 std::vector< std::pair<**TDCEvent,TDCEvent**> > TDCMeasurement::fEvents [private]

7.26.4.2 std::map<**TDCEvent::EventType,TDCEvent**> TDCMeasurement::fMap [private]

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

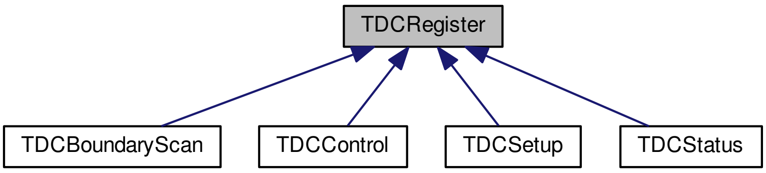
- include/TDCMeasurement.h

7.27 TDCRegister Class Reference

General register object to interact with a HPTDC chip.

```
#include <TDCRegister.h>
```

Inheritance diagram for TDCRegister:



Public Types

- typedef uint16_t bit

  LSB index.
- typedef uint32_t word_t

  Unit of the TDC register word to be successfully contained on any machine.

Public Member Functions

- TDCRegister (const unsigned int size)
- TDCRegister (const unsigned int size, const TDCRegister &r)
- TDCRegister (const unsigned int size, const std::vector< uint8_t > words)
- virtual ∼TDCRegister ()
- TDCRegister & operator= (const TDCRegister &r)
- void SetWord (const unsigned int i, const word_t word)

  Set one bit(s) subset in the register word.
- word_t GetWord (const unsigned int i) const

  Retrieve one subset from the register word.
- word_t ∗ GetWords () const
- uint8_t GetNumWords () const

  Number of words in the register.
- void DumpRegister (std::ostream &os=std::cout, const bit max_bits=-1) const
- virtual void SetConstantValues ()=0

Protected Member Functions

- void SetBits (uint16_t lsb, uint16_t word, uint8_t size)

  Set bits in the register word.
- uint16_t GetBits (uint16_t lsb, uint8_t size) const

  Extract bits from the register word.
- void Clear ()

  Set all bits in this register to '0'.

Protected Attributes

- word_t ∗ fWord

  Pointer to this register's word.
- unsigned int fNumWords
- unsigned int fWordSize

  Number of bits in this register.

7.27.1 Detailed Description

General register object to interact with a HPTDC chip.

Author

Laurent Forthomme `laurent.forthomme@cern.ch`

Date

24 Apr 2015

7.27.2 Member Typedef Documentation

7.27.2.1 typedef uint16_t TDCRegister::bit

LSB index.

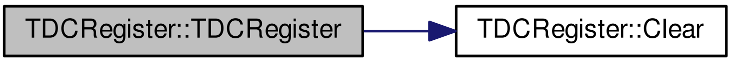
7.27.2.2 typedef uint32_t TDCRegister::word_t

Unit of the TDC register word to be successfully contained on any machine.

7.27.3 Constructor & Destructor Documentation

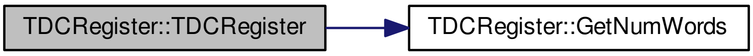
7.27.3.1 TDCRegister::TDCRegister (const unsigned int *size*) [inline]

Here is the call graph for this function:



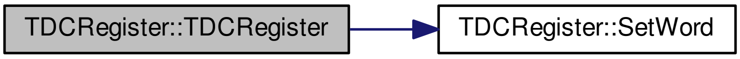
7.27.3.2 TDCRegister::TDCRegister (const unsigned int *size,* const TDCRegister & *r*) [inline]

Here is the call graph for this function:



7.27.3.3 TDCRegister::TDCRegister (const unsigned int *size,* const std::vector< uint8_t > *words*) [inline]

Here is the call graph for this function:



7.27.3.4 virtual TDCRegister::∼TDCRegister ( ) [inline], [virtual]

7.27.4 Member Function Documentation

7.27.4.1 void TDCRegister::Clear ( ) [inline], [protected]

Set all bits in this register to '0'.

7.27.4.2 void TDCRegister::DumpRegister (std::ostream & *os* = std::cout, const bit *max_bits* = -1) const [inline]

7.27.4.3 uint16_t TDCRegister::GetBits (uint16_t *lsb*, uint8_t *size*) const [inline], [protected]

Extract bits from the register word.

Extract a fixed amount of bits from the full register word

Parameters

in	*lsb*	Least significant bit of the word to retrieve
in	*size*	Size of the word to retrieve

7.27.4.4 uint8_t TDCRegister::GetNumWords ( ) const [inline]

Number of words in the register.

Return the number of words making up the full register word.

7.27.4.5 word_t TDCRegister::GetWord (const unsigned int *i*) const [inline]

Retrieve one subset from the register word.

7.27.4.6 word_t∗ TDCRegister::GetWords ( ) const [inline]

7.27.4.7 TDCRegister& TDCRegister::operator= (const TDCRegister & *r*) [inline]

7.27.4.8 void TDCRegister::SetBits (uint16_t *lsb*, uint16_t *word*, uint8_t *size*) [inline], [protected]

Set bits in the register word.

Set a fixed amount of bits in the full register word

Parameters

in	*lsb*	Least significant bit of the word to set
in	*word*	Word to set
in	*size*	Size of the word to set

7.27.4.9 virtual void TDCRegister::SetConstantValues ( ) [pure virtual]

Ensure that the critical constant values are properly set in the register word

Implemented in TDCSetup, TDCControl, TDCBoundaryScan, and TDCStatus.

7.27.4.10 void TDCRegister::SetWord (const unsigned int *i*, const word_t *word*) [inline]

Set one bit(s) subset in the register word.

7.27.5 Field Documentation

7.27.5.1 unsigned int TDCRegister::fNumWords [protected]

Number of words to fit the *fWordSize* bits of this register to this object

7.27.5.2 word_t* TDCRegister::fWord [protected]

Pointer to this register's word.

7.27.5.3 unsigned int TDCRegister::fWordSize [protected]

Number of bits in this register.

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

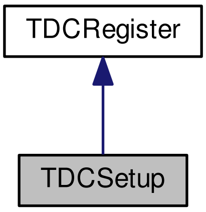
- daq/include/TDCRegister.h

7.28 TDCSetup Class Reference

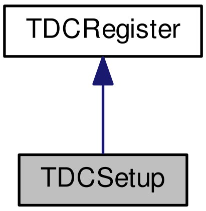
Setup word to be sent to the HPTDC chip.

`#include <TDCSetup.h>`

Inheritance diagram for TDCSetup:



Collaboration diagram for TDCSetup:



Public Types

- enum EdgeResolution {
  E_100ps =0, E_200ps, E_400ps, E_800ps,
  E_1p6ns, E_3p12ns, E_6p25ns, E_12p5ns }
- enum DeadTime { DT_5ns =0, DT_10ns, DT_30ns, DT_100ns }
- enum WidthResolution {
  W_100ps =0, W_200ps, W_400ps, W_800ps,
  W_1p6ns, W_3p2ns, W_6p25ns, W_12p5ns,
  W_25ns, W_50ns, W_100ns, W_200ns,
  W_400ns, W_800ns }
- enum EnabledError {
  VernierError =0x1, CoarseError =0x2, ChannelSelectError =0x4, L1BufferParityError =0x8,
  TriggerFIFOParityError =0x10, TriggerMatchingError =0x20, ReadoutFIFOParityError =0x40, ReadoutStateError =0x80,
  SetupParityError =0x100, ControlParityError =0x200, JTAGInstructionParityError =0x400 }
- enum DLLSpeedMode { DLL_40MHz =0x0, DLL_160MHz =0x1, DLL_320MHz =0x2, DLL_Illegal =0x3 }
- enum SerialClockSource { Serial_pll_clock_80 =0x0, Serial_pll_clock_160 =0x1, Serial_pll_clock_40 =0x2, Serial_aux_clock =0x3 }
- enum IOClockSource { IO_clock_40 =0x0, IO_pll_clock_80 =0x1, IO_pll_clock_160 =0x2, IO_aux_clock =0x3 }
- enum CoreClockSource { Core_clock_40 =0x0, Core_pll_clock_80 =0x1, Core_pll_clock_160 =0x2, Core_aux_clock =0x3 }
- enum DLLClockSource {
  DLL_clock_40 =0x0, DLL_pll_clock_40 =0x1, DLL_pll_clock_160 =0x2, DLL_pll_clock_320 =0x3,
  DLL_aux_clock =0x4 }
- enum ReadoutSpeed { RO_Fixed =0x0, RO_pll_80Mbits_s =0x1 }
- enum SerialStrobeType { SS_NoStrobe =0x0, SS_DSStrobe =0x1, SS_LeadingTrailingStrobe =0x2, SS_LeadingEdge =0x3 }
- enum ReadoutSingleCycleSpeed {
  RSC_40Mbits_s =0x0, RSC_20Mbits_s =0x1, RSC_10Mbits_s =0x2, RSC_5Mbits_s =0x3,
  RSC_2p5Mbits_s =0x4, RSC_1p25Mbits_s =0x5, RSC_625kbits_s =0x6, RSC_312p5kbits_s =0x7 }

Public Member Functions

- TDCSetup ()
- TDCSetup (const TDCSetup &c)
- void SetEnableErrorMark (const bool em)

  Mark events with error if global error signal is set.

- bool GetEnableErrorMark () const
- void SetEnableErrorBypass (const bool eb)

  Bypass TDC chip if global error signal is set.
- bool GetEnableErrorBypass () const
- void SetEnableError (const uint16_t &err)

  Enable internal error types for generation of global error signals.
- uint16_t GetEnableError () const
- void SetEnableSerial (const bool es)

  Enable of serial read-out (otherwise parallel read-out)
- bool GetEnableSerial () const
- void SetEnableJTAGReadout (const bool jr)

  Enable of read-out via JTAG.
- bool GetEnableJTAGReadout () const
- void SetReadoutFIFOSize (int rfs)

  Effective size of readout FIFO.
- int GetReadoutFIFOSize () const
- void SetRejectCountOffset (uint16_t rco)

  Set the offset in reject counter (defines reject latency together with coarse count offset)
- uint16_t GetRejectCountOffset () const

  Extract the offset in reject counter.
- void SetSearchWindow (uint16_t sw)

  Set the search window (in multiples of clock cycles: 0=25 ns, 1=50 ns, ...)
- uint16_t GetSearchWindow () const

  Extract the search window (in multiples of clock cycles: 0=25 ns, 1=50 ns, ...)
- void SetMatchWindow (uint16_t mw)

  Set the matching window (in multiples of clock cycles: 0=25 ns, 1=50 ns, ...)
- uint16_t GetMatchWindow () const

  Extract the matching window (in multiples of clock cycles: 0=25 ns, 1=50 ns, ...)
- void SetEdgeResolution (const EdgeResolution r)
- EdgeResolution GetEdgeResolution () const
- void SetMaxEventSize (int sz=-1)

  Set the maximum number of hits per event.
- uint8_t GetMaxEventSize () const

  Extract the maximum number of hits per event.
- void SetRejectFIFOFull (const bool rej=true)

  Reject hits when readout FIFO full.
- bool GetRejectFIFOFull () const

  Are hits rejected when readout FIFO is full?
- void SetEnableReadoutOccupancy (const bool ro=true)

  Enable the readout of buffer occupancies for each event (for debugging purposes)
- bool GetEnableReadoutOccupancy () const
- void SetEnableReadoutSeparator (const bool ro=true)

  Enable the readout of separators for each event (for debugging purposes, valid if readout of occupancies is enabled)
- bool GetEnableReadoutSeparator () const
- void SetEventCountOffset (uint16_t eco)

  Set offset for the event counter.
- void SetTriggerCountOffset (uint16_t tco)

  Set offset for the trigger time tag counter to set effective trigger latency.
- uint16_t GetTriggerCountOffset () const

  Extract trigger time tag count offset.
- void SetChannelOffset (int channel, uint16_t offset)

Set the time offset for one single channel.
- uint16_t GetChannelOffset (int channel) const

  Return the offset for one single channel.
- void SetAllChannelsOffset (uint16_t offset)

  Set the time offset for all channels.
- void SetCoarseCountOffset (uint16_t cco)

  Set offset for the coarse time counter.
- uint16_t GetCoarseCountOffset () const

  Extract offset for the coarse time counter.
- void SetDLLAdjustment (int tap, uint8_t adj)

  Set the DLL taps adjustments with a resolution of ∼10 ps.
- uint8_t GetDLLAdjustment (int tap) const

  Set the adjustment of DLL taps.
- void SetAllTapsDLLAdjustment (uint8_t adj)

  Extract the adjustment of DLL taps.
- void SetRCAdjustment (int tap, uint8_t adj)

  Set the adjustment of the RC delay line.
- uint8_t GetRCAdjustment (int tap)

  Extract the adjustment of the RC delay line.
- void SetWidthResolution (const WidthResolution r)

  Set the pulse width resolution when paired measurements are performed.
- WidthResolution GetWidthResolution () const

  Extract the pulse width resolution when paired measurements are performed.
- void SetVernierOffset (const uint8_t vo)

  Set the offset in vernier decoding.
- uint8_t GetVernierOffset () const

  Extract the offset in vernier decoding.
- void SetDeadTime (const DeadTime dt)

  Channel dead time between hits.
- DeadTime GetDeadTime () const
- void SetTestInvert (const bool ti=true)

  Automatic inversion of test pattern. Only used during production testing.
- bool GetTestInvert () const
- void SetTestMode (const bool tm=true)

  Test mode where hit data are taken from coretest. Only used during production testing.
- bool GetTestMode () const
- void SetTrailingMode (const bool trail=true)

  Enable/disable the detection of trailing edges.
- bool GetTrailingMode () const

  Extract the status for the detection of trailing edges.
- void SetLeadingMode (const bool lead=true)

  Enable the detection of leading edges.
- bool GetLeadingMode () const

  Extract the status for the detection of leading edges.
- void SetTriggerMatchingMode (const bool trig=true)

  Set the enable status of trigger matching mode.
- bool GetTriggerMatchingMode () const

  Extract the enable status of trigger matching mode.
- void SetEdgesPairing (const bool pair=true)

  Enable the pairing of leading and trailing edges (overrides individual enable of leading/trailing edges)
- bool GetEdgesPairing () const

- void SetSetupParity (const bool sp=true)

  Set the parity of setup data (should be an even parity)
- bool GetSetupParity () const

  Extract the parity of setup data (should be an even parity)
- void SetConstantValues ()

  Ensure that the critical constant values are properly set in the setup word.
- uint16_t GetTriggerLatency () const

  Effective trigger latency in number of clock cycles (when no counter roll-over is used)
- void SetTDCId (const uint8_t id=0x0)
- uint16_t GetTDCId () const
- void Dump (int verb=1, std::ostream &os=std::cout) const

Private Member Functions

- void SetReadoutSingleCycleSpeed (const ReadoutSingleCycleSpeed rscs=RSC_40Mbits_s)

  Serial transmission speed in single cycle mode.
- void SetSerialDelay (const uint8_t sd=0x0)

  Programmable delay of serial input, in time unit $\sim$ *1 ns.*
- void SetStrobeSelect (const SerialStrobeType ss=SS_NoStrobe)
- void SetReadoutSpeedSelect (const ReadoutSpeed rss=RO_Fixed)

  Selection of serial read-out speed.
- void SetTokenDelay (const uint8_t td=0x0)

  Programmable delay of token input, in time unit $\sim$ *1 ns.*
- void SetEnableLocalTrailer (const bool elt=true)

  Enable of local trailers in read-out.
- void SetEnableLocalHeader (const bool elh=true)

  Enable of local headers in read-out.
- void SetEnableGlobalTrailer (const bool egt=true)

  Enable of global trailers in read-out (only valid for master TDC)
- void SetEnableGlobalHeader (const bool egh=true)

  Enable of global headers in read-out (only valid for master TDC)
- void SetKeepToken (const bool kt=true)
- void SetMaster (const bool m=true)
- void SetEnableBytewise (const bool seb=true)
- void SetBypassInputs (const bool sbi=true)

  Select serial in and token in from bypass inputs.
- void SetEnableOverflowDetect (const bool eod=true)

  Enable overflow detection of L1 buffers (should always be enabled!)
- void SetEnableRelative (const bool er=true)
- void SetEnableAutomaticReject (const bool ear=true)

  Enable of automatic rejection (should always be enabled if trigger matching mode!)
- void SetEnableSetCountersOnBunchReset (const bool escobr=true)

  Enable all counters to be set on bunch count reset.
- void SetEnableMasterResetCode (const bool emrc=true)

  Enable master reset code on encoded_control.
- void SetEnableMasterResetOnEventReset (const bool emroer=true)

  Enable master reset of whole TDC on event reset.
- void SetEnableResetChannelBufferWhenSeparator (const bool ercbws=true)

  Enable reset channel buffers when separator.
- void SetEnableSeparatorOnEventReset (const bool esoer=true)

  Enable generation of separator on event reset.

- void SetEnableSeparatorOnBunchReset (const bool esobr=true)

  Enable generation of separator on bunch reset.
- void SetEnableDirectEventReset (const bool eder=true)

  Enable of direct event reset input pin (1), otherwise taken from encoded control.
- void SetEnableDirectBunchReset (const bool edbr=true)

  Enable of direct bunch reset input pin (1), otherwise taken from encoded control.
- void SetEnableDirectTrigger (const bool edt=true)

  Enable of direct trigger input pin.
- void SetLowPowerMode (const bool lpm=true)

  Low power mode of channel buffers.
- void SetDLLControl (const uint8_t dc)

  Control of DLL (DLL charge pump levels)
- void SetModeRCCompression (const bool mrc=true)

  Perform RC interpolation on-chip (only valid in very high resolution mode)
- void SetModeRC (const bool mr=true)

  Enable of RR delay lines mode (in very high resolution mode) ; only for channels 0-4-8-12-16-20-24-28 active.
- void SetDLLMode (const DLLSpeedMode dsm)

  Selection of DLL speed mode.
- void SetPLLControl (const uint8_t charge_pump_current=0x4, const bool power_down_mode=false, const bool enable_test_outputs=false, const bool invert_connection_to_status=false)

  Control of PLL.
- void SetSerialClockDelay (const bool delay_clock, const uint8_t delay)

  Delay of internal serial clock.
- void SetIOClockDelay (const bool delay_clock, const uint8_t delay)

  Delay of internal I/O clock.
- void SetCoreClockDelay (const bool delay_clock, const uint8_t delay)

  Delay of internal core clock.
- void SetDLLClockDelay (const bool delay_clock, const uint8_t delay)

  Delay of internal DLL clock.
- void SetSerialClockSource (const SerialClockSource scs)

  Selection of source for serial clock.
- void SetIOClockSource (const IOClockSource ics)

  Selection of clock source for I/O signals.
- void SetCoreClockSource (const CoreClockSource ccs)

  Selection of clock source for internal logic.
- void SetDLLClockSource (const DLLClockSource dcs)

  Selection of clock source for DLL.
- void SetRollOver (const uint16_t ro=0xFFF)

  Counter roll over value, defining maximal count value from where counters will be reset to 0.
- void SetEnableTTLSerial (const bool ts=true)

  Enable LV TTL inputs on serial registers, and disable their drivers.
- void SetEnableTTLControl (const bool tc=true)

  Enable LV TTL inputs on control registers.
- void SetEnableTTLReset (const bool tr=true)

  Enable LV TTL input on reset, otherwise uses LVDS input levels.
- void SetEnableTTLClock (const bool tc=true)

  Enable LV TTL inputs on: clk, aux_clock, otherwise uses LVDS input levels.
- void SetEnableTTLHit (const bool th=true)

  Enable LV TTL input on hit[31:0], otherwise uses LVDS input levels.
- void SetTest (const bool test=true)

Static Private Attributes

- static const bit kTestSelect = 0
- static const bit kEnableErrorMark = 4
- static const bit kEnableErrorBypass = 5
- static const bit kEnableError = 6
- static const bit kReadoutSingleCycleSpeed = 17
- static const bit kSerialDelay = 20
- static const bit kStrobeSelect = 24
- static const bit kReadoutSpeedSelect = 26
- static const bit kTokenDelay = 27
- static const bit kEnableLocalTrailer = 31
- static const bit kEnableLocalHeader = 32
- static const bit kEnableGlobalTrailer = 33
- static const bit kEnableGlobalHeader = 34
- static const bit kKeepToken = 35
- static const bit kMaster = 36
- static const bit kEnableBytewise = 37
- static const bit kEnableSerial = 38
- static const bit kEnableJTAGReadout = 39
- static const bit kTDCId = 40
- static const bit kSelectBypassInputs = 44
- static const bit kReadoutFIFOSize = 45
- static const bit kRejectCountOffset = 48
- static const bit kSearchWindow = 60
- static const bit kMatchWindow = 72
- static const bit kLeadingResolution = 84
- static const bit kMaxEventSize = 116
- static const bit kRejectFIFOFull = 120
- static const bit kEnableReadoutOccupancy = 121
- static const bit kEnableReadoutSeparator = 122
- static const bit kEnableOverflowDetect = 123
- static const bit kEnableRelative = 124
- static const bit kEnableAutomaticReject = 125
- static const bit kEventCountOffset = 126
- static const bit kTriggerCountOffset = 138
- static const bit kEnableSetCountersOnBunchReset = 150
- static const bit kEnableMasterResetCode = 151
- static const bit kEnableMasterResetOnEventReset = 152
- static const bit kEnableResetChannelBufferWhenSeparator = 153
- static const bit kEnableSeparatorOnEventReset = 154
- static const bit kEnableSeparatorOnBunchReset = 155
- static const bit kEnableDirectEventReset = 156
- static const bit kEnableDirectBunchReset = 157
- static const bit kEnableDirectTrigger = 158
- static const bit kOffset0 = 438
- static const bit kCoarseCountOffset = 447
- static const bit kDLLTapAdjust0 = 459
- static const bit kRCAdjust0 = 555
- static const bit kLowPowerMode = 570
- static const bit kWidthSelect = 571
- static const bit kVernierOffset = 575
- static const bit kDLLControl = 580
- static const bit kDeadTime = 584
- static const bit kTestInvert = 586

- static const bit kTestMode = 587
- static const bit kTrailing = 588
- static const bit kLeading = 589
- static const bit kModeRCCompression = 590
- static const bit kModeRC = 591
- static const bit kDLLMode = 592
- static const bit kPLLControl = 594
- static const bit kSerialClockDelay = 602
- static const bit kIOClockDelay = 606
- static const bit kCoreClockDelay = 610
- static const bit kDLLClockDelay = 614
- static const bit kSerialClockSource = 618
- static const bit kIOClockSource = 620
- static const bit kCoreClockSource = 622
- static const bit kDLLClockSource = 624
- static const bit kRollOver = 627
- static const bit kEnableMatching = 639
- static const bit kEnablePair = 640
- static const bit kEnableTTLSerial = 641
- static const bit kEnableTTLControl = 642
- static const bit kEnableTTLReset = 643
- static const bit kEnableTTLClock = 644
- static const bit kEnableTTLHit = 645
- static const bit kSetupParity = 646

Additional Inherited Members

7.28.1 Detailed Description

Setup word to be sent to the HPTDC chip.

Object handling the setup word provided by/to the HPTDC chip

Author

Laurent Forthomme `laurent.forthomme@cern.ch`

Date

16 Apr 2015

7.28.2 Member Enumeration Documentation

7.28.2.1 enum TDCSetup::CoreClockSource

Enumerator

Core_clock_40

Core_pll_clock_80

Core_pll_clock_160

Core_aux_clock

7.28.2.2 enum TDCSetup::DeadTime

Enumerator

DT_5ns

DT_10ns

DT_30ns

DT_100ns

7.28.2.3 enum TDCSetup::DLLClockSource

Enumerator

DLL_clock_40

DLL_pll_clock_40

DLL_pll_clock_160

DLL_pll_clock_320

DLL_aux_clock

7.28.2.4 enum TDCSetup::DLLSpeedMode

Enumerator

DLL_40MHz

DLL_160MHz

DLL_320MHz

DLL_Illegal

7.28.2.5 enum TDCSetup::EdgeResolution

Enumerator

E_100ps

E_200ps

E_400ps

E_800ps

E_1p6ns

E_3p12ns

E_6p25ns

E_12p5ns

7.28.2.6 enum TDCSetup::EnabledError

Enumerator

VernierError

CoarseError

ChannelSelectError

L1BufferParityError

TriggerFIFOParityError

TriggerMatchingError

ReadoutFIFOParityError

ReadoutStateError

SetupParityError

ControlParityError

JTAGInstructionParityError

7.28.2.7 enum TDCSetup::IOClockSource

Enumerator

IO_clock_40

IO_pll_clock_80

IO_pll_clock_160

IO_aux_clock

7.28.2.8 enum TDCSetup::ReadoutSingleCycleSpeed

Enumerator

RSC_40Mbits_s

RSC_20Mbits_s

RSC_10Mbits_s

RSC_5Mbits_s

RSC_2p5Mbits_s

RSC_1p25Mbits_s

RSC_625kbits_s

RSC_312p5kbits_s

7.28.2.9 enum TDCSetup::ReadoutSpeed

Enumerator

RO_Fixed

RO_pll_80Mbits_s

7.28.2.10 enum TDCSetup::SerialClockSource

Enumerator

Serial_pll_clock_80

Serial_pll_clock_160

Serial_pll_clock_40

Serial_aux_clock

7.28.2.11 enum TDCSetup::SerialStrobeType

Enumerator

SS_NoStrobe

SS_DSStrobe

SS_LeadingTrailingStrobe

SS_LeadingEdge

7.28.2.12 enum TDCSetup::WidthResolution

Enumerator

W_100ps

W_200ps

W_400ps

W_800ps

W_1p6ns

W_3p2ns

W_6p25ns

W_12p5ns

W_25ns

W_50ns

W_100ns

W_200ns

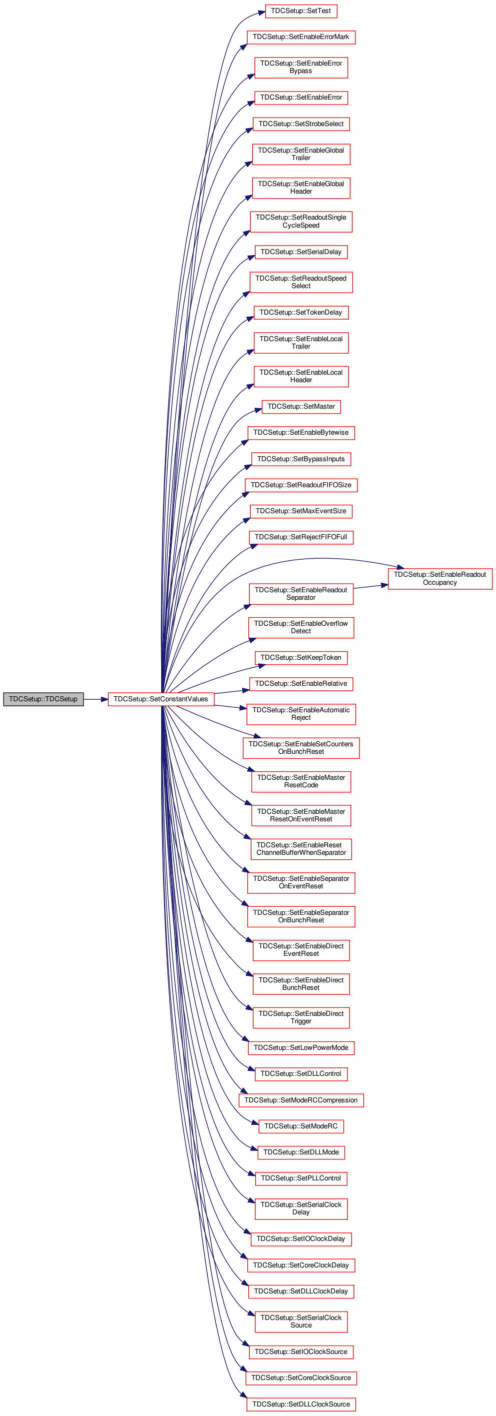
W_400ns

W_800ns

7.28.3 Constructor & Destructor Documentation

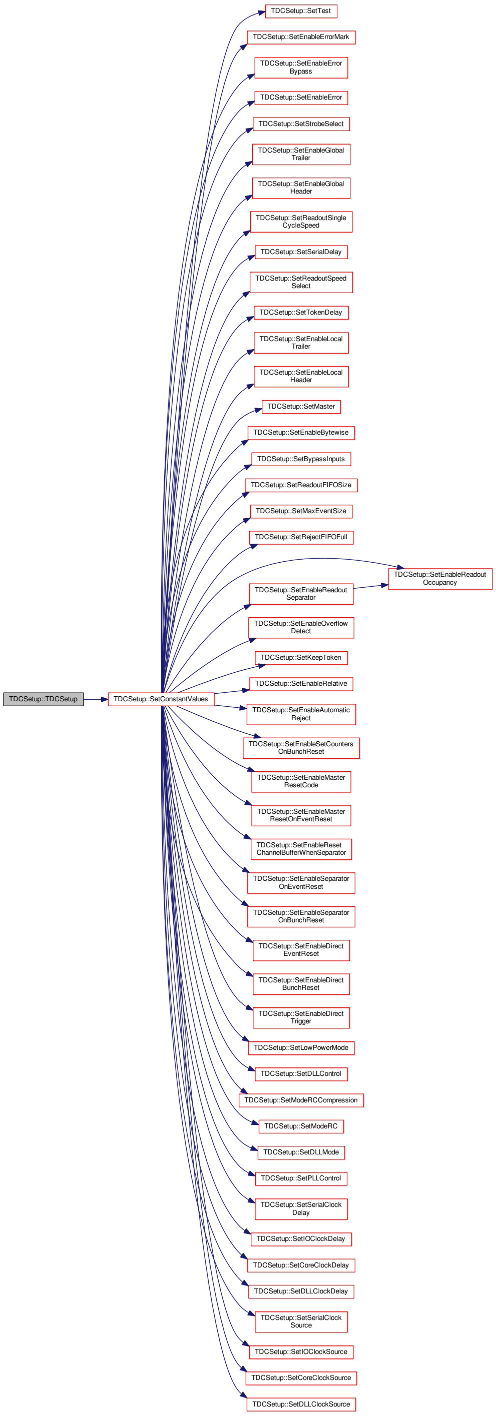
7.28.3.1 TDCSetup::TDCSetup ( ) [inline]

Here is the call graph for this function:


TDCSetup::SetTest
TDCSetup::SetEnableErrorMark
TDCSetup::SetEnableError Bypass
TDCSetup::SetEnableError
TDCSetup::SetStrobeSelect
TDCSetup::SetEnableGlobal Trailer
TDCSetup::SetEnableGlobal Header
TDCSetup::SetReadoutSingle CycleSpeed
TDCSetup::SetSerialDelay
TDCSetup::SetReadoutSpeed Select
TDCSetup::SetTokenDelay
TDCSetup::SetEnableLocal Trailer
TDCSetup::SetEnableLocal Header
TDCSetup::SetMaster
TDCSetup::SetEnableBytewise
TDCSetup::SetBypassInputs
TDCSetup::SetReadoutFIFOSize
TDCSetup::SetMaxEventSize
TDCSetup::SetRejectFIFOFull
TDCSetup::SetEnableReadout Occupancy
TDCSetup::SetEnableReadout Separator
TDCSetup::SetEnableOverflow Detect
TDCSetup::SetKeepToken
TDCSetup::TDCSetup
TDCSetup::SetConstantValues
TDCSetup::SetEnableRelative
TDCSetup::SetEnableAutomatic Reject
TDCSetup::SetEnableSetCounters OnBunchReset
TDCSetup::SetEnableMaster ResetCode
TDCSetup::SetEnableMaster ResetOnEventReset
TDCSetup::SetEnableReset ChannelBufferWhenSeparator
TDCSetup::SetEnableSeparator OnEventReset
TDCSetup::SetEnableSeparator OnBunchReset
TDCSetup::SetEnableDirect EventReset
TDCSetup::SetEnableDirect BunchReset
TDCSetup::SetEnableDirect Trigger
TDCSetup::SetLowPowerMode
TDCSetup::SetDLLControl
TDCSetup::SetModeRCCompression
TDCSetup::SetModeRC
TDCSetup::SetDLLMode
TDCSetup::SetPLLControl
TDCSetup::SetSerialClock Delay
TDCSetup::SetIOClockDelay
TDCSetup::SetCoreClockDelay
TDCSetup::SetDLLClockDelay
TDCSetup::SetSerialClock Source
TDCSetup::SetIOClockSource
TDCSetup::SetCoreClockSource
TDCSetup::SetDLLClockSource

7.28.3.2 TDCSetup::TDCSetup (const TDCSetup & *c*) [inline]

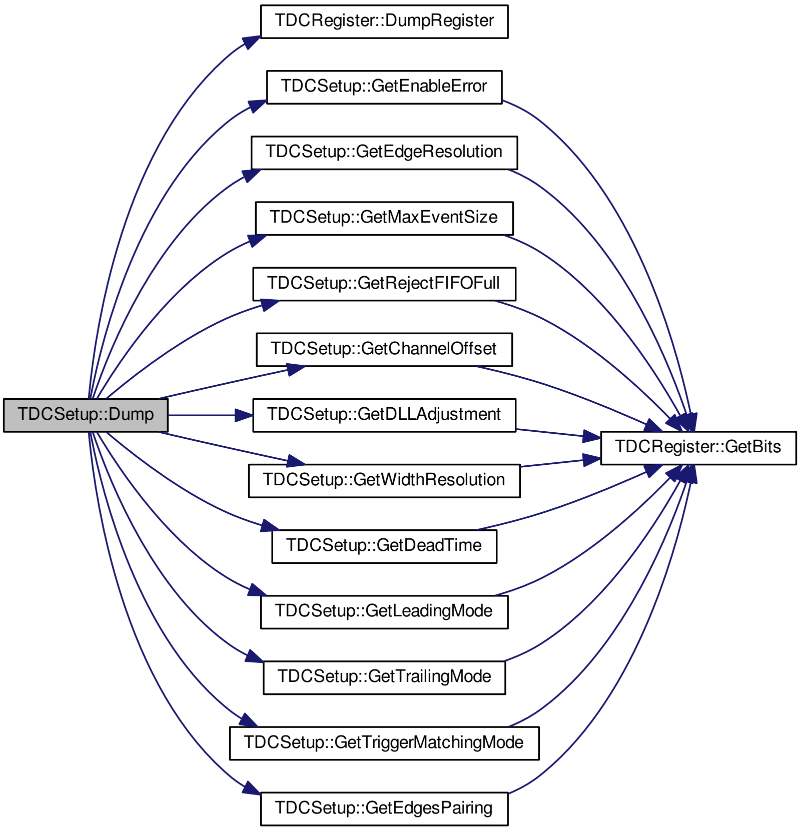
Here is the call graph for this function:


TDCSetup::TDCSetup
TDCSetup::SetConstantValues
TDCSetup::SetTest
TDCSetup::SetEnableErrorMark
TDCSetup::SetEnableError Bypass
TDCSetup::SetEnableError
TDCSetup::SetStrobeSelect
TDCSetup::SetEnableGlobal Trailer
TDCSetup::SetEnableGlobal Header
TDCSetup::SetReadoutSingle CycleSpeed
TDCSetup::SetSerialDelay
TDCSetup::SetReadoutSpeed Select
TDCSetup::SetTokenDelay
TDCSetup::SetEnableLocal Trailer
TDCSetup::SetEnableLocal Header
TDCSetup::SetMaster
TDCSetup::SetEnableBytewise
TDCSetup::SetBypassInputs
TDCSetup::SetReadoutFIFOSize
TDCSetup::SetMaxEventSize
TDCSetup::SetRejectFIFOFull
TDCSetup::SetEnableReadout Occupancy
TDCSetup::SetEnableReadout Separator
TDCSetup::SetEnableOverflow Detect
TDCSetup::SetKeepToken
TDCSetup::SetEnableRelative
TDCSetup::SetEnableAutomatic Reject
TDCSetup::SetEnableSetCounters OnBunchReset
TDCSetup::SetEnableMaster ResetCode
TDCSetup::SetEnableMaster ResetOnEventReset
TDCSetup::SetEnableReset ChannelBufferWhenSeparator
TDCSetup::SetEnableSeparator OnEventReset
TDCSetup::SetEnableSeparator OnBunchReset
TDCSetup::SetEnableDirect EventReset
TDCSetup::SetEnableDirect BunchReset
TDCSetup::SetEnableDirect Trigger
TDCSetup::SetLowPowerMode
TDCSetup::SetDLLControl
TDCSetup::SetModeRCCompression
TDCSetup::SetModeRC
TDCSetup::SetDLLMode
TDCSetup::SetPLLControl
TDCSetup::SetSerialClock Delay
TDCSetup::SetIOClockDelay
TDCSetup::SetCoreClockDelay
TDCSetup::SetDLLClockDelay
TDCSetup::SetSerialClock Source
TDCSetup::SetIOClockSource
TDCSetup::SetCoreClockSource
TDCSetup::SetDLLClockSource

7.28.4 Member Function Documentation

7.28.4.1 void TDCSetup::Dump (int *verb* = 1, std::ostream & *os* = std::cout) const

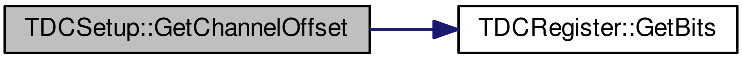
Here is the call graph for this function:


TDCRegister::DumpRegister
TDCSetup::GetEnableError
TDCSetup::GetEdgeResolution
TDCSetup::GetMaxEventSize
TDCSetup::GetRejectFIFOFull
TDCSetup::GetChannelOffset
TDCSetup::Dump
TDCSetup::GetDLLAdjustment
TDCRegister::GetBits
TDCSetup::GetWidthResolution
TDCSetup::GetDeadTime
TDCSetup::GetLeadingMode
TDCSetup::GetTrailingMode
TDCSetup::GetTriggerMatchingMode
TDCSetup::GetEdgesPairing

7.28.4.2 uint16_t TDCSetup::GetChannelOffset (int *channel*) const [inline]

Return the offset for one single channel.

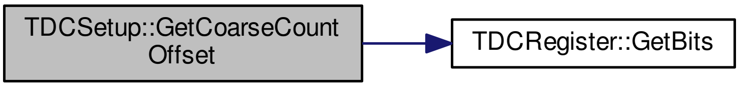
Here is the call graph for this function:



7.28.4.3 uint16_t TDCSetup::GetCoarseCountOffset ( ) const [inline]

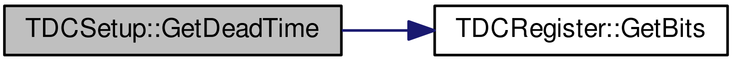
Extract offset for the coarse time counter.

Here is the call graph for this function:



7.28.4.4 DeadTime TDCSetup::GetDeadTime ( ) const [inline]

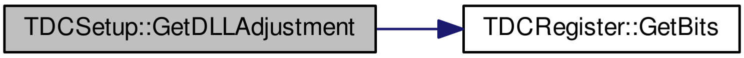
Here is the call graph for this function:



7.28.4.5 uint8_t TDCSetup::GetDLLAdjustment (int *tap*) const [inline]

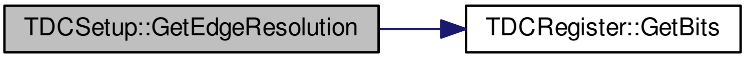
Set the adjustment of DLL taps.

Here is the call graph for this function:



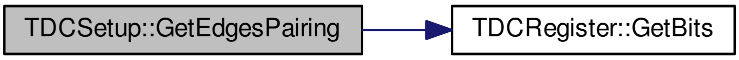
7.28.4.6 EdgeResolution TDCSetup::GetEdgeResolution ( ) const [inline]

Here is the call graph for this function:



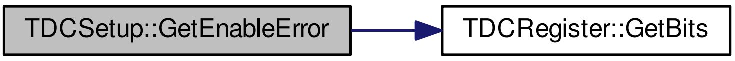
7.28.4.7 bool TDCSetup::GetEdgesPairing ( ) const [inline]

Here is the call graph for this function:



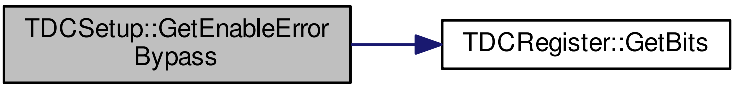
7.28.4.8 uint16_t TDCSetup::GetEnableError ( ) const [inline]

Here is the call graph for this function:



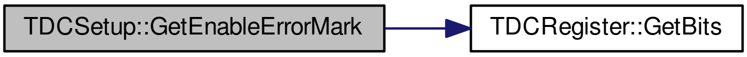
7.28.4.9 bool TDCSetup::GetEnableErrorBypass ( ) const [inline]

Here is the call graph for this function:



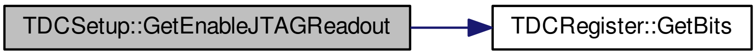
7.28.4.10 bool TDCSetup::GetEnableErrorMark ( ) const [inline]

Here is the call graph for this function:



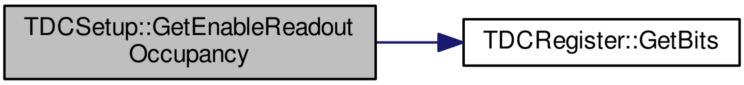
7.28.4.11 bool TDCSetup::GetEnableJTAGReadout ( ) const [inline]

Here is the call graph for this function:



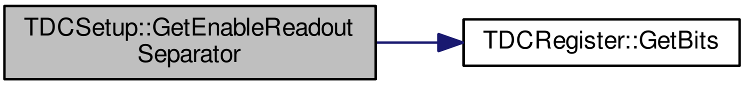
7.28.4.12 bool TDCSetup::GetEnableReadoutOccupancy ( ) const [inline]

Here is the call graph for this function:



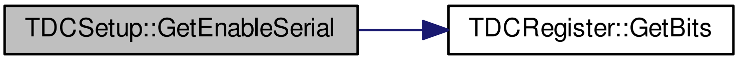
7.28.4.13 bool TDCSetup::GetEnableReadoutSeparator ( ) const [inline]

Here is the call graph for this function:



7.28.4.14 bool TDCSetup::GetEnableSerial ( ) const [inline]

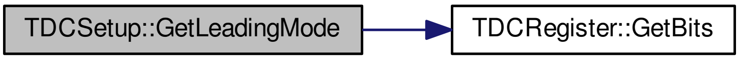
Here is the call graph for this function:



7.28.4.15 bool TDCSetup::GetLeadingMode ( ) const [inline]

Extract the status for the detection of leading edges.

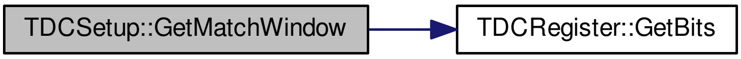
Here is the call graph for this function:



7.28.4.16 uint16_t TDCSetup::GetMatchWindow ( ) const [inline]

Extract the matching window (in multiples of clock cycles: 0=25 ns, 1=50 ns, ...)

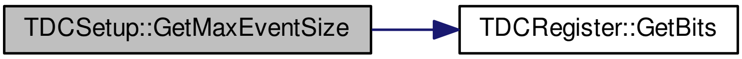
Here is the call graph for this function:



7.28.4.17 uint8_t TDCSetup::GetMaxEventSize ( ) const [inline]

Extract the maximum number of hits per event.

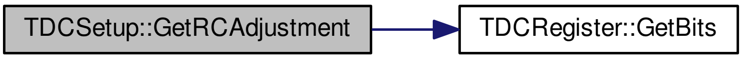
Here is the call graph for this function:



7.28.4.18 uint8_t TDCSetup::GetRCAdjustment (int *tap*) [inline]

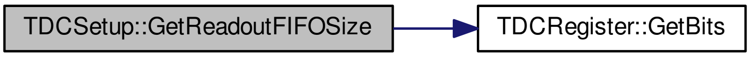
Extract the adjustment of the RC delay line.

Here is the call graph for this function:



7.28.4.19 int TDCSetup::GetReadoutFIFOSize ( ) const [inline]

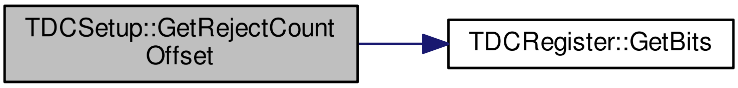
Here is the call graph for this function:



7.28.4.20 uint16_t TDCSetup::GetRejectCountOffset ( ) const [inline]

Extract the offset in reject counter.

Here is the call graph for this function:

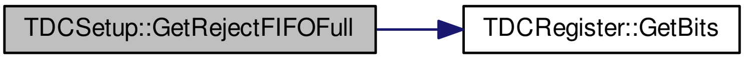


7.28.4.21 bool TDCSetup::GetRejectFIFOFull ( ) const [inline]

Are hits rejected when readout FIFO is full?

Extract whether or not hits are rejected once FIFO is full.

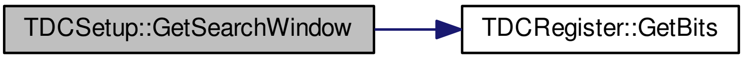
Here is the call graph for this function:



7.28.4.22 uint16_t TDCSetup::GetSearchWindow ( ) const [inline]

Extract the search window (in multiples of clock cycles: 0=25 ns, 1=50 ns, ...)

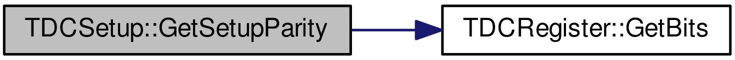
Here is the call graph for this function:



7.28.4.23 bool TDCSetup::GetSetupParity ( ) const [inline]

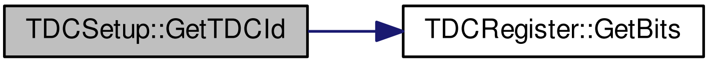
Extract the parity of setup data (should be an even parity)

Here is the call graph for this function:



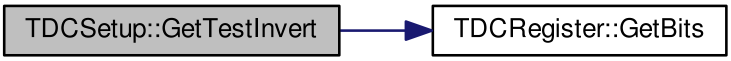
7.28.4.24 uint16_t TDCSetup::GetTDCId ( ) const [inline]

Here is the call graph for this function:



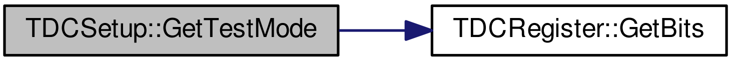
7.28.4.25 bool TDCSetup::GetTestInvert ( ) const [inline]

Here is the call graph for this function:



7.28.4.26 bool TDCSetup::GetTestMode ( ) const [inline]

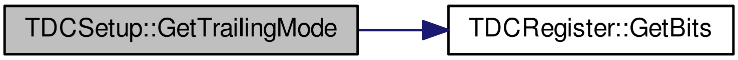
Here is the call graph for this function:



7.28.4.27 bool TDCSetup::GetTrailingMode ( ) const [inline]

Extract the status for the detection of trailing edges.

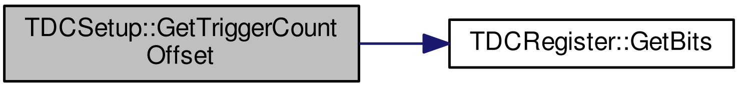
Here is the call graph for this function:



7.28.4.28 uint16_t TDCSetup::GetTriggerCountOffset ( ) const [inline]

Extract trigger time tag count offset.

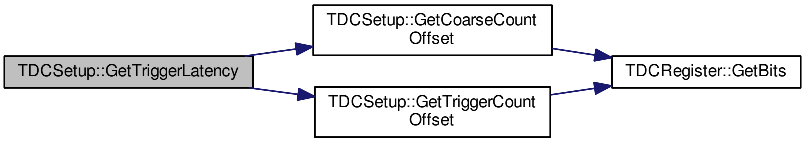
Here is the call graph for this function:



7.28.4.29 uint16_t TDCSetup::GetTriggerLatency ( ) const [inline]

Effective trigger latency in number of clock cycles (when no counter roll-over is used)

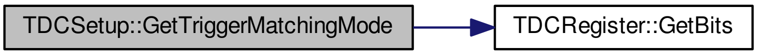
Here is the call graph for this function:


TDCSetup::GetTriggerLatency
TDCSetup::GetCoarseCount Offset
TDCSetup::GetTriggerCount Offset
TDCRegister::GetBits

7.28.4.30 bool TDCSetup::GetTriggerMatchingMode ( ) const [inline]

Extract the enable status of trigger matching mode.

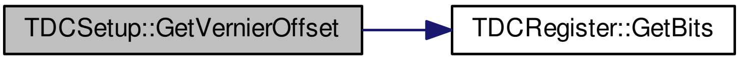
Here is the call graph for this function:



7.28.4.31 uint8_t TDCSetup::GetVernierOffset ( ) const [inline]

Extract the offset in vernier decoding.

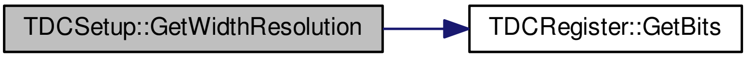
Here is the call graph for this function:



7.28.4.32 WidthResolution TDCSetup::GetWidthResolution ( ) const [inline]

Extract the pulse width resolution when paired measurements are performed.

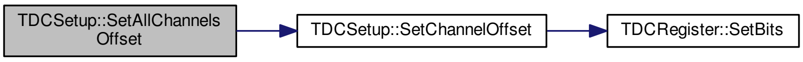
Here is the call graph for this function:



7.28.4.33 void TDCSetup::SetAllChannelsOffset (uint16_t *offset*) [inline]

Set the time offset for all channels.

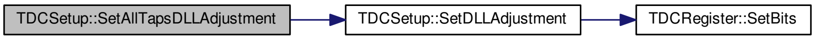
Here is the call graph for this function:



7.28.4.34 void TDCSetup::SetAllTapsDLLAdjustment (uint8_t *adj*) [inline]

Extract the adjustment of DLL taps.

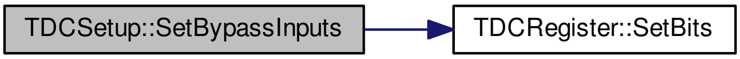
Here is the call graph for this function:



7.28.4.35 void TDCSetup::SetBypassInputs (const bool *sbi* = true) [inline], [private]

Select serial in and token in from bypass inputs.

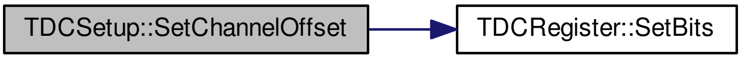
Here is the call graph for this function:



7.28.4.36 void TDCSetup::SetChannelOffset (int *channel*, uint16_t *offset*) [inline]

Set the time offset for one single channel.

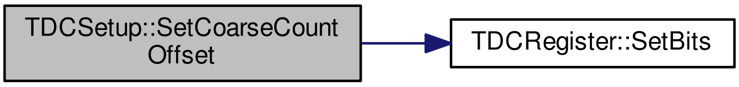
Here is the call graph for this function:



7.28.4.37 void TDCSetup::SetCoarseCountOffset (uint16_t *cco*) [inline]

Set offset for the coarse time counter.

Here is the call graph for this function:



7.28.4.38 void TDCSetup::SetConstantValues ( ) [virtual]

Ensure that the critical constant values are properly set in the setup word.

Implements TDCRegister.

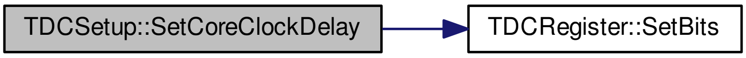
7.28.4.39 void TDCSetup::SetCoreClockDelay (const bool *delay_clock*, const uint8_t *delay*) [inline], [private]

Delay of internal core clock.

Parameters

in	*delay_clock*	Use of direct clock (0) or delayed clock (1)
in	*delay*	Delay in steps of (typically) 0.13 ns

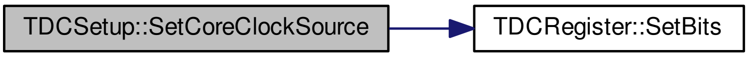
Here is the call graph for this function:



7.28.4.40 void TDCSetup::SetCoreClockSource (const CoreClockSource *ccs*) [inline], [private]

Selection of clock source for internal logic.

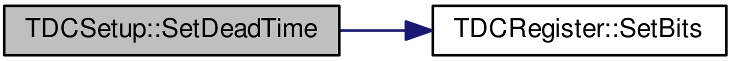
Here is the call graph for this function:



7.28.4.41 void TDCSetup::SetDeadTime (const DeadTime *dt*) [inline]

Channel dead time between hits.

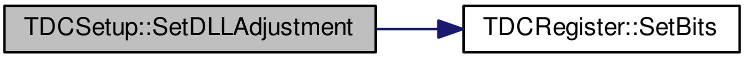
Here is the call graph for this function:



7.28.4.42 void TDCSetup::SetDLLAdjustment (int *tap*, uint8_t *adj*) [inline]

Set the DLL taps adjustments with a resolution of ∼10 ps.

Here is the call graph for this function:



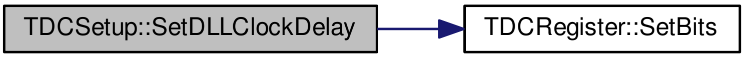
7.28.4.43 void TDCSetup::SetDLLClockDelay (const bool *delay_clock*, const uint8_t *delay*) [inline], [private]

Delay of internal DLL clock.

Parameters

in	*delay_clock*	Use of direct clock (0) or delayed clock (1)
in	*delay*	Delay in steps of (typically) 0.13 ns

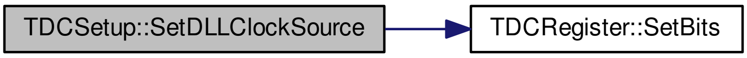
Here is the call graph for this function:



7.28.4.44 void TDCSetup::SetDLLClockSource (const DLLClockSource *dcs*) [inline], [private]

Selection of clock source for DLL.

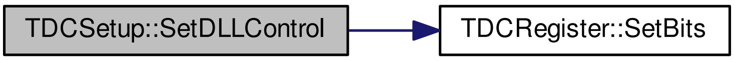
Here is the call graph for this function:



7.28.4.45 void TDCSetup::SetDLLControl (const uint8_t *dc*) [inline], [private]

Control of DLL (DLL charge pump levels)

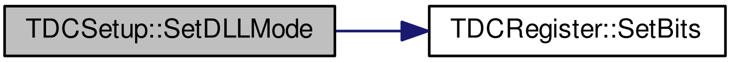
Here is the call graph for this function:



7.28.4.46 void TDCSetup::SetDLLMode (const **DLLSpeedMode** *dsm*) [inline], [private]

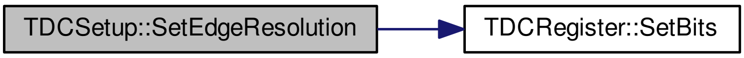
Selection of DLL speed mode.

Here is the call graph for this function:



7.28.4.47 void TDCSetup::SetEdgeResolution (const EdgeResolution *r*) [inline]

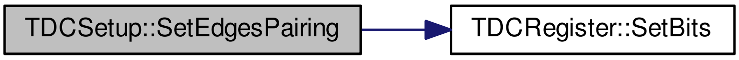
Here is the call graph for this function:



7.28.4.48 void TDCSetup::SetEdgesPairing (const bool *pair* = true) [inline]

Enable the pairing of leading and trailing edges (overrides individual enable of leading/trailing edges)

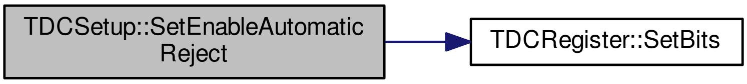
Here is the call graph for this function:



7.28.4.49 void TDCSetup::SetEnableAutomaticReject (const bool *ear* = true) [inline], [private]

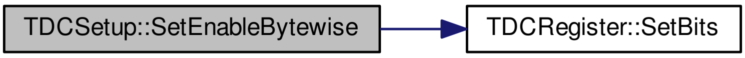
Enable of automatic rejection (should always be enabled if trigger matching mode!)

Here is the call graph for this function:



7.28.4.50 void TDCSetup::SetEnableBytewise (const bool *seb* = `true`) `[inline]`, `[private]`

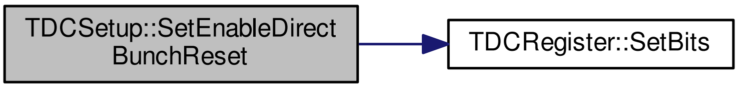
Here is the call graph for this function:



7.28.4.51 void TDCSetup::SetEnableDirectBunchReset (const bool *edbr* = `true`) `[inline]`, `[private]`

Enable of direct bunch reset input pin (1), otherwise taken from encoded control.

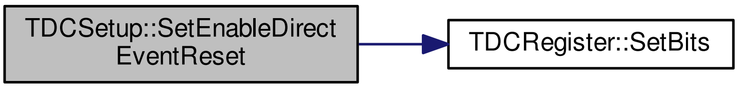
Here is the call graph for this function:



7.28.4.52 void TDCSetup::SetEnableDirectEventReset (const bool *eder* = `true`) `[inline]`, `[private]`

Enable of direct event reset input pin (1), otherwise taken from encoded control.

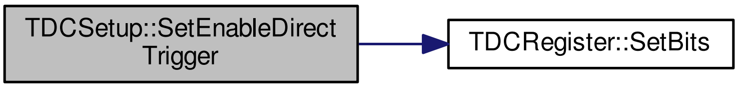
Here is the call graph for this function:



7.28.4.53 void TDCSetup::SetEnableDirectTrigger (const bool *edt* = `true`) `[inline]`, `[private]`

Enable of direct trigger input pin.

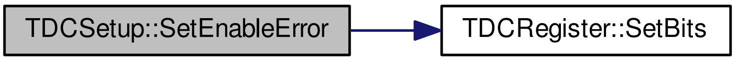
Here is the call graph for this function:



7.28.4.54 void TDCSetup::SetEnableError (const uint16_t & *err*) [inline]

Enable internal error types for generation of global error signals.

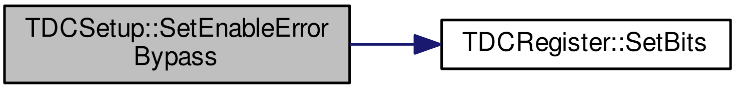
Here is the call graph for this function:



7.28.4.55 void TDCSetup::SetEnableErrorBypass (const bool *eb*) [inline]

Bypass TDC chip if global error signal is set.

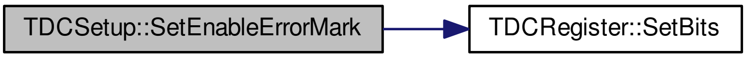
Here is the call graph for this function:



7.28.4.56 void TDCSetup::SetEnableErrorMark (const bool *em*) [inline]

Mark events with error if global error signal is set.

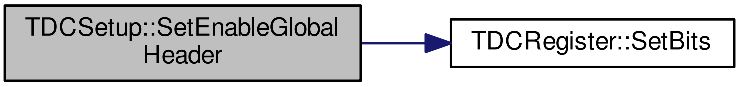
Here is the call graph for this function:



7.28.4.57 void TDCSetup::SetEnableGlobalHeader (const bool *egh* = `true`) `[inline]`, `[private]`

Enable of global headers in read-out (only valid for master TDC)

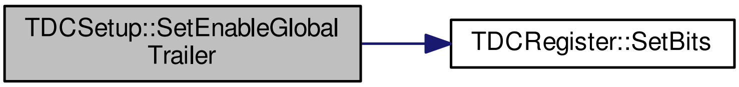
Here is the call graph for this function:



7.28.4.58 void TDCSetup::SetEnableGlobalTrailer (const bool *egt* = `true`) `[inline]`, `[private]`

Enable of global trailers in read-out (only valid for master TDC)

Here is the call graph for this function:



7.28.4.59 void TDCSetup::SetEnableJTAGReadout (const bool *jr*) `[inline]`

Enable of read-out via JTAG.

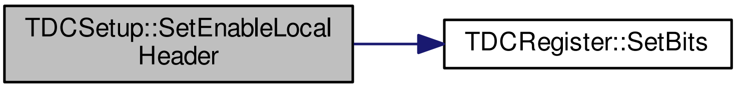
Here is the call graph for this function:

TDCSetup::SetEnableJTAGReadout
TDCRegister::SetBits

7.28.4.60 void TDCSetup::SetEnableLocalHeader (const bool *elh* = `true`) [inline], [private]

Enable of local headers in read-out.

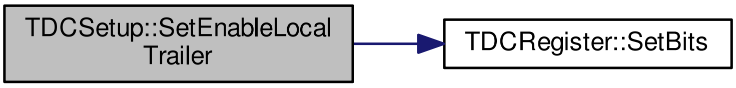
Here is the call graph for this function:



7.28.4.61 void TDCSetup::SetEnableLocalTrailer (const bool *elt* = `true`) [inline], [private]

Enable of local trailers in read-out.

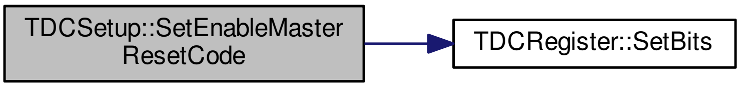
Here is the call graph for this function:



7.28.4.62 void TDCSetup::SetEnableMasterResetCode (const bool *emrc* = `true`) [inline], [private]

Enable master reset code on encoded_control.

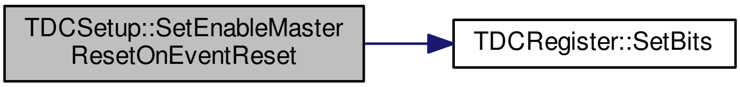
Here is the call graph for this function:



7.28.4.63 void TDCSetup::SetEnableMasterResetOnEventReset (const bool *emroer* = `true`) `[inline]`, `[private]`

Enable master reset of whole TDC on event reset.

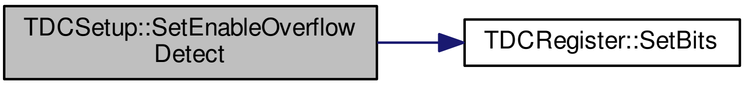
Here is the call graph for this function:



7.28.4.64 void TDCSetup::SetEnableOverflowDetect (const bool *eod* = `true`) `[inline]`, `[private]`

Enable overflow detection of L1 buffers (should always be enabled!)

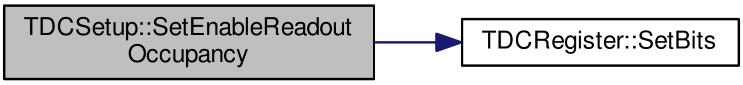
Here is the call graph for this function:



7.28.4.65 void TDCSetup::SetEnableReadoutOccupancy (const bool *ro* = `true`) `[inline]`

Enable the readout of buffer occupancies for each event (for debugging purposes)

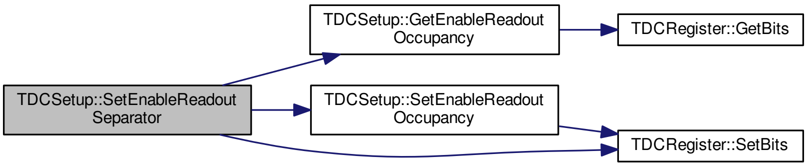
Here is the call graph for this function:



7.28.4.66 void TDCSetup::SetEnableReadoutSeparator (const bool *ro* = `true`) `[inline]`

Enable the readout of separators for each event (for debugging purposes, valid if readout of occupancies is enabled)

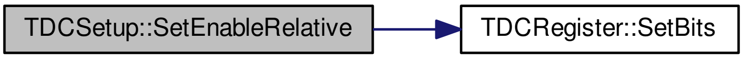
Here is the call graph for this function:


TDCSetup::SetEnableReadout
Separator
TDCSetup::GetEnableReadout
Occupancy
TDCRegister::GetBits
TDCSetup::SetEnableReadout
Occupancy
TDCRegister::SetBits

7.28.4.67 void TDCSetup::SetEnableRelative (const bool *er* = `true`) `[inline]`, `[private]`

Enable read-out of relative time to trigger time tag. Only valid when using trigger matching mode.

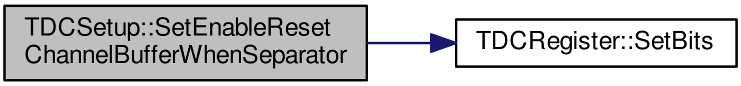
Here is the call graph for this function:



7.28.4.68 void TDCSetup::SetEnableResetChannelBufferWhenSeparator (const bool *ercbws* = `true`) `[inline]`, `[private]`

Enable reset channel buffers when separator.

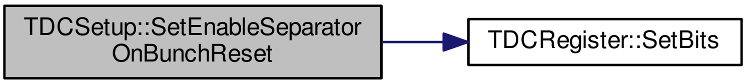
Here is the call graph for this function:



7.28.4.69 void TDCSetup::SetEnableSeparatorOnBunchReset (const bool *esobr* = `true`) `[inline]`, `[private]`

Enable generation of separator on bunch reset.

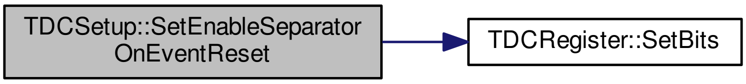
Here is the call graph for this function:



7.28.4.70 void TDCSetup::SetEnableSeparatorOnEventReset (const bool *esoer* = `true`) `[inline]`, `[private]`

Enable generation of separator on event reset.

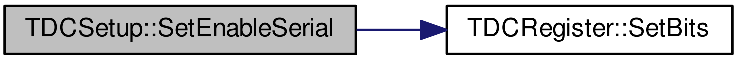
Here is the call graph for this function:



7.28.4.71 void TDCSetup::SetEnableSerial (const bool *es*) `[inline]`

Enable of serial read-out (otherwise parallel read-out)

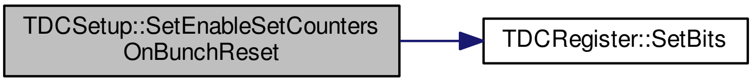
Here is the call graph for this function:



7.28.4.72 void TDCSetup::SetEnableSetCountersOnBunchReset (const bool *escobr* = true) [inline], [private]

Enable all counters to be set on bunch count reset.

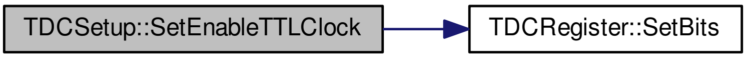
Here is the call graph for this function:



7.28.4.73 void TDCSetup::SetEnableTTLClock (const bool *tc* = true) [inline], [private]

Enable LV TTL inputs on: clk, aux_clock, otherwise uses LVDS input levels.

Here is the call graph for this function:



7.28.4.74 void TDCSetup::SetEnableTTLControl (const bool *tc* = true) [inline], [private]

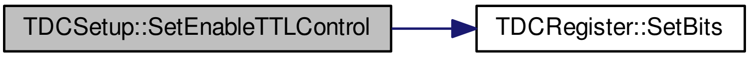
Enable LV TTL inputs on control registers.

Enable LV TTL input on:

- trigger,
- bunch_reset,
- event_reset,

- encoded_control, otherwise uses LVDS input levels.

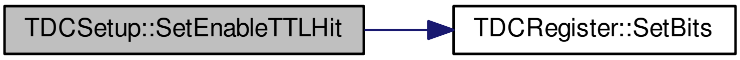
Here is the call graph for this function:



7.28.4.75 void TDCSetup::SetEnableTTLHit (const bool *th* = true) [inline], [private]

Enable LV TTL input on hit[31:0], otherwise uses LVDS input levels.

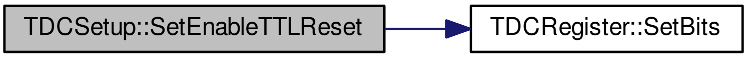
Here is the call graph for this function:



7.28.4.76 void TDCSetup::SetEnableTTLReset (const bool *tr* = true) [inline], [private]

Enable LV TTL input on reset, otherwise uses LVDS input levels.

Here is the call graph for this function:



7.28.4.77 void TDCSetup::SetEnableTTLSerial (const bool *ts* = true) [inline], [private]

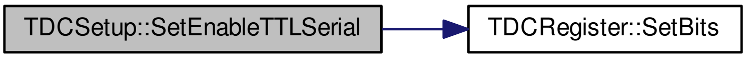
Enable LV TTL inputs on serial registers, and disable their drivers.

Enable LV TTL input on:

- serial_in,
- serial_bypass_in,

- token_in,
- token_bypass_in, otherwise uses LVDS input levels. Disable LVDS drivers on:
- serial_out,
- strobe_out,
- token_out.

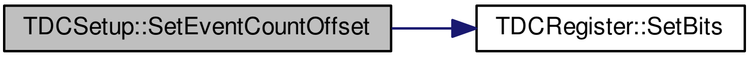
Here is the call graph for this function:



7.28.4.78 void TDCSetup::SetEventCountOffset (uint16_t *eco*) [inline]

Set offset for the event counter.

Here is the call graph for this function:



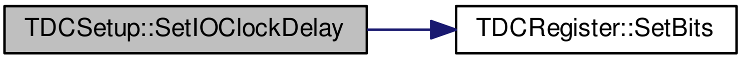
7.28.4.79 void TDCSetup::SetIOClockDelay (const bool *delay_clock,* const uint8_t *delay*) [inline], [private]

Delay of internal I/O clock.

Parameters

in	*delay_clock*	Use of direct clock (0) or delayed clock (1)
in	*delay*	Delay in steps of (typically) 0.13 ns

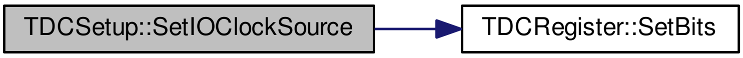
Here is the call graph for this function:



7.28.4.80 void TDCSetup::SetIOClockSource (const IOClockSource *ics*) [inline], [private]

Selection of clock source for I/O signals.

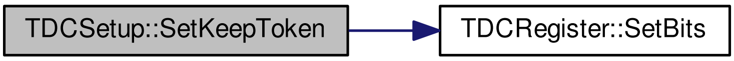
Here is the call graph for this function:



7.28.4.81 void TDCSetup::SetKeepToken (const bool *kt* = true) [inline], [private]

Keep token until end of event or no more data, otherwise pass token after each word read. Must be enabled when using trigger matching.

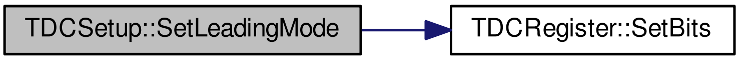
Here is the call graph for this function:



7.28.4.82 void TDCSetup::SetLeadingMode (const bool *lead* = true) [inline]

Enable the detection of leading edges.

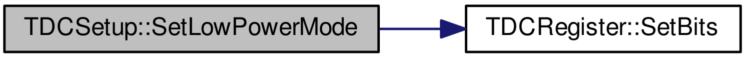
Here is the call graph for this function:



7.28.4.83 void TDCSetup::SetLowPowerMode (const bool *lpm* = true) [inline], [private]

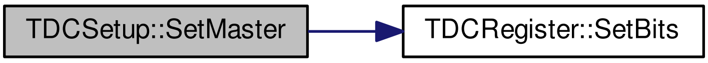
Low power mode of channel buffers.

Here is the call graph for this function:



7.28.4.84 void TDCSetup::SetMaster (const bool *m* = `true`) `[inline]`, `[private]`

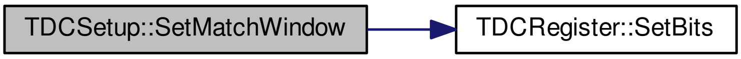
Here is the call graph for this function:



7.28.4.85 void TDCSetup::SetMatchWindow (uint16_t *mw*) `[inline]`

Set the matching window (in multiples of clock cycles: 0=25 ns, 1=50 ns, ...)

Here is the call graph for this function:

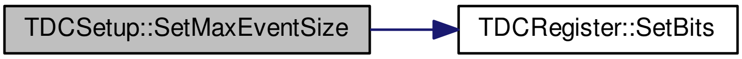


7.28.4.86 void TDCSetup::SetMaxEventSize (int *sz* = `-1`) `[inline]`

Set the maximum number of hits per event.

Set the maximum number of hits that can be recorded for each event. It is always rounded to the next power of 2 (in the range 0-128), and if lower than 0 or bigger than 128 then set to unimited.

Here is the call graph for this function:



7.28.4.87 void TDCSetup::SetModeRC (const bool *mr* = `true`) `[inline]`, `[private]`

Enable of RR delay lines mode (in very high resolution mode) ; only for channels 0-4-8-12-16-20-24-28 active.

Here is the call graph for this function:

TDCSetup::SetModeRC
TDCRegister::SetBits

7.28.4.88 void TDCSetup::SetModeRCCompression (const bool *mrc* = `true`) `[inline]`, `[private]`

Perform RC interpolation on-chip (only valid in very high resolution mode)

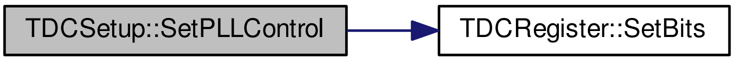
Here is the call graph for this function:

TDCSetup::SetModeRCCompression
TDCRegister::SetBits

7.28.4.89 void TDCSetup::SetPLLControl (const uint8_t *charge_pump_current* = `0x4`, const bool *power_down_mode* = `false`, const bool *enable_test_outputs* = `false`, const bool *invert_connection_to_status* = `false`) `[inline]`, `[private]`

Control of PLL.

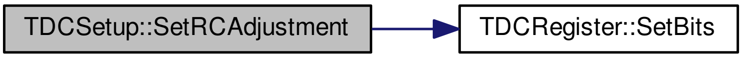
Here is the call graph for this function:



7.28.4.90 void TDCSetup::SetRCAdjustment (int *tap*, uint8_t *adj*) [inline]

Set the adjustment of the RC delay line.

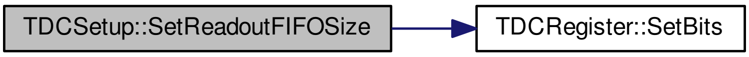
Here is the call graph for this function:



7.28.4.91 void TDCSetup::SetReadoutFIFOSize (int *rfs*) [inline]

Effective size of readout FIFO.

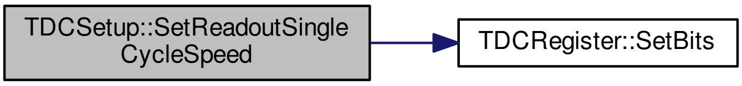
Here is the call graph for this function:



7.28.4.92 void TDCSetup::SetReadoutSingleCycleSpeed (const ReadoutSingleCycleSpeed *rscs* = RSC_40Mbits_s) [inline], [private]

Serial transmission speed in single cycle mode.

Here is the call graph for this function:



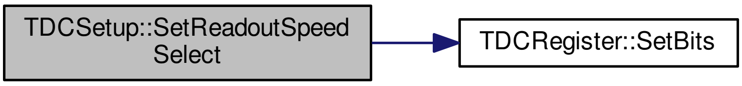
7.28.4.93 void TDCSetup::SetReadoutSpeedSelect (const ReadoutSpeed *rss* = RO_Fixed) [inline], [private]

Selection of serial read-out speed.

Parameters

in	*rss*	- 0: Selection of serial read-out speed (as defined by setup[19:17], *Set←ReadoutSingleCycleSpeed*) - 1: 80 Mbits/s (PLL lock required)

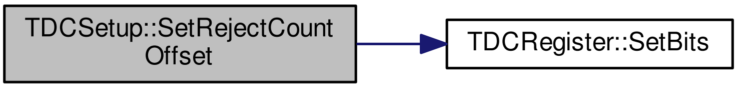
Here is the call graph for this function:



7.28.4.94 void TDCSetup::SetRejectCountOffset (uint16_t *rco*) [inline]

Set the offset in reject counter (defines reject latency together with coarse count offset)

Here is the call graph for this function:

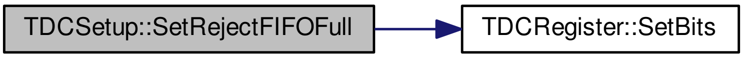


7.28.4.95 void TDCSetup::SetRejectFIFOFull (const bool *rej* = true) [inline]

Reject hits when readout FIFO full.

Set whether or not hits are rejected once FIFO is full.

Here is the call graph for this function:



7.28.4.96 void TDCSetup::SetRollOver (const uint16_t *ro* = 0xFFF) [inline], [private]

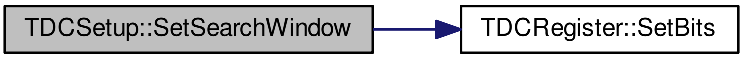
Counter roll over value, defining maximal count value from where counters will be reset to 0.

Here is the call graph for this function:

7.28.4.97 void TDCSetup::SetSearchWindow (uint16_t *sw*) [inline]

Set the search window (in multiples of clock cycles: 0=25 ns, 1=50 ns, ...)

Here is the call graph for this function:



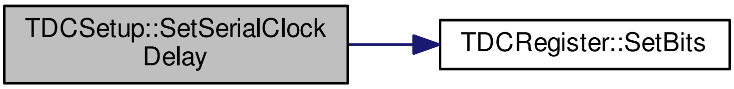
7.28.4.98 void TDCSetup::SetSerialClockDelay (const bool *delay_clock,* const uint8_t *delay*) [inline], [private]

Delay of internal serial clock.

Parameters

in	*delay_clock*	Use of direct clock (0) or delayed clock (1)
in	*delay*	Delay in steps of (typically) 0.13 ns

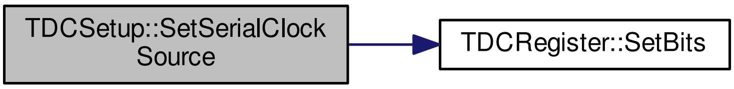
Here is the call graph for this function:



7.28.4.99 void TDCSetup::SetSerialClockSource (const SerialClockSource *scs*) [inline], [private]

Selection of source for serial clock.

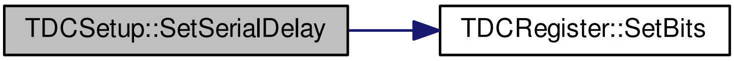
Here is the call graph for this function:



7.28.4.100 void TDCSetup::SetSerialDelay (const uint8_t *sd* = 0x0) [inline], [private]

Programmable delay of serial input, in time unit ∼ 1 ns.

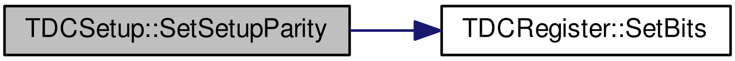
Here is the call graph for this function:



7.28.4.101 void TDCSetup::SetSetupParity (const bool *sp* = true) [inline]

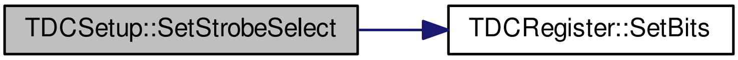
Set the parity of setup data (should be an even parity)

Here is the call graph for this function:



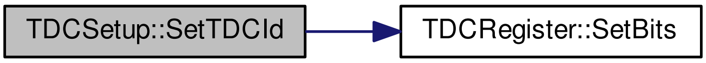
7.28.4.102 void TDCSetup::SetStrobeSelect (const SerialStrobeType *ss* = SS_NoStrobe) [inline], [private]

Here is the call graph for this function:



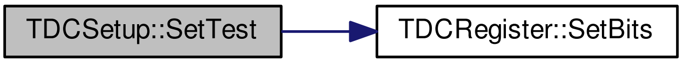
7.28.4.103 void TDCSetup::SetTDCId (const uint8_t *id* = 0x0) [inline]

Here is the call graph for this function:



7.28.4.104 void TDCSetup::SetTest (const bool *test* = `true`) `[inline]`, `[private]`

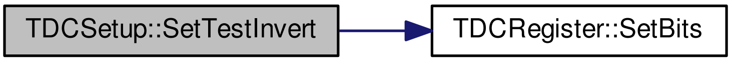
Here is the call graph for this function:



7.28.4.105 void TDCSetup::SetTestInvert (const bool *ti* = `true`) `[inline]`

Automatic inversion of test pattern. Only used during production testing.

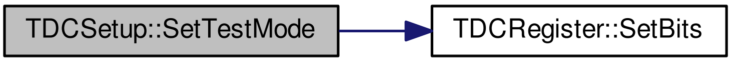
Here is the call graph for this function:



7.28.4.106 void TDCSetup::SetTestMode (const bool *tm* = `true`) `[inline]`

Test mode where hit data are taken from coretest. Only used during production testing.

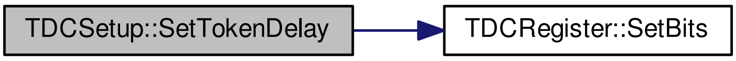
Here is the call graph for this function:



7.28.4.107 void TDCSetup::SetTokenDelay (const uint8_t *td* = `0x0`) `[inline]`, `[private]`

Programmable delay of token input, in time unit $\sim$ 1 ns.

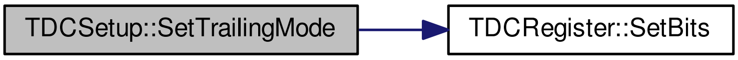
Here is the call graph for this function:



7.28.4.108 void TDCSetup::SetTrailingMode (const bool *trail* = `true`) `[inline]`

Enable/disable the detection of trailing edges.

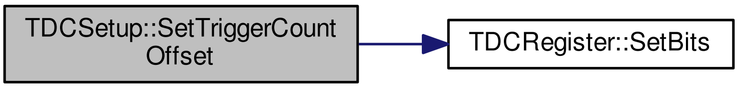
Here is the call graph for this function:



7.28.4.109 void TDCSetup::SetTriggerCountOffset (uint16_t *tco*) `[inline]`

Set offset for the trigger time tag counter to set effective trigger latency.

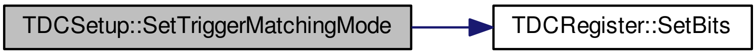
Here is the call graph for this function:



7.28.4.110 void TDCSetup::SetTriggerMatchingMode (const bool *trig* = `true`) `[inline]`

Set the enable status of trigger matching mode.

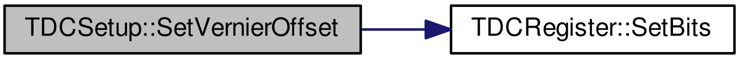
Here is the call graph for this function:



7.28.4.111 void TDCSetup::SetVernierOffset (const uint8_t *vo*) [inline]

Set the offset in vernier decoding.

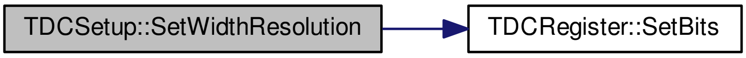
Here is the call graph for this function:



7.28.4.112 void TDCSetup::SetWidthResolution (const WidthResolution *r*) [inline]

Set the pulse width resolution when paired measurements are performed.

Here is the call graph for this function:



7.28.5 Field Documentation

7.28.5.1 const bit TDCSetup::kCoarseCountOffset = 447 [static], [private]

7.28.5.2 const bit TDCSetup::kCoreClockDelay = 610 [static], [private]

7.28.5.3 const bit TDCSetup::kCoreClockSource = 622 [static], [private]

7.28.5.4 const bit TDCSetup::kDeadTime = 584 [static], [private]

7.28.5.5 const bit TDCSetup::kDLLClockDelay = 614 [static], [private]

7.28.5.6 const bit TDCSetup::kDLLClockSource = 624 [static], [private]

7.28.5.7 const bit TDCSetup::kDLLControl = 580 [static], [private]

7.28.5.8 const bit TDCSetup::kDLLMode = 592 [static], [private]

7.28.5.9 const bit TDCSetup::kDLLTapAdjust0 = 459 [static], [private]

7.28.5.10 const bit TDCSetup::kEnableAutomaticReject = 125 [static], [private]

7.28.5.11 const bit TDCSetup::kEnableBytewise = 37 [static], [private]

7.28.5.12 const bit TDCSetup::kEnableDirectBunchReset = 157 [static], [private]

7.28.5.13 const bit TDCSetup::kEnableDirectEventReset = 156 [static], [private]

7.28.5.14 const bit TDCSetup::kEnableDirectTrigger = 158 [static], [private]

7.28.5.15 const bit TDCSetup::kEnableError = 6 [static], [private]

7.28.5.16 const bit TDCSetup::kEnableErrorBypass = 5 [static], [private]

7.28.5.17 const bit TDCSetup::kEnableErrorMark = 4 [static], [private]

7.28.5.18 const bit TDCSetup::kEnableGlobalHeader = 34 [static], [private]

7.28.5.19 const bit TDCSetup::kEnableGlobalTrailer = 33 [static], [private]

7.28.5.20 const bit TDCSetup::kEnableJTAGReadout = 39 [static], [private]

7.28.5.21 const bit TDCSetup::kEnableLocalHeader = 32 [static], [private]

7.28.5.22 const bit TDCSetup::kEnableLocalTrailer = 31 [static], [private]

7.28.5.23 const bit TDCSetup::kEnableMasterResetCode = 151 [static], [private]

7.28.5.24 const bit TDCSetup::kEnableMasterResetOnEventReset = 152 [static], [private]

7.28.5.25 const bit TDCSetup::kEnableMatching = 639 [static], [private]

7.28.5.26 const bit TDCSetup::kEnableOverflowDetect = 123 [static], [private]

7.28.5.27 const bit TDCSetup::kEnablePair = 640 [static], [private]

7.28.5.28 const bit TDCSetup::kEnableReadoutOccupancy = 121 [static], [private]

7.28.5.29 const bit TDCSetup::kEnableReadoutSeparator = 122 [static], [private]

7.28.5.30 const bit TDCSetup::kEnableRelative = 124 [static], [private]

7.28.5.31 const bit TDCSetup::kEnableResetChannelBufferWhenSeparator = 153 [static], [private]

7.28.5.32 const bit TDCSetup::kEnableSeparatorOnBunchReset = 155 [static], [private]

7.28.5.33 const bit TDCSetup::kEnableSeparatorOnEventReset = 154 [static], [private]

7.28.5.34 const bit TDCSetup::kEnableSerial = 38 [static], [private]

7.28.5.35 const bit TDCSetup::kEnableSetCountersOnBunchReset = 150 [static], [private]

7.28.5.36 const bit TDCSetup::kEnableTTLClock = 644 [static], [private]

7.28.5.37 const bit TDCSetup::kEnableTTLControl = 642 [static], [private]

7.28.5.38 const bit TDCSetup::kEnableTTLHit = 645 [static], [private]

7.28.5.39 const bit TDCSetup::kEnableTTLReset = 643 [static], [private]

7.28.5.40 const bit TDCSetup::kEnableTTLSerial = 641 [static], [private]

7.28.5.41 const bit TDCSetup::kEventCountOffset = 126 [static], [private]

7.28.5.42 const bit TDCSetup::kIOClockDelay = 606 [static], [private]

7.28.5.43 const bit TDCSetup::kIOClockSource = 620 [static], [private]

7.28.5.44 const bit TDCSetup::kKeepToken = 35 [static], [private]

7.28.5.45 const bit TDCSetup::kLeading = 589 [static], [private]

7.28.5.46 const bit TDCSetup::kLeadingResolution = 84 [static], [private]

7.28.5.47 const bit TDCSetup::kLowPowerMode = 570 [static], [private]

7.28.5.48 const bit TDCSetup::kMaster = 36 [static], [private]

7.28.5.49 const bit TDCSetup::kMatchWindow = 72 [static], [private]

7.28.5.50 const bit TDCSetup::kMaxEventSize = 116 [static], [private]

7.28.5.51 const bit TDCSetup::kModeRC = 591 [static], [private]

7.28.5.52 const bit TDCSetup::kModeRCCompression = 590 [static], [private]

7.28.5.53 const bit TDCSetup::kOffset0 = 438 [static], [private]

7.28.5.54 const bit TDCSetup::kPLLControl = 594 [static], [private]

7.28.5.55 const bit TDCSetup::kRCAdjust0 = 555 [static], [private]

7.28.5.56 const bit TDCSetup::kReadoutFIFOSize = 45 [static], [private]

7.28.5.57 const bit TDCSetup::kReadoutSingleCycleSpeed = 17 [static], [private]

7.28.5.58 const bit TDCSetup::kReadoutSpeedSelect = 26 [static], [private]

7.28.5.59 const bit TDCSetup::kRejectCountOffset = 48 [static], [private]

7.28.5.60 const bit TDCSetup::kRejectFIFOFull = 120 [static], [private]

7.28.5.61 const bit TDCSetup::kRollOver = 627 [static], [private]

7.28.5.62 const bit TDCSetup::kSearchWindow = 60 [static], [private]

7.28.5.63 const bit TDCSetup::kSelectBypassInputs = 44 [static], [private]

7.28.5.64 const bit TDCSetup::kSerialClockDelay = 602 [static], [private]

7.28.5.65 const bit TDCSetup::kSerialClockSource = 618 [static], [private]

7.28.5.66 const bit TDCSetup::kSerialDelay = 20 [static], [private]

7.28.5.67 const bit TDCSetup::kSetupParity = 646 [static], [private]

7.28.5.68 const bit TDCSetup::kStrobeSelect = 24 [static], [private]

7.28.5.69 const bit TDCSetup::kTDCId = 40 [static], [private]

7.28.5.70 const bit TDCSetup::kTestInvert = 586 [static], [private]

7.28.5.71 const bit TDCSetup::kTestMode = 587 [static], [private]

7.28.5.72 const bit TDCSetup::kTestSelect = 0 [static], [private]

7.28.5.73 const bit TDCSetup::kTokenDelay = 27 [static], [private]

7.28.5.74 const bit TDCSetup::kTrailing = 588 [static], [private]

7.28.5.75 const bit TDCSetup::kTriggerCountOffset = 138 [static], [private]

7.28.5.76 const bit TDCSetup::kVernierOffset = 575 [static], [private]

7.28.5.77 const bit TDCSetup::kWidthSelect = 571 [static], [private]

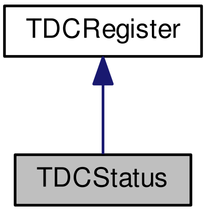
The documentation for this class was generated from the following files:

- daq/include/TDCSetup.h
- daq/src/TDCSetup.cpp

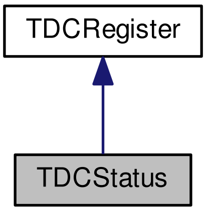
7.29 TDCStatus Class Reference

```
#include <TDCStatus.h>
```

Inheritance diagram for TDCStatus:



Collaboration diagram for TDCStatus:



Public Member Functions

- TDCStatus ()
- TDCStatus (const TDCStatus &s)
- TDCStatus (const std::vector< uint8_t > &words)
- void SetConstantValues ()
- uint16_t Error () const
- bool HaveToken () const
- uint8_t FIFOOccupancy () const
- bool FIFOFull () const
- bool FIFOEmpty () const
- uint32_t L1Occupancy () const
- uint8_t TriggerFIFOOccupancy () const
- bool TriggerFIFOFull () const
- bool TriggerFIFOEmpty () const
- bool DLLLock () const
- void Dump (int verb=1, std::ostream &os=std::cout) const

Static Private Attributes

- static const bit kError = 0

- static const bit kHaveToken = 11
- static const bit kReadoutFIFOOccupancy = 12
- static const bit kReadoutFIFOFull = 20
- static const bit kReadoutFIFOEmpty = 21
- static const bit kL1Occupancy = 22
- static const bit kTriggerFIFOOccupancy = 54
- static const bit kTriggerFIFOFull = 58
- static const bit kTriggerFIFOEmpty = 59
- static const bit kDLLLock = 60

Additional Inherited Members

7.29.1 Detailed Description

Author

Laurent Forthomme `laurent.forthomme@cern.ch`

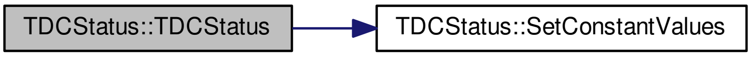
Date

27 Apr 2015

7.29.2 Constructor & Destructor Documentation

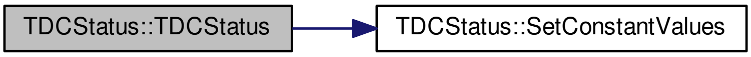
7.29.2.1 TDCStatus::TDCStatus ( ) `[inline]`

Here is the call graph for this function:



7.29.2.2 TDCStatus::TDCStatus (const TDCStatus & *s*) `[inline]`

Here is the call graph for this function:

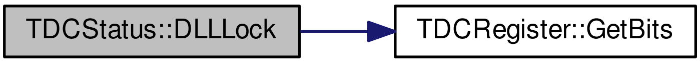


7.29.2.3 TDCStatus::TDCStatus (const std::vector< uint8_t > & *words*) [inline]

7.29.3 Member Function Documentation

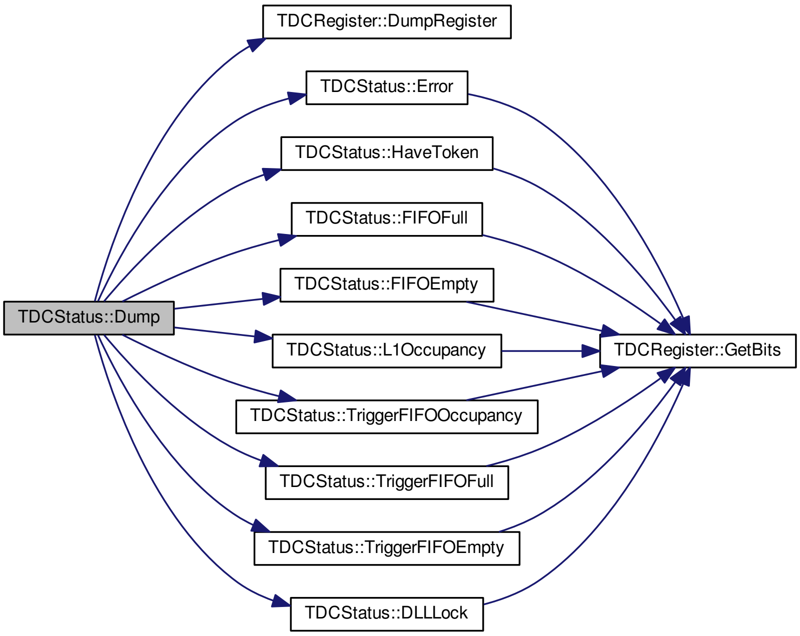
7.29.3.1 bool TDCStatus::DLLLock ( ) const [inline]

Here is the call graph for this function:



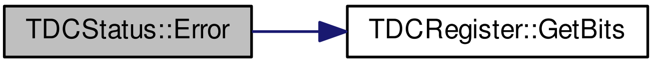
7.29.3.2 void TDCStatus::Dump (int *verb* = 1, std::ostream & *os* = std::cout) const [inline]

Here is the call graph for this function:



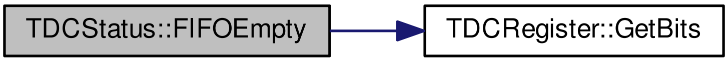
7.29.3.3 uint16_t TDCStatus::Error ( ) const [inline]

Here is the call graph for this function:



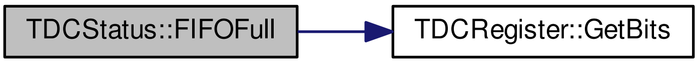
7.29.3.4 bool TDCStatus::FIFOEmpty ( ) const [inline]

Here is the call graph for this function:



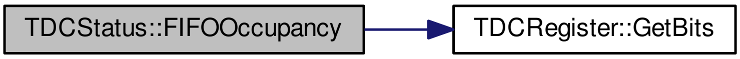
7.29.3.5 bool TDCStatus::FIFOFull ( ) const [inline]

Here is the call graph for this function:



7.29.3.6 uint8_t TDCStatus::FIFOOccupancy ( ) const [inline]

Here is the call graph for this function:



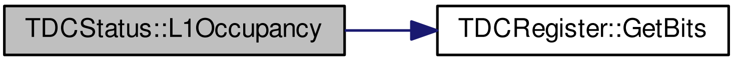
7.29.3.7 bool TDCStatus::HaveToken ( ) const [inline]

Here is the call graph for this function:

TDCStatus::HaveToken
TDCRegister::GetBits

7.29.3.8 uint32_t TDCStatus::L1Occupancy ( ) const [inline]

Here is the call graph for this function:



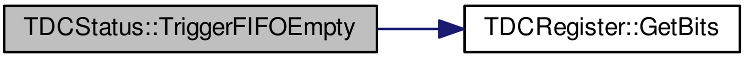
7.29.3.9 void TDCStatus::SetConstantValues ( ) [inline], [virtual]

Ensure that the critical constant values are properly set in the register word

Implements TDCRegister.

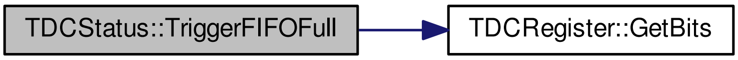
7.29.3.10 bool TDCStatus::TriggerFIFOEmpty ( ) const [inline]

Here is the call graph for this function:



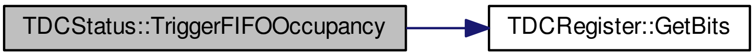
7.29.3.11 bool TDCStatus::TriggerFIFOFull ( ) const [inline]

Here is the call graph for this function:



7.29.3.12 uint8_t TDCStatus::TriggerFIFOOccupancy ( ) const [inline]

Here is the call graph for this function:



7.29.4 Field Documentation

7.29.4.1 const bit TDCStatus::kDLLLock = 60 [static], [private]

7.29.4.2 const bit TDCStatus::kError = 0 [static], [private]

7.29.4.3 const bit TDCStatus::kHaveToken = 11 [static], [private]

7.29.4.4 const bit TDCStatus::kL1Occupancy = 22 [static], [private]

7.29.4.5 const bit TDCStatus::kReadoutFIFOEmpty = 21 [static], [private]

7.29.4.6 const bit TDCStatus::kReadoutFIFOFull = 20 [static], [private]

7.29.4.7 const bit TDCStatus::kReadoutFIFOOccupancy = 12 [static], [private]

7.29.4.8 const bit TDCStatus::kTriggerFIFOEmpty = 59 [static], [private]

7.29.4.9 const bit TDCStatus::kTriggerFIFOFull = 58 [static], [private]

7.29.4.10 const bit TDCStatus::kTriggerFIFOOccupancy = 54 [static], [private]

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

- daq/include/TDCStatus.h

7.30 DAQ::QuickUSBHandler::Version Struct Reference

```
#include <QuickUSBHandler.h>
```

Data Fields

- QWORD MajorVersion
- QWORD MinorVersion
- QWORD BuildVersion

7.30.1 Field Documentation

7.30.1.1 QWORD DAQ::QuickUSBHandler::Version::BuildVersion

7.30.1.2 QWORD DAQ::QuickUSBHandler::Version::MajorVersion

7.30.1.3 QWORD DAQ::QuickUSBHandler::Version::MinorVersion

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

- daq/include/QuickUSBHandler.h

Index