2015 Test beam Run Control

Generated by Doxygen 1.6.1

Mon Jul 13 17:02:07 2015

Contents

Chapter 1

Module Index

1.1 Modules

Here is a list of all modules:

Chapter 2

Namespace Index

2.1 Namespace List

Here is a list of all namespaces with brief descriptions:

Chapter 3

Data Structure Index

3.1 Class Hierarchy

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

Chapter 4

Data Structure Index

4.1 Data Structures

Here are the data structures with brief descriptions:

Chapter 5

Module Documentation

5.1 Socket communication objects

Data Structures

- class Client

  Base client object for the socket.

- class HTTPMessage

  Message to be transmitted through a WebSocket protocol.

- 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.

Chapter 6

Namespace Documentation

6.1 VME Namespace Reference

Namespaces

- namespace TDCV1x90Opcodes

Data Structures

- class BridgeVx718Status
- class BridgeVx718Control
- class BridgeVx718

  class defining the VME bridge

- class FPGAUnitV1495Control
- class FPGAUnitV1495
- class GenericBoard
- class IOModuleV262
- class TDCErrorFlag

  Error flags handler.

- class TDCEvent

  HPTDC event parser.

- class TDCMeasurement
- struct GlobalOffset
- struct trailead_t
- class TDCV1x90Status

  TDC status register.

- class TDCV1x90Control

TDC control register.

- class TDCV1x90

Typedefs

- typedef std::vector< TDCEvent > TDCEventCollection

Enumerations

- enum BridgeType { CAEN_V1718, CAEN_V2718 }

  Compatible bridge types.

- enum FPGAUnitV1495Register {

  kV1495ScalerCounter = 0x100c, kV1495UserFWRevision = 0x1014, kV1495TDCBoardInterface = 0x1018, kV1495ClockSettings = 0x101c,

  kV1495Control = 0x1020, kV1495TriggerSettings = 0x1024, kV1495OutputSettings = 0x1028, kV1495GeoAddress = 0x8008,

  kV1495UserFPGAFlashMem = 0x8014, kV1495UserFPGAConfig = 0x8016, kV1495ModuleReset = 0x800a, kV1495FWRevision = 0x800c,

  kV1495ConfigurationROM = 0x8100, kV1495OUI2 = 0x8124, kV1495OUI1 = 0x8128, kV1495OUI0 = 0x812c,

  kV1495Board2 = 0x8134, kV1495Board1 = 0x8138, kV1495Board0 = 0x813c, kV1495HWRevision3 = 0x8140,

  kV1495HWRevision2 = 0x8144, kV1495HWRevision1 = 0x8148, kV1495HWRevision0 = 0x814c, kV1495SerNum0 = 0x8180,

  kV1495SerNum1 = 0x8184 }
- enum IOModuleV262Register {

  kECLLevelWrite = 0x04, kNIMLevelWrite = 0x06, kNIMPulseWrite = 0x08, kNIMPulseRead = 0x0a,

  kIdentifier = 0xfa, kBoardInfo0 = 0xfc, kBoardInfo1 = 0xfe }
- enum AcquisitionMode { CONT_STORAGE, TRIG_MATCH }

  TDC acquisition mode.

- enum DetectionMode { PAIR = 0x0, OTRAILING = 0x1, OLEADING = 0x2, TRAILEAD = 0x3 }
- enum trig_conf {

  MATCH_WIN_WIDTH = 0, WIN_OFFSET = 1, EXTRA_SEARCH_WIN_WIDTH = 2, REJECT_MARGIN = 3,

  TRIG_TIME_SUB = 4 }
- enum trailead_edge_lsb { r800ps = 0, r200ps = 1, r100ps = 2, r25ps = 3 }
- enum micro_handshake { WRITE_OK = 0, READ_OK = 1 }

- enum TDCV1x90Register {

  kOutputBuffer = 0x0000, kControl = 0x1000, kStatus = 0x1002, kInterruptLevel = 0x100a,

  kInterruptVector = 0x100c, kGeoAddress = 0x100e, kMCSTBase = 0x1010, kMCSTControl = 0x1012,

  kModuleReset = 0x1014, kSoftwareClear = 0x1016, kEventCounter = 0x101c, kEventStored = 0x1020,

  kBLTEventNumber = 0x1024, kFirmwareRev = 0x1026, kMicro = 0x102e, kMicroHandshake = 0x1030,

  kEventFIFO = 0x1038, kEventFIFOStoredRegister = 0x103c, kEventFIFOStatusRegister = 0x103e, kROMOui2 = 0x4024,

  kROMOui1 = 0x4028, kROMOui0 = 0x402c, kROMBoard2 = 0x4034, kROMBoard1 = 0x4038,

  kROMBoard0 = 0x403c, kROMRevis3 = 0x4040, kROMRevis2 = 0x4044, kROMRevis1 = 0x4048,

  kROMRevis0 = 0x404c, kROMSerNum1 = 0x4080, kROMSerNum0 = 0x4084 }

6.1.1 Typedef Documentation

6.1.1.1 typedef std::vector<TDCEvent> VME::TDCEventCollection

6.1.2 Enumeration Type Documentation

6.1.2.1 enum VME::AcquisitionMode

TDC acquisition mode.

Author:

Laurent Forthomme <laurent.forthomme@cern.ch>

Enumerator:

CONT_STORAGE

TRIG_MATCH

6.1.2.2 enum VME::BridgeType

Compatible bridge types.

Enumerator:

CAEN_V1718

CAEN_V2718

6.1.2.3 enum VME::DetectionMode

Enumerator:

- *PAIR*
- *OTRAILING*
- *OLEADING*
- *TRAILEAD*

6.1.2.4 enum VME::FPGAUnitV1495Register

Enumerator:

- *kV1495ScalerCounter*
- *kV1495UserFWRevision*
- *kV1495TDCBoardInterface*
- *kV1495ClockSettings*
- *kV1495Control*
- *kV1495TriggerSettings*
- *kV1495OutputSettings*
- *kV1495GeoAddress*
- *kV1495UserFPGAFlashMem*
- *kV1495UserFPGAConfig*
- *kV1495ModuleReset*
- *kV1495FWRevision*
- *kV1495ConfigurationROM*
- *kV1495OUI2*
- *kV1495OUI1*
- *kV1495OUI0*
- *kV1495Board2*
- *kV1495Board1*
- *kV1495Board0*
- *kV1495HWRevision3*
- *kV1495HWRevision2*
- *kV1495HWRevision1*
- *kV1495HWRevision0*
- *kV1495SerNum0*
- *kV1495SerNum1*

6.1.2.5 enum VME::IOModuleV262Register

Enumerator:

kECLLevelWrite

kNIMLevelWrite

kNIMPulseWrite

kNIMPulseRead

kIdentifier

kBoardInfo0

kBoardInfo1

6.1.2.6 enum VME::micro_handshake

Enumerator:

WRITE_OK Is the TDC ready for writing?

READ_OK Is the TDC ready for reading?

6.1.2.7 enum VME::TDCV1x90Register

Enumerator:

kOutputBuffer

kControl

kStatus

kInterruptLevel

kInterruptVector

kGeoAddress

kMCSTBase

kMCSTControl

kModuleReset

kSoftwareClear

kEventCounter

kEventStored

kBLTEventNumber

kFirmwareRev

kMicro

kMicroHandshake

kEventFIFO

kEventFIFOStoredRegister

kEventFIFOStatusRegister

kROMOui2

kROMOui1

kROMOui0

kROMBoard2

kROMBoard1

kROMBoard0

kROMRevis3

kROMRevis2

kROMRevis1

kROMRevis0

kROMSerNum1

kROMSerNum0

6.1.2.8 enum VME::trailead_edge_lsb

Enumerator:

r800ps

r200ps

r100ps

r25ps

6.1.2.9 enum VME::trig_conf

Enumerator:

MATCH_WIN_WIDTH

WIN_OFFSET

EXTRA_SEARCH_WIN_WIDTH

REJECT_MARGIN

TRIG_TIME_SUB

6.2 VME::TDCV1x90Opcodes Namespace Reference

Functions

- Opcode TRG_MATCH (0x0000)
- Opcode CONT_STOR (0x0100)
- Opcode READ_ACQ_MOD (0x0200)
- Opcode SET_KEEP_TOKEN (0x0300)
- Opcode CLEAR_KEEP_TOKEN (0x0400)
- Opcode LOAD_DEF_CONFIG (0x0500)
- Opcode SAVE_USER_CONFIG (0x0600)
- Opcode LOAD_USER_CONFIG (0x0700)
- Opcode AUTOLOAD_USER_CONF (0x0800)
- Opcode AUTOLOAD_DEF_CONFI (0x0900)
- Opcode SET_WIN_WIDTH (0x1000)
- Opcode SET_WIN_OFFS (0x1100)
- Opcode SET_SW_MARGIN (0x1200)
- Opcode SET_REJ_MARGIN (0x1300)
- Opcode EN_SUB_TRG (0x1400)
- Opcode DIS_SUB_TRG (0x1500)
- Opcode READ_TRG_CONF (0x1600)
- Opcode SET_DETECTION (0x2200)
- Opcode READ_DETECTION (0x2300)
- Opcode SET_TR_LEAD_LSB (0x2400)
- Opcode SET_PAIR_RES (0x2500)
- Opcode READ_RES (0x2600)
- Opcode SET_DEAD_TIME (0x2800)
- Opcode READ_DEAD_TIME (0x2900)
- Opcode EN_HEAD_TRAILER (0x3000)
- Opcode DIS_HEAD_TRAILER (0x3100)
- Opcode READ_HEAD_TRAILER (0x3200)
- Opcode SET_EVENT_SIZE (0x3300)
- Opcode READ_EVENT_SIZE (0x3400)
- Opcode EN_ERROR_MARK (0x3500)
- Opcode DIS_ERROR_MARK (0x3600)
- Opcode EN_ERROR_BYPASS (0x3700)
- Opcode DIS_ERROR_BYPASS (0x3800)
- Opcode SET_ERROR_TYPES (0x3900)
- Opcode READ_ERROR_TYPES (0x3a00)
- Opcode SET_FIFO_SIZE (0x3b00)
- Opcode READ_FIFO_SIZE (0x3c00)
- Opcode EN_CHANNEL (0x4000)
- Opcode DIS_CHANNEL (0x4100)
- Opcode EN_ALL_CHANNEL (0x4200)
- Opcode DIS_ALL_CHANNEL (0x4300)
- Opcode WRITE_EN_PATTERN (0x4400)

- Opcode READ_EN_PATTERN (0x4500)
- Opcode WRITE_EN_PATTERN32 (0x4600)
- Opcode READ_EN_PATTERN32 (0x4700)
- Opcode SET_GLOB_OFFS (0x5000)
- Opcode READ_GLOB_OFFS (0x5100)
- Opcode SET_ADJUST_CH (0x5200)
- Opcode READ_ADJUST_CH (0x5200)
- Opcode SET_RC_ADJ (0x5400)
- Opcode READ_RC_ADJ (0x5500)
- Opcode SAVE_RC_ADJ (0x5600)
- Opcode READ_TDC_ID (0x6000)
- Opcode READ_MICRO_REV (0x6100)
- Opcode RESET_DLL_PLL (0x6200)
- Opcode WRITE_SETUP_REG (0x7000)
- Opcode READ_SETUP_REG (0x7100)
- Opcode UPDATE_SETUP_REG (0x7200)
- Opcode DEFAULT_SETUP_REG (0x7300)
- Opcode READ_ERROR_STATUS (0x7400)
- Opcode READ_DLL_LOCK (0x7500)
- Opcode READ_STATUS_STREAM (0x7600)
- Opcode UPDATE_SETUP_TDC (0x7700)
- Opcode WRITE_EEPROM (0xc000)
- Opcode READ_EEPROM (0xc100)
- Opcode REV_DATE_MICRO_FW (0xc200)
- Opcode WRITE_SPARE (0xc300)
- Opcode READ_SPARE (0xc400)
- Opcode ENABLE_TEST_MODE (0xc500)
- Opcode DISABLE_TEST_MODE (0xc600)
- Opcode SET_TDC_TSET_OUTPUT (0xc700)
- Opcode SET_DLL_CLOCK (0xc800)
- Opcode READ_SETUP_SCANPATH (0xc900)

6.2.1 Function Documentation

6.2.1.1 Opcode VME::TDCV1x90Opcodes::AUTOLOAD_DEF_CONFI (0x0900)

6.2.1.2 Opcode VME::TDCV1x90Opcodes::AUTOLOAD_USER_CONF (0x0800)

6.2.1.3 Opcode VME::TDCV1x90Opcodes::CLEAR_KEEP_TOKEN (0x0400)

6.2.1.4 Opcode VME::TDCV1x90Opcodes::CONT_STOR (0x0100)

6.2.1.5 Opcode VME::TDCV1x90Opcodes::DEFAULT_SETUP_REG (0x7300)

6.2.1.6 Opcode VME::TDCV1x90Opcodes::DIS_ALL_CHANNEL (0x4300)

6.2.1.7 Opcode VME::TDCV1x90Opcodes::DIS_CHANNEL (0x4100)

6.2.1.8 Opcode VME::TDCV1x90Opcodes::DIS_ERROR_BYPASS (0x3800)

6.2.1.9 Opcode VME::TDCV1x90Opcodes::DIS_ERROR_MARK (0x3600)

6.2.1.10 Opcode VME::TDCV1x90Opcodes::DIS_HEAD_TRAILER (0x3100)

6.2.1.11 Opcode VME::TDCV1x90Opcodes::DIS_SUB_TRG (0x1500)

6.2.1.12 Opcode VME::TDCV1x90Opcodes::DISABLE_TEST_MODE (0xc600)

6.2.1.13 Opcode VME::TDCV1x90Opcodes::EN_ALL_CHANNEL (0x4200)

6.2.1.14 Opcode VME::TDCV1x90Opcodes::EN_CHANNEL (0x4000)

6.2.1.15 Opcode VME::TDCV1x90Opcodes::EN_ERROR_BYPASS (0x3700)

6.2.1.16 Opcode VME::TDCV1x90Opcodes::EN_ERROR_MARK (0x3500)

6.2.1.17 Opcode VME::TDCV1x90Opcodes::EN_HEAD_TRAILER (0x3000)

6.2.1.18 Opcode VME::TDCV1x90Opcodes::EN_SUB_TRG (0x1400)

6.2.1.19 Opcode VME::TDCV1x90Opcodes::ENABLE_TEST_MODE (0xc500)

6.2.1.20 Opcode VME::TDCV1x90Opcodes::LOAD_DEF_CONFIG (0x0500)

6.2.1.21 Opcode VME::TDCV1x90Opcodes::LOAD_USER_CONFIG (0x0700)

6.2.1.22 Opcode VME::TDCV1x90Opcodes::READ_ACQ_MOD (0x0200)

6.2.1.23 Opcode VME::TDCV1x90Opcodes::READ_ADJUST_CH (0x5200)

6.2.1.24 Opcode VME::TDCV1x90Opcodes::READ_DEAD_TIME (0x2900)

6.2.1.25 Opcode VME::TDCV1x90Opcodes::READ_DETECTION (0x2300)

6.2.1.26 Opcode VME::TDCV1x90Opcodes::READ_DLL_LOCK (0x7500)

6.2.1.27 Opcode VME::TDCV1x90Opcodes::READ_EEPROM (0xc100)

Chapter 7

Data Structure Documentation

7.1 VME::BridgeVx718 Class Reference

class defining the VME bridge

`#include <VME_BridgeVx718.h>`Inheritance diagram for VME::BridgeVx718:Collaboration diagram for VME::BridgeVx718:

Public Types

- enum IRQId {
  IRQ1 = 0x1, IRQ2 = 0x2, IRQ3 = 0x4, IRQ4 = 0x8,
  IRQ5 = 0x10, IRQ6 = 0x20, IRQ7 = 0x40 }

Public Member Functions

- BridgeVx718 (const char ∗device, BridgeType type)

  Constructor.

- ∼BridgeVx718 ()

  Destructor.

- int32_t GetHandle () const

  Bridge's handle value.

- void CheckConfiguration () const
- void TestOutputs () const
- void Reset () const

  Perform a system reset of the module.

- BridgeVx718Status GetStatus () const

- void SetIRQ (unsigned int irq, bool enable=true)
- void WaitIRQ (unsigned int irq, unsigned long timeout=1000) const
- unsigned int GetIRQStatus () const
- void OutputConf (CVOutputSelect output) const

  Set and control the output lines.

- void OutputOn (unsigned short output) const
- void OutputOff (unsigned short output) const
- void InputConf (CVInputSelect input) const

  Set and read the input lines.

- void InputRead (CVInputSelect input) const
- void StartPulser (double period, double width, unsigned int num_pulses=0) const
- void StopPulser () const
- void SinglePulse (unsigned short channel) const

Private Attributes

- bool fHasIRQ

7.1.1 Detailed Description

class defining the VME bridge This class initializes the CAEN V1718 VME bridge in order to control the crate.

Author:

Laurent Forthomme <`laurent.forthomme@cern.ch`>
Bob Velghe <`bob.velghe@cern.ch`>

Date:

Jun 2010

7.1.2 Member Enumeration Documentation

7.1.2.1 enum VME::BridgeVx718::IRQId

Enumerator:

IRQ1

IRQ2

IRQ3

IRQ4

IRQ5

IRQ6

IRQ7

7.1.3 Constructor & Destructor Documentation

7.1.3.1 VME::BridgeVx718::BridgeVx718 (const char ∗ *device*, BridgeType *type*)

Constructor. Bridge class constructor

Parameters:

← ***device*** Device identifier on the VME crate

← ***type*** Device type (1718/2718)

Here is the call graph for this function:

7.1.3.2 VME::BridgeVx718::∼BridgeVx718 ()

Destructor. Bridge class destructor

7.1.4 Member Function Documentation

7.1.4.1 void VME::BridgeVx718::CheckConfiguration () const

7.1.4.2 int32_t VME::BridgeVx718::GetHandle () const `[inline]`

Bridge's handle value.

Returns:

Handle value

7.1.4.3 unsigned int VME::BridgeVx718::GetIRQStatus () const

7.1.4.4 BridgeVx718Status VME::BridgeVx718::GetStatus () const

7.1.4.5 void VME::BridgeVx718::InputConf (CVInputSelect *input*) const

Set and read the input lines.

7.1.4.6 void VME::BridgeVx718::InputRead (CVInputSelect *input*) const

7.1.4.7 void VME::BridgeVx718::OutputConf (CVOutputSelect *output*) const

Set and control the output lines.

7.1.4.8 void VME::BridgeVx718::OutputOff (unsigned short *output*) const

7.1.4.9 void VME::BridgeVx718::OutputOn (unsigned short *output*) const

7.1.4.10 void VME::BridgeVx718::Reset () const

Perform a system reset of the module.

7.1.4.11 void VME::BridgeVx718::SetIRQ (unsigned int *irq*, bool *enable* = `true`)

7.1.4.12 void VME::BridgeVx718::SinglePulse (unsigned short *channel*) const

Here is the call graph for this function:

7.1.4.13 void VME::BridgeVx718::StartPulser (double *period*, double *width*, unsigned int *num_pulses* = `0`) const

Here is the call graph for this function:

7.1.4.14 void VME::BridgeVx718::StopPulser () const

7.1.4.15 void VME::BridgeVx718::TestOutputs () const

Here is the call graph for this function:

7.1.4.16 void VME::BridgeVx718::WaitIRQ (unsigned int *irq*, unsigned long *timeout* = `1000`) const

7.1.5 Field Documentation

7.1.5.1 bool VME::BridgeVx718::fHasIRQ `[private]`

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

- include/VME_BridgeVx718.h
- src/VME_BridgeVx718.cpp

7.2 VME::BridgeVx718Control Class Reference

```
#include <VME_BridgeVx718.h>
```

Public Member Functions

- BridgeVx718Control (uint16_t word)
- virtual ∼BridgeVx718Control ()
- bool GetArbiterType () const

  Arbiter type.

- bool GetRequesterType () const

  Requester type.

- bool GetReleaseType () const

  Release type.

- unsigned int GetBusReqLevel () const
- bool GetInterruptReq () const
- bool GetSysRes () const
- bool GetBusTimeout () const

  VME bus timeout.

- bool GetAddressIncrement () const

  Address Increment.

Private Attributes

- uint16_t fWord

7.2.1 Constructor & Destructor Documentation

7.2.1.1 VME::BridgeVx718Control::BridgeVx718Control (uint16_t *word*) `[inline]`

7.2.1.2 virtual VME::BridgeVx718Control::∼BridgeVx718Control () `[inline, virtual]`

7.2.2 Member Function Documentation

7.2.2.1 bool VME::BridgeVx718Control::GetAddressIncrement () const `[inline]`

Address Increment.

Returns:

true if enabled, else false (FIFO mode)

7.2.2.2 bool VME::BridgeVx718Control::GetArbiterType () const `[inline]`

Arbiter type.

Returns:

true if "Round Robin", else fixed priority

7.2.2.3 unsigned int VME::BridgeVx718Control::GetBusReqLevel () const `[inline]`

7.2.2.4 bool VME::BridgeVx718Control::GetBusTimeout () const `[inline]`

VME bus timeout.

Returns:

true if 1400 us, else 50 us

7.2.2.5 bool VME::BridgeVx718Control::GetInterruptReq () const `[inline]`

7.2.2.6 bool VME::BridgeVx718Control::GetReleaseType () const `[inline]`

Release type.

Returns:

true if release on request, else release when done

7.2.2.7 bool VME::BridgeVx718Control::GetRequesterType () const `[inline]`

Requester type.

Returns:

true if demand, else fair

7.2.2.8 bool VME::BridgeVx718Control::GetSysRes () const `[inline]`

7.2.3 Field Documentation

7.2.3.1 uint16_t VME::BridgeVx718Control::fWord `[private]`

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

- include/VME_BridgeVx718.h

7.3 VME::BridgeVx718Status Class Reference

```
#include <VME_BridgeVx718.h>
```

Public Member Functions

- BridgeVx718Status (uint16_t word)
- virtual ∼BridgeVx718Status ()
- void Dump () const
- bool GetSystemReset () const
- bool GetSystemControl () const
- bool GetDTACK () const
- bool GetBERR () const
- bool GetDipSwitch (unsigned int sw) const
- bool GetUSBType () const

Private Attributes

- uint16_t fWord

7.3.1 Constructor & Destructor Documentation

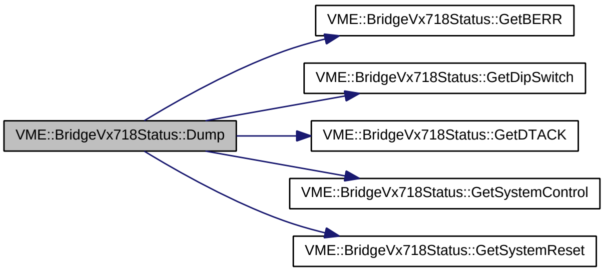
7.3.1.1 VME::BridgeVx718Status::BridgeVx718Status (uint16_t *word*) `[inline]`

7.3.1.2 virtual VME::BridgeVx718Status::∼BridgeVx718Status () `[inline, virtual]`

7.3.2 Member Function Documentation

7.3.2.1 void VME::BridgeVx718Status::Dump () const `[inline]`

Here is the call graph for this function:


VME::BridgeVx718Status::Dump
VME::BridgeVx718Status::GetBERR
VME::BridgeVx718Status::GetDipSwitch
VME::BridgeVx718Status::GetDTACK
VME::BridgeVx718Status::GetSystemControl
VME::BridgeVx718Status::GetSystemReset

7.3.2.2 bool VME::BridgeVx718Status::GetBERR () const `[inline]`

7.3.2.3 bool VME::BridgeVx718Status::GetDipSwitch (unsigned int *sw*) const `[inline]`

7.3.2.4 bool VME::BridgeVx718Status::GetDTACK () const `[inline]`

7.3.2.5 bool VME::BridgeVx718Status::GetSystemControl () const `[inline]`

7.3.2.6 bool VME::BridgeVx718Status::GetSystemReset () const `[inline]`

7.3.2.7 bool VME::BridgeVx718Status::GetUSBType () const `[inline]`

7.3.3 Field Documentation

7.3.3.1 uint16_t VME::BridgeVx718Status::fWord `[private]`

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

- include/VME_BridgeVx718.h

7.4 Client Class Reference

Base client object for the socket.

`#include <Client.h>`Inheritance diagram for Client:Collaboration diagram for Client:

Public Member Functions

- Client ()

  General void client constructor.

- Client (int port)

  Bind a socket client to a given port.

- virtual ∼Client ()
- bool Connect (const SocketType &type=CLIENT)

  Bind this client to the socket.

- void Disconnect ()

  Unbind this client from the socket.

- void Send (const Message &m) const

  Send a message to the master through the socket.

- void Send (const Exception &e) const
- SocketMessage SendAndReceive (const SocketMessage &m, const MessageKey &a) const
- void Receive ()

  Receive a socket message from the master.

- virtual void ParseMessage (const SocketMessage &m)

  Parse a SocketMessage received from the master.

- virtual SocketType GetType () const

  Socket actor type retrieval method.

Private Member Functions

- void Announce ()

  Announce our entry on the socket to its master.

Private Attributes

- int fClientId
- bool fIsConnected
- SocketType fType

7.4.1 Detailed Description

Base client object for the socket. Client object used by the server to send/receive commands from the messenger/broadcaster.

Author:

Laurent Forthomme <`laurent.forthomme@cern.ch`>

Date:

24 Mar 2015

7.4.2 Constructor & Destructor Documentation

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

General void client constructor.

7.4.2.2 Client::Client (int *port*)

Bind a socket client to a given port.

7.4.2.3 Client::∼Client () `[virtual]`

Here is the call graph for this function:

7.4.3 Member Function Documentation

7.4.3.1 void Client::Announce () `[private]`

Announce our entry on the socket to its master.

Here is the call graph for this function:

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

Bind this client to the socket.

Here is the call graph for this function:

7.4.3.3 void Client::Disconnect ()

Unbind this client from the socket.

Here is the call graph for this function:

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

Socket actor type retrieval method.

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

Parse a SocketMessage received from the master.

7.4.3.6 void Client::Receive ()

Receive a socket message from the master.

Here is the call graph for this function:

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

Here is the call graph for this function:

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

Send a message to the master through the socket.

Here is the call graph for this function:

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

Here is the call graph for this function:

7.4.4 Field Documentation

7.4.4.1 int Client::fClientId `[private]`

7.4.4.2 bool Client::fIsConnected `[private]`

7.4.4.3 SocketType Client::fType `[private]`

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

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

7.5 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
- std::string OneLine () const

Private Attributes

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

7.5.1 Detailed Description

A simple exception handler.

Author:

Laurent Forthomme <`laurent.forthomme@cern.ch`>

Date:

24 Mar 2015

7.5.2 Constructor & Destructor Documentation

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

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

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

Here is the call graph for this function:

7.5.3 Member Function Documentation

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

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

Here is the call graph for this function:

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

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

7.5.3.5 std::string Exception::OneLine () const `[inline]`

Here is the call graph for this function:

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

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

Here is the call graph for this function:

7.5.4 Field Documentation

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

7.5.4.2 int Exception::fErrorNumber `[private]`

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

7.5.4.4 ExceptionType Exception::fType `[private]`

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

- include/Exception.h

7.6 file_header_t Struct Reference

Header to the output files.

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

Data Fields

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

7.6.1 Detailed Description

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

Author:

Laurent Forthomme <laurent.forthomme@cern.ch>

Date:

14 Apr 2015

7.6.2 Field Documentation

7.6.2.1 VME::AcquisitionMode file_header_t::acq_mode

7.6.2.2 VME::DetectionMode file_header_t::det_mode

7.6.2.3 uint32_t file_header_t::magic

7.6.2.4 uint8_t file_header_t::num_hptdc

7.6.2.5 uint32_t file_header_t::run_id

7.6.2.6 uint32_t file_header_t::spill_id

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

- include/FileConstants.h

7.7 FileReader Class Reference

Handler for a TDC output file readout.

`#include <FileReader.h>`Collaboration diagram for FileReader:

Public Member Functions

- FileReader (std::string name)

  Class constructor.

- ∼FileReader ()
- unsigned int GetNumTDCs () const
- unsigned long GetNumEvents () const
- bool GetNextEvent (VME::TDCEvent ∗)
- bool GetNextMeasurement (unsigned int channel_id, VME::TDCMeasurement ∗mc)

  Fetch the next full measurement on a given channel.

Private Attributes

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

7.7.1 Detailed Description

Handler for a TDC output file readout.

Author:

Laurent Forthomme <`laurent.forthomme@cern.ch`>

Date:

Jun 2015

7.7.2 Constructor & Destructor Documentation

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

Class constructor.

Parameters:

← ***name*** Path to the file to read

← ***ro*** Data readout mode (continuous storage or trigger matching)

7.7.2.2 FileReader::∼FileReader ()

7.7.3 Member Function Documentation

7.7.3.1 bool FileReader::GetNextEvent (VME::TDCEvent ∗ *ev*)

Here is the call graph for this function:

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

Fetch the next full measurement on a given channel.

Parameters:

← ***channel_id*** Unique identifier of the channel number to retrieve

→ ***m*** A full measurement with leading, trailing times, ...

Returns:

A boolean stating the success of retrieval operation

Here is the call graph for this function:

7.7.3.3 unsigned long FileReader::GetNumEvents () const `[inline]`

7.7.3.4 unsigned int FileReader::GetNumTDCs () const `[inline]`

7.7.4 Field Documentation

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

7.7.4.2 file_header_t FileReader::fHeader `[private]`

7.7.4.3 unsigned long FileReader::fNumEvents `[private]`

7.7.4.4 VME::AcquisitionMode FileReader::fReadoutMode `[private]`

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

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

7.8 VME::FPGAUnitV1495 Class Reference

`#include <VME_FPGAUnitV1495.h>`Inheritance diagram for VME::FPGAUnitV1495:Collaboration diagram for VME::FPGAUnitV1495:

Public Types

- enum TDCBits { kReset = 0x1, kTrigger = 0x2, kClear = 0x4 }

Public Member Functions

- FPGAUnitV1495 (int32_t bhandle, uint32_t baseaddr)
- ~FPGAUnitV1495 ()
- unsigned short GetCAENFirmwareRevision () const
- unsigned short GetUserFirmwareRevision () const
- unsigned int GetHardwareRevision () const
- unsigned short GetSerialNumber () const
- unsigned short GetGeoAddress () const
- void CheckBoardVersion () const
- void ResetFPGA () const
- void DumpFWInformation () const
- void SetTDCBits (unsigned short bits) const

  Set a pattern of bits to be sent to all TDCs through the ECL mezzanine.

- void PulseTDCBits (unsigned short bits, unsigned int time_us=10) const

  Send a pulse to TDCs' front panel.

- unsigned short GetTDCBits () const

  Retrieve the current bits sent to TDCs' front panel.

- FPGAUnitV1495Control GetControl () const

  Retrieve the user-defined control word.

- void SetControl (const FPGAUnitV1495Control &control) const

  Set the user-defined control word.

- void SetInternalClockPeriod (uint32_t period) const

  Set the internal clock period.

- uint32_t GetInternalClockPeriod () const

  Retrieve the internal clock period.

- void SetInternalTriggerPeriod (uint32_t period) const

  Set the internal trigger period.

- uint32_t GetInternalTriggerPeriod () const

  Retrieve the internal trigger period.

- uint32_t GetOutputPulser () const
- void ClearOutputPulser () const
- void SetOutputPulser (unsigned short id, bool enable=true) const
- void StartScaler ()

  Start the inner triggers counter.

- void StopScaler ()

  Stop the inner triggers counter.

- uint32_t GetScalerValue () const

  Return the inner triggers counter value.

Private Attributes

- bool fScalerStarted

7.8.1 Detailed Description

Handler for the multi-purposes FPGA unit (CAEN V1495)

Author:

Laurent Forthomme <laurent.forthomme@cern.ch>

Date:

25 Jun 2015

7.8.2 Member Enumeration Documentation

7.8.2.1 enum VME::FPGAUnitV1495::TDCBits

Enumerator:

kReset

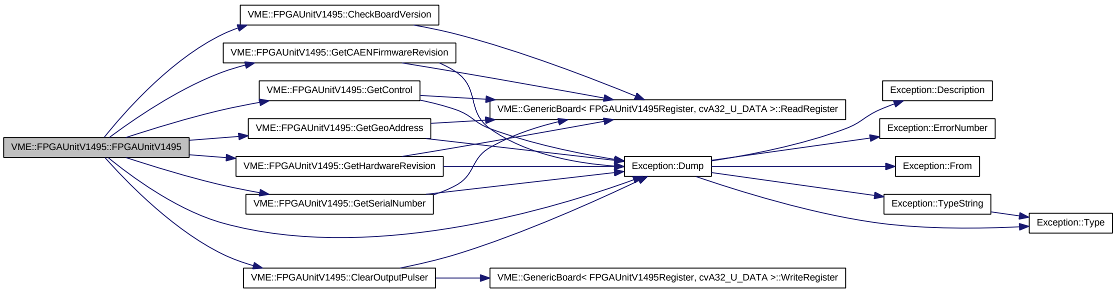
kTrigger

kClear

7.8.3 Constructor & Destructor Documentation

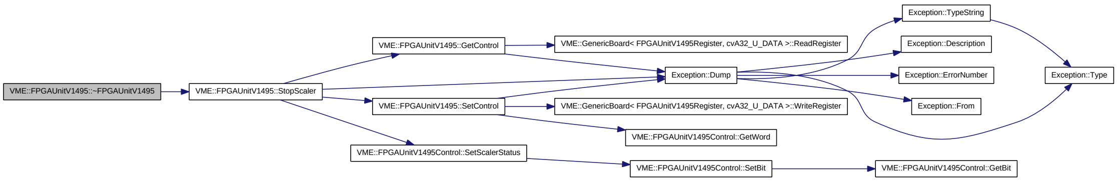
7.8.3.1 VME::FPGAUnitV1495::FPGAUnitV1495 (int32_t *bhandle*, uint32_t *baseaddr*)

Here is the call graph for this function:


VME::FPGAUnitV1495::CheckBoardVersion
VME::FPGAUnitV1495::GetCAENFirmwareRevision
VME::FPGAUnitV1495::GetControl
VME::GenericBoard< FPGAUnitV1495Register, cvA32_U_DATA >::ReadRegister
Exception::Description
VME::FPGAUnitV1495::GetGeoAddress
Exception::ErrorNumber
VME::FPGAUnitV1495::FPGAUnitV1495
VME::FPGAUnitV1495::GetHardwareRevision
Exception::Dump
Exception::From
VME::FPGAUnitV1495::GetSerialNumber
Exception::TypeString
Exception::Type
VME::FPGAUnitV1495::ClearOutputPulser
VME::GenericBoard< FPGAUnitV1495Register, cvA32_U_DATA >::WriteRegister

7.8.3.2 VME::FPGAUnitV1495::∼FPGAUnitV1495 ()

Here is the call graph for this function:


Exception::TypeString
VME::FPGAUnitV1495::GetControl
VME::GenericBoard< FPGAUnitV1495Register, cvA32_U_DATA >::ReadRegister
Exception::Description
Exception::Dump
Exception::ErrorNumber
Exception::Type
VME::FPGAUnitV1495::~FPGAUnitV1495
VME::FPGAUnitV1495::StopScaler
VME::FPGAUnitV1495::SetControl
VME::GenericBoard< FPGAUnitV1495Register, cvA32_U_DATA >::WriteRegister
Exception::From
VME::FPGAUnitV1495Control::GetWord
VME::FPGAUnitV1495Control::SetScalerStatus
VME::FPGAUnitV1495Control::SetBit
VME::FPGAUnitV1495Control::GetBit

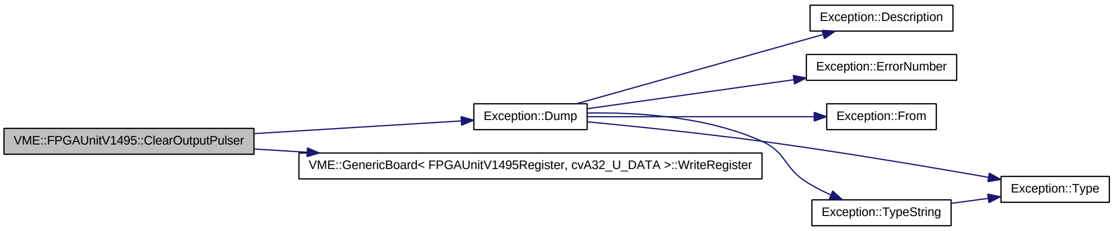
7.8.4 Member Function Documentation

7.8.4.1 void VME::FPGAUnitV1495::CheckBoardVersion () const

Here is the call graph for this function:

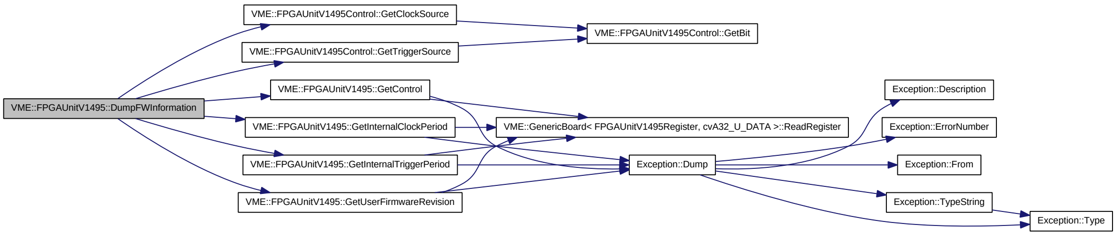
7.8.4.2 void VME::FPGAUnitV1495::ClearOutputPulser () const

Here is the call graph for this function:



7.8.4.3 void VME::FPGAUnitV1495::DumpFWInformation () const

Here is the call graph for this function:


VME::FPGAUnitV1495Control::GetClockSource
VME::FPGAUnitV1495Control::GetBit
VME::FPGAUnitV1495Control::GetTriggerSource
VME::FPGAUnitV1495::GetControl
Exception::Description
VME::FPGAUnitV1495::DumpFWInformation
VME::FPGAUnitV1495::GetInternalClockPeriod
VME::GenericBoard< FPGAUnitV1495Register, cvA32_U_DATA >::ReadRegister
Exception::ErrorNumber
VME::FPGAUnitV1495::GetInternalTriggerPeriod
Exception::Dump
Exception::From
VME::FPGAUnitV1495::GetUserFirmwareRevision
Exception::TypeString
Exception::Type

7.8.4.4 unsigned short VME::FPGAUnitV1495::GetCAENFirmwareRevision () const

Here is the call graph for this function:

7.8.4.5 FPGAUnitV1495Control VME::FPGAUnitV1495::GetControl () const

Retrieve the user-defined control word.

Here is the call graph for this function:

7.8.4.6 unsigned short VME::FPGAUnitV1495::GetGeoAddress () const

Here is the call graph for this function:

7.8.4.7 unsigned int VME::FPGAUnitV1495::GetHardwareRevision () const

Here is the call graph for this function:

7.8.4.8 uint32_t VME::FPGAUnitV1495::GetInternalClockPeriod () const

Retrieve the internal clock period.

Returns:

Clock period (in units of 25 ns)

Here is the call graph for this function:

7.8.4.9 uint32_t VME::FPGAUnitV1495::GetInternalTriggerPeriod () const

Retrieve the internal trigger period.

Returns:

Trigger period (in units of 50 ns)

Here is the call graph for this function:

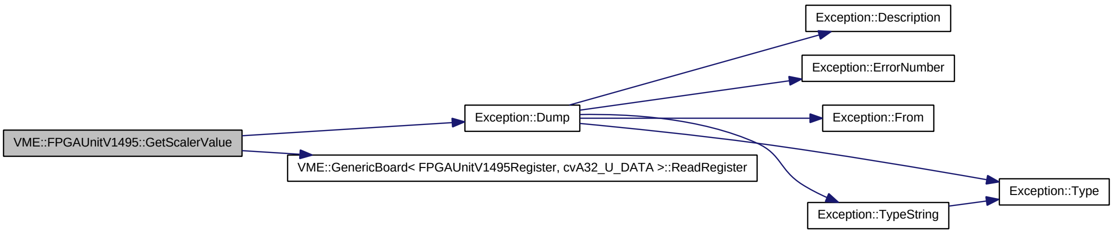
7.8.4.10 uint32_t VME::FPGAUnitV1495::GetOutputPulser () const

Here is the call graph for this function:

7.8.4.11 uint32_t VME::FPGAUnitV1495::GetScalerValue () const

Return the inner triggers counter value.

Here is the call graph for this function:



7.8.4.12 unsigned short VME::FPGAUnitV1495::GetSerialNumber () const

Here is the call graph for this function:

7.8.4.13 unsigned short VME::FPGAUnitV1495::GetTDCBits () const

Retrieve the current bits sent to TDCs' front panel.

Returns:

A 3-bit word PoI

Here is the call graph for this function:

7.8.4.14 unsigned short VME::FPGAUnitV1495::GetUserFirmwareRevision () const

Here is the call graph for this function:

7.8.4.15 void VME::FPGAUnitV1495::PulseTDCBits (unsigned short *bits*, unsigned int *time_us* = `10`) const

Send a pulse to TDCs' front panel.

Parameters:

← ***bits*** The pattern to send (3 bits)

← ***time_us*** Pulse width (in us)

Here is the call graph for this function:

7.8.4.16 void VME::FPGAUnitV1495::ResetFPGA () const

Here is the call graph for this function:

7.8.4.17 void VME::FPGAUnitV1495::SetControl (const FPGAUnitV1495Control & *control*) const

Set the user-defined control word.

Here is the call graph for this function:

7.8.4.18 void VME::FPGAUnitV1495::SetInternalClockPeriod (uint32_t *period*) const

Set the internal clock period.

Parameters:

← ***period*** Clock period (in units of 25 ns)

Here is the call graph for this function:

7.8.4.19 void VME::FPGAUnitV1495::SetInternalTriggerPeriod (uint32_t *period*) const

Set the internal trigger period.

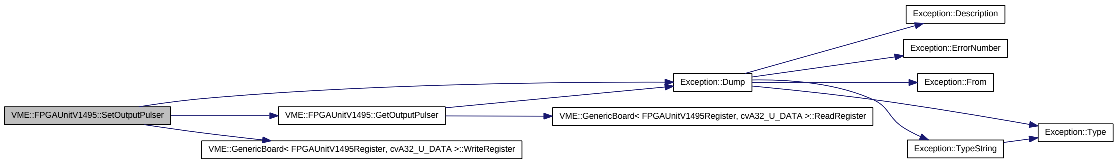
Parameters:

← ***period*** Trigger period (in units of 50 ns)

Here is the call graph for this function:

7.8.4.20 void VME::FPGAUnitV1495::SetOutputPulser (unsigned short *id*, bool *enable* = `true`) const

Here is the call graph for this function:


Exception::Description
Exception::ErrorNumber
Exception::Dump
Exception::From
VME::FPGAUnitV1495::SetOutputPulser
VME::FPGAUnitV1495::GetOutputPulser
VME::GenericBoard< FPGAUnitV1495Register, cvA32_U_DATA >::ReadRegister
Exception::Type
VME::GenericBoard< FPGAUnitV1495Register, cvA32_U_DATA >::WriteRegister
Exception::TypeString

7.8.4.21 void VME::FPGAUnitV1495::SetTDCBits (unsigned short *bits*) const

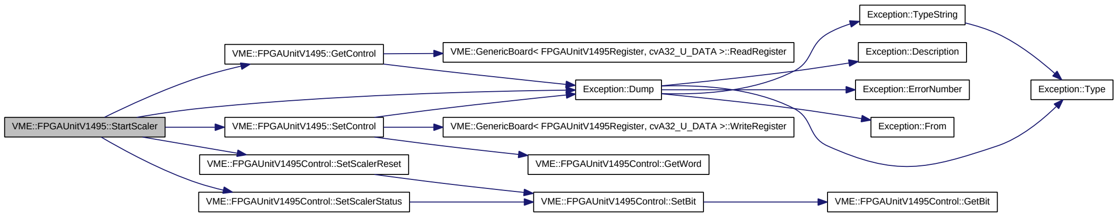
Set a pattern of bits to be sent to all TDCs through the ECL mezzanine.

Here is the call graph for this function:

7.8.4.22 void VME::FPGAUnitV1495::StartScaler ()

Start the inner triggers counter.

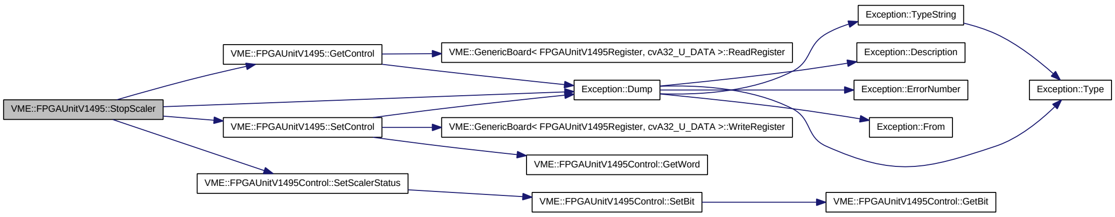
Here is the call graph for this function:


Exception::TypeString
VME::FPGAUnitV1495::GetControl
VME::GenericBoard< FPGAUnitV1495Register, cvA32_U_DATA >::ReadRegister
Exception::Description
Exception::Dump
Exception::ErrorNumber
Exception::Type
VME::FPGAUnitV1495::StartScaler
VME::FPGAUnitV1495::SetControl
VME::GenericBoard< FPGAUnitV1495Register, cvA32_U_DATA >::WriteRegister
Exception::From
VME::FPGAUnitV1495Control::SetScalerReset
VME::FPGAUnitV1495Control::GetWord
VME::FPGAUnitV1495Control::SetScalerStatus
VME::FPGAUnitV1495Control::SetBit
VME::FPGAUnitV1495Control::GetBit

7.8.4.23 void VME::FPGAUnitV1495::StopScaler ()

Stop the inner triggers counter.

Here is the call graph for this function:


VME::FPGAUnitV1495::StopScaler
VME::FPGAUnitV1495::GetControl
VME::FPGAUnitV1495::SetControl
VME::FPGAUnitV1495Control::SetScalerStatus
VME::GenericBoard< FPGAUnitV1495Register, cvA32_U_DATA >::ReadRegister
VME::GenericBoard< FPGAUnitV1495Register, cvA32_U_DATA >::WriteRegister
Exception::Dump
VME::FPGAUnitV1495Control::GetWord
VME::FPGAUnitV1495Control::SetBit
Exception::TypeString
Exception::Description
Exception::ErrorNumber
Exception::From
VME::FPGAUnitV1495Control::GetBit
Exception::Type

7.8.5 Field Documentation

7.8.5.1 bool VME::FPGAUnitV1495::fScalerStarted `[private]`

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

- include/VME_FPGAUnitV1495.h
- src/VME_FPGAUnitV1495.cpp

7.9 VME::FPGAUnitV1495Control Class Reference

```
#include <VME_FPGAUnitV1495.h>
```

Public Types

- enum ClockSource { InternalClock = 0x0, ExternalClock = 0x1 }
- enum TriggerSource { InternalTrigger = 0x0, ExternalTrigger = 0x1 }
- enum SignalSource { InternalSignal = 0x0, ExternalSignal = 0x1 }

Public Member Functions

- FPGAUnitV1495Control (uint32_t word)
- virtual ~FPGAUnitV1495Control ()
- void Dump () const
- uint32_t GetWord () const
- ClockSource GetClockSource () const

  Get the clock source.
- void SetClockSource (const ClockSource &cs)

  Switch between internal and external clock source.
- TriggerSource GetTriggerSource () const

  Get the trigger source.
- void SetTriggerSource (const TriggerSource &cs)

  Switch between internal and external trigger source.
- bool GetScalerStatus () const
- void SetScalerStatus (bool start=true)
- void SetScalerReset (bool reset=true)
- SignalSource GetSignalSource (unsigned short map_id) const
- void SetSignalSource (unsigned short map_id, const SignalSource &s)

Private Member Functions

- bool GetBit (unsigned short id) const
- void SetBit (unsigned short id, unsigned short value=0x1)

Private Attributes

- uint32_t fWord

7.9.1 Detailed Description

User-defined control word to be propagated to the CAEN V1495 board firmware.

Author:

Laurent Forthomme <laurent.forthomme@cern.ch>

Date:

27 Jun 2015

7.9.2 Member Enumeration Documentation

7.9.2.1 enum VME::FPGAUnitV1495Control::ClockSource

Enumerator:

InternalClock

ExternalClock

7.9.2.2 enum VME::FPGAUnitV1495Control::SignalSource

Enumerator:

InternalSignal

ExternalSignal

7.9.2.3 enum VME::FPGAUnitV1495Control::TriggerSource

Enumerator:

InternalTrigger

ExternalTrigger

7.9.3 Constructor & Destructor Documentation

7.9.3.1 VME::FPGAUnitV1495Control::FPGAUnitV1495Control (uint32_t *word*) `[inline]`

7.9.3.2 virtual VME::FPGAUnitV1495Control::∼FPGAUnitV1495Control () `[inline, virtual]`

7.9.4 Member Function Documentation

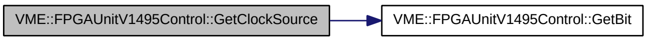
7.9.4.1 void VME::FPGAUnitV1495Control::Dump () const `[inline]`

7.9.4.2 bool VME::FPGAUnitV1495Control::GetBit (unsigned short *id*) const `[inline, private]`

7.9.4.3 ClockSource VME::FPGAUnitV1495Control::GetClockSource () const `[inline]`

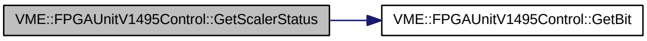
Get the clock source.

Here is the call graph for this function:


VME::FPGAUnitV1495Control::GetClockSource
VME::FPGAUnitV1495Control::GetBit

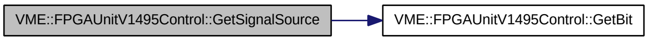
7.9.4.4 bool VME::FPGAUnitV1495Control::GetScalerStatus () const `[inline]`

Here is the call graph for this function:


VME::FPGAUnitV1495Control::GetScalerStatus
VME::FPGAUnitV1495Control::GetBit

7.9.4.5 SignalSource VME::FPGAUnitV1495Control::GetSignalSource (unsigned short *map_id*) const `[inline]`

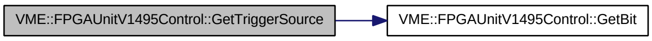
Here is the call graph for this function:


VME::FPGAUnitV1495Control::GetSignalSource
VME::FPGAUnitV1495Control::GetBit

7.9.4.6 TriggerSource VME::FPGAUnitV1495Control::GetTriggerSource () const `[inline]`

Get the trigger source.

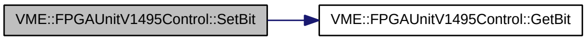
Here is the call graph for this function:


VME::FPGAUnitV1495Control::GetTriggerSource
VME::FPGAUnitV1495Control::GetBit

7.9.4.7 uint32_t VME::FPGAUnitV1495Control::GetWord () const `[inline]`

7.9.4.8 void VME::FPGAUnitV1495Control::SetBit (unsigned short *id*, unsigned short *value* = `0x1`) `[inline, private]`

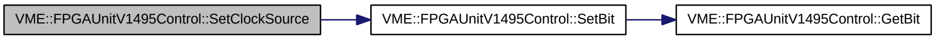
Here is the call graph for this function:


VME::FPGAUnitV1495Control::SetBit
VME::FPGAUnitV1495Control::GetBit

7.9.4.9 void VME::FPGAUnitV1495Control::SetClockSource (const ClockSource & *cs*) `[inline]`

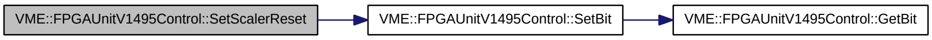
Switch between internal and external clock source.

Here is the call graph for this function:


VME::FPGAUnitV1495Control::SetClockSource
VME::FPGAUnitV1495Control::SetBit
VME::FPGAUnitV1495Control::GetBit

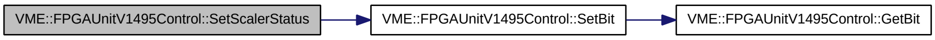
7.9.4.10 void VME::FPGAUnitV1495Control::SetScalerReset (bool *reset* = `true`) `[inline]`

Here is the call graph for this function:


VME::FPGAUnitV1495Control::SetScalerReset
VME::FPGAUnitV1495Control::SetBit
VME::FPGAUnitV1495Control::GetBit

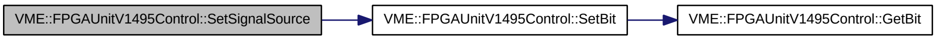
7.9.4.11 void VME::FPGAUnitV1495Control::SetScalerStatus (bool *start* = `true`) `[inline]`

Here is the call graph for this function:


VME::FPGAUnitV1495Control::SetScalerStatus
VME::FPGAUnitV1495Control::SetBit
VME::FPGAUnitV1495Control::GetBit

7.9.4.12 void VME::FPGAUnitV1495Control::SetSignalSource (unsigned short *map_id*, const SignalSource & *s*) `[inline]`

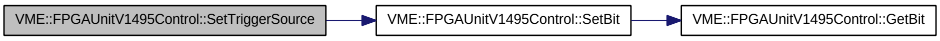
Here is the call graph for this function:


VME::FPGAUnitV1495Control::SetSignalSource
VME::FPGAUnitV1495Control::SetBit
VME::FPGAUnitV1495Control::GetBit

7.9.4.13 void VME::FPGAUnitV1495Control::SetTriggerSource (const TriggerSource & *cs*) `[inline]`

Switch between internal and external trigger source.

Here is the call graph for this function:


VME::FPGAUnitV1495Control::SetTriggerSource
VME::FPGAUnitV1495Control::SetBit
VME::FPGAUnitV1495Control::GetBit

7.9.5 Field Documentation

7.9.5.1 uint32_t VME::FPGAUnitV1495Control::fWord `[private]`

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

- include/VME_FPGAUnitV1495.h

7.10 VME::GenericBoard< Register, am > Class Template Reference

```
#include <VME_GenericBoard.h>
```

Public Member Functions

- GenericBoard (int32_t bhandle, uint32_t baseaddr)
- virtual ∼GenericBoard ()

Protected Member Functions

- void WriteRegister (const Register &reg, const uint16_t &data) const

  Write on register.
- void WriteRegister (const Register &reg, const uint32_t &data) const

  Write on register.
- void ReadRegister (const Register &reg, uint16_t ∗data) const

  Read on register.
- void ReadRegister (const Register &reg, uint32_t ∗data) const

  Read on register.

Protected Attributes

- int32_t fHandle
- uint32_t fBaseAddr

template<class Register, CVAddressModifier am> class VME::GenericBoard< Register, am >

7.10.1 Constructor & Destructor Documentation

7.10.1.1 template<class Register, CVAddressModifier am> VME::GenericBoard< Register, am >::GenericBoard (int32_t *bhandle*, uint32_t *baseaddr*) `[inline]`

7.10.1.2 template<class Register, CVAddressModifier am> virtual VME::GenericBoard< Register, am >::∼GenericBoard () `[inline, virtual]`

7.10.2 Member Function Documentation

7.10.2.1 template<class Register, CVAddressModifier am> void VME::GenericBoard< Register, am >::ReadRegister (const Register & *reg*, uint32_t ∗ *data*) const `[inline, protected]`

Read on register. Read a 32-bit word in the register

Parameters:

← ***addr*** register

→ ***data*** word

7.10.2.2 template<class Register, CVAddressModifier am> void VME::GenericBoard< Register, am >::ReadRegister (const Register & *reg*, uint16_t ∗ *data*) const `[inline, protected]`

Read on register. Read a 16-bit word in the register

Parameters:

← ***addr*** register

→ ***data*** word

7.10.2.3 template<class Register, CVAddressModifier am> void VME::GenericBoard< Register, am >::WriteRegister (const Register & *reg*, const uint32_t & *data*) const `[inline, protected]`

Write on register. Write a 32-bit word in the register

Parameters:

← ***addr*** register

← ***data*** word

7.10.2.4 template<class Register, CVAddressModifier am> void VME::GenericBoard< Register, am >::WriteRegister (const Register & *reg*, const uint16_t & *data*) const `[inline, protected]`

Write on register. Write a 16-bit word in the register

Parameters:

← ***addr*** register

← ***data*** word

7.10.3 Field Documentation

7.10.3.1 template<class Register, CVAddressModifier am> uint32_t VME::GenericBoard< Register, am >::fBaseAddr `[protected]`

7.10.3.2 template<class Register, CVAddressModifier am> int32_t VME::GenericBoard< Register, am >::fHandle `[protected]`

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

- include/VME_GenericBoard.h

7.11 VME::GlobalOffset Struct Reference

```
#include <VME_TDCV1x90.h>
```

Data Fields

- uint16_t coarse
- uint16_t fine

7.11.1 Field Documentation

7.11.1.1 uint16_t VME::GlobalOffset::coarse

7.11.1.2 uint16_t VME::GlobalOffset::fine

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

- include/VME_TDCV1x90.h

7.12 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

  Placeholder for the MessageKey retrieval method.
- void Dump (std::ostream &os=std::cout) const

Private Attributes

- WebSocket ∗ fWS
- std::string fOriginalString

7.12.1 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

7.12.2 Constructor & Destructor Documentation

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

Here is the call graph for this function:

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

Here is the call graph for this function:

7.12.3 Member Function Documentation

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

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

Reimplemented from Message.

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

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

Placeholder for the MessageKey retrieval method.

Reimplemented from Message.

7.12.4 Field Documentation

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

7.12.4.2 WebSocket∗ HTTPMessage::fWS `[private]`

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

- include/HTTPMessage.h

7.13 VME::IOModuleV262 Class Reference

`#include <VME_IOModuleV262.h>`Inheritance diagram for VME::IOModuleV262:Collaboration diagram for VME::IOModuleV262:

Public Member Functions

- IOModuleV262 (int32_t bhandle, uint32_t baseaddr)
- ∼IOModuleV262 ()
- unsigned short GetSerialNumber () const
- unsigned short GetModuleVersion () const
- unsigned short GetModuleType () const
- unsigned short GetManufacturerId () const
- unsigned short GetIdentifier () const

7.13.1 Constructor & Destructor Documentation

7.13.1.1 VME::IOModuleV262::IOModuleV262 (int32_t *bhandle*, uint32_t *baseaddr*)

Here is the call graph for this function:

7.13.1.2 VME::IOModuleV262::∼IOModuleV262 () `[inline]`

7.13.2 Member Function Documentation

7.13.2.1 unsigned short VME::IOModuleV262::GetIdentifier () const

Here is the call graph for this function:

7.13.2.2 unsigned short VME::IOModuleV262::GetManufacturerId () const

Here is the call graph for this function:

7.13.2.3 unsigned short VME::IOModuleV262::GetModuleType () const

Here is the call graph for this function:

7.13.2.4 unsigned short VME::IOModuleV262::GetModuleVersion () const

Here is the call graph for this function:

7.13.2.5 unsigned short VME::IOModuleV262::GetSerialNumber () const

Here is the call graph for this function:

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

- include/VME_IOModuleV262.h
- src/VME_IOModuleV262.cpp

7.14 Message Class Reference

Base socket message type.

`#include <Message.h>`Inheritance diagram for Message:

Public Member Functions

- Message ()

  Void message constructor.

- Message (const char ∗msg)

  Construct a message from a string.

- Message (std::string msg)

  Construct a message from a string.

- virtual ∼Message ()
- MessageKey GetKey () const

  Placeholder for the MessageKey retrieval method.

- std::string GetString () const

  Retrieve the string carried by this message as a whole.

- bool IsFromWeb () const

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

- void Dump (std::ostream &os=std::cout) const

Protected Attributes

- std::string fString

7.14.1 Detailed Description

Base socket message type. Base handler for messages to be transmitted through the socket

Author:

Laurent Forthomme <`laurent.forthomme@cern.ch`>

Date:

6 Apr 2015

7.14.2 Constructor & Destructor Documentation

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

Void message constructor.

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

Construct a message from a string.

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

Construct a message from a string.

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

7.14.3 Member Function Documentation

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

Reimplemented in HTTPMessage, and SocketMessage.

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

Placeholder for the MessageKey retrieval method.

Reimplemented in HTTPMessage, and SocketMessage.

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

Retrieve the string carried by this message as a whole.

Reimplemented in SocketMessage.

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

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

7.14.4 Field Documentation

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

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

- include/Message.h

7.15 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.

- void StartAcquisition ()

  Start the data acquisition.

- void StopAcquisition ()
- SocketType GetType () const

  Socket actor type retrieval method.

Private Member Functions

- void AddClient ()

  Add a client to listen to.

- void DisconnectClient (int sid, MessageKey key, bool force=false)

  Disconnect a client.

- void SwitchClientType (int sid, Socket::SocketType type)
- void ProcessMessage (SocketMessage m, int sid)

  Process a message received from the socket.

Private Attributes

- WebSocket ∗ fWS
- int fNumAttempts
- pid_t fPID
- int fStdoutPipe [2]
- int fStderrPipe [2]

7.15.1 Detailed Description

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

Author:

Laurent Forthomme <`laurent.forthomme@cern.ch`>

Date:

23 Mar 2015

7.15.2 Constructor & Destructor Documentation

7.15.2.1 Messenger::Messenger ()

Build a void master object or socket actor.

7.15.2.2 Messenger::Messenger (int *port*)

Build a master object to control the socket.

Here is the call graph for this function:

7.15.2.3 Messenger::∼Messenger ()

Here is the call graph for this function:

7.15.3 Member Function Documentation

7.15.3.1 void Messenger::AddClient () `[private]`

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

Here is the call graph for this function:

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

Emit a message to all clients connected through the socket.

Parameters:

← ***m*** Message to transmit

Here is the call graph for this function:

7.15.3.3 bool Messenger::Connect ()

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

Here is the call graph for this function:

7.15.3.4 void Messenger::Disconnect ()

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

Here is the call graph for this function:

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

Disconnect a client. Ask to a client to disconnect from this socket.

Parameters:

← ***sid*** Unique identifier of the client to disconnect

← ***key*** Key to the message to transmit for disconnection

← ***force*** Do we need to force the client out of this socket ?

Here is the call graph for this function:

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

Socket actor type retrieval method.

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

Process a message received from the socket.

Parameters:

← ***Unique*** identifier of the client sending the message

Here is the call graph for this function:

7.15.3.8 void Messenger::Receive ()

Handle a message reception from a client.

Here is the call graph for this function:

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

Send any type of message to any client.

Parameters:

← ***m*** Message to transmit

← ***sid*** Unique identifier of the client on this socket

Here is the call graph for this function:

7.15.3.10 void Messenger::StartAcquisition ()

Start the data acquisition.

Here is the call graph for this function:

7.15.3.11 void Messenger::StopAcquisition ()

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

Here is the call graph for this function:

7.15.4 Field Documentation

7.15.4.1 int Messenger::fNumAttempts `[private]`

7.15.4.2 pid_t Messenger::fPID `[private]`

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

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

7.15.4.5 WebSocket∗ Messenger::fWS `[private]`

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

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

7.16 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

7.16.1 Detailed Description

Base socket object from which clients/master from a socket inherit. General object providing all useful method to connect/bind/send/receive information through system sockets.

Author:

Laurent Forthomme <laurent.forthomme@cern.ch>

Date:

23 Mar 2015

7.16.2 Member Typedef Documentation

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

7.16.3 Member Enumeration Documentation

7.16.3.1 enum Socket::SocketType

Type of actor playing a role on the socket.

Enumerator:

INVALID

MASTER

WEBSOCKET_CLIENT

CLIENT

DETECTOR

7.16.4 Constructor & Destructor Documentation

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

7.16.4.2 Socket::Socket (int *port*)

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

7.16.5 Member Function Documentation

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

Accept connection from a client. Set the socket to accept connections any client transmitting through the socket

Parameters:

inout] socket Master/client object to enable on the socket

Here is the call graph for this function:

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

Bind a name to a socket.

Returns:

Success of the operation

Here is the call graph for this function:

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

Configure the socket object for communication.

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

Create an endpoint for communication.

7.16.5.5 void Socket::DumpConnected () const

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

Receive a message from a socket.

Returns:

Received message as a std::string

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

Retrieve the port used for this socket.

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

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

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

Here is the call graph for this function:

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

Listen to incoming messages. Set the socket to listen to any message coming from outside

Here is the call graph for this function:

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

Here is the call graph for this function:

7.16.5.13 void Socket::SelectConnections ()

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

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

Send a message on a socket.

Here is the call graph for this function:

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

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

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

Start the socket. Launch all mandatory operations to set the socket to be used

Returns:

Success of the operation

Here is the call graph for this function:

7.16.5.18 void Socket::Stop ()

Terminates the socket and all attached communications.

7.16.6 Field Documentation

7.16.6.1 struct sockaddr_in Socket::fAddress `[read, private]`

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

7.16.6.3 fd_set Socket::fMaster `[protected]`

Master file descriptor list.

7.16.6.4 int Socket::fPort `[protected]`

7.16.6.5 fd_set Socket::fReadFds `[protected]`

Temp file descriptor list for select().

7.16.6.6 int Socket::fSocketId `[private]`

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

7.16.6.7 SocketCollection Socket::fSocketsConnected `[protected]`

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

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

7.17 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

7.17.1 Detailed Description

Socket-passed message type.

Author:

Laurent Forthomme <`laurent.forthomme@cern.ch`>

Date:

26 Mar 2015

7.17.2 Constructor & Destructor Documentation

7.17.2.1 SocketMessage::SocketMessage () `[inline]`

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

Here is the call graph for this function:

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

Here is the call graph for this function:

7.17.2.4 SocketMessage::SocketMessage (std::string *msg_s*) `[inline]`

Here is the call graph for this function:

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

Construct a socket message out of a key.

Here is the call graph for this function:

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

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

Here is the call graph for this function:

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

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

Here is the call graph for this function:

7.17.2.8 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:

7.17.2.9 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:

7.17.2.10 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:

7.17.2.11 SocketMessage::SocketMessage (MessageMap *msg_m*) `[inline]`

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

7.17.2.12 SocketMessage::∼SocketMessage () `[inline]`

7.17.3 Member Function Documentation

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

Reimplemented from Message.

Here is the call graph for this function:

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

Extract the message's integer value.

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

Extract the message's key.

Reimplemented from Message.

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

Extract the whole key:value message.

Reimplemented from Message.

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

Extract the message's string value.

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

Extract the message's vector of string value.

Here is the call graph for this function:

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

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

Double-valued message.

Here is the call graph for this function:

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

Float-valued message.

Here is the call graph for this function:

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

Send an integer-valued message.

Here is the call graph for this function:

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

String-valued message.

Here is the call graph for this function:

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

7.17.4 Field Documentation

7.17.4.1 MessageMap SocketMessage::fMessage `[private]`

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

- include/SocketMessage.h

7.18 VME::TDCErrorFlag Class Reference

Error flags handler.

`#include <VME_TDCEvent.h>`

Public Member Functions

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

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

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

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

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

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

  Internal fatal chip error has been detected.

Private Attributes

- uint16_t fWord

Friends

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

7.18.1 Detailed Description

Error flags handler.

Author:

Laurent Forthomme <`laurent.forthomme@cern.ch`>

Date:

22 Jun 2015

7.18.2 Constructor & Destructor Documentation

7.18.2.1 VME::TDCErrorFlag::TDCErrorFlag (uint16_t *ef*) `[inline]`

7.18.2.2 virtual VME::TDCErrorFlag::∼TDCErrorFlag () `[inline, virtual]`

7.18.3 Member Function Documentation

7.18.3.1 void VME::TDCErrorFlag::Dump () const `[inline]`

7.18.3.2 uint16_t VME::TDCErrorFlag::GetWord () const `[inline]`

7.18.3.3 bool VME::TDCErrorFlag::HasGroupError (unsigned int *group_id*) const `[inline]`

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

7.18.3.4 bool VME::TDCErrorFlag::HasInternalChipError () const `[inline]`

Internal fatal chip error has been detected.

7.18.3.5 bool VME::TDCErrorFlag::HasL1BufferOverflow (unsigned int *group_id*) const `[inline]`

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

7.18.3.6 bool VME::TDCErrorFlag::HasReachedEventSizeLimit () const `[inline]`

Hits rejected because of programmed event size limit.

7.18.3.7 bool VME::TDCErrorFlag::HasReadoutFIFOOverflow (unsigned int *group_id*) const `[inline]`

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

7.18.3.8 bool VME::TDCErrorFlag::HasTriggerFIFOOverflow () const `[inline]`

Event lost (trigger FIFO overflow).

7.18.4 Friends And Related Function Documentation

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

7.18.5 Field Documentation

7.18.5.1 uint16_t VME::TDCErrorFlag::fWord `[private]`

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

- include/VME_TDCEvent.h

7.19 VME::TDCEvent Class Reference

HPTDC event parser.

```
#include <VME_TDCEvent.h>
```

Public Types

- enum EventType {

  TDCMeasurement = 0x0, TDCHeader = 0x1, TDCTrailer = 0x3, TDCError = 0x4,

  GlobalHeader = 0x8, GlobalTrailer = 0x10, ETTT = 0x11, Filler = 0x18 }

Public Member Functions

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

  Type of packet read out from the TDC.

- unsigned int GetTDCId () const

  Programmed identifier of master TDC providing the event.

- uint16_t GetEventId () const

  Event identifier from event counter.

- uint16_t GetWordCount () const

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

- unsigned int GetGeo () const
- unsigned int GetChannelId () const

  Channel number for.

- uint32_t GetEventCount () const

  Total number of events.

- uint16_t GetBunchId () const

  Bunch identifier of trigger (or trigger time tag).

- bool IsTrailing () const

Are we dealing with a trailing or a leading measurement?

- uint32_t GetETTT () const

  Extended trigger time tag.

- uint32_t GetTime (bool pair=false) const

  Edge measurement in programmed time resolution.

- unsigned int GetWidth () const

  Width of pulse in programmed time resolution.

- unsigned int GetStatus () const
- TDCErrorFlag GetErrorFlags () const

  Return error flags if an error condition has been detected.

Private Attributes

- uint32_t fWord

7.19.1 Detailed Description

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

Author:

Laurent Forthomme <`laurent.forthomme@cern.ch`>

Date:

4 May 2015

7.19.2 Member Enumeration Documentation

7.19.2.1 enum VME::TDCEvent::EventType

Enumerator:

TDCMeasurement

TDCHeader

TDCTrailer

TDCError

GlobalHeader

GlobalTrailer

ETTT

Filler

7.19.3 Constructor & Destructor Documentation

7.19.3.1 VME::TDCEvent::TDCEvent () `[inline]`

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

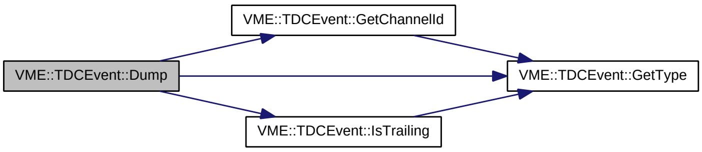
7.19.3.3 VME::TDCEvent::TDCEvent (const uint32_t & *word*) `[inline]`

7.19.3.4 virtual VME::TDCEvent::∼TDCEvent () `[inline, virtual]`

7.19.4 Member Function Documentation

7.19.4.1 void VME::TDCEvent::Dump () const `[inline]`

Here is the call graph for this function:



7.19.4.2 uint16_t VME::TDCEvent::GetBunchId () const `[inline]`

Bunch identifier of trigger (or trigger time tag).

Here is the call graph for this function:

7.19.4.3 unsigned int VME::TDCEvent::GetChannelId () const `[inline]`

Channel number for.

Here is the call graph for this function:

7.19.4.4 TDCErrorFlag VME::TDCEvent::GetErrorFlags () const `[inline]`

Return error flags if an error condition has been detected.

Here is the call graph for this function:

7.19.4.5 uint32_t VME::TDCEvent::GetETTT () const `[inline]`

Extended trigger time tag.

Here is the call graph for this function:

7.19.4.6 uint32_t VME::TDCEvent::GetEventCount () const `[inline]`

Total number of events.

Here is the call graph for this function:

7.19.4.7 uint16_t VME::TDCEvent::GetEventId () const `[inline]`

Event identifier from event counter.

Here is the call graph for this function:

7.19.4.8 unsigned int VME::TDCEvent::GetGeo () const `[inline]`

Here is the call graph for this function:

7.19.4.9 unsigned int VME::TDCEvent::GetStatus () const `[inline]`

Here is the call graph for this function:

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

Programmed identifier of master TDC providing the event.

Here is the call graph for this function:

7.19.4.11 uint32_t VME::TDCEvent::GetTime (bool *pair* = `false`) const `[inline]`

Edge measurement in programmed time resolution.

Parameters:

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

Here is the call graph for this function:

VME::TDCEvent::GetTime
VME::TDCEvent::GetType

7.19.4.12 EventType VME::TDCEvent::GetType () const `[inline]`

Type of packet read out from the TDC.

7.19.4.13 unsigned int VME::TDCEvent::GetWidth () const `[inline]`

Width of pulse in programmed time resolution.

Here is the call graph for this function:

7.19.4.14 uint32_t VME::TDCEvent::GetWord () const `[inline]`

7.19.4.15 uint16_t VME::TDCEvent::GetWordCount () const `[inline]`

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

Here is the call graph for this function:

7.19.4.16 bool VME::TDCEvent::IsTrailing () const `[inline]`

Are we dealing with a trailing or a leading measurement?

Here is the call graph for this function:

7.19.4.17 void VME::TDCEvent::SetWord (const uint32_t & *word*) `[inline]`

7.19.5 Field Documentation

7.19.5.1 uint32_t VME::TDCEvent::fWord `[private]`

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

- include/VME_TDCEvent.h

7.20 VME::TDCMeasurement Class Reference

```
#include <VME_TDCMeasurement.h>
```

Public Member Functions

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

Private Attributes

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

7.20.1 Constructor & Destructor Documentation

7.20.1.1 VME::TDCMeasurement::TDCMeasurement () `[inline]`

7.20.1.2 VME::TDCMeasurement::TDCMeasurement (const std::vector< TDCEvent > & *v*) `[inline]`

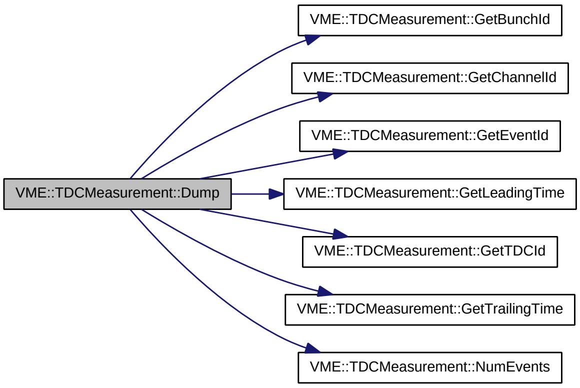
Here is the call graph for this function:

7.20.1.3 VME::TDCMeasurement::∼TDCMeasurement () `[inline]`

7.20.2 Member Function Documentation

7.20.2.1 void VME::TDCMeasurement::Dump () `[inline]`

Here is the call graph for this function:


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

7.20.2.2 uint16_t VME::TDCMeasurement::GetBunchId () `[inline]`

7.20.2.3 uint16_t VME::TDCMeasurement::GetChannelId (unsigned short *event_id* = `0`) `[inline]`

7.20.2.4 uint32_t VME::TDCMeasurement::GetETTT () `[inline]`

7.20.2.5 uint16_t VME::TDCMeasurement::GetEventId () `[inline]`

7.20.2.6 uint32_t VME::TDCMeasurement::GetLeadingTime (unsigned short *event_id* = `0`) `[inline]`

7.20.2.7 uint16_t VME::TDCMeasurement::GetTDCId () `[inline]`

7.20.2.8 uint16_t VME::TDCMeasurement::GetToT (unsigned short *event_id* = `0`) `[inline]`

Here is the call graph for this function:

7.20.2.9 uint32_t VME::TDCMeasurement::GetTrailingTime (unsigned short *event_id* = 0) `[inline]`

7.20.2.10 size_t VME::TDCMeasurement::NumEvents () const `[inline]`

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

7.20.3 Field Documentation

7.20.3.1 std::vector< std::pair<TDCEvent,TDCEvent> > VME::TDCMeasurement::fEvents `[private]`

7.20.3.2 std::map<TDCEvent::EventType,TDCEvent> VME::TDCMeasurement::fMap `[private]`

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

- include/VME_TDCMeasurement.h

7.21 VME::TDCV1x90 Class Reference

`#include <VME_TDCV1x90.h>`Inheritance diagram for VME::TDCV1x90:Collaboration diagram for VME::TDCV1x90:

Public Types

- enum DLLMode { DLL_Direct_LowRes = 0x0, DLL_PLL_LowRes = 0x1, DLL_PLL_MedRes = 0x2, DLL_PLL_HighRes = 0x3 }

Public Member Functions

- TDCV1x90 (int32_t bhandle, uint32_t baseaddr)
- ~TDCV1x90 ()
- void SetVerboseLevel (unsigned short verb=1)
- void SetTestMode (bool en=true) const
- bool GetTestMode () const
- uint32_t GetModel () const
- uint32_t GetOUI () const
- uint32_t GetSerialNumber () const
- uint16_t GetFirmwareRevision () const
- void CheckConfiguration () const
- void EnableChannel (short) const
- void DisableChannel (short) const
- void SetPoI (uint16_t word1, uint16_t word2) const
- std::map< unsigned short, bool > GetPoI () const
- void SetLSBTraileadEdge (trailead_edge_lsb) const
- void SetAcquisitionMode (const AcquisitionMode &)
- AcquisitionMode GetAcquisitionMode ()
- void SetTriggerMatching ()
- void SetContinuousStorage ()
- void SetDetectionMode (const DetectionMode &detm)
- DetectionMode GetDetectionMode ()
- void SetDLLClock (const DLLMode &dll) const
- DLLMode GetDLLClock () const
- void SetGlobalOffset (const GlobalOffset &) const
- GlobalOffset GetGlobalOffset () const
- void SetRCAdjust (int, uint16_t) const
- uint16_t GetRCAdjust (int) const
- uint32_t GetEventCounter () const

  Number of occured triggers.

- uint16_t GetEventStored () const

  Number of events currently stored in the output buffer.

- void SetTDCEncapsulation (bool) const
- bool GetTDCEncapsulation () const
- void SetErrorMarks (bool mode=true)
- bool GetErrorMarks () const
- void SetPairModeResolution (int, int) const
- uint16_t GetResolution () const
- void SetBLTEventNumberRegister (const uint16_t &) const
- uint16_t GetBLTEventNumberRegister () const
- void SetWindowWidth (const uint16_t &)
- uint16_t GetWindowWidth () const
- void SetWindowOffset (const int16_t &) const
- int16_t GetWindowOffset () const
- uint16_t GetTriggerConfiguration (const trig_conf &) const
- bool SoftwareClear () const
- bool SoftwareReset () const
- bool HardwareReset () const
- void SetETTT (bool ettt=true) const
- bool GetETTT () const
- void SetStatus (const TDCV1x90Status &) const
- TDCV1x90Status GetStatus () const
- void SetControl (const TDCV1x90Control &) const
- TDCV1x90Control GetControl () const
- TDCEventCollection FetchEvents ()
- void SetChannelDeadTime (unsigned short dt) const
- unsigned short GetChannelDeadTime () const
- void SetFIFOSize (const uint16_t &) const
- uint16_t GetFIFOSize () const
- void abort ()

Private Member Functions

- bool WaitMicro (micro_handshake mode) const
- void ReadAcquisitionMode ()
- void ReadDetectionMode ()

Private Attributes

- unsigned short fVerb
- AcquisitionMode fAcquisitionMode
- DetectionMode fDetectionMode
- bool fErrorMarks
- uint16_t fWindowWidth
- uint32_t ∗ fBuffer
- uint32_t nchannels
- bool gEnd
- std::string pair_lead_res [8]
- std::string pair_width_res [16]

7.21.1 Detailed Description

Author:

Laurent Forthomme <laurent.forthomme@cern.ch>
Bob Velghe <bob.velghe@cern.ch>

Date:

Jun 2010 (NA62-Gigatracker)
May 2015 (CMS-TOTEM PPS)

7.21.2 Member Enumeration Documentation

7.21.2.1 enum VME::TDCV1x90::DLLMode

Enumerator:

DLL_Direct_LowRes

DLL_PLL_LowRes

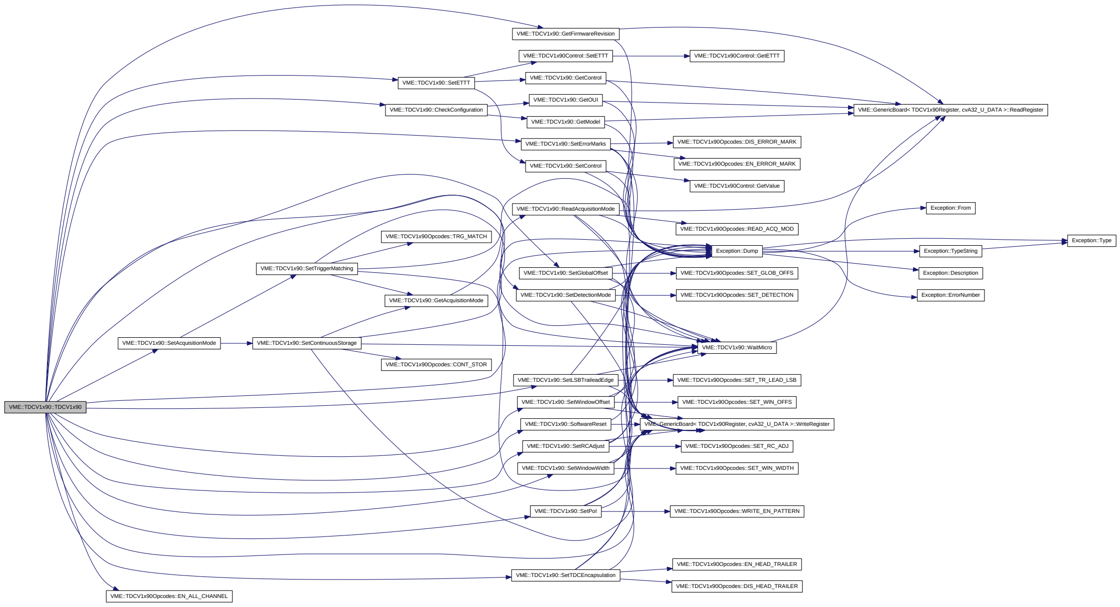
DLL_PLL_MedRes

DLL_PLL_HighRes

7.21.3 Constructor & Destructor Documentation

7.21.3.1 VME::TDCV1x90::TDCV1x90 (int32_t *bhandle*, uint32_t *baseaddr*)

Here is the call graph for this function:


VME::TDCV1x90::GetFirmwareRevision
VME::TDCV1x90Control::SetETTT
VME::TDCV1x90Control::GetETTT
VME::TDCV1x90::SetETTT
VME::TDCV1x90::GetControl
VME::TDCV1x90::GetOUI
VME::TDCV1x90::CheckConfiguration
VME::GenericBoard< TDCV1x90Register, cvA32_U_DATA >::ReadRegister
VME::TDCV1x90::GetModel
VME::TDCV1x90::SetErrorMarks
VME::TDCV1x90Opcodes::DIS_ERROR_MARK
VME::TDCV1x90Opcodes::EN_ERROR_MARK
VME::TDCV1x90::SetControl
VME::TDCV1x90Control::GetValue
VME::TDCV1x90::ReadAcquisitionMode
Exception::From
VME::TDCV1x90Opcodes::READ_ACQ_MOD
VME::TDCV1x90Opcodes::TRG_MATCH
Exception::Type
Exception::Dump
Exception::TypeString
VME::TDCV1x90::SetTriggerMatching
VME::TDCV1x90::SetGlobalOffset
VME::TDCV1x90Opcodes::SET_GLOB_OFFS
Exception::Description
VME::TDCV1x90::GetAcquisitionMode
VME::TDCV1x90::SetDetectionMode
VME::TDCV1x90Opcodes::SET_DETECTION
Exception::ErrorNumber
VME::TDCV1x90::SetAcquisitionMode
VME::TDCV1x90::SetContinuousStorage
VME::TDCV1x90::WaitMicro
VME::TDCV1x90Opcodes::CONT_STOR
VME::TDCV1x90::SetLSBTraileadEdge
VME::TDCV1x90Opcodes::SET_TR_LEAD_LSB
VME::TDCV1x90::TDCV1x90
VME::TDCV1x90::SetWindowOffset
VME::TDCV1x90Opcodes::SET_WIN_OFFS
VME::TDCV1x90::SoftwareReset
VME::GenericBoard< TDCV1x90Register, cvA32_U_DATA >::WriteRegister
VME::TDCV1x90::SetRCAdjust
VME::TDCV1x90Opcodes::SET_RC_ADJ
VME::TDCV1x90::SetWindowWidth
VME::TDCV1x90Opcodes::SET_WIN_WIDTH
VME::TDCV1x90::SetPol
VME::TDCV1x90Opcodes::WRITE_EN_PATTERN
VME::TDCV1x90Opcodes::EN_HEAD_TRAILER
VME::TDCV1x90::SetTDCEncapsulation
VME::TDCV1x90Opcodes::DIS_HEAD_TRAILER
VME::TDCV1x90Opcodes::EN_ALL_CHANNEL

7.21.3.2 VME::TDCV1x90::∼TDCV1x90 ()

7.21.4 Member Function Documentation

7.21.4.1 void VME::TDCV1x90::abort ()

7.21.4.2 void VME::TDCV1x90::CheckConfiguration () const

Here is the call graph for this function:

7.21.4.3 void VME::TDCV1x90::DisableChannel (short *channel_id*) const

Here is the call graph for this function:

7.21.4.4 void VME::TDCV1x90::EnableChannel (short *channel_id*) const

Here is the call graph for this function:

7.21.4.5 TDCEventCollection VME::TDCV1x90::FetchEvents ()

Here is the call graph for this function:

7.21.4.6 AcquisitionMode VME::TDCV1x90::GetAcquisitionMode () `[inline]`

Here is the call graph for this function:

7.21.4.7 uint16_t VME::TDCV1x90::GetBLTEventNumberRegister () const

Here is the call graph for this function:

7.21.4.8 unsigned short VME::TDCV1x90::GetChannelDeadTime () const

Here is the call graph for this function:

7.21.4.9 TDCV1x90Control VME::TDCV1x90::GetControl () const

Here is the call graph for this function:

7.21.4.10 DetectionMode VME::TDCV1x90::GetDetectionMode () `[inline]`

Here is the call graph for this function:

7.21.4.11 DLLMode VME::TDCV1x90::GetDLLClock () const

7.21.4.12 bool VME::TDCV1x90::GetErrorMarks () const [inline]

7.21.4.13 bool VME::TDCV1x90::GetETTT () const [inline]

Here is the call graph for this function:

7.21.4.14 uint32_t VME::TDCV1x90::GetEventCounter () const

Number of occured triggers. Number of acquired events since the latest module's reset/clear; this counter works in trigger Matching Mode only.

Here is the call graph for this function:

7.21.4.15 uint16_t VME::TDCV1x90::GetEventStored () const

Number of events currently stored in the output buffer.

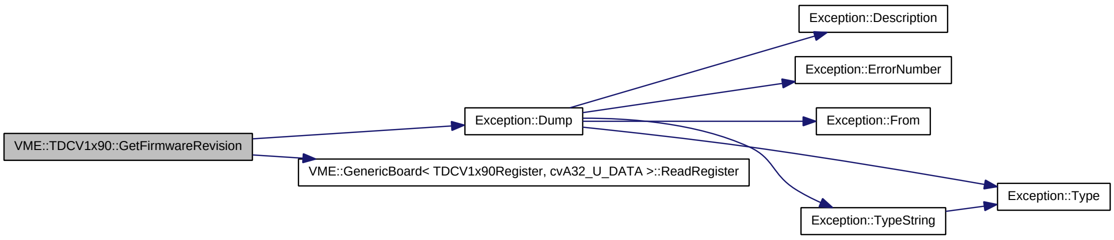
Here is the call graph for this function:

7.21.4.16 uint16_t VME::TDCV1x90::GetFIFOSize () const

Here is the call graph for this function:

7.21.4.17 uint16_t VME::TDCV1x90::GetFirmwareRevision () const

Here is the call graph for this function:



7.21.4.18 GlobalOffset VME::TDCV1x90::GetGlobalOffset () const

Here is the call graph for this function:

7.21.4.19 uint32_t VME::TDCV1x90::GetModel () const

Here is the call graph for this function:

7.21.4.20 uint32_t VME::TDCV1x90::GetOUI () const

Here is the call graph for this function:

7.21.4.21 std::map< unsigned short, bool > VME::TDCV1x90::GetPoI () const

Here is the call graph for this function:

7.21.4.22 uint16_t VME::TDCV1x90::GetRCAdjust (int *tdc*) const

Here is the call graph for this function:

7.21.4.23 uint16_t VME::TDCV1x90::GetResolution () const

Here is the call graph for this function:

7.21.4.24 uint32_t VME::TDCV1x90::GetSerialNumber () const

Here is the call graph for this function:

7.21.4.25 TDCV1x90Status VME::TDCV1x90::GetStatus () const

Here is the call graph for this function:

7.21.4.26 bool VME::TDCV1x90::GetTDCEncapsulation () const

Here is the call graph for this function:

7.21.4.27 bool VME::TDCV1x90::GetTestMode () const

7.21.4.28 uint16_t VME::TDCV1x90::GetTriggerConfiguration (const trig_conf & *type*) const

Here is the call graph for this function:

7.21.4.29 int16_t VME::TDCV1x90::GetWindowOffset () const

7.21.4.30 uint16_t VME::TDCV1x90::GetWindowWidth () const `[inline]`

7.21.4.31 bool VME::TDCV1x90::HardwareReset () const

7.21.4.32 void VME::TDCV1x90::ReadAcquisitionMode () `[private]`

Here is the call graph for this function:

7.21.4.33 void VME::TDCV1x90::ReadDetectionMode () `[private]`

Here is the call graph for this function:

7.21.4.34 void VME::TDCV1x90::SetAcquisitionMode (const AcquisitionMode & *mode*)

Here is the call graph for this function:

7.21.4.35 void VME::TDCV1x90::SetBLTEventNumberRegister (const uint16_t & *value*) const

Here is the call graph for this function:

7.21.4.36 void VME::TDCV1x90::SetChannelDeadTime (unsigned short *dt*) const

Here is the call graph for this function:

7.21.4.37 void VME::TDCV1x90::SetContinuousStorage ()

Here is the call graph for this function:

7.21.4.38 void VME::TDCV1x90::SetControl (const TDCV1x90Control & *control*) const

Here is the call graph for this function:

7.21.4.39 void VME::TDCV1x90::SetDetectionMode (const DetectionMode & *detm*)

Here is the call graph for this function:

7.21.4.40 void VME::TDCV1x90::SetDLLClock (const DLLMode & *dll*) const

Here is the call graph for this function:

7.21.4.41 void VME::TDCV1x90::SetErrorMarks (bool *mode* = `true`)

Here is the call graph for this function:

7.21.4.42 void VME::TDCV1x90::SetETTT (bool *ettt* = `true`) const `[inline]`

Here is the call graph for this function:

7.21.4.43 void VME::TDCV1x90::SetFIFOSize (const uint16_t & *size*) const

Here is the call graph for this function:

7.21.4.44 void VME::TDCV1x90::SetGlobalOffset (const GlobalOffset & *offs*) const

Here is the call graph for this function:

7.21.4.45 void VME::TDCV1x90::SetLSBTraileadEdge (trailead_edge_lsb *conf*) const

Here is the call graph for this function:

7.21.4.46 void VME::TDCV1x90::SetPairModeResolution (int *lead_time_res*, int *pulse_width_res*) const

Here is the call graph for this function:

7.21.4.47 void VME::TDCV1x90::SetPoI (uint16_t *word1*, uint16_t *word2*) const

Here is the call graph for this function:

7.21.4.48 void VME::TDCV1x90::SetRCAdjust (int *tdc*, uint16_t *value*) const

Here is the call graph for this function:

7.21.4.49 void VME::TDCV1x90::SetStatus (const TDCV1x90Status & *status*) const

Here is the call graph for this function:

7.21.4.50 void VME::TDCV1x90::SetTDCEncapsulation (bool *mode*) const

Here is the call graph for this function:

7.21.4.51 void VME::TDCV1x90::SetTestMode (bool *en* = `true`) const

Here is the call graph for this function:

7.21.4.52 void VME::TDCV1x90::SetTriggerMatching ()

Here is the call graph for this function:

7.21.4.53 void VME::TDCV1x90::SetVerboseLevel (unsigned short *verb* = `1`) `[inline]`

7.21.4.54 void VME::TDCV1x90::SetWindowOffset (const int16_t & *offs*) const

Set the offset of the match window with respect to the trigger itself, i.e. the time difference (expressed in clock cycles) between the start of the match window and the trigger time

Parameters:

← ***Window*** offset, in units of clock cycles

Here is the call graph for this function:

7.21.4.55 void VME::TDCV1x90::SetWindowWidth (const uint16_t & *width*)

Set the width of the match window (in number of clock cycles)

Parameters:

← ***Window*** width, in units of clock cycles

Here is the call graph for this function:

7.21.4.56 bool VME::TDCV1x90::SoftwareClear () const

Here is the call graph for this function:

7.21.4.57 bool VME::TDCV1x90::SoftwareReset () const

Here is the call graph for this function:

7.21.4.58 bool VME::TDCV1x90::WaitMicro (micro_handshake *mode*) const `[private]`

Here is the call graph for this function:

7.21.5 Field Documentation

7.21.5.1 AcquisitionMode VME::TDCV1x90::fAcquisitionMode `[private]`

7.21.5.2 uint32_t* VME::TDCV1x90::fBuffer `[private]`

7.21.5.3 DetectionMode VME::TDCV1x90::fDetectionMode `[private]`

7.21.5.4 bool VME::TDCV1x90::fErrorMarks `[private]`

7.21.5.5 unsigned short VME::TDCV1x90::fVerb `[private]`

7.21.5.6 uint16_t VME::TDCV1x90::fWindowWidth `[private]`

7.21.5.7 bool VME::TDCV1x90::gEnd `[private]`

7.21.5.8 uint32_t VME::TDCV1x90::nchannels `[private]`

7.21.5.9 std::string VME::TDCV1x90::pair_lead_res[8] `[private]`

7.21.5.10 std::string VME::TDCV1x90::pair_width_res[16] `[private]`

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

- include/VME_TDCV1x90.h
- src/VME_TDCV1x90.cpp

7.22 VME::TDCV1x90Control Class Reference

TDC control register.

```
#include <VME_TDCV1x90.h>
```

Public Member Functions

- TDCV1x90Control (const uint16_t &word)
- virtual ~TDCV1x90Control ()
- void Dump () const
- uint16_t GetValue () const
- bool GetBusError () const
- void SetBusError (bool sw)
- bool GetTermination () const
- void SetTermination (bool sw)
- bool GetSWTermination () const
- void SetSWTermination (bool sw)
- bool GetEmptyEvent () const
- void SetEmptyEvent (bool sw)
- bool GetAlign64 () const
- void SetAlign64 (bool sw)
- bool GetCompensation () const
- void SetCompensation (bool sw)
- bool GetTestFIFO () const
- void SetTestFIFO (bool sw)
- bool GetSRAMCompensation () const
- void SetSRAMCompensation (bool sw)
- bool GetEventFIFO () const
- void SetEventFIFO (bool sw)
- bool GetETTT () const
- void SetETTT (bool sw)
- bool GetMEBAccess () const
- void SetMEBAccess (bool sw)

Private Attributes

- uint16_t fWord

7.22.1 Detailed Description

TDC control register.

Author:

Laurent Forthomme <laurent.forthomme@cern.ch>

Date:

Jun 2015

7.22.2 Constructor & Destructor Documentation

7.22.2.1 VME::TDCV1x90Control::TDCV1x90Control (const uint16_t & *word*) `[inline]`

7.22.2.2 virtual VME::TDCV1x90Control::∼TDCV1x90Control () `[inline, virtual]`

7.22.3 Member Function Documentation

7.22.3.1 void VME::TDCV1x90Control::Dump () const `[inline]`

Here is the call graph for this function:

7.22.3.2 bool VME::TDCV1x90Control::GetAlign64 () const `[inline]`

7.22.3.3 bool VME::TDCV1x90Control::GetBusError () const `[inline]`

7.22.3.4 bool VME::TDCV1x90Control::GetCompensation () const `[inline]`

7.22.3.5 bool VME::TDCV1x90Control::GetEmptyEvent () const `[inline]`

7.22.3.6 bool VME::TDCV1x90Control::GetETTT () const `[inline]`

7.22.3.7 bool VME::TDCV1x90Control::GetEventFIFO () const `[inline]`

7.22.3.8 bool VME::TDCV1x90Control::GetMEBAccess () const `[inline]`

7.22.3.9 bool VME::TDCV1x90Control::GetSRAMCompensation () const `[inline]`

7.22.3.10 bool VME::TDCV1x90Control::GetSWTermination () const `[inline]`

7.22.3.11 bool VME::TDCV1x90Control::GetTermination () const `[inline]`

7.22.3.12 bool VME::TDCV1x90Control::GetTestFIFO () const `[inline]`

7.22.3.13 uint16_t VME::TDCV1x90Control::GetValue () const `[inline]`

7.22.3.14 void VME::TDCV1x90Control::SetAlign64 (bool *sw*) `[inline]`

Here is the call graph for this function:

7.22.3.15 void VME::TDCV1x90Control::SetBusError (bool *sw*) `[inline]`

Here is the call graph for this function:

7.22.3.16 void VME::TDCV1x90Control::SetCompensation (bool *sw*) `[inline]`

Here is the call graph for this function:

7.22.3.17 void VME::TDCV1x90Control::SetEmptyEvent (bool *sw*) `[inline]`

Here is the call graph for this function:

7.22.3.18 void VME::TDCV1x90Control::SetETTT (bool *sw*) `[inline]`

Here is the call graph for this function:

7.22.3.19 void VME::TDCV1x90Control::SetEventFIFO (bool *sw*) `[inline]`

Here is the call graph for this function:

7.22.3.20 void VME::TDCV1x90Control::SetMEBAccess (bool *sw*) `[inline]`

Here is the call graph for this function:

7.22.3.21 void VME::TDCV1x90Control::SetSRAMCompensation (bool *sw*) `[inline]`

Here is the call graph for this function:

7.22.3.22 void VME::TDCV1x90Control::SetSWTermination (bool *sw*) `[inline]`

Here is the call graph for this function:

7.22.3.23 void VME::TDCV1x90Control::SetTermination (bool *sw*) `[inline]`

Here is the call graph for this function:

7.22.3.24 void VME::TDCV1x90Control::SetTestFIFO (bool *sw*) [inline]

Here is the call graph for this function:

7.22.4 Field Documentation

7.22.4.1 uint16_t VME::TDCV1x90Control::fWord [private]

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

- include/VME_TDCV1x90.h

7.23 VME::TDCV1x90Status Class Reference

TDC status register.

`#include <VME_TDCV1x90.h>`

Public Types

- enum TDCResolution { R_800ps = 0x0, R_200ps = 0x1, R_100ps = 0x2, R_25ps = 0x3 }

Public Member Functions

- TDCV1x90Status (const uint16_t &word)
- virtual ~TDCV1x90Status ()
- void Dump () const
- uint16_t GetValue () const
- bool DataReady () const
- bool AlmostFull () const
- bool Full () const
- bool TriggerMatching () const
- bool HeadersEnabled () const
- bool TerminationOn () const
- bool Error (const unsigned int &id) const
- bool Error () const
- bool BusError () const
- bool Purged () const
- TDCResolution Resolution () const
- bool PairMode () const
- bool TriggerLost () const

Private Attributes

- uint16_t fWord

7.23.1 Detailed Description

TDC status register.

Author:

Laurent Forthomme <`laurent.forthomme@cern.ch`>

Date:

Jun 2015

7.23.2 Member Enumeration Documentation

7.23.2.1 enum VME::TDCV1x90Status::TDCResolution

Enumerator:

R_800ps

R_200ps

R_100ps

R_25ps

7.23.3 Constructor & Destructor Documentation

7.23.3.1 VME::TDCV1x90Status::TDCV1x90Status (const uint16_t & *word*) `[inline]`

7.23.3.2 virtual VME::TDCV1x90Status::∼TDCV1x90Status () `[inline, virtual]`

7.23.4 Member Function Documentation

7.23.4.1 bool VME::TDCV1x90Status::AlmostFull () const `[inline]`

7.23.4.2 bool VME::TDCV1x90Status::BusError () const `[inline]`

7.23.4.3 bool VME::TDCV1x90Status::DataReady () const `[inline]`

7.23.4.4 void VME::TDCV1x90Status::Dump () const `[inline]`

Here is the call graph for this function:

7.23.4.5 bool VME::TDCV1x90Status::Error () const `[inline]`

Here is the call graph for this function:

7.23.4.6 bool VME::TDCV1x90Status::Error (const unsigned int & *id*) const [inline]

7.23.4.7 bool VME::TDCV1x90Status::Full () const [inline]

7.23.4.8 uint16_t VME::TDCV1x90Status::GetValue () const [inline]

7.23.4.9 bool VME::TDCV1x90Status::HeadersEnabled () const [inline]

7.23.4.10 bool VME::TDCV1x90Status::PairMode () const [inline]

7.23.4.11 bool VME::TDCV1x90Status::Purged () const [inline]

7.23.4.12 TDCResolution VME::TDCV1x90Status::Resolution () const [inline]

7.23.4.13 bool VME::TDCV1x90Status::TerminationOn () const [inline]

7.23.4.14 bool VME::TDCV1x90Status::TriggerLost () const [inline]

7.23.4.15 bool VME::TDCV1x90Status::TriggerMatching () const [inline]

7.23.5 Field Documentation

7.23.5.1 uint16_t VME::TDCV1x90Status::fWord [private]

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

- include/VME_TDCV1x90.h

7.24 VME::trailead_t Struct Reference

```
#include <VME_TDCV1x90.h>
```

Data Fields

- uint32_t event_count
- int total_hits [16]
- std::multimap< int32_t, int32_t > leading
- std::multimap< int32_t, int32_t > trailing
- uint32_t ettt

7.24.1 Field Documentation

7.24.1.1 uint32_t VME::trailead_t::ettt

7.24.1.2 uint32_t VME::trailead_t::event_count

7.24.1.3 std::multimap<int32_t,int32_t> VME::trailead_t::leading

7.24.1.4 int VME::trailead_t::total_hits[16]

7.24.1.5 std::multimap<int32_t,int32_t> VME::trailead_t::trailing

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

- include/VME_TDCV1x90.h

7.25 VMEReader Class Reference

`#include <VMEReader.h>`Inheritance diagram for VMEReader:Collaboration diagram for VMEReader:

Public Member Functions

- VMEReader (const char ∗device, VME::BridgeType type, bool on_socket=true)
- virtual ∼VMEReader ()
- void AddTDC (uint32_t address)

  Add a TDC to handle.

- VME::TDCV1x90 ∗ GetTDC (uint32_t address)

  Get a TDC on the VME bus Return a pointer to the TDC object, given its physical address on the VME bus.

- void AddIOModule (uint32_t address)
- VME::IOModuleV262 ∗ GetIOModule ()
- void AddFPGAUnit (uint32_t address)

  Add a multi-purposes FPGA board (CAEN V1495) to the crate controller.

- VME::FPGAUnitV1495 ∗ GetFPGAUnit ()

  Return the pointer to the FPGA board connected to this controller (if any ; 0 otherwise).

- unsigned int GetRunNumber ()

  Ask the socket master a run number.

- void StartPulser (double period, double width, unsigned int num_pulses=0)

  Start the bridge's pulse generator [faulty].

- void StopPulser ()

  Stop the bridge's pulse generator [faulty].

- void SendPulse (unsigned short output=0) const

  Send a single pulse to the output register/plug connected to TDC boards.

- void SendClear () const

  Send a clear signal to both the TDC boards.

- void SetOutputFile (std::string filename)

  Set the path to the output file where the DAQ is to write.

- std::string GetOutputFile () const

  Return the path to the output file the DAQ is currently writing to.

- void Abort ()

  Abort data collection for all modules on the bus handled by the bridge.

Private Types

- typedef std::map< uint32_t, VME::TDCV1x90 ∗ > TDCCollection

  Mapper from physical VME addresses to pointers to TDC objects.

Private Attributes

- VME::BridgeVx718 ∗ fBridge

  The VME bridge object to handle.

- TDCCollection fTDCCollection

  A set of pointers to TDC objects indexed by their physical VME address.

- VME::IOModuleV262 ∗ fSG

  Pointer to the VME input/output module object.

- VME::FPGAUnitV1495 ∗ fFPGA

  Pointer to the VME general purpose FPGA unit object.

- bool fOnSocket

  Are we dealing with socket message passing?

- bool fIsPulserStarted

  Is the bridge's pulser already started?

- std::string fOutputFile

  Path to the current output file the DAQ is writing to.

7.25.1 Detailed Description

VME reader object to fetch events on a HPTDC board

Author:

Laurent Forthomme <laurent.forthomme@cern.ch>

Date:

4 May 2015

7.25.2 Member Typedef Documentation

7.25.2.1 typedef std::map<uint32_t,VME::TDCV1x90∗> VMEReader::TDCCollection `[private]`

Mapper from physical VME addresses to pointers to TDC objects.

7.25.3 Constructor & Destructor Documentation

7.25.3.1 VMEReader::VMEReader (const char ∗ *device*, VME::BridgeType *type*, bool *on_socket* = `true`)

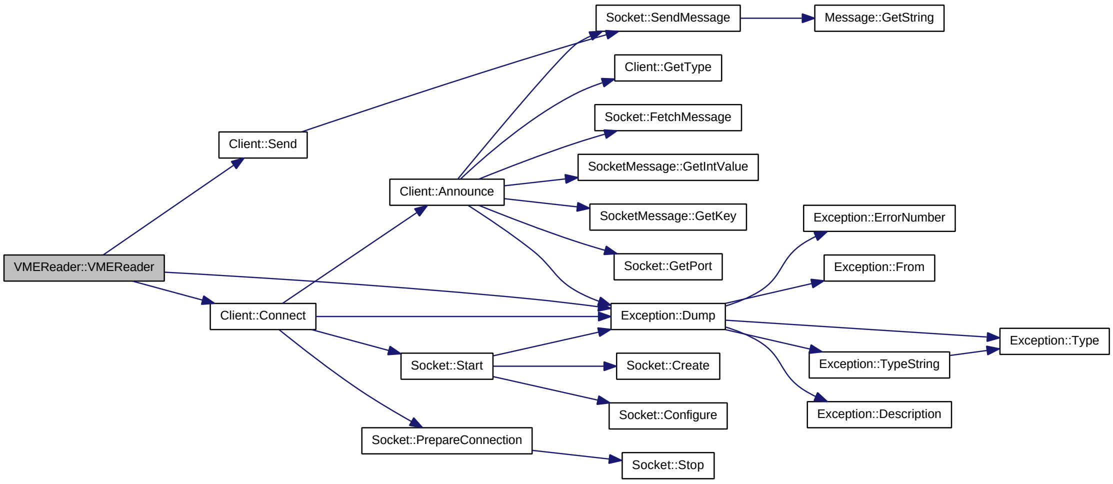
Parameters:

← ***device*** Path to the device (/dev/xxx)

← ***type*** Bridge model

← ***on_socket*** Are we trying to connect through the socket?

Here is the call graph for this function:



7.25.3.2 VMEReader::∼VMEReader () `[virtual]`

Here is the call graph for this function:

7.25.4 Member Function Documentation

7.25.4.1 void VMEReader::Abort ()

Abort data collection for all modules on the bus handled by the bridge.

Here is the call graph for this function:

7.25.4.2 void VMEReader::AddFPGAUnit (uint32_t *address*)

Add a multi-purposes FPGA board (CAEN V1495) to the crate controller.

Parameters:

← ***address*** 32-bit address of the TDC module on the VME bus

Here is the call graph for this function:

7.25.4.3 void VMEReader::AddIOModule (uint32_t *address*)

Here is the call graph for this function:

7.25.4.4 void VMEReader::AddTDC (uint32_t *address*)

Add a TDC to handle.

Parameters:

← ***address*** 32-bit address of the TDC module on the VME bus Create a new TDC handler for the VME bus

Here is the call graph for this function:

7.25.4.5 VME::FPGAUnitV1495∗ VMEReader::GetFPGAUnit () `[inline]`

Return the pointer to the FPGA board connected to this controller (if any ; 0 otherwise).

7.25.4.6 VME::IOModuleV262∗ VMEReader::GetIOModule () `[inline]`

7.25.4.7 std::string VMEReader::GetOutputFile () const `[inline]`

Return the path to the output file the DAQ is currently writing to.

7.25.4.8 unsigned int VMEReader::GetRunNumber ()

Ask the socket master a run number.

Here is the call graph for this function:

7.25.4.9 VME::TDCV1x90∗ VMEReader::GetTDC (uint32_t *address*) `[inline]`

Get a TDC on the VME bus Return a pointer to the TDC object, given its physical address on the VME bus.

7.25.4.10 void VMEReader::SendClear () const `[inline]`

Send a clear signal to both the TDC boards.

Here is the call graph for this function:

7.25.4.11 void VMEReader::SendPulse (unsigned short *output* = `0`) const `[inline]`

Send a single pulse to the output register/plug connected to TDC boards.

Here is the call graph for this function:

7.25.4.12 void VMEReader::SetOutputFile (std::string *filename*) `[inline]`

Set the path to the output file where the DAQ is to write.

7.25.4.13 void VMEReader::StartPulser (double *period*, double *width*, unsigned int *num_pulses* = `0`) `[inline]`

Start the bridge's pulse generator [faulty].

Here is the call graph for this function:

7.25.4.14 void VMEReader::StopPulser () `[inline]`

Stop the bridge's pulse generator [faulty].

Here is the call graph for this function:

7.25.5 Field Documentation

7.25.5.1 VME::BridgeVx718∗ VMEReader::fBridge `[private]`

The VME bridge object to handle.

7.25.5.2 VME::FPGAUnitV1495∗ VMEReader::fFPGA `[private]`

Pointer to the VME general purpose FPGA unit object.

7.25.5.3 bool VMEReader::fIsPulserStarted [private]

Is the bridge's pulser already started?

7.25.5.4 bool VMEReader::fOnSocket [private]

Are we dealing with socket message passing?

7.25.5.5 std::string VMEReader::fOutputFile [private]

Path to the current output file the DAQ is writing to.

7.25.5.6 VME::IOModuleV262* VMEReader::fSG [private]

Pointer to the VME input/output module object.

7.25.5.7 TDCCollection VMEReader::fTDCCollection [private]

A set of pointers to TDC objects indexed by their physical VME address.

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

- include/VMEReader.h
- src/VMEReader.cpp

Index