2015 Test beam Run Control

Generated by Doxygen 1.8.9.1

Fri Apr 24 2015 00:45:56

Contents

1 Module Index **1**

1.1 Modules 1

2 Hierarchical Index **3**

2.1 Class Hierarchy 3

3 Data Structure Index **5**

3.1 Data Structures 5

4 Module Documentation **7**

4.1 Socket communication objects 7

4.1.1 Detailed Description 7

4.1.2 Enumeration Type Documentation 7

4.1.2.1 SocketType 7

4.2 HPTDC chip control 9

4.2.1 Detailed Description 10

4.2.2 Enumeration Type Documentation 10

4.2.2.1 CoreClockSource 10

4.2.2.2 DeadTime 10

4.2.2.3 DLLClockSource 10

4.2.2.4 DLLSpeedMode 10

4.2.2.5 EdgeResolution 11

4.2.2.6 EnabledError 11

4.2.2.7 EventType 11

4.2.2.8 IOClockSource 11

4.2.2.9 ReadoutSingleCycleSpeed 12

4.2.2.10 ReadoutSpeed 12

4.2.2.11 SerialClockSource 12

4.2.2.12 SerialStrobeType 12

4.2.2.13 WidthResolution 12

5 Data Structure Documentation **15**

5.1 Client Class Reference 15

5.1.1 Detailed Description 16
5.1.2 Constructor & Destructor Documentation 17
5.1.2.1 Client 17
5.1.2.2 Client 17
5.1.2.3 ∼Client 17
5.1.3 Member Function Documentation 17
5.1.3.1 Announce 17
5.1.3.2 Connect 18
5.1.3.3 Disconnect 18
5.1.3.4 GetType 19
5.1.3.5 ParseMessage 19
5.1.3.6 Receive 19
5.1.3.7 Send 20
5.1.4 Field Documentation 20
5.1.4.1 fClientId 20
5.1.4.2 fIsConnected 20
5.2 Exception Class Reference 20
5.2.1 Detailed Description 21
5.2.2 Constructor & Destructor Documentation 21
5.2.2.1 Exception 21
5.2.2.2 Exception 21
5.2.2.3 ∼Exception 21
5.2.3 Member Function Documentation 21
5.2.3.1 Description 21
5.2.3.2 Dump 22
5.2.3.3 ErrorNumber 22
5.2.3.4 From 22
5.2.3.5 Type 22
5.2.3.6 TypeString 22
5.2.4 Field Documentation 22
5.2.4.1 fDescription 22
5.2.4.2 fErrorNumber 22
5.2.4.3 fFrom 22
5.2.4.4 fType 22
5.3 file_header_t Struct Reference 23
5.3.1 Detailed Description 23
5.3.2 Field Documentation 23
5.3.2.1 config 23
5.3.2.2 magic 23
5.3.2.3 run_id 23

5.3.2.4 spill_id . . . 23
5.4 FPGAHandler Class Reference . . . 24
5.4.1 Detailed Description . . . 25
5.4.2 Constructor & Destructor Documentation . . . 26
5.4.2.1 FPGAHandler . . . 26
5.4.2.2 ∼FPGAHandler . . . 26
5.4.3 Member Function Documentation . . . 26
5.4.3.1 CloseFile . . . 26
5.4.3.2 GetConfiguration . . . 26
5.4.3.3 GetFilename . . . 27
5.4.3.4 GetType . . . 27
5.4.3.5 OpenFile . . . 27
5.4.3.6 ReadBuffer . . . 27
5.4.3.7 ReadConfiguration . . . 27
5.4.3.8 SendConfiguration . . . 27
5.4.3.9 SetConfiguration . . . 28
5.4.4 Field Documentation . . . 28
5.4.4.1 fConfig . . . 28
5.4.4.2 fFilename . . . 28
5.4.4.3 fIsFileOpen . . . 28
5.4.4.4 fIsTDCInReadout . . . 28
5.4.4.5 fOutput . . . 28
5.5 HTTPMessage Class Reference . . . 29
5.5.1 Detailed Description . . . 30
5.5.2 Constructor & Destructor Documentation . . . 30
5.5.2.1 HTTPMessage . . . 30
5.5.2.2 HTTPMessage . . . 30
5.5.3 Member Function Documentation . . . 31
5.5.3.1 Decode . . . 31
5.5.3.2 Dump . . . 31
5.5.3.3 Encode . . . 31
5.5.3.4 GetKey . . . 31
5.5.4 Field Documentation . . . 31
5.5.4.1 fOriginalString . . . 31
5.5.4.2 fWS . . . 31
5.6 ListenerInfo Struct Reference . . . 31
5.6.1 Detailed Description . . . 31
5.6.2 Field Documentation . . . 31
5.6.2.1 name . . . 31
5.6.2.2 type . . . 31

5.7 Message Class Reference . . . 31
5.7.1 Detailed Description . . . 32
5.7.2 Constructor & Destructor Documentation . . . 33
5.7.2.1 Message . . . 33
5.7.2.2 Message . . . 33
5.7.2.3 Message . . . 33
5.7.2.4 ∼Message . . . 33
5.7.3 Member Function Documentation . . . 33
5.7.3.1 Dump . . . 33
5.7.3.2 GetKey . . . 33
5.7.3.3 GetString . . . 33
5.7.3.4 IsFromWeb . . . 33
5.7.4 Field Documentation . . . 33
5.7.4.1 fString . . . 33
5.8 Messenger Class Reference . . . 33
5.8.1 Detailed Description . . . 35
5.8.2 Constructor & Destructor Documentation . . . 35
5.8.2.1 Messenger . . . 35
5.8.2.2 Messenger . . . 35
5.8.2.3 ∼Messenger . . . 36
5.8.3 Member Function Documentation . . . 36
5.8.3.1 AddClient . . . 36
5.8.3.2 Broadcast . . . 36
5.8.3.3 Connect . . . 37
5.8.3.4 Disconnect . . . 37
5.8.3.5 DisconnectClient . . . 38
5.8.3.6 GetType . . . 38
5.8.3.7 ProcessMessage . . . 38
5.8.3.8 Receive . . . 39
5.8.3.9 Send . . . 39
5.8.3.10 SwitchClientType . . . 40
5.8.4 Field Documentation . . . 40
5.8.4.1 fListenersInfo . . . 40
5.8.4.2 fNumAttempts . . . 40
5.8.4.3 fWS . . . 40
5.9 Socket Class Reference . . . 41
5.9.1 Detailed Description . . . 42
5.9.2 Member Typedef Documentation . . . 43
5.9.2.1 SocketCollection . . . 43
5.9.3 Constructor & Destructor Documentation . . . 43

5.9.3.1 Socket . . . 43
5.9.3.2 Socket . . . 43
5.9.3.3 ∼Socket . . . 43
5.9.4 Member Function Documentation . . . 43
5.9.4.1 AcceptConnections . . . 43
5.9.4.2 Bind . . . 43
5.9.4.3 Configure . . . 44
5.9.4.4 Create . . . 44
5.9.4.5 DumpConnected . . . 44
5.9.4.6 FetchMessage . . . 44
5.9.4.7 GetPort . . . 44
5.9.4.8 GetSocketId . . . 44
5.9.4.9 GetSocketType . . . 44
5.9.4.10 IsWebSocket . . . 44
5.9.4.11 Listen . . . 44
5.9.4.12 PrepareConnection . . . 45
5.9.4.13 SelectConnections . . . 45
5.9.4.14 SendMessage . . . 45
5.9.4.15 SetPort . . . 45
5.9.4.16 SetSocketId . . . 45
5.9.4.17 Start . . . 46
5.9.4.18 Stop . . . 46
5.9.5 Field Documentation . . . 46
5.9.5.1 fAddress . . . 46
5.9.5.2 fBuffer . . . 46
5.9.5.3 fMaster . . . 46
5.9.5.4 fPort . . . 46
5.9.5.5 fReadFds . . . 46
5.9.5.6 fSocketId . . . 46
5.9.5.7 fSocketsConnected . . . 46
5.10 SocketMessage Class Reference . . . 47
5.10.1 Detailed Description . . . 48
5.10.2 Constructor & Destructor Documentation . . . 49
5.10.2.1 SocketMessage . . . 49
5.10.2.2 SocketMessage . . . 49
5.10.2.3 SocketMessage . . . 49
5.10.2.4 SocketMessage . . . 49
5.10.2.5 SocketMessage . . . 49
5.10.2.6 SocketMessage . . . 50
5.10.2.7 SocketMessage . . . 50

5.10.2.8 SocketMessage 50
5.10.2.9 SocketMessage 50
5.10.2.10 SocketMessage 51
5.10.2.11 SocketMessage 51
5.10.2.12 ∼SocketMessage 51
5.10.3 Member Function Documentation 51
5.10.3.1 Dump 51
5.10.3.2 GetIntValue 51
5.10.3.3 GetKey 52
5.10.3.4 GetString 52
5.10.3.5 GetValue 52
5.10.3.6 GetVectorValue 52
5.10.3.7 Object 52
5.10.3.8 SetKeyValue 52
5.10.3.9 SetKeyValue 52
5.10.3.10 SetKeyValue 53
5.10.3.11 SetKeyValue 53
5.10.3.12 String 53
5.10.4 Field Documentation 53
5.10.4.1 fMessage 53
5.11 TDCConfiguration Class Reference 53
5.11.1 Detailed Description 60
5.11.2 Member Typedef Documentation 60
5.11.2.1 bit 60
5.11.2.2 word_t 60
5.11.3 Constructor & Destructor Documentation 61
5.11.3.1 TDCConfiguration 61
5.11.3.2 TDCConfiguration 62
5.11.3.3 ∼TDCConfiguration 62
5.11.4 Member Function Documentation 62
5.11.4.1 Dump 63
5.11.4.2 GetBits 63
5.11.4.3 GetChannelOffset 63
5.11.4.4 GetCoarseCountOffset 64
5.11.4.5 GetDeadTime 64
5.11.4.6 GetDLLAdjustment 64
5.11.4.7 GetEdgeResolution 65
5.11.4.8 GetEdgesPairing 65
5.11.4.9 GetEnableError 65
5.11.4.10 GetEnableErrorBypass 66

5.11.4.11 GetEnableErrorMark 66
5.11.4.12 GetEnableJTAGReadout 66
5.11.4.13 GetEnableReadoutOccupancy 67
5.11.4.14 GetEnableReadoutSeparator 67
5.11.4.15 GetEnableSerial 67
5.11.4.16 GetLeadingMode 67
5.11.4.17 GetMatchWindow 68
5.11.4.18 GetMaxEventSize 68
5.11.4.19 GetNumWords 68
5.11.4.20 GetRCAdjustment 68
5.11.4.21 GetReadoutFIFOSize 69
5.11.4.22 GetRejectCountOffset 69
5.11.4.23 GetRejectFIFOFull 69
5.11.4.24 GetSearchWindow 70
5.11.4.25 GetSetupParity 70
5.11.4.26 GetTestInvert 71
5.11.4.27 GetTestMode 71
5.11.4.28 GetTrailingMode 71
5.11.4.29 GetTriggerCountOffset 71
5.11.4.30 GetTriggerLatency 72
5.11.4.31 GetTriggerMatchingMode 72
5.11.4.32 GetVernierOffset 72
5.11.4.33 GetWidthResolution 73
5.11.4.34 GetWord 73
5.11.4.35 SetAllChannelsOffset 73
5.11.4.36 SetAllTapsDLLAdjustment 73
5.11.4.37 SetBits 74
5.11.4.38 SetBypassInputs 74
5.11.4.39 SetChannelOffset 74
5.11.4.40 SetCoarseCountOffset 75
5.11.4.41 SetConstantValues 75
5.11.4.42 SetCoreClockDelay 76
5.11.4.43 SetCoreClockSource 77
5.11.4.44 SetDeadTime 77
5.11.4.45 SetDLLAdjustment 77
5.11.4.46 SetDLLClockDelay 78
5.11.4.47 SetDLLClockSource 78
5.11.4.48 SetDLLControl 78
5.11.4.49 SetDLLMode 79
5.11.4.50 SetEdgeResolution 79

5.11.4.51 SetEdgesPairing . . . 79
5.11.4.52 SetEnableAutomaticReject . . . 80
5.11.4.53 SetEnableBytewise . . . 80
5.11.4.54 SetEnableDirectBunchReset . . . 80
5.11.4.55 SetEnableDirectEventReset . . . 81
5.11.4.56 SetEnableDirectTrigger . . . 81
5.11.4.57 SetEnableError . . . 81
5.11.4.58 SetEnableErrorBypass . . . 82
5.11.4.59 SetEnableErrorMark . . . 82
5.11.4.60 SetEnableGlobalHeader . . . 82
5.11.4.61 SetEnableGlobalTrailer . . . 83
5.11.4.62 SetEnableJTAGReadout . . . 83
5.11.4.63 SetEnableLocalHeader . . . 83
5.11.4.64 SetEnableLocalTrailer . . . 84
5.11.4.65 SetEnableMasterResetCode . . . 84
5.11.4.66 SetEnableMasterResetOnEventReset . . . 84
5.11.4.67 SetEnableOverflowDetect . . . 85
5.11.4.68 SetEnableReadoutOccupancy . . . 85
5.11.4.69 SetEnableReadoutSeparator . . . 85
5.11.4.70 SetEnableRelative . . . 86
5.11.4.71 SetEnableResetChannelBufferWhenSeparator . . . 86
5.11.4.72 SetEnableSeparatorOnBunchReset . . . 86
5.11.4.73 SetEnableSeparatorOnEventReset . . . 87
5.11.4.74 SetEnableSerial . . . 87
5.11.4.75 SetEnableSetCountersOnBunchReset . . . 87
5.11.4.76 SetEnableTTLClock . . . 88
5.11.4.77 SetEnableTTLControl . . . 88
5.11.4.78 SetEnableTTLHit . . . 89
5.11.4.79 SetEnableTTLReset . . . 89
5.11.4.80 SetEnableTTLSerial . . . 89
5.11.4.81 SetEventCountOffset . . . 90
5.11.4.82 SetIOClockDelay . . . 90
5.11.4.83 SetIOClockSource . . . 90
5.11.4.84 SetKeepToken . . . 91
5.11.4.85 SetLeadingMode . . . 91
5.11.4.86 SetLowPowerMode . . . 91
5.11.4.87 SetMaster . . . 92
5.11.4.88 SetMatchWindow . . . 92
5.11.4.89 SetMaxEventSize . . . 92
5.11.4.90 SetModeRC . . . 93

5.11.4.91 SetModeRCCompression 93
5.11.4.92 SetPLLControl 93
5.11.4.93 SetRCAdjustment 94
5.11.4.94 SetReadoutFIFOSize 94
5.11.4.95 SetReadoutSingleCycleSpeed 94
5.11.4.96 SetReadoutSpeedSelect 95
5.11.4.97 SetRejectCountOffset 95
5.11.4.98 SetRejectFIFOFull 96
5.11.4.99 SetRollOver 96
5.11.4.100SetSearchWindow 96
5.11.4.101SetSerialClockDelay 96
5.11.4.102SetSerialClockSource 97
5.11.4.103SetSerialDelay 97
5.11.4.104SetSetupParity 97
5.11.4.105SetStrobeSelect 98
5.11.4.106SetTestInvert 98
5.11.4.107SetTestMode 98
5.11.4.108SetTokenDelay 99
5.11.4.109SetTrailingMode 99
5.11.4.110SetTriggerCountOffset 99
5.11.4.111SetTriggerMatchingMode 100
5.11.4.112SetVernierOffset 100
5.11.4.113SetWidthResolution 100
5.11.4.114SetWord 101
5.11.5 Field Documentation 101
5.11.5.1 fWord 101
5.11.5.2 kCoarseCountOffset 101
5.11.5.3 kCoreClockDelay 101
5.11.5.4 kCoreClockSource 101
5.11.5.5 kDeadTime 101
5.11.5.6 kDLLClockDelay 101
5.11.5.7 kDLLClockSource 101
5.11.5.8 kDLLControl 101
5.11.5.9 kDLLMode 101
5.11.5.10 kDLLTapAdjust0 101
5.11.5.11 kEnableAutomaticReject 101
5.11.5.12 kEnableBytewise 101
5.11.5.13 kEnableDirectBunchReset 101
5.11.5.14 kEnableDirectEventReset 101
5.11.5.15 kEnableDirectTrigger 101

5.11.5.16 kEnableError 101
5.11.5.17 kEnableErrorBypass 101
5.11.5.18 kEnableErrorMark 101
5.11.5.19 kEnableGlobalHeader 102
5.11.5.20 kEnableGlobalTrailer 102
5.11.5.21 kEnableJTAGReadout 102
5.11.5.22 kEnableLocalHeader 102
5.11.5.23 kEnableLocalTrailer 102
5.11.5.24 kEnableMasterResetCode 102
5.11.5.25 kEnableMasterResetOnEventReset 102
5.11.5.26 kEnableMatching 102
5.11.5.27 kEnableOverflowDetect 102
5.11.5.28 kEnablePair 102
5.11.5.29 kEnableReadoutOccupancy 102
5.11.5.30 kEnableReadoutSeparator 102
5.11.5.31 kEnableRelative 102
5.11.5.32 kEnableResetChannelBufferWhenSeparator 102
5.11.5.33 kEnableSeparatorOnBunchReset 102
5.11.5.34 kEnableSeparatorOnEventReset 102
5.11.5.35 kEnableSerial 102
5.11.5.36 kEnableSetCountersOnBunchReset 102
5.11.5.37 kEnableTTLClock 102
5.11.5.38 kEnableTTLControl 102
5.11.5.39 kEnableTTLHit 102
5.11.5.40 kEnableTTLReset 102
5.11.5.41 kEnableTTLSerial 102
5.11.5.42 kEventCountOffset 102
5.11.5.43 kIOClockDelay 102
5.11.5.44 kIOClockSource 102
5.11.5.45 kKeepToken 102
5.11.5.46 kLeading 102
5.11.5.47 kLeadingResolution 103
5.11.5.48 kLowPowerMode 103
5.11.5.49 kMaster 103
5.11.5.50 kMatchWindow 103
5.11.5.51 kMaxEventSize 103
5.11.5.52 kModeRC 103
5.11.5.53 kModeRCCompression 103
5.11.5.54 kNumWords 103
5.11.5.55 kOffset0 103

5.11.5.56 kPLLControl 103
5.11.5.57 kRCAdjust0 103
5.11.5.58 kReadoutFIFOSize 103
5.11.5.59 kReadoutSingleCycleSpeed 103
5.11.5.60 kReadoutSpeedSelect 103
5.11.5.61 kRejectCountOffset 103
5.11.5.62 kRejectFIFOFull 103
5.11.5.63 kRollOver 103
5.11.5.64 kSearchWindow 103
5.11.5.65 kSelectBypassInputs 103
5.11.5.66 kSerialClockDelay 103
5.11.5.67 kSerialClockSource 103
5.11.5.68 kSerialDelay 103
5.11.5.69 kSetupParity 103
5.11.5.70 kStrobeSelect 103
5.11.5.71 kTDCId 103
5.11.5.72 kTestInvert 103
5.11.5.73 kTestMode 103
5.11.5.74 kTestSelect 103
5.11.5.75 kTokenDelay 104
5.11.5.76 kTrailing 104
5.11.5.77 kTriggerCountOffset 104
5.11.5.78 kVernierOffset 104
5.11.5.79 kWidthSelect 104
5.12 TDCEvent Class Reference 104
5.12.1 Detailed Description 105
5.12.2 Constructor & Destructor Documentation 105
5.12.2.1 TDCEvent 105
5.12.2.2 ∼TDCEvent 105
5.12.3 Member Function Documentation 105
5.12.3.1 GetBunchId 105
5.12.3.2 GetErrorFlags 105
5.12.3.3 GetEventId 106
5.12.3.4 GetLeadingTime 106
5.12.3.5 GetTDCId 106
5.12.3.6 GetTrailingTime 106
5.12.3.7 GetType 107
5.12.3.8 GetWidth 107
5.12.3.9 GetWordCount 107
5.12.4 Field Documentation 107

5.12.4.1 fWord 107
5.13 USBHandler Class Reference 107
5.13.1 Detailed Description 108
5.13.2 Constructor & Destructor Documentation 108
5.13.2.1 USBHandler 108
5.13.2.2 ∼USBHandler 108
5.13.3 Member Function Documentation 109
5.13.3.1 DumpDevice 109
5.13.3.2 Fetch 109
5.13.3.3 Init 109
5.13.3.4 Write 109
5.13.4 Field Documentation 109
5.13.4.1 fDevice 109
5.13.4.2 fHandle 109

Index **111**

Chapter 1

Module Index

1.1 Modules

Here is a list of all modules:

Socket communication objects 7
HPTDC chip control 9

Chapter 2

Hierarchical Index

2.1 Class Hierarchy

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

- Exception . . . 20
- file_header_t . . . 23
- ListenerInfo . . . 31
- Message . . . 31
  - HTTPMessage . . . 29
  - SocketMessage . . . 47
- Socket . . . 41
  - Client . . . 15
    - FPGAHandler . . . 24
  - Messenger . . . 33
- TDCConfiguration . . . 53
- TDCEvent . . . 104
- USBHandler . . . 107
  - FPGAHandler . . . 24

Chapter 3

Data Structure Index

3.1 Data Structures

Here are the data structures with brief descriptions:

Client
Base client object for the socket 15
Exception
A simple exception handler 20
file_header_t
Header to the output files 23
FPGAHandler
Driver for timing detectors' FPGA readout 24
HTTPMessage
Message to be transmitted through a WebSocket protocol 29
ListenerInfo
Information on a socket's listener 31
Message
Base socket message type 31
Messenger
Base master object for the socket 33
Socket
Base socket object from which clients/master from a socket inherit 41
SocketMessage
Socket-passed message type 47
TDCConfiguration
Setup word to be sent to the HPTDC chip 53
TDCEvent
HPTDC event parser 104
USBHandler
Generic USB communication handler 107

Chapter 4

Module Documentation

4.1 Socket communication objects

Data Structures

- class Client

  Base client object for the socket.
- class HTTPMessage

  Message to be transmitted through a WebSocket protocol.
- struct ListenerInfo

  Information on a socket's listener.
- 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.

Enumerations

- enum Socket::SocketType {
  Socket::INVALID =-1, Socket::MASTER =0, Socket::WEBSOCKET_CLIENT, Socket::CLIENT,
  Socket::DETECTOR }

  Type of actor playing a role on the socket.

4.1.1 Detailed Description

4.1.2 Enumeration Type Documentation

4.1.2.1 enum Socket::SocketType

Type of actor playing a role on the socket.

Enumerator

INVALID

MASTER

WEBSOCKET_CLIENT

CLIENT

DETECTOR

4.2 HPTDC chip control

Data Structures

- class TDCConfiguration

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

  HPTDC event parser.

Enumerations

- enum TDCConfiguration::EdgeResolution {
  TDCConfiguration::E_100ps =0, TDCConfiguration::E_200ps, TDCConfiguration::E_400ps, TDC↩
  Configuration::E_800ps,
  TDCConfiguration::E_1p6ns, TDCConfiguration::E_3p12ns, TDCConfiguration::E_6p25ns, TDCConfiguration↩
  ::E_12p5ns }
- enum TDCConfiguration::DeadTime { TDCConfiguration::DT_5ns =0, TDCConfiguration::DT_10ns, TDC↩
  Configuration::DT_30ns, TDCConfiguration::DT_100ns }
- enum TDCConfiguration::WidthResolution {
  TDCConfiguration::W_100ps =0, TDCConfiguration::W_200ps, TDCConfiguration::W_400ps, TDC↩
  Configuration::W_800ps,
  TDCConfiguration::W_1p6ns, TDCConfiguration::W_3p2ns, TDCConfiguration::W_6p25ns, TDC↩
  Configuration::W_12p5ns,
  TDCConfiguration::W_25ns, TDCConfiguration::W_50ns, TDCConfiguration::W_100ns, TDCConfiguration↩
  ::W_200ns,
  TDCConfiguration::W_400ns, TDCConfiguration::W_800ns }
- enum TDCConfiguration::EnabledError {
  TDCConfiguration::VernierError =0x1, TDCConfiguration::CoarseError =0x2, TDCConfiguration::Channel↩
  SelectError =0x4, TDCConfiguration::L1BufferParityError =0x8,
  TDCConfiguration::TriggerFIFOParityError =0x10, TDCConfiguration::TriggerMatchingError =0x20, TDC↩
  Configuration::ReadoutFIFOParityError =0x40, TDCConfiguration::ReadoutStateError =0x80,
  TDCConfiguration::SetupParityError =0x100, TDCConfiguration::ControlParityError =0x200, TDC↩
  Configuration::JTAGInstructionParityError =0x400 }
- enum TDCConfiguration::DLLSpeedMode { TDCConfiguration::DLL_40MHz =0x0, TDCConfiguration::DLL↩
  _160MHz =0x1, TDCConfiguration::DLL_320MHz =0x2, TDCConfiguration::DLL_Illegal =0x3 }
- enum TDCConfiguration::SerialClockSource { TDCConfiguration::Serial_pll_clock_80 =0x0, TDC↩
  Configuration::Serial_pll_clock_160 =0x1, TDCConfiguration::Serial_pll_clock_40 =0x2, TDCConfiguration↩
  ::Serial_aux_clock =0x3 }
- enum TDCConfiguration::IOClockSource { TDCConfiguration::IO_clock_40 =0x0, TDCConfiguration::IO_↩
  pll_clock_80 =0x1, TDCConfiguration::IO_pll_clock_160 =0x2, TDCConfiguration::IO_aux_clock =0x3 }
- enum TDCConfiguration::CoreClockSource { TDCConfiguration::Core_clock_40 =0x0, TDCConfiguration↩
  ::Core_pll_clock_80 =0x1, TDCConfiguration::Core_pll_clock_160 =0x2, TDCConfiguration::Core_aux_clock
  =0x3 }
- enum TDCConfiguration::DLLClockSource {
  TDCConfiguration::DLL_clock_40 =0x0, TDCConfiguration::DLL_pll_clock_40 =0x1, TDCConfiguration::D↩
  LL_pll_clock_160 =0x2, TDCConfiguration::DLL_pll_clock_320 =0x3,
  TDCConfiguration::DLL_aux_clock =0x4 }
- enum TDCConfiguration::ReadoutSpeed { TDCConfiguration::RO_Fixed =0x0, TDCConfiguration::RO_pll↩
  _80Mbits_s =0x1 }
- enum TDCConfiguration::SerialStrobeType { TDCConfiguration::SS_NoStrobe =0x0, TDCConfiguration::S↩
  S_DSStrobe =0x1, TDCConfiguration::SS_LeadingTrailingStrobe =0x2, TDCConfiguration::SS_LeadingEdge
  =0x3 }
- enum TDCConfiguration::ReadoutSingleCycleSpeed {
  TDCConfiguration::RSC_40Mbits_s =0x0, TDCConfiguration::RSC_20Mbits_s =0x1, TDCConfiguration::R↩
  SC_10Mbits_s =0x2, TDCConfiguration::RSC_5Mbits_s =0x3,
  TDCConfiguration::RSC_2p5Mbits_s =0x4, TDCConfiguration::RSC_1p25Mbits_s =0x5, TDCConfiguration↩
  ::RSC_625kbits_s =0x6, TDCConfiguration::RSC_312p5kbits_s =0x7 }

- enum TDCEvent::EventType {
  TDCEvent::Invalid =-1, TDCEvent::GroupHeader =0, TDCEvent::GroupTrailer, TDCEvent::TDCHeader,
  TDCEvent::TDCTrailer, TDCEvent::LeadingEdge, TDCEvent::TrailingEdge, TDCEvent::Error,
  TDCEvent::Debug }

4.2.1 Detailed Description

4.2.2 Enumeration Type Documentation

4.2.2.1 enum TDCConfiguration::CoreClockSource

Enumerator

Core_clock_40

Core_pll_clock_80

Core_pll_clock_160

Core_aux_clock

4.2.2.2 enum TDCConfiguration::DeadTime

Enumerator

DT_5ns

DT_10ns

DT_30ns

DT_100ns

4.2.2.3 enum TDCConfiguration::DLLClockSource

Enumerator

DLL_clock_40

DLL_pll_clock_40

DLL_pll_clock_160

DLL_pll_clock_320

DLL_aux_clock

4.2.2.4 enum TDCConfiguration::DLLSpeedMode

Enumerator

DLL_40MHz

DLL_160MHz

DLL_320MHz

DLL_Illegal

4.2.2.5 enum TDCConfiguration::EdgeResolution

Enumerator

E_100ps

E_200ps

E_400ps

E_800ps

E_1p6ns

E_3p12ns

E_6p25ns

E_12p5ns

4.2.2.6 enum TDCConfiguration::EnabledError

Enumerator

VernierError

CoarseError

ChannelSelectError

L1BufferParityError

TriggerFIFOParityError

TriggerMatchingError

ReadoutFIFOParityError

ReadoutStateError

SetupParityError

ControlParityError

JTAGInstructionParityError

4.2.2.7 enum TDCEvent::EventType

Enumerator

Invalid

GroupHeader

GroupTrailer

TDCHeader

TDCTrailer

LeadingEdge

TrailingEdge

Error

Debug

4.2.2.8 enum TDCConfiguration::IOClockSource

Enumerator

IO_clock_40

IO_pll_clock_80

IO_pll_clock_160

IO_aux_clock

4.2.2.9 enum TDCConfiguration::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

4.2.2.10 enum TDCConfiguration::ReadoutSpeed

Enumerator

RO_Fixed

RO_pll_80Mbits_s

4.2.2.11 enum TDCConfiguration::SerialClockSource

Enumerator

Serial_pll_clock_80

Serial_pll_clock_160

Serial_pll_clock_40

Serial_aux_clock

4.2.2.12 enum TDCConfiguration::SerialStrobeType

Enumerator

SS_NoStrobe

SS_DSStrobe

SS_LeadingTrailingStrobe

SS_LeadingEdge

4.2.2.13 enum TDCConfiguration::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

Chapter 5

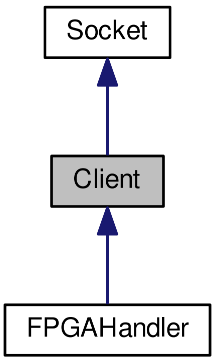
Data Structure Documentation

5.1 Client Class Reference

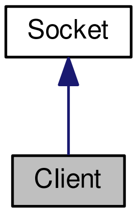
Base client object for the socket.

`#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 ()

  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 Receive ()

  Receive a socket message from the master.
- 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

Additional Inherited Members

5.1.1 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

5.1.2 Constructor & Destructor Documentation

5.1.2.1 Client::Client ( ) [inline]

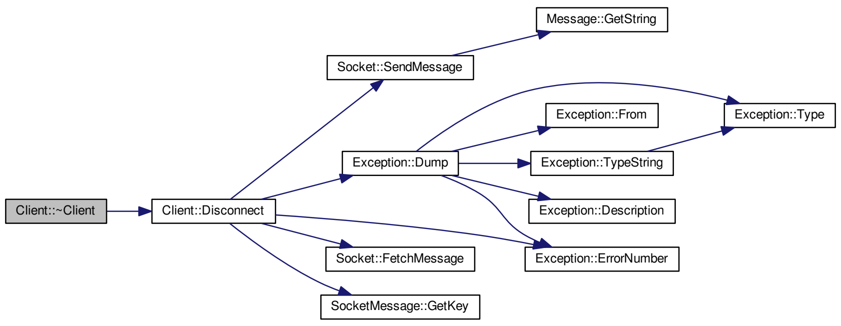
General void client constructor.

5.1.2.2 Client::Client (int *port*)

Bind a socket client to a given port.

5.1.2.3 Client::∼Client ( ) [virtual]

Here is the call graph for this function:

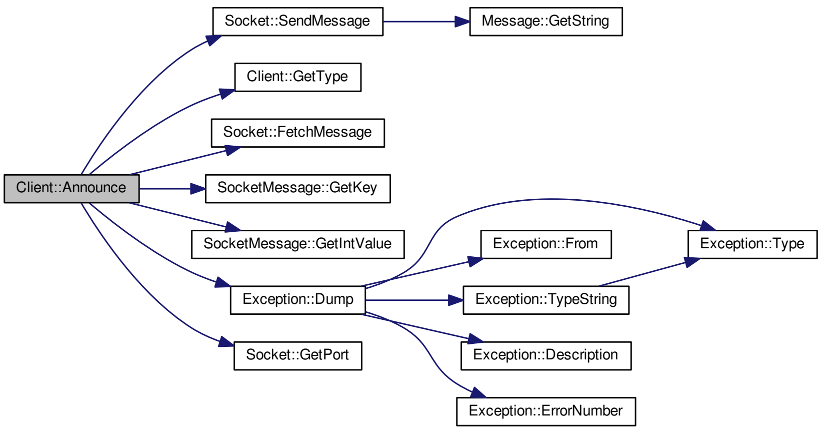


5.1.3 Member Function Documentation

5.1.3.1 void Client::Announce ( ) [private]

Announce our entry on the socket to its master.

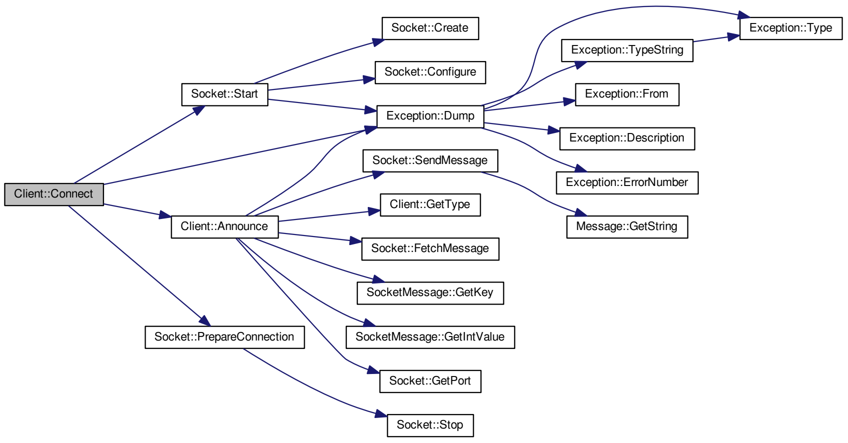
Here is the call graph for this function:



5.1.3.2 bool Client::Connect ( )

Bind this client to the socket.

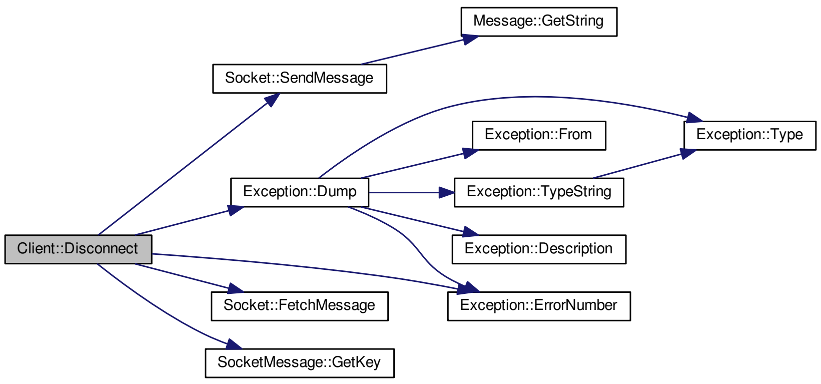
Here is the call graph for this function:



5.1.3.3 void Client::Disconnect ( )

Unbind this client from the socket.

Here is the call graph for this function:



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

Socket actor type retrieval method.

Reimplemented in FPGAHandler.

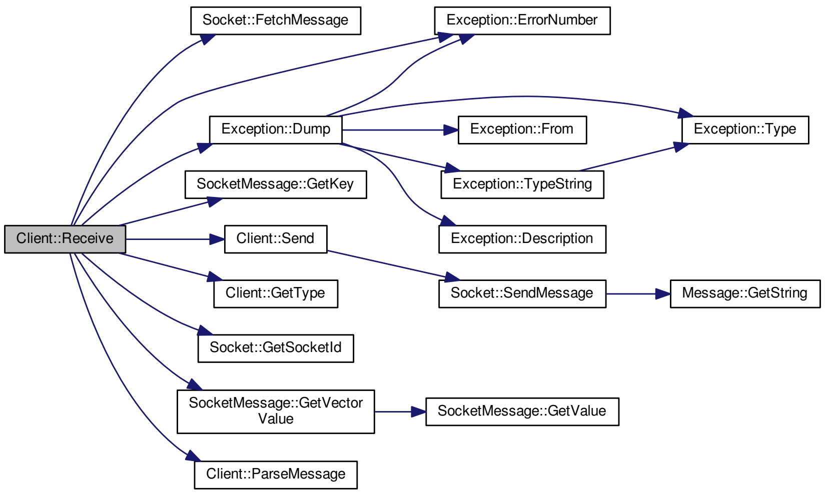
5.1.3.5 virtual void Client::ParseMessage (const SocketMessage & *m*) [inline], [virtual]

Parse a SocketMessage received from the master.

5.1.3.6 void Client::Receive ( )

Receive a socket message from the master.

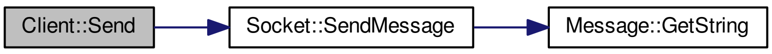
Here is the call graph for this function:



5.1.3.7 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:



5.1.4 Field Documentation

5.1.4.1 int Client::fClientId `[private]`

5.1.4.2 bool Client::fIsConnected `[private]`

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

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

5.2 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

Private Attributes

- std::string fFrom
- std::string fDescription
- ExceptionType fType
- int fErrorNumber

5.2.1 Detailed Description

A simple exception handler.

Author

Laurent Forthomme laurent.forthomme@cern.ch

Date

24 Mar 2015

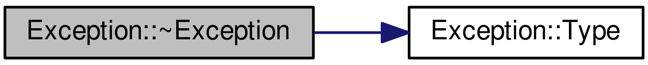
5.2.2 Constructor & Destructor Documentation

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

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

5.2.2.3 Exception::∼Exception ( ) `[inline]`

Here is the call graph for this function:

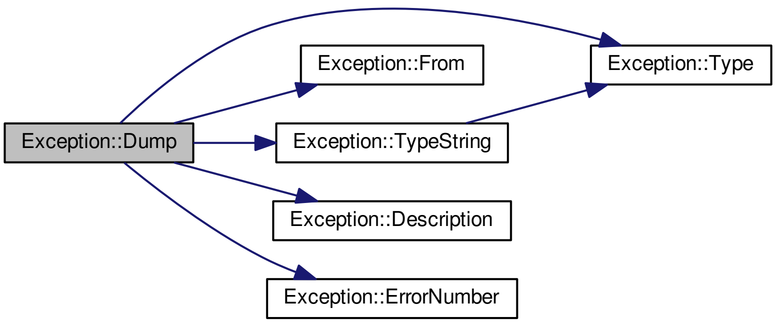


5.2.3 Member Function Documentation

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

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

Here is the call graph for this function:



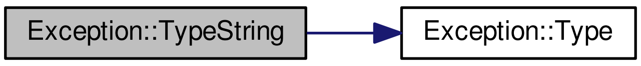
5.2.3.3 int Exception::ErrorNumber ( ) const `[inline]`

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

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

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

Here is the call graph for this function:



5.2.4 Field Documentation

5.2.4.1 std::string Exception::fDescription `[private]`

5.2.4.2 int Exception::fErrorNumber `[private]`

5.2.4.3 std::string Exception::fFrom `[private]`

5.2.4.4 ExceptionType Exception::fType `[private]`

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

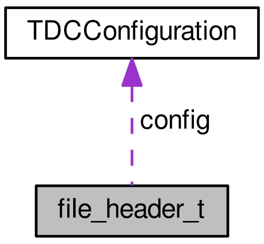
- include/Exception.h

5.3 file_header_t Struct Reference

Header to the output files.

`#include <FPGAHandler.h>`

Collaboration diagram for file_header_t:



Data Fields

- uint32_t magic
- uint32_t run_id
- uint32_t spill_id
- TDCConfiguration config

5.3.1 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

5.3.2 Field Documentation

5.3.2.1 TDCConfiguration file_header_t::config

5.3.2.2 uint32_t file_header_t::magic

5.3.2.3 uint32_t file_header_t::run_id

5.3.2.4 uint32_t file_header_t::spill_id

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

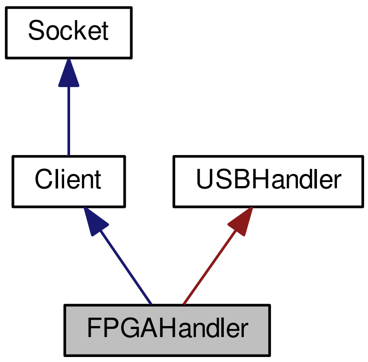
- include/FPGAHandler.h

5.4 FPGAHandler Class Reference

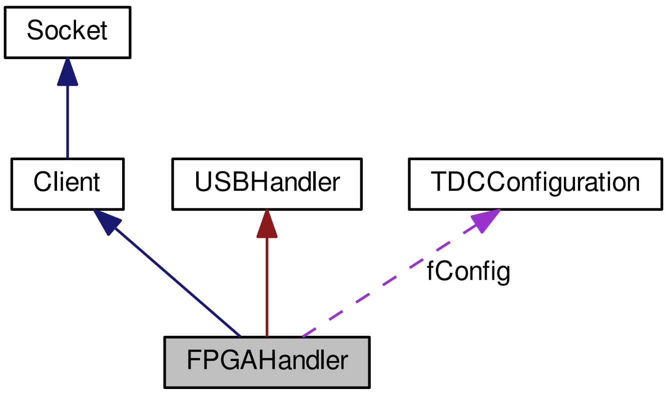
Driver for timing detectors' FPGA readout.

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

Inheritance diagram for FPGAHandler:



Collaboration diagram for FPGAHandler:



Public Member Functions

- FPGAHandler (int port, const char ∗dev)

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

- virtual ∼FPGAHandler ()
- 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.

- void SetConfiguration (const TDCConfiguration &c)

    Submit the HPTDC setup word as a TDCConfiguration object.

- TDCConfiguration GetConfiguration ()

    Retrieve the HPTDC setup word as a TDCConfiguration object.

- void ReadBuffer ()
- SocketType GetType () const

    Socket actor type retrieval method.

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.

Private Attributes

- std::string fFilename
- std::ofstream fOutput
- TDCConfiguration fConfig
- bool fIsFileOpen
- bool fIsTDCInReadout

Additional Inherited Members

5.4.1 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

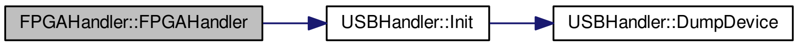
14 Apr 2015

5.4.2 Constructor & Destructor Documentation

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

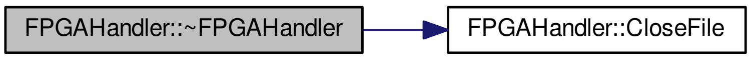
Bind to a FPGA through the USB protocol, and to the socket.

Here is the call graph for this function:



5.4.2.2 FPGAHandler::∼FPGAHandler ( ) [virtual]

Here is the call graph for this function:



5.4.3 Member Function Documentation

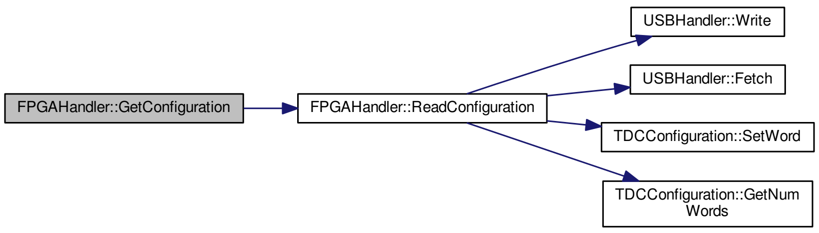
5.4.3.1 void FPGAHandler::CloseFile ( )

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

5.4.3.2 TDCConfiguration FPGAHandler::GetConfiguration ( ) [inline]

Retrieve the HPTDC setup word as a TDCConfiguration object.

Here is the call graph for this function:


FPGAHandler::GetConfiguration
FPGAHandler::ReadConfiguration
USBHandler::Write
USBHandler::Fetch
TDCConfiguration::SetWord
TDCConfiguration::GetNum Words

5.4.3.3 std::string FPGAHandler::GetFilename ( ) const [inline]

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

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

Socket actor type retrieval method.

Reimplemented from Client.

5.4.3.5 void FPGAHandler::OpenFile ( )

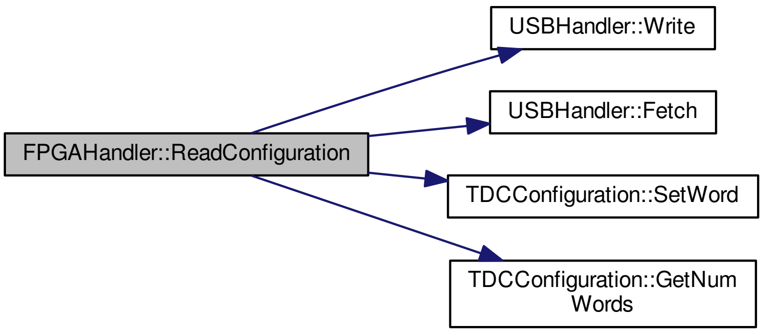
Open an output file to store header/HPTDC events.

5.4.3.6 void FPGAHandler::ReadBuffer ( )

5.4.3.7 void FPGAHandler::ReadConfiguration ( ) [private]

Read the setup word from the HPTDC internal setup register.

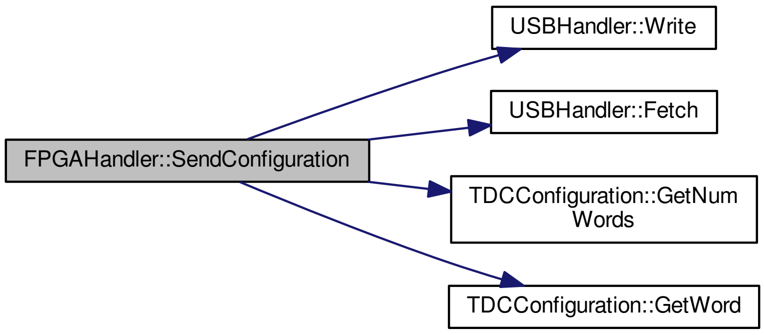
Here is the call graph for this function:


USBHandler::Write
USBHandler::Fetch
FPGAHandler::ReadConfiguration
TDCConfiguration::SetWord
TDCConfiguration::GetNum Words

5.4.3.8 void FPGAHandler::SendConfiguration ( ) [private]

Set the setup word to the HPTDC internal setup register.

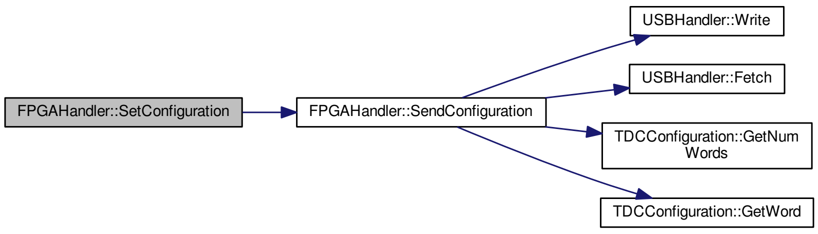
Here is the call graph for this function:


USBHandler::Write
USBHandler::Fetch
FPGAHandler::SendConfiguration
TDCConfiguration::GetNum Words
TDCConfiguration::GetWord

5.4.3.9 void FPGAHandler::SetConfiguration (const TDCConfiguration & *c*) [inline]

Submit the HPTDC setup word as a TDCConfiguration object.

Here is the call graph for this function:


FPGAHandler::SetConfiguration
FPGAHandler::SendConfiguration
USBHandler::Write
USBHandler::Fetch
TDCConfiguration::GetNum Words
TDCConfiguration::GetWord

5.4.4 Field Documentation

5.4.4.1 TDCConfiguration FPGAHandler::fConfig [private]

5.4.4.2 std::string FPGAHandler::fFilename [private]

5.4.4.3 bool FPGAHandler::fIsFileOpen [private]

5.4.4.4 bool FPGAHandler::fIsTDCInReadout [private]

5.4.4.5 std::ofstream FPGAHandler::fOutput [private]

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

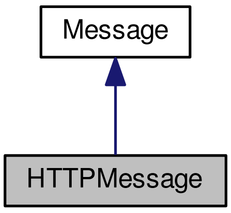
- include/FPGAHandler.h
- src/FPGAHandler.cpp

5.5 HTTPMessage Class Reference

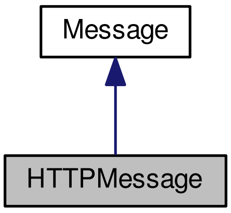
Message to be transmitted through a WebSocket protocol.

`#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

Private Attributes

- WebSocket ∗ fWS
- std::string fOriginalString

Additional Inherited Members

5.5.1 Detailed Description

Message to be transmitted through a WebSocket protocol.

Type of message compatible to the transmission through a WebSocket protocol. It enables a direct conversion of standards from any socket message format used elsewhere in this code using the *MessageAction* statement.

Author

Laurent Forthomme `laurent.forthomme@cern.ch`

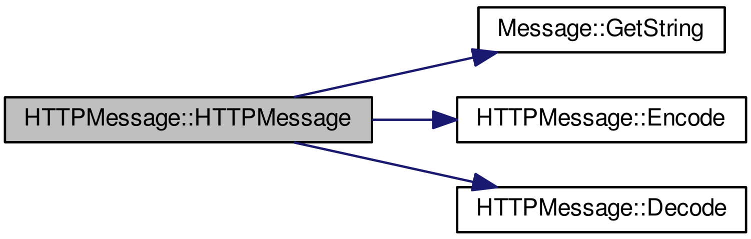
Date

1 Apr 2015

5.5.2 Constructor & Destructor Documentation

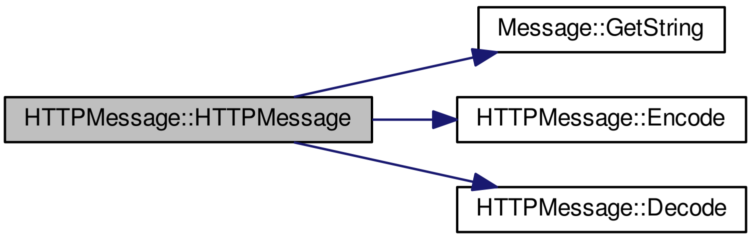
5.5.2.1 HTTPMessage::HTTPMessage (WebSocket ∗ *ws,* Message *m,* MessageAction *a*) `[inline]`

Here is the call graph for this function:



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

Here is the call graph for this function:



5.5.3 Member Function Documentation

5.5.3.1 void HTTPMessage::Decode ( ) `[inline]`

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

5.5.3.3 void HTTPMessage::Encode ( ) `[inline]`

5.5.3.4 MessageKey HTTPMessage::GetKey ( ) const `[inline]`

5.5.4 Field Documentation

5.5.4.1 std::string HTTPMessage::fOriginalString `[private]`

5.5.4.2 WebSocket* HTTPMessage::fWS `[private]`

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

- include/HTTPMessage.h

5.6 ListenerInfo Struct Reference

Information on a socket's listener.

`#include <Messenger.h>`

Data Fields

- std::string name
- Socket::SocketType type

5.6.1 Detailed Description

Information on a socket's listener.

Structure handling its name and type for any listener/client to be used in the socket management parts of this code.

5.6.2 Field Documentation

5.6.2.1 std::string ListenerInfo::name

5.6.2.2 Socket::SocketType ListenerInfo::type

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

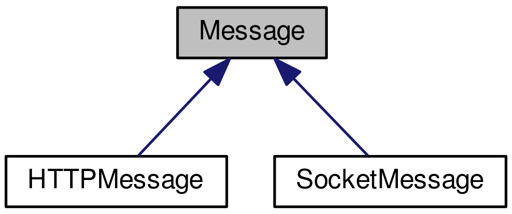
- include/Messenger.h

5.7 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

5.7.1 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

5.7.2 Constructor & Destructor Documentation

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

Void message constructor.

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

Construct a message from a string.

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

Construct a message from a string.

5.7.2.4 virtual Message::∼Message ( ) `[inline]`, `[virtual]`

5.7.3 Member Function Documentation

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

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

Placeholder for the MessageKey retrieval method.

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

Retrieve the string carried by this message as a whole.

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

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

5.7.4 Field Documentation

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

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

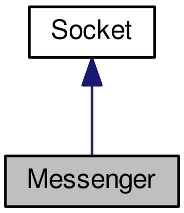
- include/Message.h

5.8 Messenger Class Reference

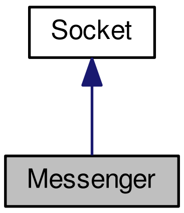
Base master object for the socket.

`#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 Receive ()

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

  Emit a message to all clients connected through the socket.
- 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

- WebSocket $*$ fWS
- int fNumAttempts
- std::vector$<$ ListenerInfo $>$ fListenersInfo

Additional Inherited Members

5.8.1 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

5.8.2 Constructor & Destructor Documentation

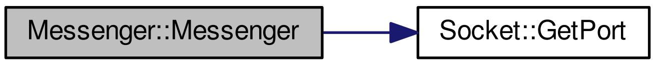
5.8.2.1 Messenger::Messenger ( )

Build a void master object or socket actor.

5.8.2.2 Messenger::Messenger (int *port*)

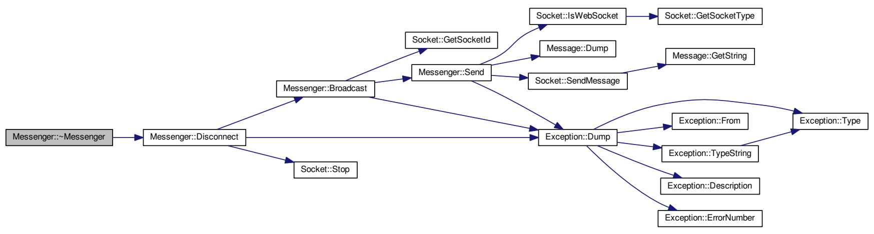
Build a master object to control the socket.

Here is the call graph for this function:



5.8.2.3 Messenger::∼Messenger ( )

Here is the call graph for this function:



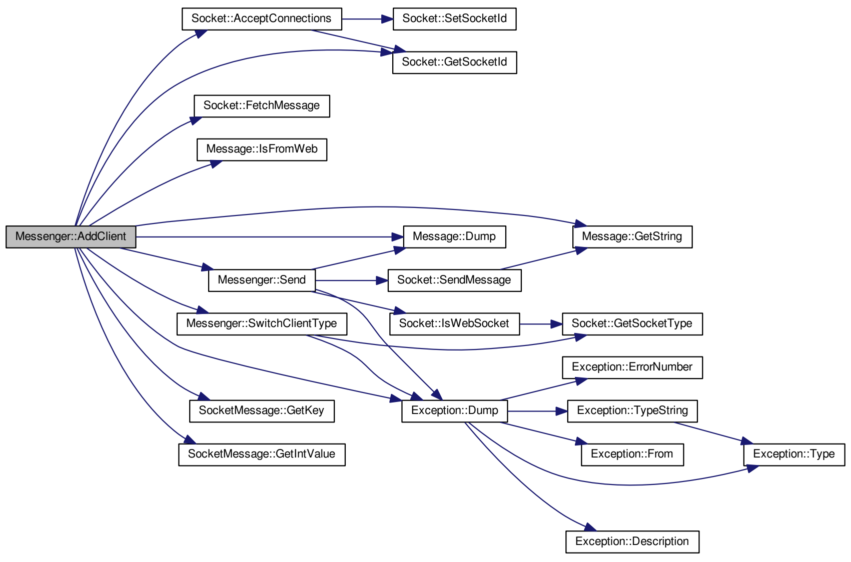
5.8.3 Member Function Documentation

5.8.3.1 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:



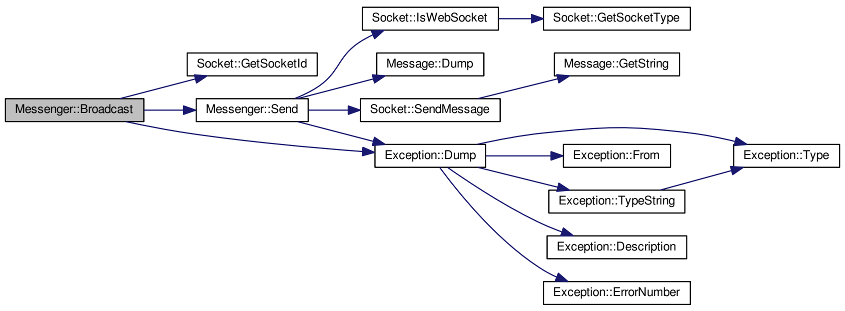
5.8.3.2 void Messenger::Broadcast (const Message & *m*) const

Emit a message to all clients connected through the socket.

Parameters

in	*m*	Message to transmit

Here is the call graph for this function:

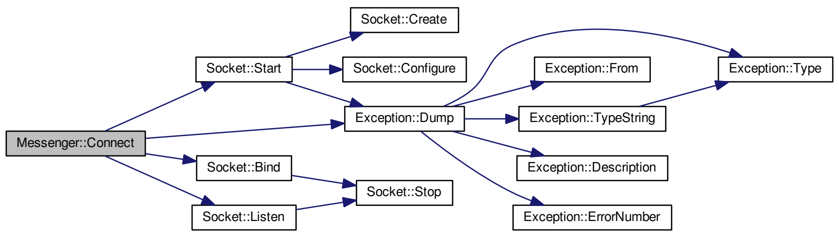


5.8.3.3 bool Messenger::Connect ( )

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:

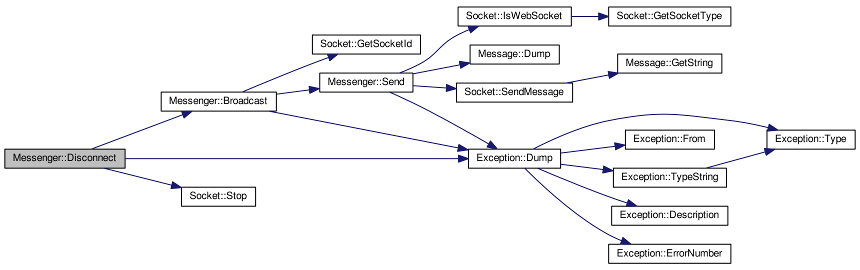


5.8.3.4 void Messenger::Disconnect ( )

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:



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

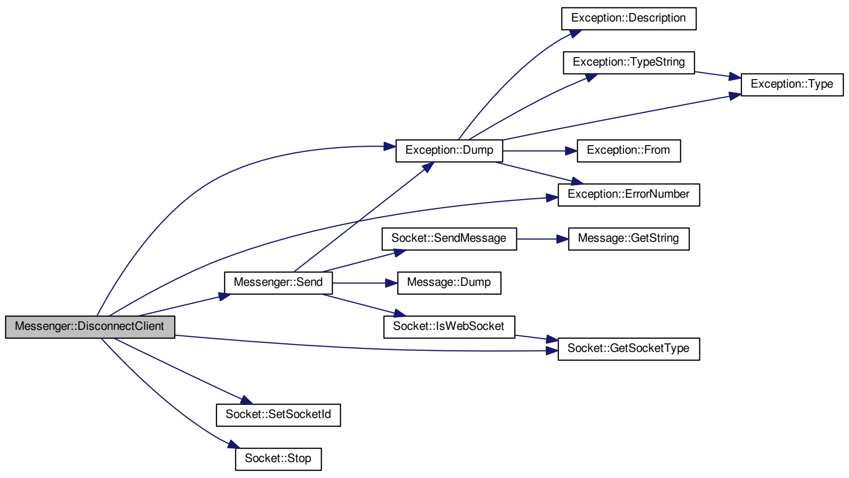
Disconnect a client.

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:



5.8.3.6 SocketType Messenger::GetType ( ) const `[inline]`

Socket actor type retrieval method.

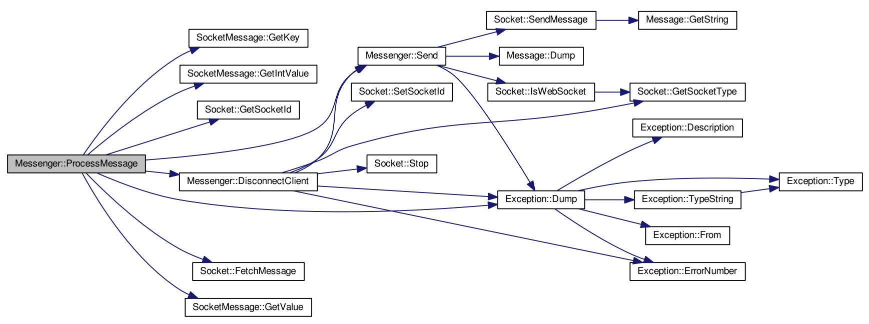
5.8.3.7 void Messenger::ProcessMessage (SocketMessage *m,* int *sid*) `[private]`

Process a message received from the socket.

Parameters

in	*Unique*	identifier of the client sending the message

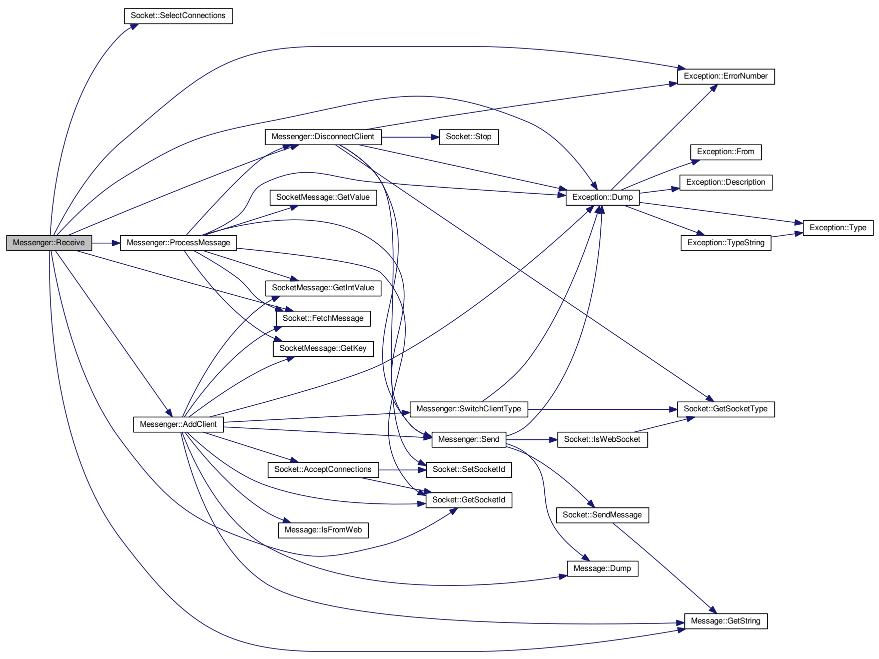
Here is the call graph for this function:



5.8.3.8 void Messenger::Receive ( )

Handle a message reception from a client.

Here is the call graph for this function:



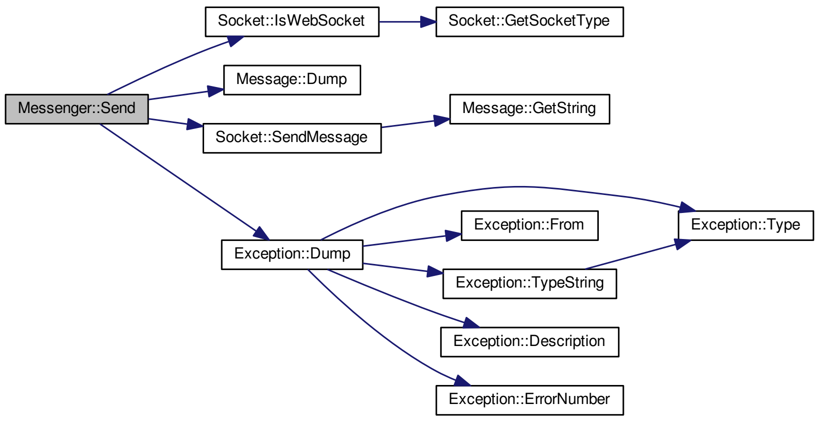
5.8.3.9 void Messenger::Send (const Message & *m*, int *sid*) const [inline]

Send any type of message to any client.

Parameters

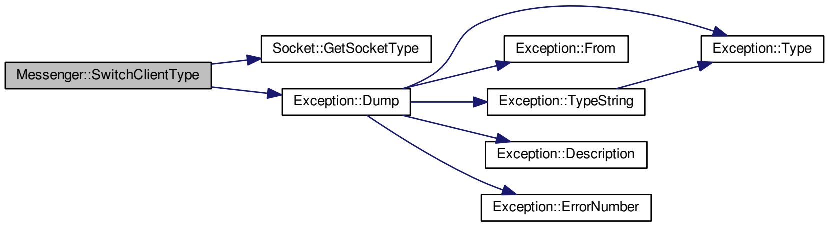
in	*m*	Message to transmit
in	*sid*	Unique identifier of the client on this socket

Here is the call graph for this function:



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

Here is the call graph for this function:



5.8.4 Field Documentation

5.8.4.1 std::vector< ListenerInfo > Messenger::fListenersInfo [private]

5.8.4.2 int Messenger::fNumAttempts [private]

5.8.4.3 WebSocket* Messenger::fWS [private]

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

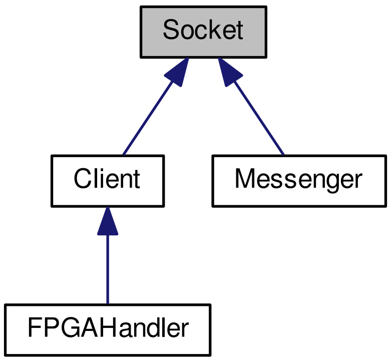
- include/Messenger.h
- src/Messenger.cpp

5.9 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 }

  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

5.9.1 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

5.9.2 Member Typedef Documentation

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

5.9.3 Constructor & Destructor Documentation

5.9.3.1 Socket::Socket ( ) [inline]

5.9.3.2 Socket::Socket (int *port*)

5.9.3.3 Socket::∼Socket ( ) [virtual]

5.9.4 Member Function Documentation

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

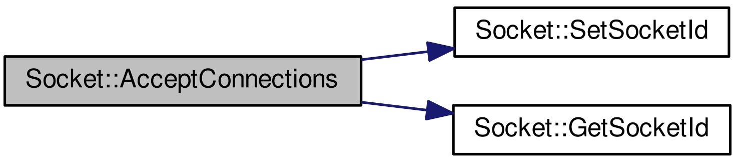
Accept connection from a client.

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:



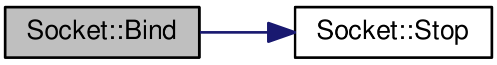
5.9.4.2 void Socket::Bind ( ) [protected]

Bind a name to a socket.

Returns

Success of the operation

Here is the call graph for this function:



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

Configure the socket object for communication.

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

Create an endpoint for communication.

5.9.4.5 void Socket::DumpConnected ( ) const

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

Receive a message from a socket.

Returns

Received message as a std::string

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

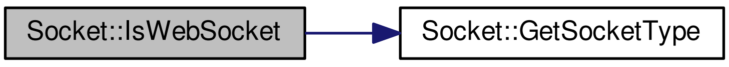
Retrieve the port used for this socket.

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

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

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

Here is the call graph for this function:

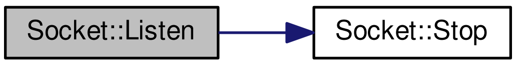


5.9.4.11 void Socket::Listen (int *maxconn*) `[protected]`

Listen to incoming messages.

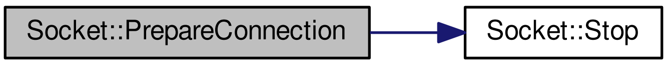
Set the socket to listen to any message coming from outside

Here is the call graph for this function:



5.9.4.12 void Socket::PrepareConnection ( ) [protected]

Here is the call graph for this function:



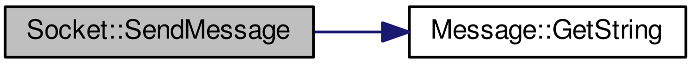
5.9.4.13 void Socket::SelectConnections ( )

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

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

Send a message on a socket.

Here is the call graph for this function:



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

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

5.9.4.17 bool Socket::Start ( ) [protected]

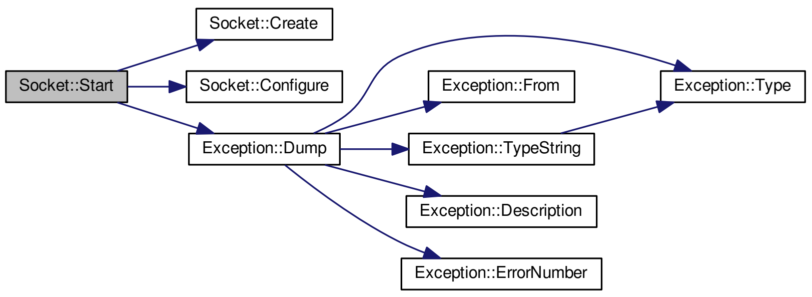
Start the socket.

Launch all mandatory operations to set the socket to be used

Returns

Success of the operation

Here is the call graph for this function:



5.9.4.18 void Socket::Stop ( )

Terminates the socket and all attached communications.

5.9.5 Field Documentation

5.9.5.1 struct sockaddr_in Socket::fAddress [private]

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

5.9.5.3 fd_set Socket::fMaster [protected]

Master file descriptor list.

5.9.5.4 int Socket::fPort [protected]

5.9.5.5 fd_set Socket::fReadFds [protected]

Temp file descriptor list for select()

5.9.5.6 int Socket::fSocketId [private]

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

5.9.5.7 SocketCollection Socket::fSocketsConnected [protected]

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

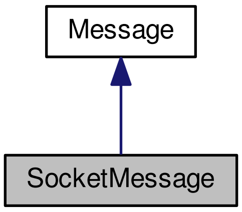
- include/Socket.h
- src/Socket.cpp

5.10 SocketMessage Class Reference

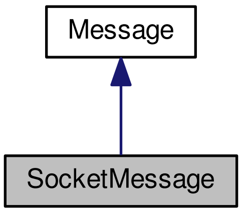
Socket-passed message type.

`#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 (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 int value)

    Construct a socket message out of a key and an 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, int int_value)

    Send an 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.
- 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

5.10.1 Detailed Description

Socket-passed message type.

Author

Laurent Forthomme `laurent.forthomme@cern.ch`

Date

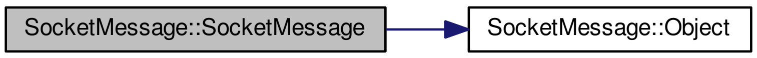
26 Mar 2015

5.10.2 Constructor & Destructor Documentation

5.10.2.1 SocketMessage::SocketMessage ( ) [inline]

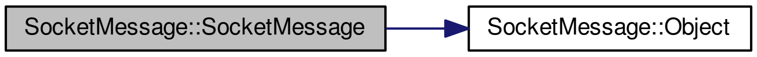
5.10.2.2 SocketMessage::SocketMessage (const Message & *msg*) [inline]

Here is the call graph for this function:



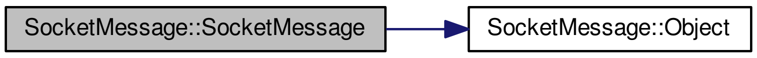
5.10.2.3 SocketMessage::SocketMessage (const char ∗ *msg_s*) [inline]

Here is the call graph for this function:



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

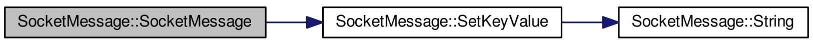
Here is the call graph for this function:



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

Construct a socket message out of a key.

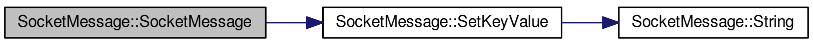
Here is the call graph for this function:



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

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

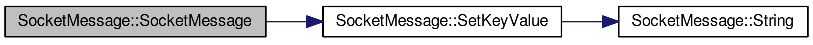
Here is the call graph for this function:



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

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

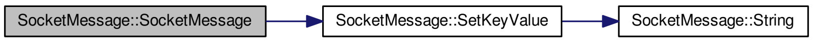
Here is the call graph for this function:



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

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

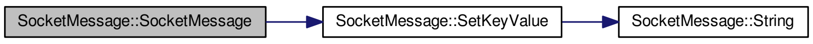
Here is the call graph for this function:



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

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

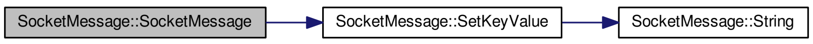
Here is the call graph for this function:



5.10.2.10 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:



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

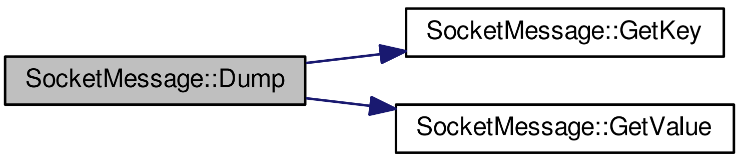
Construct a socket message out of a map of key/string-type value.

5.10.2.12 SocketMessage::∼SocketMessage ( ) [inline]

5.10.3 Member Function Documentation

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

Here is the call graph for this function:



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

Extract the message's integer value.

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

Extract the message's key.

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

Extract the whole key:value message.

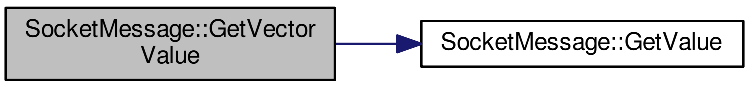
5.10.3.5 std::string SocketMessage::GetValue ( ) const [inline]

Extract the message's string value.

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

Extract the message's vector of string value.

Here is the call graph for this function:

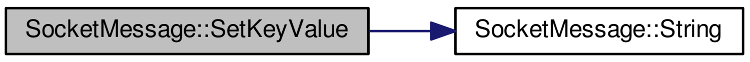


5.10.3.7 MessageMap SocketMessage::Object ( ) const [inline], [private]

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

String-valued message.

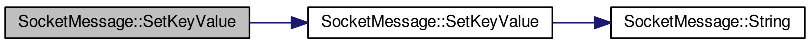
Here is the call graph for this function:



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

Send an integer-valued message.

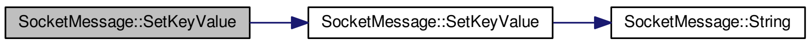
Here is the call graph for this function:



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

Float-valued message.

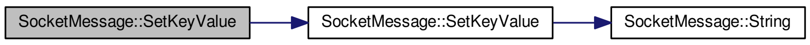
Here is the call graph for this function:



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

Double-valued message.

Here is the call graph for this function:



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

5.10.4 Field Documentation

5.10.4.1 MessageMap SocketMessage::fMessage `[private]`

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

- include/SocketMessage.h

5.11 TDCConfiguration Class Reference

Setup word to be sent to the HPTDC chip.

`#include <TDCConfiguration.h>`

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, ReadoutState↩
  Error =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 }
- typedef uint16_t bit

  LSB index.
- typedef uint32_t word_t

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

Public Member Functions

- TDCConfiguration ()
- TDCConfiguration (const TDCConfiguration &c)
- virtual ∼TDCConfiguration ()
- void SetWord (const unsigned int i, const word_t word)

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

  Retrieve one subset from the setup word.
- uint8_t GetNumWords () const

  Number of words in the configuration.
- 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 $\sim$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 Dump (int verb=1, std::ostream &os=std::cout) const

Private Member Functions

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

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

  Extract bits from the configuration word.
- 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 ∼ 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 ∼ 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.

Private Attributes

- word_t fWord [kNumWords]

Static Private Attributes

- static const uint8_t kNumWords = BITS_NUM/WORD_SIZE+1
- 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

5.11.1 Detailed Description

Setup word to be sent to the HPTDC chip.

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

Author

Laurent Forthomme `laurent.forthomme@cern.ch`

Date

16 Apr 2015

5.11.2 Member Typedef Documentation

5.11.2.1 typedef uint16_t TDCConfiguration::bit

LSB index.

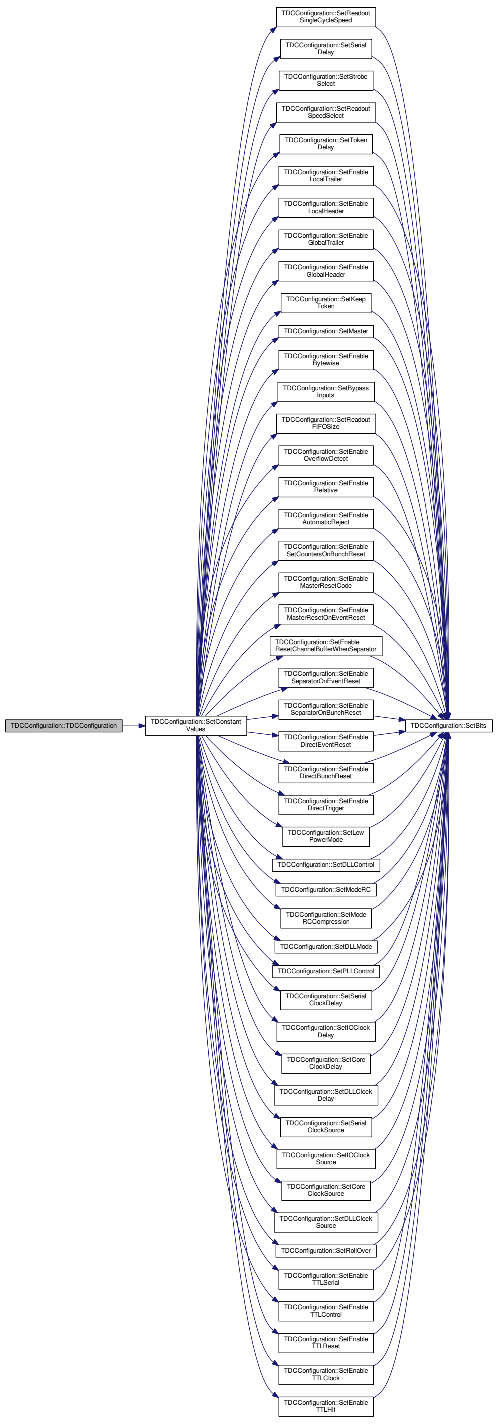
5.11.2.2 typedef uint32_t TDCConfiguration::word_t

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

5.11.3 Constructor & Destructor Documentation

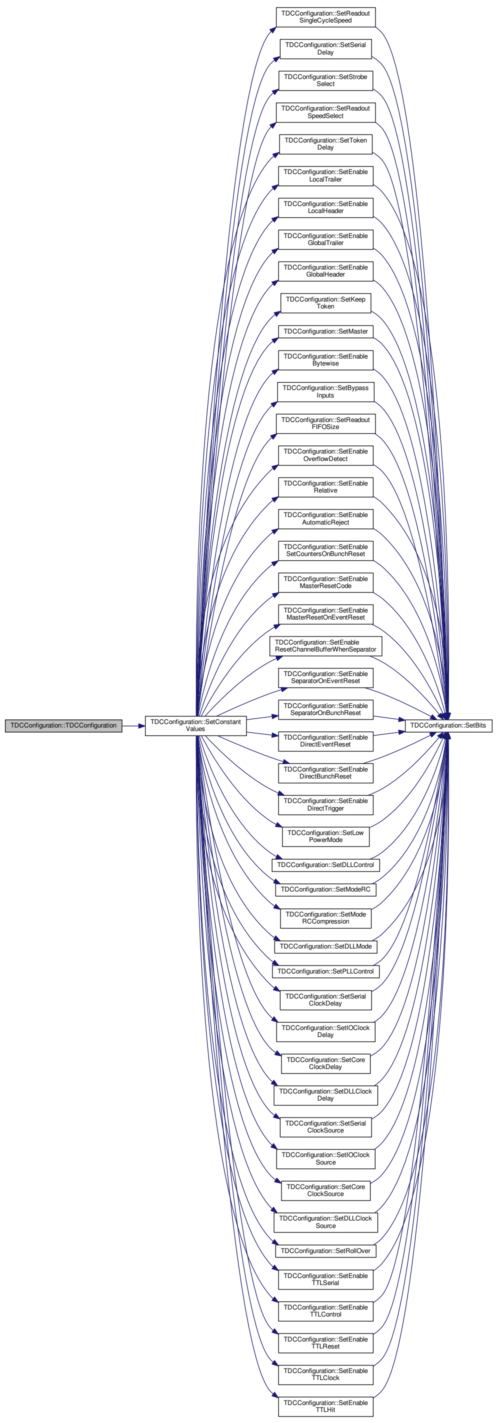
5.11.3.1 TDCConfiguration::TDCConfiguration ( )

Here is the call graph for this function:


TDCConfiguration::TDCConfiguration
TDCConfiguration::SetConstant Values
TDCConfiguration::SetReadout SingleCycleSpeed
TDCConfiguration::SetSerial Delay
TDCConfiguration::SetStrobe Select
TDCConfiguration::SetReadout SpeedSelect
TDCConfiguration::SetToken Delay
TDCConfiguration::SetEnable LocalTrailer
TDCConfiguration::SetEnable LocalHeader
TDCConfiguration::SetEnable GlobalTrailer
TDCConfiguration::SetEnable GlobalHeader
TDCConfiguration::SetKeep Token
TDCConfiguration::SetMaster
TDCConfiguration::SetEnable Bytewise
TDCConfiguration::SetBypass Inputs
TDCConfiguration::SetReadout FIFOSize
TDCConfiguration::SetEnable OverflowDetect
TDCConfiguration::SetEnable Relative
TDCConfiguration::SetEnable AutomaticReject
TDCConfiguration::SetEnable SetCountersOnBunchReset
TDCConfiguration::SetEnable MasterResetCode
TDCConfiguration::SetEnable MasterResetOnEventReset
TDCConfiguration::SetEnable ResetChannelBufferWhenSeparator
TDCConfiguration::SetEnable SeparatorOnEventReset
TDCConfiguration::SetEnable SeparatorOnBunchReset
TDCConfiguration::SetEnable DirectEventReset
TDCConfiguration::SetEnable DirectBunchReset
TDCConfiguration::SetEnable DirectTrigger
TDCConfiguration::SetLow PowerMode
TDCConfiguration::SetDLLControl
TDCConfiguration::SetModeRC
TDCConfiguration::SetMode RCCompression
TDCConfiguration::SetDLLMode
TDCConfiguration::SetPLLControl
TDCConfiguration::SetSerial ClockDelay
TDCConfiguration::SetIOClock Delay
TDCConfiguration::SetCore ClockDelay
TDCConfiguration::SetDLLClock Delay
TDCConfiguration::SetSerial ClockSource
TDCConfiguration::SetIOClock Source
TDCConfiguration::SetCore ClockSource
TDCConfiguration::SetDLLClock Source
TDCConfiguration::SetRollOver
TDCConfiguration::SetEnable TTLSerial
TDCConfiguration::SetEnable TTLControl
TDCConfiguration::SetEnable TTLReset
TDCConfiguration::SetEnable TTLClock
TDCConfiguration::SetEnable TTLHit
TDCConfiguration::SetBits

5.11.3.2 TDCConfiguration::TDCConfiguration (const **TDCConfiguration** & *c*)

Here is the call graph for this function:

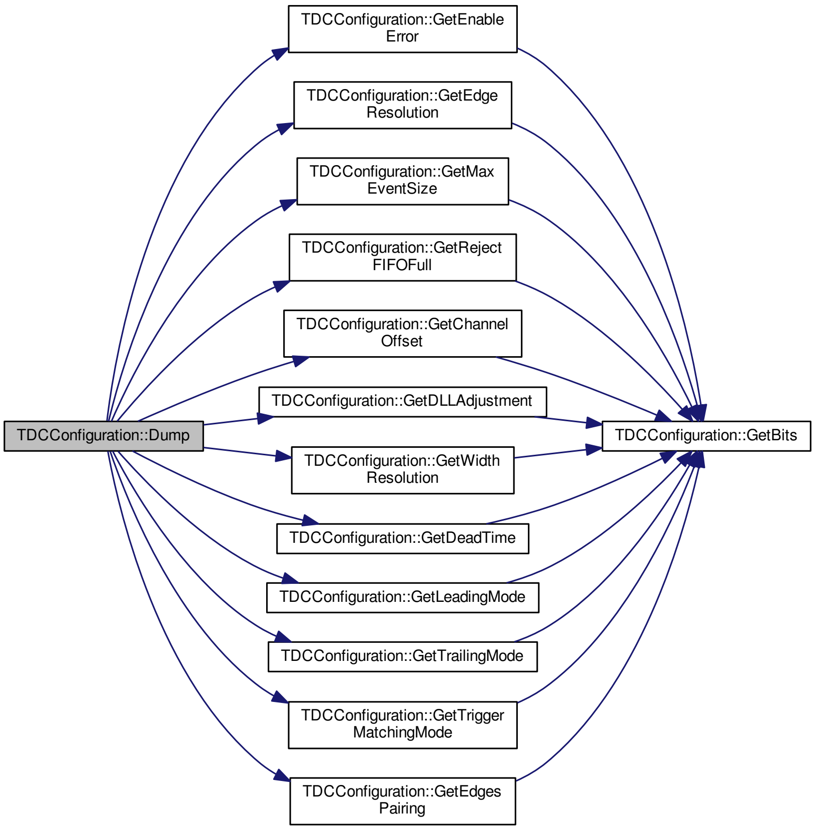

TDCConfiguration::TDCConfiguration
TDCConfiguration::SetConstant Values
TDCConfiguration::SetReadout SingleCycleSpeed
TDCConfiguration::SetSerial Delay
TDCConfiguration::SetStrobe Select
TDCConfiguration::SetReadout SpeedSelect
TDCConfiguration::SetToken Delay
TDCConfiguration::SetEnable LocalTrailer
TDCConfiguration::SetEnable LocalHeader
TDCConfiguration::SetEnable GlobalTrailer
TDCConfiguration::SetEnable GlobalHeader
TDCConfiguration::SetKeep Token
TDCConfiguration::SetMaster
TDCConfiguration::SetEnable Bytewise
TDCConfiguration::SetBypass Inputs
TDCConfiguration::SetReadout FIFOSize
TDCConfiguration::SetEnable OverflowDetect
TDCConfiguration::SetEnable Relative
TDCConfiguration::SetEnable AutomaticReject
TDCConfiguration::SetEnable SetCountersOnBunchReset
TDCConfiguration::SetEnable MasterResetCode
TDCConfiguration::SetEnable MasterResetOnEventReset
TDCConfiguration::SetEnable ResetChannelBufferWhenSeparator
TDCConfiguration::SetEnable SeparatorOnEventReset
TDCConfiguration::SetEnable SeparatorOnBunchReset
TDCConfiguration::SetEnable DirectEventReset
TDCConfiguration::SetEnable DirectBunchReset
TDCConfiguration::SetEnable DirectTrigger
TDCConfiguration::SetLow PowerMode
TDCConfiguration::SetDLLControl
TDCConfiguration::SetModeRC
TDCConfiguration::SetMode RCCompression
TDCConfiguration::SetDLLMode
TDCConfiguration::SetPLLControl
TDCConfiguration::SetSerial ClockDelay
TDCConfiguration::SetIOClock Delay
TDCConfiguration::SetCore ClockDelay
TDCConfiguration::SetDLLClock Delay
TDCConfiguration::SetSerial ClockSource
TDCConfiguration::SetIOClock Source
TDCConfiguration::SetCore ClockSource
TDCConfiguration::SetDLLClock Source
TDCConfiguration::SetRollOver
TDCConfiguration::SetEnable TTLSerial
TDCConfiguration::SetEnable TTLControl
TDCConfiguration::SetEnable TTLReset
TDCConfiguration::SetEnable TTLClock
TDCConfiguration::SetEnable TTLHit
TDCConfiguration::SetBits

5.11.3.3 virtual TDCConfiguration::∼TDCConfiguration ( ) `[inline]`, `[virtual]`

5.11.4 Member Function Documentation

5.11.4.1 void TDCConfiguration::Dump (int *verb* = `1`, std::ostream & *os* = `std::cout`) const

Here is the call graph for this function:


TDCConfiguration::GetEnable Error
TDCConfiguration::GetEdge Resolution
TDCConfiguration::GetMax EventSize
TDCConfiguration::GetReject FIFOFull
TDCConfiguration::GetChannel Offset
TDCConfiguration::GetDLLAdjustment
TDCConfiguration::Dump
TDCConfiguration::GetBits
TDCConfiguration::GetWidth Resolution
TDCConfiguration::GetDeadTime
TDCConfiguration::GetLeadingMode
TDCConfiguration::GetTrailingMode
TDCConfiguration::GetTrigger MatchingMode
TDCConfiguration::GetEdges Pairing

5.11.4.2 uint16_t TDCConfiguration::GetBits (uint16_t *lsb*, uint8_t *size*) const `[private]`

Extract bits from the configuration word.

Extract a fixed amount of bits from the full configuration word

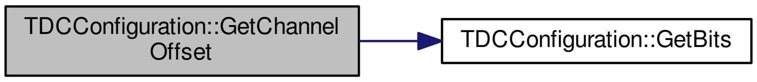
Parameters

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

5.11.4.3 uint16_t TDCConfiguration::GetChannelOffset (int *channel*) const `[inline]`

Return the offset for one single channel.

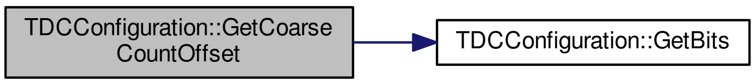
Here is the call graph for this function:


TDCConfiguration::GetChannel Offset
TDCConfiguration::GetBits

5.11.4.4 uint16_t TDCConfiguration::GetCoarseCountOffset ( ) const [inline]

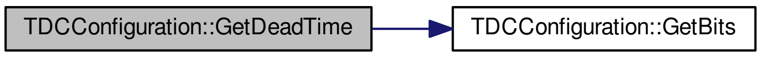
Extract offset for the coarse time counter.

Here is the call graph for this function:


TDCConfiguration::GetCoarse CountOffset
TDCConfiguration::GetBits

5.11.4.5 DeadTime TDCConfiguration::GetDeadTime ( ) const [inline]

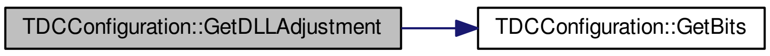
Here is the call graph for this function:



5.11.4.6 uint8_t TDCConfiguration::GetDLLAdjustment (int *tap*) const [inline]

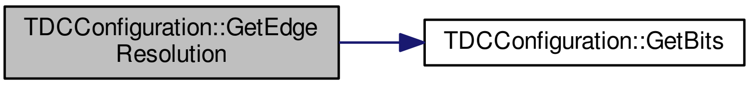
Set the adjustment of DLL taps.

Here is the call graph for this function:



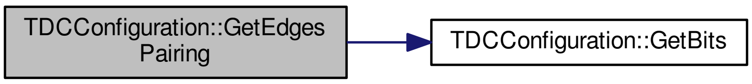
5.11.4.7 EdgeResolution TDCConfiguration::GetEdgeResolution ( ) const [inline]

Here is the call graph for this function:


TDCConfiguration::GetEdge Resolution
TDCConfiguration::GetBits

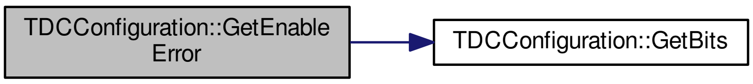
5.11.4.8 bool TDCConfiguration::GetEdgesPairing ( ) const [inline]

Here is the call graph for this function:



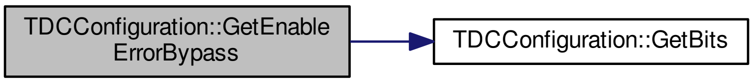
5.11.4.9 uint16_t TDCConfiguration::GetEnableError ( ) const [inline]

Here is the call graph for this function:



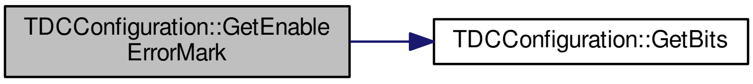
5.11.4.10 bool TDCConfiguration::GetEnableErrorBypass ( ) const [inline]

Here is the call graph for this function:


TDCConfiguration::GetEnable ErrorBypass
TDCConfiguration::GetBits

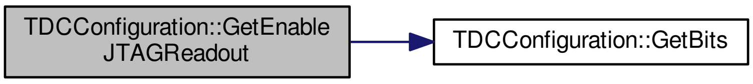
5.11.4.11 bool TDCConfiguration::GetEnableErrorMark ( ) const [inline]

Here is the call graph for this function:


TDCConfiguration::GetEnable ErrorMark
TDCConfiguration::GetBits

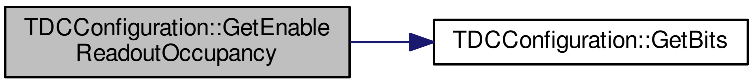
5.11.4.12 bool TDCConfiguration::GetEnableJTAGReadout ( ) const [inline]

Here is the call graph for this function:


TDCConfiguration::GetEnable JTAGReadout
TDCConfiguration::GetBits

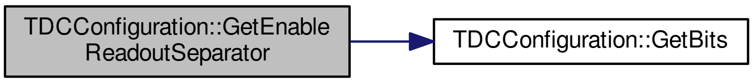
5.11.4.13 bool TDCConfiguration::GetEnableReadoutOccupancy ( ) const [inline]

Here is the call graph for this function:


TDCConfiguration::GetEnable ReadoutOccupancy
TDCConfiguration::GetBits

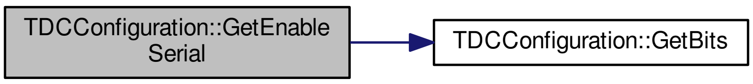
5.11.4.14 bool TDCConfiguration::GetEnableReadoutSeparator ( ) const [inline]

Here is the call graph for this function:


TDCConfiguration::GetEnable ReadoutSeparator
TDCConfiguration::GetBits

5.11.4.15 bool TDCConfiguration::GetEnableSerial ( ) const [inline]

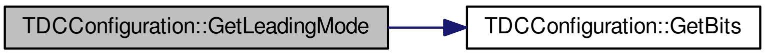
Here is the call graph for this function:



5.11.4.16 bool TDCConfiguration::GetLeadingMode ( ) const [inline]

Extract the status for the detection of leading edges.

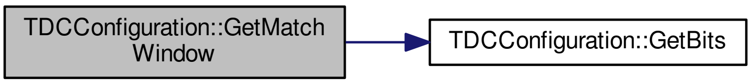
Here is the call graph for this function:



5.11.4.17 uint16_t TDCConfiguration::GetMatchWindow ( ) const [inline]

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

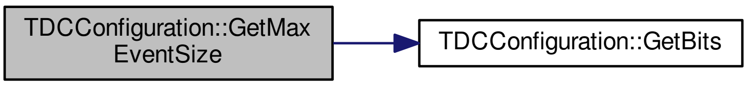
Here is the call graph for this function:



5.11.4.18 uint8_t TDCConfiguration::GetMaxEventSize ( ) const [inline]

Extract the maximum number of hits per event.

Here is the call graph for this function:



5.11.4.19 uint8_t TDCConfiguration::GetNumWords ( ) const [inline]

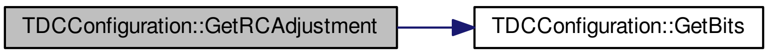
Number of words in the configuration.

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

5.11.4.20 uint8_t TDCConfiguration::GetRCAdjustment (int *tap*) [inline]

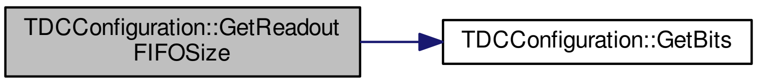
Extract the adjustment of the RC delay line.

Here is the call graph for this function:



5.11.4.21 int TDCConfiguration::GetReadoutFIFOSize ( ) const [inline]

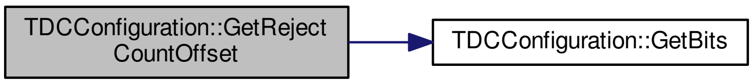
Here is the call graph for this function:


TDCConfiguration::GetReadout FIFOSize
TDCConfiguration::GetBits

5.11.4.22 uint16_t TDCConfiguration::GetRejectCountOffset ( ) const [inline]

Extract the offset in reject counter.

Here is the call graph for this function:

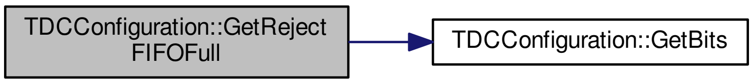

TDCConfiguration::GetReject CountOffset
TDCConfiguration::GetBits

5.11.4.23 bool TDCConfiguration::GetRejectFIFOFull ( ) const [inline]

Are hits rejected when readout FIFO is full?

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

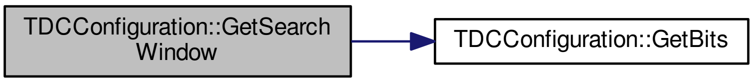
Here is the call graph for this function:


TDCConfiguration::GetReject FIFOFull
TDCConfiguration::GetBits

5.11.4.24 uint16_t TDCConfiguration::GetSearchWindow ( ) const [inline]

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

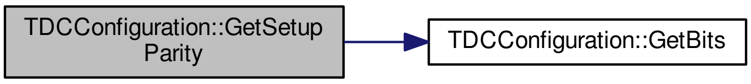
Here is the call graph for this function:



5.11.4.25 bool TDCConfiguration::GetSetupParity ( ) const [inline]

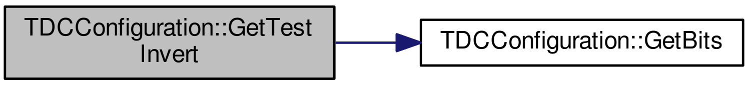
Extract the parity of setup data (should be an even parity)

Here is the call graph for this function:



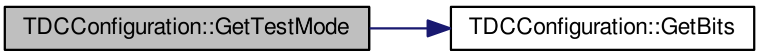
5.11.4.26 bool TDCConfiguration::GetTestInvert ( ) const `[inline]`

Here is the call graph for this function:



5.11.4.27 bool TDCConfiguration::GetTestMode ( ) const `[inline]`

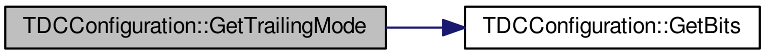
Here is the call graph for this function:



5.11.4.28 bool TDCConfiguration::GetTrailingMode ( ) const `[inline]`

Extract the status for the detection of trailing edges.

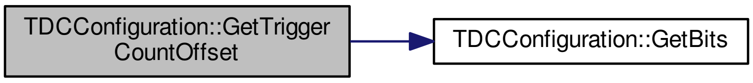
Here is the call graph for this function:



5.11.4.29 uint16_t TDCConfiguration::GetTriggerCountOffset ( ) const `[inline]`

Extract trigger time tag count offset.

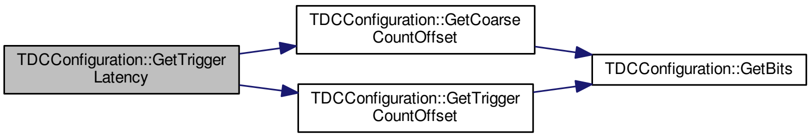
Here is the call graph for this function:


TDCConfiguration::GetTrigger CountOffset
TDCConfiguration::GetBits

5.11.4.30 uint16_t TDCConfiguration::GetTriggerLatency ( ) const [inline]

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

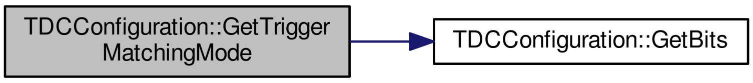
Here is the call graph for this function:


TDCConfiguration::GetTrigger Latency
TDCConfiguration::GetCoarse CountOffset
TDCConfiguration::GetTrigger CountOffset
TDCConfiguration::GetBits

5.11.4.31 bool TDCConfiguration::GetTriggerMatchingMode ( ) const [inline]

Extract the enable status of trigger matching mode.

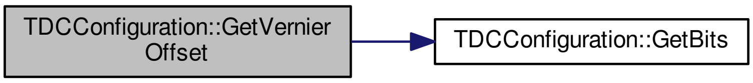
Here is the call graph for this function:


TDCConfiguration::GetTrigger MatchingMode
TDCConfiguration::GetBits

5.11.4.32 uint8_t TDCConfiguration::GetVernierOffset ( ) const [inline]

Extract the offset in vernier decoding.

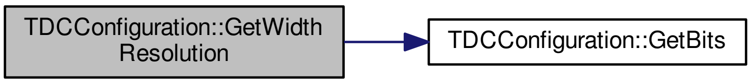
Here is the call graph for this function:


TDCConfiguration::GetVernier Offset
TDCConfiguration::GetBits

5.11.4.33 WidthResolution TDCConfiguration::GetWidthResolution ( ) const [inline]

Extract the pulse width resolution when paired measurements are performed.

Here is the call graph for this function:


TDCConfiguration::GetWidth Resolution
TDCConfiguration::GetBits

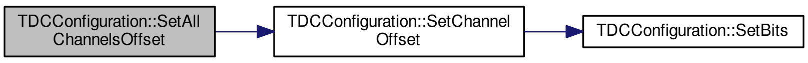
5.11.4.34 word_t TDCConfiguration::GetWord (const unsigned int *i*) const [inline]

Retrieve one subset from the setup word.

5.11.4.35 void TDCConfiguration::SetAllChannelsOffset (uint16_t *offset*) [inline]

Set the time offset for all channels.

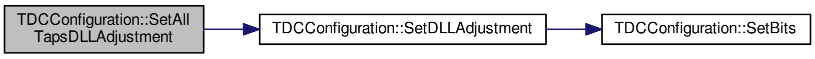
Here is the call graph for this function:


TDCConfiguration::SetAll ChannelsOffset
TDCConfiguration::SetChannel Offset
TDCConfiguration::SetBits

5.11.4.36 void TDCConfiguration::SetAllTapsDLLAdjustment (uint8_t *adj*) [inline]

Extract the adjustment of DLL taps.

Here is the call graph for this function:


TDCConfiguration::SetAll TapsDLLAdjustment
TDCConfiguration::SetDLLAdjustment
TDCConfiguration::SetBits

5.11.4.37 void TDCConfiguration::SetBits (uint16_t *lsb*, uint16_t *word*, uint8_t *size*) `[private]`

Set bits in the configuration word.

Set a fixed amount of bits in the full configuration word

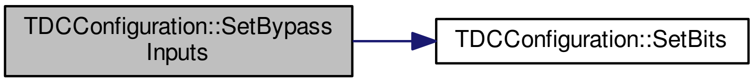
Parameters

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

5.11.4.38 void TDCConfiguration::SetBypassInputs (const bool *sbi* = `true`) `[inline]`, `[private]`

Select serial in and token in from bypass inputs.

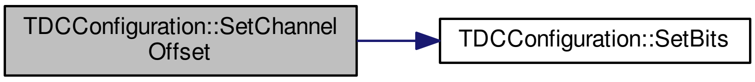
Here is the call graph for this function:



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

Set the time offset for one single channel.

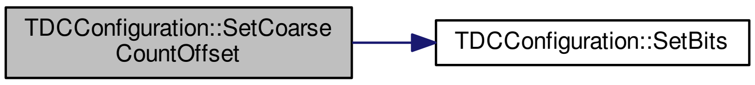
Here is the call graph for this function:


TDCConfiguration::SetChannel Offset
TDCConfiguration::SetBits

5.11.4.40 void TDCConfiguration::SetCoarseCountOffset (uint16_t *cco*) [inline]

Set offset for the coarse time counter.

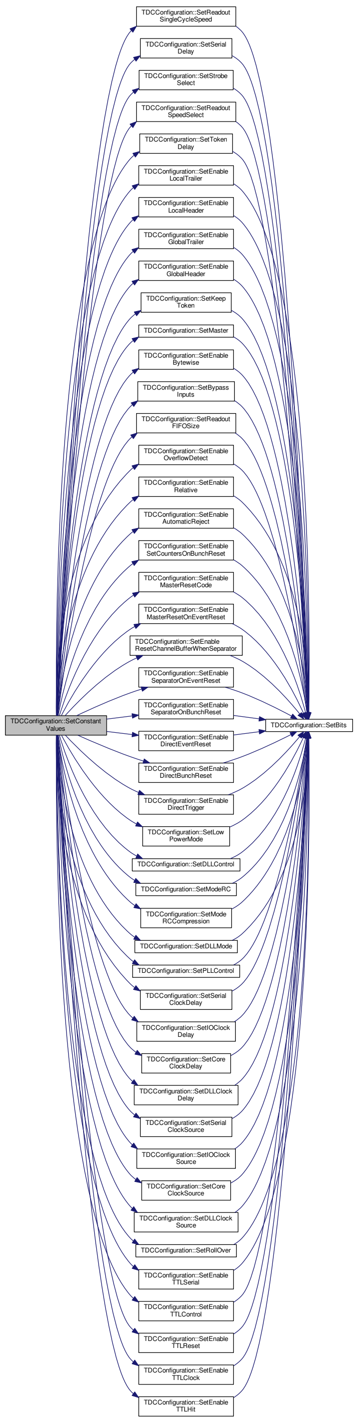
Here is the call graph for this function:


TDCConfiguration::SetCoarse CountOffset
TDCConfiguration::SetBits

5.11.4.41 void TDCConfiguration::SetConstantValues ( )

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

Here is the call graph for this function:


TDCConfiguration::SetReadout SingleCycleSpeed
TDCConfiguration::SetSerial Delay
TDCConfiguration::SetStrobe Select
TDCConfiguration::SetReadout SpeedSelect
TDCConfiguration::SetToken Delay
TDCConfiguration::SetEnable LocalTrailer
TDCConfiguration::SetEnable LocalHeader
TDCConfiguration::SetEnable GlobalTrailer
TDCConfiguration::SetEnable GlobalHeader
TDCConfiguration::SetKeep Token
TDCConfiguration::SetMaster
TDCConfiguration::SetEnable Bytewise
TDCConfiguration::SetBypass Inputs
TDCConfiguration::SetReadout FIFOSize
TDCConfiguration::SetEnable OverflowDetect
TDCConfiguration::SetEnable Relative
TDCConfiguration::SetEnable AutomaticReject
TDCConfiguration::SetEnable SetCountersOnBunchReset
TDCConfiguration::SetEnable MasterResetCode
TDCConfiguration::SetEnable MasterResetOnEventReset
TDCConfiguration::SetEnable ResetChannelBufferWhenSeparator
TDCConfiguration::SetEnable SeparatorOnEventReset
TDCConfiguration::SetEnable SeparatorOnBunchReset
TDCConfiguration::SetConstant Values
TDCConfiguration::SetBits
TDCConfiguration::SetEnable DirectEventReset
TDCConfiguration::SetEnable DirectBunchReset
TDCConfiguration::SetEnable DirectTrigger
TDCConfiguration::SetLow PowerMode
TDCConfiguration::SetDLLControl
TDCConfiguration::SetModeRC
TDCConfiguration::SetMode RCCompression
TDCConfiguration::SetDLLMode
TDCConfiguration::SetPLLControl
TDCConfiguration::SetSerial ClockDelay
TDCConfiguration::SetIOClock Delay
TDCConfiguration::SetCore ClockDelay
TDCConfiguration::SetDLLClock Delay
TDCConfiguration::SetSerial ClockSource
TDCConfiguration::SetIOClock Source
TDCConfiguration::SetCore ClockSource
TDCConfiguration::SetDLLClock Source
TDCConfiguration::SetRollOver
TDCConfiguration::SetEnable TTLSerial
TDCConfiguration::SetEnable TTLControl
TDCConfiguration::SetEnable TTLReset
TDCConfiguration::SetEnable TTLClock
TDCConfiguration::SetEnable TTLHit

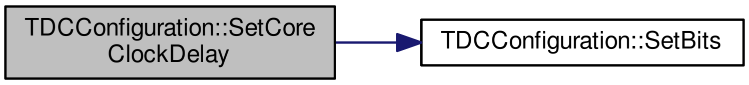
5.11.4.42 void TDCConfiguration::SetCoreClockDelay (const bool *delay_clock*, const uint8_t *delay*) [inline], [private]

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

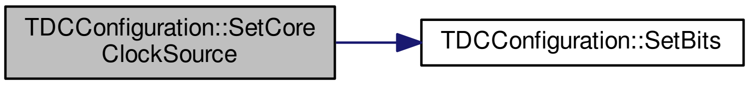
Here is the call graph for this function:


TDCConfiguration::SetCore ClockDelay
TDCConfiguration::SetBits

5.11.4.43 void TDCConfiguration::SetCoreClockSource (const CoreClockSource *ccs*) `[inline]`, `[private]`

Selection of clock source for internal logic.

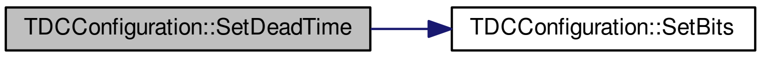
Here is the call graph for this function:


TDCConfiguration::SetCore ClockSource
TDCConfiguration::SetBits

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

Channel dead time between hits.

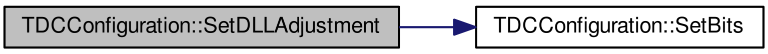
Here is the call graph for this function:



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

Set the DLL taps adjustments with a resolution of $\sim$10 ps.

Here is the call graph for this function:



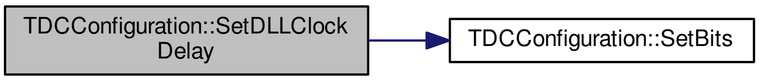
5.11.4.46 void TDCConfiguration::SetDLLClockDelay (const bool *delay_clock*, const uint8_t *delay*) [inline], [private]

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

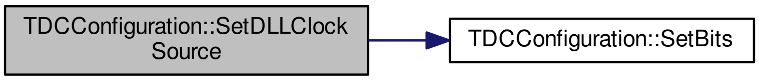
Here is the call graph for this function:


TDCConfiguration::SetDLLClock Delay
TDCConfiguration::SetBits

5.11.4.47 void TDCConfiguration::SetDLLClockSource (const DLLClockSource *dcs*) [inline], [private]

Selection of clock source for DLL.

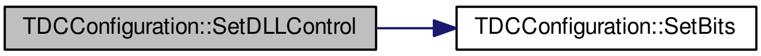
Here is the call graph for this function:


TDCConfiguration::SetDLLClock Source
TDCConfiguration::SetBits

5.11.4.48 void TDCConfiguration::SetDLLControl (const uint8_t *dc*) [inline], [private]

Control of DLL (DLL charge pump levels)

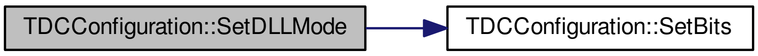
Here is the call graph for this function:



5.11.4.49 void TDCConfiguration::SetDLLMode (const DLLSpeedMode *dsm*) [inline], [private]

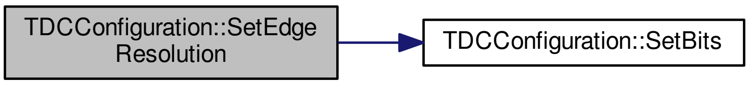
Selection of DLL speed mode.

Here is the call graph for this function:



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

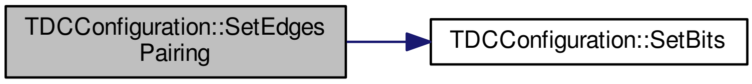
Here is the call graph for this function:


TDCConfiguration::SetEdge Resolution
TDCConfiguration::SetBits

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

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

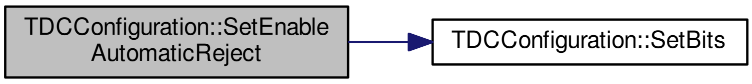
Here is the call graph for this function:



5.11.4.52 void TDCConfiguration::SetEnableAutomaticReject (const bool *ear* = `true`) `[inline]`, `[private]`

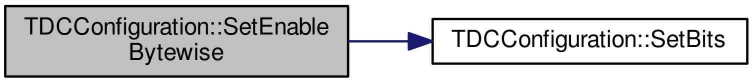
Enable of automatic rejection (should always be enabled if trigger matching mode!)

Here is the call graph for this function:


TDCConfiguration::SetEnable AutomaticReject
TDCConfiguration::SetBits

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

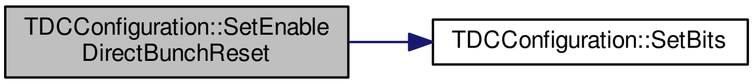
Here is the call graph for this function:


TDCConfiguration::SetEnable Bytewise
TDCConfiguration::SetBits

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

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

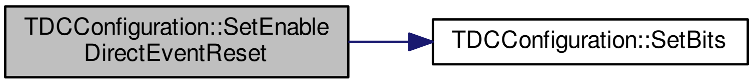
Here is the call graph for this function:



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

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

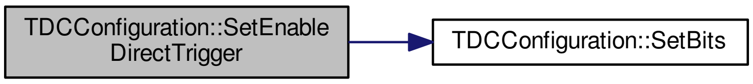
Here is the call graph for this function:



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

Enable of direct trigger input pin.

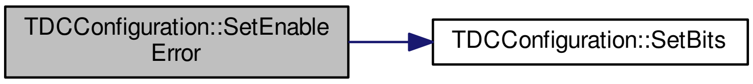
Here is the call graph for this function:



5.11.4.57 void TDCConfiguration::SetEnableError (const uint16_t & *err*) `[inline]`

Enable internal error types for generation of global error signals.

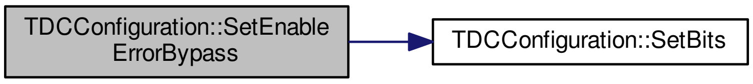
Here is the call graph for this function:



5.11.4.58 void TDCConfiguration::SetEnableErrorBypass (const bool *eb*) [inline]

Bypass TDC chip if global error signal is set.

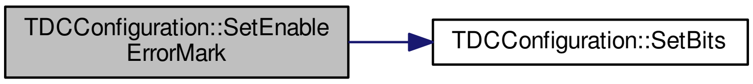
Here is the call graph for this function:



5.11.4.59 void TDCConfiguration::SetEnableErrorMark (const bool *em*) [inline]

Mark events with error if global error signal is set.

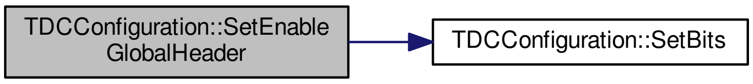
Here is the call graph for this function:



5.11.4.60 void TDCConfiguration::SetEnableGlobalHeader (const bool *egh* = true) [inline], [private]

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

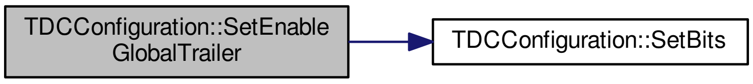
Here is the call graph for this function:


TDCConfiguration::SetEnable GlobalHeader
TDCConfiguration::SetBits

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

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

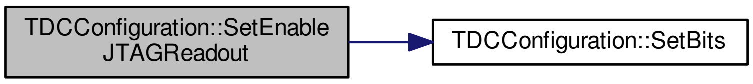
Here is the call graph for this function:


TDCConfiguration::SetEnable GlobalTrailer
TDCConfiguration::SetBits

5.11.4.62 void TDCConfiguration::SetEnableJTAGReadout (const bool *jr*) `[inline]`

Enable of read-out via JTAG.

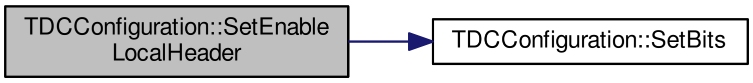
Here is the call graph for this function:


TDCConfiguration::SetEnable JTAGReadout
TDCConfiguration::SetBits

5.11.4.63 void TDCConfiguration::SetEnableLocalHeader (const bool *elh* = `true`) `[inline]`, `[private]`

Enable of local headers in read-out.

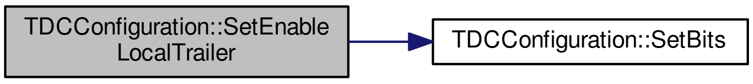
Here is the call graph for this function:


TDCConfiguration::SetEnable LocalHeader
TDCConfiguration::SetBits

5.11.4.64 void TDCConfiguration::SetEnableLocalTrailer (const bool *elt* = `true`) `[inline]`, `[private]`

Enable of local trailers in read-out.

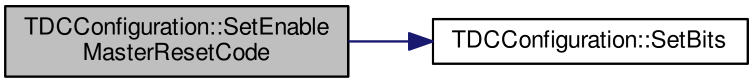
Here is the call graph for this function:


TDCConfiguration::SetEnable LocalTrailer
TDCConfiguration::SetBits

5.11.4.65 void TDCConfiguration::SetEnableMasterResetCode (const bool *emrc* = `true`) `[inline]`, `[private]`

Enable master reset code on encoded_control.

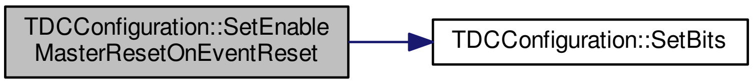
Here is the call graph for this function:


TDCConfiguration::SetEnable MasterResetCode
TDCConfiguration::SetBits

5.11.4.66 void TDCConfiguration::SetEnableMasterResetOnEventReset (const bool *emroer* = `true`) `[inline]`, `[private]`

Enable master reset of whole TDC on event reset.

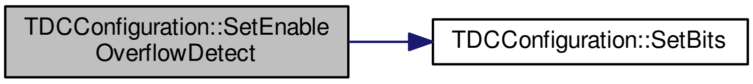
Here is the call graph for this function:


TDCConfiguration::SetEnable MasterResetOnEventReset
TDCConfiguration::SetBits

5.11.4.67 void TDCConfiguration::SetEnableOverflowDetect (const bool *eod* = `true`) `[inline]`, `[private]`

Enable overflow detection of L1 buffers (should always be enabled!)

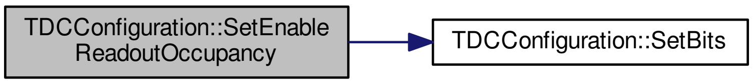
Here is the call graph for this function:


TDCConfiguration::SetEnable OverflowDetect
TDCConfiguration::SetBits

5.11.4.68 void TDCConfiguration::SetEnableReadoutOccupancy (const bool *ro* = `true`) `[inline]`

Enable the readout of buffer occupancies for each event (for debugging purposes)

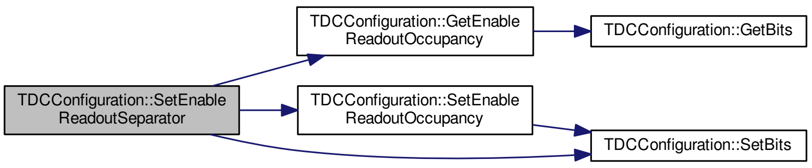
Here is the call graph for this function:


TDCConfiguration::SetEnable ReadoutOccupancy
TDCConfiguration::SetBits

5.11.4.69 void TDCConfiguration::SetEnableReadoutSeparator (const bool *ro* = `true`) `[inline]`

Enable the readout of separators for each event (for debugging purposes, valid if readout of occupancies is enabled)

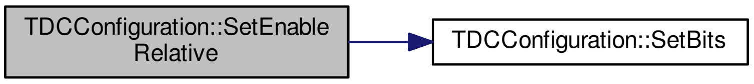
Here is the call graph for this function:


TDCConfiguration::SetEnable ReadoutSeparator
TDCConfiguration::GetEnable ReadoutOccupancy
TDCConfiguration::SetEnable ReadoutOccupancy
TDCConfiguration::GetBits
TDCConfiguration::SetBits

5.11.4.70 void TDCConfiguration::SetEnableRelative (const bool *er* = `true`) `[inline]`, `[private]`

Enable read-out of relative time to trigger time tag. Only valid when using trigger matching mode.

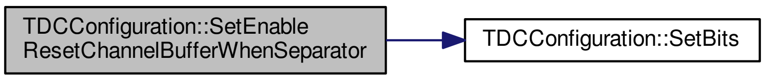
Here is the call graph for this function:


TDCConfiguration::SetEnable Relative
TDCConfiguration::SetBits

5.11.4.71 void TDCConfiguration::SetEnableResetChannelBufferWhenSeparator (const bool *ercbws* = `true`) `[inline]`, `[private]`

Enable reset channel buffers when separator.

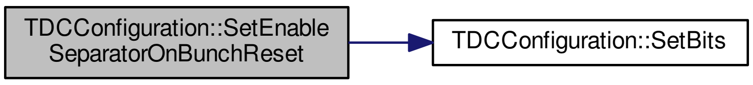
Here is the call graph for this function:


TDCConfiguration::SetEnable ResetChannelBufferWhenSeparator
TDCConfiguration::SetBits

5.11.4.72 void TDCConfiguration::SetEnableSeparatorOnBunchReset (const bool *esobr* = `true`) `[inline]`, `[private]`

Enable generation of separator on bunch reset.

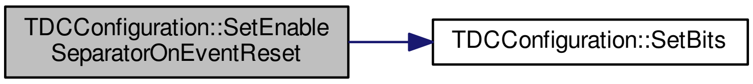
Here is the call graph for this function:



5.11.4.73 void TDCConfiguration::SetEnableSeparatorOnEventReset (const bool *esoer* = `true`) `[inline]`, `[private]`

Enable generation of separator on event reset.

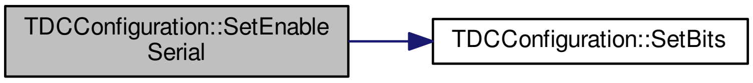
Here is the call graph for this function:



5.11.4.74 void TDCConfiguration::SetEnableSerial (const bool *es*) `[inline]`

Enable of serial read-out (otherwise parallel read-out)

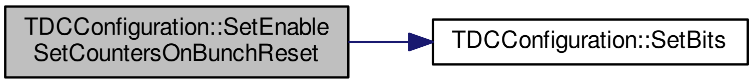
Here is the call graph for this function:



5.11.4.75 void TDCConfiguration::SetEnableSetCountersOnBunchReset (const bool *escobr* = `true`) `[inline]`, `[private]`

Enable all counters to be set on bunch count reset.

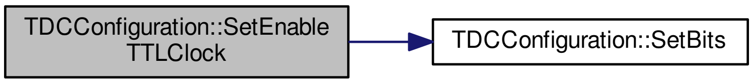
Here is the call graph for this function:



5.11.4.76 void TDCConfiguration::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:



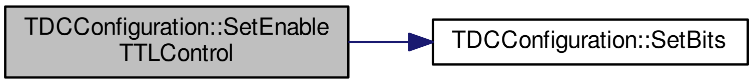
5.11.4.77 void TDCConfiguration::SetEnableTTLControl (const bool *tc* = `true`) `[inline]`, `[private]`

Enable LV TTL inputs on control registers.

Enable LV TTL input on:

- trigger,
- bunch_reset,
- event_reset,
- encoded_control, otherwise uses LVDS input levels.

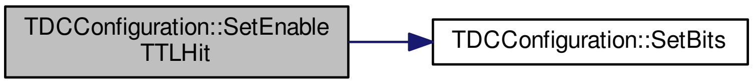
Here is the call graph for this function:



5.11.4.78 void TDCConfiguration::SetEnableTTLHit (const bool *th* = true) [inline], [private]

Enable LV TTL input on hit[31:0], otherwise uses LVDS input levels.

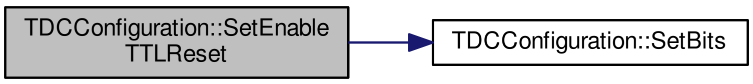
Here is the call graph for this function:



5.11.4.79 void TDCConfiguration::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:


TDCConfiguration::SetEnable TTLReset
TDCConfiguration::SetBits

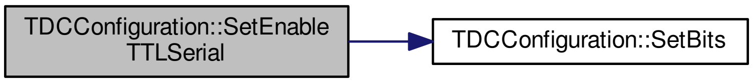
5.11.4.80 void TDCConfiguration::SetEnableTTLSerial (const bool *ts* = true) [inline], [private]

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.

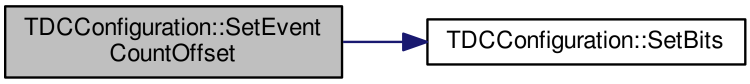
Here is the call graph for this function:


TDCConfiguration::SetEnable TTLSerial
TDCConfiguration::SetBits

5.11.4.81 void TDCConfiguration::SetEventCountOffset (uint16_t *eco*) `[inline]`

Set offset for the event counter.

Here is the call graph for this function:


TDCConfiguration::SetEvent CountOffset
TDCConfiguration::SetBits

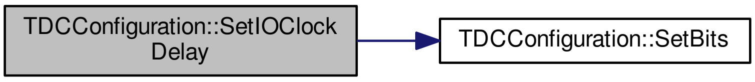
5.11.4.82 void TDCConfiguration::SetIOClockDelay (const bool *delay_clock,* const uint8_t *delay*) `[inline]`, `[private]`

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

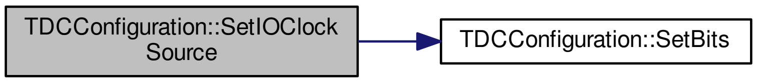
Here is the call graph for this function:



5.11.4.83 void TDCConfiguration::SetIOClockSource (const **IOClockSource** *ics*) `[inline]`, `[private]`

Selection of clock source for I/O signals.

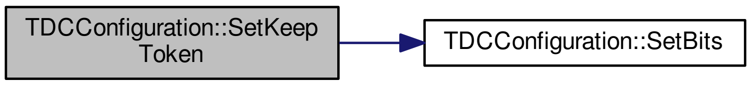
Here is the call graph for this function:


TDCConfiguration::SetIOClock Source
TDCConfiguration::SetBits

5.11.4.84 void TDCConfiguration::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.

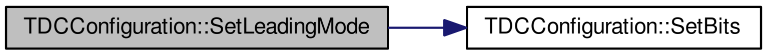
Here is the call graph for this function:



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

Enable the detection of leading edges.

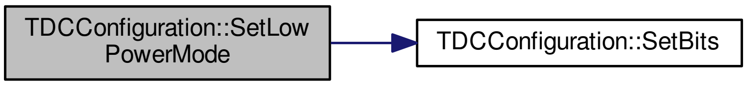
Here is the call graph for this function:



5.11.4.86 void TDCConfiguration::SetLowPowerMode (const bool *lpm* = `true`) `[inline]`, `[private]`

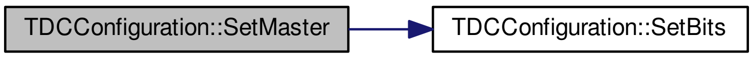
Low power mode of channel buffers.

Here is the call graph for this function:



5.11.4.87 void TDCConfiguration::SetMaster (const bool *m* = `true`) `[inline]`, `[private]`

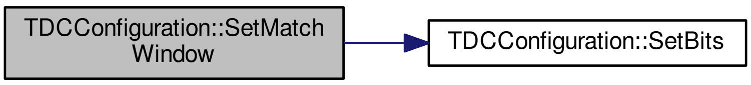
Here is the call graph for this function:



5.11.4.88 void TDCConfiguration::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:

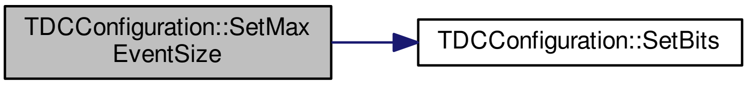


5.11.4.89 void TDCConfiguration::SetMaxEventSize (int *sz* = `-1`) `[inline]`

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.

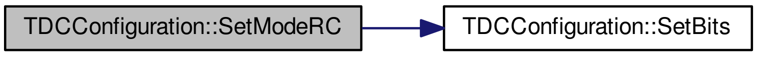
Here is the call graph for this function:



5.11.4.90 void TDCConfiguration::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.

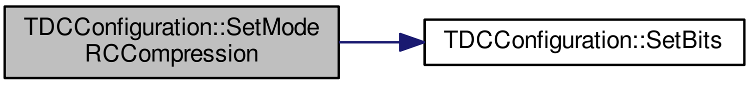
Here is the call graph for this function:



5.11.4.91 void TDCConfiguration::SetModeRCCompression (const bool *mrc* = `true`) `[inline]`, `[private]`

Perform RC interpolation on-chip (only valid in very high resolution mode)

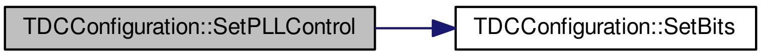
Here is the call graph for this function:


TDCConfiguration::SetMode RCCompression
TDCConfiguration::SetBits

5.11.4.92 void TDCConfiguration::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.

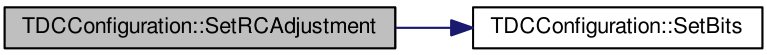
Here is the call graph for this function:



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

Set the adjustment of the RC delay line.

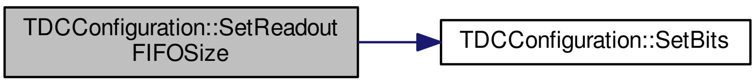
Here is the call graph for this function:



5.11.4.94 void TDCConfiguration::SetReadoutFIFOSize (int *rfs*) `[inline]`

Effective size of readout FIFO.

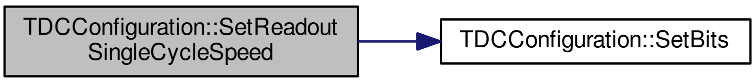
Here is the call graph for this function:


TDCConfiguration::SetReadout FIFOSize
TDCConfiguration::SetBits

5.11.4.95 void TDCConfiguration::SetReadoutSingleCycleSpeed (const ReadoutSingleCycleSpeed *rscs* = RSC_40Mbits_s) `[inline]`, `[private]`

Serial transmission speed in single cycle mode.

Here is the call graph for this function:


TDCConfiguration::SetReadout SingleCycleSpeed
TDCConfiguration::SetBits

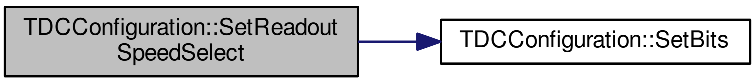
5.11.4.96 void TDCConfiguration::SetReadoutSpeedSelect (const ReadoutSpeed *rss* = RO_Fixed) [inline], [private]

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)

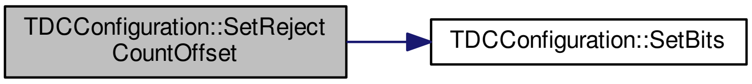
Here is the call graph for this function:


TDCConfiguration::SetReadout SpeedSelect
TDCConfiguration::SetBits

5.11.4.97 void TDCConfiguration::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:

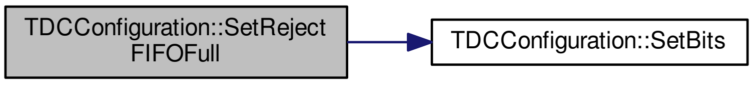

TDCConfiguration::SetReject CountOffset
TDCConfiguration::SetBits

5.11.4.98 void TDCConfiguration::SetRejectFIFOFull (const bool *rej* = `true`) `[inline]`

Reject hits when readout FIFO full.

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

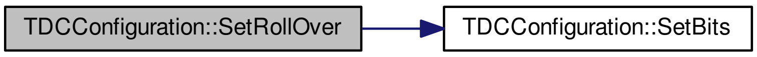
Here is the call graph for this function:


TDCConfiguration::SetReject FIFOFull
TDCConfiguration::SetBits

5.11.4.99 void TDCConfiguration::SetRollOver (const uint16_t *ro* = `0xFFF`) `[inline]`, `[private]`

Counter roll over value, defining maximal count value from where counters will be reset to 0.

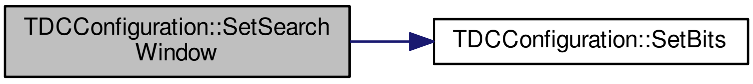
Here is the call graph for this function:



5.11.4.100 void TDCConfiguration::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:



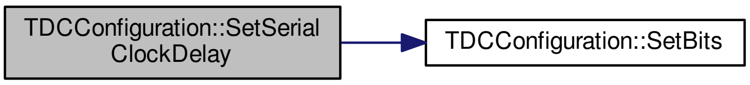
5.11.4.101 void TDCConfiguration::SetSerialClockDelay (const bool *delay_clock*, const uint8_t *delay*) `[inline]`, `[private]`

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

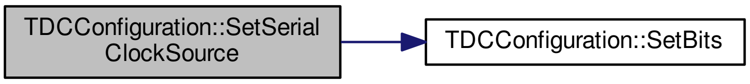
Here is the call graph for this function:


TDCConfiguration::SetSerial ClockDelay
TDCConfiguration::SetBits

5.11.4.102 void TDCConfiguration::SetSerialClockSource (const SerialClockSource *scs*) `[inline]`, `[private]`

Selection of source for serial clock.

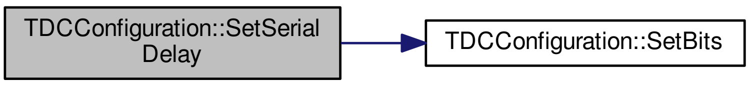
Here is the call graph for this function:


TDCConfiguration::SetSerial ClockSource
TDCConfiguration::SetBits

5.11.4.103 void TDCConfiguration::SetSerialDelay (const uint8_t *sd* = `0x0`) `[inline]`, `[private]`

Programmable delay of serial input, in time unit $\sim$ 1 ns.

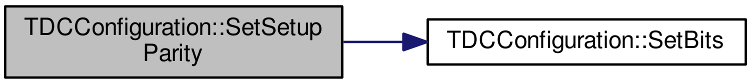
Here is the call graph for this function:



5.11.4.104 void TDCConfiguration::SetSetupParity (const bool *sp* = `true`) `[inline]`

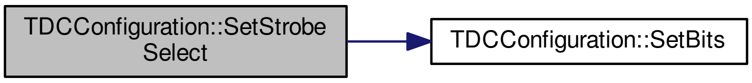
Set the parity of setup data (should be an even parity)

Here is the call graph for this function:



5.11.4.105 void TDCConfiguration::SetStrobeSelect (const SerialStrobeType *ss* = SS_NoStrobe) [inline], [private]

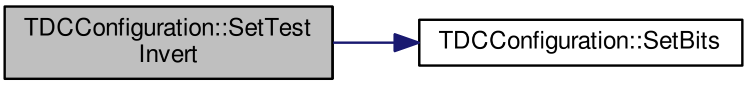
Here is the call graph for this function:



5.11.4.106 void TDCConfiguration::SetTestInvert (const bool *ti* = true) [inline]

Automatic inversion of test pattern. Only used during production testing.

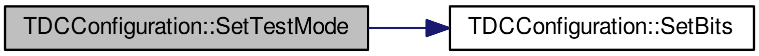
Here is the call graph for this function:



5.11.4.107 void TDCConfiguration::SetTestMode (const bool *tm* = true) [inline]

Test mode where hit data are taken from coretest. Only used during production testing.

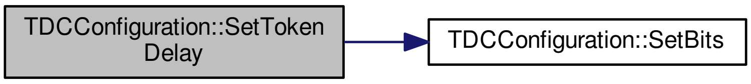
Here is the call graph for this function:



5.11.4.108 void TDCConfiguration::SetTokenDelay (const uint8_t *td* = 0x0) [inline], [private]

Programmable delay of token input, in time unit ∼ 1 ns.

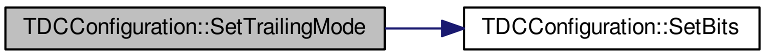
Here is the call graph for this function:



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

Enable/disable the detection of trailing edges.

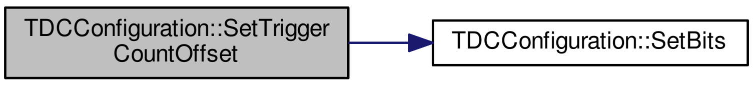
Here is the call graph for this function:



5.11.4.110 void TDCConfiguration::SetTriggerCountOffset (uint16_t *tco*) [inline]

Set offset for the trigger time tag counter to set effective trigger latency.

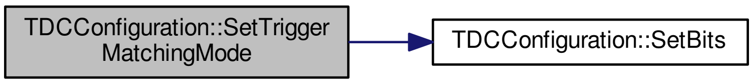
Here is the call graph for this function:


TDCConfiguration::SetTrigger CountOffset
TDCConfiguration::SetBits

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

Set the enable status of trigger matching mode.

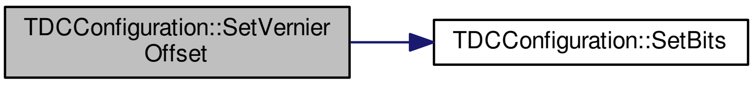
Here is the call graph for this function:


TDCConfiguration::SetTrigger MatchingMode
TDCConfiguration::SetBits

5.11.4.112 void TDCConfiguration::SetVernierOffset (const uint8_t *vo*) `[inline]`

Set the offset in vernier decoding.

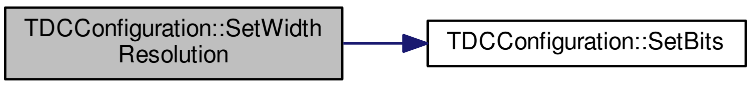
Here is the call graph for this function:


TDCConfiguration::SetVernier Offset
TDCConfiguration::SetBits

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

Set the pulse width resolution when paired measurements are performed.

Here is the call graph for this function:


TDCConfiguration::SetWidth Resolution
TDCConfiguration::SetBits

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

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

5.11.5 Field Documentation

5.11.5.1 word_t TDCConfiguration::fWord[kNumWords] [private]

5.11.5.2 const bit TDCConfiguration::kCoarseCountOffset = 447 [static], [private]

5.11.5.3 const bit TDCConfiguration::kCoreClockDelay = 610 [static], [private]

5.11.5.4 const bit TDCConfiguration::kCoreClockSource = 622 [static], [private]

5.11.5.5 const bit TDCConfiguration::kDeadTime = 584 [static], [private]

5.11.5.6 const bit TDCConfiguration::kDLLClockDelay = 614 [static], [private]

5.11.5.7 const bit TDCConfiguration::kDLLClockSource = 624 [static], [private]

5.11.5.8 const bit TDCConfiguration::kDLLControl = 580 [static], [private]

5.11.5.9 const bit TDCConfiguration::kDLLMode = 592 [static], [private]

5.11.5.10 const bit TDCConfiguration::kDLLTapAdjust0 = 459 [static], [private]

5.11.5.11 const bit TDCConfiguration::kEnableAutomaticReject = 125 [static], [private]

5.11.5.12 const bit TDCConfiguration::kEnableBytewise = 37 [static], [private]

5.11.5.13 const bit TDCConfiguration::kEnableDirectBunchReset = 157 [static], [private]

5.11.5.14 const bit TDCConfiguration::kEnableDirectEventReset = 156 [static], [private]

5.11.5.15 const bit TDCConfiguration::kEnableDirectTrigger = 158 [static], [private]

5.11.5.16 const bit TDCConfiguration::kEnableError = 6 [static], [private]

5.11.5.17 const bit TDCConfiguration::kEnableErrorBypass = 5 [static], [private]

5.11.5.18 const bit TDCConfiguration::kEnableErrorMark = 4 [static], [private]

5.11.5.19 const bit TDCConfiguration::kEnableGlobalHeader = 34 [static], [private]

5.11.5.20 const bit TDCConfiguration::kEnableGlobalTrailer = 33 [static], [private]

5.11.5.21 const bit TDCConfiguration::kEnableJTAGReadout = 39 [static], [private]

5.11.5.22 const bit TDCConfiguration::kEnableLocalHeader = 32 [static], [private]

5.11.5.23 const bit TDCConfiguration::kEnableLocalTrailer = 31 [static], [private]

5.11.5.24 const bit TDCConfiguration::kEnableMasterResetCode = 151 [static], [private]

5.11.5.25 const bit TDCConfiguration::kEnableMasterResetOnEventReset = 152 [static], [private]

5.11.5.26 const bit TDCConfiguration::kEnableMatching = 639 [static], [private]

5.11.5.27 const bit TDCConfiguration::kEnableOverflowDetect = 123 [static], [private]

5.11.5.28 const bit TDCConfiguration::kEnablePair = 640 [static], [private]

5.11.5.29 const bit TDCConfiguration::kEnableReadoutOccupancy = 121 [static], [private]

5.11.5.30 const bit TDCConfiguration::kEnableReadoutSeparator = 122 [static], [private]

5.11.5.31 const bit TDCConfiguration::kEnableRelative = 124 [static], [private]

5.11.5.32 const bit TDCConfiguration::kEnableResetChannelBufferWhenSeparator = 153 [static], [private]

5.11.5.33 const bit TDCConfiguration::kEnableSeparatorOnBunchReset = 155 [static], [private]

5.11.5.34 const bit TDCConfiguration::kEnableSeparatorOnEventReset = 154 [static], [private]

5.11.5.35 const bit TDCConfiguration::kEnableSerial = 38 [static], [private]

5.11.5.36 const bit TDCConfiguration::kEnableSetCountersOnBunchReset = 150 [static], [private]

5.11.5.37 const bit TDCConfiguration::kEnableTTLClock = 644 [static], [private]

5.11.5.38 const bit TDCConfiguration::kEnableTTLControl = 642 [static], [private]

5.11.5.39 const bit TDCConfiguration::kEnableTTLHit = 645 [static], [private]

5.11.5.40 const bit TDCConfiguration::kEnableTTLReset = 643 [static], [private]

5.11.5.41 const bit TDCConfiguration::kEnableTTLSerial = 641 [static], [private]

5.11.5.42 const bit TDCConfiguration::kEventCountOffset = 126 [static], [private]

5.11.5.43 const bit TDCConfiguration::kIOClockDelay = 606 [static], [private]

5.11.5.44 const bit TDCConfiguration::kIOClockSource = 620 [static], [private]

5.11.5.45 const bit TDCConfiguration::kKeepToken = 35 [static], [private]

5.11.5.46 const bit TDCConfiguration::kLeading = 589 [static], [private]

5.11.5.47 const bit TDCConfiguration::kLeadingResolution = 84 [static], [private]

5.11.5.48 const bit TDCConfiguration::kLowPowerMode = 570 [static], [private]

5.11.5.49 const bit TDCConfiguration::kMaster = 36 [static], [private]

5.11.5.50 const bit TDCConfiguration::kMatchWindow = 72 [static], [private]

5.11.5.51 const bit TDCConfiguration::kMaxEventSize = 116 [static], [private]

5.11.5.52 const bit TDCConfiguration::kModeRC = 591 [static], [private]

5.11.5.53 const bit TDCConfiguration::kModeRCCompression = 590 [static], [private]

5.11.5.54 const uint8_t TDCConfiguration::kNumWords = BITS_NUM/WORD_SIZE+1 [static], [private]

5.11.5.55 const bit TDCConfiguration::kOffset0 = 438 [static], [private]

5.11.5.56 const bit TDCConfiguration::kPLLControl = 594 [static], [private]

5.11.5.57 const bit TDCConfiguration::kRCAdjust0 = 555 [static], [private]

5.11.5.58 const bit TDCConfiguration::kReadoutFIFOSize = 45 [static], [private]

5.11.5.59 const bit TDCConfiguration::kReadoutSingleCycleSpeed = 17 [static], [private]

5.11.5.60 const bit TDCConfiguration::kReadoutSpeedSelect = 26 [static], [private]

5.11.5.61 const bit TDCConfiguration::kRejectCountOffset = 48 [static], [private]

5.11.5.62 const bit TDCConfiguration::kRejectFIFOFull = 120 [static], [private]

5.11.5.63 const bit TDCConfiguration::kRollOver = 627 [static], [private]

5.11.5.64 const bit TDCConfiguration::kSearchWindow = 60 [static], [private]

5.11.5.65 const bit TDCConfiguration::kSelectBypassInputs = 44 [static], [private]

5.11.5.66 const bit TDCConfiguration::kSerialClockDelay = 602 [static], [private]

5.11.5.67 const bit TDCConfiguration::kSerialClockSource = 618 [static], [private]

5.11.5.68 const bit TDCConfiguration::kSerialDelay = 20 [static], [private]

5.11.5.69 const bit TDCConfiguration::kSetupParity = 646 [static], [private]

5.11.5.70 const bit TDCConfiguration::kStrobeSelect = 24 [static], [private]

5.11.5.71 const bit TDCConfiguration::kTDCId = 40 [static], [private]

5.11.5.72 const bit TDCConfiguration::kTestInvert = 586 [static], [private]

5.11.5.73 const bit TDCConfiguration::kTestMode = 587 [static], [private]

5.11.5.74 const bit TDCConfiguration::kTestSelect = 0 [static], [private]

5.11.5.75 const bit TDCConfiguration::kTokenDelay = 27 `[static]`, `[private]`

5.11.5.76 const bit TDCConfiguration::kTrailing = 588 `[static]`, `[private]`

5.11.5.77 const bit TDCConfiguration::kTriggerCountOffset = 138 `[static]`, `[private]`

5.11.5.78 const bit TDCConfiguration::kVernierOffset = 575 `[static]`, `[private]`

5.11.5.79 const bit TDCConfiguration::kWidthSelect = 571 `[static]`, `[private]`

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

- include/TDCConfiguration.h
- src/TDCConfiguration.cpp

5.12 TDCEvent Class Reference

HPTDC event parser.

`#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.

Private Attributes

- uint32_t fWord

5.12.1 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

20 Apr 2015

5.12.2 Constructor & Destructor Documentation

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

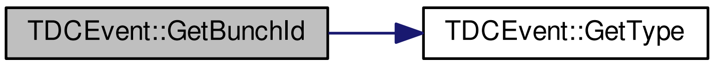
5.12.2.2 virtual TDCEvent::∼TDCEvent ( ) `[inline], [virtual]`

5.12.3 Member Function Documentation

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

Bunch identifier of trigger (or trigger time tag)

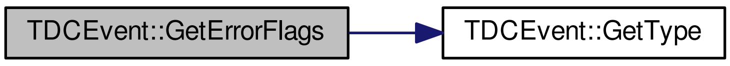
Here is the call graph for this function:



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

Return error flags if an error condition has been detected.

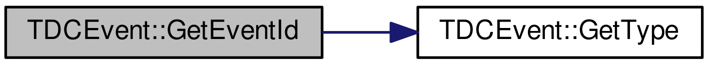
Here is the call graph for this function:



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

Event identifier from event counter.

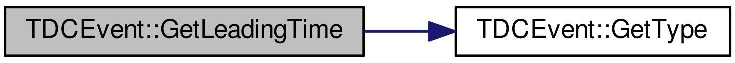
Here is the call graph for this function:



5.12.3.4 uint32_t TDCEvent::GetLeadingTime (bool *pair* = false) const [inline]

Leading edge measurement in programmed time resolution.

Here is the call graph for this function:



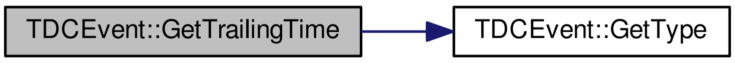
5.12.3.5 unsigned int TDCEvent::GetTDCId ( ) const [inline]

Programmed identifier of master TDC.

5.12.3.6 uint32_t TDCEvent::GetTrailingTime ( ) const [inline]

Trailing edge measurement in programmed time resolution.

Here is the call graph for this function:



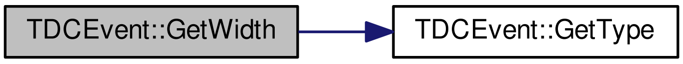
5.12.3.7 EventType TDCEvent::GetType ( ) const [inline]

Type of packet read out from the TDC.

5.12.3.8 uint8_t TDCEvent::GetWidth ( ) const [inline]

Width of pulse in programmed time resolution.

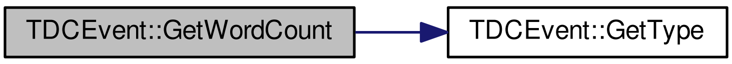
Here is the call graph for this function:



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

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

Here is the call graph for this function:



5.12.4 Field Documentation

5.12.4.1 uint32_t TDCEvent::fWord [private]

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

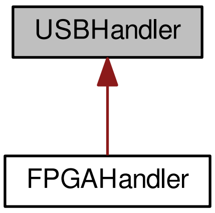
- include/TDCEvent.h

5.13 USBHandler Class Reference

Generic USB communication handler.

`#include <USBHandler.h>`

Inheritance diagram for USBHandler:



Public Member Functions

- USBHandler (const char ∗dev)
- virtual ∼USBHandler ()
- void Init ()
- void DumpDevice (libusb_device ∗dev, int verb=1, std::ostream &out=std::cout)

Protected Member Functions

- void Write (uint32_t word, uint8_t size) const

  Write a word to the USB device.
- uint32_t Fetch (uint8_t size) const

  Receive a word from the USB device.

Private Attributes

- std::string fDevice
- libusb_device_handle ∗ fHandle

5.13.1 Detailed Description

Generic USB communication handler.

Date

21 Apr 2015

Author

Laurent Forthomme `laurent.forthomme@cern.ch`

5.13.2 Constructor & Destructor Documentation

5.13.2.1 USBHandler::USBHandler (const char ∗ *dev*)

5.13.2.2 virtual USBHandler::∼USBHandler ( ) `[inline], [virtual]`

5.13.3 Member Function Documentation

5.13.3.1 void USBHandler::DumpDevice (libusb_device ∗ *dev,* int *verb* = `1`, std::ostream & *out* = `std::cout`)

5.13.3.2 uint32_t USBHandler::Fetch (uint8_t *size*) const `[inline]`, `[protected]`

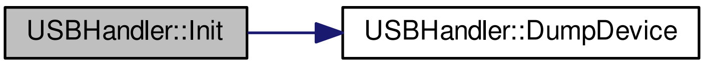
Receive a word from the USB device.

5.13.3.3 void USBHandler::Init ( )

Pointer to a pointer of devices used to retrieve a list of them

A libusb session

Here is the call graph for this function:



5.13.3.4 void USBHandler::Write (uint32_t *word,* uint8_t *size*) const `[inline]`, `[protected]`

Write a word to the USB device.

5.13.4 Field Documentation

5.13.4.1 std::string USBHandler::fDevice `[private]`

5.13.4.2 libusb_device_handle∗ USBHandler::fHandle `[private]`

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

- include/USBHandler.h
- src/USBHandler.cpp

Index

∼Client
  Client, 17
∼Exception
  Exception, 21
∼FPGAHandler
  FPGAHandler, 26
∼Message
  Message, 33
∼Messenger
  Messenger, 35
∼Socket
  Socket, 43
∼SocketMessage
  SocketMessage, 51
∼TDCConfiguration
  TDCConfiguration, 62
∼TDCEvent
  TDCEvent, 105
∼USBHandler
  USBHandler, 108

AcceptConnections
  Socket, 43
AddClient
  Messenger, 36
Announce
  Client, 17

Bind
  Socket, 43
bit
  TDCConfiguration, 60
Broadcast
  Messenger, 36

CLIENT
  Socket communication objects, 7
ChannelSelectError
  HPTDC chip control, 11
Client, 15
  ∼Client, 17
  Announce, 17
  Client, 17
  Connect, 18
  Disconnect, 18
  fClientId, 20
  fIsConnected, 20
  GetType, 19
  ParseMessage, 19
  Receive, 19
  Send, 19
CloseFile
  FPGAHandler, 26
CoarseError
  HPTDC chip control, 11
config
  file_header_t, 23
Configure
  Socket, 43
Connect
  Client, 18
  Messenger, 37
ControlParityError
  HPTDC chip control, 11
Core_aux_clock
  HPTDC chip control, 10
Core_clock_40
  HPTDC chip control, 10
Core_pll_clock_160
  HPTDC chip control, 10
Core_pll_clock_80
  HPTDC chip control, 10
CoreClockSource
  HPTDC chip control, 10
Create
  Socket, 44

DETECTOR
  Socket communication objects, 8
DLL_160MHz
  HPTDC chip control, 10
DLL_320MHz
  HPTDC chip control, 10
DLL_40MHz
  HPTDC chip control, 10
DLL_Illegal
  HPTDC chip control, 10
DLL_aux_clock
  HPTDC chip control, 10
DLL_clock_40
  HPTDC chip control, 10
DLL_pll_clock_160
  HPTDC chip control, 10
DLL_pll_clock_320
  HPTDC chip control, 10
DLL_pll_clock_40
  HPTDC chip control, 10
DLLClockSource
  HPTDC chip control, 10
DLLSpeedMode

HPTDC chip control, 10
DT_100ns
HPTDC chip control, 10
DT_10ns
HPTDC chip control, 10
DT_30ns
HPTDC chip control, 10
DT_5ns
HPTDC chip control, 10
DeadTime
HPTDC chip control, 10
Debug
HPTDC chip control, 11
Decode
HTTPMessage, 31
Description
Exception, 21
Disconnect
Client, 18
Messenger, 37
DisconnectClient
Messenger, 38
Dump
Exception, 21
HTTPMessage, 31
Message, 33
SocketMessage, 51
TDCConfiguration, 62
DumpConnected
Socket, 44
DumpDevice
USBHandler, 109

E_100ps
HPTDC chip control, 11
E_12p5ns
HPTDC chip control, 11
E_1p6ns
HPTDC chip control, 11
E_200ps
HPTDC chip control, 11
E_3p12ns
HPTDC chip control, 11
E_400ps
HPTDC chip control, 11
E_6p25ns
HPTDC chip control, 11
E_800ps
HPTDC chip control, 11
EdgeResolution
HPTDC chip control, 10
EnabledError
HPTDC chip control, 11
Encode
HTTPMessage, 31
Error
HPTDC chip control, 11
ErrorNumber
Exception, 22
EventType
HPTDC chip control, 11
Exception, 20
∼Exception, 21
Description, 21
Dump, 21
ErrorNumber, 22
Exception, 21
fDescription, 22
fErrorNumber, 22
fFrom, 22
fType, 22
From, 22
Type, 22
TypeString, 22

fAddress
Socket, 46
fBuffer
Socket, 46
fClientId
Client, 20
fConfig
FPGAHandler, 28
fDescription
Exception, 22
fDevice
USBHandler, 109
fErrorNumber
Exception, 22
fFilename
FPGAHandler, 28
fFrom
Exception, 22
fHandle
USBHandler, 109
fIsConnected
Client, 20
fIsFileOpen
FPGAHandler, 28
fIsTDCInReadout
FPGAHandler, 28
fListenersInfo
Messenger, 40
fMaster
Socket, 46
fMessage
SocketMessage, 53
fNumAttempts
Messenger, 40
fOriginalString
HTTPMessage, 31
fOutput
FPGAHandler, 28
FPGAHandler, 24
∼FPGAHandler, 26
CloseFile, 26
fConfig, 28
fFilename, 28

fIsFileOpen, 28
fIsTDCInReadout, 28
fOutput, 28
FPGAHandler, 26
GetConfiguration, 26
GetFilename, 26
GetType, 27
OpenFile, 27
ReadBuffer, 27
ReadConfiguration, 27
SendConfiguration, 27
SetConfiguration, 28
fPort
Socket, 46
fReadFds
Socket, 46
fSocketId
Socket, 46
fSocketsConnected
Socket, 46
fString
Message, 33
fType
Exception, 22
fWS
HTTPMessage, 31
Messenger, 40
fWord
TDCConfiguration, 101
TDCEvent, 107
Fetch
USBHandler, 109
FetchMessage
Socket, 44
file_header_t, 23
config, 23
magic, 23
run_id, 23
spill_id, 23
From
Exception, 22

GetBits
TDCConfiguration, 63
GetBunchId
TDCEvent, 105
GetChannelOffset
TDCConfiguration, 63
GetCoarseCountOffset
TDCConfiguration, 64
GetConfiguration
FPGAHandler, 26
GetDLLAdjustment
TDCConfiguration, 64
GetDeadTime
TDCConfiguration, 64
GetEdgeResolution
TDCConfiguration, 65
GetEdgesPairing
TDCConfiguration, 65
GetEnableError
TDCConfiguration, 65
GetEnableErrorBypass
TDCConfiguration, 65
GetEnableErrorMark
TDCConfiguration, 66
GetEnableJTAGReadout
TDCConfiguration, 66
GetEnableReadoutOccupancy
TDCConfiguration, 66
GetEnableReadoutSeparator
TDCConfiguration, 67
GetEnableSerial
TDCConfiguration, 67
GetErrorFlags
TDCEvent, 105
GetEventId
TDCEvent, 106
GetFilename
FPGAHandler, 26
GetIntValue
SocketMessage, 51
GetKey
HTTPMessage, 31
Message, 33
SocketMessage, 51
GetLeadingMode
TDCConfiguration, 67
GetLeadingTime
TDCEvent, 106
GetMatchWindow
TDCConfiguration, 68
GetMaxEventSize
TDCConfiguration, 68
GetNumWords
TDCConfiguration, 68
GetPort
Socket, 44
GetRCAdjustment
TDCConfiguration, 68
GetReadoutFIFOSize
TDCConfiguration, 69
GetRejectCountOffset
TDCConfiguration, 69
GetRejectFIFOFull
TDCConfiguration, 69
GetSearchWindow
TDCConfiguration, 70
GetSetupParity
TDCConfiguration, 70
GetSocketId
Socket, 44
GetSocketType
Socket, 44
GetString
Message, 33
SocketMessage, 52

GetTDCId
  TDCEvent, 106
GetTestInvert
  TDCConfiguration, 70
GetTestMode
  TDCConfiguration, 71
GetTrailingMode
  TDCConfiguration, 71
GetTrailingTime
  TDCEvent, 106
GetTriggerCountOffset
  TDCConfiguration, 71
GetTriggerLatency
  TDCConfiguration, 72
GetTriggerMatchingMode
  TDCConfiguration, 72
GetType
  Client, 19
  FPGAHandler, 27
  Messenger, 38
  TDCEvent, 106
GetValue
  SocketMessage, 52
GetVectorValue
  SocketMessage, 52
GetVernierOffset
  TDCConfiguration, 72
GetWidth
  TDCEvent, 107
GetWidthResolution
  TDCConfiguration, 73
GetWord
  TDCConfiguration, 73
GetWordCount
  TDCEvent, 107
GroupHeader
  HPTDC chip control, 11
GroupTrailer
  HPTDC chip control, 11

HPTDC chip control, 9
  ChannelSelectError, 11
  CoarseError, 11
  ControlParityError, 11
  Core_aux_clock, 10
  Core_clock_40, 10
  Core_pll_clock_160, 10
  Core_pll_clock_80, 10
  CoreClockSource, 10
  DLL_160MHz, 10
  DLL_320MHz, 10
  DLL_40MHz, 10
  DLL_Illegal, 10
  DLL_aux_clock, 10
  DLL_clock_40, 10
  DLL_pll_clock_160, 10
  DLL_pll_clock_320, 10
  DLL_pll_clock_40, 10
  DLLClockSource, 10
  DLLSpeedMode, 10
  DT_100ns, 10
  DT_10ns, 10
  DT_30ns, 10
  DT_5ns, 10
  DeadTime, 10
  Debug, 11
  E_100ps, 11
  E_12p5ns, 11
  E_1p6ns, 11
  E_200ps, 11
  E_3p12ns, 11
  E_400ps, 11
  E_6p25ns, 11
  E_800ps, 11
  EdgeResolution, 10
  EnabledError, 11
  Error, 11
  EventType, 11
  GroupHeader, 11
  GroupTrailer, 11
  IO_aux_clock, 11
  IO_clock_40, 11
  IO_pll_clock_160, 11
  IO_pll_clock_80, 11
  IOClockSource, 11
  Invalid, 11
  JTAGInstructionParityError, 11
  L1BufferParityError, 11
  LeadingEdge, 11
  RO_Fixed, 12
  RO_pll_80Mbits_s, 12
  RSC_10Mbits_s, 12
  RSC_1p25Mbits_s, 12
  RSC_20Mbits_s, 12
  RSC_2p5Mbits_s, 12
  RSC_312p5kbits_s, 12
  RSC_40Mbits_s, 12
  RSC_5Mbits_s, 12
  RSC_625kbits_s, 12
  ReadoutFIFOParityError, 11
  ReadoutSingleCycleSpeed, 11
  ReadoutSpeed, 12
  ReadoutStateError, 11
  SS_DSStrobe, 12
  SS_LeadingEdge, 12
  SS_LeadingTrailingStrobe, 12
  SS_NoStrobe, 12
  Serial_aux_clock, 12
  Serial_pll_clock_160, 12
  Serial_pll_clock_40, 12
  Serial_pll_clock_80, 12
  SerialClockSource, 12
  SerialStrobeType, 12
  SetupParityError, 11
  TDCHeader, 11
  TDCTrailer, 11
  TrailingEdge, 11

TriggerFIFOParityError, 11
TriggerMatchingError, 11
VernierError, 11
W_100ns, 13
W_100ps, 12
W_12p5ns, 12
W_1p6ns, 12
W_200ns, 13
W_200ps, 12
W_25ns, 13
W_3p2ns, 12
W_400ns, 13
W_400ps, 12
W_50ns, 13
W_6p25ns, 12
W_800ns, 13
W_800ps, 12
WidthResolution, 12
HTTPMessage, 29
Decode, 31
Dump, 31
Encode, 31
fOriginalString, 31
fWS, 31
GetKey, 31
HTTPMessage, 30

INVALID
Socket communication objects, 7
IO_aux_clock
HPTDC chip control, 11
IO_clock_40
HPTDC chip control, 11
IO_pll_clock_160
HPTDC chip control, 11
IO_pll_clock_80
HPTDC chip control, 11
IOClockSource
HPTDC chip control, 11
Init
USBHandler, 109
Invalid
HPTDC chip control, 11
IsFromWeb
Message, 33
IsWebSocket
Socket, 44

JTAGInstructionParityError
HPTDC chip control, 11

kCoarseCountOffset
TDCConfiguration, 101
kCoreClockDelay
TDCConfiguration, 101
kCoreClockSource
TDCConfiguration, 101
kDLLClockDelay
TDCConfiguration, 101
kDLLClockSource
TDCConfiguration, 101
kDLLControl
TDCConfiguration, 101
kDLLMode
TDCConfiguration, 101
kDLLTapAdjust0
TDCConfiguration, 101
kDeadTime
TDCConfiguration, 101
kEnableAutomaticReject
TDCConfiguration, 101
kEnableBytewise
TDCConfiguration, 101
kEnableDirectBunchReset
TDCConfiguration, 101
kEnableDirectEventReset
TDCConfiguration, 101
kEnableDirectTrigger
TDCConfiguration, 101
kEnableError
TDCConfiguration, 101
kEnableErrorBypass
TDCConfiguration, 101
kEnableErrorMark
TDCConfiguration, 101
kEnableGlobalHeader
TDCConfiguration, 101
kEnableGlobalTrailer
TDCConfiguration, 102
kEnableJTAGReadout
TDCConfiguration, 102
kEnableLocalHeader
TDCConfiguration, 102
kEnableLocalTrailer
TDCConfiguration, 102
kEnableMasterResetCode
TDCConfiguration, 102
kEnableMasterResetOnEventReset
TDCConfiguration, 102
kEnableMatching
TDCConfiguration, 102
kEnableOverflowDetect
TDCConfiguration, 102
kEnablePair
TDCConfiguration, 102
kEnableReadoutOccupancy
TDCConfiguration, 102
kEnableReadoutSeparator
TDCConfiguration, 102
kEnableRelative
TDCConfiguration, 102
kEnableResetChannelBufferWhenSeparator
TDCConfiguration, 102
kEnableSeparatorOnBunchReset
TDCConfiguration, 102
kEnableSeparatorOnEventReset
TDCConfiguration, 102

kEnableSerial
TDCConfiguration, 102
kEnableSetCountersOnBunchReset
TDCConfiguration, 102
kEnableTTLClock
TDCConfiguration, 102
kEnableTTLControl
TDCConfiguration, 102
kEnableTTLHit
TDCConfiguration, 102
kEnableTTLReset
TDCConfiguration, 102
kEnableTTLSerial
TDCConfiguration, 102
kEventCountOffset
TDCConfiguration, 102
kIOClockDelay
TDCConfiguration, 102
kIOClockSource
TDCConfiguration, 102
kKeepToken
TDCConfiguration, 102
kLeading
TDCConfiguration, 102
kLeadingResolution
TDCConfiguration, 102
kLowPowerMode
TDCConfiguration, 103
kMaster
TDCConfiguration, 103
kMatchWindow
TDCConfiguration, 103
kMaxEventSize
TDCConfiguration, 103
kModeRC
TDCConfiguration, 103
kModeRCCompression
TDCConfiguration, 103
kNumWords
TDCConfiguration, 103
kOffset0
TDCConfiguration, 103
kPLLControl
TDCConfiguration, 103
kRCAdjust0
TDCConfiguration, 103
kReadoutFIFOSize
TDCConfiguration, 103
kReadoutSingleCycleSpeed
TDCConfiguration, 103
kReadoutSpeedSelect
TDCConfiguration, 103
kRejectCountOffset
TDCConfiguration, 103
kRejectFIFOFull
TDCConfiguration, 103
kRollOver
TDCConfiguration, 103
kSearchWindow
TDCConfiguration, 103
kSelectBypassInputs
TDCConfiguration, 103
kSerialClockDelay
TDCConfiguration, 103
kSerialClockSource
TDCConfiguration, 103
kSerialDelay
TDCConfiguration, 103
kSetupParity
TDCConfiguration, 103
kStrobeSelect
TDCConfiguration, 103
kTDCId
TDCConfiguration, 103
kTestInvert
TDCConfiguration, 103
kTestMode
TDCConfiguration, 103
kTestSelect
TDCConfiguration, 103
kTokenDelay
TDCConfiguration, 103
kTrailing
TDCConfiguration, 104
kTriggerCountOffset
TDCConfiguration, 104
kVernierOffset
TDCConfiguration, 104
kWidthSelect
TDCConfiguration, 104

L1BufferParityError
HPTDC chip control, 11
LeadingEdge
HPTDC chip control, 11
Listen
Socket, 44
ListenerInfo, 31
name, 31
type, 31

MASTER
Socket communication objects, 7
magic
file_header_t, 23
Message, 31
∼Message, 33
Dump, 33
fString, 33
GetKey, 33
GetString, 33
IsFromWeb, 33
Message, 33
Messenger, 33
∼Messenger, 35
AddClient, 36
Broadcast, 36

Connect, 37
Disconnect, 37
DisconnectClient, 38
fListenersInfo, 40
fNumAttempts, 40
fWS, 40
GetType, 38
Messenger, 35
ProcessMessage, 38
Receive, 39
Send, 39
SwitchClientType, 40

name
ListenerInfo, 31

Object
SocketMessage, 52
OpenFile
FPGAHandler, 27

ParseMessage
Client, 19
PrepareConnection
Socket, 45
ProcessMessage
Messenger, 38

RO_Fixed
HPTDC chip control, 12
RO_pll_80Mbits_s
HPTDC chip control, 12
RSC_10Mbits_s
HPTDC chip control, 12
RSC_1p25Mbits_s
HPTDC chip control, 12
RSC_20Mbits_s
HPTDC chip control, 12
RSC_2p5Mbits_s
HPTDC chip control, 12
RSC_312p5kbits_s
HPTDC chip control, 12
RSC_40Mbits_s
HPTDC chip control, 12
RSC_5Mbits_s
HPTDC chip control, 12
RSC_625kbits_s
HPTDC chip control, 12
ReadBuffer
FPGAHandler, 27
ReadConfiguration
FPGAHandler, 27
ReadoutFIFOParityError
HPTDC chip control, 11
ReadoutSingleCycleSpeed
HPTDC chip control, 11
ReadoutSpeed
HPTDC chip control, 12
ReadoutStateError
HPTDC chip control, 11
Receive
Client, 19
Messenger, 39
run_id
file_header_t, 23

SS_DSStrobe
HPTDC chip control, 12
SS_LeadingEdge
HPTDC chip control, 12
SS_LeadingTrailingStrobe
HPTDC chip control, 12
SS_NoStrobe
HPTDC chip control, 12
SelectConnections
Socket, 45
Send
Client, 19
Messenger, 39
SendConfiguration
FPGAHandler, 27
SendMessage
Socket, 45
Serial_aux_clock
HPTDC chip control, 12
Serial_pll_clock_160
HPTDC chip control, 12
Serial_pll_clock_40
HPTDC chip control, 12
Serial_pll_clock_80
HPTDC chip control, 12
SerialClockSource
HPTDC chip control, 12
SerialStrobeType
HPTDC chip control, 12
SetAllChannelsOffset
TDCConfiguration, 73
SetAllTapsDLLAdjustment
TDCConfiguration, 73
SetBits
TDCConfiguration, 74
SetBypassInputs
TDCConfiguration, 74
SetChannelOffset
TDCConfiguration, 74
SetCoarseCountOffset
TDCConfiguration, 74
SetConfiguration
FPGAHandler, 28
SetConstantValues
TDCConfiguration, 75
SetCoreClockDelay
TDCConfiguration, 76
SetCoreClockSource
TDCConfiguration, 77
SetDLLAdjustment
TDCConfiguration, 77
SetDLLClockDelay

TDCConfiguration, 78
SetDLLClockSource
TDCConfiguration, 78
SetDLLControl
TDCConfiguration, 78
SetDLLMode
TDCConfiguration, 79
SetDeadTime
TDCConfiguration, 77
SetEdgeResolution
TDCConfiguration, 79
SetEdgesPairing
TDCConfiguration, 79
SetEnableAutomaticReject
TDCConfiguration, 80
SetEnableBytewise
TDCConfiguration, 80
SetEnableDirectBunchReset
TDCConfiguration, 80
SetEnableDirectEventReset
TDCConfiguration, 81
SetEnableDirectTrigger
TDCConfiguration, 81
SetEnableError
TDCConfiguration, 81
SetEnableErrorBypass
TDCConfiguration, 82
SetEnableErrorMark
TDCConfiguration, 82
SetEnableGlobalHeader
TDCConfiguration, 82
SetEnableGlobalTrailer
TDCConfiguration, 83
SetEnableJTAGReadout
TDCConfiguration, 83
SetEnableLocalHeader
TDCConfiguration, 83
SetEnableLocalTrailer
TDCConfiguration, 84
SetEnableMasterResetCode
TDCConfiguration, 84
SetEnableMasterResetOnEventReset
TDCConfiguration, 84
SetEnableOverflowDetect
TDCConfiguration, 85
SetEnableReadoutOccupancy
TDCConfiguration, 85
SetEnableReadoutSeparator
TDCConfiguration, 85
SetEnableRelative
TDCConfiguration, 86
SetEnableResetChannelBufferWhenSeparator
TDCConfiguration, 86
SetEnableSeparatorOnBunchReset
TDCConfiguration, 86
SetEnableSeparatorOnEventReset
TDCConfiguration, 87
SetEnableSerial
TDCConfiguration, 87
SetEnableSetCountersOnBunchReset
TDCConfiguration, 87
SetEnableTTLClock
TDCConfiguration, 88
SetEnableTTLControl
TDCConfiguration, 88
SetEnableTTLHit
TDCConfiguration, 88
SetEnableTTLReset
TDCConfiguration, 89
SetEnableTTLSerial
TDCConfiguration, 89
SetEventCountOffset
TDCConfiguration, 90
SetIOClockDelay
TDCConfiguration, 90
SetIOClockSource
TDCConfiguration, 90
SetKeepToken
TDCConfiguration, 91
SetKeyValue
SocketMessage, 52, 53
SetLeadingMode
TDCConfiguration, 91
SetLowPowerMode
TDCConfiguration, 91
SetMaster
TDCConfiguration, 92
SetMatchWindow
TDCConfiguration, 92
SetMaxEventSize
TDCConfiguration, 92
SetModeRC
TDCConfiguration, 93
SetModeRCCompression
TDCConfiguration, 93
SetPLLControl
TDCConfiguration, 93
SetPort
Socket, 45
SetRCAdjustment
TDCConfiguration, 94
SetReadoutFIFOSize
TDCConfiguration, 94
SetReadoutSingleCycleSpeed
TDCConfiguration, 94
SetReadoutSpeedSelect
TDCConfiguration, 95
SetRejectCountOffset
TDCConfiguration, 95
SetRejectFIFOFull
TDCConfiguration, 95
SetRollOver
TDCConfiguration, 96
SetSearchWindow
TDCConfiguration, 96
SetSerialClockDelay

TDCConfiguration, 96
SetSerialClockSource
TDCConfiguration, 97
SetSerialDelay
TDCConfiguration, 97
SetSetupParity
TDCConfiguration, 97
SetSocketId
Socket, 45
SetStrobeSelect
TDCConfiguration, 98
SetTestInvert
TDCConfiguration, 98
SetTestMode
TDCConfiguration, 98
SetTokenDelay
TDCConfiguration, 99
SetTrailingMode
TDCConfiguration, 99
SetTriggerCountOffset
TDCConfiguration, 99
SetTriggerMatchingMode
TDCConfiguration, 100
SetVernierOffset
TDCConfiguration, 100
SetWidthResolution
TDCConfiguration, 100
SetWord
TDCConfiguration, 101
SetupParityError
HPTDC chip control, 11
Socket, 41
∼Socket, 43
AcceptConnections, 43
Bind, 43
Configure, 43
Create, 44
DumpConnected, 44
fAddress, 46
fBuffer, 46
fMaster, 46
fPort, 46
fReadFds, 46
fSocketId, 46
fSocketsConnected, 46
FetchMessage, 44
GetPort, 44
GetSocketId, 44
GetSocketType, 44
IsWebSocket, 44
Listen, 44
PrepareConnection, 45
SelectConnections, 45
SendMessage, 45
SetPort, 45
SetSocketId, 45
Socket, 43
SocketCollection, 43
Start, 45
Stop, 46
Socket communication objects, 7
CLIENT, 7
DETECTOR, 8
INVALID, 7
MASTER, 7
SocketType, 7
WEBSOCKET_CLIENT, 7
SocketCollection
Socket, 43
SocketMessage, 47
∼SocketMessage, 51
Dump, 51
fMessage, 53
GetIntValue, 51
GetKey, 51
GetString, 52
GetValue, 52
GetVectorValue, 52
Object, 52
SetKeyValue, 52, 53
SocketMessage, 49–51
String, 53
SocketType
Socket communication objects, 7
spill_id
file_header_t, 23
Start
Socket, 45
Stop
Socket, 46
String
SocketMessage, 53
SwitchClientType
Messenger, 40

TDCConfiguration, 53
∼TDCConfiguration, 62
bit, 60
Dump, 62
fWord, 101
GetBits, 63
GetChannelOffset, 63
GetCoarseCountOffset, 64
GetDLLAdjustment, 64
GetDeadTime, 64
GetEdgeResolution, 65
GetEdgesPairing, 65
GetEnableError, 65
GetEnableErrorBypass, 65
GetEnableErrorMark, 66
GetEnableJTAGReadout, 66
GetEnableReadoutOccupancy, 66
GetEnableReadoutSeparator, 67
GetEnableSerial, 67
GetLeadingMode, 67
GetMatchWindow, 68
GetMaxEventSize, 68

GetNumWords, 68
GetRCAdjustment, 68
GetReadoutFIFOSize, 69
GetRejectCountOffset, 69
GetRejectFIFOFull, 69
GetSearchWindow, 70
GetSetupParity, 70
GetTestInvert, 70
GetTestMode, 71
GetTrailingMode, 71
GetTriggerCountOffset, 71
GetTriggerLatency, 72
GetTriggerMatchingMode, 72
GetVernierOffset, 72
GetWidthResolution, 73
GetWord, 73
kCoarseCountOffset, 101
kCoreClockDelay, 101
kCoreClockSource, 101
kDLLClockDelay, 101
kDLLClockSource, 101
kDLLControl, 101
kDLLMode, 101
kDLLTapAdjust0, 101
kDeadTime, 101
kEnableAutomaticReject, 101
kEnableBytewise, 101
kEnableDirectBunchReset, 101
kEnableDirectEventReset, 101
kEnableDirectTrigger, 101
kEnableError, 101
kEnableErrorBypass, 101
kEnableErrorMark, 101
kEnableGlobalHeader, 101
kEnableGlobalTrailer, 102
kEnableJTAGReadout, 102
kEnableLocalHeader, 102
kEnableLocalTrailer, 102
kEnableMasterResetCode, 102
kEnableMasterResetOnEventReset, 102
kEnableMatching, 102
kEnableOverflowDetect, 102
kEnablePair, 102
kEnableReadoutOccupancy, 102
kEnableReadoutSeparator, 102
kEnableRelative, 102
kEnableResetChannelBufferWhenSeparator, 102
kEnableSeparatorOnBunchReset, 102
kEnableSeparatorOnEventReset, 102
kEnableSerial, 102
kEnableSetCountersOnBunchReset, 102
kEnableTTLClock, 102
kEnableTTLControl, 102
kEnableTTLHit, 102
kEnableTTLReset, 102
kEnableTTLSerial, 102
kEventCountOffset, 102
kIOClockDelay, 102
kIOClockSource, 102
kKeepToken, 102
kLeading, 102
kLeadingResolution, 102
kLowPowerMode, 103
kMaster, 103
kMatchWindow, 103
kMaxEventSize, 103
kModeRC, 103
kModeRCCompression, 103
kNumWords, 103
kOffset0, 103
kPLLControl, 103
kRCAdjust0, 103
kReadoutFIFOSize, 103
kReadoutSingleCycleSpeed, 103
kReadoutSpeedSelect, 103
kRejectCountOffset, 103
kRejectFIFOFull, 103
kRollOver, 103
kSearchWindow, 103
kSelectBypassInputs, 103
kSerialClockDelay, 103
kSerialClockSource, 103
kSerialDelay, 103
kSetupParity, 103
kStrobeSelect, 103
kTDCId, 103
kTestInvert, 103
kTestMode, 103
kTestSelect, 103
kTokenDelay, 103
kTrailing, 104
kTriggerCountOffset, 104
kVernierOffset, 104
kWidthSelect, 104
SetAllChannelsOffset, 73
SetAllTapsDLLAdjustment, 73
SetBits, 74
SetBypassInputs, 74
SetChannelOffset, 74
SetCoarseCountOffset, 74
SetConstantValues, 75
SetCoreClockDelay, 76
SetCoreClockSource, 77
SetDLLAdjustment, 77
SetDLLClockDelay, 78
SetDLLClockSource, 78
SetDLLControl, 78
SetDLLMode, 79
SetDeadTime, 77
SetEdgeResolution, 79
SetEdgesPairing, 79
SetEnableAutomaticReject, 80
SetEnableBytewise, 80
SetEnableDirectBunchReset, 80
SetEnableDirectEventReset, 81
SetEnableDirectTrigger, 81

SetEnableError, 81
SetEnableErrorBypass, 82
SetEnableErrorMark, 82
SetEnableGlobalHeader, 82
SetEnableGlobalTrailer, 83
SetEnableJTAGReadout, 83
SetEnableLocalHeader, 83
SetEnableLocalTrailer, 84
SetEnableMasterResetCode, 84
SetEnableMasterResetOnEventReset, 84
SetEnableOverflowDetect, 85
SetEnableReadoutOccupancy, 85
SetEnableReadoutSeparator, 85
SetEnableRelative, 86
SetEnableResetChannelBufferWhenSeparator, 86
SetEnableSeparatorOnBunchReset, 86
SetEnableSeparatorOnEventReset, 87
SetEnableSerial, 87
SetEnableSetCountersOnBunchReset, 87
SetEnableTTLClock, 88
SetEnableTTLControl, 88
SetEnableTTLHit, 88
SetEnableTTLReset, 89
SetEnableTTLSerial, 89
SetEventCountOffset, 90
SetIOClockDelay, 90
SetIOClockSource, 90
SetKeepToken, 91
SetLeadingMode, 91
SetLowPowerMode, 91
SetMaster, 92
SetMatchWindow, 92
SetMaxEventSize, 92
SetModeRC, 93
SetModeRCCompression, 93
SetPLLControl, 93
SetRCAdjustment, 94
SetReadoutFIFOSize, 94
SetReadoutSingleCycleSpeed, 94
SetReadoutSpeedSelect, 95
SetRejectCountOffset, 95
SetRejectFIFOFull, 95
SetRollOver, 96
SetSearchWindow, 96
SetSerialClockDelay, 96
SetSerialClockSource, 97
SetSerialDelay, 97
SetSetupParity, 97
SetStrobeSelect, 98
SetTestInvert, 98
SetTestMode, 98
SetTokenDelay, 99
SetTrailingMode, 99
SetTriggerCountOffset, 99
SetTriggerMatchingMode, 100
SetVernierOffset, 100
SetWidthResolution, 100
SetWord, 101
TDCConfiguration, 61
word_t, 60
TDCEvent, 104
~TDCEvent, 105
fWord, 107
GetBunchId, 105
GetErrorFlags, 105
GetEventId, 106
GetLeadingTime, 106
GetTDCId, 106
GetTrailingTime, 106
GetType, 106
GetWidth, 107
GetWordCount, 107
TDCEvent, 105
TDCHeader
HPTDC chip control, 11
TDCTrailer
HPTDC chip control, 11
TrailingEdge
HPTDC chip control, 11
TriggerFIFOParityError
HPTDC chip control, 11
TriggerMatchingError
HPTDC chip control, 11
Type
Exception, 22
type
ListenerInfo, 31
TypeString
Exception, 22

USBHandler, 107
~USBHandler, 108
DumpDevice, 109
fDevice, 109
fHandle, 109
Fetch, 109
Init, 109
USBHandler, 108
Write, 109

VernierError
HPTDC chip control, 11

W_100ns
HPTDC chip control, 13
W_100ps
HPTDC chip control, 12
W_12p5ns
HPTDC chip control, 12
W_1p6ns
HPTDC chip control, 12
W_200ns
HPTDC chip control, 13
W_200ps
HPTDC chip control, 12
W_25ns
HPTDC chip control, 13

W_3p2ns
    HPTDC chip control, 12
W_400ns
    HPTDC chip control, 13
W_400ps
    HPTDC chip control, 12
W_50ns
    HPTDC chip control, 13
W_6p25ns
    HPTDC chip control, 12
W_800ns
    HPTDC chip control, 13
W_800ps
    HPTDC chip control, 12
WEBSOCKET_CLIENT
    Socket communication objects, 7
WidthResolution
    HPTDC chip control, 12
word_t
    TDCConfiguration, 60
Write
    USBHandler, 109