

2015 Test beam Run Control

Generated by Doxygen 1.8.9.1

Mon Apr 20 2015 21:39:37

Contents

1	Hierarchical Index	1
1.1	Class Hierarchy	1
2	Data Structure Index	3
2.1	Data Structures	3
3	Data Structure Documentation	5
3.1	Client Class Reference	5
3.1.1	Detailed Description	6
3.1.2	Constructor & Destructor Documentation	6
3.1.2.1	Client	6
3.1.2.2	Client	6
3.1.2.3	~Client	6
3.1.3	Member Function Documentation	6
3.1.3.1	Connect	6
3.1.3.2	Disconnect	6
3.1.3.3	GetType	6
3.1.3.4	ParseMessage	6
3.1.3.5	Receive	6
3.1.3.6	Send	7
3.2	Exception Class Reference	7
3.2.1	Detailed Description	7
3.2.2	Constructor & Destructor Documentation	7
3.2.2.1	Exception	7
3.2.2.2	Exception	8
3.2.2.3	~Exception	8
3.2.3	Member Function Documentation	8
3.2.3.1	Description	8
3.2.3.2	Dump	8
3.2.3.3	ErrorNumber	9
3.2.3.4	From	9
3.2.3.5	Type	9

3.2.3.6	TypeString	10
3.3	file_header_t Struct Reference	10
3.3.1	Field Documentation	10
3.3.1.1	magic	10
3.3.1.2	run_id	10
3.3.1.3	spill_id	10
3.4	FPGAHandler Class Reference	11
3.4.1	Detailed Description	12
3.4.2	Constructor & Destructor Documentation	12
3.4.2.1	FPGAHandler	12
3.4.2.2	~FPGAHandler	12
3.4.3	Member Function Documentation	12
3.4.3.1	GetFilename	12
3.4.3.2	GetType	12
3.4.3.3	OpenFile	12
3.4.3.4	ReadBuffer	12
3.4.3.5	ReadConfiguration	12
3.4.3.6	SendConfiguration	12
3.5	HTTPMessage Class Reference	12
3.5.1	Constructor & Destructor Documentation	13
3.5.1.1	HTTPMessage	14
3.5.1.2	HTTPMessage	14
3.5.2	Member Function Documentation	14
3.5.2.1	Decode	14
3.5.2.2	Dump	15
3.5.2.3	Encode	15
3.5.2.4	GetKey	15
3.6	ListenerInfo Struct Reference	15
3.6.1	Field Documentation	15
3.6.1.1	name	15
3.6.1.2	type	15
3.7	Message Class Reference	15
3.7.1	Detailed Description	16
3.7.2	Constructor & Destructor Documentation	16
3.7.2.1	Message	16
3.7.2.2	Message	16
3.7.2.3	Message	16
3.7.2.4	~Message	16
3.7.3	Member Function Documentation	17
3.7.3.1	Dump	17

3.7.3.2	GetKey	17
3.7.3.3	GetString	17
3.7.3.4	IsFromWeb	17
3.7.4	Field Documentation	17
3.7.4.1	fString	17
3.8	Messenger Class Reference	17
3.8.1	Detailed Description	18
3.8.2	Constructor & Destructor Documentation	18
3.8.2.1	Messenger	18
3.8.2.2	Messenger	18
3.8.2.3	~Messenger	19
3.8.3	Member Function Documentation	19
3.8.3.1	Broadcast	19
3.8.3.2	Connect	19
3.8.3.3	Disconnect	19
3.8.3.4	Receive	19
3.8.3.5	Send	19
3.9	Socket Class Reference	19
3.9.1	Detailed Description	21
3.9.2	Constructor & Destructor Documentation	21
3.9.2.1	Socket	21
3.9.2.2	Socket	21
3.9.2.3	~Socket	21
3.9.3	Member Function Documentation	21
3.9.3.1	AcceptConnections	21
3.9.3.2	Bind	21
3.9.3.3	DumpConnected	21
3.9.3.4	FetchMessage	21
3.9.3.5	GetPort	22
3.9.3.6	GetSocketId	22
3.9.3.7	GetSocketType	22
3.9.3.8	IsWebSocket	22
3.9.3.9	Listen	22
3.9.3.10	PrepareConnection	22
3.9.3.11	SelectConnections	22
3.9.3.12	SendMessage	23
3.9.3.13	SetPort	23
3.9.3.14	SetSocketId	23
3.9.3.15	Start	23
3.9.3.16	Stop	23

3.9.4	Field Documentation	23
3.9.4.1	fBuffer	23
3.9.4.2	fMaster	23
3.9.4.3	fPort	23
3.9.4.4	fReadFds	23
3.9.4.5	fSocketsConnected	23
3.10	SocketMessage Class Reference	24
3.10.1	Detailed Description	25
3.10.2	Constructor & Destructor Documentation	25
3.10.2.1	SocketMessage	25
3.10.2.2	SocketMessage	25
3.10.2.3	SocketMessage	25
3.10.2.4	SocketMessage	25
3.10.2.5	SocketMessage	25
3.10.2.6	SocketMessage	26
3.10.2.7	SocketMessage	26
3.10.2.8	SocketMessage	26
3.10.2.9	SocketMessage	26
3.10.2.10	SocketMessage	27
3.10.2.11	SocketMessage	27
3.10.2.12	~SocketMessage	27
3.10.3	Member Function Documentation	27
3.10.3.1	Dump	27
3.10.3.2	GetIntValue	27
3.10.3.3	GetKey	27
3.10.3.4	GetString	28
3.10.3.5	GetValue	28
3.10.3.6	GetVectorValue	28
3.10.3.7	SetKeyValue	28
3.10.3.8	SetKeyValue	29
3.10.3.9	SetKeyValue	29
3.10.3.10	SetKeyValue	29
3.10.3.11	SetKeyValue	29
3.11	TDCConfiguration Class Reference	30
3.11.1	Detailed Description	31
3.11.2	Member Enumeration Documentation	31
3.11.2.1	DeadTime	31
3.11.2.2	EdgeResolution	32
3.11.2.3	WidthResolution	32
3.11.3	Constructor & Destructor Documentation	32

3.11.3.1	TDCCConfiguration	32
3.11.3.2	~TDCCConfiguration	32
3.11.4	Member Function Documentation	32
3.11.4.1	Dump	32
3.11.4.2	GetChannelOffset	32
3.11.4.3	GetDeadTime	32
3.11.4.4	GetDLLAdjustment	32
3.11.4.5	GetEdgeResolution	32
3.11.4.6	GetEdgesPairing	33
3.11.4.7	GetLeadingMode	33
3.11.4.8	GetMaxEventSize	33
3.11.4.9	GetNumWords	33
3.11.4.10	GetRCAdjustment	33
3.11.4.11	GetRejectFIFOFull	33
3.11.4.12	GetTrailingMode	33
3.11.4.13	GetTriggerMatchingMode	33
3.11.4.14	GetWidthResolution	33
3.11.4.15	GetWord	33
3.11.4.16	SetAllChannelsOffset	33
3.11.4.17	SetChannelOffset	34
3.11.4.18	SetDeadTime	34
3.11.4.19	SetDLLAdjustment	34
3.11.4.20	SetEdgeResolution	34
3.11.4.21	SetEdgesPairing	34
3.11.4.22	SetLeadingMode	34
3.11.4.23	SetMaxEventSize	34
3.11.4.24	SetRCAdjustment	34
3.11.4.25	SetRejectFIFOFull	34
3.11.4.26	SetTrailingMode	34
3.11.4.27	SetTriggerMatchingMode	34
3.11.4.28	SetWidthResolution	34
3.11.4.29	SetWord	34
3.12	TDCEvent Class Reference	35
3.12.1	Member Enumeration Documentation	35
3.12.1.1	EventType	35
3.12.2	Constructor & Destructor Documentation	36
3.12.2.1	TDCEvent	36
3.12.2.2	~TDCEvent	36
3.12.3	Member Function Documentation	36
3.12.3.1	GetBunchId	36

3.12.3.2	GetErrorFlags	36
3.12.3.3	GetEventId	36
3.12.3.4	GetLeadingTime	37
3.12.3.5	GetTDCId	37
3.12.3.6	GetTrailingTime	37
3.12.3.7	GetType	37
3.12.3.8	GetWidth	38
3.12.3.9	GetWordCount	38
Index		41

Chapter 1

Hierarchical Index

1.1 Class Hierarchy

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

Exception	7
file_header_t	10
ListenerInfo	15
Message	15
HTTPMessage	12
SocketMessage	24
Socket	19
Client	5
FPGAHandler	11
Messenger	17
TDCCConfiguration	30
TDCEvent	35

Chapter 2

Data Structure Index

2.1 Data Structures

Here are the data structures with brief descriptions:

Client	5
Exception	
A simple exception handler	7
file_header_t	10
FPGAHandler	11
HTTPMessage	12
ListenerInfo	15
Message	
Base message type	15
Messenger	17
Socket	19
SocketMessage	
Socket-passed message type	24
TDCCConfiguration	30
TDCEvent	35

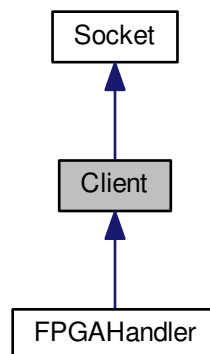
Chapter 3

Data Structure Documentation

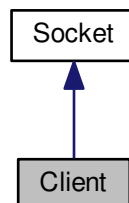
3.1 Client Class Reference

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

Inheritance diagram for Client:



Collaboration diagram for Client:



Public Member Functions

- [Client](#) ()
- [Client](#) (int port)
- [~Client](#) ()
- bool [Connect](#) ()
- void [Disconnect](#) ()
- void [Send](#) (const [Message](#) &m) const
- void [Receive](#) ()
- virtual void [ParseMessage](#) (const [SocketMessage](#) &m)
- virtual SocketType [GetType](#) () const

Additional Inherited Members

3.1.1 Detailed Description

[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

3.1.2 Constructor & Destructor Documentation

3.1.2.1 [Client::Client](#) () `[inline]`

3.1.2.2 [Client::Client](#) (int *port*)

3.1.2.3 [Client::~~Client](#) ()

3.1.3 Member Function Documentation

3.1.3.1 bool [Client::Connect](#) ()

3.1.3.2 void [Client::Disconnect](#) ()

3.1.3.3 virtual SocketType [Client::GetType](#) () const `[inline],[virtual]`

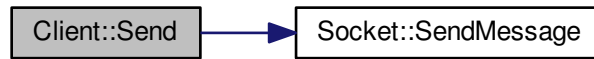
Reimplemented in [FPGAHandler](#).

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

3.1.3.5 void [Client::Receive](#) ()

3.1.3.6 `void Client::Send (const Message & m) const` `[inline]`

Here is the call graph for this function:



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

- `include/Client.h`

3.2 Exception Class Reference

A simple exception handler.

```
#include <Exception.h>
```

Public Member Functions

- [Exception](#) (const char *from, std::string desc, ExceptionType type=Undefined, const int id=0)
- [Exception](#) (const char *from, const char *desc, ExceptionType type=Undefined, const int id=0)
- [~Exception](#) ()
- std::string [From](#) () const
- int [ErrorNumber](#) () const
- std::string [Description](#) () const
- ExceptionType [Type](#) () const
- std::string [TypeString](#) () const
- void [Dump](#) (std::ostream &os=std::cerr) const

3.2.1 Detailed Description

A simple exception handler.

Author

Laurent Forthomme laurent.forthomme@cern.ch

Date

24 Mar 2015

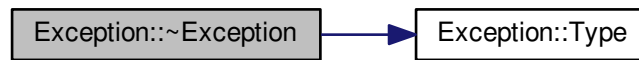
3.2.2 Constructor & Destructor Documentation

3.2.2.1 `Exception::Exception (const char * from, std::string desc, ExceptionType type = Undefined, const int id = 0)`
`[inline]`

3.2.2.2 `Exception::Exception (const char * from, const char * desc, ExceptionType type = Undefined, const int id = 0)` `[inline]`

3.2.2.3 `Exception::~~Exception ()` `[inline]`

Here is the call graph for this function:



3.2.3 Member Function Documentation

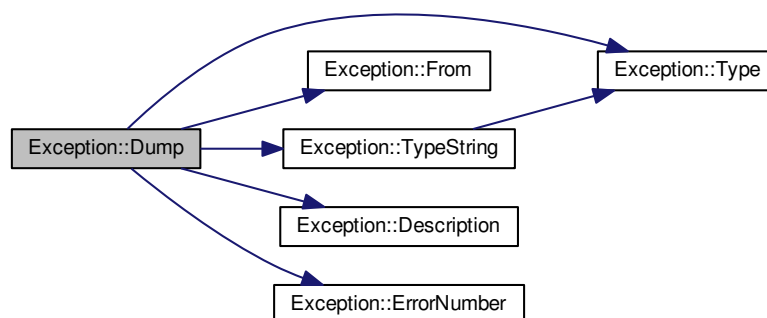
3.2.3.1 `std::string Exception::Description () const` `[inline]`

Here is the caller graph for this function:



3.2.3.2 `void Exception::Dump (std::ostream & os = std::cerr) const` `[inline]`

Here is the call graph for this function:



3.2.3.3 `int Exception::ErrorNumber () const [inline]`

Here is the caller graph for this function:



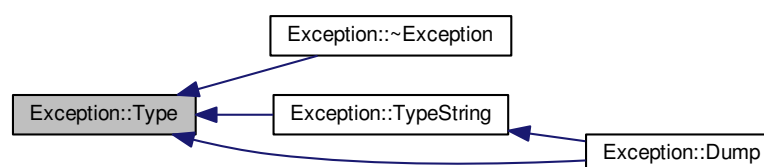
3.2.3.4 `std::string Exception::From () const [inline]`

Here is the caller graph for this function:



3.2.3.5 `ExceptionType Exception::Type () const [inline]`

Here is the caller graph for this function:



3.2.3.6 `std::string Exception::TypeString () const [inline]`

Here is the call graph for this function:



Here is the caller graph for this function:



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

- `include/Exception.h`

3.3 `file_header_t` Struct Reference

```
#include <FPGAHandler.h>
```

Data Fields

- `uint32_t magic`
- `uint32_t run_id`
- `uint32_t spill_id`

3.3.1 Field Documentation

3.3.1.1 `uint32_t file_header_t::magic`

3.3.1.2 `uint32_t file_header_t::run_id`

3.3.1.3 `uint32_t file_header_t::spill_id`

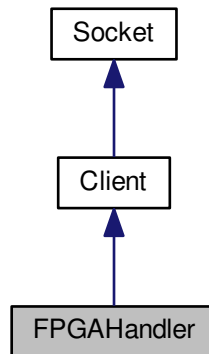
The documentation for this struct was generated from the following file:

- `include/FPGAHandler.h`

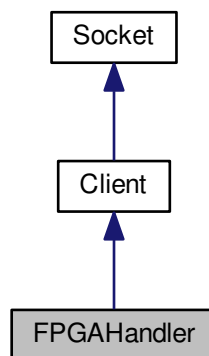
3.4 FPGAHandler Class Reference

```
#include <FPGAHandler.h>
```

Inheritance diagram for FPGAHandler:



Collaboration diagram for FPGAHandler:



Public Member Functions

- [FPGAHandler](#) (int port, const char *dev)
- virtual [~FPGAHandler](#) ()
- void [OpenFile](#) ()
- std::string [GetFilename](#) () const
- void [SendConfiguration](#) (const [TDCCConfiguration](#) &c)
- [TDCCConfiguration](#) [ReadConfiguration](#) ()
- void [ReadBuffer](#) ()
- SocketType [GetType](#) () const

Additional Inherited Members

3.4.1 Detailed Description

Author

Laurent Forthomme laurent.forthomme@cern.ch

Date

14 Apr 2015

3.4.2 Constructor & Destructor Documentation

3.4.2.1 `FPGAHandler::FPGAHandler (int port, const char * dev)`

3.4.2.2 `virtual FPGAHandler::~~FPGAHandler ()` `[virtual]`

3.4.3 Member Function Documentation

3.4.3.1 `std::string FPGAHandler::GetFilename ()` `const` `[inline]`

3.4.3.2 `SocketType FPGAHandler::GetType ()` `const` `[inline]`, `[virtual]`

Reimplemented from [Client](#).

3.4.3.3 `void FPGAHandler::OpenFile ()`

3.4.3.4 `void FPGAHandler::ReadBuffer ()`

3.4.3.5 `TDCConfiguration FPGAHandler::ReadConfiguration ()`

3.4.3.6 `void FPGAHandler::SendConfiguration (const TDCConfiguration & c)`

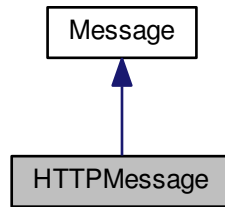
The documentation for this class was generated from the following file:

- `include/FPGAHandler.h`

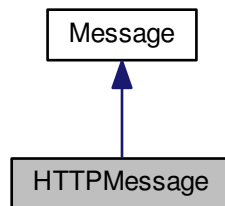
3.5 HTTPMessage Class Reference

```
#include <HTTPMessage.h>
```

Inheritance diagram for HTTPMessage:



Collaboration diagram for HTTPMessage:



Public Member Functions

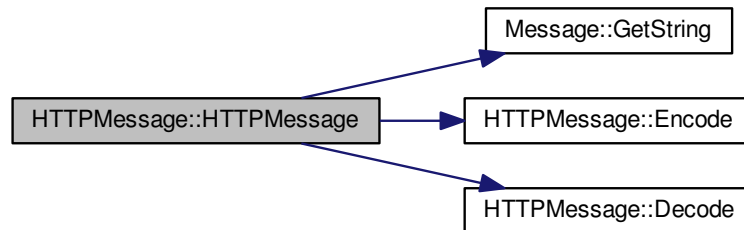
- [HTTPMessage](#) (WebSocket *ws, [Message](#) m, MessageAction a)
- [HTTPMessage](#) (WebSocket *ws, const char *msg, MessageAction a)
- void [Decode](#) ()
- void [Encode](#) ()
- MessageKey [GetKey](#) () const
- void [Dump](#) (std::ostream &os=std::cout) const

Additional Inherited Members

3.5.1 Constructor & Destructor Documentation

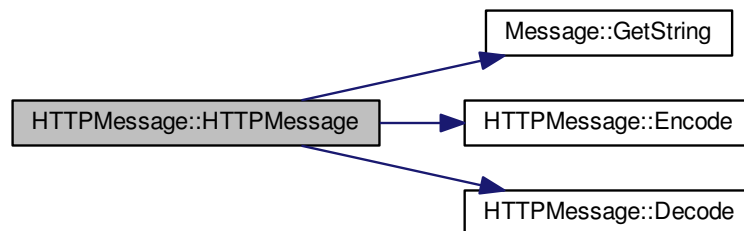
3.5.1.1 HTTPMessage::HTTPMessage (WebSocket * *ws*, Message *m*, MessageAction *a*) [inline]

Here is the call graph for this function:



3.5.1.2 HTTPMessage::HTTPMessage (WebSocket * *ws*, const char * *msg*, MessageAction *a*) [inline]

Here is the call graph for this function:



3.5.2 Member Function Documentation

3.5.2.1 void HTTPMessage::Decode () [inline]

Here is the caller graph for this function:



3.5.2.2 void HTTPMessage::Dump (std::ostream & os = std::cout) const [inline]

3.5.2.3 void HTTPMessage::Encode () [inline]

Here is the caller graph for this function:



3.5.2.4 MessageKey HTTPMessage::GetKey () const [inline]

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

- include/HTTPMessage.h

3.6 ListenerInfo Struct Reference

```
#include <Messenger.h>
```

Data Fields

- std::string [name](#)
- SocketType [type](#)

3.6.1 Field Documentation

3.6.1.1 std::string ListenerInfo::name

3.6.1.2 SocketType ListenerInfo::type

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

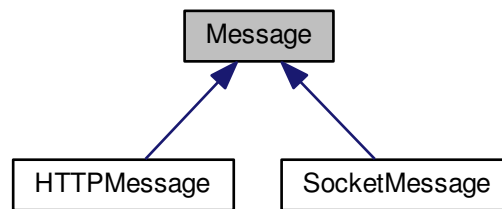
- include/Messenger.h

3.7 Message Class Reference

Base message type.

```
#include <Message.h>
```

Inheritance diagram for Message:



Public Member Functions

- `Message ()`
- `Message (const char *msg)`
- `Message (std::string msg)`
- `~Message ()`
- `MessageKey GetKey () const`
- `std::string GetString () const`
- `bool IsFromWeb () const`
- `void Dump (std::ostream &os=std::cout) const`

Protected Attributes

- `std::string fString`

3.7.1 Detailed Description

Base message type.

Author

Laurent Forthomme laurent.forthomme@cern.ch

Date

6 Apr 2015

3.7.2 Constructor & Destructor Documentation

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

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

3.7.2.3 `Message::Message (std::string msg) [inline]`

3.7.2.4 `Message::~~Message () [inline]`

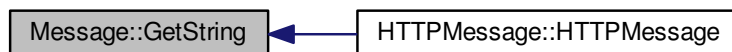
3.7.3 Member Function Documentation

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

3.7.3.2 `MessageKey Message::GetKey () const` [inline]

3.7.3.3 `std::string Message::GetString () const` [inline]

Here is the caller graph for this function:



3.7.3.4 `bool Message::IsFromWeb () const` [inline]

3.7.4 Field Documentation

3.7.4.1 `std::string Message::fString` [protected]

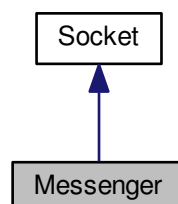
The documentation for this class was generated from the following file:

- include/Message.h

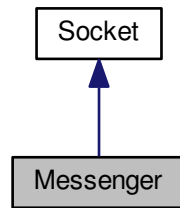
3.8 Messenger Class Reference

```
#include <Messenger.h>
```

Inheritance diagram for Messenger:



Collaboration diagram for Messenger:



Public Member Functions

- [Messenger](#) ()
- [Messenger](#) (int port)
- [~Messenger](#) ()
- bool [Connect](#) ()
 - Connect the master.*
- void [Disconnect](#) ()
 - Remove the master.*
- void [Send](#) (const [Message](#) &m, int sid) const
 - Send any type of message to any client.*
- MessageKey [Receive](#) ()
 - Handle a message reception from a client.*
- void [Broadcast](#) (const [Message](#) &m) const
 - Emit a message to all clients connected through the socket.*

Additional Inherited Members

3.8.1 Detailed Description

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

3.8.2 Constructor & Destructor Documentation

3.8.2.1 Messenger::Messenger ()

3.8.2.2 Messenger::Messenger (int port)

3.8.2.3 Messenger::~Messenger ()

3.8.3 Member Function Documentation

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

Emit a message to all clients connected through the socket.

Parameters

in	<i>m</i>	Message to transmit
----	----------	-------------------------------------

3.8.3.2 bool Messenger::Connect ()

Connect the master.

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

3.8.3.3 void Messenger::Disconnect ()

Remove the master.

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

3.8.3.4 MessageKey Messenger::Receive ()

Handle a message reception from a client.

Returns

The key to the message received if successfully parsed

3.8.3.5 void Messenger::Send (const Message & *m*, int *sid*) const [inline]

Send any type of message to any client.

Parameters

in	<i>m</i>	Message to transmit
in	<i>sid</i>	Unique identifier of the client on this socket

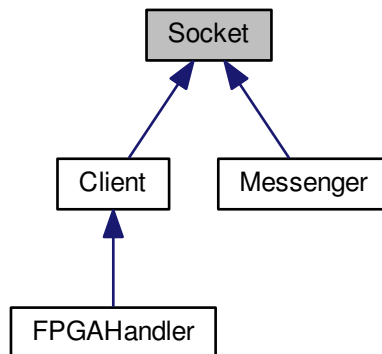
The documentation for this class was generated from the following file:

- include/Messenger.h

3.9 Socket Class Reference

```
#include <Socket.h>
```

Inheritance diagram for Socket:



Public Member Functions

- [Socket](#) ()
- [Socket](#) (int port)
- virtual [~Socket](#) ()
- 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 [Stop](#) ()
Terminates the socket and all attached communications.
- 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()

3.9.1 Detailed Description

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

3.9.2 Constructor & Destructor Documentation

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

3.9.2.2 `Socket::Socket (int port)`

3.9.2.3 `virtual Socket::~~Socket ()` [virtual]

3.9.3 Member Function Documentation

3.9.3.1 `void Socket::AcceptConnections (Socket & socket)`

Accept connection from a client.

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

Parameters

<code>in, out</code>	<code>socket</code>	Master/client object to enable on the socket
----------------------	---------------------	--

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

Bind a name to a socket.

Returns

Success of the operation

3.9.3.3 `void Socket::DumpConnected ()` const

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

Receive a message from a socket.

Returns

Received message as a `std::string`

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

Retrieve the port used for this socket.

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

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

Here is the caller graph for this function:



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

Here is the call graph for this function:



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

Listen to incoming messages.

Set the socket to listen to any message coming from outside

3.9.3.10 `void Socket::PrepareConnection ()` `[protected]`

3.9.3.11 `void Socket::SelectConnections ()`

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

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

Send a message on a socket.

Here is the caller graph for this function:



3.9.3.13 `void Socket::SetPort (int port)` [inline]

3.9.3.14 `void Socket::SetSocketId (int sid)` [inline]

3.9.3.15 `bool Socket::Start ()` [protected]

Start the socket.

Launch all mandatory operations to set the socket to be used

Returns

Success of the operation

3.9.3.16 `void Socket::Stop ()` [protected]

Terminates the socket and all attached communications.

3.9.4 Field Documentation

3.9.4.1 `char Socket::fBuffer[MAX_WORD_LENGTH]` [protected]

3.9.4.2 `fd_set Socket::fMaster` [protected]

Master file descriptor list.

3.9.4.3 `int Socket::fPort` [protected]

3.9.4.4 `fd_set Socket::fReadFds` [protected]

Temp file descriptor list for select()

3.9.4.5 `SocketCollection Socket::fSocketsConnected` [protected]

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

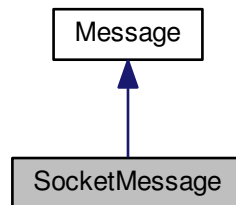
- include/Socket.h

3.10 SocketMessage Class Reference

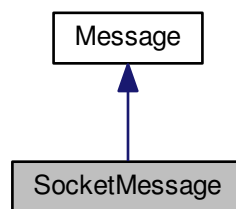
Socket-passed message type.

```
#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](#) (MessageKey key)
- [SocketMessage](#) (MessageKey key, const char *value)
- [SocketMessage](#) (MessageKey key, std::string value)
- [SocketMessage](#) (MessageKey key, const int value)
- [SocketMessage](#) (MessageKey key, const float value)
- [SocketMessage](#) (MessageKey key, const double value)
- [SocketMessage](#) (MessageMap msg_m)
- [~SocketMessage](#) ()
- void [SetKeyValue](#) (MessageKey key, std::string value)

Send a string-valued message.

- void [SetKeyValue](#) (MessageKey key, const char *value)
- void [SetKeyValue](#) (MessageKey key, int int_value)
Send an integer-valued message.
- void [SetKeyValue](#) (MessageKey key, float float_value)
Send an float-valued message.
- void [SetKeyValue](#) (MessageKey key, double double_value)
Send an double-valued message.
- std::string [GetString](#) () const
- MessageKey [GetKey](#) () const
- std::string [GetValue](#) () const
- int [GetIntValue](#) () const
- VectorValue [GetVectorValue](#) () const
- void [Dump](#) (std::ostream &os=std::cout) const

Additional Inherited Members

3.10.1 Detailed Description

Socket-passed message type.

Author

Laurent Forthomme laurent.forthomme@cern.ch

Date

26 Mar 2015

3.10.2 Constructor & Destructor Documentation

3.10.2.1 `SocketMessage::SocketMessage ()` [\[inline\]](#)

3.10.2.2 `SocketMessage::SocketMessage (const Message & msg)` [\[inline\]](#)

3.10.2.3 `SocketMessage::SocketMessage (const char * msg_s)` [\[inline\]](#)

3.10.2.4 `SocketMessage::SocketMessage (std::string msg_s)` [\[inline\]](#)

3.10.2.5 `SocketMessage::SocketMessage (MessageKey key)` [\[inline\]](#)

Here is the call graph for this function:



3.10.2.6 SocketMessage::SocketMessage (MessageKey key, const char * value) [inline]

Here is the call graph for this function:



3.10.2.7 SocketMessage::SocketMessage (MessageKey key, std::string value) [inline]

Here is the call graph for this function:



3.10.2.8 SocketMessage::SocketMessage (MessageKey key, const int value) [inline]

Here is the call graph for this function:



3.10.2.9 SocketMessage::SocketMessage (MessageKey key, const float value) [inline]

Here is the call graph for this function:



3.10.2.10 `SocketMessage::SocketMessage (MessageKey key, const double value)` [inline]

Here is the call graph for this function:



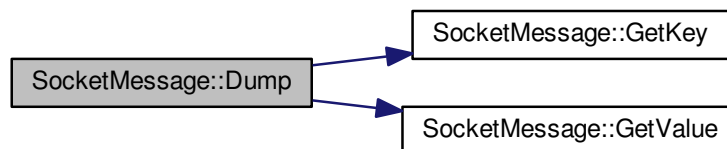
3.10.2.11 `SocketMessage::SocketMessage (MessageMap msg_m)` [inline]

3.10.2.12 `SocketMessage::~~SocketMessage ()` [inline]

3.10.3 Member Function Documentation

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

Here is the call graph for this function:



3.10.3.2 `int SocketMessage::GetIntValue () const` [inline]

3.10.3.3 `MessageKey SocketMessage::GetKey () const` [inline]

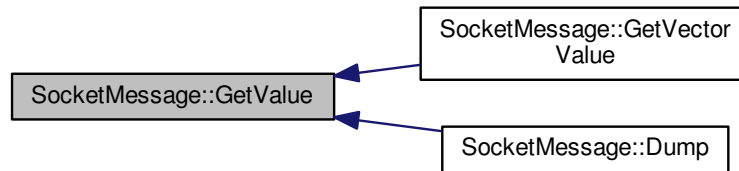
Here is the caller graph for this function:



3.10.3.4 `std::string SocketMessage::GetString () const` `[inline]`

3.10.3.5 `std::string SocketMessage::GetValue () const` `[inline]`

Here is the caller graph for this function:



3.10.3.6 `VectorValue SocketMessage::GetVectorValue () const` `[inline]`

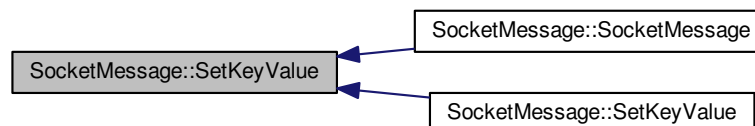
Here is the call graph for this function:



3.10.3.7 `void SocketMessage::SetKeyValue (MessageKey key, std::string value)` `[inline]`

Send a string-valued message.

Here is the caller graph for this function:



3.10.3.8 `void SocketMessage::SetKeyValue (MessageKey key, const char * value)` [inline]

Here is the call graph for this function:



3.10.3.9 `void SocketMessage::SetKeyValue (MessageKey key, int int_value)` [inline]

Send an integer-valued message.

Here is the call graph for this function:



3.10.3.10 `void SocketMessage::SetKeyValue (MessageKey key, float float_value)` [inline]

Send an float-valued message.

Here is the call graph for this function:



3.10.3.11 `void SocketMessage::SetKeyValue (MessageKey key, double double_value)` [inline]

Send an double-valued message.

Here is the call graph for this function:



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

- include/SocketMessage.h

3.11 TDCConfiguration Class Reference

```
#include <TDCConfiguration.h>
```

Public Types

- enum [EdgeResolution](#) {
[E_100PS](#) =0, [E_200PS](#), [E_400PS](#), [E_800PS](#),
[E_1600PS](#), [E_3120PS](#), [E_6250PS](#), [E_12500PS](#) }
- 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](#) }

Public Member Functions

- [TDCConfiguration](#) ()
- virtual [~TDCConfiguration](#) ()
- void [SetWord](#) (const unsigned int i, const word_t word)
Set one single word in the configuration.
- word_t [GetWord](#) (const unsigned int i) const
Retrieve one single word from the configuration.
- uint8_t [GetNumWords](#) () const
Number of words in the configuration.
- void [SetEdgeResolution](#) (const [EdgeResolution](#) r)
- [EdgeResolution](#) [GetEdgeResolution](#) () const
- void [SetMaxEventSize](#) (unsigned int sz)
Set the maximum number of hits per event.
- uint8_t [GetMaxEventSize](#) () const
Extract the maximum number of hits per event.
- void [SetRejectFIFOFull](#) (bool rej=true)
Reject hits when readout FIFO full.
- bool [GetRejectFIFOFull](#) () const
Are hits rejected when readout FIFO is full?
- void [SetChannelOffset](#) (int channel, uint16_t offset)

- uint16_t [GetChannelOffset](#) (int channel)
- void [SetAllChannelsOffset](#) (uint16_t offset)
- void [SetDLLAdjustment](#) (int tap, uint8_t adj)
- uint8_t [GetDLLAdjustment](#) (int tap)
- void [SetRCAdjustment](#) (int tap, uint8_t adj)
- uint8_t [GetRCAdjustment](#) (int tap)
- void [SetWidthResolution](#) (const [WidthResolution](#) r)
- [WidthResolution](#) [GetWidthResolution](#) () const
- void [SetDeadTime](#) (const [DeadTime](#) dt)
- [DeadTime](#) [GetDeadTime](#) () const
- 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 [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 [SetTriggerMatchingMode](#) (const bool trig=true)
- bool [GetTriggerMatchingMode](#) () const
- void [SetEdgesPairing](#) (const bool pair=true)
- bool [GetEdgesPairing](#) () const
- void [Dump](#) (std::ostream &os=std::cout) const

3.11.1 Detailed Description

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

Author

Laurent Forthomme laurent.forthomme@cern.ch

Date

16 Apr 2015

3.11.2 Member Enumeration Documentation

3.11.2.1 enum TDCCConfiguration::DeadTime

Enumerator

DT_5NS

DT_10NS

DT_30NS

DT_100NS

3.11.2.2 enum TDCCConfiguration::EdgeResolution

Enumerator

E_100PS
E_200PS
E_400PS
E_800PS
E_1600PS
E_3120PS
E_6250PS
E_12500PS

3.11.2.3 enum TDCCConfiguration::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

3.11.3 Constructor & Destructor Documentation

3.11.3.1 TDCCConfiguration::TDCCConfiguration ()

3.11.3.2 virtual TDCCConfiguration::~~TDCCConfiguration () [inline], [virtual]

3.11.4 Member Function Documentation

3.11.4.1 void TDCCConfiguration::Dump (std::ostream & *os* = std::cout) const

3.11.4.2 uint16_t TDCCConfiguration::GetChannelOffset (int *channel*) [inline]

3.11.4.3 DeadTime TDCCConfiguration::GetDeadTime () const [inline]

3.11.4.4 uint8_t TDCCConfiguration::GetDLLAdjustment (int *tap*) [inline]

3.11.4.5 EdgeResolution TDCCConfiguration::GetEdgeResolution () const [inline]

3.11.4.6 `bool TDCCConfiguration::GetEdgesPairing () const [inline]`

3.11.4.7 `bool TDCCConfiguration::GetLeadingMode () const [inline]`

Extract the status for the detection of leading edges.

3.11.4.8 `uint8_t TDCCConfiguration::GetMaxEventSize () const [inline]`

Extract the maximum number of hits per event.

3.11.4.9 `uint8_t TDCCConfiguration::GetNumWords () const [inline]`

Number of words in the configuration.

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

3.11.4.10 `uint8_t TDCCConfiguration::GetRCAdjustment (int tap) [inline]`

3.11.4.11 `bool TDCCConfiguration::GetRejectFIFOFull () const [inline]`

Are hits rejected when readout FIFO is full?

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

3.11.4.12 `bool TDCCConfiguration::GetTrailingMode () const [inline]`

Extract the status for the detection of trailing edges.

3.11.4.13 `bool TDCCConfiguration::GetTriggerMatchingMode () const [inline]`

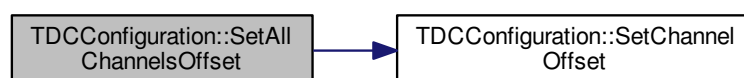
3.11.4.14 `WidthResolution TDCCConfiguration::GetWidthResolution () const [inline]`

3.11.4.15 `word_t TDCCConfiguration::GetWord (const unsigned int i) const [inline]`

Retrieve one single word from the configuration.

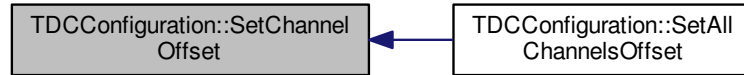
3.11.4.16 `void TDCCConfiguration::SetAllChannelsOffset (uint16_t offset) [inline]`

Here is the call graph for this function:



3.11.4.17 void TDCConfiguration::SetChannelOffset (int *channel*, uint16_t *offset*) [inline]

Here is the caller graph for this function:



3.11.4.18 void TDCConfiguration::SetDeadTime (const DeadTime *dt*) [inline]

3.11.4.19 void TDCConfiguration::SetDLLAdjustment (int *tap*, uint8_t *adj*) [inline]

3.11.4.20 void TDCConfiguration::SetEdgeResolution (const EdgeResolution *r*) [inline]

3.11.4.21 void TDCConfiguration::SetEdgesPairing (const bool *pair* = true) [inline]

3.11.4.22 void TDCConfiguration::SetLeadingMode (const bool *lead* = true) [inline]

Enable the detection of leading edges.

3.11.4.23 void TDCConfiguration::SetMaxEventSize (unsigned int *sz*) [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 bigger than 128 then set to unlimited.

3.11.4.24 void TDCConfiguration::SetRCAdjustment (int *tap*, uint8_t *adj*) [inline]

3.11.4.25 void TDCConfiguration::SetRejectFIFOFull (bool *rej* = true) [inline]

Reject hits when readout FIFO full.

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

3.11.4.26 void TDCConfiguration::SetTrailingMode (const bool *trail* = true) [inline]

Enable/disable the detection of trailing edges.

3.11.4.27 void TDCConfiguration::SetTriggerMatchingMode (const bool *trig* = true) [inline]

3.11.4.28 void TDCConfiguration::SetWidthResolution (const WidthResolution *r*) [inline]

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

Set one single word in the configuration.

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

- include/TDCConfiguration.h

3.12 TDCEvent Class Reference

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

Public Types

- enum [EventType](#) {
[Invalid](#) =-1, [GroupHeader](#) =0, [GroupTrailer](#), [TDCHeader](#),
[TDCTrailer](#), [LeadingEdge](#), [TrailingEdge](#), [Error](#),
[Debug](#) }

Public Member Functions

- [TDCEvent](#) (const uint32_t &word)
- virtual [~TDCEvent](#) ()
- [EventType](#) [GetType](#) () const
Type of packet read out from the TDC.
- unsigned int [GetTDCId](#) () const
Programmed identifier of master TDC.
- uint16_t [GetEventId](#) () const
Event identifier from event counter.
- uint16_t [GetWordCount](#) () const
Total number of words in event (including headers and trailers)
- uint16_t [GetBunchId](#) () const
Bunch identifier of trigger (or trigger time tag)
- uint32_t [GetLeadingTime](#) (bool pair=false) const
Leading edge measurement in programmed time resolution.
- uint8_t [GetWidth](#) () const
Width of pulse in programmed time resolution.
- uint32_t [GetTrailingTime](#) () const
Trailing edge measurement in programmed time resolution.
- uint16_t [GetErrorFlags](#) () const
Return error flags if an error condition has been detected.

3.12.1 Member Enumeration Documentation

3.12.1.1 enum TDCEvent::EventType

Enumerator

Invalid
GroupHeader
GroupTrailer
TDCHeader
TDCTrailer
LeadingEdge
TrailingEdge
Error
Debug

3.12.2 Constructor & Destructor Documentation

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

3.12.2.2 `virtual TDCEvent::~~TDCEvent () [inline],[virtual]`

3.12.3 Member Function Documentation

3.12.3.1 `uint16_t TDCEvent::GetBunchId () const [inline]`

Bunch identifier of trigger (or trigger time tag)

Here is the call graph for this function:



3.12.3.2 `uint16_t TDCEvent::GetErrorFlags () const [inline]`

Return error flags if an error condition has been detected.

Here is the call graph for this function:



3.12.3.3 `uint16_t TDCEvent::GetEventId () const [inline]`

Event identifier from event counter.

Here is the call graph for this function:



3.12.3.4 `uint32_t TDCEvent::GetLeadingTime (bool pair = false) const` `[inline]`

Leading edge measurement in programmed time resolution.

Here is the call graph for this function:



3.12.3.5 `unsigned int TDCEvent::GetTDCId () const` `[inline]`

Programmed identifier of master TDC.

3.12.3.6 `uint32_t TDCEvent::GetTrailingTime () const` `[inline]`

Trailing edge measurement in programmed time resolution.

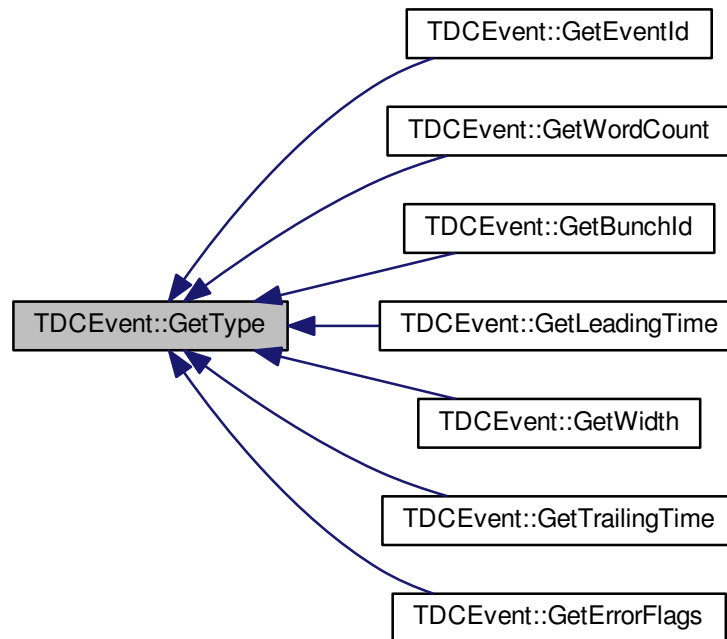
Here is the call graph for this function:



3.12.3.7 `EventType TDCEvent::GetType () const` `[inline]`

Type of packet read out from the TDC.

Here is the caller graph for this function:



3.12.3.8 `uint8_t TDCEvent::GetWidth () const [inline]`

Width of pulse in programmed time resolution.

Here is the call graph for this function:



3.12.3.9 `uint16_t TDCEvent::GetWordCount () const [inline]`

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

Here is the call graph for this function:



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

- `include/TDCEvent.h`

Index

- ~Client
 - Client, [6](#)
- ~Exception
 - Exception, [8](#)
- ~FPGAHandler
 - FPGAHandler, [12](#)
- ~Message
 - Message, [16](#)
- ~Messenger
 - Messenger, [18](#)
- ~Socket
 - Socket, [21](#)
- ~SocketMessage
 - SocketMessage, [27](#)
- ~TDCConfiguration
 - TDCConfiguration, [32](#)
- ~TDCEvent
 - TDCEvent, [36](#)
- AcceptConnections
 - Socket, [21](#)
- Bind
 - Socket, [21](#)
- Broadcast
 - Messenger, [19](#)
- Client, [5](#)
 - ~Client, [6](#)
 - Client, [6](#)
 - Connect, [6](#)
 - Disconnect, [6](#)
 - GetType, [6](#)
 - ParseMessage, [6](#)
 - Receive, [6](#)
 - Send, [6](#)
- Connect
 - Client, [6](#)
 - Messenger, [19](#)
- DT_100NS
 - TDCConfiguration, [31](#)
- DT_10NS
 - TDCConfiguration, [31](#)
- DT_30NS
 - TDCConfiguration, [31](#)
- DT_5NS
 - TDCConfiguration, [31](#)
- DeadTime
 - TDCConfiguration, [31](#)
- Debug
 - TDCEvent, [35](#)
- Decode
 - HTTPMessage, [14](#)
- Description
 - Exception, [8](#)
- Disconnect
 - Client, [6](#)
 - Messenger, [19](#)
- Dump
 - Exception, [8](#)
 - HTTPMessage, [14](#)
 - Message, [17](#)
 - SocketMessage, [27](#)
 - TDCConfiguration, [32](#)
- DumpConnected
 - Socket, [21](#)
- E_100PS
 - TDCConfiguration, [32](#)
- E_12500PS
 - TDCConfiguration, [32](#)
- E_1600PS
 - TDCConfiguration, [32](#)
- E_200PS
 - TDCConfiguration, [32](#)
- E_3120PS
 - TDCConfiguration, [32](#)
- E_400PS
 - TDCConfiguration, [32](#)
- E_6250PS
 - TDCConfiguration, [32](#)
- E_800PS
 - TDCConfiguration, [32](#)
- EdgeResolution
 - TDCConfiguration, [31](#)
- Encode
 - HTTPMessage, [15](#)
- Error
 - TDCEvent, [35](#)
- ErrorNumber
 - Exception, [8](#)
- EventType
 - TDCEvent, [35](#)
- Exception, [7](#)
 - ~Exception, [8](#)
 - Description, [8](#)
 - Dump, [8](#)
 - ErrorNumber, [8](#)
 - Exception, [7](#)

- From, 9
- Type, 9
- TypeString, 9
- fBuffer
 - Socket, 23
- fMaster
 - Socket, 23
- FPGAHandler, 11
 - ~FPGAHandler, 12
 - FPGAHandler, 12
 - GetFilename, 12
 - GetType, 12
 - OpenFile, 12
 - ReadBuffer, 12
 - ReadConfiguration, 12
 - SendConfiguration, 12
- fPort
 - Socket, 23
- fReadFds
 - Socket, 23
- fSocketsConnected
 - Socket, 23
- fString
 - Message, 17
- FetchMessage
 - Socket, 21
- file_header_t, 10
 - magic, 10
 - run_id, 10
 - spill_id, 10
- From
 - Exception, 9
- GetBunchId
 - TDCEvent, 36
- GetChannelOffset
 - TDCConfiguration, 32
- GetDLLAdjustment
 - TDCConfiguration, 32
- GetDeadTime
 - TDCConfiguration, 32
- GetEdgeResolution
 - TDCConfiguration, 32
- GetEdgesPairing
 - TDCConfiguration, 32
- GetErrorFlags
 - TDCEvent, 36
- GetEventId
 - TDCEvent, 36
- GetFilename
 - FPGAHandler, 12
- GetIntValue
 - SocketMessage, 27
- GetKey
 - HTTPMessage, 15
 - Message, 17
 - SocketMessage, 27
- GetLeadingMode
 - TDCConfiguration, 33
- GetLeadingTime
 - TDCEvent, 36
- GetMaxEventSize
 - TDCConfiguration, 33
- GetNumWords
 - TDCConfiguration, 33
- GetPort
 - Socket, 22
- GetRCAdjustment
 - TDCConfiguration, 33
- GetRejectFIFOFull
 - TDCConfiguration, 33
- GetSocketId
 - Socket, 22
- GetSocketType
 - Socket, 22
- GetString
 - Message, 17
 - SocketMessage, 27
- GetTDCId
 - TDCEvent, 37
- GetTrailingMode
 - TDCConfiguration, 33
- GetTrailingTime
 - TDCEvent, 37
- GetTriggerMatchingMode
 - TDCConfiguration, 33
- GetType
 - Client, 6
 - FPGAHandler, 12
 - TDCEvent, 37
- GetValue
 - SocketMessage, 28
- GetVectorValue
 - SocketMessage, 28
- GetWidth
 - TDCEvent, 38
- GetWidthResolution
 - TDCConfiguration, 33
- GetWord
 - TDCConfiguration, 33
- GetWordCount
 - TDCEvent, 38
- GroupHeader
 - TDCEvent, 35
- GroupTrailer
 - TDCEvent, 35
- HTTPMessage, 12
 - Decode, 14
 - Dump, 14
 - Encode, 15
 - GetKey, 15
 - HTTPMessage, 13, 14
- Invalid
 - TDCEvent, 35
- IsFromWeb

- Message, 17
- IsWebSocket
 - Socket, 22
- LeadingEdge
 - TDCEvent, 35
- Listen
 - Socket, 22
- ListenerInfo, 15
 - name, 15
 - type, 15
- magic
 - file_header_t, 10
- Message, 15
 - ~Message, 16
 - Dump, 17
 - fString, 17
 - GetKey, 17
 - GetString, 17
 - IsFromWeb, 17
 - Message, 16
- Messenger, 17
 - ~Messenger, 18
 - Broadcast, 19
 - Connect, 19
 - Disconnect, 19
 - Messenger, 18
 - Receive, 19
 - Send, 19
- name
 - ListenerInfo, 15
- OpenFile
 - FPGAHandler, 12
- ParseMessage
 - Client, 6
- PrepareConnection
 - Socket, 22
- ReadBuffer
 - FPGAHandler, 12
- ReadConfiguration
 - FPGAHandler, 12
- Receive
 - Client, 6
 - Messenger, 19
- run_id
 - file_header_t, 10
- SelectConnections
 - Socket, 22
- Send
 - Client, 6
 - Messenger, 19
- SendConfiguration
 - FPGAHandler, 12
- SendMessage
- Socket, 22
 - SetAllChannelsOffset
 - TDCCConfiguration, 33
 - SetChannelOffset
 - TDCCConfiguration, 33
 - SetDLLAdjustment
 - TDCCConfiguration, 34
 - SetDeadTime
 - TDCCConfiguration, 34
 - SetEdgeResolution
 - TDCCConfiguration, 34
 - SetEdgesPairing
 - TDCCConfiguration, 34
 - SetKeyValue
 - SocketMessage, 28, 29
 - SetLeadingMode
 - TDCCConfiguration, 34
 - SetMaxEventSize
 - TDCCConfiguration, 34
 - SetPort
 - Socket, 23
 - SetRCAdjustment
 - TDCCConfiguration, 34
 - SetRejectFIFOFull
 - TDCCConfiguration, 34
 - SetSocketId
 - Socket, 23
 - SetTrailingMode
 - TDCCConfiguration, 34
 - SetTriggerMatchingMode
 - TDCCConfiguration, 34
 - SetWidthResolution
 - TDCCConfiguration, 34
 - SetWord
 - TDCCConfiguration, 34
- Socket, 19
 - ~Socket, 21
 - AcceptConnections, 21
 - Bind, 21
 - DumpConnected, 21
 - fBuffer, 23
 - fMaster, 23
 - fPort, 23
 - fReadFds, 23
 - fSocketsConnected, 23
 - FetchMessage, 21
 - GetPort, 22
 - GetSocketId, 22
 - GetSocketType, 22
 - IsWebSocket, 22
 - Listen, 22
 - PrepareConnection, 22
 - SelectConnections, 22
 - SendMessage, 22
 - SetPort, 23
 - SetSocketId, 23
 - Socket, 21
 - Start, 23

- Stop, [23](#)
- SocketMessage, [24](#)
 - ~SocketMessage, [27](#)
 - Dump, [27](#)
 - GetIntValue, [27](#)
 - GetKey, [27](#)
 - GetString, [27](#)
 - GetValue, [28](#)
 - GetVectorValue, [28](#)
 - SetKeyValue, [28](#), [29](#)
 - SocketMessage, [25–27](#)
- spill_id
 - file_header_t, [10](#)
- Start
 - Socket, [23](#)
- Stop
 - Socket, [23](#)
- TDCConfiguration, [30](#)
 - ~TDCConfiguration, [32](#)
 - DT_100NS, [31](#)
 - DT_10NS, [31](#)
 - DT_30NS, [31](#)
 - DT_5NS, [31](#)
 - DeadTime, [31](#)
 - Dump, [32](#)
 - E_100PS, [32](#)
 - E_12500PS, [32](#)
 - E_1600PS, [32](#)
 - E_200PS, [32](#)
 - E_3120PS, [32](#)
 - E_400PS, [32](#)
 - E_6250PS, [32](#)
 - E_800PS, [32](#)
 - EdgeResolution, [31](#)
 - GetChannelOffset, [32](#)
 - GetDLLAdjustment, [32](#)
 - GetDeadTime, [32](#)
 - GetEdgeResolution, [32](#)
 - GetEdgesPairing, [32](#)
 - GetLeadingMode, [33](#)
 - GetMaxEventSize, [33](#)
 - GetNumWords, [33](#)
 - GetRCAdjustment, [33](#)
 - GetRejectFIFOFull, [33](#)
 - GetTrailingMode, [33](#)
 - GetTriggerMatchingMode, [33](#)
 - GetWidthResolution, [33](#)
 - GetWord, [33](#)
 - SetAllChannelsOffset, [33](#)
 - SetChannelOffset, [33](#)
 - SetDLLAdjustment, [34](#)
 - SetDeadTime, [34](#)
 - SetEdgeResolution, [34](#)
 - SetEdgesPairing, [34](#)
 - SetLeadingMode, [34](#)
 - SetMaxEventSize, [34](#)
 - SetRCAdjustment, [34](#)
 - SetRejectFIFOFull, [34](#)
 - SetTrailingMode, [34](#)
 - SetTriggerMatchingMode, [34](#)
 - SetWidthResolution, [34](#)
 - SetWord, [34](#)
 - TDCConfiguration, [32](#)
 - W_100NS, [32](#)
 - W_100PS, [32](#)
 - W_12p5NS, [32](#)
 - W_1p6NS, [32](#)
 - W_200NS, [32](#)
 - W_200PS, [32](#)
 - W_25NS, [32](#)
 - W_3p2NS, [32](#)
 - W_400NS, [32](#)
 - W_400PS, [32](#)
 - W_50NS, [32](#)
 - W_6p25NS, [32](#)
 - W_800NS, [32](#)
 - W_800PS, [32](#)
 - WidthResolution, [32](#)
- TDCEvent, [35](#)
 - ~TDCEvent, [36](#)
 - Debug, [35](#)
 - Error, [35](#)
 - EventType, [35](#)
 - GetBunchId, [36](#)
 - GetErrorFlags, [36](#)
 - GetEventId, [36](#)
 - GetLeadingTime, [36](#)
 - GetTDCId, [37](#)
 - GetTrailingTime, [37](#)
 - GetType, [37](#)
 - GetWidth, [38](#)
 - GetWordCount, [38](#)
 - GroupHeader, [35](#)
 - GroupTrailer, [35](#)
 - Invalid, [35](#)
 - LeadingEdge, [35](#)
 - TDCEvent, [36](#)
 - TDCHeader, [35](#)
 - TDCTrailer, [35](#)
 - TrailingEdge, [35](#)
- TDCHeader
 - TDCEvent, [35](#)
- TDCTrailer
 - TDCEvent, [35](#)
- TrailingEdge
 - TDCEvent, [35](#)
- Type
 - Exception, [9](#)
- type
 - ListenerInfo, [15](#)
- TypeString
 - Exception, [9](#)
- W_100NS
 - TDCConfiguration, [32](#)
- W_100PS
 - TDCConfiguration, [32](#)

W_12p5NS
 TDCConfiguration, [32](#)
W_1p6NS
 TDCConfiguration, [32](#)
W_200NS
 TDCConfiguration, [32](#)
W_200PS
 TDCConfiguration, [32](#)
W_25NS
 TDCConfiguration, [32](#)
W_3p2NS
 TDCConfiguration, [32](#)
W_400NS
 TDCConfiguration, [32](#)
W_400PS
 TDCConfiguration, [32](#)
W_50NS
 TDCConfiguration, [32](#)
W_6p25NS
 TDCConfiguration, [32](#)
W_800NS
 TDCConfiguration, [32](#)
W_800PS
 TDCConfiguration, [32](#)
WidthResolution
 TDCConfiguration, [32](#)