CT-PPS Motherboard registers library

Generated by Doxygen 1.8.11

Contents

Chapter 1

Hierarchical Index

1.1 Class Hierarchy

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

Chapter 2

Class Index

2.1 Class List

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

Chapter 3

Class Documentation

3.1 PPSTimingMB::BoardAddress Struct Reference

Content holder for a motherboard address.

```
#include <BoardAddress.h>
```

Public Member Functions

- BoardAddress (unsigned int mfec, unsigned int ccu, unsigned int i2c)

  Construct an object from the full address content.
- void Dump (std::ostream &out=std::cout) const

  Dump the full address into the output stream.

Public Attributes

- unsigned int mfec

  FEC identifier.
- unsigned int ccu

  CCU ring identifier.
- unsigned int i2c

  I2C address.

3.1.1 Detailed Description

Content holder for a motherboard address.

Author

Laurent Forthomme `laurent.forthomme@cern.ch`

Date

30 Jun 2016

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

- include/BoardAddress.h

3.2 PPSTimingMB::TDCStatus::ErrorType Struct Reference

Type of error encountered by the HPTDC.

```
#include <TDCStatus.h>
```

Public Member Functions

- **ErrorType** (uint16_t word)
- bool ParityError () const

  Error related on the parity of any register/buffer.
- bool MeasurementError () const

  Error related to the Vernier or Coarse measurement.
- bool GlobalError () const

  Has any error occured?

Public Attributes

- bool **Vernier**
- bool **Coarse**
- bool **ChannelSelect**
- bool **L1BufferParity**
- bool **TriggerFIFOParity**
- bool **TriggerMatchingState**
- bool **ReadoutFIFOParity**
- bool **ReadoutState**
- bool **SetupParity**
- bool **ControlParity**
- bool **JTAGInstruction**

3.2.1 Detailed Description

Type of error encountered by the HPTDC.

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

- include/TDCStatus.h

3.3 PPSTimingMB::TDCInternalCoreTest::HitData Struct Reference

Public Attributes

- uint32_t **first_vernier**
- uint32_t **first_coarse1**
- uint32_t **first_coarse2**
- bool **first_coarse1_parity**
- bool **first_coarse2_parity**
- bool **first_edge_type**
- uint32_t **second_vernier**
- uint32_t **second_coarse1**
- uint32_t **second_coarse2**
- bool **second_coarse1_parity**
- bool **second_coarse2_parity**
- bool **second_edge_type**

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

- include/TDCInternalCoreTest.h

3.4 PPSTimingMB::TDCInternalCoreTest::L1Data Struct Reference

Public Attributes

- uint16_t **edge_fine_time**
- uint16_t **edge_coarse_time**
- bool **edge_type**
- uint16_t **width**
- uint16_t **channel**
- bool **error**
- bool **overflow_start**
- bool **overflow_stop**
- bool **separator**
- bool **parity**

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

- include/TDCInternalCoreTest.h

3.5 PPSTimingMB::NINOThresholds Struct Reference

Thresholds for all 8-channel NINO boards on a PPS motherboard.

```
#include <NINOThresholds.h>
```

Public Member Functions

- NINOThresholds (unsigned int, unsigned int, unsigned int, unsigned int)

  Construct the object out of the four threshold values.
- void Dump (std::ostream &os=std::cout)

  Dump the threshold for all 4 groups into the output stream.

Public Attributes

- unsigned int group0

  NINO threshold for channels 0-7.
- unsigned int group1

  NINO threshold for channels 8-15.
- unsigned int group2

  NINO threshold for channels 16-23.
- unsigned int group3

  NINO threshold for channels 24-31.

3.5.1 Detailed Description

Thresholds for all 8-channel NINO boards on a PPS motherboard.

Author

Laurent Forthomme `laurent.forthomme@cern.ch`

Date

30 Jun 2016

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

- include/NINOThresholds.h

3.6 PPSTimingMB::XMLHandler::PropertiesMap Class Reference

A map of properties retrieved from a parsed XML file.

```
#include <XMLHandler.h>
```

Public Member Functions

- void AddProperty (const char ∗name, const char ∗value)

  Feed a new key/value property to the map.
- bool HasProperty (const char ∗name)

  Check if a key is present in the map.
- std::string GetProperty (const char ∗name)

  Retrieve the (string) value associated with a key.
- unsigned int GetUIntProperty (const char ∗name)

  Retrieve the (unsigned integer) value associated with a key.

3.6.1 Detailed Description

A map of properties retrieved from a parsed XML file.

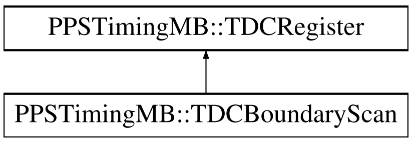
The documentation for this class was generated from the following file:

- include/XMLHandler.h

3.7 PPSTimingMB::TDCBoundaryScan Class Reference

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

Inheritance diagram for PPSTimingMB::TDCBoundaryScan:



Public Member Functions

- **TDCBoundaryScan** (const TDCBoundaryScan &bs)
- **TDCBoundaryScan** (const std::vector< uint8_t > &words)
- bool **IsTokenOut** () const
- bool **IsStrobeOut** () const
- bool **IsSerialOut** () const
- bool **IsTest** () const
- bool **IsError** () const
- bool **IsDataReady** () const
- bool **IsParallelEnabled** () const
- bool **HasParallelDataOut** (unsigned short channel_id) const
- bool **IsEncodedControl** () const
- bool **IsTrigger** () const
- bool **HasTrigger** () const
- bool **HasEventReset** () const
- bool **HasBunchReset** () const
- bool **IsGettingData** () const
- bool **IsSerialBypassIn** () const
- bool **IsSerialIn** () const
- bool **IsTokenBypassIn** () const
- bool **IsTokenIn** () const
- bool **IsReset** () const
- bool **HasAuxiliaryClock** () const
- bool **HasClock** () const
- bool **HasHit** (unsigned short channel_id) const
- void SetConstantValues ()

  Set all hardcoded values to this register.
- void Dump () const

  Printout all useful values of this status register into an output stream.

Additional Inherited Members

3.7.1 Detailed Description

Author

Laurent Forthomme laurent.forthomme@cern.ch

Date

24 Apr 2015
May 2016

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

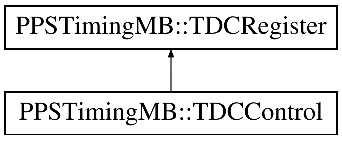
- include/TDCBoundaryScan.h

3.8 PPSTimingMB::TDCControl Class Reference

Control word to be sent to the HPTDC chip.

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

Inheritance diagram for PPSTimingMB::TDCControl:



Public Types

- enum **EnablePattern** { **OutputEnabled** =0x5, **OutputDisabled** =0x4 }
- typedef enum PPSTimingMB::TDCControl::EnablePattern **EnablePattern**

Public Member Functions

- **TDCControl** (const TDCControl &c)
- **TDCControl** (const std::vector< uint8_t > &words)
- void **SetEnablePattern** (const EnablePattern &ep=OutputEnabled)
- EnablePattern **GetEnablePattern** () const
- void **SetGlobalReset** (const bool gr=true)
- bool **GetGlobalReset** () const
- void **SetDLLReset** (const bool dr=true)
- bool **GetDLLReset** () const
- void **SetPLLReset** (const bool pr=true)
- bool **GetPLLReset** () const
- void **EnableChannel** (unsigned int id)
- void **EnableAllChannels** ()
- void **DisableChannel** (unsigned int id)
- void **DisableAllChannels** ()
- bool **IsChannelEnabled** (unsigned int id) const
- void **SetEnabledChannels** (uint32_t ch)
- void **SetEnabledChannels** (uint16_t group0, uint16_t group1)
- void **SetEnabledChannelsGroup0** (uint16_t poi)
- void **SetEnabledChannelsGroup1** (uint16_t poi)
- uint16_t **GetEnabledChannelsGroup0** () const
- uint16_t **GetEnabledChannelsGroup1** () const
- uint32_t **GetEnabledChannels** () const
- void **SetParity** (const bool cp=true)
- bool **GetParity** () const
- void **ComputeParity** ()
- void Dump (int verb=1, std::ostream &os=std::cout) const

  Printout all useful values of this control register into an output stream.
- void **SetConstantValues** ()
- uint32_t **GetValue** (const TDCControlRegister &v)

Additional Inherited Members

3.8.1 Detailed Description

Control word to be sent to the HPTDC chip.

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

Author

Laurent Forthomme `laurent.forthomme@cern.ch`
Lara Lloret Iglesias `lara@cern.ch`

Date

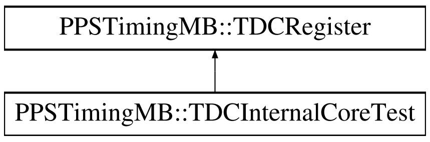
24 Apr 2015

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

- include/TDCControl.h

3.9 PPSTimingMB::TDCInternalCoreTest Class Reference

Inheritance diagram for PPSTimingMB::TDCInternalCoreTest:



Classes

- struct HitData
- struct L1Data
- struct TriggerData

Public Types

- enum **CommonMatchingState** {
  cmsIdle = 0x1, **cmsHeader** = 0x2, **cmsLostHeader** = 0x4, **cmsError** = 0x8,
  cmsTrailer = 0x10, **cmsLostTrailer** = 0x20, **cmsSeparator** = 0x40, **cmsOccupancy** = 0x80,
  cmsMatching = 0x100 }
- enum **MatchingState** {
  msInvalid = 0x0, **msIdle** = 0x1, **msWriteOccupancy** = 0x2, **msActive** = 0x4,
  msWaitingForSeparator = 0x8, **msWaitEnd** = 0x10 }

Public Member Functions

- **TDCInternalCoreTest** (const TDCInternalCoreTest &c)
- **TDCInternalCoreTest** (const std::vector< uint8_t > &words)
- CommonMatchingState **GetCommonMatchingState** () const
- TriggerData **GetTriggerData** () const
- MatchingState **GetMatchingState** (unsigned short group_id) const
- MatchingState **GetMatchingStateGroup3** () const
- MatchingState **GetMatchingStateGroup2** () const
- MatchingState **GetMatchingStateGroup1** () const
- MatchingState **GetMatchingStateGroup0** () const
- L1Data **GetL1Data** (unsigned short group_id) const
- L1Data **GetL1DataGroup3** () const
- L1Data **GetL1DataGroup2** () const
- L1Data **GetL1DataGroup1** () const
- L1Data **GetL1DataGroup0** () const
- bool **GetL1Empty** (unsigned short group_id) const
- bool **GetL1EmptyGroup3** () const
- bool **GetL1EmptyGroup2** () const
- bool **GetL1EmptyGroup1** () const
- bool **GetL1EmptyGroup0** () const
- bool **GetL1Ready** (unsigned short group_id) const
- bool **GetL1ReadyGroup3** () const
- bool **GetL1ReadyGroup2** () const
- bool **GetL1ReadyGroup1** () const
- bool **GetL1ReadyGroup0** () const
- HitData **GetHitData** (unsigned short group_id) const
- HitData **GetHitDataGroup3** () const
- HitData **GetHitDataGroup2** () const
- HitData **GetHitDataGroup1** () const
- HitData **GetHitDataGroup0** () const
- uint16_t **GetHitChannel** (unsigned short group_id) const
- uint16_t **GetHitChannelGroup3** () const
- uint16_t **GetHitChannelGroup2** () const
- uint16_t **GetHitChannelGroup1** () const
- uint16_t **GetHitChannelGroup0** () const
- bool **GetHitSelectError** (unsigned short group_id) const
- bool **GetHitSelectErrorGroup3** () const
- bool **GetHitSelectErrorGroup2** () const
- bool **GetHitSelectErrorGroup1** () const
- bool **GetHitSelectErrorGroup0** () const
- bool **GetHitLoad** (unsigned short group_id) const
- bool **GetHitLoadGroup3** () const
- bool **GetHitLoadGroup2** () const
- bool **GetHitLoadGroup1** () const
- bool **GetHitLoadGroup0** () const
- void **Dump** (int verb=1, std::ostream &os=std::cout) const
- void **SetConstantValues** ()

Additional Inherited Members

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

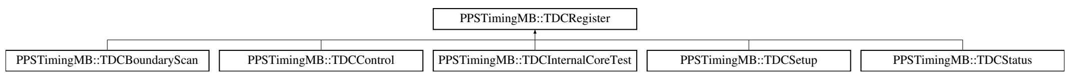
- include/TDCInternalCoreTest.h

3.10 PPSTimingMB::TDCRegister Class Reference

General register object to interact with a HPTDC chip.

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

Inheritance diagram for PPSTimingMB::TDCRegister:



Public Types

- typedef uint16_t bit

  LSB index.
- typedef uint32_t word_t

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

Public Member Functions

- TDCRegister (const unsigned int size)

  Initialise an empty register.
- TDCRegister (const unsigned int size, const TDCRegister &r)

  Initialise and fill a register.
- TDCRegister (const unsigned int size, const std::vector< uint8_t > &words, bool reversed=false)

  Initialise and fill a register.
- virtual ~TDCRegister ()

  Destroy the register and its content.
- TDCRegister & operator= (const TDCRegister &r)

  Assign values from another register to this one.
- void SetWord (const unsigned int i, const word_t word)

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

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

  Retrieve the whole array of sub-words composing this register.
- std::vector< uint8_t > GetBytesVector () const

  Retrieve a vector of 8-bit words composing this register.
- uint8_t GetNumWords () const

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

  Printout all useful information handled by the register.
- void SetConstantValues ()

  Ensure that the critical constant values are properly set in the register word.
- template< class T >
  uint32_t GetValue (const T &)

  Return a given value as a 32-bit word.
- bool ComputeParityBit (unsigned short begin=0, short end=-1) const

  Compute the parity bit of the full register word.

Protected Member Functions

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

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

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

  Set all bits in this register to '0'.

Protected Attributes

- word_t ∗ fWord

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

  Number of words to fit the fWordSize bits of this register to this object.
- unsigned int fWordSize

  Number of bits in this register.

3.10.1 Detailed Description

General register object to interact with a HPTDC chip.

Author

Laurent Forthomme `laurent.forthomme@cern.ch`

Date

24 Apr 2015

3.10.2 Member Function Documentation

3.10.2.1 uint16_t PPSTimingMB::TDCRegister::GetBits (uint16_t *lsb*, uint8_t *size*) const `[protected]`

Extract bits from the register word.

Extract a fixed amount of bits from the full register word

Parameters

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

3.10.2.2 uint8_t PPSTimingMB::TDCRegister::GetNumWords ( ) const `[inline]`

Number of words in the register.

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

3.10.2.3 void PPSTimingMB::TDCRegister::SetBits (uint16_t *lsb,* uint16_t *word,* uint8_t *size*) `[protected]`

Set bits in the register word.

Set a fixed amount of bits in the full register word

Parameters

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

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

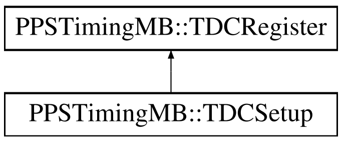
- include/TDCRegister.h

3.11 PPSTimingMB::TDCSetup Class Reference

Setup word to be sent to the HPTDC chip.

`#include <TDCSetup.h>`

Inheritance diagram for PPSTimingMB::TDCSetup:



Public Types

- enum **EdgeResolution** {
  E_100ps =0, **E_200ps**, **E_400ps**, **E_800ps**,
  E_1p6ns, **E_3p12ns**, **E_6p25ns**, **E_12p5ns** }
- enum **DeadTime** { **DT_5ns** =0, **DT_10ns**, **DT_30ns**, **DT_100ns** }
- enum **WidthResolution** {
  W_100ps =0, **W_200ps**, **W_400ps**, **W_800ps**,
  W_1p6ns, **W_3p2ns**, **W_6p25ns**, **W_12p5ns**,
  W_25ns, **W_50ns**, **W_100ns**, **W_200ns**,
  W_400ns, **W_800ns** }
- enum **EnabledError** {
  VernierError =0x1, **CoarseError** =0x2, **ChannelSelectError** =0x4, **L1BufferParityError** =0x8,
  TriggerFIFOParityError =0x10, **TriggerMatchingError** =0x20, **ReadoutFIFOParityError** =0x40,
  ReadoutStateError =0x80,
  SetupParityError =0x100, **ControlParityError** =0x200, **JTAGInstructionParityError** =0x400 }
- enum **DLLSpeedMode** { **DLL_40MHz** =0x0, **DLL_160MHz** =0x1, **DLL_320MHz** =0x2, **DLL_Illegal** =0x3 }
- enum **SerialClockSource** { **Serial_pll_clock_80** =0x0, **Serial_pll_clock_160** =0x1, **Serial_pll_clock_40** =0x2, **Serial_aux_clock** =0x3 }

- enum **IOClockSource** { **IO_clock_40** =0x0, **IO_pll_clock_80** =0x1, **IO_pll_clock_160** =0x2, **IO_aux_clock** =0x3 }
- enum **CoreClockSource** { **Core_clock_40** =0x0, **Core_pll_clock_80** =0x1, **Core_pll_clock_160** =0x2, **Core_aux_clock** =0x3 }
- enum **DLLClockSource** {
DLL_clock_40 =0x0, **DLL_pll_clock_40** =0x1, **DLL_pll_clock_160** =0x2, **DLL_pll_clock_320** =0x3, **DLL_aux_clock** =0x4 }
- enum **ReadoutSpeed** { **RO_Fixed** =0x0, **RO_pll_80Mbits_s** =0x1 }
- enum **SerialStrobeType** { **SS_NoStrobe** =0x0, **SS_DSStrobe** =0x1, **SS_LeadingTrailingStrobe** =0x2, **S**↩ **S_LeadingEdge** =0x3 }
- enum **ReadoutSingleCycleSpeed** {
RSC_40Mbits_s =0x0, **RSC_20Mbits_s** =0x1, **RSC_10Mbits_s** =0x2, **RSC_5Mbits_s** =0x3, **RSC_2p5Mbits_s** =0x4, **RSC_1p25Mbits_s** =0x5, **RSC_625kbits_s** =0x6, **RSC_312p5kbits_s** =0x7 }

Public Member Functions

- **TDCSetup** (const TDCSetup &c)
- **TDCSetup** (const std::vector< uint8_t > &v, bool reverse=false)
- void **SetTest** (const bool test=true)
- bool **IsTest** () const
- void SetEnableErrorMark (const bool em)

  Mark events with error if global error signal is set.
- bool **GetEnableErrorMark** () const
- void SetEnableErrorBypass (const bool eb)

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

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

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

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

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

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

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

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

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

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

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

- void SetMaxEventSize (int sz=-1)

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

  Extract the adjustment of DLL taps.
- void **SetDLLAdjustmentWord** (uint16_t word)
- uint16_t **GetDLLAdjustmentWord** () const
- void SetRCAdjustment (int tap, uint8_t adj)

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

  Extract the adjustment of the RC delay line.
- void **SetRCAdjustmentWord** (uint16_t word)
- uint16_t **GetRCAdjustmentWord** () const
- void SetWidthResolution (const WidthResolution r)

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

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

  Set the offset in vernier decoding.

- uint8_t GetVernierOffset () const

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

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

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

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

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

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

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

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

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

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

  Enable the pairing of leading and trailing edges (overrides individual enable of leading/trailing edges)
- bool **GetEdgesPairing** () const
- void SetParity (const bool sp=true)

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

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

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

  Effective trigger latency in number of clock cycles (when no counter roll-over is used)
- void SetTDCId (const uint8_t id=0x0)

  Set the unique identifier of the TDC object on the board.
- uint16_t GetTDCId () const

  Get the unique identifier of the TDC object on the board.
- void SetEnableTTLSerial (const bool ts=true)

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

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

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

  Enable LV TTL inputs on: clk, aux_clock, otherwise uses LVDS input levels.
- bool **GetEnableTTLClock** () const

- void SetEnableTTLHit (const bool th=true)

  Enable LV TTL input on hit[31:0], otherwise uses LVDS input levels.
- bool **GetEnableTTLHit** () const
- void SetRollOver (const uint16_t ro=0xFFF)

  Counter roll over value, defining maximal count value from where counters will be reset to 0.
- uint16_t **GetRollOver** () const
- void SetPLLControl (const uint8_t charge_pump_current=0x4, const bool power_down_mode=false, const bool enable_test_outputs=false, const bool invert_connection_to_status=false)

  Control of PLL.
- void **SetPLLControlWord** (uint16_t word)
- uint16_t **GetPLLControlWord** () const
- void SetDLLMode (const DLLSpeedMode dsm)

  Selection of DLL speed mode.
- DLLSpeedMode **GetDLLMode** () const
- void SetModeRC (const bool mr=true)

  Enable of RR delay lines mode (in very high resolution mode) ; only for channels 0-4-8-12-16-20-24-28 active.
- bool **GetModeRC** () const
- void SetModeRCCompression (const bool mrc=true)

  Perform RC interpolation on-chip (only valid in very high resolution mode)
- bool **GetModeRCCompression** () const
- void SetEnableRelative (const bool er=true)
- bool **GetEnableRelative** () const
- void SetReadoutSingleCycleSpeed (const ReadoutSingleCycleSpeed rscs=RSC_40Mbits_s)

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

  Programmable delay of serial input, in time unit ∼ 1 ns.
- uint8_t **GetSerialDelay** () const
- void **SetStrobeSelect** (const SerialStrobeType ss=SS_NoStrobe)
- SerialStrobeType **GetStrobeSelect** () const
- void SetReadoutSpeedSelect (const ReadoutSpeed rss=RO_Fixed)

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

  Programmable delay of token input, in time unit ∼ 1 ns.
- uint8_t **GetTokenDelay** () const
- void SetEnableLocalTrailer (const bool elt=true)

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

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

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

  Enable of global headers in read-out (only valid for master TDC)
- bool **GetEnableGlobalHeader** () const
- void SetKeepToken (const bool kt=true)
- bool **GetKeepToken** () const
- void **SetMaster** (const bool m=true)
- bool **GetMaster** () const

- void **SetEnableBytewise** (const bool seb=true)
- bool **GetEnableBytewise** () const
- void SetBypassInputs (const bool sbi=true)

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

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

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

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

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

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

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

  Enable generation of separator on event reset.
- bool **GetEnableSeparatorOnEventReset** () const
- void SetEnableSeparatorOnBunchReset (const bool esobr=true)

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

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

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

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

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

  Control of DLL (DLL charge pump levels)
- uint8_t **GetDLLControl** () const
- void SetSerialClockSource (const SerialClockSource scs)

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

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

  Selection of clock source for internal logic.

- CoreClockSource **GetCoreClockSource** () const
- void SetDLLClockSource (const DLLClockSource dcs)

  Selection of clock source for DLL.
- DLLClockSource **GetDLLClockSource** () const
- void Dump (int verb=1, std::ostream &os=std::cout) const

  Printout all useful values of this setup register into an output stream.
- std::string **GetXML** () const
- uint32_t **GetValue** (const TDCSetupRegister &v)

Additional Inherited Members

3.11.1 Detailed Description

Setup word to be sent to the HPTDC chip.

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

Author

Laurent Forthomme `laurent.forthomme@cern.ch`
Lara Lloret Iglesias `lara@cern.ch`

Date

16 Apr 2015
May 2016

3.11.2 Member Function Documentation

3.11.2.1 bool PPSTimingMB::TDCSetup::GetRejectFIFOFull ( ) const [inline]

Are hits rejected when readout FIFO is full?

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

3.11.2.2 void PPSTimingMB::TDCSetup::SetEnableRelative (const bool *er* = true) [inline]

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

3.11.2.3 void PPSTimingMB::TDCSetup::SetEnableTTLControl (const bool *tc* = true) [inline]

Enable LV TTL inputs on control registers.

Enable LV TTL input on:

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

3.11.2.4 void PPSTimingMB::TDCSetup::SetEnableTTLSerial (const bool *ts* = `true`) `[inline]`

Enable LV TTL inputs on serial registers, and disable their drivers.

Enable LV TTL input on:

- serial_in,
- serial_bypass_in,
- token_in,
- token_bypass_in, otherwise uses LVDS input levels. Disable LVDS drivers on:
- serial_out,
- strobe_out,
- token_out.

3.11.2.5 void PPSTimingMB::TDCSetup::SetKeepToken (const bool *kt* = `true`) `[inline]`

Keep token until end of event or no more data, otherwise pass token after each word read. Must be enabled when using trigger matching.

3.11.2.6 void PPSTimingMB::TDCSetup::SetMaxEventSize (int *sz* = `-1`) `[inline]`

Set the maximum number of hits per event.

Set the maximum number of hits that can be recorded for each event. It is always rounded to the next power of 2 (in the range 0-128), and if lower than 0 or bigger than 128 then set to unimited.

3.11.2.7 void PPSTimingMB::TDCSetup::SetReadoutSpeedSelect (const ReadoutSpeed *rss* = `RO_Fixed`) `[inline]`

Selection of serial read-out speed.

Parameters

`in`	*rss*	• 0: Selection of serial read-out speed (as defined by setup[19:17], *SetReadoutSingleCycleSpeed*) • 1: 80 Mbits/s (PLL lock required)

3.11.2.8 void PPSTimingMB::TDCSetup::SetRejectFIFOFull (const bool *rej* = `true`) `[inline]`

Reject hits when readout FIFO full.

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

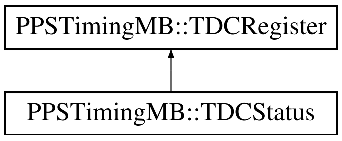
The documentation for this class was generated from the following file:

- include/TDCSetup.h

3.12 PPSTimingMB::TDCStatus Class Reference

```
#include <TDCStatus.h>
```

Inheritance diagram for PPSTimingMB::TDCStatus:



Classes

- struct ErrorType

  Type of error encountered by the HPTDC.

Public Types

- typedef struct PPSTimingMB::TDCStatus::ErrorType ErrorType

  Type of error encountered by the HPTDC.

Public Member Functions

- TDCStatus ()

  Initialise a status register with all hardcoded values.
- TDCStatus (const std::vector< uint8_t > &words)

  Initialise a status register from a vector of 8-bit words.
- void SetConstantValues ()

  Set the hardcoded values to the register.
- ErrorType Error () const

  Retrieve the list of errors monitored.
- bool HasToken () const

  TDC have read-out token.
- uint16_t FIFOOccupancy () const

  Occupancy of readout FIFO.
- bool FIFOFull () const

  It the readout FIFO full?
- bool FIFOEmpty () const

  It the readout FIFO empty?
- uint32_t L1Occupancy (unsigned short group=-1) const

  Occupancy of L1 buffer in channels of a group (or all groups)
- uint16_t TriggerFIFOOccupancy () const

Occupancy of trigger FIFO.
- bool TriggerFIFOFull () const

  Is the trigger FIFO full?
- bool TriggerFIFOEmpty () const

  Is the trigger FIFO empty?
- bool DLLLock () const

  Is the DLL in lock state?
- void Dump (int verb=1, std::ostream &os=std::cout) const

  Printout all useful values of this status register into an output stream.

Friends

- std::ostream & operator<< (std::ostream &out, const ErrorType &err)

  Printout the error type(s) into the output stream.

Additional Inherited Members

3.12.1 Detailed Description

Author

Laurent Forthomme `laurent.forthomme@cern.ch`
Lara Lloret Iglesias `lara@cern.ch`

Date

27 Apr 2015
May 2016

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

- include/TDCStatus.h

3.13 PPSTimingMB::TDCInternalCoreTest::TriggerData Struct Reference

Public Attributes

- uint16_t **bunch_id**
- uint16_t **event_id**
- bool **separator**
- bool **trigger_lost**
- bool **parity**

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

- include/TDCInternalCoreTest.h

3.14 PPSTimingMB::XMLHandler Class Reference

XML input/output handler.

```
#include <XMLHandler.h>
```

Classes

- class PropertiesMap

  A map of properties retrieved from a parsed XML file.

Public Member Functions

- std::string WriteRegister (const TDCControl &r, unsigned int mfec, unsigned int ccu, unsigned int i2c)

  Extract a XML output of a TDCControl register.
- std::string WriteRegister (const TDCControl &r, const BoardAddress &addr)

  Extract a XML output of a TDCControl register.
- bool ReadRegister (std::string str, TDCControl ∗c, unsigned int mfec, unsigned int ccu, unsigned int i2c)

  Parse a TDCControl register out of a XML configuration file.
- bool ReadRegister (std::string, TDCControl ∗c, const BoardAddress &addr)

  Parse a TDCControl register out of a XML configuration file.
- std::string WriteRegister (const TDCSetup &r, unsigned int mfec, unsigned int ccu, unsigned int i2c)

  Extract a XML output of a TDCSetup register.
- std::string WriteRegister (const TDCSetup &r, const BoardAddress &addr)

  Extract a XML output of a TDCSetup register.
- bool ReadRegister (std::string str, TDCSetup ∗s, unsigned int mfec, unsigned int ccu, unsigned int i2c)

  Parse a TDCSetup register out of a XML configuration file.
- bool ReadRegister (std::string, TDCSetup ∗s, const BoardAddress &addr)

  Parse a TDCSetup register out of a XML configuration file.
- std::string WriteRegister (const NINOThresholds &n, unsigned int mfec, unsigned int ccu, unsigned int i2c)

  Extract a XML output of a NINO thresholds register.
- std::string WriteRegister (const NINOThresholds &n, const BoardAddress &addr)

  Extract a XML output of a NINO thresholds register.
- bool ReadRegister (std::string str, NINOThresholds ∗n, unsigned int mfec, unsigned int ccu, unsigned int i2c)

  Parse a NINO thresholds register out of a XML configuration file.
- bool ReadRegister (std::string, NINOThresholds ∗n, const BoardAddress &addr)

  Parse a NINO thresholds register out of a XML configuration file.
- std::string WriteRegister (const TDCControl &c, const TDCSetup &s, unsigned int mfec, unsigned int ccu, unsigned int i2c)

  Extract a XML output of a TDCControl and a TDCSetup register.
- std::string WriteRegister (const TDCControl &c, const TDCSetup &s, const BoardAddress &addr)

  Extract a XML output of a TDCControl and a TDCSetup register.
- std::string WriteRegister (const TDCControl &c, const TDCSetup &s, const NINOThresholds &n, unsigned int mfec, unsigned int ccu, unsigned int i2c)

  Extract a XML output of a TDCControl, a TDCSetup, and a NINO thresholds register.
- std::string WriteRegister (const TDCControl &c, const TDCSetup &s, const NINOThresholds &n, const BoardAddress &addr)

  Extract a XML output of a TDCControl, a TDCSetup, and a NINO thresholds register.

3.14.1 Detailed Description

XML input/output handler.

Author

Laurent Forthomme `laurent.forthomme@cern.ch`

Date

23 May 2016

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

- include/XMLHandler.h

Index