

## 2015 Test beam Run Control

Generated by Doxygen 1.8.9.1

Wed Jun 17 2015 18:57:43



# Contents

<b>1</b>	<b>Module Index</b>	<b>1</b>
1.1	Modules	1
<b>2</b>	<b>Namespace Index</b>	<b>3</b>
2.1	Namespace List	3
<b>3</b>	<b>Hierarchical Index</b>	<b>5</b>
3.1	Class Hierarchy	5
<b>4</b>	<b>Data Structure Index</b>	<b>7</b>
4.1	Data Structures	7
<b>5</b>	<b>Module Documentation</b>	<b>9</b>
5.1	Socket communication objects	9
5.1.1	Detailed Description	9
5.1.2	Enumeration Type Documentation	9
5.1.2.1	SocketType	9
<b>6</b>	<b>Namespace Documentation</b>	<b>11</b>
6.1	VME Namespace Reference	11
6.1.1	Typedef Documentation	12
6.1.1.1	TDCEventCollection	12
6.1.2	Enumeration Type Documentation	12
6.1.2.1	acq_mode	12
6.1.2.2	BridgeType	12
6.1.2.3	ctl_reg	12
6.1.2.4	det_mode	13
6.1.2.5	micro_handshake	13
6.1.2.6	mod_reg	13
6.1.2.7	stat_reg	14
6.1.2.8	trailead_edge_lsb	14
6.1.2.9	trig_conf	14
6.2	VME::TDCV1x90Opcodes Namespace Reference	14

6.2.1	Function Documentation	16
6.2.1.1	AUTOLOAD_DEF_CONFI	16
6.2.1.2	AUTOLOAD_USER_CONF	16
6.2.1.3	CLEAR_KEEP_TOKEN	16
6.2.1.4	CONT_STOR	16
6.2.1.5	DEFAULT_SETUP_REG	16
6.2.1.6	DIS_ALL_CHANNEL	16
6.2.1.7	DIS_CHANNEL	16
6.2.1.8	DIS_ERROR_BYPASS	16
6.2.1.9	DIS_ERROR_MARK	16
6.2.1.10	DIS_HEAD_TRAILER	16
6.2.1.11	DIS_SUB_TRG	16
6.2.1.12	DISABLE_TEST_MODE	16
6.2.1.13	EN_ALL_CHANNEL	16
6.2.1.14	EN_CHANNEL	16
6.2.1.15	EN_ERROR_BYPASS	16
6.2.1.16	EN_ERROR_MARK	16
6.2.1.17	EN_HEAD_TRAILER	16
6.2.1.18	EN_SUB_TRG	16
6.2.1.19	ENABLE_TEST_MODE	16
6.2.1.20	LOAD_DEF_CONFIG	17
6.2.1.21	LOAD_USER_CONFIG	17
6.2.1.22	READ_ACQ_MOD	17
6.2.1.23	READ_ADJUST_CH	17
6.2.1.24	READ_DEAD_TIME	17
6.2.1.25	READ_DETECTION	17
6.2.1.26	READ_DLL_LOCK	17
6.2.1.27	READ_EEPROM	17
6.2.1.28	READ_EN_PATTERN	17
6.2.1.29	READ_EN_PATTERN32	17
6.2.1.30	READ_ERROR_STATUS	17
6.2.1.31	READ_ERROR_TYPES	17
6.2.1.32	READ_EVENT_SIZE	17
6.2.1.33	READ_FIFO_SIZE	17
6.2.1.34	READ_GLOB_OFFS	17
6.2.1.35	READ_HEAD_TRAILER	17
6.2.1.36	READ_MICRO_REV	17
6.2.1.37	READ_RC_ADJ	17
6.2.1.38	READ_RES	17
6.2.1.39	READ_SETUP_REG	17

6.2.1.40	READ_SETUP_SCANPATH	17
6.2.1.41	READ_SPARE	17
6.2.1.42	READ_STATUS_STREAM	17
6.2.1.43	READ_TDC_ID	17
6.2.1.44	READ_TRG_CONF	17
6.2.1.45	RESET_DLL_PLL	17
6.2.1.46	REV_DATE_MICRO_FW	17
6.2.1.47	SAVE_RC_ADJ	17
6.2.1.48	SAVE_USER_CONFIG	18
6.2.1.49	SET_ADJUST_CH	18
6.2.1.50	SET_DEAD_TIME	18
6.2.1.51	SET_DETECTION	18
6.2.1.52	SET_DLL_CLOCK	18
6.2.1.53	SET_ERROR_TYPES	18
6.2.1.54	SET_EVENT_SIZE	18
6.2.1.55	SET_FIFO_SIZE	18
6.2.1.56	SET_GLOB_OFFS	18
6.2.1.57	SET_KEEP_TOKEN	18
6.2.1.58	SET_PAIR_RES	18
6.2.1.59	SET_RC_ADJ	18
6.2.1.60	SET_REJ_MARGIN	18
6.2.1.61	SET_SW_MARGIN	18
6.2.1.62	SET_TDC_TSET_OUTPUT	18
6.2.1.63	SET_TR_LEAD_LSB	18
6.2.1.64	SET_WIN_OFFS	18
6.2.1.65	SET_WIN_WIDTH	18
6.2.1.66	TRG_MATCH	18
6.2.1.67	UPDATE_SETUP_REG	18
6.2.1.68	UPDATE_SETUP_TDC	18
6.2.1.69	WRITE_EEPROM	18
6.2.1.70	WRITE_EN_PATTERN	18
6.2.1.71	WRITE_EN_PATTERN32	18
6.2.1.72	WRITE_SETUP_REG	18
6.2.1.73	WRITE_SPARE	18
<b>7</b>	<b>Data Structure Documentation</b>	<b>19</b>
7.1	VME::BridgeVx718 Class Reference	19
7.1.1	Detailed Description	19
7.1.2	Constructor & Destructor Documentation	20
7.1.2.1	BridgeVx718	20

7.1.2.2	~BridgeVx718	20
7.1.3	Member Function Documentation	20
7.1.3.1	CheckConfiguration	20
7.1.3.2	GetHandle	20
7.1.3.3	InputConf	20
7.1.3.4	InputRead	21
7.1.3.5	OutputConf	21
7.1.3.6	OutputOff	21
7.1.3.7	OutputOn	21
7.1.4	Field Documentation	21
7.1.4.1	fHandle	21
7.1.4.2	fPortMapping	21
7.2	Client Class Reference	21
7.2.1	Detailed Description	23
7.2.2	Constructor & Destructor Documentation	23
7.2.2.1	Client	23
7.2.2.2	Client	23
7.2.2.3	~Client	23
7.2.3	Member Function Documentation	23
7.2.3.1	Announce	23
7.2.3.2	Connect	24
7.2.3.3	Disconnect	24
7.2.3.4	GetType	25
7.2.3.5	ParseMessage	25
7.2.3.6	Receive	25
7.2.3.7	Send	26
7.2.3.8	SendAndReceive	26
7.2.4	Field Documentation	26
7.2.4.1	fClientId	26
7.2.4.2	fIsConnected	26
7.3	Exception Class Reference	26
7.3.1	Detailed Description	27
7.3.2	Constructor & Destructor Documentation	27
7.3.2.1	Exception	27
7.3.2.2	Exception	27
7.3.2.3	~Exception	27
7.3.3	Member Function Documentation	28
7.3.3.1	Description	28
7.3.3.2	Dump	28
7.3.3.3	ErrorNumber	28

7.3.3.4	From	28
7.3.3.5	Type	28
7.3.3.6	TypeString	28
7.3.4	Field Documentation	28
7.3.4.1	fDescription	28
7.3.4.2	fErrorNumber	28
7.3.4.3	fFrom	28
7.3.4.4	fType	29
7.4	file_header_t Struct Reference	29
7.4.1	Detailed Description	29
7.4.2	Field Documentation	29
7.4.2.1	magic	29
7.4.2.2	num_hptdc	29
7.4.2.3	run_id	29
7.4.2.4	spill_id	29
7.5	FileReader Class Reference	29
7.5.1	Constructor & Destructor Documentation	30
7.5.1.1	FileReader	30
7.5.1.2	~FileReader	30
7.5.2	Member Function Documentation	30
7.5.2.1	GetNextEvent	30
7.5.2.2	GetNumTDCs	30
7.5.3	Field Documentation	30
7.5.3.1	fFile	30
7.5.3.2	fHeader	30
7.6	VME::glob_offs Struct Reference	31
7.6.1	Field Documentation	31
7.6.1.1	coarse	31
7.6.1.2	fine	31
7.7	HTTPMessage Class Reference	31
7.7.1	Detailed Description	32
7.7.2	Constructor & Destructor Documentation	32
7.7.2.1	HTTPMessage	33
7.7.2.2	HTTPMessage	33
7.7.3	Member Function Documentation	33
7.7.3.1	Decode	33
7.7.3.2	Dump	33
7.7.3.3	Encode	33
7.7.3.4	GetKey	33
7.7.4	Field Documentation	33

7.7.4.1	fOriginalString	33
7.7.4.2	fWS	33
7.8	Message Class Reference	34
7.8.1	Detailed Description	34
7.8.2	Constructor & Destructor Documentation	35
7.8.2.1	Message	35
7.8.2.2	Message	35
7.8.2.3	Message	35
7.8.2.4	~Message	35
7.8.3	Member Function Documentation	35
7.8.3.1	Dump	35
7.8.3.2	GetKey	35
7.8.3.3	GetString	35
7.8.3.4	IsFromWeb	35
7.8.4	Field Documentation	35
7.8.4.1	fString	35
7.9	Messenger Class Reference	36
7.9.1	Detailed Description	37
7.9.2	Constructor & Destructor Documentation	37
7.9.2.1	Messenger	37
7.9.2.2	Messenger	37
7.9.2.3	~Messenger	38
7.9.3	Member Function Documentation	38
7.9.3.1	AddClient	38
7.9.3.2	Broadcast	39
7.9.3.3	Connect	39
7.9.3.4	Disconnect	39
7.9.3.5	DisconnectClient	39
7.9.3.6	GetType	40
7.9.3.7	ProcessMessage	40
7.9.3.8	Receive	41
7.9.3.9	Send	41
7.9.3.10	StartAcquisition	42
7.9.3.11	StopAcquisition	42
7.9.3.12	SwitchClientType	42
7.9.4	Field Documentation	42
7.9.4.1	fNumAttempts	42
7.9.4.2	fPID	42
7.9.4.3	fWS	42
7.10	Socket Class Reference	43



7.10.1 Detailed Description . . . . .	44
7.10.2 Member Typedef Documentation . . . . .	45
7.10.2.1 SocketCollection . . . . .	45
7.10.3 Constructor & Destructor Documentation . . . . .	45
7.10.3.1 Socket . . . . .	45
7.10.3.2 Socket . . . . .	45
7.10.3.3 ~Socket . . . . .	45
7.10.4 Member Function Documentation . . . . .	45
7.10.4.1 AcceptConnections . . . . .	45
7.10.4.2 Bind . . . . .	45
7.10.4.3 Configure . . . . .	46
7.10.4.4 Create . . . . .	46
7.10.4.5 DumpConnected . . . . .	46
7.10.4.6 FetchMessage . . . . .	46
7.10.4.7 GetPort . . . . .	46
7.10.4.8 GetSocketId . . . . .	46
7.10.4.9 GetSocketType . . . . .	46
7.10.4.10 IsWebSocket . . . . .	46
7.10.4.11 Listen . . . . .	46
7.10.4.12 PrepareConnection . . . . .	47
7.10.4.13 SelectConnections . . . . .	47
7.10.4.14 SendMessage . . . . .	47
7.10.4.15 SetPort . . . . .	47
7.10.4.16 SetSocketId . . . . .	47
7.10.4.17 Start . . . . .	48
7.10.4.18 Stop . . . . .	48
7.10.5 Field Documentation . . . . .	48
7.10.5.1 fAddress . . . . .	48
7.10.5.2 fBuffer . . . . .	48
7.10.5.3 fMaster . . . . .	48
7.10.5.4 fPort . . . . .	48
7.10.5.5 fReadFds . . . . .	48
7.10.5.6 fSocketId . . . . .	48
7.10.5.7 fSocketsConnected . . . . .	48
7.11 SocketMessage Class Reference . . . . .	49
7.11.1 Detailed Description . . . . .	50
7.11.2 Constructor & Destructor Documentation . . . . .	51
7.11.2.1 SocketMessage . . . . .	51
7.11.2.2 SocketMessage . . . . .	51
7.11.2.3 SocketMessage . . . . .	51

7.11.2.4	SocketMessage	51
7.11.2.5	SocketMessage	51
7.11.2.6	SocketMessage	52
7.11.2.7	SocketMessage	52
7.11.2.8	SocketMessage	52
7.11.2.9	SocketMessage	52
7.11.2.10	SocketMessage	53
7.11.2.11	SocketMessage	53
7.11.2.12	~SocketMessage	53
7.11.3	Member Function Documentation	53
7.11.3.1	Dump	53
7.11.3.2	GetIntValue	53
7.11.3.3	GetKey	54
7.11.3.4	GetString	54
7.11.3.5	GetValue	54
7.11.3.6	GetVectorValue	54
7.11.3.7	Object	54
7.11.3.8	SetKeyValue	54
7.11.3.9	SetKeyValue	54
7.11.3.10	SetKeyValue	55
7.11.3.11	SetKeyValue	55
7.11.3.12	String	55
7.11.4	Field Documentation	55
7.11.4.1	fMessage	55
7.12	VME::TDCEvent Class Reference	55
7.12.1	Detailed Description	56
7.12.2	Member Enumeration Documentation	57
7.12.2.1	EventType	57
7.12.3	Constructor & Destructor Documentation	57
7.12.3.1	TDCEvent	57
7.12.3.2	TDCEvent	57
7.12.3.3	~TDCEvent	57
7.12.4	Member Function Documentation	57
7.12.4.1	Dump	57
7.12.4.2	GetBunchId	57
7.12.4.3	GetChannelId	58
7.12.4.4	GetErrorFlags	58
7.12.4.5	GetETTT	58
7.12.4.6	GetEventCount	59
7.12.4.7	GetEventId	59

7.12.4.8	GetGeo	59
7.12.4.9	GetLeadingTime	59
7.12.4.10	GetStatus	60
7.12.4.11	GetTDCId	60
7.12.4.12	GetTrailingTime	60
7.12.4.13	GetType	61
7.12.4.14	GetWidth	61
7.12.4.15	GetWordCount	61
7.12.4.16	IsTrailing	61
7.12.4.17	SetWord	61
7.12.5	Field Documentation	61
7.12.5.1	fWord	62
7.13	VME::TDCV1x90 Class Reference	62
7.13.1	Detailed Description	63
7.13.2	Constructor & Destructor Documentation	64
7.13.2.1	TDCV1x90	64
7.13.2.2	~TDCV1x90	64
7.13.3	Member Function Documentation	64
7.13.3.1	abort	64
7.13.3.2	CheckConfiguration	65
7.13.3.3	DisableChannel	65
7.13.3.4	EnableChannel	65
7.13.3.5	FetchEvents	66
7.13.3.6	GetBLTEventNumberRegister	66
7.13.3.7	GetCtlRegister	66
7.13.3.8	GetETTT	67
7.13.3.9	GetEventCounter	67
7.13.3.10	GetEventStored	67
7.13.3.11	GetFirmwareRev	68
7.13.3.12	GetModel	68
7.13.3.13	GetOUI	68
7.13.3.14	GetSerialNumber	69
7.13.3.15	GetStatusRegister	69
7.13.3.16	GetTDCEncapsulation	69
7.13.3.17	HardwareReset	69
7.13.3.18	IsTriggerMatching	70
7.13.3.19	ReadDetection	70
7.13.3.20	ReadFIFOSize	71
7.13.3.21	ReadGlobalOffset	71
7.13.3.22	ReadRCAdjust	72

7.13.3.23 ReadRegister . . . . .	72
7.13.3.24 ReadRegister . . . . .	72
7.13.3.25 ReadResolution . . . . .	73
7.13.3.26 ReadTrigConf . . . . .	73
7.13.3.27 SetAcquisitionMode . . . . .	73
7.13.3.28 SetBLTEventNumberRegister . . . . .	74
7.13.3.29 SetContinuousStorage . . . . .	74
7.13.3.30 SetCtlRegister . . . . .	74
7.13.3.31 SetDetection . . . . .	75
7.13.3.32 SetETTT . . . . .	75
7.13.3.33 SetFIFOSize . . . . .	75
7.13.3.34 SetGlobalOffset . . . . .	76
7.13.3.35 SetLSBTraileadEdge . . . . .	76
7.13.3.36 SetPairModeResolution . . . . .	76
7.13.3.37 SetPol . . . . .	77
7.13.3.38 SetRCAdjust . . . . .	77
7.13.3.39 SetStatusRegister . . . . .	77
7.13.3.40 SetTDCEncapsulation . . . . .	77
7.13.3.41 SetTDCErrorsMarks . . . . .	78
7.13.3.42 SetTriggerMatching . . . . .	78
7.13.3.43 SetVerboseLevel . . . . .	78
7.13.3.44 SetWindowOffset . . . . .	79
7.13.3.45 SetWindowWidth . . . . .	79
7.13.3.46 SoftwareClear . . . . .	79
7.13.3.47 SoftwareReset . . . . .	80
7.13.3.48 WaitMicro . . . . .	80
7.13.3.49 WriteRegister . . . . .	80
7.13.3.50 WriteRegister . . . . .	80
7.13.4 Field Documentation . . . . .	80
7.13.4.1 acqm . . . . .	80
7.13.4.2 am . . . . .	81
7.13.4.3 am_blt . . . . .	81
7.13.4.4 detm . . . . .	81
7.13.4.5 fBaseAddr . . . . .	81
7.13.4.6 fBuffer . . . . .	81
7.13.4.7 fDetMode . . . . .	81
7.13.4.8 fHandle . . . . .	81
7.13.4.9 fVerb . . . . .	81
7.13.4.10 gEnd . . . . .	81
7.13.4.11 nchannels . . . . .	81

7.13.4.12 outBufTDCErr . . . . .	81
7.13.4.13 outBufTDCHeadTrail . . . . .	81
7.13.4.14 outBufTDCTTT . . . . .	81
7.13.4.15 pair_lead_res . . . . .	81
7.13.4.16 pair_width_res . . . . .	81
7.13.4.17 trailead_edge_res . . . . .	81
7.14 VME::trailead_t Struct Reference . . . . .	81
7.14.1 Field Documentation . . . . .	81
7.14.1.1 ettt . . . . .	81
7.14.1.2 event_count . . . . .	82
7.14.1.3 leading . . . . .	82
7.14.1.4 total_hits . . . . .	82
7.14.1.5 trailing . . . . .	82
7.15 VMEReader Class Reference . . . . .	82
7.15.1 Detailed Description . . . . .	84
7.15.2 Member Typedef Documentation . . . . .	84
7.15.2.1 TDCCollection . . . . .	84
7.15.3 Constructor & Destructor Documentation . . . . .	84
7.15.3.1 VMEReader . . . . .	84
7.15.3.2 ~VMEReader . . . . .	84
7.15.4 Member Function Documentation . . . . .	84
7.15.4.1 Abort . . . . .	84
7.15.4.2 AddTDC . . . . .	85
7.15.4.3 GetRunNumber . . . . .	86
7.15.4.4 GetTDC . . . . .	86
7.15.5 Field Documentation . . . . .	86
7.15.5.1 fBridge . . . . .	86
7.15.5.2 fOnSocket . . . . .	86
7.15.5.3 fTDCCollection . . . . .	86
<b>Index</b>	<b>89</b>



# Chapter 1

## Module Index

### 1.1 Modules

Here is a list of all modules:

Socket communication objects . . . . .	9
--	---





## Chapter 2

# Namespace Index

### 2.1 Namespace List

Here is a list of all namespaces with brief descriptions:

<a href="#">VME</a> . . . . .	<a href="#">11</a>
<a href="#">VME::TDCV1x90Opcodes</a> . . . . .	<a href="#">14</a>



## Chapter 3

# Hierarchical Index

### 3.1 Class Hierarchy

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

VME::BridgeVx718 . . . . .	19
Exception . . . . .	26
file_header_t . . . . .	29
FileReader . . . . .	29
VME::glob_offs . . . . .	31
Message . . . . .	34
HTTPMessage . . . . .	31
SocketMessage . . . . .	49
Socket . . . . .	43
Client . . . . .	21
VMEReader . . . . .	82
Messenger . . . . .	36
VME::TDCEvent . . . . .	55
VME::TDCV1x90 . . . . .	62
VME::trailead_t . . . . .	81



## Chapter 4

# Data Structure Index

### 4.1 Data Structures

Here are the data structures with brief descriptions:

<a href="#">VME::BridgeVx718</a>	
Class defining the <a href="#">VME</a> bridge . . . . .	19
<a href="#">Client</a>	
Base client object for the socket . . . . .	21
<a href="#">Exception</a>	
A simple exception handler . . . . .	26
<a href="#">file_header_t</a>	
Header to the output files . . . . .	29
<a href="#">FileReader</a> . . . . .	29
<a href="#">VME::glob_offs</a> . . . . .	31
<a href="#">HTTPMessage</a>	
<a href="#">Message</a> to be transmitted through a WebSocket protocol . . . . .	31
<a href="#">Message</a>	
Base socket message type . . . . .	34
<a href="#">Messenger</a>	
Base master object for the socket . . . . .	36
<a href="#">Socket</a>	
Base socket object from which clients/master from a socket inherit . . . . .	43
<a href="#">SocketMessage</a>	
Socket-passed message type . . . . .	49
<a href="#">VME::TDCEvent</a>	
HPTDC event parser . . . . .	55
<a href="#">VME::TDCV1x90</a> . . . . .	62
<a href="#">VME::trailead_t</a> . . . . .	81
<a href="#">VMEReader</a> . . . . .	82



## 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.*

#### Enumerations

- enum [Socket::SocketType](#) {  
[Socket::INVALID](#) = -1, [Socket::MASTER](#) = 0, [Socket::WEBSOCKET\\_CLIENT](#), [Socket::CLIENT](#),  
[Socket::DETECTOR](#) }  
*Type of actor playing a role on the socket.*

#### 5.1.1 Detailed Description

#### 5.1.2 Enumeration Type Documentation

##### 5.1.2.1 enum [Socket::SocketType](#)

Type of actor playing a role on the socket.

Enumerator

***INVALID***  
***MASTER***  
***WEBSOCKET\_CLIENT***  
***CLIENT***  
***DETECTOR***





## Chapter 6

# Namespace Documentation

### 6.1 VME Namespace Reference

#### Namespaces

- [TDCV1x90Opcodes](#)

#### Data Structures

- class [BridgeVx718](#)  
*class defining the VME bridge*
- struct [glob\\_offs](#)
- class [TDCEvent](#)  
*HPTDC event parser.*
- class [TDCV1x90](#)
- struct [trailead\\_t](#)

#### Typedefs

- typedef std::vector< [TDCEvent](#) > [TDCEventCollection](#)

#### Enumerations

- enum [BridgeType](#) { [CAEN\\_V1718](#), [CAEN\\_V2718](#) }  
*Compatible bridge types.*
- 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 [acq\\_mode](#) { [CONT\\_STORAGE](#), [TRIG\\_MATCH](#) }
- enum [det\\_mode](#) { [PAIR](#) = 0, [OTRILING](#) = 1, [OLEADING](#) = 2, [TRAILEAD](#) = 3 }
- enum [stat\\_reg](#) {  
    [DATA\\_READY](#) = 0, [ALM\\_FULL](#) = 1, [FULL](#) = 2, [TRG\\_MATCH](#) = 3,  
    [HEADER\\_EN](#) = 4, [TERM\\_ON](#) = 5, [ERROR0](#) = 6, [ERROR1](#) = 7,  
    [ERROR2](#) = 8, [ERROR3](#) = 9, [BERR\\_FLAG](#) = 10, [PURG](#) = 11,  
    [RES\\_1](#) = 12, [RES\\_2](#) = 13, [PAIRED](#) = 14, [TRIGGER\\_LOST](#) = 15 }

- enum `ctl_reg` {  
`BERREN` = 0, `TERM` = 1, `TERM_SW` = 2, `EMPTY_EVENT` = 3,  
`ALIGN64` = 4, `COMPENSATION_ENABLE` = 5, `TEST_FIFO_ENABLE` = 6, `READ_COMPENSATION_SRAM_ENABLE` = 7,  
`EVENT_FIFO_ENABLE` = 8, `EXTENDED_TRIGGER_TIME_TAG_ENABLE` = 9 }
- enum `mod_reg` {  
`Control` = 0x1000, `Status` = 0x1002, `InterruptLevel` = 0x100a, `InterruptVector` = 0x100c,  
`GeoAddress` = 0x100e, `MCSTBase` = 0x1010, `MCSTControl` = 0x1012, `ModuleReset` = 0x1014,  
`kSoftwareClear` = 0x1016, `EventCounter` = 0x101c, `EventStored` = 0x1020, `BLTEventNumber` = 0x1024,  
`FirmwareRev` = 0x1026, `Micro` = 0x102e, `MicroHandshake` = 0x1030, `EventFIFO` = 0x1038,  
`EventFIFOStoredRegister` = 0x103c, `EventFIFOStatusRegister` = 0x103e, `ROMOui2` = 0x4024, `ROMOui1` = 0x4028,  
`ROMOui0` = 0x402c, `ROMBoard2` = 0x4034, `ROMBoard1` = 0x4038, `ROMBoard0` = 0x403c,  
`ROMRevis3` = 0x4040, `ROMRevis2` = 0x4044, `ROMRevis1` = 0x4048, `ROMRevis0` = 0x404c,  
`ROMSerNum1` = 0x4080, `ROMSerNum0` = 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::acq\_mode

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::ctl\_reg

Enumerator

***BERREN***  
***TERM***  
***TERM\_SW***  
***EMPTY\_EVENT***  
***ALIGN64***  
***COMPENSATION\_ENABLE***  
***TEST\_FIFO\_ENABLE***  
***READ\_COMPENSATION\_SRAM\_ENABLE***  
***EVENT\_FIFO\_ENABLE***  
***EXTENDED\_TRIGGER\_TIME\_TAG\_ENABLE***

## 6.1.2.4 enum VME::det\_mode

Enumerator

***PAIR***  
***OTRILING***  
***OLEADING***  
***TRAILEAD***

## 6.1.2.5 enum VME::micro\_handshake

Enumerator

***WRITE\_OK*** Is the TDC ready for writing?  
***READ\_OK*** Is the TDC ready for reading?

## 6.1.2.6 enum VME::mod\_reg

Enumerator

***Control***  
***Status***  
***InterruptLevel***  
***InterruptVector***  
***GeoAddress***  
***MCSTBase***  
***MCSTControl***  
***ModuleReset***  
***kSoftwareClear***  
***EventCounter***  
***EventStored***  
***BLTEventNumber***  
***FirmwareRev***  
***Micro***  
***MicroHandshake***  
***EventFIFO***  
***EventFIFOStoredRegister***  
***EventFIFOStatusRegister***  
***ROMOui2***  
***ROMOui1***  
***ROMOui0***  
***ROMBoard2***  
***ROMBoard1***  
***ROMBoard0***  
***ROMRevis3***  
***ROMRevis2***  
***ROMRevis1***  
***ROMRevis0***  
***ROMSerNum1***  
***ROMSerNum0***

## 6.1.2.7 enum VME::stat\_reg

Enumerator

***DATA\_READY***  
***ALM\_FULL***  
***FULL***  
***TRG\_MATCH***  
***HEADER\_EN***  
***TERM\_ON***  
***ERROR0***  
***ERROR1***  
***ERROR2***  
***ERROR3***  
***BERR\_FLAG***  
***PURG***  
***RES\_1***  
***RES\_2***  
***PAIRED***  
***TRIGGER\_LOST***

## 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 )
- 6.2.1.28 Opcode VME::TDCV1x90Opcodes::READ\_EN\_PATTERN ( 0x4500 )
- 6.2.1.29 Opcode VME::TDCV1x90Opcodes::READ\_EN\_PATTERN32 ( 0x4700 )
- 6.2.1.30 Opcode VME::TDCV1x90Opcodes::READ\_ERROR\_STATUS ( 0x7400 )
- 6.2.1.31 Opcode VME::TDCV1x90Opcodes::READ\_ERROR\_TYPES ( 0x3a00 )
- 6.2.1.32 Opcode VME::TDCV1x90Opcodes::READ\_EVENT\_SIZE ( 0x3400 )
- 6.2.1.33 Opcode VME::TDCV1x90Opcodes::READ\_FIFO\_SIZE ( 0x3c00 )
- 6.2.1.34 Opcode VME::TDCV1x90Opcodes::READ\_GLOB\_OFFS ( 0x5100 )
- 6.2.1.35 Opcode VME::TDCV1x90Opcodes::READ\_HEAD\_TRAILER ( 0x3200 )
- 6.2.1.36 Opcode VME::TDCV1x90Opcodes::READ\_MICRO\_REV ( 0x6100 )
- 6.2.1.37 Opcode VME::TDCV1x90Opcodes::READ\_RC\_ADJ ( 0x5500 )
- 6.2.1.38 Opcode VME::TDCV1x90Opcodes::READ\_RES ( 0x2600 )
- 6.2.1.39 Opcode VME::TDCV1x90Opcodes::READ\_SETUP\_REG ( 0x7100 )
- 6.2.1.40 Opcode VME::TDCV1x90Opcodes::READ\_SETUP\_SCANPATH ( 0xc900 )
- 6.2.1.41 Opcode VME::TDCV1x90Opcodes::READ\_SPARE ( 0xc400 )
- 6.2.1.42 Opcode VME::TDCV1x90Opcodes::READ\_STATUS\_STREAM ( 0x7600 )
- 6.2.1.43 Opcode VME::TDCV1x90Opcodes::READ\_TDC\_ID ( 0x6000 )
- 6.2.1.44 Opcode VME::TDCV1x90Opcodes::READ\_TRG\_CONF ( 0x1600 )
- 6.2.1.45 Opcode VME::TDCV1x90Opcodes::RESET\_DLL\_PLL ( 0x6200 )
- 6.2.1.46 Opcode VME::TDCV1x90Opcodes::REV\_DATE\_MICRO\_FW ( 0xc200 )
- 6.2.1.47 Opcode VME::TDCV1x90Opcodes::SAVE\_RC\_ADJ ( 0x5600 )

- 6.2.1.48 Opcode VME::TDCV1x90OpCodes::SAVE\_USER\_CONFIG ( 0x0600 )
- 6.2.1.49 Opcode VME::TDCV1x90OpCodes::SET\_ADJUST\_CH ( 0x5200 )
- 6.2.1.50 Opcode VME::TDCV1x90OpCodes::SET\_DEAD\_TIME ( 0x2800 )
- 6.2.1.51 Opcode VME::TDCV1x90OpCodes::SET\_DETECTION ( 0x2200 )
- 6.2.1.52 Opcode VME::TDCV1x90OpCodes::SET\_DLL\_CLOCK ( 0xc800 )
- 6.2.1.53 Opcode VME::TDCV1x90OpCodes::SET\_ERROR\_TYPES ( 0x3900 )
- 6.2.1.54 Opcode VME::TDCV1x90OpCodes::SET\_EVENT\_SIZE ( 0x3300 )
- 6.2.1.55 Opcode VME::TDCV1x90OpCodes::SET\_FIFO\_SIZE ( 0x3b00 )
- 6.2.1.56 Opcode VME::TDCV1x90OpCodes::SET\_GLOB\_OFFS ( 0x5000 )
- 6.2.1.57 Opcode VME::TDCV1x90OpCodes::SET\_KEEP\_TOKEN ( 0x0300 )
- 6.2.1.58 Opcode VME::TDCV1x90OpCodes::SET\_PAIR\_RES ( 0x2500 )
- 6.2.1.59 Opcode VME::TDCV1x90OpCodes::SET\_RC\_ADJ ( 0x5400 )
- 6.2.1.60 Opcode VME::TDCV1x90OpCodes::SET\_REJ\_MARGIN ( 0x1300 )
- 6.2.1.61 Opcode VME::TDCV1x90OpCodes::SET\_SW\_MARGIN ( 0x1200 )
- 6.2.1.62 Opcode VME::TDCV1x90OpCodes::SET\_TDC\_TSET\_OUTPUT ( 0xc700 )
- 6.2.1.63 Opcode VME::TDCV1x90OpCodes::SET\_TR\_LEAD\_LSB ( 0x2400 )
- 6.2.1.64 Opcode VME::TDCV1x90OpCodes::SET\_WIN\_OFFS ( 0x1100 )
- 6.2.1.65 Opcode VME::TDCV1x90OpCodes::SET\_WIN\_WIDTH ( 0x1000 )
- 6.2.1.66 Opcode VME::TDCV1x90OpCodes::TRG\_MATCH ( 0x0000 )
- 6.2.1.67 Opcode VME::TDCV1x90OpCodes::UPDATE\_SETUP\_REG ( 0x7200 )
- 6.2.1.68 Opcode VME::TDCV1x90OpCodes::UPDATE\_SETUP\_TDC ( 0x7700 )
- 6.2.1.69 Opcode VME::TDCV1x90OpCodes::WRITE\_EEPROM ( 0xc000 )
- 6.2.1.70 Opcode VME::TDCV1x90OpCodes::WRITE\_EN\_PATTERN ( 0x4400 )
- 6.2.1.71 Opcode VME::TDCV1x90OpCodes::WRITE\_EN\_PATTERN32 ( 0x4600 )
- 6.2.1.72 Opcode VME::TDCV1x90OpCodes::WRITE\_SETUP\_REG ( 0x7000 )
- 6.2.1.73 Opcode VME::TDCV1x90OpCodes::WRITE\_SPARE ( 0xc300 )



## Chapter 7

# Data Structure Documentation

### 7.1 VME::BridgeVx718 Class Reference

class defining the VME bridge

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

#### Public Member Functions

- [BridgeVx718](#) (const char \*device, [BridgeType](#) type)  
*Constructor.*
- [~BridgeVx718](#) ()  
*Destructor.*
- int32\_t [GetHandle](#) () const  
*Gets bhandle.*
- void [CheckConfiguration](#) () const
- void [OutputConf](#) (CVOutputSelect output)  
*Set and control the output lines.*
- void [OutputOn](#) (CVOutputSelect output)
- void [OutputOff](#) (CVOutputSelect output)
- void [InputConf](#) (CVInputSelect input)  
*Set and read the input lines.*
- void [InputRead](#) (CVInputSelect input)

#### Private Attributes

- std::map< CVOutputSelect, CVOutputRegisterBits > [fPortMapping](#)  
*Map output lines [0,4] to corresponding register.*
- int32\_t [fHandle](#)  
*Device handle.*

#### 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](mailto:laurent.forthomme@cern.ch)  
 Bob Velghe [bob.velghe@cern.ch](mailto:bob.velghe@cern.ch)

**Date**

Jun 2010

**7.1.2 Constructor & Destructor Documentation****7.1.2.1 VME::BridgeVx718::BridgeVx718 ( const char \* *device*, BridgeType *type* )**

Constructor.

Bridge class constructor

**Parameters**

in	<i>device</i>	Device identifier on the <a href="#">VME</a> crate
in	<i>type</i>	Device type (1718/2718)

Here is the call graph for this function:

**7.1.2.2 VME::BridgeVx718::~~BridgeVx718 ( )**

Destructor.

Bridge class destructor

**7.1.3 Member Function Documentation****7.1.3.1 void VME::BridgeVx718::CheckConfiguration ( ) const****7.1.3.2 int32\_t VME::BridgeVx718::GetHandle ( ) const [inline]**

Gets bhandle.

Gives bhandle value

Returns

bhandle value

**7.1.3.3 void VME::BridgeVx718::InputConf ( CVInputSelect *input* )**

Set and read the input lines.

7.1.3.4 void VME::BridgeVx718::InputRead ( CVInputSelect *input* )

7.1.3.5 void VME::BridgeVx718::OutputConf ( CVOutputSelect *output* )

Set and control the output lines.

7.1.3.6 void VME::BridgeVx718::OutputOff ( CVOutputSelect *output* )

7.1.3.7 void VME::BridgeVx718::OutputOn ( CVOutputSelect *output* )

## 7.1.4 Field Documentation

7.1.4.1 int32\_t VME::BridgeVx718::fHandle [private]

Device handle.

7.1.4.2 std::map<CVOutputSelect,CVOutputRegisterBits> VME::BridgeVx718::fPortMapping [private]

Map output lines [0,4] to corresponding register.

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

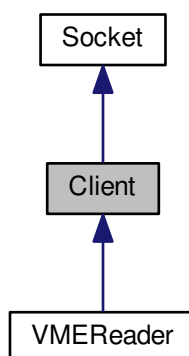
- include/VME\_BridgeVx718.h
- src/VME\_BridgeVx718.cpp

## 7.2 Client Class Reference

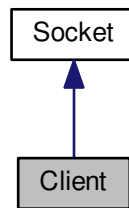
Base client object for the socket.

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

Inheritance diagram for Client:



Collaboration diagram for Client:



## Public Member Functions

- [Client](#) ()  
*General void client constructor.*
- [Client](#) (int port)  
*Bind a socket client to a given port.*
- virtual [~Client](#) ()
- bool [Connect](#) ()  
*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.*
- [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 [flsConnected](#)

## Additional Inherited Members

### 7.2.1 Detailed Description

Base client object for the socket.

[Client](#) object used by the server to send/receive commands from the messenger/broadcaster.

#### Author

Laurent Forthomme [laurent.forthomme@cern.ch](mailto:laurent.forthomme@cern.ch)

#### Date

24 Mar 2015

### 7.2.2 Constructor & Destructor Documentation

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

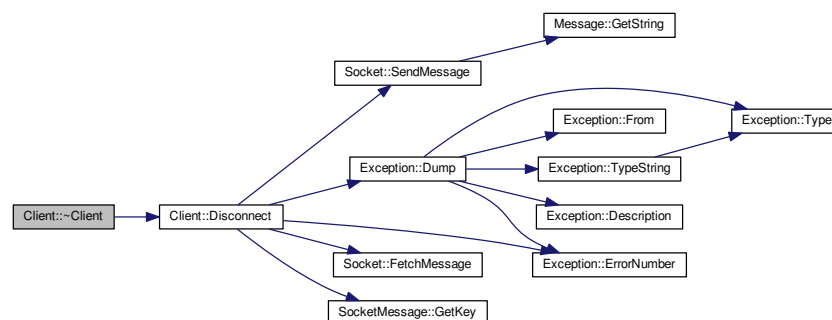
General void client constructor.

#### 7.2.2.2 `Client::Client ( int port )`

Bind a socket client to a given port.

#### 7.2.2.3 `Client::~~Client ( )` `[virtual]`

Here is the call graph for this function:

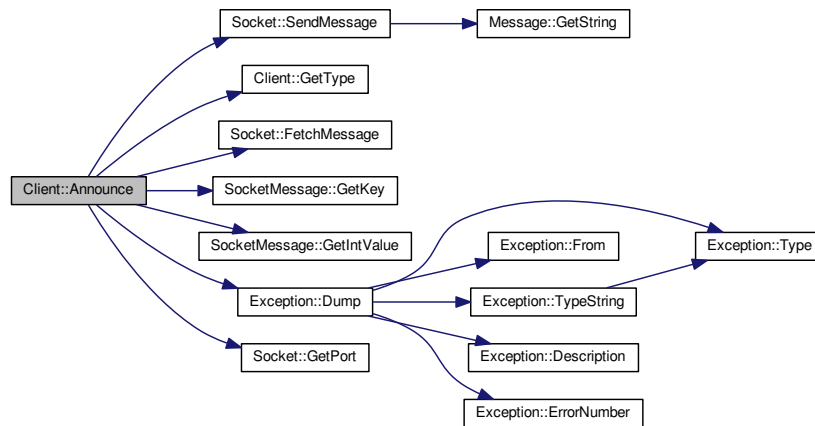


### 7.2.3 Member Function Documentation

#### 7.2.3.1 `void Client::Announce ( )` `[private]`

Announce our entry on the socket to its master.

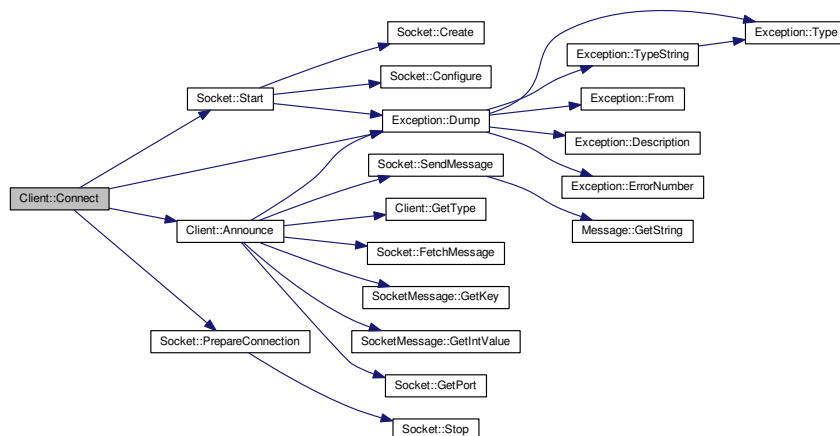
Here is the call graph for this function:



### 7.2.3.2 bool Client::Connect ( )

Bind this client to the socket.

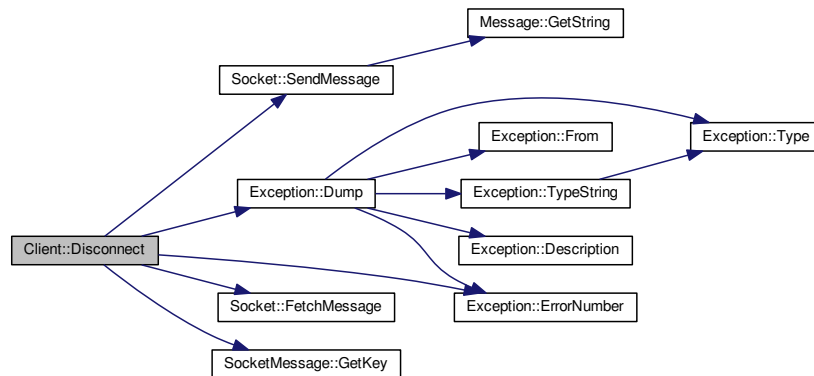
Here is the call graph for this function:



### 7.2.3.3 void Client::Disconnect ( )

Unbind this client from the socket.

Here is the call graph for this function:



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

Socket actor type retrieval method.

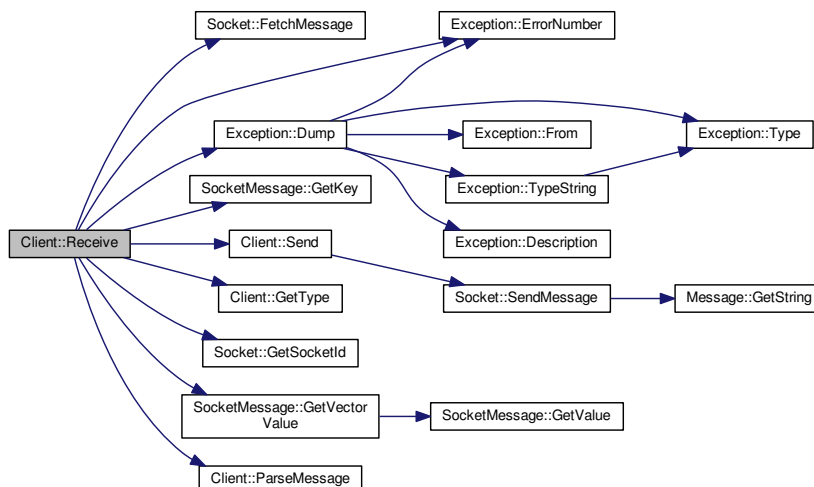
#### 7.2.3.5 virtual void Client::ParseMessage ( const SocketMessage & m ) [inline],[virtual]

Parse a SocketMessage received from the master.

#### 7.2.3.6 void Client::Receive ( )

Receive a socket message from the master.

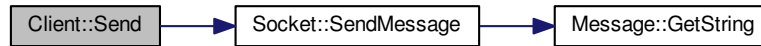
Here is the call graph for this function:



### 7.2.3.7 void Client::Send ( const Message & m ) const [inline]

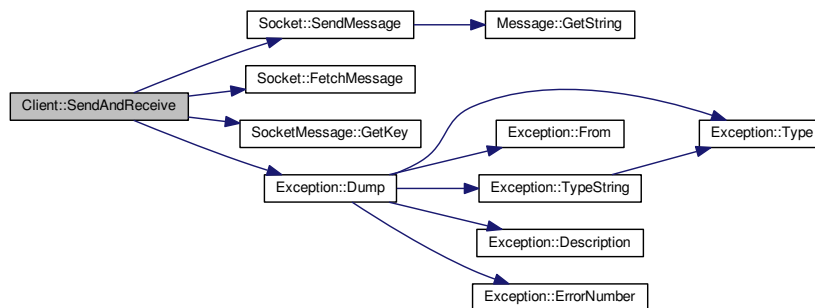
Send a message to the master through the socket.

Here is the call graph for this function:



### 7.2.3.8 SocketMessage Client::SendAndReceive ( const SocketMessage & m, const MessageKey & a ) const [inline]

Here is the call graph for this function:



## 7.2.4 Field Documentation

### 7.2.4.1 int Client::fClientId [private]

### 7.2.4.2 bool Client::fIsConnected [private]

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

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

## 7.3 Exception Class Reference

A simple exception handler.

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

### Public Member Functions

- [Exception](#) (const char \*from, std::string desc, ExceptionType type=Undefined, const int id=0)



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

### Private Attributes

- std::string [fFrom](#)
- std::string [fDescription](#)
- ExceptionType [fType](#)
- int [fErrorNumber](#)

#### 7.3.1 Detailed Description

A simple exception handler.

##### Author

Laurent Forthomme [laurent.forthomme@cern.ch](mailto:laurent.forthomme@cern.ch)

##### Date

24 Mar 2015

#### 7.3.2 Constructor & Destructor Documentation

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

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

**7.3.2.3** `Exception::~~Exception ( )` [inline]

Here is the call graph for this function:

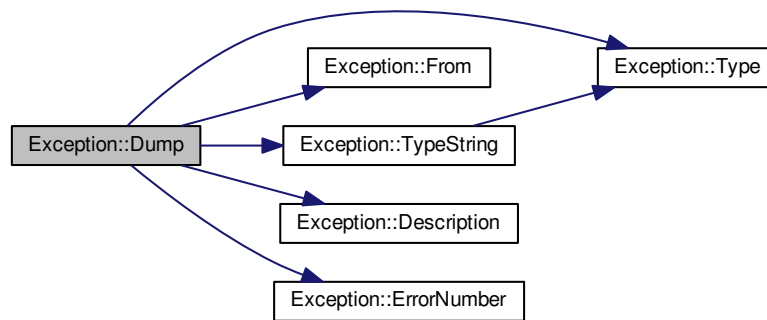


### 7.3.3 Member Function Documentation

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

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

Here is the call graph for this function:



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

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

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

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

Here is the call graph for this function:



### 7.3.4 Field Documentation

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

7.3.4.2 `int Exception::fErrorNumber` [private]

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

#### 7.3.4.4 ExceptionType Exception::fType [private]

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

- include/Exception.h

## 7.4 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](#)

### 7.4.1 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](mailto:laurent.forthomme@cern.ch)

#### Date

14 Apr 2015

### 7.4.2 Field Documentation

#### 7.4.2.1 uint32\_t file\_header\_t::magic

#### 7.4.2.2 uint8\_t file\_header\_t::num\_hptdc

#### 7.4.2.3 uint32\_t file\_header\_t::run\_id

#### 7.4.2.4 uint32\_t file\_header\_t::spill\_id

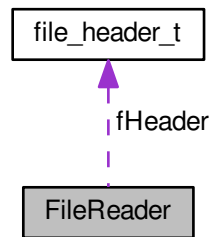
The documentation for this struct was generated from the following file:

- include/FileConstants.h

## 7.5 FileReader Class Reference

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

Collaboration diagram for FileReader:



## Public Member Functions

- [FileReader](#) (std::string name)
- [~FileReader](#) ()
- unsigned int [GetNumTDCs](#) () const
- [VME::TDCEvent](#) [GetNextEvent](#) ()

## Private Attributes

- std::ifstream [fFile](#)
- [file\\_header\\_t](#) [fHeader](#)

## 7.5.1 Constructor & Destructor Documentation

7.5.1.1 `FileReader::FileReader ( std::string name )`

7.5.1.2 `FileReader::~~FileReader ( )`

## 7.5.2 Member Function Documentation

7.5.2.1 `VME::TDCEvent FileReader::GetNextEvent ( )` `[inline]`

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

## 7.5.3 Field Documentation

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

7.5.3.2 `file_header_t FileReader::fHeader` `[private]`

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

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

## 7.6 VME::glob\_offs Struct Reference

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

### Data Fields

- uint16\_t [coarse](#)
- uint16\_t [fine](#)

### 7.6.1 Field Documentation

7.6.1.1 uint16\_t VME::glob\_offs::coarse

7.6.1.2 uint16\_t VME::glob\_offs::fine

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

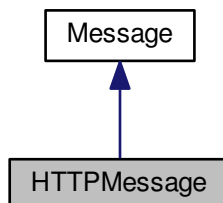
- include/VME\_TDCV1x90.h

## 7.7 HTTPMessage Class Reference

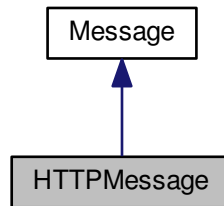
[Message](#) to be transmitted through a WebSocket protocol.

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

Inheritance diagram for HTTPMessage:



Collaboration diagram for HTTPMessage:



### Public Member Functions

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

### Private Attributes

- WebSocket \* [fWS](#)
- std::string [fOriginalString](#)

### Additional Inherited Members

#### 7.7.1 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](mailto:laurent.forthomme@cern.ch)

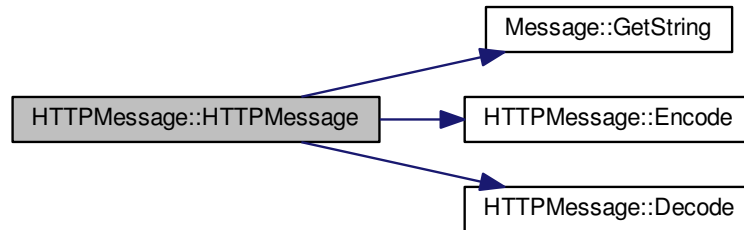
#### Date

1 Apr 2015

#### 7.7.2 Constructor & Destructor Documentation

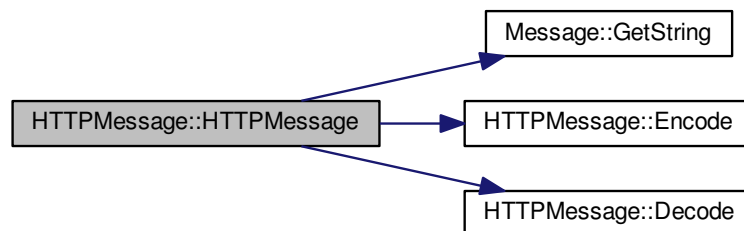
#### 7.7.2.1 HTTPMessage::HTTPMessage ( WebSocket \* *ws*, Message *m*, MessageAction *a* ) [inline]

Here is the call graph for this function:



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

Here is the call graph for this function:



### 7.7.3 Member Function Documentation

#### 7.7.3.1 void HTTPMessage::Decode ( ) [inline]

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

#### 7.7.3.3 void HTTPMessage::Encode ( ) [inline]

#### 7.7.3.4 MessageKey HTTPMessage::GetKey ( ) const [inline]

### 7.7.4 Field Documentation

#### 7.7.4.1 std::string HTTPMessage::fOriginalString [private]

#### 7.7.4.2 WebSocket\* HTTPMessage::fWS [private]

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

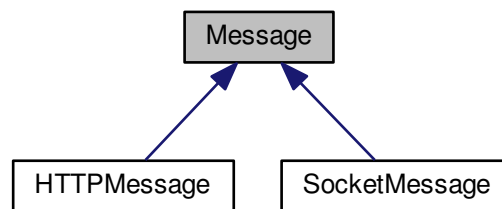
- `include/HTTPMessage.h`

## 7.8 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`

#### 7.8.1 Detailed Description

Base socket message type.

Base handler for messages to be transmitted through the socket



**Author**

Laurent Forthomme [laurent.forthomme@cern.ch](mailto:laurent.forthomme@cern.ch)

**Date**

6 Apr 2015

**7.8.2 Constructor & Destructor Documentation****7.8.2.1 Message::Message ( ) [inline]**

Void message constructor.

**7.8.2.2 Message::Message ( const char \* *msg* ) [inline]**

Construct a message from a string.

**7.8.2.3 Message::Message ( std::string *msg* ) [inline]**

Construct a message from a string.

**7.8.2.4 virtual Message::~Message ( ) [inline],[virtual]****7.8.3 Member Function Documentation****7.8.3.1 void Message::Dump ( std::ostream & *os* = std::cout ) const [inline]****7.8.3.2 MessageKey Message::GetKey ( ) const [inline]**

Placeholder for the MessageKey retrieval method.

**7.8.3.3 std::string Message::GetString ( ) const [inline]**

Retrieve the string carried by this message as a whole.

**7.8.3.4 bool Message::IsFromWeb ( ) const [inline]**

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

**7.8.4 Field Documentation****7.8.4.1 std::string Message::fString [protected]**

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

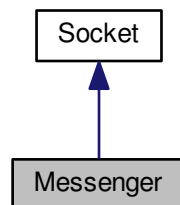
- include/Message.h

## 7.9 Messenger Class Reference

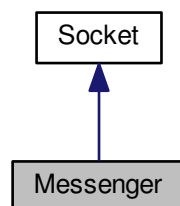
Base master object for the socket.

```
#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](#)

## Additional Inherited Members

### 7.9.1 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](mailto:laurent.forthomme@cern.ch)

#### Date

23 Mar 2015

### 7.9.2 Constructor & Destructor Documentation

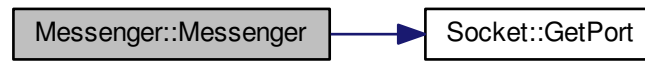
#### 7.9.2.1 [Messenger::Messenger](#) ( )

Build a void master object or socket actor.

#### 7.9.2.2 [Messenger::Messenger](#) ( int *port* )

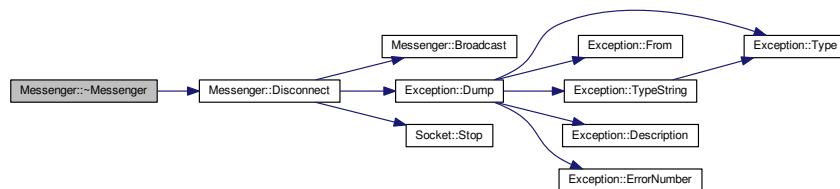
Build a master object to control the socket.

Here is the call graph for this function:



### 7.9.2.3 Messenger::~~Messenger ( )

Here is the call graph for this function:



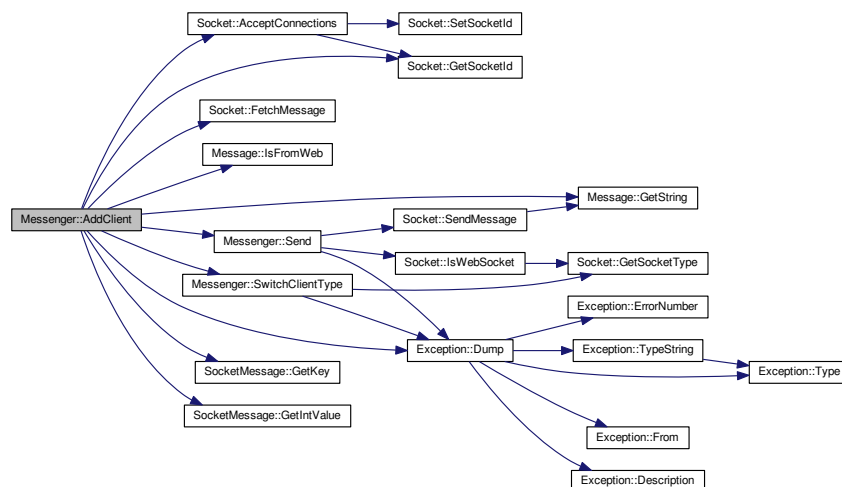
## 7.9.3 Member Function Documentation

### 7.9.3.1 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:



### 7.9.3.2 void Messenger::Broadcast ( const Message & m ) const

Emit a message to all clients connected through the socket.

Parameters

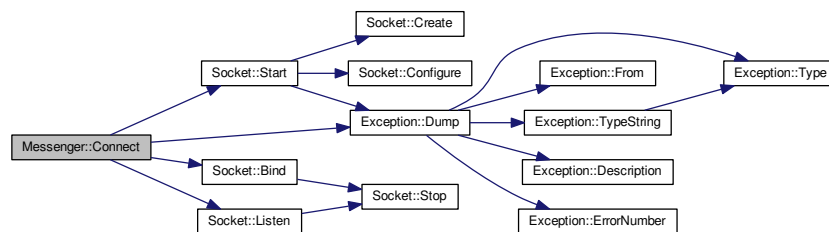
in	<i>m</i>	<a href="#">Message</a> to transmit
----	----------	-------------------------------------

### 7.9.3.3 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:

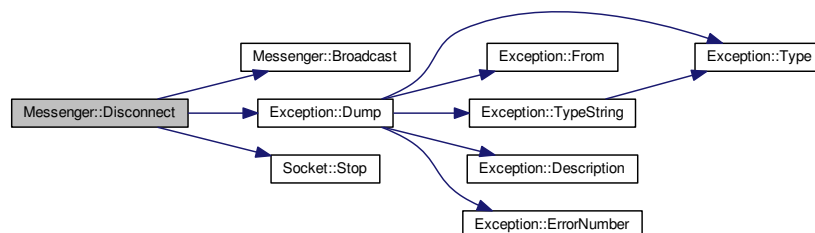


### 7.9.3.4 void Messenger::Disconnect ( )

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:



### 7.9.3.5 void Messenger::DisconnectClient ( int sid, MessageKey key, bool force = false ) [private]

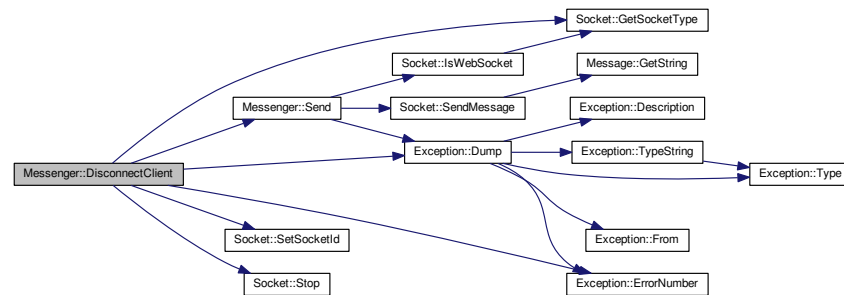
Disconnect a client.

Ask to a client to disconnect from this socket.

## Parameters

in	<i>sid</i>	Unique identifier of the client to disconnect
in	<i>key</i>	Key to the message to transmit for disconnection
in	<i>force</i>	Do we need to force the client out of this socket ?

Here is the call graph for this function:



### 7.9.3.6 SocketType Messenger::GetType ( ) const [inline]

Socket actor type retrieval method.

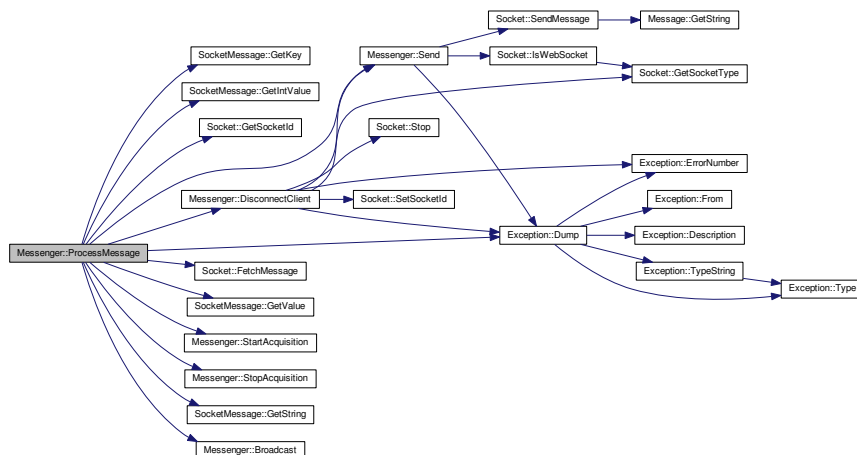
### 7.9.3.7 void Messenger::ProcessMessage ( SocketMessage m, int sid ) [private]

Process a message received from the socket.

## Parameters

in	<i>Unique</i>	identifier of the client sending the message
----	---------------	--

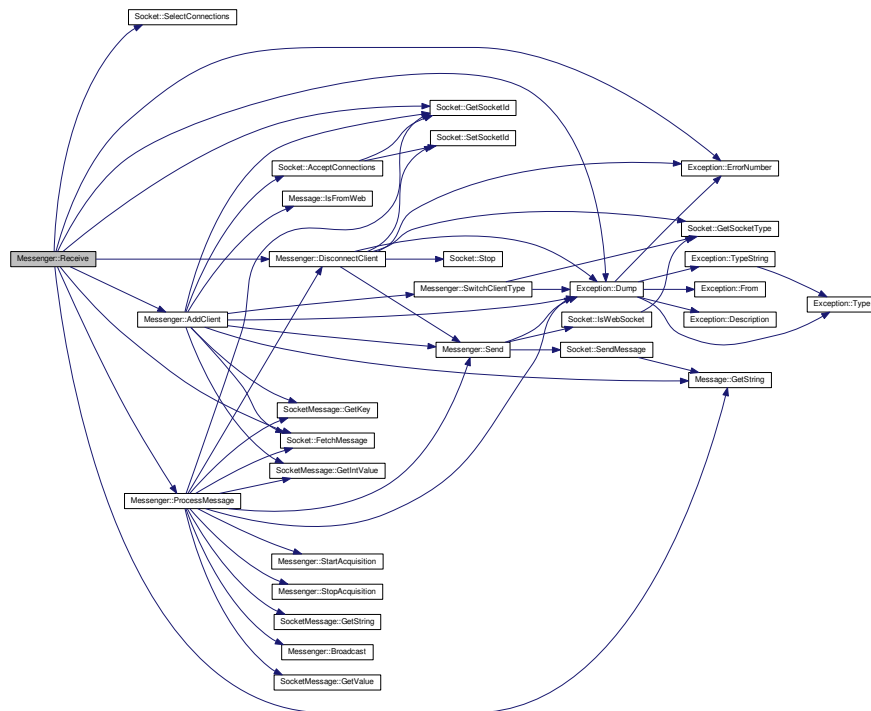
Here is the call graph for this function:



## 7.9.3.8 void Messenger::Receive ( )

Handle a message reception from a client.

Here is the call graph for this function:



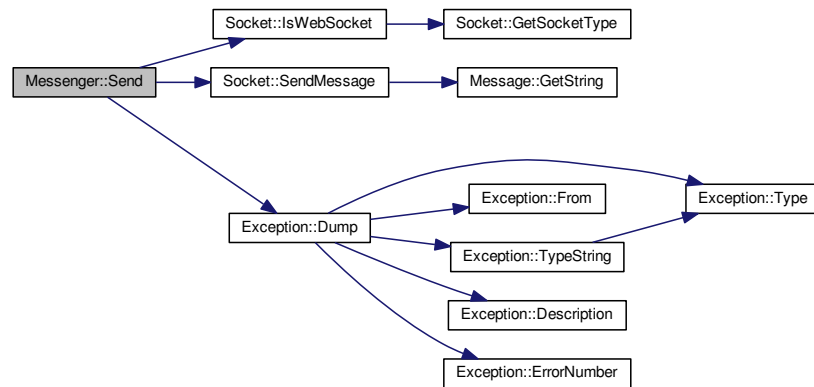
## 7.9.3.9 void Messenger::Send ( const Message &amp; m, int sid ) const [inline]

Send any type of message to any client.

## Parameters

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

Here is the call graph for this function:



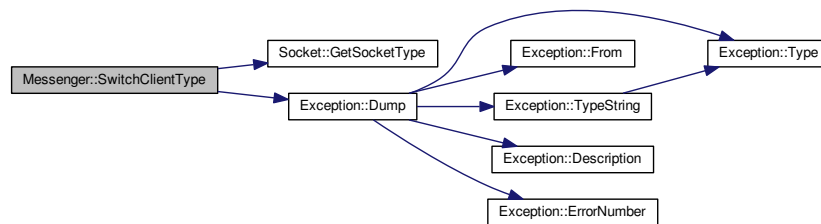
#### 7.9.3.10 void Messenger::StartAcquisition ( )

Start the data acquisition.

#### 7.9.3.11 void Messenger::StopAcquisition ( )

#### 7.9.3.12 void Messenger::SwitchClientType ( int sid, Socket::SocketType type ) [private]

Here is the call graph for this function:



### 7.9.4 Field Documentation

#### 7.9.4.1 int Messenger::fNumAttempts [private]

#### 7.9.4.2 pid\_t Messenger::fPID [private]

#### 7.9.4.3 WebSocket\* Messenger::fWS [private]

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

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

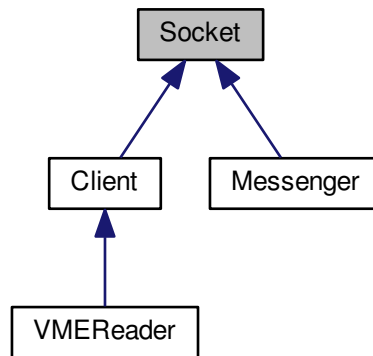


## 7.10 Socket Class Reference

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](#)

### 7.10.1 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](mailto:laurent.forthomme@cern.ch)

#### Date

23 Mar 2015

## 7.10.2 Member Typedef Documentation

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

## 7.10.3 Constructor & Destructor Documentation

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

7.10.3.2 `Socket::Socket ( int port )`

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

## 7.10.4 Member Function Documentation

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

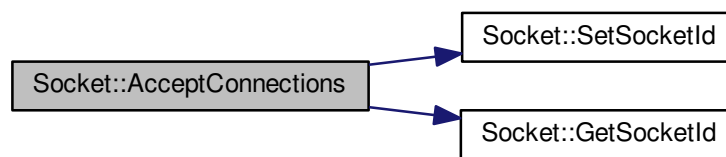
Accept connection from a client.

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

### Parameters

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

Here is the call graph for this function:



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

Bind a name to a socket.

### Returns

Success of the operation

Here is the call graph for this function:



#### 7.10.4.3 void Socket::Configure ( ) [private]

Configure the socket object for communication.

#### 7.10.4.4 void Socket::Create ( ) [private]

Create an endpoint for communication.

#### 7.10.4.5 void Socket::DumpConnected ( ) const

#### 7.10.4.6 Message Socket::FetchMessage ( int id = -1 ) const [protected]

Receive a message from a socket.

##### Returns

Received message as a std::string

#### 7.10.4.7 int Socket::GetPort ( ) const [inline]

Retrieve the port used for this socket.

#### 7.10.4.8 int Socket::GetSocketId ( ) const [inline]

#### 7.10.4.9 SocketType Socket::GetSocketType ( int sid ) const [inline]

#### 7.10.4.10 bool Socket::IsWebSocket ( int sid ) const [inline]

Here is the call graph for this function:



#### 7.10.4.11 void Socket::Listen ( int maxconn ) [protected]

Listen to incoming messages.

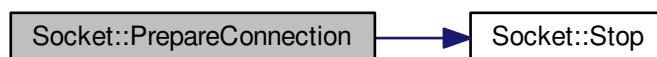
Set the socket to listen to any message coming from outside

Here is the call graph for this function:



#### 7.10.4.12 void Socket::PrepareConnection ( ) [protected]

Here is the call graph for this function:



#### 7.10.4.13 void Socket::SelectConnections ( )

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

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

Send a message on a socket.

Here is the call graph for this function:



#### 7.10.4.15 void Socket::SetPort ( int port ) [inline]

#### 7.10.4.16 void Socket::SetSocketId ( int sid ) [inline]

#### 7.10.4.17 `bool Socket::Start ( )` [protected]

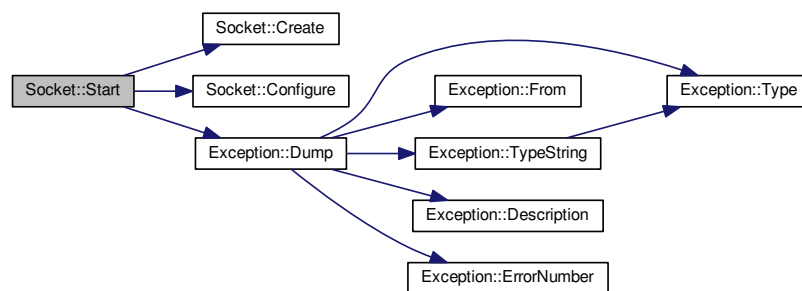
Start the socket.

Launch all mandatory operations to set the socket to be used

##### Returns

Success of the operation

Here is the call graph for this function:



#### 7.10.4.18 `void Socket::Stop ( )`

Terminates the socket and all attached communications.

### 7.10.5 Field Documentation

#### 7.10.5.1 `struct sockaddr_in Socket::fAddress` [private]

#### 7.10.5.2 `char Socket::fBuffer[MAX_WORD_LENGTH]` [protected]

#### 7.10.5.3 `fd_set Socket::fMaster` [protected]

Master file descriptor list.

#### 7.10.5.4 `int Socket::fPort` [protected]

#### 7.10.5.5 `fd_set Socket::fReadFds` [protected]

Temp file descriptor list for select()

#### 7.10.5.6 `int Socket::fSocketId` [private]

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

#### 7.10.5.7 `SocketCollection Socket::fSocketsConnected` [protected]

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

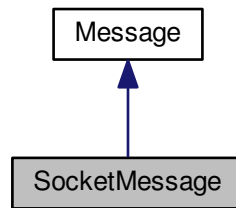
- `include/Socket.h`
- `src/Socket.cpp`

## 7.11 SocketMessage Class Reference

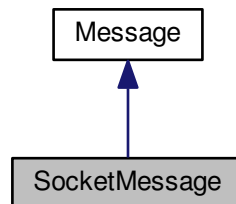
Socket-passed message type.

```
#include <SocketMessage.h>
```

Inheritance diagram for SocketMessage:



Collaboration diagram for SocketMessage:



### Public Member Functions

- [SocketMessage](#) ()
- [SocketMessage](#) (const [Message](#) &msg)
- [SocketMessage](#) (const char \*msg\_s)
- [SocketMessage](#) (std::string msg\_s)
- [SocketMessage](#) (const MessageKey &key)  
*Construct a socket message out of a key.*
- [SocketMessage](#) (const MessageKey &key, const char \*value)  
*Construct a socket message out of a key and a string-type value.*
- [SocketMessage](#) (const MessageKey &key, std::string value)  
*Construct a socket message out of a key and a string-type value.*

- [SocketMessage](#) (const MessageKey &key, const int value)  
*Construct a socket message out of a key and an integer-type value.*
- [SocketMessage](#) (const MessageKey &key, const float value)  
*Construct a socket message out of a key and a float-type value.*
- [SocketMessage](#) (const MessageKey &key, const double value)  
*Construct a socket message out of a key and a double precision-type value.*
- [SocketMessage](#) (MessageMap msg\_m)  
*Construct a socket message out of a map of key/string-type value.*
- [~SocketMessage](#) ()
- void [SetKeyValue](#) (const MessageKey &key, const char \*value)  
*String-valued message.*
- void [SetKeyValue](#) (const MessageKey &key, int int\_value)  
*Send an integer-valued message.*
- void [SetKeyValue](#) (const MessageKey &key, float float\_value)  
*Float-valued message.*
- void [SetKeyValue](#) (const MessageKey &key, double double\_value)  
*Double-valued message.*
- std::string [GetString](#) () const  
*Extract the whole key:value message.*
- MessageKey [GetKey](#) () const  
*Extract the message's key.*
- std::string [GetValue](#) () const  
*Extract the message's string value.*
- int [GetIntValue](#) () const  
*Extract the message's integer value.*
- VectorValue [GetVectorValue](#) () const  
*Extract the message's vector of string value.*
- void [Dump](#) (std::ostream &os=std::cout) const

## Private Member Functions

- MessageMap [Object](#) () const
- std::string [String](#) () const

## Private Attributes

- MessageMap [fMessage](#)

## Additional Inherited Members

### 7.11.1 Detailed Description

Socket-passed message type.

#### Author

Laurent Forthomme [laurent.forthomme@cern.ch](mailto:laurent.forthomme@cern.ch)

#### Date

26 Mar 2015



## 7.11.2 Constructor & Destructor Documentation

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

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

Here is the call graph for this function:



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

Here is the call graph for this function:



7.11.2.4 `SocketMessage::SocketMessage ( std::string msg_s ) [inline]`

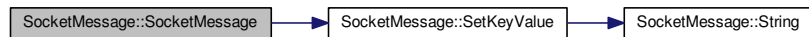
Here is the call graph for this function:



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

Construct a socket message out of a key.

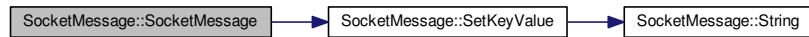
Here is the call graph for this function:



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

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

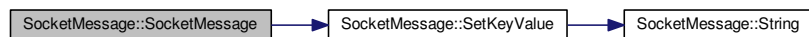
Here is the call graph for this function:



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

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

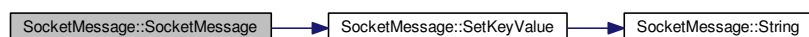
Here is the call graph for this function:



#### 7.11.2.8 `SocketMessage::SocketMessage ( const MessageKey & key, const int value ) [inline]`

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

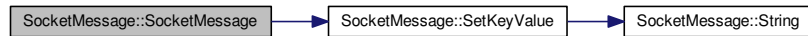
Here is the call graph for this function:



#### 7.11.2.9 `SocketMessage::SocketMessage ( const MessageKey & key, const float value ) [inline]`

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

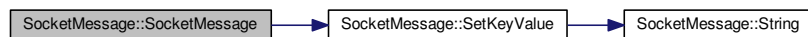
Here is the call graph for this function:



#### 7.11.2.10 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:



#### 7.11.2.11 SocketMessage::SocketMessage ( MessageMap msg\_m ) [inline]

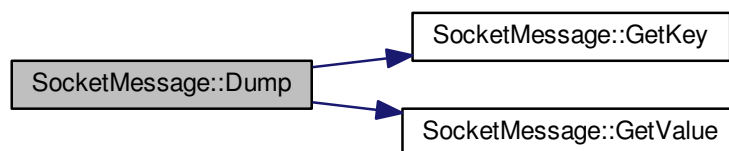
Construct a socket message out of a map of key/string-type value.

#### 7.11.2.12 SocketMessage::~~SocketMessage ( ) [inline]

### 7.11.3 Member Function Documentation

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

Here is the call graph for this function:



#### 7.11.3.2 int SocketMessage::GetIntValue ( ) const [inline]

Extract the message's integer value.

### 7.11.3.3 MessageKey SocketMessage::GetKey ( ) const [inline]

Extract the message's key.

### 7.11.3.4 std::string SocketMessage::GetString ( ) const [inline]

Extract the whole key:value message.

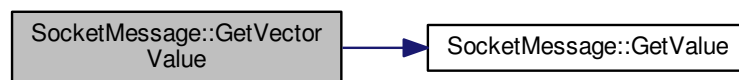
### 7.11.3.5 std::string SocketMessage::GetValue ( ) const [inline]

Extract the message's string value.

### 7.11.3.6 VectorValue SocketMessage::GetVectorValue ( ) const [inline]

Extract the message's vector of string value.

Here is the call graph for this function:



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

### 7.11.3.8 void SocketMessage::SetKeyValue ( const MessageKey & key, const char \* value ) [inline]

String-valued message.

Here is the call graph for this function:



### 7.11.3.9 void SocketMessage::SetKeyValue ( const MessageKey & key, int int\_value ) [inline]

Send an integer-valued message.

Here is the call graph for this function:



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

Float-valued message.

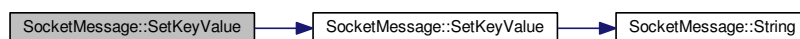
Here is the call graph for this function:



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

Double-valued message.

Here is the call graph for this function:



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

## 7.11.4 Field Documentation

7.11.4.1 `MessageMap SocketMessage::fMessage [private]`

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

- `include/SocketMessage.h`

## 7.12 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 uint32\_t &word)
- virtual `~TDCEvent` ()
- void `Dump` () const
- void `SetWord` (const uint32\_t &word)
- `EventType` `GetType` () const  
*Type of packet read out from the TDC.*
- uint8\_t `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)*
- uint8\_t `GetGeo` () const
- uint8\_t `GetChannelId` () const
- 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 `GetLeadingTime` (bool pair=false) const  
*Leading edge measurement in programmed time resolution.*
- uint8\_t `GetWidth` () const  
*Width of pulse in programmed time resolution.*
- uint32\_t `GetTrailingTime` () const  
*Trailing edge measurement in programmed time resolution.*
- uint8\_t `GetStatus` () const
- uint16\_t `GetErrorFlags` () const  
*Return error flags if an error condition has been detected.*

## Private Attributes

- uint32\_t `fWord`

### 7.12.1 Detailed Description

HPTDC event parser.

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

## Author

Laurent Forthomme [laurent.forthomme@cern.ch](mailto:laurent.forthomme@cern.ch)

## Date

4 May 2015

## 7.12.2 Member Enumeration Documentation

### 7.12.2.1 enum VME::TDCEvent::EventType

## Enumerator

***TDCMeasurement***

***TDCHeader***

***TDCTrailer***

***TDCErrors***

***GlobalHeader***

***GlobalTrailer***

***ETTT***

***Filler***

## 7.12.3 Constructor & Destructor Documentation

7.12.3.1 VME::TDCEvent::TDCEvent ( ) [inline]

7.12.3.2 VME::TDCEvent::TDCEvent ( const uint32\_t & word ) [inline]

7.12.3.3 virtual VME::TDCEvent::~~TDCEvent ( ) [inline], [virtual]

## 7.12.4 Member Function Documentation

7.12.4.1 void VME::TDCEvent::Dump ( ) const [inline]

Here is the call graph for this function:



7.12.4.2 uint16\_t VME::TDCEvent::GetBunchId ( ) const [inline]

Bunch identifier of trigger (or trigger time tag)

Here is the call graph for this function:



#### 7.12.4.3 `uint8_t VME::TDCEvent::GetChannelId ( ) const [inline]`

Here is the call graph for this function:



#### 7.12.4.4 `uint16_t VME::TDCEvent::GetErrorFlags ( ) const [inline]`

Return error flags if an error condition has been detected.

Here is the call graph for this function:



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

Extended trigger time tag.

Here is the call graph for this function:





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

Total number of events.

Here is the call graph for this function:



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

Event identifier from event counter.

Here is the call graph for this function:



#### 7.12.4.8 `uint8_t VME::TDCEvent::GetGeo ( ) const [inline]`

Here is the call graph for this function:



#### 7.12.4.9 `uint32_t VME::TDCEvent::GetLeadingTime ( bool pair = false ) const [inline]`

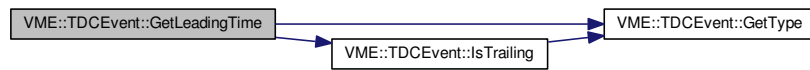
Leading edge measurement in programmed time resolution.

**Parameters**

---

<i>in</i>	<i>pair</i>	Are we dealing with a pair measurement?
-----------	-------------	---

Here is the call graph for this function:



#### 7.12.4.10 `uint8_t VME::TDCEvent::GetStatus ( ) const [inline]`

Here is the call graph for this function:



#### 7.12.4.11 `uint8_t VME::TDCEvent::GetTDCId ( ) const [inline]`

Programmed identifier of master TDC providing the event.

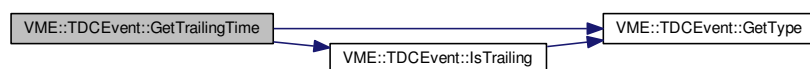
Here is the call graph for this function:



#### 7.12.4.12 `uint32_t VME::TDCEvent::GetTrailingTime ( ) const [inline]`

Trailing edge measurement in programmed time resolution.

Here is the call graph for this function:



#### 7.12.4.13 EventType VME::TDCEvent::GetType ( ) const [inline]

Type of packet read out from the TDC.

#### 7.12.4.14 uint8\_t VME::TDCEvent::GetWidth ( ) const [inline]

Width of pulse in programmed time resolution.

Here is the call graph for this function:



#### 7.12.4.15 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:



#### 7.12.4.16 bool VME::TDCEvent::IsTrailing ( ) const [inline]

Are we dealing with a trailing or a leading measurement?

Here is the call graph for this function:



#### 7.12.4.17 void VME::TDCEvent::SetWord ( const uint32\_t & word ) [inline]

### 7.12.5 Field Documentation

### 7.12.5.1 uint32\_t VME::TDCEvent::fWord [private]

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

- include/VME\_TDCEvent.h

## 7.13 VME::TDCV1x90 Class Reference

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

### Public Member Functions

- [TDCV1x90](#) (int32\_t, uint32\_t, [acq\\_mode](#) acqm=TRIG\_MATCH, [det\\_mode](#) detm=TRAILEAD)
- [~TDCV1x90](#) ()
- void [SetVerboseLevel](#) (unsigned short verb=1)
- uint32\_t [GetModel](#) ()
- uint32\_t [GetOUI](#) ()
- uint32\_t [GetSerialNumber](#) ()
- void [CheckConfiguration](#) ()
- void [EnableChannel](#) (short)
- void [DisableChannel](#) (short)
- void [SetPol](#) (uint16\_t)
- void [SetLSBTraileadEdge](#) (trailead\_edge\_lsb)
- void [SetAcquisitionMode](#) (acq\_mode)
- bool [SetTriggerMatching](#) ()
- bool [IsTriggerMatching](#) ()
- bool [SetContinuousStorage](#) ()
- void [GetFirmwareRev](#) ()
- void [SetGlobalOffset](#) (uint16\_t, uint16\_t)
- [glob\\_offs](#) [ReadGlobalOffset](#) ()
- void [SetRCAdjust](#) (int, uint16\_t)
- uint16\_t [ReadRCAdjust](#) (int)
- uint32\_t [GetEventCounter](#) ()
- uint16\_t [GetEventStored](#) ()
- void [SetDetection](#) (det\_mode)
- [det\\_mode](#) [ReadDetection](#) ()
- void [SetTDCEncapsulation](#) (bool)
- bool [GetTDCEncapsulation](#) ()
- void [SetTDCErrorMarks](#) (bool)
- void [ReadResolution](#) (det\_mode)
- void [SetPairModeResolution](#) (int, int)
- void [SetBLTEventNumberRegister](#) (uint16\_t)
- uint16\_t [GetBLTEventNumberRegister](#) ()
- void [SetWindowWidth](#) (uint16\_t)
- void [SetWindowOffset](#) (int16\_t)
- uint16\_t [ReadTrigConf](#) (trig\_conf)
- bool [WaitMicro](#) (micro\_handshake)
- bool [SoftwareClear](#) ()
- bool [SoftwareReset](#) ()
- bool [HardwareReset](#) ()
- bool [GetStatusRegister](#) (stat\_reg)
- void [SetStatusRegister](#) (stat\_reg, bool)
- bool [GetCtlRegister](#) (ctl\_reg)

- void [SetCtlRegister](#) ([ctl\\_reg](#), bool)
- void [SetETTT](#) (bool)
- bool [GetETTT](#) ()
- [TDCEventCollection](#) [FetchEvents](#) ()
- void [SetFIFOSize](#) (uint16\_t)
- void [ReadFIFOSize](#) ()
- void [abort](#) ()
- void [WriteRegister](#) ([mod\\_reg](#), uint16\_t \*)  
*Write on register.*
- void [WriteRegister](#) ([mod\\_reg](#), uint32\_t \*)  
*Write on register.*
- void [ReadRegister](#) ([mod\\_reg](#), uint16\_t \*)  
*Read on register.*
- void [ReadRegister](#) ([mod\\_reg](#), uint32\_t \*)  
*Read on register.*

### Private Attributes

- uint32\_t [fBaseAddr](#)
- int32\_t [fHandle](#)
- [det\\_mode](#) [fDetMode](#)
- unsigned short [fVerb](#)
- CVAddressModifier [am](#)
- CVAddressModifier [am\\_blt](#)
- uint32\_t \* [fBuffer](#)
- [det\\_mode](#) [detm](#)
- [acq\\_mode](#) [acqm](#)
- bool [outBufTDCHeadTrail](#)
- bool [outBufTDCErr](#)
- bool [outBufTDCTTT](#)
- uint32\_t [nchannels](#)
- bool [gEnd](#)
- std::string [pair\\_lead\\_res](#) [8]
- std::string [pair\\_width\\_res](#) [16]
- std::string [trailead\\_edge\\_res](#) [4]

### 7.13.1 Detailed Description

#### Author

Laurent Forthomme [laurent.forthomme@cern.ch](mailto:laurent.forthomme@cern.ch)  
 Bob Velghe [bob.velghe@cern.ch](mailto:bob.velghe@cern.ch)

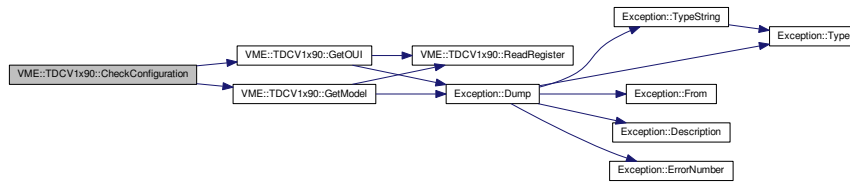
#### Date

Jun 2010



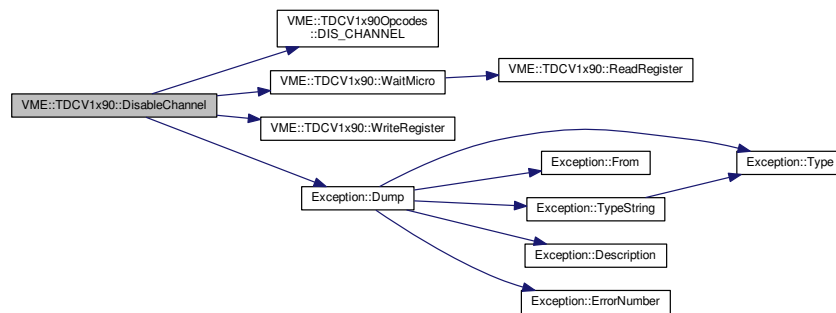
## 7.13.3.2 void VME::TDCV1x90::CheckConfiguration ( )

Here is the call graph for this function:



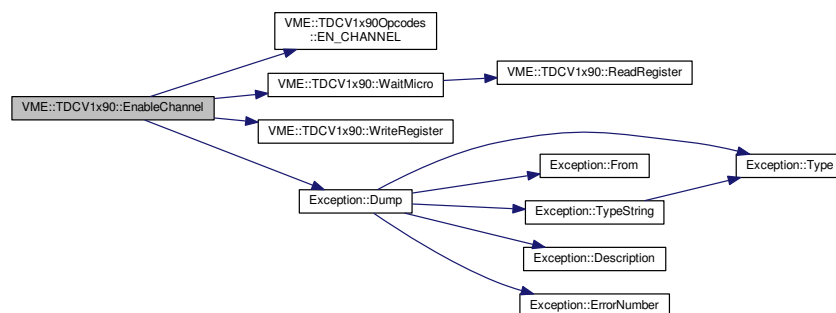
## 7.13.3.3 void VME::TDCV1x90::DisableChannel ( short channel\_id )

Here is the call graph for this function:



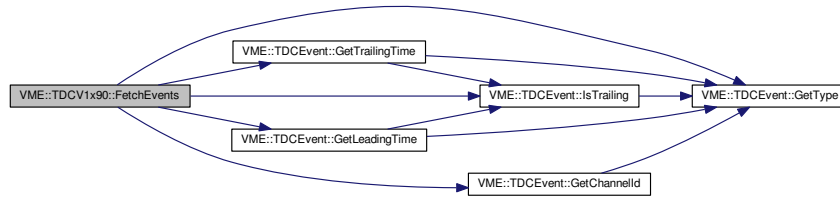
## 7.13.3.4 void VME::TDCV1x90::EnableChannel ( short channel\_id )

Here is the call graph for this function:



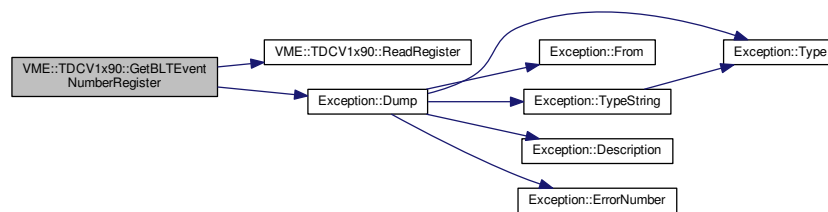
### 7.13.3.5 TDCEventCollection VME::TDCV1x90::FetchEvents ( )

Here is the call graph for this function:



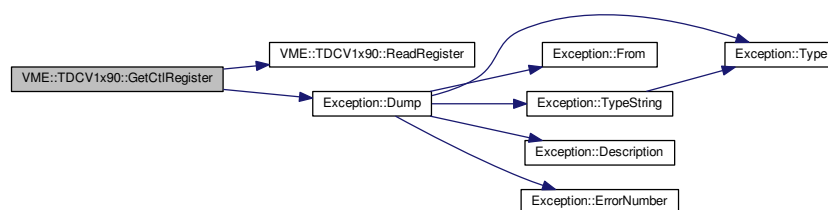
### 7.13.3.6 uint16\_t VME::TDCV1x90::GetBLTEventNumberRegister ( )

Here is the call graph for this function:



### 7.13.3.7 bool VME::TDCV1x90::GetCtlRegister ( ctl\_reg bit )

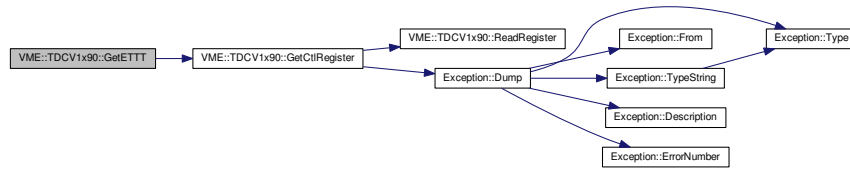
Here is the call graph for this function:





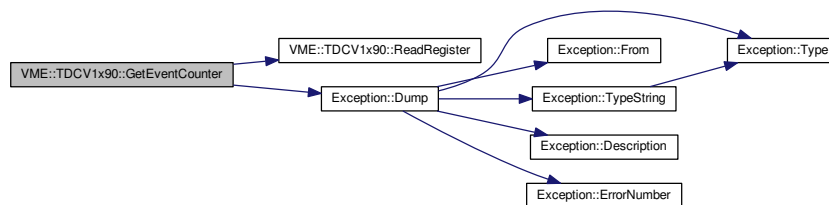
## 7.13.3.8 bool VME::TDCV1x90::GetETTT ( )

Here is the call graph for this function:



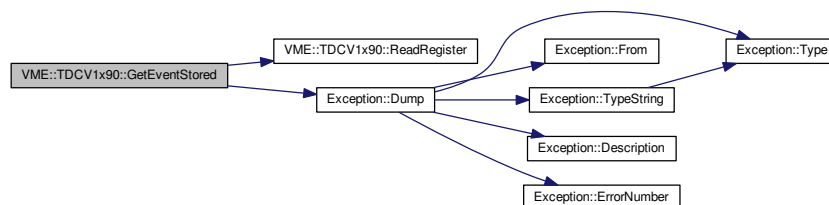
## 7.13.3.9 uint32\_t VME::TDCV1x90::GetEventCounter ( )

Here is the call graph for this function:



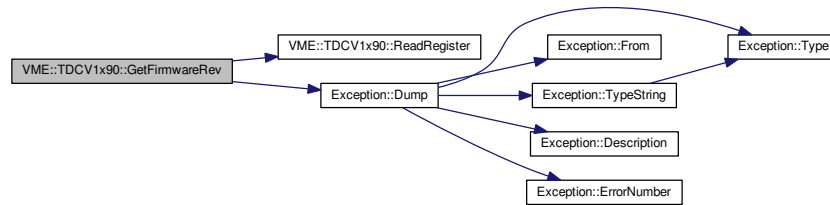
## 7.13.3.10 uint16\_t VME::TDCV1x90::GetEventStored ( )

Here is the call graph for this function:



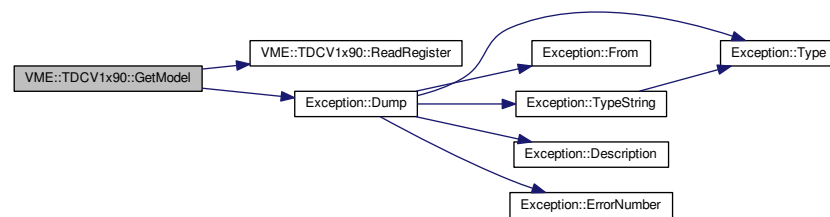
#### 7.13.3.11 void VME::TDCV1x90::GetFirmwareRev ( )

Here is the call graph for this function:



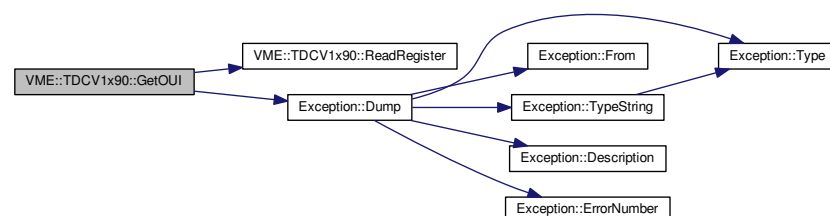
#### 7.13.3.12 uint32\_t VME::TDCV1x90::GetModel ( )

Here is the call graph for this function:



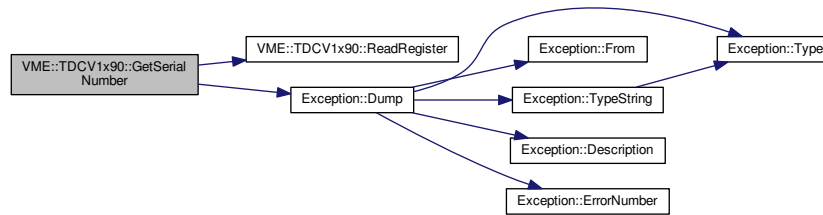
#### 7.13.3.13 uint32\_t VME::TDCV1x90::GetOUI ( )

Here is the call graph for this function:



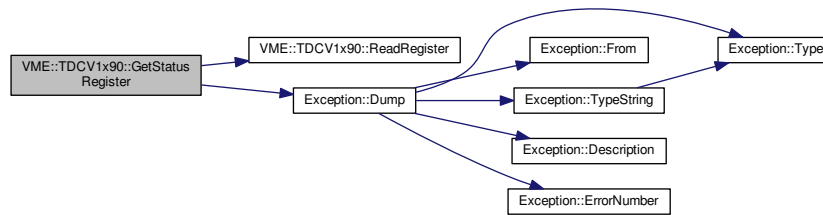
## 7.13.3.14 uint32\_t VME::TDCV1x90::GetSerialNumber ( )

Here is the call graph for this function:



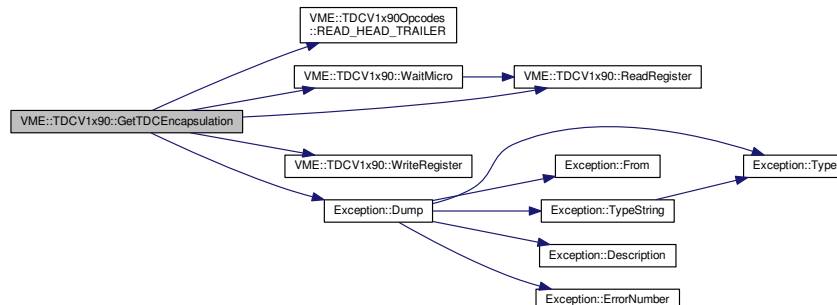
## 7.13.3.15 bool VME::TDCV1x90::GetStatusRegister ( stat\_reg bit )

Here is the call graph for this function:



## 7.13.3.16 bool VME::TDCV1x90::GetTDCEncapsulation ( )

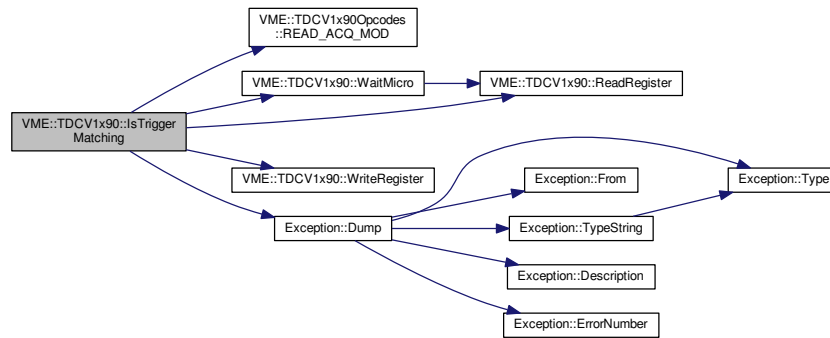
Here is the call graph for this function:



## 7.13.3.17 bool VME::TDCV1x90::HardwareReset ( )

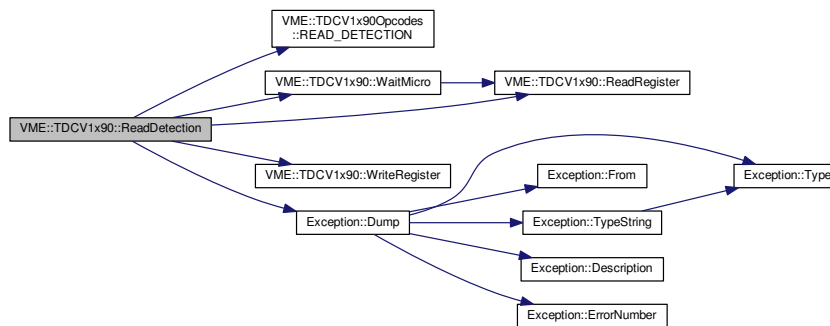
### 7.13.3.18 bool VME::TDCV1x90::IsTriggerMatching ( )

Here is the call graph for this function:



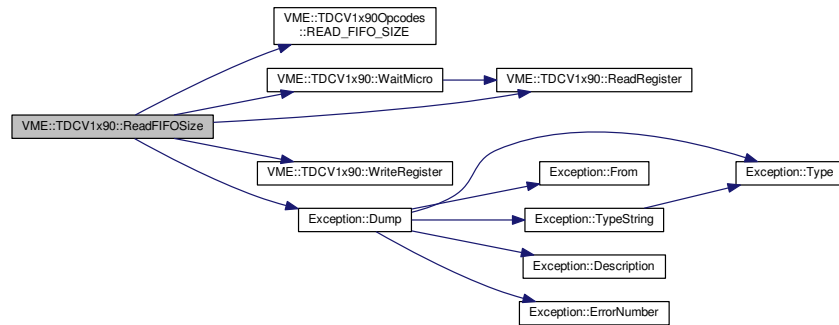
### 7.13.3.19 det\_mode VME::TDCV1x90::ReadDetection ( )

Here is the call graph for this function:



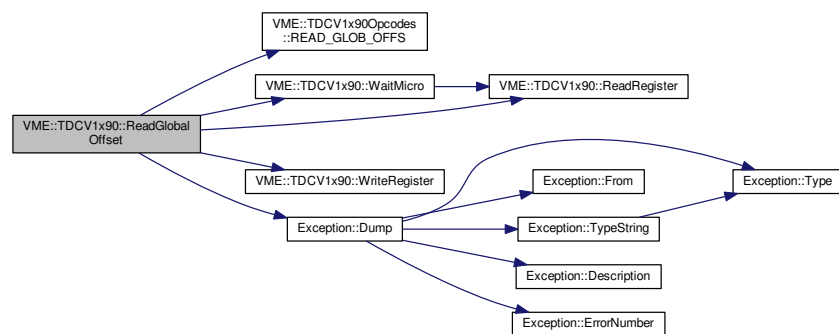
## 7.13.3.20 void VME::TDCV1x90::ReadFIFOSize ( )

Here is the call graph for this function:



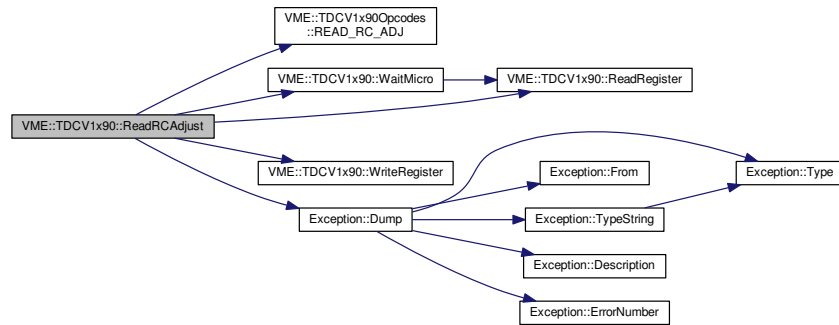
## 7.13.3.21 glob\_offs VME::TDCV1x90::ReadGlobalOffset ( )

Here is the call graph for this function:



### 7.13.3.22 uint16\_t VME::TDCV1x90::ReadRCAdjust ( int *tdc* )

Here is the call graph for this function:



### 7.13.3.23 void VME::TDCV1x90::ReadRegister ( mod\_reg *addr*, uint16\_t \* *data* )

Read on register.

Read a 16-bit word in the register

#### Parameters

in	<i>addr</i>	register
out	<i>data</i>	word

### 7.13.3.24 void VME::TDCV1x90::ReadRegister ( mod\_reg *addr*, uint32\_t \* *data* )

Read on register.

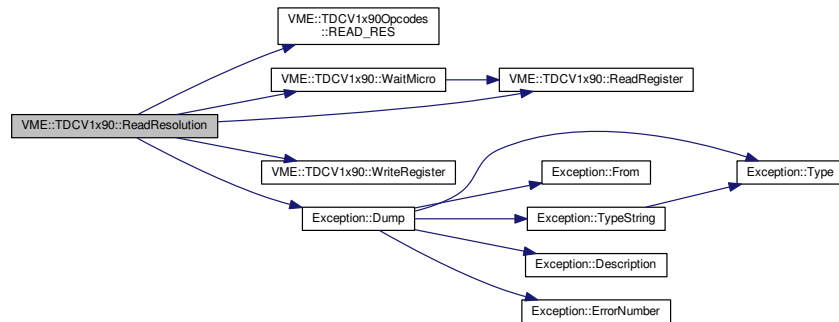
Read a 32-bit word in the register

#### Parameters

in	<i>addr</i>	register
out	<i>data</i>	word

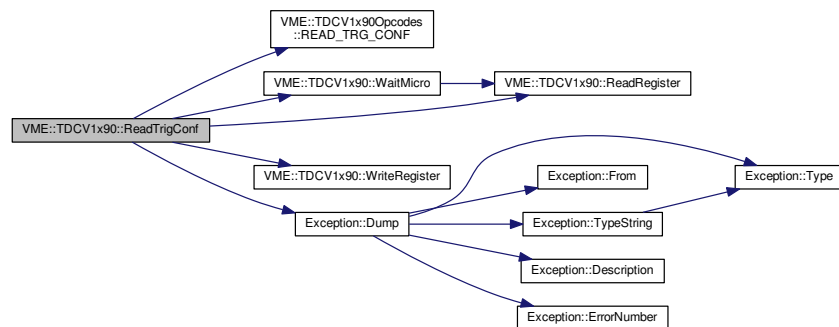
## 7.13.3.25 void VME::TDCV1x90::ReadResolution ( det\_mode det )

Here is the call graph for this function:



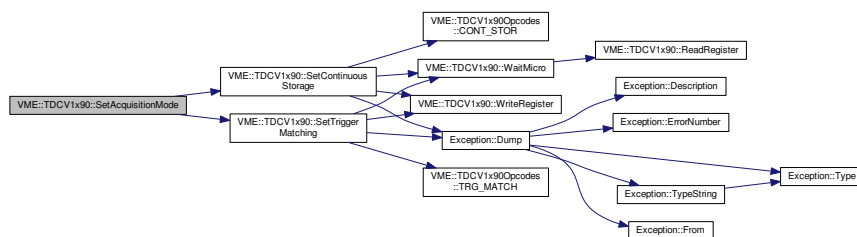
## 7.13.3.26 uint16\_t VME::TDCV1x90::ReadTrigConf ( trig\_conf type )

Here is the call graph for this function:



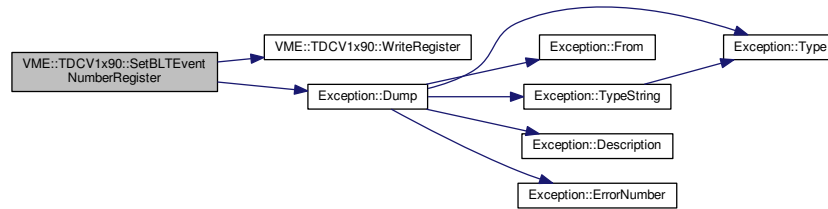
## 7.13.3.27 void VME::TDCV1x90::SetAcquisitionMode ( acq\_mode mode )

Here is the call graph for this function:



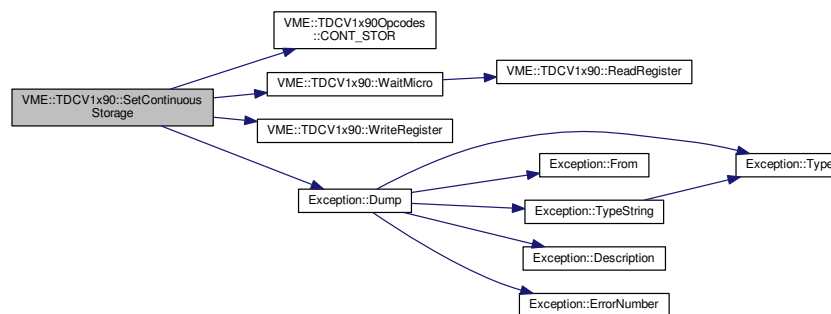
### 7.13.3.28 void VME::TDCV1x90::SetBLTEventNumberRegister ( uint16\_t value )

Here is the call graph for this function:



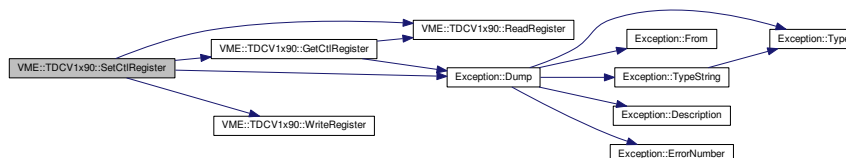
### 7.13.3.29 bool VME::TDCV1x90::SetContinuousStorage ( )

Here is the call graph for this function:



### 7.13.3.30 void VME::TDCV1x90::SetCtlRegister ( ctl\_reg reg, bool value )

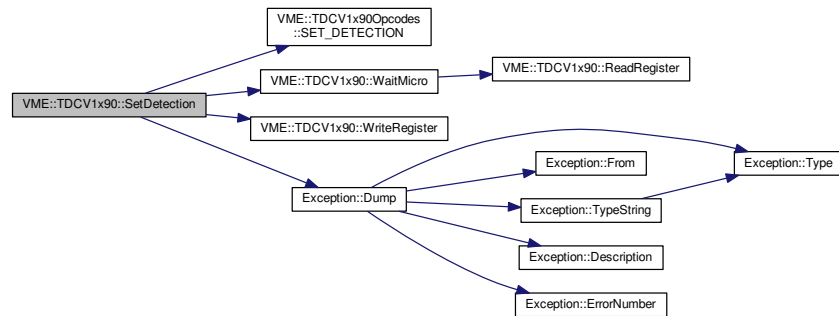
Here is the call graph for this function:





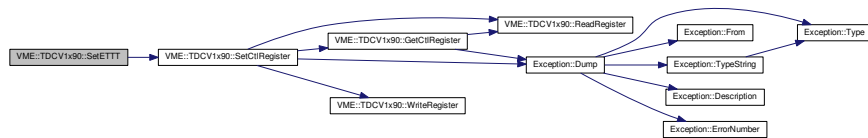
## 7.13.3.31 void VME::TDCV1x90::SetDetection ( det\_mode mode )

Here is the call graph for this function:



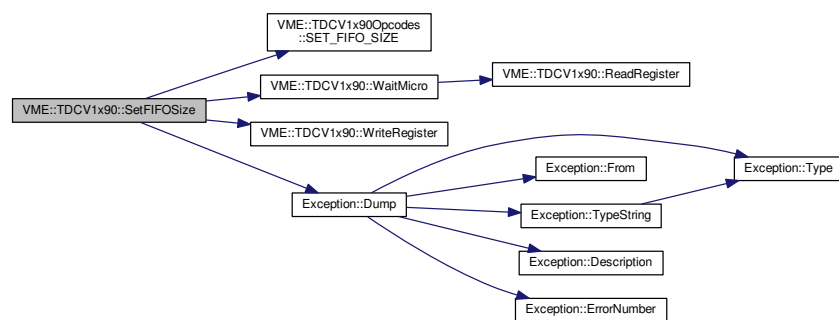
## 7.13.3.32 void VME::TDCV1x90::SetETTT ( bool mode )

Here is the call graph for this function:



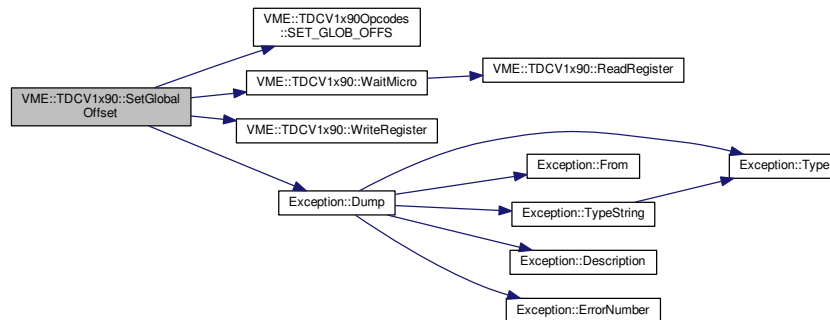
## 7.13.3.33 void VME::TDCV1x90::SetFIFOSize ( uint16\_t size )

Here is the call graph for this function:



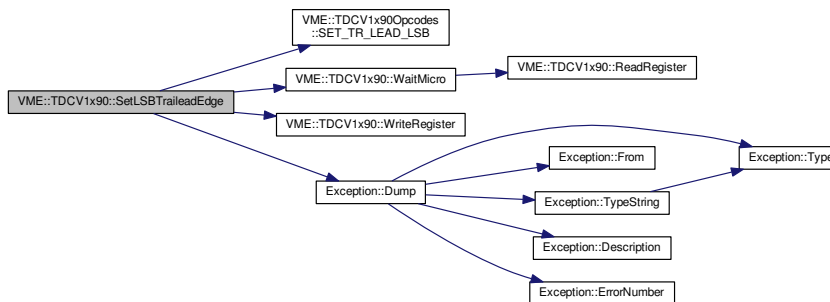
#### 7.13.3.34 void VME::TDCV1x90::SetGlobalOffset ( uint16\_t word1, uint16\_t word2 )

Here is the call graph for this function:



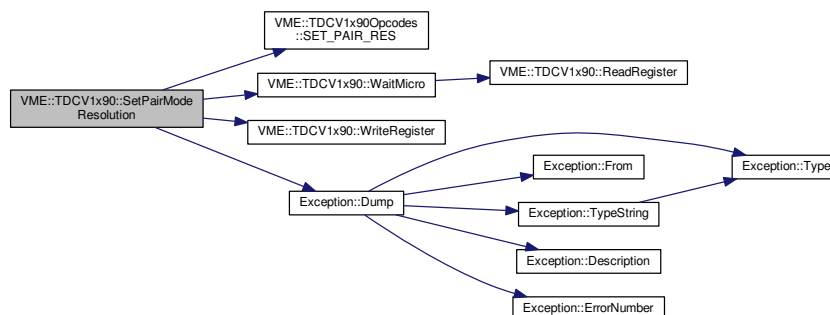
#### 7.13.3.35 void VME::TDCV1x90::SetLSBTrailEdge ( trailEdge\_lsb conf )

Here is the call graph for this function:



#### 7.13.3.36 void VME::TDCV1x90::SetPairModeResolution ( int lead\_time\_res, int pulse\_width\_res )

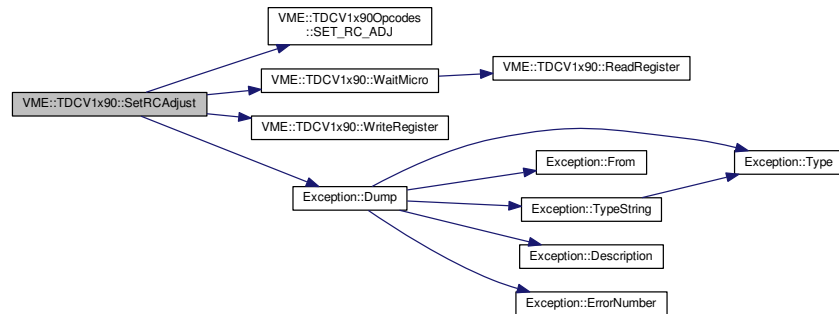
Here is the call graph for this function:



7.13.3.37 void VME::TDCV1x90::SetPol ( uint16\_t word )

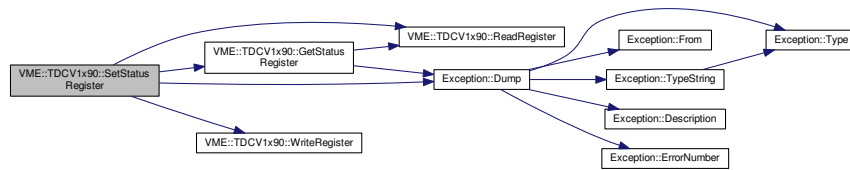
7.13.3.38 void VME::TDCV1x90::SetRCAdjust ( int tdc, uint16\_t value )

Here is the call graph for this function:



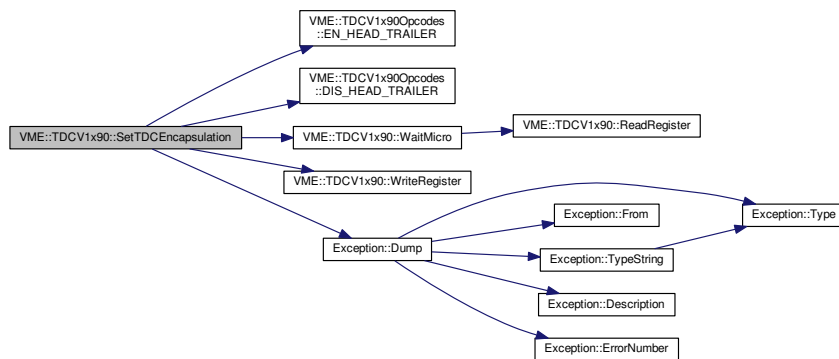
7.13.3.39 void VME::TDCV1x90::SetStatusRegister ( stat\_reg reg, bool value )

Here is the call graph for this function:



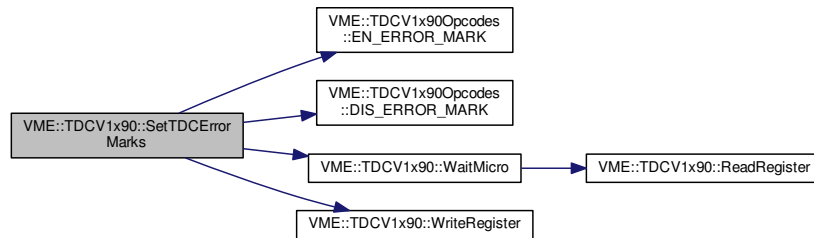
7.13.3.40 void VME::TDCV1x90::SetTDCEncapsulation ( bool mode )

Here is the call graph for this function:



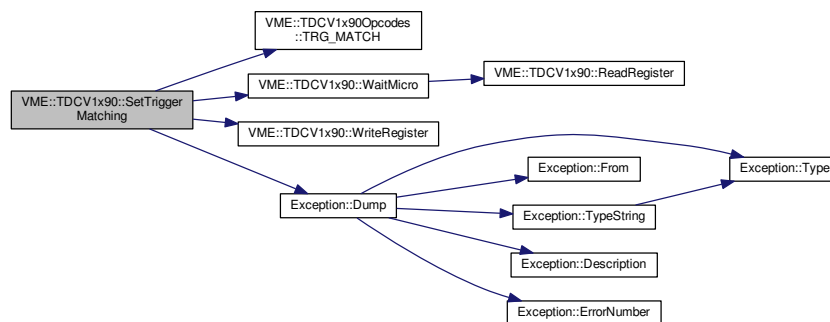
#### 7.13.3.41 void VME::TDCV1x90::SetTDCErrorMarks ( bool mode )

Here is the call graph for this function:



#### 7.13.3.42 bool VME::TDCV1x90::SetTriggerMatching ( )

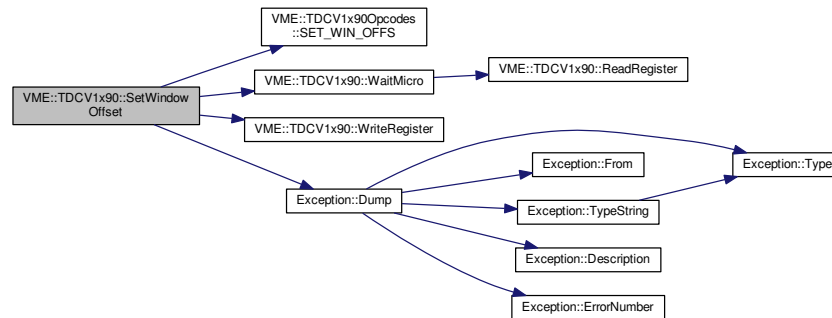
Here is the call graph for this function:



#### 7.13.3.43 void VME::TDCV1x90::SetVerboseLevel ( unsigned short verb = 1 ) [inline]

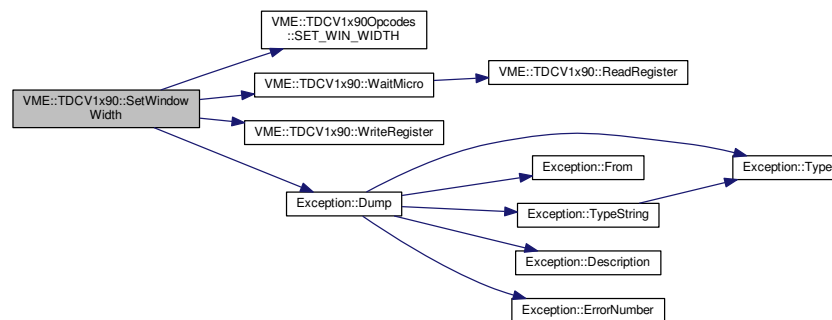
## 7.13.3.44 void VME::TDCV1x90::SetWindowOffset ( int16\_t offs )

Here is the call graph for this function:



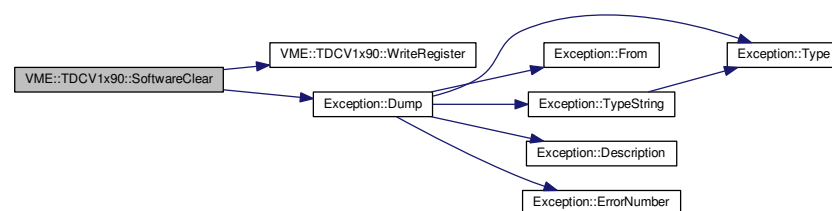
## 7.13.3.45 void VME::TDCV1x90::SetWindowWidth ( uint16\_t width )

Here is the call graph for this function:



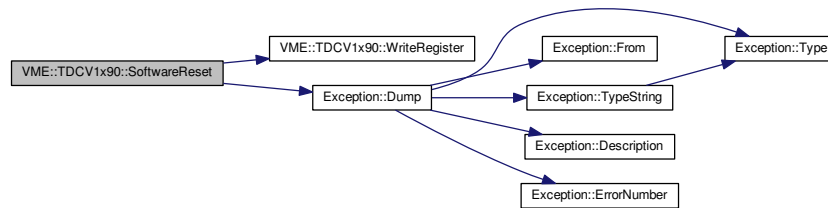
## 7.13.3.46 bool VME::TDCV1x90::SoftwareClear ( )

Here is the call graph for this function:



#### 7.13.3.47 bool VME::TDCV1x90::SoftwareReset ( )

Here is the call graph for this function:



#### 7.13.3.48 bool VME::TDCV1x90::WaitMicro ( micro\_handshake mode )

Here is the call graph for this function:



#### 7.13.3.49 void VME::TDCV1x90::WriteRegister ( mod\_reg addr, uint16\_t \* data )

Write on register.

Write a 16-bit word in the register

Parameters

in	<i>addr</i>	register
in	<i>data</i>	word

#### 7.13.3.50 void VME::TDCV1x90::WriteRegister ( mod\_reg addr, uint32\_t \* data )

Write on register.

Write a 32-bit word in the register

Parameters

in	<i>addr</i>	register
in	<i>data</i>	word

### 7.13.4 Field Documentation

#### 7.13.4.1 acq\_mode VME::TDCV1x90::acqm [private]

- 7.13.4.2 CVAddressModifier VME::TDCV1x90::am [private]
- 7.13.4.3 CVAddressModifier VME::TDCV1x90::am\_blt [private]
- 7.13.4.4 det\_mode VME::TDCV1x90::detm [private]
- 7.13.4.5 uint32\_t VME::TDCV1x90::fBaseAddr [private]
- 7.13.4.6 uint32\_t\* VME::TDCV1x90::fBuffer [private]
- 7.13.4.7 det\_mode VME::TDCV1x90::fDetMode [private]
- 7.13.4.8 int32\_t VME::TDCV1x90::fHandle [private]
- 7.13.4.9 unsigned short VME::TDCV1x90::fVerb [private]
- 7.13.4.10 bool VME::TDCV1x90::gEnd [private]
- 7.13.4.11 uint32\_t VME::TDCV1x90::nchannels [private]
- 7.13.4.12 bool VME::TDCV1x90::outBufTDCErr [private]
- 7.13.4.13 bool VME::TDCV1x90::outBufTDCHeadTrail [private]
- 7.13.4.14 bool VME::TDCV1x90::outBufTDCTTT [private]
- 7.13.4.15 std::string VME::TDCV1x90::pair\_lead\_res[8] [private]
- 7.13.4.16 std::string VME::TDCV1x90::pair\_width\_res[16] [private]
- 7.13.4.17 std::string VME::TDCV1x90::trailead\_edge\_res[4] [private]

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

- include/VME\_TDCV1x90.h
- src/VME\_TDCV1x90.cpp

## 7.14 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 [ett](#)

### 7.14.1 Field Documentation

- 7.14.1.1 uint32\_t VME::trailead\_t::ett

7.14.1.2 `uint32_t VME::trailead_t::event_count`

7.14.1.3 `std::multimap<int32_t,int32_t> VME::trailead_t::leading`

7.14.1.4 `int VME::trailead_t::total_hits[16]`

7.14.1.5 `std::multimap<int32_t,int32_t> VME::trailead_t::trailing`

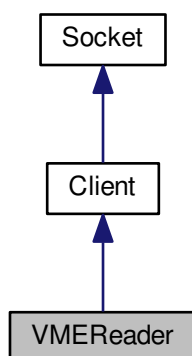
The documentation for this struct was generated from the following file:

- `include/VME_TDCV1x90.h`

## 7.15 VMEReader Class Reference

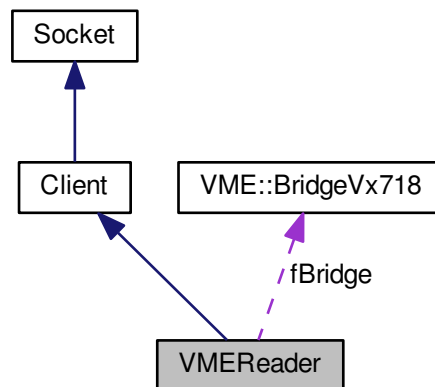
```
#include <VMEReader.h>
```

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.*
- unsigned int [GetRunNumber](#) ()  
*Ask the socket master a run number.*
- 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.*
- bool [fOnSocket](#)  
*Are we dealing with socket message passing?*

## Additional Inherited Members

### 7.15.1 Detailed Description

VME reader object to fetch events on a HPTDC board

#### Author

Laurent Forthomme [laurent.forthomme@cern.ch](mailto:laurent.forthomme@cern.ch)

#### Date

4 May 2015

### 7.15.2 Member Typedef Documentation

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

Mapper from physical VME addresses to pointers to TDC objects.

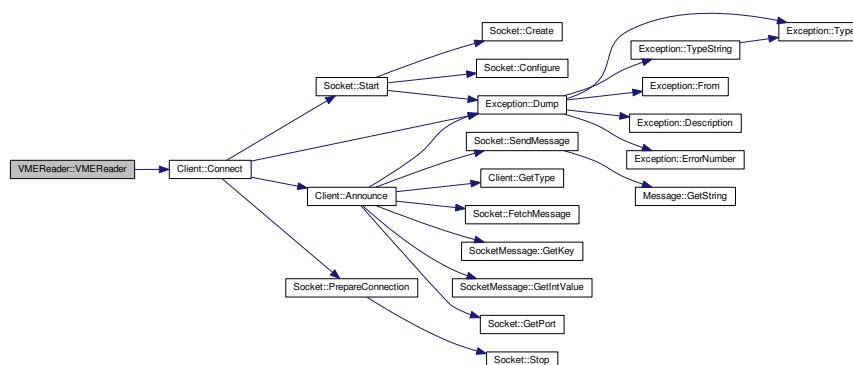
### 7.15.3 Constructor & Destructor Documentation

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

#### Parameters

in	<i>device</i>	Path to the device (/dev/xxx)
in	<i>type</i>	Bridge model
in	<i>on_socket</i>	Are we trying to connect through the socket?

Here is the call graph for this function:



7.15.3.2 `VMEReader::~~VMEReader ( )` [virtual]

### 7.15.4 Member Function Documentation

7.15.4.1 `void VMEReader::Abort ( )`

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

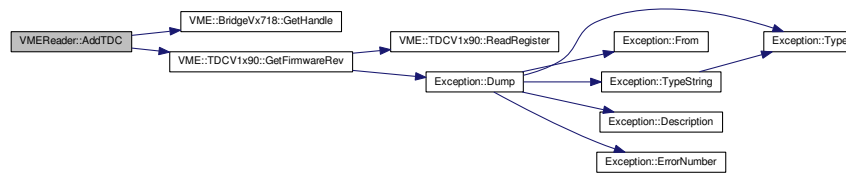
#### 7.15.4.2 void VMEReader::AddTDC ( uint32\_t *address* )

Add a TDC to handle.

## Parameters

in	<i>address</i>	32-bit address of the TDC module on the <a href="#">VME</a> bus Create a new TDC handler for the <a href="#">VME</a> bus
----	----------------	--

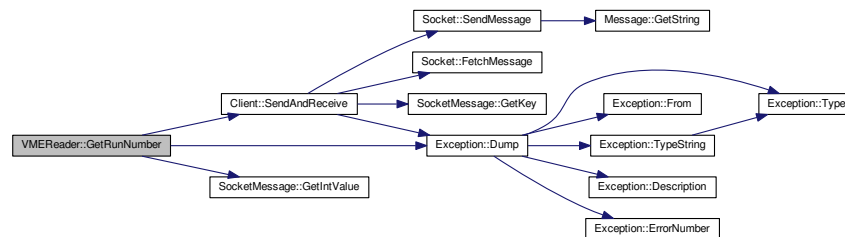
Here is the call graph for this function:



#### 7.15.4.3 unsigned int VMEReader::GetRunNumber ( )

Ask the socket master a run number.

Here is the call graph for this function:



#### 7.15.4.4 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.

### 7.15.5 Field Documentation

#### 7.15.5.1 VME::BridgeVx718\* VMEReader::fBridge [private]

The [VME](#) bridge object to handle.

#### 7.15.5.2 bool VMEReader::fOnSocket [private]

Are we dealing with socket message passing?

#### 7.15.5.3 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

- ~BridgeVx718
  - VME::BridgeVx718, [20](#)
- ~Client
  - Client, [23](#)
- ~Exception
  - Exception, [27](#)
- ~FileReader
  - FileReader, [30](#)
- ~Message
  - Message, [35](#)
- ~Messenger
  - Messenger, [38](#)
- ~Socket
  - Socket, [45](#)
- ~SocketMessage
  - SocketMessage, [53](#)
- ~TDCEvent
  - VME::TDCEvent, [57](#)
- ~TDCV1x90
  - VME::TDCV1x90, [64](#)
- ~VMEReader
  - VMEReader, [84](#)
- ALIGN64
  - VME, [12](#)
- ALM\_FULL
  - VME, [14](#)
- AUTOLOAD\_DEF\_CONFI
  - VME::TDCV1x90Opcodes, [16](#)
- AUTOLOAD\_USER\_CONF
  - VME::TDCV1x90Opcodes, [16](#)
- Abort
  - VMEReader, [84](#)
- abort
  - VME::TDCV1x90, [64](#)
- AcceptConnections
  - Socket, [45](#)
- acq\_mode
  - VME, [12](#)
- acqm
  - VME::TDCV1x90, [80](#)
- AddClient
  - Messenger, [38](#)
- AddTDC
  - VMEReader, [84](#)
- am
  - VME::TDCV1x90, [80](#)
- am\_blt
  - VME::TDCV1x90, [81](#)
- Announce
  - Client, [23](#)
- BERR\_FLAG
  - VME, [14](#)
- BERREN
  - VME, [12](#)
- BLTEventNumber
  - VME, [13](#)
- Bind
  - Socket, [45](#)
- BridgeType
  - VME, [12](#)
- BridgeVx718
  - VME::BridgeVx718, [20](#)
- Broadcast
  - Messenger, [38](#)
- CAEN\_V1718
  - VME, [12](#)
- CAEN\_V2718
  - VME, [12](#)
- CLEAR\_KEEP\_TOKEN
  - VME::TDCV1x90Opcodes, [16](#)
- CLIENT
  - Socket communication objects, [9](#)
- COMPENSATION\_ENABLE
  - VME, [12](#)
- CONT\_STOR
  - VME::TDCV1x90Opcodes, [16](#)
- CONT\_STORAGE
  - VME, [12](#)
- CheckConfiguration
  - VME::BridgeVx718, [20](#)
  - VME::TDCV1x90, [64](#)
- Client, [21](#)
  - ~Client, [23](#)
  - Announce, [23](#)
  - Client, [23](#)
  - Connect, [24](#)
  - Disconnect, [24](#)
  - fClientId, [26](#)
  - flsConnected, [26](#)
  - GetType, [25](#)
  - ParseMessage, [25](#)
  - Receive, [25](#)
  - Send, [25](#)
  - SendAndReceive, [26](#)
- coarse
  - VME::glob\_offs, [31](#)
- Configure

- Socket, [45](#)
- Connect
  - Client, [24](#)
  - Messenger, [39](#)
- Control
  - VME, [13](#)
- Create
  - Socket, [46](#)
- ctl\_reg
  - VME, [12](#)
- DATA\_READY
  - VME, [14](#)
- DEFAULT\_SETUP\_REG
  - VME::TDCV1x90Opcodes, [16](#)
- DETECTOR
  - Socket communication objects, [9](#)
- DIS\_ALL\_CHANNEL
  - VME::TDCV1x90Opcodes, [16](#)
- DIS\_CHANNEL
  - VME::TDCV1x90Opcodes, [16](#)
- DIS\_ERROR\_BYPASS
  - VME::TDCV1x90Opcodes, [16](#)
- DIS\_ERROR\_MARK
  - VME::TDCV1x90Opcodes, [16](#)
- DIS\_HEAD\_TRAILER
  - VME::TDCV1x90Opcodes, [16](#)
- DIS\_SUB\_TRG
  - VME::TDCV1x90Opcodes, [16](#)
- DISABLE\_TEST\_MODE
  - VME::TDCV1x90Opcodes, [16](#)
- Decode
  - HTTPMessage, [33](#)
- Description
  - Exception, [28](#)
- det\_mode
  - VME, [12](#)
- detm
  - VME::TDCV1x90, [81](#)
- DisableChannel
  - VME::TDCV1x90, [65](#)
- Disconnect
  - Client, [24](#)
  - Messenger, [39](#)
- DisconnectClient
  - Messenger, [39](#)
- Dump
  - Exception, [28](#)
  - HTTPMessage, [33](#)
  - Message, [35](#)
  - SocketMessage, [53](#)
  - VME::TDCEvent, [57](#)
- DumpConnected
  - Socket, [46](#)
- EMPTY\_EVENT
  - VME, [12](#)
- EN\_ALL\_CHANNEL
  - VME::TDCV1x90Opcodes, [16](#)
- EN\_CHANNEL
  - VME::TDCV1x90Opcodes, [16](#)
- EN\_ERROR\_BYPASS
  - VME::TDCV1x90Opcodes, [16](#)
- EN\_ERROR\_MARK
  - VME::TDCV1x90Opcodes, [16](#)
- EN\_HEAD\_TRAILER
  - VME::TDCV1x90Opcodes, [16](#)
- EN\_SUB\_TRG
  - VME::TDCV1x90Opcodes, [16](#)
- ENABLE\_TEST\_MODE
  - VME::TDCV1x90Opcodes, [16](#)
- ERROR0
  - VME, [14](#)
- ERROR1
  - VME, [14](#)
- ERROR2
  - VME, [14](#)
- ERROR3
  - VME, [14](#)
- ETTT
  - VME::TDCEvent, [57](#)
- EVENT\_FIFO\_ENABLE
  - VME, [12](#)
- EXTENDED\_TRIGGER\_TIME\_TAG\_ENABLE
  - VME, [12](#)
- EXTRA\_SEARCH\_WIN\_WIDTH
  - VME, [14](#)
- EnableChannel
  - VME::TDCV1x90, [65](#)
- Encode
  - HTTPMessage, [33](#)
- ErrorNumber
  - Exception, [28](#)
- ettt
  - VME::trailead\_t, [81](#)
- event\_count
  - VME::trailead\_t, [81](#)
- EventCounter
  - VME, [13](#)
- EventFIFO
  - VME, [13](#)
- EventFIFOStatusRegister
  - VME, [13](#)
- EventFIFOStoredRegister
  - VME, [13](#)
- EventStored
  - VME, [13](#)
- EventType
  - VME::TDCEvent, [57](#)
- Exception, [26](#)
  - ~Exception, [27](#)
  - Description, [28](#)
  - Dump, [28](#)
  - ErrorNumber, [28](#)
  - Exception, [27](#)
  - fDescription, [28](#)
  - fErrorNumber, [28](#)



- fFrom, [28](#)
  - fType, [28](#)
  - From, [28](#)
  - Type, [28](#)
  - TypeString, [28](#)
- fAddress
  - Socket, [48](#)
- fBaseAddr
  - VME::TDCV1x90, [81](#)
- fBridge
  - VMEReader, [86](#)
- fBuffer
  - Socket, [48](#)
  - VME::TDCV1x90, [81](#)
- fClientId
  - Client, [26](#)
- fDescription
  - Exception, [28](#)
- fDetMode
  - VME::TDCV1x90, [81](#)
- fErrorNumber
  - Exception, [28](#)
- fFile
  - FileReader, [30](#)
- fFrom
  - Exception, [28](#)
- fHandle
  - VME::BridgeVx718, [21](#)
  - VME::TDCV1x90, [81](#)
- fHeader
  - FileReader, [30](#)
- flsConnected
  - Client, [26](#)
- fMaster
  - Socket, [48](#)
- fMessage
  - SocketMessage, [55](#)
- fNumAttempts
  - Messenger, [42](#)
- fOnSocket
  - VMEReader, [86](#)
- fOriginalString
  - HTTPMessage, [33](#)
- fPID
  - Messenger, [42](#)
- fPort
  - Socket, [48](#)
- fPortMapping
  - VME::BridgeVx718, [21](#)
- fReadFds
  - Socket, [48](#)
- fSocketId
  - Socket, [48](#)
- fSocketsConnected
  - Socket, [48](#)
- fString
  - Message, [35](#)
- fTDCCollection
  - VMEReader, [86](#)
- fType
  - Exception, [28](#)
- FULL
  - VME, [14](#)
- fVerb
  - VME::TDCV1x90, [81](#)
- fWS
  - HTTPMessage, [33](#)
  - Messenger, [42](#)
- fWord
  - VME::TDCEvent, [61](#)
- FetchEvents
  - VME::TDCV1x90, [65](#)
- FetchMessage
  - Socket, [46](#)
- file\_header\_t, [29](#)
  - magic, [29](#)
  - num\_hptdc, [29](#)
  - run\_id, [29](#)
  - spill\_id, [29](#)
- FileReader, [29](#)
  - ~FileReader, [30](#)
  - fFile, [30](#)
  - fHeader, [30](#)
  - FileReader, [30](#)
  - GetNextEvent, [30](#)
  - GetNumTDCs, [30](#)
- Filler
  - VME::TDCEvent, [57](#)
- fine
  - VME::glob\_offs, [31](#)
- FirmwareRev
  - VME, [13](#)
- From
  - Exception, [28](#)
- gEnd
  - VME::TDCV1x90, [81](#)
- GeoAddress
  - VME, [13](#)
- GetBLTEventNumberRegister
  - VME::TDCV1x90, [66](#)
- GetBunchId
  - VME::TDCEvent, [57](#)
- GetChannelId
  - VME::TDCEvent, [58](#)
- GetCtlRegister
  - VME::TDCV1x90, [66](#)
- GetETTT
  - VME::TDCEvent, [58](#)
  - VME::TDCV1x90, [66](#)
- GetErrorFlags
  - VME::TDCEvent, [58](#)
- GetEventCount
  - VME::TDCEvent, [58](#)
- GetEventCounter
  - VME::TDCV1x90, [67](#)
- GetEventId

- VME::TDCEvent, 59
- GetEventStored
  - VME::TDCV1x90, 67
- GetFirmwareRev
  - VME::TDCV1x90, 67
- GetGeo
  - VME::TDCEvent, 59
- GetHandle
  - VME::BridgeVx718, 20
- GetIntValue
  - SocketMessage, 53
- GetKey
  - HTTPMessage, 33
  - Message, 35
  - SocketMessage, 53
- GetLeadingTime
  - VME::TDCEvent, 59
- GetModel
  - VME::TDCV1x90, 68
- GetNextEvent
  - FileReader, 30
- GetNumTDCs
  - FileReader, 30
- GetOUI
  - VME::TDCV1x90, 68
- GetPort
  - Socket, 46
- GetRunNumber
  - VMEReader, 86
- GetSerialNumber
  - VME::TDCV1x90, 68
- GetSocketId
  - Socket, 46
- GetSocketType
  - Socket, 46
- GetStatus
  - VME::TDCEvent, 60
- GetStatusRegister
  - VME::TDCV1x90, 69
- GetString
  - Message, 35
  - SocketMessage, 54
- GetTDC
  - VMEReader, 86
- GetTDCEncapsulation
  - VME::TDCV1x90, 69
- GetTDCId
  - VME::TDCEvent, 60
- GetTrailingTime
  - VME::TDCEvent, 60
- GetType
  - Client, 25
  - Messenger, 40
  - VME::TDCEvent, 60
- GetValue
  - SocketMessage, 54
- GetVectorValue
  - SocketMessage, 54
- GetWidth
  - VME::TDCEvent, 61
- GetWordCount
  - VME::TDCEvent, 61
- GlobalHeader
  - VME::TDCEvent, 57
- GlobalTrailer
  - VME::TDCEvent, 57
- HEADER\_EN
  - VME, 14
- HTTPMessage, 31
  - Decode, 33
  - Dump, 33
  - Encode, 33
  - fOriginalString, 33
  - fWS, 33
  - GetKey, 33
  - HTTPMessage, 32, 33
- HardwareReset
  - VME::TDCV1x90, 69
- INVALID
  - Socket communication objects, 9
- InputConf
  - VME::BridgeVx718, 20
- InputRead
  - VME::BridgeVx718, 20
- InterruptLevel
  - VME, 13
- InterruptVector
  - VME, 13
- IsFromWeb
  - Message, 35
- IsTrailing
  - VME::TDCEvent, 61
- IsTriggerMatching
  - VME::TDCV1x90, 69
- IsWebSocket
  - Socket, 46
- kSoftwareClear
  - VME, 13
- LOAD\_DEF\_CONFIG
  - VME::TDCV1x90Opcodes, 16
- LOAD\_USER\_CONFIG
  - VME::TDCV1x90Opcodes, 17
- leading
  - VME::trailead\_t, 82
- Listen
  - Socket, 46
- MASTER
  - Socket communication objects, 9
- MATCH\_WIN\_WIDTH
  - VME, 14
- MCSTBase
  - VME, 13

MCSTControl  
     VME, 13  
 magic  
     file\_header\_t, 29  
 Message, 34  
     ~Message, 35  
     Dump, 35  
     fString, 35  
     GetKey, 35  
     GetString, 35  
     IsFromWeb, 35  
     Message, 35  
 Messenger, 36  
     ~Messenger, 38  
     AddClient, 38  
     Broadcast, 38  
     Connect, 39  
     Disconnect, 39  
     DisconnectClient, 39  
     fNumAttempts, 42  
     fPID, 42  
     fWS, 42  
     GetType, 40  
     Messenger, 37  
     ProcessMessage, 40  
     Receive, 40  
     Send, 41  
     StartAcquisition, 42  
     StopAcquisition, 42  
     SwitchClientType, 42  
 Micro  
     VME, 13  
 micro\_handshake  
     VME, 13  
 MicroHandshake  
     VME, 13  
 mod\_reg  
     VME, 13  
 ModuleReset  
     VME, 13  
  
 nchannels  
     VME::TDCV1x90, 81  
 num\_hptdc  
     file\_header\_t, 29  
  
 OLEADING  
     VME, 13  
 OTRAILING  
     VME, 13  
 Object  
     SocketMessage, 54  
 outBufTDCErr  
     VME::TDCV1x90, 81  
 outBufTDCHeadTrail  
     VME::TDCV1x90, 81  
 outBufTDCTTT  
     VME::TDCV1x90, 81  
 OutputConf  
     VME::BridgeVx718, 21  
 OutputOff  
     VME::BridgeVx718, 21  
 OutputOn  
     VME::BridgeVx718, 21  
  
 PAIR  
     VME, 13  
 PAIRED  
     VME, 14  
 PURG  
     VME, 14  
 pair\_lead\_res  
     VME::TDCV1x90, 81  
 pair\_width\_res  
     VME::TDCV1x90, 81  
 ParseMessage  
     Client, 25  
 PrepareConnection  
     Socket, 47  
 ProcessMessage  
     Messenger, 40  
  
 r100ps  
     VME, 14  
 r200ps  
     VME, 14  
 r25ps  
     VME, 14  
 r800ps  
     VME, 14  
 READ\_ACQ\_MOD  
     VME::TDCV1x90Opcodes, 17  
 READ\_ADJUST\_CH  
     VME::TDCV1x90Opcodes, 17  
 READ\_COMPENSATION\_SRAM\_ENABLE  
     VME, 12  
 READ\_DEAD\_TIME  
     VME::TDCV1x90Opcodes, 17  
 READ\_DETECTION  
     VME::TDCV1x90Opcodes, 17  
 READ\_DLL\_LOCK  
     VME::TDCV1x90Opcodes, 17  
 READ\_EEPROM  
     VME::TDCV1x90Opcodes, 17  
 READ\_EN\_PATTERN  
     VME::TDCV1x90Opcodes, 17  
 READ\_EN\_PATTERN32  
     VME::TDCV1x90Opcodes, 17  
 READ\_ERROR\_STATUS  
     VME::TDCV1x90Opcodes, 17  
 READ\_ERROR\_TYPES  
     VME::TDCV1x90Opcodes, 17  
 READ\_EVENT\_SIZE  
     VME::TDCV1x90Opcodes, 17  
 READ\_FIFO\_SIZE  
     VME::TDCV1x90Opcodes, 17  
 READ\_GLOB\_OFFS  
     VME::TDCV1x90Opcodes, 17

READ\_HEAD\_TRAILER  
     VME::TDCV1x90Opcodes, 17  
 READ\_MICRO\_REV  
     VME::TDCV1x90Opcodes, 17  
 READ\_OK  
     VME, 13  
 READ\_RC\_ADJ  
     VME::TDCV1x90Opcodes, 17  
 READ\_RES  
     VME::TDCV1x90Opcodes, 17  
 READ\_SETUP\_REG  
     VME::TDCV1x90Opcodes, 17  
 READ\_SETUP\_SCANPATH  
     VME::TDCV1x90Opcodes, 17  
 READ\_SPARE  
     VME::TDCV1x90Opcodes, 17  
 READ\_STATUS\_STREAM  
     VME::TDCV1x90Opcodes, 17  
 READ\_TDC\_ID  
     VME::TDCV1x90Opcodes, 17  
 READ\_TRG\_CONF  
     VME::TDCV1x90Opcodes, 17  
 REJECT\_MARGIN  
     VME, 14  
 RES\_1  
     VME, 14  
 RES\_2  
     VME, 14  
 RESET\_DLL\_PLL  
     VME::TDCV1x90Opcodes, 17  
 REV\_DATE\_MICRO\_FW  
     VME::TDCV1x90Opcodes, 17  
 ROMBoard0  
     VME, 13  
 ROMBoard1  
     VME, 13  
 ROMBoard2  
     VME, 13  
 ROMUi0  
     VME, 13  
 ROMUi1  
     VME, 13  
 ROMUi2  
     VME, 13  
 ROMRevis0  
     VME, 13  
 ROMRevis1  
     VME, 13  
 ROMRevis2  
     VME, 13  
 ROMRevis3  
     VME, 13  
 ROMSerNum0  
     VME, 13  
 ROMSerNum1  
     VME, 13  
 ReadDetection  
     VME::TDCV1x90, 70  
 ReadFIFOSize  
     VME::TDCV1x90, 70  
 ReadGlobalOffset  
     VME::TDCV1x90, 71  
 ReadRCAdjust  
     VME::TDCV1x90, 71  
 ReadRegister  
     VME::TDCV1x90, 72  
 ReadResolution  
     VME::TDCV1x90, 72  
 ReadTrigConf  
     VME::TDCV1x90, 73  
 Receive  
     Client, 25  
     Messenger, 40  
 run\_id  
     file\_header\_t, 29  
 SAVE\_RC\_ADJ  
     VME::TDCV1x90Opcodes, 17  
 SAVE\_USER\_CONFIG  
     VME::TDCV1x90Opcodes, 17  
 SET\_ADJUST\_CH  
     VME::TDCV1x90Opcodes, 18  
 SET\_DEAD\_TIME  
     VME::TDCV1x90Opcodes, 18  
 SET\_DETECTION  
     VME::TDCV1x90Opcodes, 18  
 SET\_DLL\_CLOCK  
     VME::TDCV1x90Opcodes, 18  
 SET\_ERROR\_TYPES  
     VME::TDCV1x90Opcodes, 18  
 SET\_EVENT\_SIZE  
     VME::TDCV1x90Opcodes, 18  
 SET\_FIFO\_SIZE  
     VME::TDCV1x90Opcodes, 18  
 SET\_GLOB\_OFFS  
     VME::TDCV1x90Opcodes, 18  
 SET\_KEEP\_TOKEN  
     VME::TDCV1x90Opcodes, 18  
 SET\_PAIR\_RES  
     VME::TDCV1x90Opcodes, 18  
 SET\_RC\_ADJ  
     VME::TDCV1x90Opcodes, 18  
 SET\_REJ\_MARGIN  
     VME::TDCV1x90Opcodes, 18  
 SET\_SW\_MARGIN  
     VME::TDCV1x90Opcodes, 18  
 SET\_TDC\_TSET\_OUTPUT  
     VME::TDCV1x90Opcodes, 18  
 SET\_TR\_LEAD\_LSB  
     VME::TDCV1x90Opcodes, 18  
 SET\_WIN\_OFFS  
     VME::TDCV1x90Opcodes, 18  
 SET\_WIN\_WIDTH  
     VME::TDCV1x90Opcodes, 18  
 SelectConnections  
     Socket, 47  
 Send

- Client, [25](#)
- Messenger, [41](#)
- SendAndReceive
  - Client, [26](#)
- SendMessage
  - Socket, [47](#)
- SetAcquisitionMode
  - VME::TDCV1x90, [73](#)
- SetBLTEventNumberRegister
  - VME::TDCV1x90, [73](#)
- SetContinuousStorage
  - VME::TDCV1x90, [74](#)
- SetCtlRegister
  - VME::TDCV1x90, [74](#)
- SetDetection
  - VME::TDCV1x90, [74](#)
- SetETTT
  - VME::TDCV1x90, [75](#)
- SetFIFOSize
  - VME::TDCV1x90, [75](#)
- SetGlobalOffset
  - VME::TDCV1x90, [75](#)
- SetKeyValue
  - SocketMessage, [54](#), [55](#)
- SetLSBTraileadEdge
  - VME::TDCV1x90, [76](#)
- SetPairModeResolution
  - VME::TDCV1x90, [76](#)
- SetPol
  - VME::TDCV1x90, [77](#)
- SetPort
  - Socket, [47](#)
- SetRCAdjust
  - VME::TDCV1x90, [77](#)
- SetSocketId
  - Socket, [47](#)
- SetStatusRegister
  - VME::TDCV1x90, [77](#)
- SetTDCEncapsulation
  - VME::TDCV1x90, [77](#)
- SetTDCErrorMarks
  - VME::TDCV1x90, [77](#)
- SetTriggerMatching
  - VME::TDCV1x90, [78](#)
- SetVerboseLevel
  - VME::TDCV1x90, [78](#)
- SetWindowOffset
  - VME::TDCV1x90, [78](#)
- SetWindowWidth
  - VME::TDCV1x90, [79](#)
- SetWord
  - VME::TDCEvent, [61](#)
- Socket, [43](#)
  - ~Socket, [45](#)
  - AcceptConnections, [45](#)
  - Bind, [45](#)
  - Configure, [45](#)
  - Create, [46](#)
  - DumpConnected, [46](#)
  - fAddress, [48](#)
  - fBuffer, [48](#)
  - fMaster, [48](#)
  - fPort, [48](#)
  - fReadFds, [48](#)
  - fSocketId, [48](#)
  - fSocketsConnected, [48](#)
  - FetchMessage, [46](#)
  - GetPort, [46](#)
  - GetSocketId, [46](#)
  - GetSocketType, [46](#)
  - IsWebSocket, [46](#)
  - Listen, [46](#)
  - PrepareConnection, [47](#)
  - SelectConnections, [47](#)
  - SendMessage, [47](#)
  - SetPort, [47](#)
  - SetSocketId, [47](#)
  - Socket, [45](#)
  - SocketCollection, [45](#)
  - Start, [47](#)
  - Stop, [48](#)
- Socket communication objects, [9](#)
  - CLIENT, [9](#)
  - DETECTOR, [9](#)
  - INVALID, [9](#)
  - MASTER, [9](#)
  - SocketType, [9](#)
  - WEBSOCKET\_CLIENT, [9](#)
- SocketCollection
  - Socket, [45](#)
- SocketMessage, [49](#)
  - ~SocketMessage, [53](#)
  - Dump, [53](#)
  - fMessage, [55](#)
  - GetIntValue, [53](#)
  - GetKey, [53](#)
  - GetString, [54](#)
  - GetValue, [54](#)
  - GetVectorValue, [54](#)
  - Object, [54](#)
  - SetKeyValue, [54](#), [55](#)
  - SocketMessage, [51–53](#)
  - String, [55](#)
- SocketType
  - Socket communication objects, [9](#)
- SoftwareClear
  - VME::TDCV1x90, [79](#)
- SoftwareReset
  - VME::TDCV1x90, [79](#)
- spill\_id
  - file\_header\_t, [29](#)
- Start
  - Socket, [47](#)
- StartAcquisition
  - Messenger, [42](#)
- stat\_reg

- VME, 13
- Status
  - VME, 13
- Stop
  - Socket, 48
- StopAcquisition
  - Messenger, 42
- String
  - SocketMessage, 55
- SwitchClientType
  - Messenger, 42
- TDCCollection
  - VMEReaders, 84
- TDCError
  - VME::TDCEvent, 57
- TDCEvent
  - VME::TDCEvent, 57
- TDCEventCollection
  - VME, 12
- TDCHeader
  - VME::TDCEvent, 57
- TDCMeasurement
  - VME::TDCEvent, 57
- TDCTrailer
  - VME::TDCEvent, 57
- TDCV1x90
  - VME::TDCV1x90, 64
- TERM
  - VME, 12
- TERM\_ON
  - VME, 14
- TERM\_SW
  - VME, 12
- TEST\_FIFO\_ENABLE
  - VME, 12
- TRAILEAD
  - VME, 13
- TRG\_MATCH
  - VME, 14
  - VME::TDCV1x90Opcodes, 18
- TRIG\_MATCH
  - VME, 12
- TRIG\_TIME\_SUB
  - VME, 14
- TRIGGER\_LOST
  - VME, 14
- total\_hits
  - VME::trailead\_t, 82
- trailead\_edge\_lsb
  - VME, 14
- trailead\_edge\_res
  - VME::TDCV1x90, 81
- trailing
  - VME::trailead\_t, 82
- trig\_conf
  - VME, 14
- Type
  - Exception, 28
- TypeString
  - Exception, 28
- UPDATE\_SETUP\_REG
  - VME::TDCV1x90Opcodes, 18
- UPDATE\_SETUP\_TDC
  - VME::TDCV1x90Opcodes, 18
- VME, 11
  - ALIGN64, 12
  - ALM\_FULL, 14
  - acq\_mode, 12
  - BERR\_FLAG, 14
  - BERREN, 12
  - BLTEventNumber, 13
  - BridgeType, 12
  - CAEN\_V1718, 12
  - CAEN\_V2718, 12
  - COMPENSATION\_ENABLE, 12
  - CONT\_STORAGE, 12
  - Control, 13
  - ctl\_reg, 12
  - DATA\_READY, 14
  - det\_mode, 12
  - EMPTY\_EVENT, 12
  - ERROR0, 14
  - ERROR1, 14
  - ERROR2, 14
  - ERROR3, 14
  - EVENT\_FIFO\_ENABLE, 12
  - EXTENDED\_TRIGGER\_TIME\_TAG\_ENABLE, 12
  - EXTRA\_SEARCH\_WIN\_WIDTH, 14
  - EventCounter, 13
  - EventFIFO, 13
  - EventFIFOStatusRegister, 13
  - EventFIFOStoredRegister, 13
  - EventStored, 13
  - FULL, 14
  - FirmwareRev, 13
  - GeoAddress, 13
  - HEADER\_EN, 14
  - InterruptLevel, 13
  - InterruptVector, 13
  - kSoftwareClear, 13
  - MATCH\_WIN\_WIDTH, 14
  - MCSTBase, 13
  - MCSTControl, 13
  - Micro, 13
  - micro\_handshake, 13
  - MicroHandshake, 13
  - mod\_reg, 13
  - ModuleReset, 13
  - OLEADING, 13
  - OTRILING, 13
  - PAIR, 13
  - PAIRED, 14
  - PURG, 14
  - r100ps, 14
  - r200ps, 14

- r25ps, [14](#)
- r800ps, [14](#)
- READ\_COMPENSATION\_SRAM\_ENABLE, [12](#)
- READ\_OK, [13](#)
- REJECT\_MARGIN, [14](#)
- RES\_1, [14](#)
- RES\_2, [14](#)
- ROMBoard0, [13](#)
- ROMBoard1, [13](#)
- ROMBoard2, [13](#)
- ROMOui0, [13](#)
- ROMOui1, [13](#)
- ROMOui2, [13](#)
- ROMRevis0, [13](#)
- ROMRevis1, [13](#)
- ROMRevis2, [13](#)
- ROMRevis3, [13](#)
- ROMSerNum0, [13](#)
- ROMSerNum1, [13](#)
- stat\_reg, [13](#)
- Status, [13](#)
- TDCEventCollection, [12](#)
- TERM, [12](#)
- TERM\_ON, [14](#)
- TERM\_SW, [12](#)
- TEST\_FIFO\_ENABLE, [12](#)
- TRAILEAD, [13](#)
- TRG\_MATCH, [14](#)
- TRIG\_MATCH, [12](#)
- TRIG\_TIME\_SUB, [14](#)
- TRIGGER\_LOST, [14](#)
- trailead\_edge\_lsb, [14](#)
- trig\_conf, [14](#)
- WIN\_OFFSET, [14](#)
- WRITE\_OK, [13](#)
- VME::BridgeVx718, [19](#)
  - ~BridgeVx718, [20](#)
  - BridgeVx718, [20](#)
  - CheckConfiguration, [20](#)
  - fHandle, [21](#)
  - fPortMapping, [21](#)
  - GetHandle, [20](#)
  - InputConf, [20](#)
  - InputRead, [20](#)
  - OutputConf, [21](#)
  - OutputOff, [21](#)
  - OutputOn, [21](#)
- VME::TDCEvent, [55](#)
  - ~TDCEvent, [57](#)
  - Dump, [57](#)
  - ETTT, [57](#)
  - EventType, [57](#)
  - fWord, [61](#)
  - Filler, [57](#)
  - GetBunchId, [57](#)
  - GetChannelId, [58](#)
  - GetETTT, [58](#)
  - GetErrorFlags, [58](#)
  - GetEventCount, [58](#)
  - GetEventId, [59](#)
  - GetGeo, [59](#)
  - GetLeadingTime, [59](#)
  - GetStatus, [60](#)
  - GetTDCId, [60](#)
  - GetTrailingTime, [60](#)
  - GetType, [60](#)
  - GetWidth, [61](#)
  - GetWordCount, [61](#)
  - GlobalHeader, [57](#)
  - GlobalTrailer, [57](#)
  - IsTrailing, [61](#)
  - SetWord, [61](#)
  - TDCErr, [57](#)
  - TDCEvent, [57](#)
  - TDCHeader, [57](#)
  - TDCMeasurement, [57](#)
  - TDCTrailer, [57](#)
- VME::TDCV1x90, [62](#)
  - ~TDCV1x90, [64](#)
  - abort, [64](#)
  - acqm, [80](#)
  - am, [80](#)
  - am\_blt, [81](#)
  - CheckConfiguration, [64](#)
  - detm, [81](#)
  - DisableChannel, [65](#)
  - EnableChannel, [65](#)
  - fBaseAddr, [81](#)
  - fBuffer, [81](#)
  - fDetMode, [81](#)
  - fHandle, [81](#)
  - fVerb, [81](#)
  - FetchEvents, [65](#)
  - gEnd, [81](#)
  - GetBLTEventNumberRegister, [66](#)
  - GetCtlRegister, [66](#)
  - GetETTT, [66](#)
  - GetEventCounter, [67](#)
  - GetEventStored, [67](#)
  - GetFirmwareRev, [67](#)
  - GetModel, [68](#)
  - GetOUI, [68](#)
  - GetSerialNumber, [68](#)
  - GetStatusRegister, [69](#)
  - GetTDCEncapsulation, [69](#)
  - HardwareReset, [69](#)
  - IsTriggerMatching, [69](#)
  - nchannels, [81](#)
  - outBufTDCErr, [81](#)
  - outBufTDCHeadTrail, [81](#)
  - outBufTDCETTT, [81](#)
  - pair\_lead\_res, [81](#)
  - pair\_width\_res, [81](#)
  - ReadDetection, [70](#)
  - ReadFIFOSize, [70](#)
  - ReadGlobalOffset, [71](#)

- ReadRCAdjust, 71
- ReadRegister, 72
- ReadResolution, 72
- ReadTrigConf, 73
- SetAcquisitionMode, 73
- SetBLTEventNumberRegister, 73
- SetContinuousStorage, 74
- SetCtlRegister, 74
- SetDetection, 74
- SetETTT, 75
- SetFIFOSize, 75
- SetGlobalOffset, 75
- SetLSBTrailEdge, 76
- SetPairModeResolution, 76
- SetPol, 77
- SetRCAdjust, 77
- SetStatusRegister, 77
- SetTDCEncapsulation, 77
- SetTDCErrorMarks, 77
- SetTriggerMatching, 78
- SetVerboseLevel, 78
- SetWindowOffset, 78
- SetWindowWidth, 79
- SoftwareClear, 79
- SoftwareReset, 79
- TDCV1x90, 64
- trailEdge\_res, 81
- WaitMicro, 80
- WriteRegister, 80
- VME::TDCV1x90Opcodes, 14
  - AUTOLOAD\_DEF\_CONFI, 16
  - AUTOLOAD\_USER\_CONF, 16
  - CLEAR\_KEEP\_TOKEN, 16
  - CONT\_STOR, 16
  - DEFAULT\_SETUP\_REG, 16
  - DIS\_ALL\_CHANNEL, 16
  - DIS\_CHANNEL, 16
  - DIS\_ERROR\_BYPASS, 16
  - DIS\_ERROR\_MARK, 16
  - DIS\_HEAD\_TRAILER, 16
  - DIS\_SUB\_TRG, 16
  - DISABLE\_TEST\_MODE, 16
  - EN\_ALL\_CHANNEL, 16
  - EN\_CHANNEL, 16
  - EN\_ERROR\_BYPASS, 16
  - EN\_ERROR\_MARK, 16
  - EN\_HEAD\_TRAILER, 16
  - EN\_SUB\_TRG, 16
  - ENABLE\_TEST\_MODE, 16
  - LOAD\_DEF\_CONFIG, 16
  - LOAD\_USER\_CONFIG, 17
  - READ\_ACQ\_MOD, 17
  - READ\_ADJUST\_CH, 17
  - READ\_DEAD\_TIME, 17
  - READ\_DETECTION, 17
  - READ\_DLL\_LOCK, 17
  - READ\_EEPROM, 17
  - READ\_EN\_PATTERN, 17
  - READ\_EN\_PATTERN32, 17
  - READ\_ERROR\_STATUS, 17
  - READ\_ERROR\_TYPES, 17
  - READ\_EVENT\_SIZE, 17
  - READ\_FIFO\_SIZE, 17
  - READ\_GLOB\_OFFS, 17
  - READ\_HEAD\_TRAILER, 17
  - READ\_MICRO\_REV, 17
  - READ\_RC\_ADJ, 17
  - READ\_RES, 17
  - READ\_SETUP\_REG, 17
  - READ\_SETUP\_SCANPATH, 17
  - READ\_SPARE, 17
  - READ\_STATUS\_STREAM, 17
  - READ\_TDC\_ID, 17
  - READ\_TRG\_CONF, 17
  - RESET\_DLL\_PLL, 17
  - REV\_DATE\_MICRO\_FW, 17
  - SAVE\_RC\_ADJ, 17
  - SAVE\_USER\_CONFIG, 17
  - SET\_ADJUST\_CH, 18
  - SET\_DEAD\_TIME, 18
  - SET\_DETECTION, 18
  - SET\_DLL\_CLOCK, 18
  - SET\_ERROR\_TYPES, 18
  - SET\_EVENT\_SIZE, 18
  - SET\_FIFO\_SIZE, 18
  - SET\_GLOB\_OFFS, 18
  - SET\_KEEP\_TOKEN, 18
  - SET\_PAIR\_RES, 18
  - SET\_RC\_ADJ, 18
  - SET\_REJ\_MARGIN, 18
  - SET\_SW\_MARGIN, 18
  - SET\_TDC\_TSET\_OUTPUT, 18
  - SET\_TR\_LEAD\_LSB, 18
  - SET\_WIN\_OFFS, 18
  - SET\_WIN\_WIDTH, 18
  - TRG\_MATCH, 18
  - UPDATE\_SETUP\_REG, 18
  - UPDATE\_SETUP\_TDC, 18
  - WRITE\_EEPROM, 18
  - WRITE\_EN\_PATTERN, 18
  - WRITE\_EN\_PATTERN32, 18
  - WRITE\_SETUP\_REG, 18
  - WRITE\_SPARE, 18
- VME::glob\_offs, 31
  - coarse, 31
  - fine, 31
- VME::trailEdge\_t, 81
  - ettt, 81
  - event\_count, 81
  - leading, 82
  - total\_hits, 82
  - trailing, 82
- VMEReader, 82
  - ~VMEReader, 84
  - Abort, 84
  - AddTDC, 84



- fBridge, [86](#)
- fOnSocket, [86](#)
- fTDCCollection, [86](#)
- GetRunNumber, [86](#)
- GetTDC, [86](#)
- TDCCollection, [84](#)
- VMEReader, [84](#)
- WEBSOCKET\_CLIENT
  - Socket communication objects, [9](#)
- WIN\_OFFSET
  - VME, [14](#)
- WRITE\_EEPROM
  - VME::TDCV1x90Opcodes, [18](#)
- WRITE\_EN\_PATTERN
  - VME::TDCV1x90Opcodes, [18](#)
- WRITE\_EN\_PATTERN32
  - VME::TDCV1x90Opcodes, [18](#)
- WRITE\_OK
  - VME, [13](#)
- WRITE\_SETUP\_REG
  - VME::TDCV1x90Opcodes, [18](#)
- WRITE\_SPARE
  - VME::TDCV1x90Opcodes, [18](#)
- WaitMicro
  - VME::TDCV1x90, [80](#)
- WriteRegister
  - VME::TDCV1x90, [80](#)