

## 2015 Test beam Run Control

Generated by Doxygen 1.8.9.1

Tue Jun 23 2015 21:30:17



# 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	AcquisitionMode	12
6.1.2.2	BridgeType	12
6.1.2.3	DetectionMode	12
6.1.2.4	micro_handshake	12
6.1.2.5	trailead_edge_lsb	13
6.1.2.6	trig_conf	13
6.2	VME::TDCV1x90Opcodes Namespace Reference	13
6.2.1	Function Documentation	14
6.2.1.1	AUTOLOAD_DEF_CONF1	14
6.2.1.2	AUTOLOAD_USER_CONF	14

6.2.1.3	CLEAR_KEEP_TOKEN . . . . .	14
6.2.1.4	CONT_STOR . . . . .	15
6.2.1.5	DEFAULT_SETUP_REG . . . . .	15
6.2.1.6	DIS_ALL_CHANNEL . . . . .	15
6.2.1.7	DIS_CHANNEL . . . . .	15
6.2.1.8	DIS_ERROR_BYPASS . . . . .	15
6.2.1.9	DIS_ERROR_MARK . . . . .	15
6.2.1.10	DIS_HEAD_TRAILER . . . . .	15
6.2.1.11	DIS_SUB_TRG . . . . .	15
6.2.1.12	DISABLE_TEST_MODE . . . . .	15
6.2.1.13	EN_ALL_CHANNEL . . . . .	15
6.2.1.14	EN_CHANNEL . . . . .	15
6.2.1.15	EN_ERROR_BYPASS . . . . .	15
6.2.1.16	EN_ERROR_MARK . . . . .	15
6.2.1.17	EN_HEAD_TRAILER . . . . .	15
6.2.1.18	EN_SUB_TRG . . . . .	15
6.2.1.19	ENABLE_TEST_MODE . . . . .	15
6.2.1.20	LOAD_DEF_CONFIG . . . . .	15
6.2.1.21	LOAD_USER_CONFIG . . . . .	15
6.2.1.22	READ_ACQ_MOD . . . . .	15
6.2.1.23	READ_ADJUST_CH . . . . .	15
6.2.1.24	READ_DEAD_TIME . . . . .	15
6.2.1.25	READ_DETECTION . . . . .	15
6.2.1.26	READ_DLL_LOCK . . . . .	15
6.2.1.27	READ_EEPROM . . . . .	15
6.2.1.28	READ_EN_PATTERN . . . . .	15
6.2.1.29	READ_EN_PATTERN32 . . . . .	15
6.2.1.30	READ_ERROR_STATUS . . . . .	15
6.2.1.31	READ_ERROR_TYPES . . . . .	15
6.2.1.32	READ_EVENT_SIZE . . . . .	16
6.2.1.33	READ_FIFO_SIZE . . . . .	16
6.2.1.34	READ_GLOB_OFFS . . . . .	16
6.2.1.35	READ_HEAD_TRAILER . . . . .	16
6.2.1.36	READ_MICRO_REV . . . . .	16
6.2.1.37	READ_RC_ADJ . . . . .	16
6.2.1.38	READ_RES . . . . .	16
6.2.1.39	READ_SETUP_REG . . . . .	16
6.2.1.40	READ_SETUP_SCANPATH . . . . .	16
6.2.1.41	READ_SPARE . . . . .	16
6.2.1.42	READ_STATUS_STREAM . . . . .	16

6.2.1.43	READ_TDC_ID	16
6.2.1.44	READ_TRG_CONF	16
6.2.1.45	RESET_DLL_PLL	16
6.2.1.46	REV_DATE_MICRO_FW	16
6.2.1.47	SAVE_RC_ADJ	16
6.2.1.48	SAVE_USER_CONFIG	16
6.2.1.49	SET_ADJUST_CH	16
6.2.1.50	SET_DEAD_TIME	16
6.2.1.51	SET_DETECTION	16
6.2.1.52	SET_DLL_CLOCK	16
6.2.1.53	SET_ERROR_TYPES	16
6.2.1.54	SET_EVENT_SIZE	16
6.2.1.55	SET_FIFO_SIZE	16
6.2.1.56	SET_GLOB_OFFS	16
6.2.1.57	SET_KEEP_TOKEN	16
6.2.1.58	SET_PAIR_RES	16
6.2.1.59	SET_RC_ADJ	16
6.2.1.60	SET_REJ_MARGIN	17
6.2.1.61	SET_SW_MARGIN	17
6.2.1.62	SET_TDC_TSET_OUTPUT	17
6.2.1.63	SET_TR_LEAD_LSB	17
6.2.1.64	SET_WIN_OFFS	17
6.2.1.65	SET_WIN_WIDTH	17
6.2.1.66	TRG_MATCH	17
6.2.1.67	UPDATE_SETUP_REG	17
6.2.1.68	UPDATE_SETUP_TDC	17
6.2.1.69	WRITE_EEPROM	17
6.2.1.70	WRITE_EN_PATTERN	17
6.2.1.71	WRITE_EN_PATTERN32	17
6.2.1.72	WRITE_SETUP_REG	17
6.2.1.73	WRITE_SPARE	17
<b>7</b>	<b>Data Structure Documentation</b>	<b>19</b>
7.1	VME::BridgeVx718 Class Reference	19
7.1.1	Detailed Description	20
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	21
7.1.3.4	InputRead	21
7.1.3.5	OutputConf	21
7.1.3.6	OutputOff	21
7.1.3.7	OutputOn	21
7.1.3.8	ReadRegister	21
7.1.3.9	ReadRegister	21
7.1.3.10	StartPulser	21
7.1.3.11	StopPulser	21
7.1.3.12	WriteRegister	21
7.1.3.13	WriteRegister	21
7.1.4	Field Documentation	21
7.1.4.1	fBaseAddr	21
7.1.4.2	fHandle	21
7.2	VME::BridgeVx718Control Class Reference	22
7.2.1	Constructor & Destructor Documentation	22
7.2.1.1	BridgeVx718Control	22
7.2.1.2	~BridgeVx718Control	22
7.2.2	Member Function Documentation	22
7.2.2.1	GetAddressIncrement	22
7.2.2.2	GetArbiterType	22
7.2.2.3	GetBusReqLevel	23
7.2.2.4	GetBusTimeout	23
7.2.2.5	GetInterruptReq	23
7.2.2.6	GetReleaseType	23
7.2.2.7	GetRequesterType	23
7.2.2.8	GetSysRes	23
7.2.3	Field Documentation	23
7.2.3.1	fWord	23
7.3	VME::BridgeVx718Status Class Reference	23
7.3.1	Constructor & Destructor Documentation	24
7.3.1.1	BridgeVx718Status	24
7.3.1.2	~BridgeVx718Status	24
7.3.2	Member Function Documentation	24
7.3.2.1	Dump	24
7.3.2.2	GetBERR	24
7.3.2.3	GetDipSwitch	24
7.3.2.4	GetDTACK	24
7.3.2.5	GetSystemControl	24

7.3.2.6	GetSystemReset	24
7.3.2.7	GetUSBType	24
7.3.3	Field Documentation	24
7.3.3.1	fWord	24
7.4	Client Class Reference	24
7.4.1	Detailed Description	26
7.4.2	Constructor & Destructor Documentation	26
7.4.2.1	Client	26
7.4.2.2	Client	26
7.4.2.3	~Client	27
7.4.3	Member Function Documentation	27
7.4.3.1	Announce	27
7.4.3.2	Connect	27
7.4.3.3	Disconnect	28
7.4.3.4	GetType	28
7.4.3.5	ParseMessage	28
7.4.3.6	Receive	29
7.4.3.7	Send	29
7.4.3.8	SendAndReceive	30
7.4.4	Field Documentation	30
7.4.4.1	fClientId	30
7.4.4.2	fIsConnected	30
7.5	Exception Class Reference	30
7.5.1	Detailed Description	31
7.5.2	Constructor & Destructor Documentation	31
7.5.2.1	Exception	31
7.5.2.2	Exception	31
7.5.2.3	~Exception	31
7.5.3	Member Function Documentation	31
7.5.3.1	Description	31
7.5.3.2	Dump	32
7.5.3.3	ErrorNumber	32
7.5.3.4	From	32
7.5.3.5	Type	32
7.5.3.6	TypeString	32
7.5.4	Field Documentation	32
7.5.4.1	fDescription	32
7.5.4.2	fErrorNumber	32
7.5.4.3	fFrom	32
7.5.4.4	fType	32

7.6	file_header_t Struct Reference	33
7.6.1	Detailed Description	33
7.6.2	Field Documentation	33
7.6.2.1	acq_mode	33
7.6.2.2	det_mode	33
7.6.2.3	magic	33
7.6.2.4	num_hptdc	33
7.6.2.5	run_id	33
7.6.2.6	spill_id	33
7.7	FileReader Class Reference	33
7.7.1	Detailed Description	34
7.7.2	Constructor & Destructor Documentation	34
7.7.2.1	FileReader	34
7.7.2.2	~FileReader	35
7.7.3	Member Function Documentation	35
7.7.3.1	GetNextEvent	35
7.7.3.2	GetNextMeasurement	35
7.7.3.3	GetNumTDCs	36
7.7.4	Field Documentation	36
7.7.4.1	fFile	36
7.7.4.2	fHeader	36
7.7.4.3	fReadoutMode	36
7.8	VME::GlobalOffset Struct Reference	36
7.8.1	Field Documentation	36
7.8.1.1	coarse	36
7.8.1.2	fine	36
7.9	HTTPMessage Class Reference	36
7.9.1	Detailed Description	37
7.9.2	Constructor & Destructor Documentation	38
7.9.2.1	HTTPMessage	38
7.9.2.2	HTTPMessage	38
7.9.3	Member Function Documentation	38
7.9.3.1	Decode	38
7.9.3.2	Dump	38
7.9.3.3	Encode	38
7.9.3.4	GetKey	39
7.9.4	Field Documentation	39
7.9.4.1	fOriginalString	39
7.9.4.2	fWS	39
7.10	Message Class Reference	39



7.10.1 Detailed Description . . . . .	40
7.10.2 Constructor & Destructor Documentation . . . . .	40
7.10.2.1 Message . . . . .	40
7.10.2.2 Message . . . . .	40
7.10.2.3 Message . . . . .	40
7.10.2.4 ~Message . . . . .	40
7.10.3 Member Function Documentation . . . . .	40
7.10.3.1 Dump . . . . .	40
7.10.3.2 GetKey . . . . .	40
7.10.3.3 GetString . . . . .	40
7.10.3.4 IsFromWeb . . . . .	40
7.10.4 Field Documentation . . . . .	40
7.10.4.1 fString . . . . .	40
7.11 Messenger Class Reference . . . . .	41
7.11.1 Detailed Description . . . . .	42
7.11.2 Constructor & Destructor Documentation . . . . .	42
7.11.2.1 Messenger . . . . .	42
7.11.2.2 Messenger . . . . .	42
7.11.2.3 ~Messenger . . . . .	43
7.11.3 Member Function Documentation . . . . .	43
7.11.3.1 AddClient . . . . .	43
7.11.3.2 Broadcast . . . . .	44
7.11.3.3 Connect . . . . .	44
7.11.3.4 Disconnect . . . . .	44
7.11.3.5 DisconnectClient . . . . .	45
7.11.3.6 GetType . . . . .	45
7.11.3.7 ProcessMessage . . . . .	45
7.11.3.8 Receive . . . . .	46
7.11.3.9 Send . . . . .	47
7.11.3.10 StartAcquisition . . . . .	48
7.11.3.11 StopAcquisition . . . . .	48
7.11.3.12 SwitchClientType . . . . .	49
7.11.4 Field Documentation . . . . .	49
7.11.4.1 fNumAttempts . . . . .	49
7.11.4.2 fPID . . . . .	49
7.11.4.3 fWS . . . . .	49
7.12 Socket Class Reference . . . . .	49
7.12.1 Detailed Description . . . . .	51
7.12.2 Member Typedef Documentation . . . . .	51
7.12.2.1 SocketCollection . . . . .	51

7.12.3	Constructor & Destructor Documentation	51
7.12.3.1	Socket	51
7.12.3.2	Socket	51
7.12.3.3	~Socket	51
7.12.4	Member Function Documentation	51
7.12.4.1	AcceptConnections	51
7.12.4.2	Bind	52
7.12.4.3	Configure	52
7.12.4.4	Create	52
7.12.4.5	DumpConnected	52
7.12.4.6	FetchMessage	52
7.12.4.7	GetPort	53
7.12.4.8	GetSocketId	53
7.12.4.9	GetSocketType	53
7.12.4.10	IsWebSocket	53
7.12.4.11	Listen	53
7.12.4.12	PrepareConnection	53
7.12.4.13	SelectConnections	54
7.12.4.14	SendMessage	54
7.12.4.15	SetPort	54
7.12.4.16	SetSocketId	54
7.12.4.17	Start	54
7.12.4.18	Stop	54
7.12.5	Field Documentation	55
7.12.5.1	fAddress	55
7.12.5.2	fBuffer	55
7.12.5.3	fMaster	55
7.12.5.4	fPort	55
7.12.5.5	fReadFds	55
7.12.5.6	fSocketId	55
7.12.5.7	fSocketsConnected	55
7.13	SocketMessage Class Reference	55
7.13.1	Detailed Description	57
7.13.2	Constructor & Destructor Documentation	57
7.13.2.1	SocketMessage	57
7.13.2.2	SocketMessage	57
7.13.2.3	SocketMessage	58
7.13.2.4	SocketMessage	58
7.13.2.5	SocketMessage	58
7.13.2.6	SocketMessage	58

7.13.2.7	SocketMessage	59
7.13.2.8	SocketMessage	59
7.13.2.9	SocketMessage	59
7.13.2.10	SocketMessage	59
7.13.2.11	SocketMessage	60
7.13.2.12	~SocketMessage	60
7.13.3	Member Function Documentation	60
7.13.3.1	Dump	60
7.13.3.2	GetIntValue	60
7.13.3.3	GetKey	60
7.13.3.4	GetString	60
7.13.3.5	GetValue	60
7.13.3.6	GetVectorValue	60
7.13.3.7	Object	61
7.13.3.8	SetKeyValue	61
7.13.3.9	SetKeyValue	61
7.13.3.10	SetKeyValue	61
7.13.3.11	SetKeyValue	62
7.13.3.12	String	62
7.13.4	Field Documentation	62
7.13.4.1	fMessage	62
7.14	VME::TDCErrorFlag Class Reference	62
7.14.1	Detailed Description	63
7.14.2	Constructor & Destructor Documentation	63
7.14.2.1	TDCErrorFlag	63
7.14.2.2	~TDCErrorFlag	63
7.14.3	Member Function Documentation	63
7.14.3.1	Dump	63
7.14.3.2	GetWord	63
7.14.3.3	HasGroupError	63
7.14.3.4	HasInternalChipError	63
7.14.3.5	HasL1BufferOverflow	63
7.14.3.6	HasReachedEventSizeLimit	63
7.14.3.7	HasReadoutFIFOOverflow	63
7.14.3.8	HasTriggerFIFOOverflow	63
7.14.4	Friends And Related Function Documentation	64
7.14.4.1	operator<<	64
7.14.5	Field Documentation	64
7.14.5.1	fWord	64
7.15	VME::TDCEvent Class Reference	64

7.15.1 Detailed Description . . . . .	65
7.15.2 Member Enumeration Documentation . . . . .	65
7.15.2.1 EventType . . . . .	65
7.15.3 Constructor & Destructor Documentation . . . . .	65
7.15.3.1 TDCEvent . . . . .	65
7.15.3.2 TDCEvent . . . . .	65
7.15.3.3 TDCEvent . . . . .	65
7.15.3.4 ~TDCEvent . . . . .	65
7.15.4 Member Function Documentation . . . . .	66
7.15.4.1 Dump . . . . .	66
7.15.4.2 GetBunchId . . . . .	66
7.15.4.3 GetChannelId . . . . .	66
7.15.4.4 GetErrorFlags . . . . .	66
7.15.4.5 GetETTT . . . . .	67
7.15.4.6 GetEventCount . . . . .	67
7.15.4.7 GetEventId . . . . .	67
7.15.4.8 GetGeo . . . . .	68
7.15.4.9 GetLeadingTime . . . . .	68
7.15.4.10 GetStatus . . . . .	68
7.15.4.11 GetTDCId . . . . .	69
7.15.4.12 GetTrailingTime . . . . .	69
7.15.4.13 GetType . . . . .	69
7.15.4.14 GetWidth . . . . .	69
7.15.4.15 GetWord . . . . .	69
7.15.4.16 GetWordCount . . . . .	69
7.15.4.17 IsTrailing . . . . .	70
7.15.4.18 SetWord . . . . .	70
7.15.5 Field Documentation . . . . .	70
7.15.5.1 fWord . . . . .	70
7.16 VME::TDCMeasurement Class Reference . . . . .	70
7.16.1 Member Enumeration Documentation . . . . .	71
7.16.1.1 Type . . . . .	71
7.16.2 Constructor & Destructor Documentation . . . . .	71
7.16.2.1 TDCMeasurement . . . . .	71
7.16.2.2 TDCMeasurement . . . . .	71
7.16.2.3 ~TDCMeasurement . . . . .	71
7.16.3 Member Function Documentation . . . . .	71
7.16.3.1 Dump . . . . .	72
7.16.3.2 GetBunchId . . . . .	72
7.16.3.3 GetChannelId . . . . .	72

7.16.3.4	GetEventId	72
7.16.3.5	GetLeadingTime	72
7.16.3.6	GetTDCId	72
7.16.3.7	GetTrailingTime	72
7.16.3.8	SetEventsCollection	72
7.16.4	Field Documentation	72
7.16.4.1	fMap	72
7.17	VME::TDCV1x90 Class Reference	73
7.17.1	Detailed Description	75
7.17.2	Member Enumeration Documentation	75
7.17.2.1	DLLMode	75
7.17.2.2	mod_reg	75
7.17.3	Constructor & Destructor Documentation	76
7.17.3.1	TDCV1x90	76
7.17.3.2	~TDCV1x90	77
7.17.4	Member Function Documentation	77
7.17.4.1	abort	77
7.17.4.2	CheckConfiguration	77
7.17.4.3	DisableChannel	77
7.17.4.4	EnableChannel	77
7.17.4.5	FetchEvents	78
7.17.4.6	GetAcquisitionMode	78
7.17.4.7	GetBLTEventNumberRegister	78
7.17.4.8	GetChannelDeadTime	79
7.17.4.9	GetControl	79
7.17.4.10	GetDetectionMode	79
7.17.4.11	GetDLLClock	79
7.17.4.12	GetErrorMarks	80
7.17.4.13	GetETTT	80
7.17.4.14	GetEventCounter	80
7.17.4.15	GetEventStored	80
7.17.4.16	GetFIFOSize	81
7.17.4.17	GetFirmwareRevision	81
7.17.4.18	GetGlobalOffset	81
7.17.4.19	GetModel	82
7.17.4.20	GetOUI	82
7.17.4.21	GetPol	82
7.17.4.22	GetRCAdjust	83
7.17.4.23	GetResolution	83
7.17.4.24	GetSerialNumber	83

7.17.4.25	GetStatus	84
7.17.4.26	GetTDCEncapsulation	84
7.17.4.27	GetTestMode	84
7.17.4.28	GetTriggerConfiguration	84
7.17.4.29	GetWindowOffset	85
7.17.4.30	GetWindowWidth	85
7.17.4.31	HardwareReset	85
7.17.4.32	ReadAcquisitionMode	85
7.17.4.33	ReadDetectionMode	85
7.17.4.34	ReadRegister	85
7.17.4.35	ReadRegister	86
7.17.4.36	SetAcquisitionMode	86
7.17.4.37	SetBLTEventNumberRegister	87
7.17.4.38	SetChannelDeadTime	87
7.17.4.39	SetContinuousStorage	87
7.17.4.40	SetControl	88
7.17.4.41	SetDetectionMode	88
7.17.4.42	SetDLLClock	88
7.17.4.43	SetErrorMarks	89
7.17.4.44	SetETTT	89
7.17.4.45	SetFIFOSize	89
7.17.4.46	SetGlobalOffset	90
7.17.4.47	SetLSBTraileadEdge	90
7.17.4.48	SetPairModeResolution	90
7.17.4.49	SetPol	91
7.17.4.50	SetRCAdjust	91
7.17.4.51	SetStatus	91
7.17.4.52	SetTDCEncapsulation	92
7.17.4.53	SetTestMode	92
7.17.4.54	SetTriggerMatching	92
7.17.4.55	SetVerboseLevel	93
7.17.4.56	SetWindowOffset	93
7.17.4.57	SetWindowWidth	93
7.17.4.58	SoftwareClear	93
7.17.4.59	SoftwareReset	94
7.17.4.60	WaitMicro	94
7.17.4.61	WriteRegister	94
7.17.4.62	WriteRegister	94
7.17.5	Field Documentation	95
7.17.5.1	am	95

7.17.5.2	am_blt	95
7.17.5.3	fAcquisitionMode	95
7.17.5.4	fBaseAddr	95
7.17.5.5	fBuffer	95
7.17.5.6	fDetectionMode	95
7.17.5.7	fErrorMarks	95
7.17.5.8	fHandle	95
7.17.5.9	fVerb	95
7.17.5.10	fWindowWidth	95
7.17.5.11	gEnd	95
7.17.5.12	nchannels	95
7.17.5.13	pair_lead_res	95
7.17.5.14	pair_width_res	95
7.17.5.15	trailead_edge_res	95
7.18	VME::TDCV1x90Control Class Reference	95
7.18.1	Detailed Description	96
7.18.2	Constructor & Destructor Documentation	96
7.18.2.1	TDCV1x90Control	96
7.18.2.2	~TDCV1x90Control	96
7.18.3	Member Function Documentation	96
7.18.3.1	Dump	97
7.18.3.2	GetAlign64	97
7.18.3.3	GetBusError	97
7.18.3.4	GetCompensation	98
7.18.3.5	GetEmptyEvent	98
7.18.3.6	GetETTT	98
7.18.3.7	GetEventFIFO	98
7.18.3.8	GetMEBAccess	98
7.18.3.9	GetSRAMCompensation	98
7.18.3.10	GetSWTermination	98
7.18.3.11	GetTermination	98
7.18.3.12	GetTestFIFO	98
7.18.3.13	GetValue	98
7.18.3.14	SetAlign64	98
7.18.3.15	SetBusError	98
7.18.3.16	SetCompensation	99
7.18.3.17	SetEmptyEvent	99
7.18.3.18	SetETTT	99
7.18.3.19	SetEventFIFO	100
7.18.3.20	SetMEBAccess	100

7.18.3.21 SetSRAMCompensation . . . . .	100
7.18.3.22 SetSWTermination . . . . .	101
7.18.3.23 SetTermination . . . . .	101
7.18.3.24 SetTestFIFO . . . . .	101
7.18.4 Field Documentation . . . . .	101
7.18.4.1 fWord . . . . .	101
7.19 VME::TDCV1x90Status Class Reference . . . . .	102
7.19.1 Detailed Description . . . . .	102
7.19.2 Member Enumeration Documentation . . . . .	102
7.19.2.1 TDCResolution . . . . .	102
7.19.3 Constructor & Destructor Documentation . . . . .	103
7.19.3.1 TDCV1x90Status . . . . .	103
7.19.3.2 ~TDCV1x90Status . . . . .	103
7.19.4 Member Function Documentation . . . . .	103
7.19.4.1 AlmostFull . . . . .	103
7.19.4.2 BusError . . . . .	103
7.19.4.3 DataReady . . . . .	103
7.19.4.4 Dump . . . . .	104
7.19.4.5 Error . . . . .	104
7.19.4.6 Error . . . . .	105
7.19.4.7 Full . . . . .	105
7.19.4.8 GetValue . . . . .	105
7.19.4.9 HeadersEnabled . . . . .	105
7.19.4.10 PairMode . . . . .	105
7.19.4.11 Purged . . . . .	105
7.19.4.12 Resolution . . . . .	105
7.19.4.13 TerminationOn . . . . .	105
7.19.4.14 TriggerLost . . . . .	105
7.19.4.15 TriggerMatching . . . . .	105
7.19.5 Field Documentation . . . . .	105
7.19.5.1 fWord . . . . .	105
7.20 VME::trailead_t Struct Reference . . . . .	105
7.20.1 Field Documentation . . . . .	106
7.20.1.1 ettt . . . . .	106
7.20.1.2 event_count . . . . .	106
7.20.1.3 leading . . . . .	106
7.20.1.4 total_hits . . . . .	106
7.20.1.5 trailing . . . . .	106
7.21 VMEReader Class Reference . . . . .	106
7.21.1 Detailed Description . . . . .	108



7.21.2	Member Typedef Documentation . . . . .	108
7.21.2.1	TDCCollection . . . . .	108
7.21.3	Constructor & Destructor Documentation . . . . .	108
7.21.3.1	VMEReader . . . . .	108
7.21.3.2	~VMEReader . . . . .	109
7.21.4	Member Function Documentation . . . . .	109
7.21.4.1	Abort . . . . .	109
7.21.4.2	AddTDC . . . . .	109
7.21.4.3	GetRunNumber . . . . .	109
7.21.4.4	GetTDC . . . . .	110
7.21.4.5	StartPulser . . . . .	110
7.21.4.6	StopPulser . . . . .	110
7.21.5	Field Documentation . . . . .	110
7.21.5.1	fBridge . . . . .	110
7.21.5.2	flsPulserStarted . . . . .	111
7.21.5.3	fOnSocket . . . . .	111
7.21.5.4	fTDCCollection . . . . .	111
<b>Index</b>		<b>113</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="#">13</a>



## Chapter 3

# Hierarchical Index

### 3.1 Class Hierarchy

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

VME::BridgeVx718 . . . . .	19
VME::BridgeVx718Control . . . . .	22
VME::BridgeVx718Status . . . . .	23
Exception . . . . .	30
file_header_t . . . . .	33
FileReader . . . . .	33
VME::GlobalOffset . . . . .	36
Message . . . . .	39
HTTPMessage . . . . .	36
SocketMessage . . . . .	55
Socket . . . . .	49
Client . . . . .	24
VMEReader . . . . .	106
Messenger . . . . .	41
VME::TDCErrorFlag . . . . .	62
VME::TDCEvent . . . . .	64
VME::TDCMeasurement . . . . .	70
VME::TDCV1x90 . . . . .	73
VME::TDCV1x90Control . . . . .	95
VME::TDCV1x90Status . . . . .	102
VME::trailead_t . . . . .	105





## 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="#">VME::BridgeVx718Control</a>	22
<a href="#">VME::BridgeVx718Status</a>	23
<a href="#">Client</a>	
Base client object for the socket	24
<a href="#">Exception</a>	
A simple exception handler	30
<a href="#">file_header_t</a>	
Header to the output files	33
<a href="#">FileReader</a>	
Handler for a TDC output file readout	33
<a href="#">VME::GlobalOffset</a>	36
<a href="#">HTTPMessage</a>	
<a href="#">Message</a> to be transmitted through a WebSocket protocol	36
<a href="#">Message</a>	
Base socket message type	39
<a href="#">Messenger</a>	
Base master object for the socket	41
<a href="#">Socket</a>	
Base socket object from which clients/master from a socket inherit	49
<a href="#">SocketMessage</a>	
Socket-passed message type	55
<a href="#">VME::TDCErrorFlag</a>	
Error flags handler	62
<a href="#">VME::TDCEvent</a>	
HPTDC event parser	64
<a href="#">VME::TDCMeasurement</a>	70
<a href="#">VME::TDCV1x90</a>	73
<a href="#">VME::TDCV1x90Control</a>	
TDC control register	95
<a href="#">VME::TDCV1x90Status</a>	
TDC status register	102
<a href="#">VME::trailead_t</a>	105
<a href="#">VMEReader</a>	106



## 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*
- class [BridgeVx718Control](#)
- class [BridgeVx718Status](#)
- struct [GlobalOffset](#)
- class [TDCErrorFlag](#)  
*Error flags handler.*
- class [TDCEvent](#)  
*HPTDC event parser.*
- class [TDCMeasurement](#)
- class [TDCV1x90](#)
- class [TDCV1x90Control](#)  
*TDC control register.*
- class [TDCV1x90Status](#)  
*TDC status register.*
- struct [trailead\\_t](#)

#### Typedefs

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

#### Enumerations

- enum [BridgeType](#) { [CAEN\\_V1718](#), [CAEN\\_V2718](#) }  
*Compatible bridge types.*
- enum [AcquisitionMode](#) { [CONT\\_STORAGE](#), [TRIG\\_MATCH](#) }  
*TDC acquisition mode.*
- enum [DetectionMode](#) { [PAIR](#) = 0x0, [OTRILING](#) = 0x1, [OLEADING](#) = 0x2, [TRAILEAD](#) = 0x3 }

- enum `trig_conf` {  
`MATCH_WIN_WIDTH` = 0, `WIN_OFFSET` = 1, `EXTRA_SEARCH_WIN_WIDTH` = 2, `REJECT_MARGIN` = 3,  
`TRIG_TIME_SUB` = 4 }
- enum `trailead_edge_lsbs` { `r800ps` = 0, `r200ps` = 1, `r100ps` = 2, `r25ps` = 3 }
- enum `micro_handshake` { `WRITE_OK` = 0, `READ_OK` = 1 }

### 6.1.1 Typedef Documentation

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

### 6.1.2 Enumeration Type Documentation

#### 6.1.2.1 enum VME::AcquisitionMode

TDC acquisition mode.

Author

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

Enumerator

***CONT\_STORAGE***

***TRIG\_MATCH***

#### 6.1.2.2 enum VME::BridgeType

Compatible bridge types.

Enumerator

***CAEN\_V1718***

***CAEN\_V2718***

#### 6.1.2.3 enum VME::DetectionMode

Enumerator

***PAIR***

***OTRILING***

***OLEADING***

***TRAILEAD***

#### 6.1.2.4 enum VME::micro\_handshake

Enumerator

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

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

## 6.1.2.5 enum VME::trailead\_edge\_lsb

Enumerator

*r800ps*  
*r200ps*  
*r100ps*  
*r25ps*

## 6.1.2.6 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::TDCV1x900pcodes::AUTOLOAD_DEF_CONFI ( 0x0900 )`

6.2.1.2 Opcode `VME::TDCV1x900pcodes::AUTOLOAD_USER_CONF ( 0x0800 )`

6.2.1.3 Opcode `VME::TDCV1x900pcodes::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) const  
*Set and control the output lines.*
- void [OutputOn](#) (CVOutputSelect output) const
- void [OutputOff](#) (CVOutputSelect output) const
- void [InputConf](#) (CVInputSelect input) const  
*Set and read the input lines.*
- void [InputRead](#) (CVInputSelect input) const
- void [StartPulser](#) (double period, double width, unsigned char num\_pulses=0) const
- void [StopPulser](#) () const

#### Private Member Functions

- void [WriteRegister](#) (CVRegisters addr, const uint16\_t &data) const
- void [WriteRegister](#) (CVRegisters addr, const uint32\_t &data) const
- void [ReadRegister](#) (CVRegisters addr, uint16\_t \*data) const
- void [ReadRegister](#) (CVRegisters addr, uint32\_t \*data) const

#### Private Attributes

- int32\_t [fHandle](#)  
*Device handle.*
- uint32\_t [fBaseAddr](#)

### 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* ) const

Set and read the input lines.

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

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

Set and control the output lines.

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

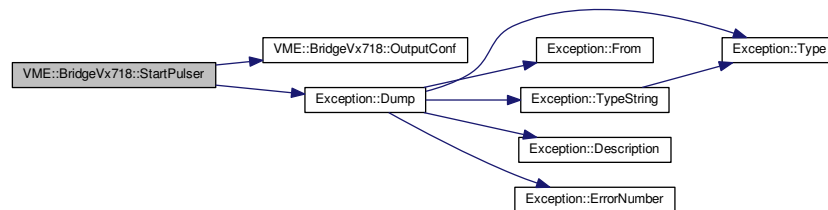
7.1.3.7 void VME::BridgeVx718::OutputOn ( CVOutputSelect *output* ) const

7.1.3.8 void VME::BridgeVx718::ReadRegister ( CVRegisters *addr*, uint16\_t \* *data* ) const [private]

7.1.3.9 void VME::BridgeVx718::ReadRegister ( CVRegisters *addr*, uint32\_t \* *data* ) const [private]

7.1.3.10 void VME::BridgeVx718::StartPulser ( double *period*, double *width*, unsigned char *num\_pulses* = 0 ) const

Here is the call graph for this function:



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

7.1.3.12 void VME::BridgeVx718::WriteRegister ( CVRegisters *addr*, const uint16\_t & *data* ) const [private]

7.1.3.13 void VME::BridgeVx718::WriteRegister ( CVRegisters *addr*, const uint32\_t & *data* ) const [private]

## 7.1.4 Field Documentation

7.1.4.1 uint32\_t VME::BridgeVx718::fBaseAddr [private]

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

Device handle.

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

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

## 7.2 VME::BridgeVx718Control Class Reference

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

### Public Member Functions

- [BridgeVx718Control](#) (uint16\_t word)
- virtual [~BridgeVx718Control](#) ()
- bool [GetArbiterType](#) () const  
*Arbiter type.*
- bool [GetRequesterType](#) () const  
*Requester type.*
- bool [GetReleaseType](#) () const  
*Release type.*
- unsigned int [GetBusReqLevel](#) () const
- bool [GetInterruptReq](#) () const
- bool [GetSysRes](#) () const
- bool [GetBusTimeout](#) () const  
*VME bus timeout.*
- bool [GetAddressIncrement](#) () const  
*Address Increment.*

### Private Attributes

- uint16\_t [fWord](#)

### 7.2.1 Constructor & Destructor Documentation

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

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

### 7.2.2 Member Function Documentation

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

Address Increment.

Returns

true if enabled, else false (FIFO mode)

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

Arbiter type.

Returns

true if "Round Robin", else fixed priority



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

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

VME bus timeout.

Returns

true if 1400 us, else 50 us

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

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

Release type.

Returns

true if release on request, else release when done

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

Requester type.

Returns

true if demand, else fair

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

## 7.2.3 Field Documentation

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

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

- include/VME\_BridgeVx718.h

## 7.3 VME::BridgeVx718Status Class Reference

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

### Public Member Functions

- [BridgeVx718Status](#) (uint16\_t word)
- virtual [~BridgeVx718Status](#) ()
- void [Dump](#) () const
- bool [GetSystemReset](#) () const
- bool [GetSystemControl](#) () const
- bool [GetDTACK](#) () const
- bool [GetBERR](#) () const
- bool [GetDipSwitch](#) (unsigned int sw) const
- bool [GetUSBType](#) () const

## Private Attributes

- uint16\_t [fWord](#)

### 7.3.1 Constructor & Destructor Documentation

7.3.1.1 VME::BridgeVx718Status::BridgeVx718Status ( uint16\_t *word* ) [\[inline\]](#)

7.3.1.2 virtual VME::BridgeVx718Status::~~BridgeVx718Status ( ) [\[inline\]](#), [\[virtual\]](#)

### 7.3.2 Member Function Documentation

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

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

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

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

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

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

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

### 7.3.3 Field Documentation

7.3.3.1 uint16\_t VME::BridgeVx718Status::fWord [\[private\]](#)

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

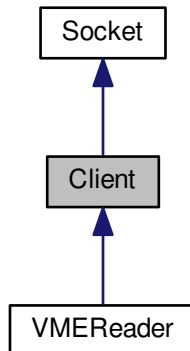
- include/VME\_BridgeVx718.h

## 7.4 Client Class Reference

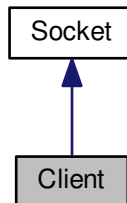
Base client object for the socket.

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

Inheritance diagram for Client:



Collaboration diagram for Client:



## Public Member Functions

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

### 7.4.1 Detailed Description

Base client object for the socket.

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

#### Author

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

#### Date

24 Mar 2015

### 7.4.2 Constructor & Destructor Documentation

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

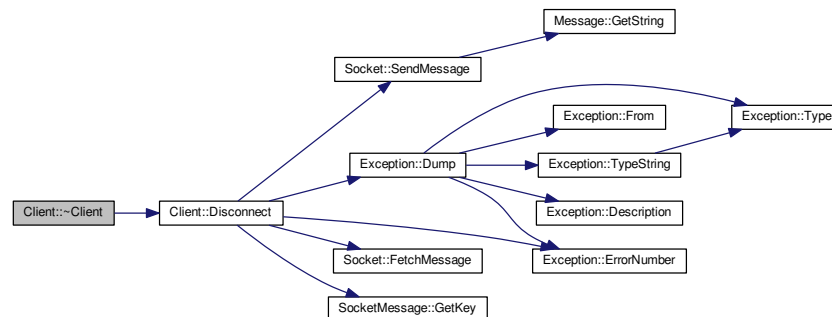
General void client constructor.

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

Bind a socket client to a given port.

### 7.4.2.3 Client::~~Client( ) [virtual]

Here is the call graph for this function:

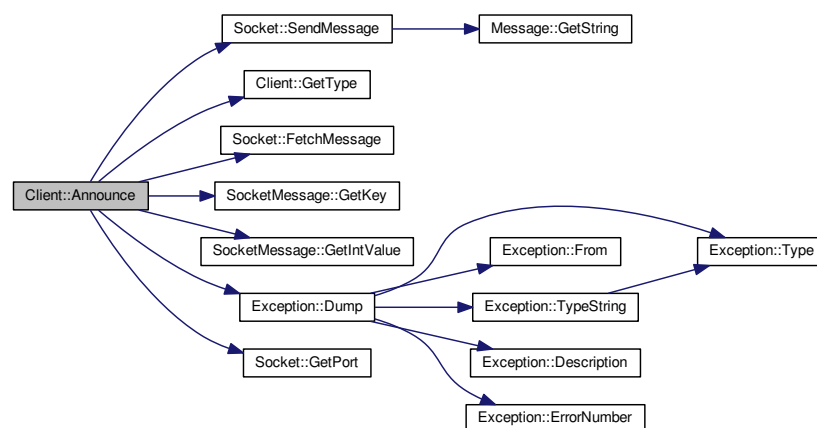


## 7.4.3 Member Function Documentation

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

Announce our entry on the socket to its master.

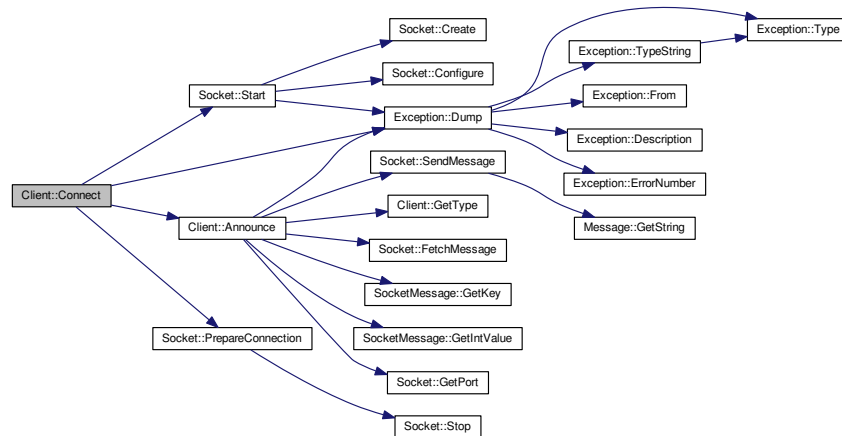
Here is the call graph for this function:



### 7.4.3.2 bool Client::Connect( )

Bind this client to the socket.

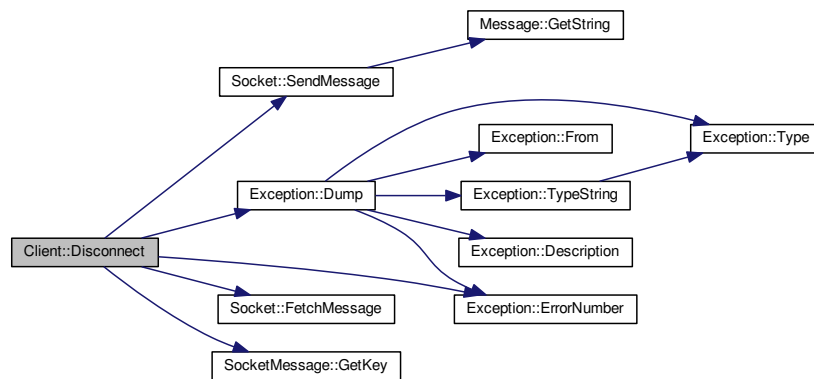
Here is the call graph for this function:



#### 7.4.3.3 void Client::Disconnect ( )

Unbind this client from the socket.

Here is the call graph for this function:



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

[Socket](#) actor type retrieval method.

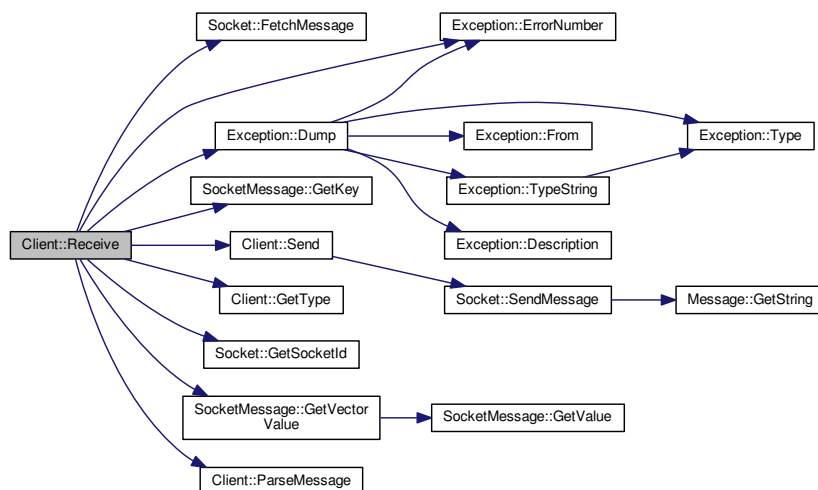
#### 7.4.3.5 virtual void Client::ParseMessage ( const SocketMessage & m ) [inline],[virtual]

Parse a [SocketMessage](#) received from the master.

## 7.4.3.6 void Client::Receive ( )

Receive a socket message from the master.

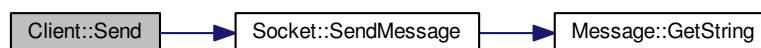
Here is the call graph for this function:



## 7.4.3.7 void Client::Send ( const Message &amp; m ) const [inline]

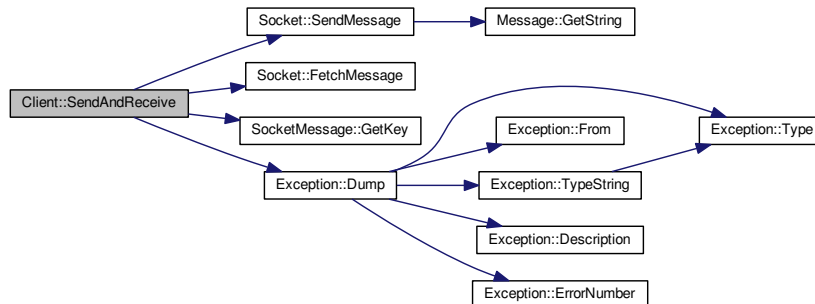
Send a message to the master through the socket.

Here is the call graph for this function:



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

Here is the call graph for this function:



### 7.4.4 Field Documentation

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

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

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

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

## 7.5 Exception Class Reference

A simple exception handler.

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

### Public Member Functions

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

### Private Attributes

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



### 7.5.1 Detailed Description

A simple exception handler.

#### Author

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

#### Date

24 Mar 2015

### 7.5.2 Constructor & Destructor Documentation

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

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

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

Here is the call graph for this function:

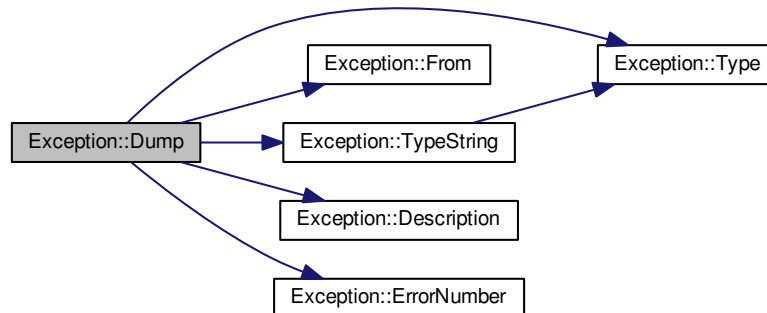


### 7.5.3 Member Function Documentation

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

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

Here is the call graph for this function:



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

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

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

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

Here is the call graph for this function:



## 7.5.4 Field Documentation

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

7.5.4.2 `int Exception::fErrorNumber [private]`

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

7.5.4.4 `ExceptionType Exception::fType [private]`

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

- `include/Exception.h`

## 7.6 file\_header\_t Struct Reference

Header to the output files.

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

### Data Fields

- uint32\_t [magic](#)
- uint32\_t [run\\_id](#)
- uint32\_t [spill\\_id](#)
- uint8\_t [num\\_hptdc](#)
- [VME::AcquisitionMode](#) [acq\\_mode](#)
- [VME::DetectionMode](#) [det\\_mode](#)

### 7.6.1 Detailed Description

Header to the output files.

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

#### Author

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

#### Date

14 Apr 2015

### 7.6.2 Field Documentation

7.6.2.1 [VME::AcquisitionMode](#) [file\\_header\\_t::acq\\_mode](#)

7.6.2.2 [VME::DetectionMode](#) [file\\_header\\_t::det\\_mode](#)

7.6.2.3 [uint32\\_t](#) [file\\_header\\_t::magic](#)

7.6.2.4 [uint8\\_t](#) [file\\_header\\_t::num\\_hptdc](#)

7.6.2.5 [uint32\\_t](#) [file\\_header\\_t::run\\_id](#)

7.6.2.6 [uint32\\_t](#) [file\\_header\\_t::spill\\_id](#)

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

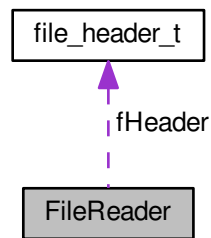
- [include/FileConstants.h](#)

## 7.7 FileReader Class Reference

Handler for a TDC output file readout.

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

Collaboration diagram for FileReader:



## Public Member Functions

- `FileReader` (std::string name, const `VME::AcquisitionMode` &ro)  
*Class constructor.*
- `~FileReader` ()
- unsigned int `GetNumTDCs` () const
- bool `GetNextEvent` (`VME::TDCEvent` \*)
- bool `GetNextMeasurement` (unsigned int channel\_id, `VME::TDCMeasurement` \*m)  
*Fetch the next full measurement on a given channel.*

## Private Attributes

- std::ifstream `fFile`
- `file_header_t` `fHeader`
- `VME::AcquisitionMode` `fReadoutMode`

### 7.7.1 Detailed Description

Handler for a TDC output file readout.

#### Author

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

#### Date

Jun 2015

### 7.7.2 Constructor & Destructor Documentation

#### 7.7.2.1 `FileReader::FileReader` ( std::string name, const `VME::AcquisitionMode` & ro )

Class constructor.

## Parameters

in	<i>name</i>	Path to the file to read
in	<i>ro</i>	Data readout mode (continuous storage or trigger matching)

## 7.7.2.2 FileReader::~~FileReader ( )

## 7.7.3 Member Function Documentation

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

Here is the call graph for this function:

7.7.3.2 bool FileReader::GetNextMeasurement ( unsigned int *channel\_id*, VME::TDCMeasurement \* *m* )

Fetch the next full measurement on a given channel.

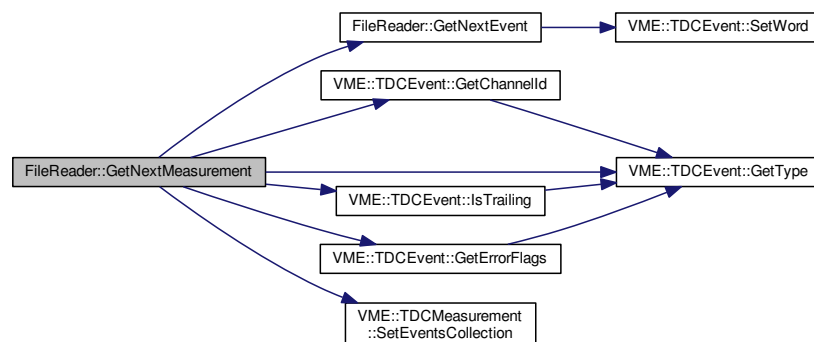
## Parameters

in	<i>channel_id</i>	Unique identifier of the channel number to retrieve
out	<i>m</i>	A full measurement with leading, trailing times, ...

## Returns

A boolean stating the success of retrieval operation

Here is the call graph for this function:



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

## 7.7.4 Field Documentation

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

7.7.4.2 `file_header_t FileReader::fHeader [private]`

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

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

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

## 7.8 VME::GlobalOffset Struct Reference

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

### Data Fields

- `uint16_t coarse`
- `uint16_t fine`

### 7.8.1 Field Documentation

7.8.1.1 `uint16_t VME::GlobalOffset::coarse`

7.8.1.2 `uint16_t VME::GlobalOffset::fine`

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

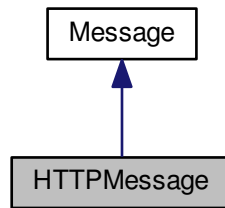
- `include/VME_TDCV1x90.h`

## 7.9 HTTPMessage Class Reference

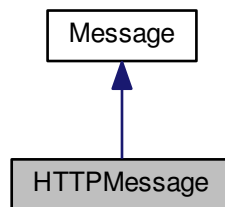
[Message](#) to be transmitted through a WebSocket protocol.

```
#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

### 7.9.1 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)

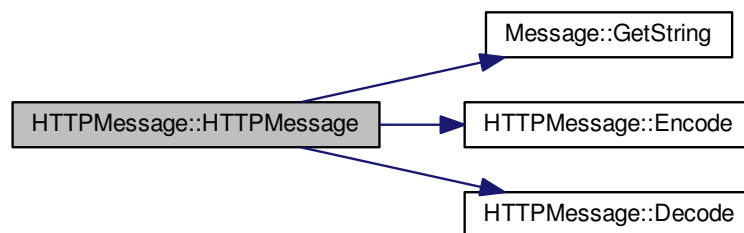
## Date

1 Apr 2015

## 7.9.2 Constructor & Destructor Documentation

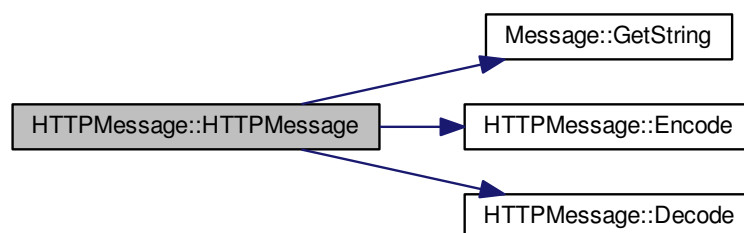
### 7.9.2.1 HTTPMessage::HTTPMessage ( WebSocket \* ws, Message m, MessageAction a ) [inline]

Here is the call graph for this function:



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

Here is the call graph for this function:



## 7.9.3 Member Function Documentation

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

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

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



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

## 7.9.4 Field Documentation

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

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

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

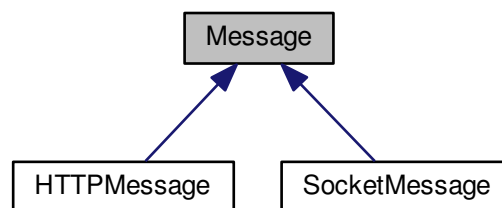
- `include/HTTPMessage.h`

## 7.10 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`

### 7.10.1 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

### 7.10.2 Constructor & Destructor Documentation

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

Void message constructor.

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

Construct a message from a string.

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

Construct a message from a string.

7.10.2.4 `virtual Message::~Message ( ) [inline],[virtual]`

### 7.10.3 Member Function Documentation

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

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

Placeholder for the MessageKey retrieval method.

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

Retrieve the string carried by this message as a whole.

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

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

### 7.10.4 Field Documentation

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

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

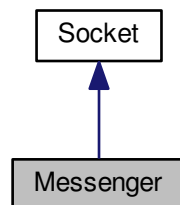
- include/Message.h

## 7.11 Messenger Class Reference

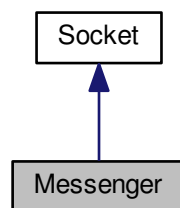
Base master object for the socket.

```
#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

### 7.11.1 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

### 7.11.2 Constructor & Destructor Documentation

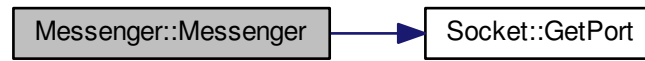
#### 7.11.2.1 [Messenger::Messenger](#) ( )

Build a void master object or socket actor.

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

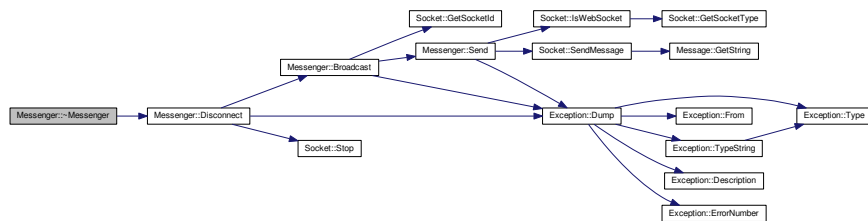
Build a master object to control the socket.

Here is the call graph for this function:



### 7.11.2.3 Messenger::~~Messenger ( )

Here is the call graph for this function:



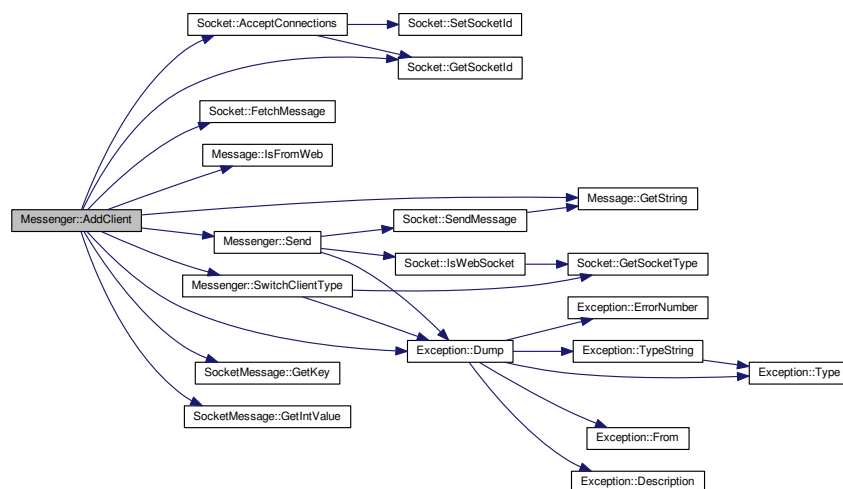
## 7.11.3 Member Function Documentation

### 7.11.3.1 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:



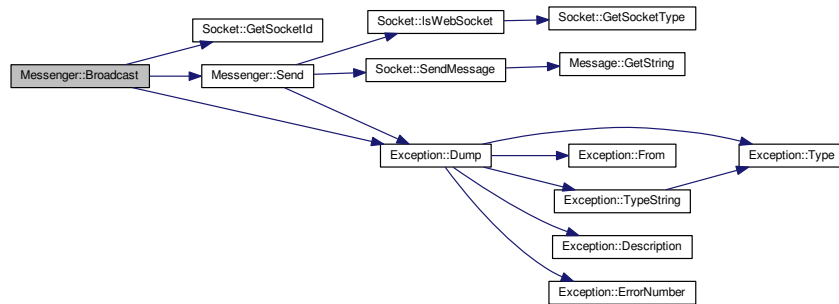
### 7.11.3.2 void Messenger::Broadcast ( const Message & m ) const

Emit a message to all clients connected through the socket.

#### Parameters

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

Here is the call graph for this function:

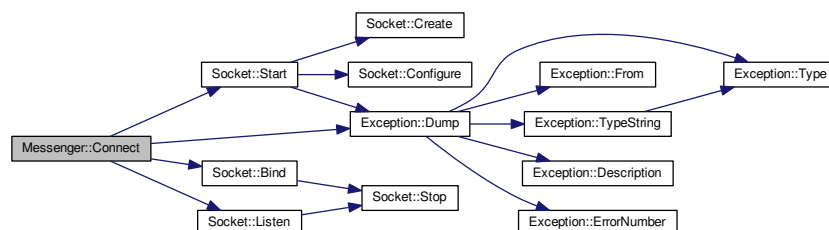


### 7.11.3.3 bool Messenger::Connect ( )

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:

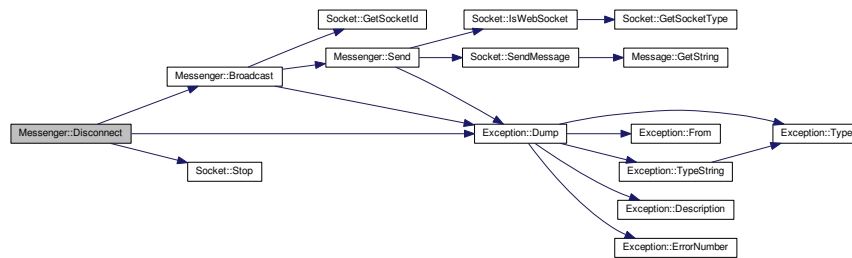


### 7.11.3.4 void Messenger::Disconnect ( )

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:



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

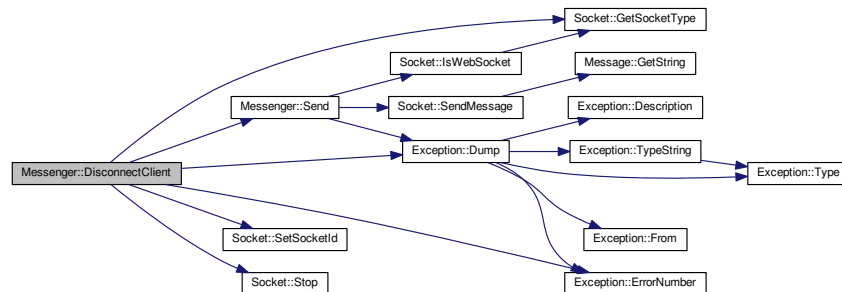
Disconnect a client.

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:



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

[Socket](#) actor type retrieval method.

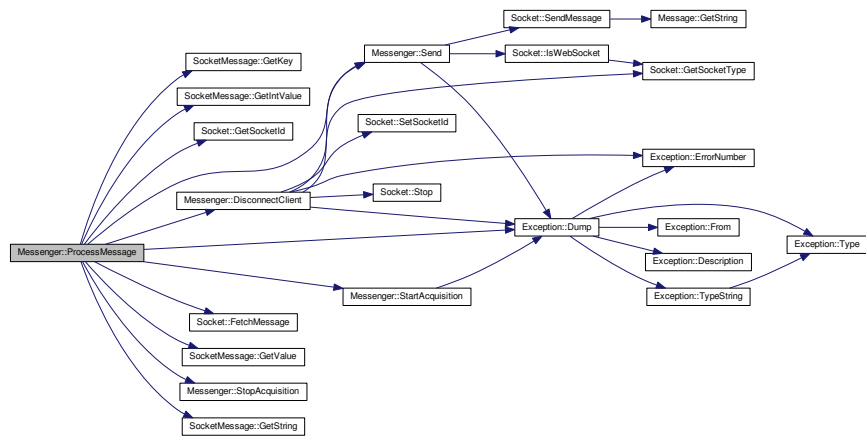
#### 7.11.3.7 void Messenger::ProcessMessage ( SocketMessage *m*, int *sid* ) [private]

Process a message received from the socket.

##### Parameters

in	Unique	identifier of the client sending the message
----	--------	--

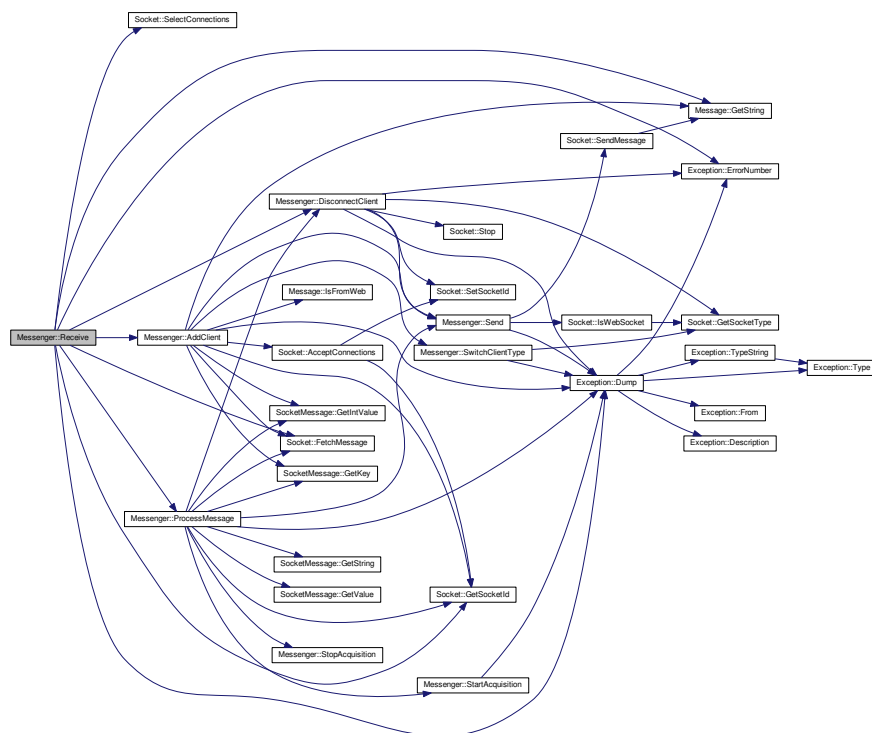
Here is the call graph for this function:



#### 7.11.3.8 void Messenger::Receive ( )

Handle a message reception from a client.

Here is the call graph for this function:





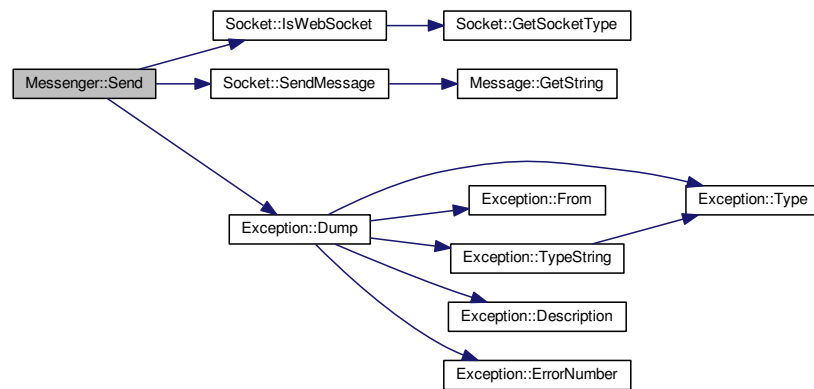
7.11.3.9 void Messenger::Send ( const Message & *m*, int *sid* ) const [inline]

Send any type of message to any client.

## Parameters

in	<i>m</i>	<a href="#">Message</a> to transmit
in	<i>sid</i>	Unique identifier of the client on this socket

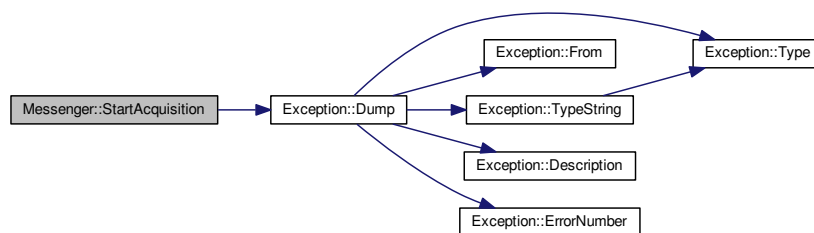
Here is the call graph for this function:



#### 7.11.3.10 void Messenger::StartAcquisition ( )

Start the data acquisition.

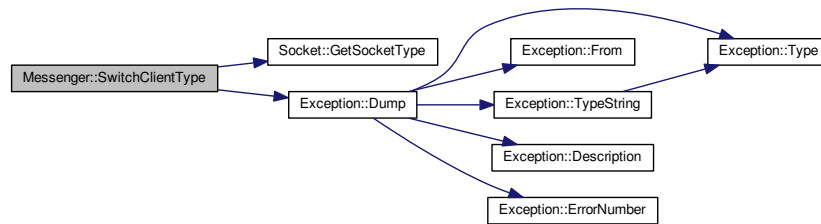
Here is the call graph for this function:



#### 7.11.3.11 void Messenger::StopAcquisition ( )

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

Here is the call graph for this function:



## 7.11.4 Field Documentation

7.11.4.1 `int Messenger::fNumAttempts` [private]

7.11.4.2 `pid_t Messenger::fPID` [private]

7.11.4.3 `WebSocket* Messenger::fWS` [private]

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

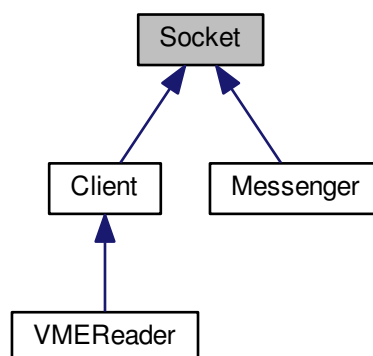
- include/Messenger.h
- src/Messenger.cpp

## 7.12 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](#)

### 7.12.1 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

### 7.12.2 Member Typedef Documentation

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

### 7.12.3 Constructor & Destructor Documentation

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

7.12.3.2 `Socket::Socket ( int port )`

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

### 7.12.4 Member Function Documentation

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

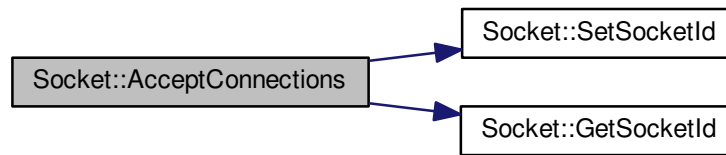
Accept connection from a client.

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:



#### 7.12.4.2 `void Socket::Bind ( )` [protected]

Bind a name to a socket.

##### Returns

Success of the operation

Here is the call graph for this function:



#### 7.12.4.3 `void Socket::Configure ( )` [private]

Configure the socket object for communication.

#### 7.12.4.4 `void Socket::Create ( )` [private]

Create an endpoint for communication.

#### 7.12.4.5 `void Socket::DumpConnected ( )` const

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

Receive a message from a socket.

##### Returns

Received message as a `std::string`

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

Retrieve the port used for this socket.

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

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

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

Here is the call graph for this function:



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

Listen to incoming messages.

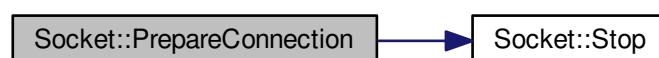
Set the socket to listen to any message coming from outside

Here is the call graph for this function:



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

Here is the call graph for this function:



#### 7.12.4.13 void Socket::SelectConnections ( )

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

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

Send a message on a socket.

Here is the call graph for this function:



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

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

#### 7.12.4.17 bool Socket::Start ( ) [protected]

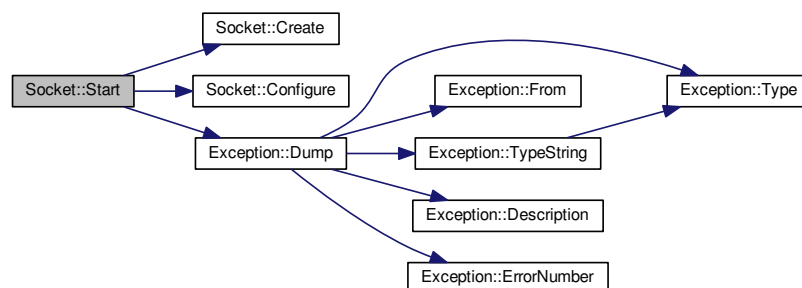
Start the socket.

Launch all mandatory operations to set the socket to be used

#### Returns

Success of the operation

Here is the call graph for this function:



#### 7.12.4.18 void Socket::Stop ( )

Terminates the socket and all attached communications.



### 7.12.5 Field Documentation

7.12.5.1 `struct sockaddr_in Socket::fAddress` [private]

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

7.12.5.3 `fd_set Socket::fMaster` [protected]

Master file descriptor list.

7.12.5.4 `int Socket::fPort` [protected]

7.12.5.5 `fd_set Socket::fReadFds` [protected]

Temp file descriptor list for select()

7.12.5.6 `int Socket::fSocketId` [private]

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

7.12.5.7 **SocketCollection** `Socket::fSocketsConnected` [protected]

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

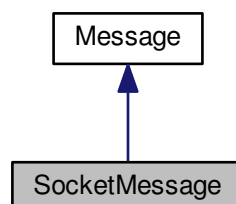
- include/Socket.h
- src/Socket.cpp

## 7.13 SocketMessage Class Reference

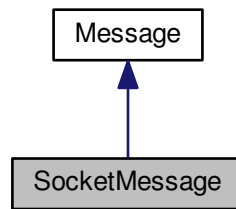
Socket-passed message type.

```
#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

#### 7.13.1 Detailed Description

Socket-passed message type.

#### Author

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

#### Date

26 Mar 2015

#### 7.13.2 Constructor & Destructor Documentation

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

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

Here is the call graph for this function:



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

Here is the call graph for this function:



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

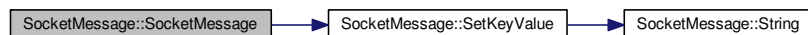
Here is the call graph for this function:



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

Construct a socket message out of a key.

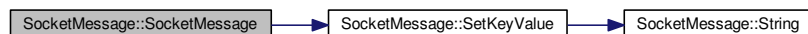
Here is the call graph for this function:



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

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

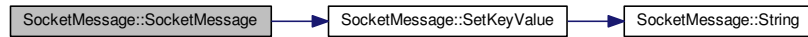
Here is the call graph for this function:



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

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

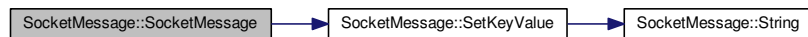
Here is the call graph for this function:



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

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

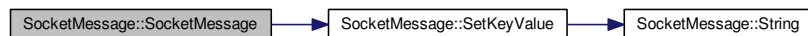
Here is the call graph for this function:



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

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

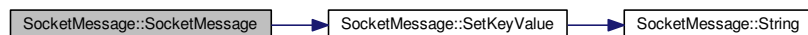
Here is the call graph for this function:



#### 7.13.2.10 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:



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

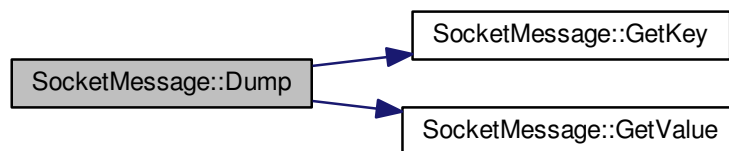
Construct a socket message out of a map of key/string-type value.

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

### 7.13.3 Member Function Documentation

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

Here is the call graph for this function:



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

Extract the message's integer value.

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

Extract the message's key.

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

Extract the whole key:value message.

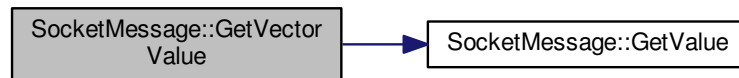
#### 7.13.3.5 std::string SocketMessage::GetValue ( ) const [inline]

Extract the message's string value.

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

Extract the message's vector of string value.

Here is the call graph for this function:

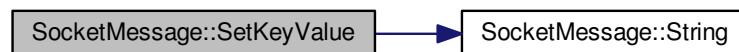


**7.13.3.7** `MessageMap SocketMessage::Object ( ) const` `[inline], [private]`

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

String-valued message.

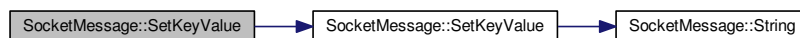
Here is the call graph for this function:



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

Send an integer-valued message.

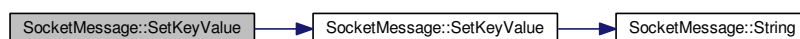
Here is the call graph for this function:



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

Float-valued message.

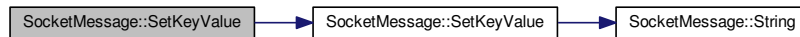
Here is the call graph for this function:



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

Double-valued message.

Here is the call graph for this function:



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

## 7.13.4 Field Documentation

7.13.4.1 `MessageMap SocketMessage::fMessage [private]`

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

- `include/SocketMessage.h`

## 7.14 VME::TDCErrorFlag Class Reference

Error flags handler.

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

### Public Member Functions

- [TDCErrorFlag](#) (uint16\_t ef)
- virtual [~TDCErrorFlag](#) ()
- uint16\_t [GetWord](#) () const
- void [Dump](#) () const
- bool [HasReadoutFIFOOverflow](#) (unsigned int group\_id) const  
*Check whether hits have been lost from read-out FIFO overflow in a given group.*
- bool [HasL1BufferOverflow](#) (unsigned int group\_id) const  
*Check whether hits have been lost from L1 buffer overflow in a given group.*
- bool [HasGroupError](#) (unsigned int group\_id) const  
*Check whether hits have been lost due to error in a given group.*
- bool [HasReachedEventSizeLimit](#) () const  
*Hits rejected because of programmed event size limit.*
- bool [HasTriggerFIFOOverflow](#) () const  
*Event lost (trigger FIFO overflow)*
- bool [HasInternalChipError](#) () const  
*Internal fatal chip error has been detected.*

### Private Attributes

- uint16\_t [fWord](#)



## Friends

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

### 7.14.1 Detailed Description

Error flags handler.

#### Author

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

#### Date

22 Jun 2015

### 7.14.2 Constructor & Destructor Documentation

7.14.2.1 `VME::TDCErrorFlag::TDCErrorFlag ( uint16_t ef ) [inline]`

7.14.2.2 `virtual VME::TDCErrorFlag::~~TDCErrorFlag ( ) [inline], [virtual]`

### 7.14.3 Member Function Documentation

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

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

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

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

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

Internal fatal chip error has been detected.

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

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

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

Hits rejected because of programmed event size limit.

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

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

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

Event lost (trigger FIFO overflow)

## 7.14.4 Friends And Related Function Documentation

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

## 7.14.5 Field Documentation

7.14.5.1 `uint16_t VME::TDCErrorFlag::fWord` [*private*]

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

- `include/VME_TDCEvent.h`

## 7.15 VME::TDCEvent Class Reference

HPTDC event parser.

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

### Public Types

- enum `EventType` {  
`TDCMeasurement` = 0x0, `TDCHeader` = 0x1, `TDCTrailer` = 0x3, `TDCError` = 0x4,  
`GlobalHeader` = 0x8, `GlobalTrailer` = 0x10, `ETTT` = 0x11, `Filler` = 0x18 }

### Public Member Functions

- `TDCEvent ()`
- `TDCEvent (const TDCEvent &ev)`
- `TDCEvent (const uint32_t &word)`
- virtual `~TDCEvent ()`
- void `Dump ()` const
- void `SetWord (const uint32_t &word)`
- `uint32_t GetWord ()` const
- `EventType GetType ()` const  
*Type of packet read out from the TDC.*
- unsigned int `GetTDCId ()` const  
*Programmed identifier of master TDC providing the event.*
- `uint16_t GetEventId ()` const  
*Event identifier from event counter.*
- `uint16_t GetWordCount ()` const  
*Total number of words in event (including headers and trailers)*
- unsigned int `GetGeo ()` const
- unsigned int `GetChannelId ()` const
- `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.*
- unsigned int [GetWidth](#) () const  
*Width of pulse in programmed time resolution.*
- uint32\_t [GetTrailingTime](#) () const  
*Trailing edge measurement in programmed time resolution.*
- unsigned int [GetStatus](#) () const
- [TDCErrorFlag](#) [GetErrorFlags](#) () const  
*Return error flags if an error condition has been detected.*

### Private Attributes

- uint32\_t [fWord](#)

### 7.15.1 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

### 7.15.2 Member Enumeration Documentation

#### 7.15.2.1 enum VME::TDCEvent::EventType

Enumerator

***TDCTMeasurement***

***TDCTHeader***

***TDCTTrailer***

***TDCTError***

***GlobalHeader***

***GlobalTrailer***

***ETTT***

***Filler***

### 7.15.3 Constructor & Destructor Documentation

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

7.15.3.2 VME::TDCEvent::TDCEvent ( const TDCEvent & ev ) [inline]

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

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

## 7.15.4 Member Function Documentation

### 7.15.4.1 void VME::TDCEvent::Dump ( ) const [inline]

Here is the call graph for this function:



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

Bunch identifier of trigger (or trigger time tag)

Here is the call graph for this function:



### 7.15.4.3 unsigned int VME::TDCEvent::GetChannelId ( ) const [inline]

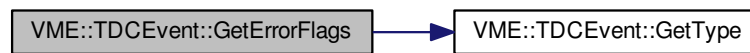
Here is the call graph for this function:



### 7.15.4.4 TDCErrorFlag VME::TDCEvent::GetErrorFlags ( ) const [inline]

Return error flags if an error condition has been detected.

Here is the call graph for this function:



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

Extended trigger time tag.

Here is the call graph for this function:



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

Total number of events.

Here is the call graph for this function:



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

Event identifier from event counter.

Here is the call graph for this function:



#### 7.15.4.8 unsigned int VME::TDCEvent::GetGeo ( ) const [inline]

Here is the call graph for this function:



#### 7.15.4.9 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:



#### 7.15.4.10 unsigned int VME::TDCEvent::GetStatus ( ) const [inline]

Here is the call graph for this function:



#### 7.15.4.11 unsigned int VME::TDCEvent::GetTDCId ( ) const [inline]

Programmed identifier of master TDC providing the event.

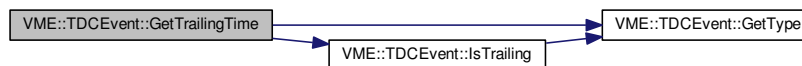
Here is the call graph for this function:



#### 7.15.4.12 uint32\_t VME::TDCEvent::GetTrailingTime ( ) const [inline]

Trailing edge measurement in programmed time resolution.

Here is the call graph for this function:



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

Type of packet read out from the TDC.

#### 7.15.4.14 unsigned int VME::TDCEvent::GetWidth ( ) const [inline]

Width of pulse in programmed time resolution.

Here is the call graph for this function:



#### 7.15.4.15 uint32\_t VME::TDCEvent::GetWord ( ) const [inline]

#### 7.15.4.16 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:



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

Are we dealing with a trailing or a leading measurement?

Here is the call graph for this function:



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

## 7.15.5 Field Documentation

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

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

- include/VME\_TDCEvent.h

## 7.16 VME::TDCMeasurement Class Reference

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

### Public Types

- enum [Type](#) {  
[GlobalHeader](#) = 0x0, [GlobalTrailer](#) = 0x1, [TDCHeader](#) = 0x2, [TDCTrailer](#) = 0x3,  
[LeadingEdge](#) = 0x4, [TrailingEdge](#) = 0x5, [ETTT](#) = 0x6 }

### Public Member Functions

- [TDCMeasurement](#) ()
- [TDCMeasurement](#) (const std::vector< [TDCEvent](#) > &v)
- [~TDCMeasurement](#) ()



- void [Dump](#) ()
- void [SetEventsCollection](#) (const std::vector< [TDCEvent](#) > &v)
- uint32\_t [GetLeadingTime](#) ()
- uint32\_t [GetTrailingTime](#) ()
- uint16\_t [GetChannelId](#) ()
- uint16\_t [GetTDCId](#) ()
- uint16\_t [GetEventId](#) ()
- uint16\_t [GetBunchId](#) ()

### Private Attributes

- std::map< [Type](#), [TDCEvent](#) > fMap

## 7.16.1 Member Enumeration Documentation

### 7.16.1.1 enum VME::TDCMeasurement::Type

Enumerator

***GlobalHeader***

***GlobalTrailer***

***TDCHeader***

***TDCTrailer***

***LeadingEdge***

***TrailingEdge***

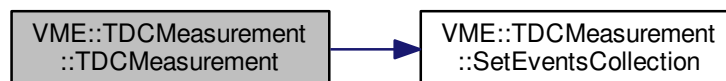
***ETTT***

## 7.16.2 Constructor & Destructor Documentation

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

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

Here is the call graph for this function:

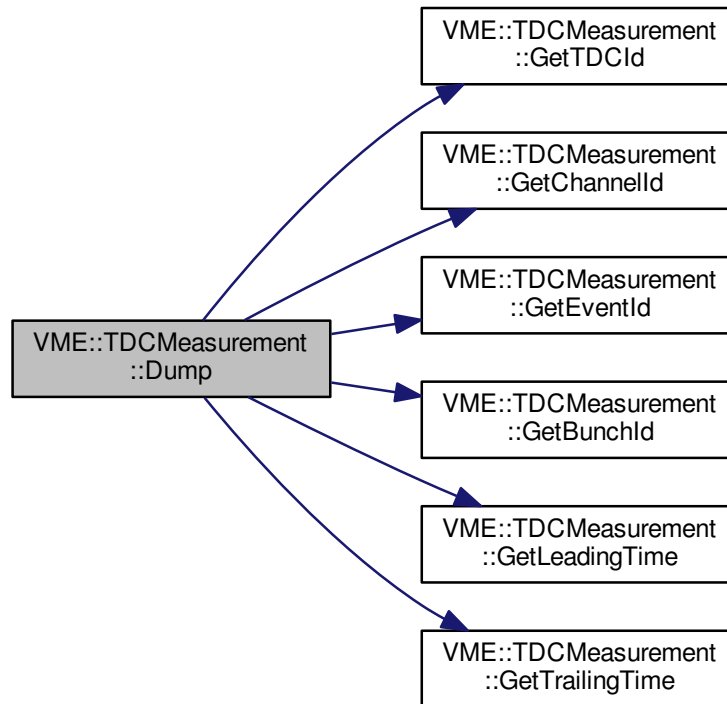


7.16.2.3 VME::TDCMeasurement::~~TDCMeasurement ( ) `[inline]`

## 7.16.3 Member Function Documentation

### 7.16.3.1 void VME::TDCMeasurement::Dump ( ) [inline]

Here is the call graph for this function:



### 7.16.3.2 uint16\_t VME::TDCMeasurement::GetBunchId ( ) [inline]

### 7.16.3.3 uint16\_t VME::TDCMeasurement::GetChannelId ( ) [inline]

### 7.16.3.4 uint16\_t VME::TDCMeasurement::GetEventId ( ) [inline]

### 7.16.3.5 uint32\_t VME::TDCMeasurement::GetLeadingTime ( ) [inline]

### 7.16.3.6 uint16\_t VME::TDCMeasurement::GetTDCId ( ) [inline]

### 7.16.3.7 uint32\_t VME::TDCMeasurement::GetTrailingTime ( ) [inline]

### 7.16.3.8 void VME::TDCMeasurement::SetEventsCollection ( const std::vector< TDCEvent > & v ) [inline]

## 7.16.4 Field Documentation

### 7.16.4.1 std::map<Type,TDCEvent> VME::TDCMeasurement::fMap [private]

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

- include/VME\_TDCMeasurement.h

## 7.17 VME::TDCV1x90 Class Reference

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

### Public Types

- enum [DLLMode](#) { [DLL\\_Direct\\_LowRes](#) = 0x0, [DLL\\_PLL\\_LowRes](#) = 0x1, [DLL\\_PLL\\_MedRes](#) = 0x2, [DLL\\_PLL\\_HighRes](#) = 0x3 }

### Public Member Functions

- [TDCV1x90](#) (int32\_t, uint32\_t, const [AcquisitionMode](#) &acqm=TRIG\_MATCH, const [DetectionMode](#) &detm=TRAILEAD)
- [~TDCV1x90](#) ()
- void [SetVerboseLevel](#) (unsigned short verb=1)
- void [SetTestMode](#) (bool en=true) const
- bool [GetTestMode](#) () const
- uint32\_t [GetModel](#) () const
- uint32\_t [GetOUI](#) () const
- uint32\_t [GetSerialNumber](#) () const
- void [GetFirmwareRevision](#) () const
- void [CheckConfiguration](#) () const
- void [EnableChannel](#) (short) const
- void [DisableChannel](#) (short) const
- void [SetPol](#) (uint16\_t word1, uint16\_t word2) const
- std::map< unsigned short, bool > [GetPol](#) () const
- void [SetLSBTraileadEdge](#) (trailead\_edge\_lsb) const
- void [SetAcquisitionMode](#) (const [AcquisitionMode](#) &)
- [AcquisitionMode](#) [GetAcquisitionMode](#) ()
- void [SetTriggerMatching](#) ()
- void [SetContinuousStorage](#) ()
- void [SetDetectionMode](#) (const [DetectionMode](#) &detm)
- [DetectionMode](#) [GetDetectionMode](#) ()
- void [SetDLLClock](#) (const [DLLMode](#) &dll) const
- [DLLMode](#) [GetDLLClock](#) () const
- void [SetGlobalOffset](#) (const [GlobalOffset](#) &) const
- [GlobalOffset](#) [GetGlobalOffset](#) () const
- void [SetRCAdjust](#) (int, uint16\_t) const
- uint16\_t [GetRCAdjust](#) (int) const
- uint32\_t [GetEventCounter](#) () const
- Number of occurred triggers.*
- uint16\_t [GetEventStored](#) () const
- Number of events currently stored in the output buffer.*
- void [SetTDCEncapsulation](#) (bool) const
- bool [GetTDCEncapsulation](#) () const
- void [SetErrorMarks](#) (bool mode=true)
- bool [GetErrorMarks](#) () const
- void [SetPairModeResolution](#) (int, int) const
- uint16\_t [GetResolution](#) () const
- void [SetBLTEventNumberRegister](#) (const uint16\_t &) const
- uint16\_t [GetBLTEventNumberRegister](#) () const
- void [SetWindowWidth](#) (const uint16\_t &)
- uint16\_t [GetWindowWidth](#) () const

- void [SetWindowOffset](#) (const int16\_t &) const
- int16\_t [GetWindowOffset](#) () const
- uint16\_t [GetTriggerConfiguration](#) (const [trig\\_conf](#) &) const
- bool [SoftwareClear](#) () const
- bool [SoftwareReset](#) () const
- bool [HardwareReset](#) () const
- void [SetETTT](#) (bool ettt=true) const
- bool [GetETTT](#) () const
- void [SetStatus](#) (const [TDCV1x90Status](#) &) const
- [TDCV1x90Status](#) [GetStatus](#) () const
- void [SetControl](#) (const [TDCV1x90Control](#) &) const
- [TDCV1x90Control](#) [GetControl](#) () const
- [TDCEventCollection](#) [FetchEvents](#) ()
- void [SetChannelDeadTime](#) (unsigned short dt) const
- unsigned short [GetChannelDeadTime](#) () const
- void [SetFIFOSize](#) (const uint16\_t &) const
- uint16\_t [GetFIFOSize](#) () const
- void [abort](#) ()

### Private Types

- enum [mod\\_reg](#) {  
[kOutputBuffer](#) = 0x0000, [kControl](#) = 0x1000, [kStatus](#) = 0x1002, [kInterruptLevel](#) = 0x100a,  
[kInterruptVector](#) = 0x100c, [kGeoAddress](#) = 0x100e, [kMCSTBase](#) = 0x1010, [kMCSTControl](#) = 0x1012,  
[kModuleReset](#) = 0x1014, [kSoftwareClear](#) = 0x1016, [kEventCounter](#) = 0x101c, [kEventStored](#) = 0x1020,  
[kBLTEventNumber](#) = 0x1024, [kFirmwareRev](#) = 0x1026, [kMicro](#) = 0x102e, [kMicroHandshake](#) = 0x1030,  
[kEventFIFO](#) = 0x1038, [kEventFIFOStoredRegister](#) = 0x103c, [kEventFIFOStatusRegister](#) = 0x103e, [kROM](#)↵  
[Oui2](#) = 0x4024,  
[kROMOui1](#) = 0x4028, [kROMOui0](#) = 0x402c, [kROMBoard2](#) = 0x4034, [kROMBoard1](#) = 0x4038,  
[kROMBoard0](#) = 0x403c, [kROMRevis3](#) = 0x4040, [kROMRevis2](#) = 0x4044, [kROMRevis1](#) = 0x4048,  
[kROMRevis0](#) = 0x404c, [kROMSerNum1](#) = 0x4080, [kROMSerNum0](#) = 0x4084 }

### Private Member Functions

- bool [WaitMicro](#) ([micro\\_handshake](#) mode) const
- void [WriteRegister](#) ([mod\\_reg](#), const uint16\_t &) const  
*Write on register.*
- void [WriteRegister](#) ([mod\\_reg](#), const uint32\_t &) const  
*Write on register.*
- void [ReadRegister](#) ([mod\\_reg](#), uint16\_t \*) const  
*Read on register.*
- void [ReadRegister](#) ([mod\\_reg](#), uint32\_t \*) const  
*Read on register.*
- void [ReadAcquisitionMode](#) ()
- void [ReadDetectionMode](#) ()

### Private Attributes

- uint32\_t [fBaseAddr](#)
- int32\_t [fHandle](#)
- unsigned short [fVerb](#)
- [AcquisitionMode](#) [fAcquisitionMode](#)
- [DetectionMode](#) [fDetectionMode](#)

- bool [fErrorMarks](#)
- uint16\_t [fWindowWidth](#)
- CVAddressModifier [am](#)
- CVAddressModifier [am\\_blt](#)
- uint32\_t \* [fBuffer](#)
- uint32\_t [nchannels](#)
- bool [gEnd](#)
- std::string [pair\\_lead\\_res](#) [8]
- std::string [pair\\_width\\_res](#) [16]
- std::string [trailead\\_edge\\_res](#) [4]

### 7.17.1 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 (NA62-Gigatracker)  
 May 2015 (CMS-TOTEM PPS)

### 7.17.2 Member Enumeration Documentation

#### 7.17.2.1 enum VME::TDCV1x90::DLLMode

##### Enumerator

***DLL\_Direct\_LowRes***  
***DLL\_PLL\_LowRes***  
***DLL\_PLL\_MedRes***  
***DLL\_PLL\_HighRes***

#### 7.17.2.2 enum VME::TDCV1x90::mod\_reg [private]

##### Enumerator

***kOutputBuffer***  
***kControl***  
***kStatus***  
***kInterruptLevel***  
***kInterruptVector***  
***kGeoAddress***  
***kMCSTBase***  
***kMCSTControl***  
***kModuleReset***  
***kSoftwareClear***  
***kEventCounter***  
***kEventStored***  
***kBLTEventNumber***



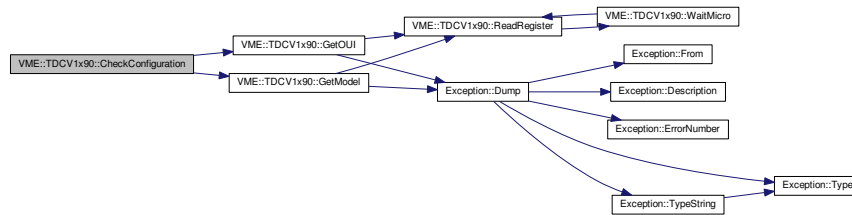
## 7.17.3.2 VME::TDCV1x90::~~TDCV1x90 ( )

## 7.17.4 Member Function Documentation

## 7.17.4.1 void VME::TDCV1x90::abort ( )

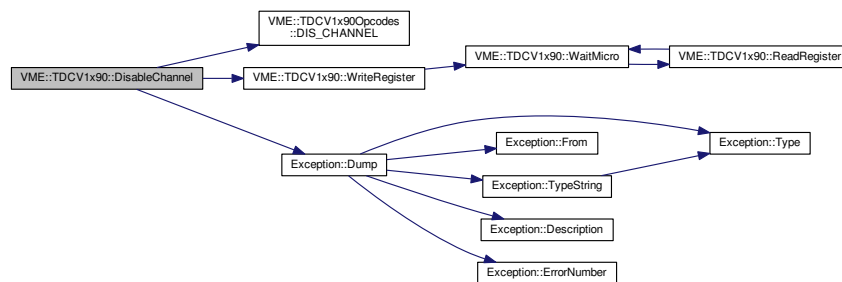
## 7.17.4.2 void VME::TDCV1x90::CheckConfiguration ( ) const

Here is the call graph for this function:



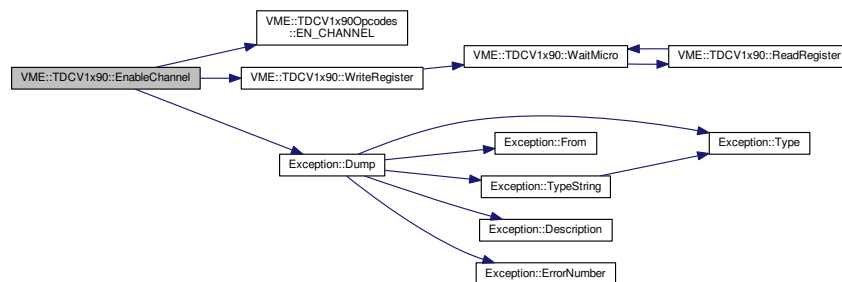
## 7.17.4.3 void VME::TDCV1x90::DisableChannel ( short channel\_id ) const

Here is the call graph for this function:



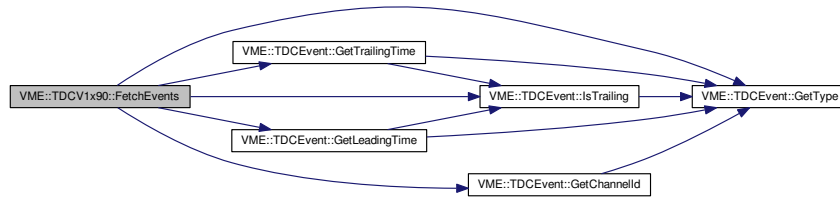
## 7.17.4.4 void VME::TDCV1x90::EnableChannel ( short channel\_id ) const

Here is the call graph for this function:



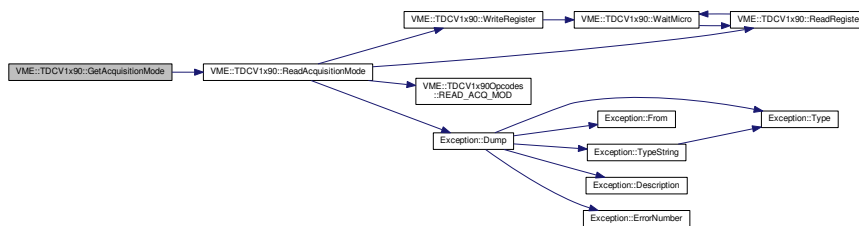
#### 7.17.4.5 TDCEventCollection VME::TDCV1x90::FetchEvents ( )

Here is the call graph for this function:



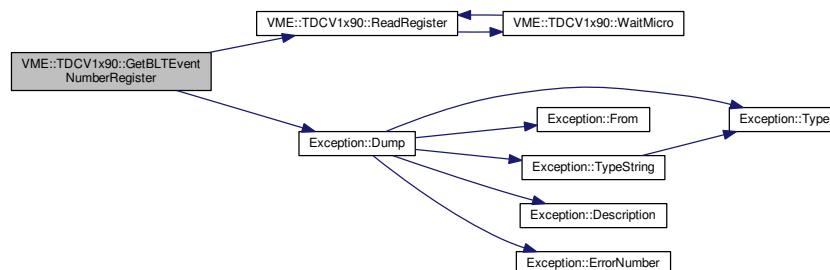
#### 7.17.4.6 AcquisitionMode VME::TDCV1x90::GetAcquisitionMode ( ) [inline]

Here is the call graph for this function:



#### 7.17.4.7 uint16\_t VME::TDCV1x90::GetBLTEventNumberRegister ( ) const

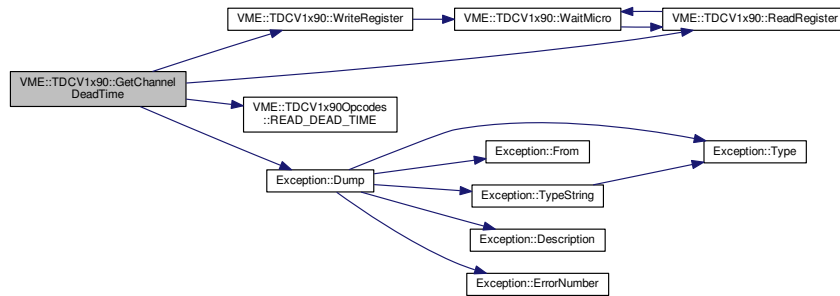
Here is the call graph for this function:





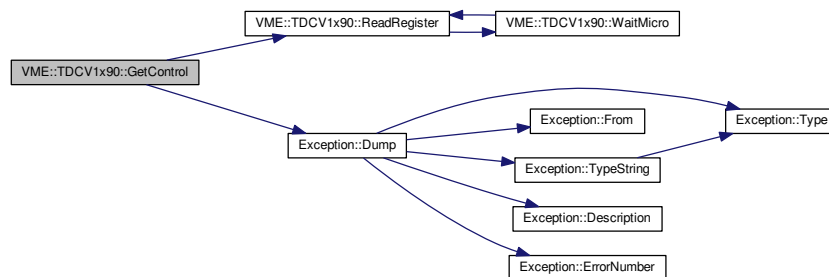
## 7.17.4.8 unsigned short VME::TDCV1x90::GetChannelDeadTime ( ) const

Here is the call graph for this function:



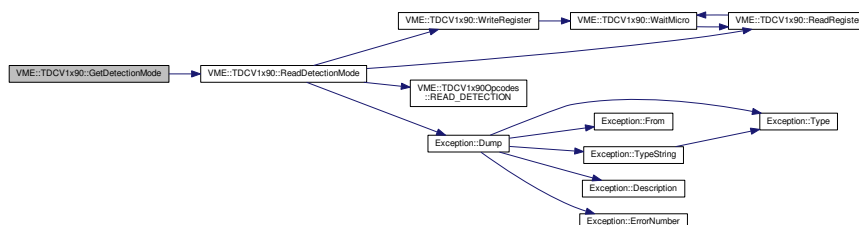
## 7.17.4.9 TDCV1x90Control VME::TDCV1x90::GetControl ( ) const

Here is the call graph for this function:



## 7.17.4.10 DetectionMode VME::TDCV1x90::GetDetectionMode ( ) [inline]

Here is the call graph for this function:

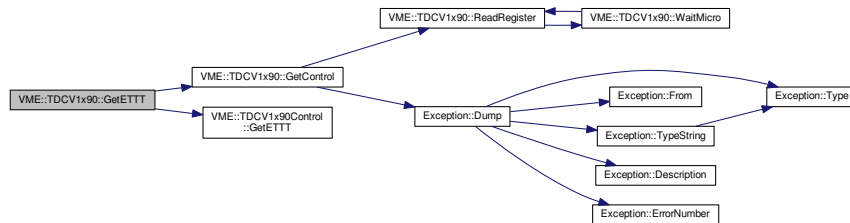


## 7.17.4.11 DLLMode VME::TDCV1x90::GetDLLClock ( ) const

7.17.4.12 `bool VME::TDCV1x90::GetErrorMarks ( ) const [inline]`

7.17.4.13 `bool VME::TDCV1x90::GetETTT ( ) const [inline]`

Here is the call graph for this function:

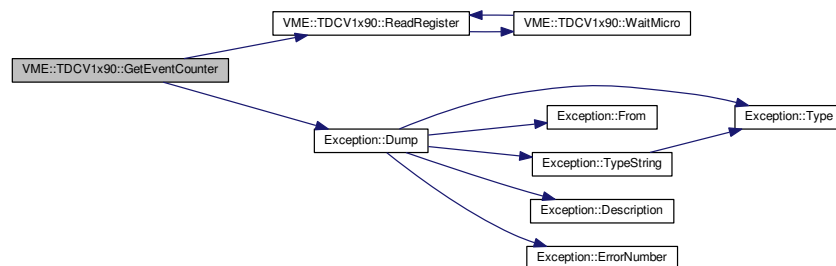


7.17.4.14 `uint32_t VME::TDCV1x90::GetEventCounter ( ) const`

Number of occurred triggers.

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

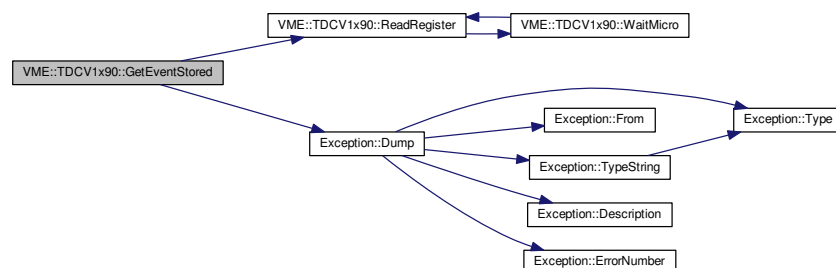
Here is the call graph for this function:



7.17.4.15 `uint16_t VME::TDCV1x90::GetEventStored ( ) const`

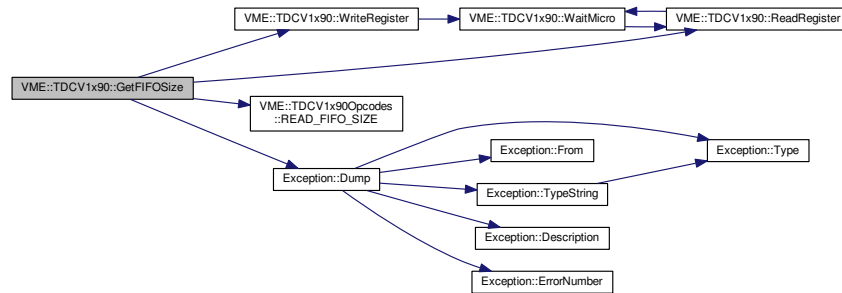
Number of events currently stored in the output buffer.

Here is the call graph for this function:



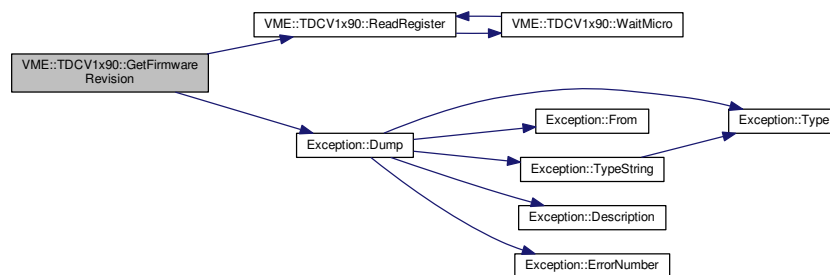
## 7.17.4.16 uint16\_t VME::TDCV1x90::GetFIFOSize ( ) const

Here is the call graph for this function:



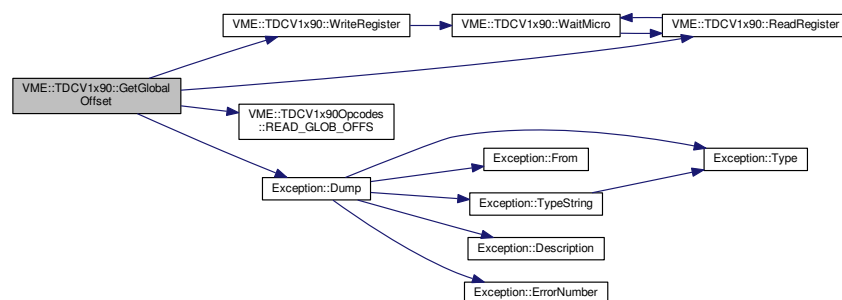
## 7.17.4.17 void VME::TDCV1x90::GetFirmwareRevision ( ) const

Here is the call graph for this function:



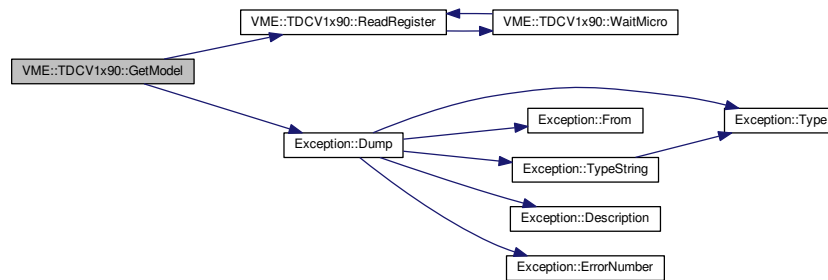
## 7.17.4.18 GlobalOffset VME::TDCV1x90::GetGlobalOffset ( ) const

Here is the call graph for this function:



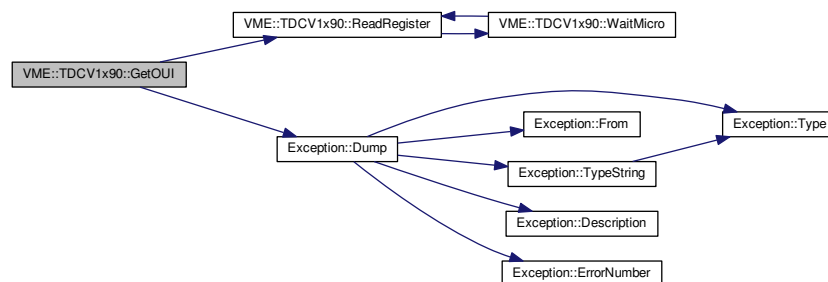
#### 7.17.4.19 uint32\_t VME::TDCV1x90::GetModel ( ) const

Here is the call graph for this function:



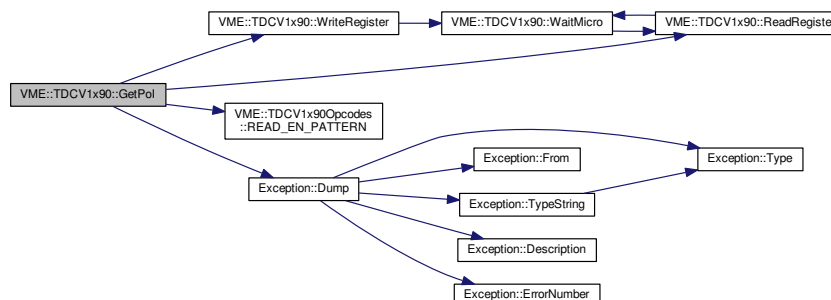
#### 7.17.4.20 uint32\_t VME::TDCV1x90::GetOUI ( ) const

Here is the call graph for this function:



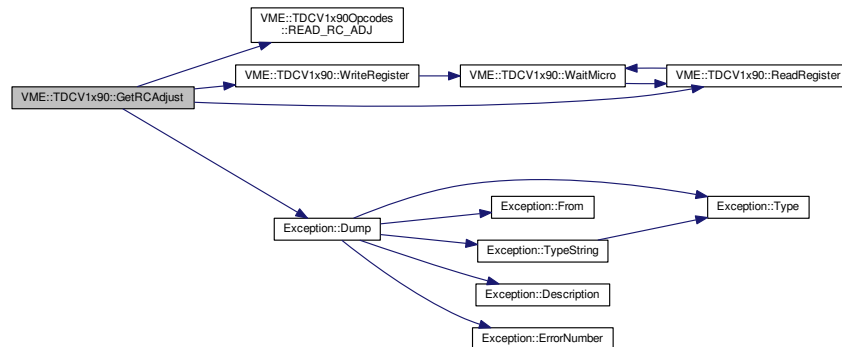
#### 7.17.4.21 std::map< unsigned short, bool > VME::TDCV1x90::GetPol ( ) const

Here is the call graph for this function:



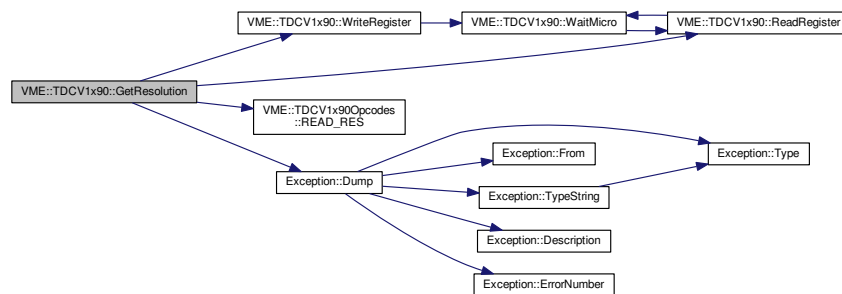
7.17.4.22 uint16\_t VME::TDCV1x90::GetRCAdjust ( int *tdc* ) const

Here is the call graph for this function:



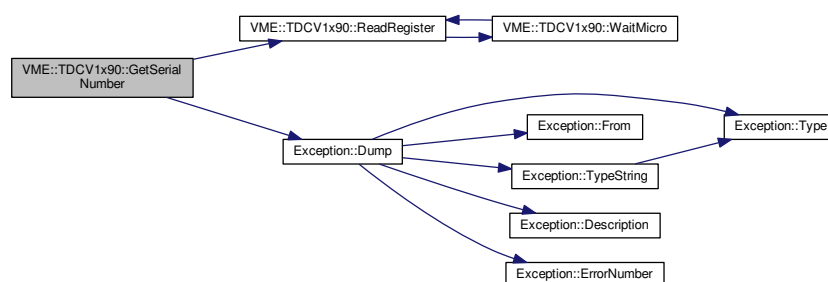
## 7.17.4.23 uint16\_t VME::TDCV1x90::GetResolution ( ) const

Here is the call graph for this function:



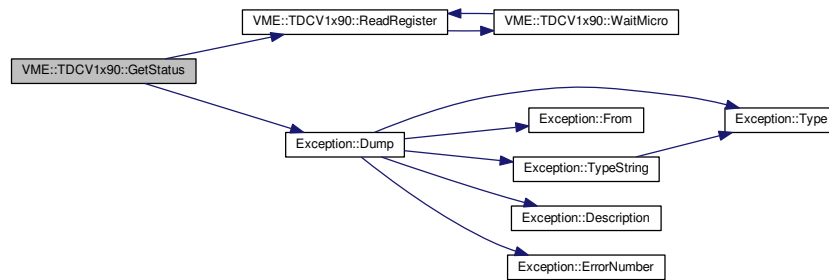
## 7.17.4.24 uint32\_t VME::TDCV1x90::GetSerialNumber ( ) const

Here is the call graph for this function:



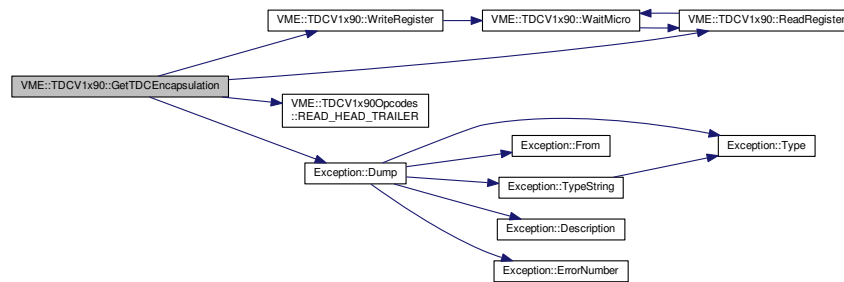
#### 7.17.4.25 TDCV1x90Status VME::TDCV1x90::GetStatus ( ) const

Here is the call graph for this function:



#### 7.17.4.26 bool VME::TDCV1x90::GetTDCEncapsulation ( ) const

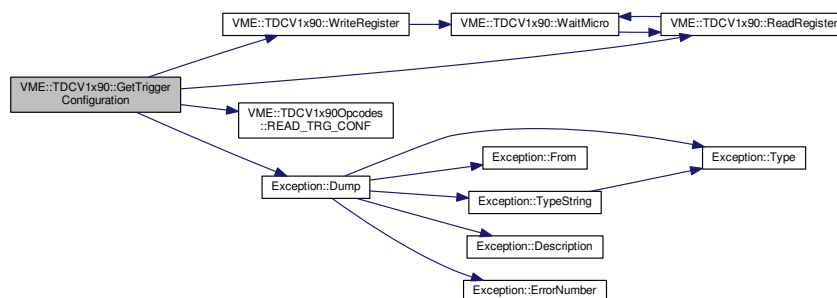
Here is the call graph for this function:



#### 7.17.4.27 bool VME::TDCV1x90::GetTestMode ( ) const

#### 7.17.4.28 uint16\_t VME::TDCV1x90::GetTriggerConfiguration ( const trig\_conf & type ) const

Here is the call graph for this function:



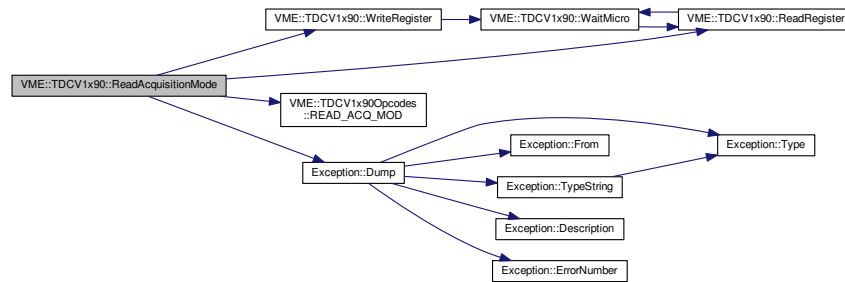
7.17.4.29 `int16_t VME::TDCV1x90::GetWindowOffset ( ) const`

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

7.17.4.31 `bool VME::TDCV1x90::HardwareReset ( ) const`

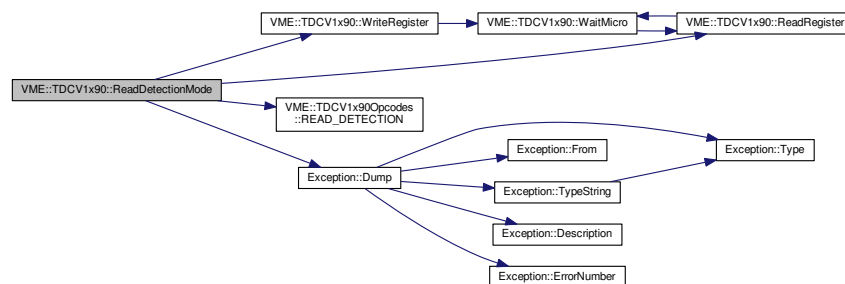
7.17.4.32 `void VME::TDCV1x90::ReadAcquisitionMode ( ) [private]`

Here is the call graph for this function:



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

Here is the call graph for this function:



7.17.4.34 `void VME::TDCV1x90::ReadRegister ( mod_reg addr, uint16_t * data ) const [private]`

Read on register.

Read a 16-bit word in the register

Parameters

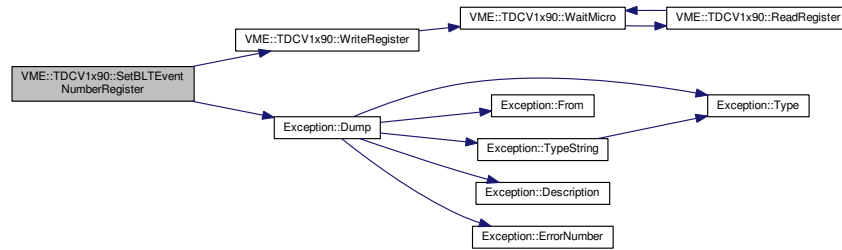
in	<i>addr</i>	register
out	<i>data</i>	word





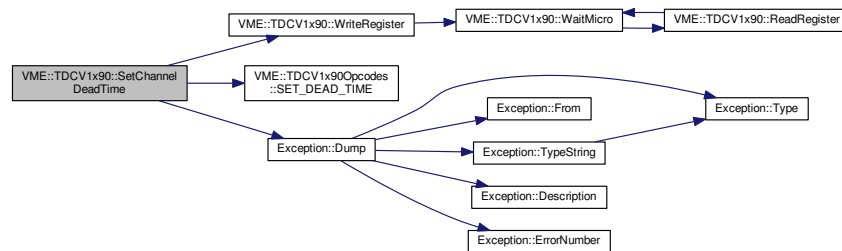
#### 7.17.4.37 void VME::TDCV1x90::SetBLTEventNumberRegister ( const uint16\_t & value ) const

Here is the call graph for this function:



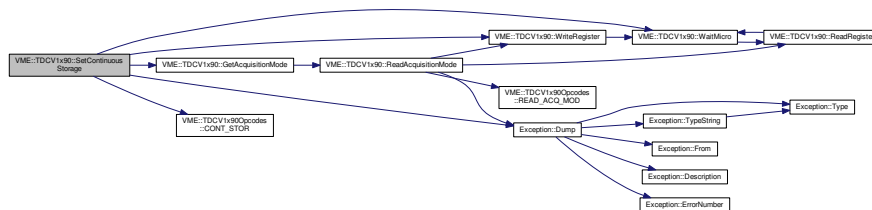
#### 7.17.4.38 void VME::TDCV1x90::SetChannelDeadTime ( unsigned short dt ) const

Here is the call graph for this function:



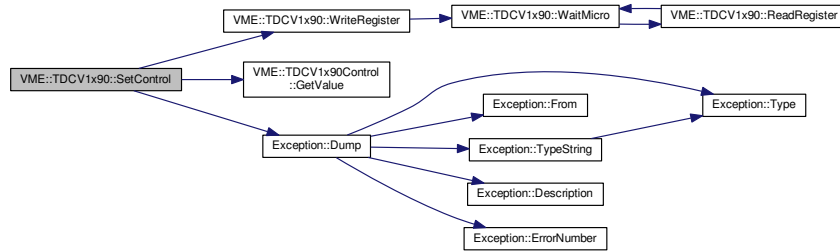
#### 7.17.4.39 void VME::TDCV1x90::SetContinuousStorage ( )

Here is the call graph for this function:



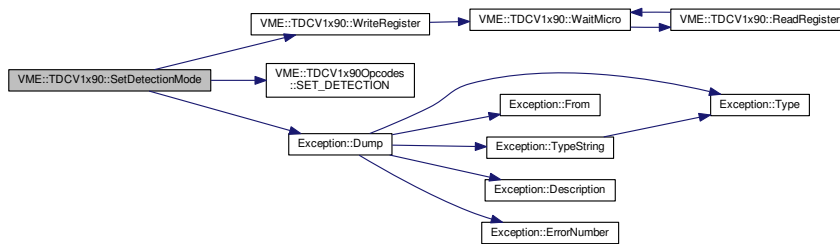
7.17.4.40 void VME::TDCV1x90::SetControl ( const TDCV1x90Control & *control* ) const

Here is the call graph for this function:



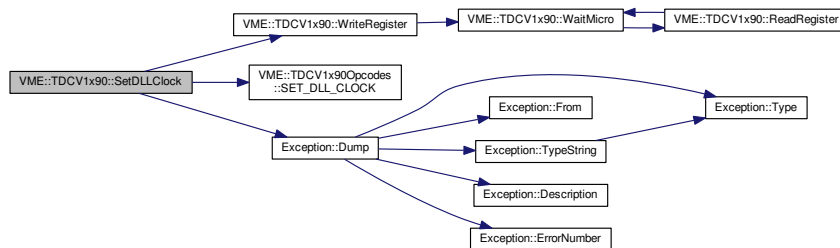
7.17.4.41 void VME::TDCV1x90::SetDetectionMode ( const DetectionMode & *detm* )

Here is the call graph for this function:



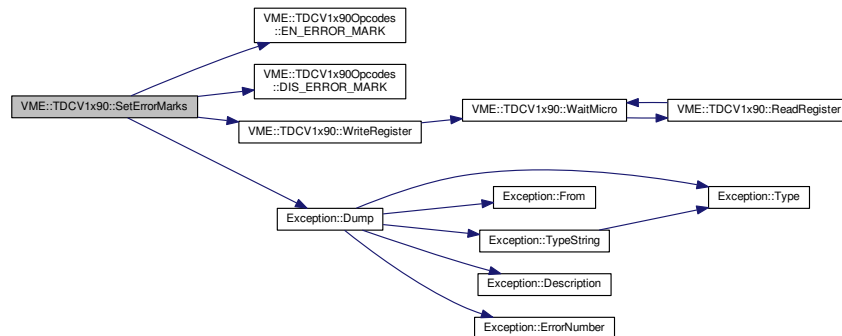
7.17.4.42 void VME::TDCV1x90::SetDLLClock ( const DLLMode & *dll* ) const

Here is the call graph for this function:

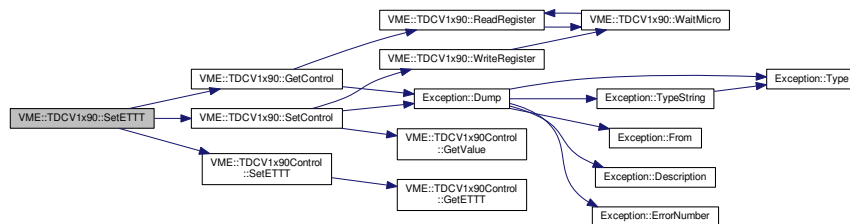


7.17.4.43 void VME::TDCV1x90::SetErrorMarks ( bool *mode* = true )

Here is the call graph for this function:

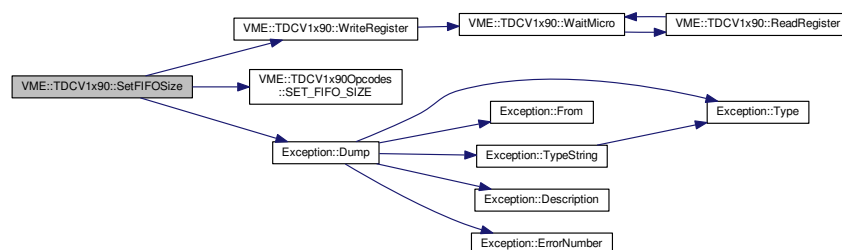
7.17.4.44 void VME::TDCV1x90::SetETTT ( bool *ettt* = true ) const [inline]

Here is the call graph for this function:



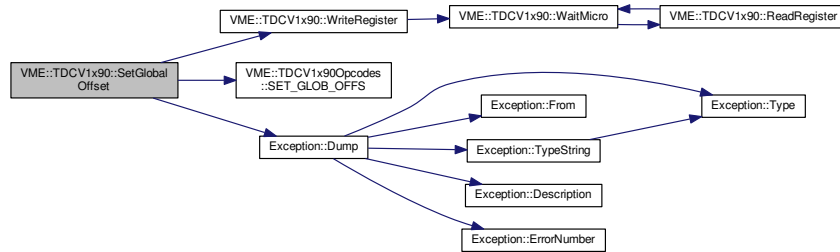
## 7.17.4.45 void VME::TDCV1x90::SetFIFOSize ( const uint16\_t &amp;size ) const

Here is the call graph for this function:



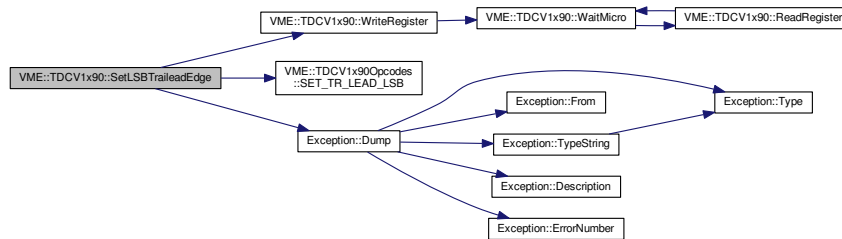
#### 7.17.4.46 void VME::TDCV1x90::SetGlobalOffset ( const GlobalOffset & offs ) const

Here is the call graph for this function:



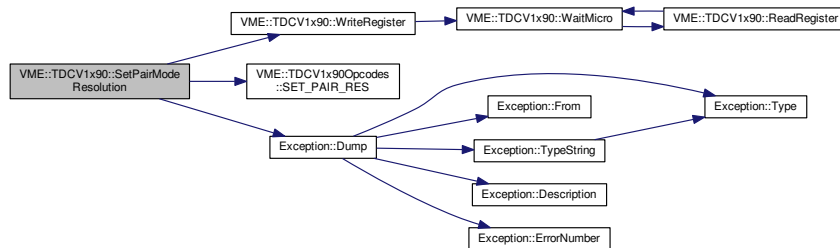
#### 7.17.4.47 void VME::TDCV1x90::SetLSBTrailEdge ( trailEdge\_lsb conf ) const

Here is the call graph for this function:



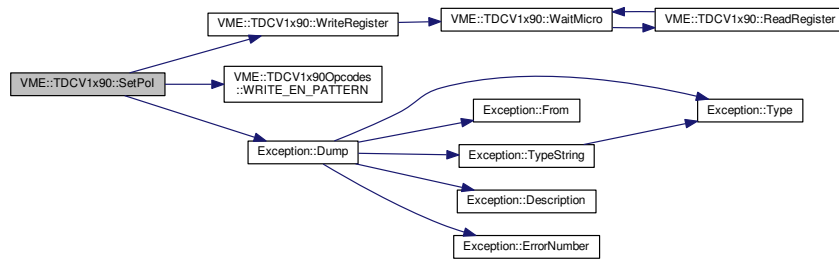
#### 7.17.4.48 void VME::TDCV1x90::SetPairModeResolution ( int lead\_time\_res, int pulse\_width\_res ) const

Here is the call graph for this function:



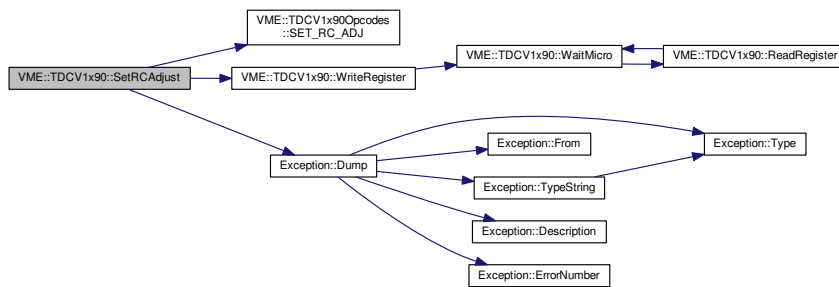
## 7.17.4.49 void VME::TDCV1x90::SetPol ( uint16\_t word1, uint16\_t word2 ) const

Here is the call graph for this function:



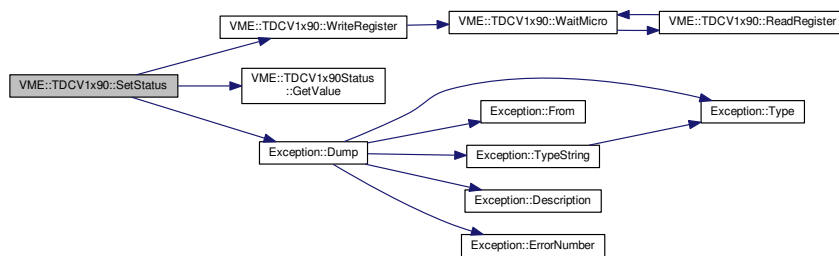
## 7.17.4.50 void VME::TDCV1x90::SetRCAdjust ( int tdc, uint16\_t value ) const

Here is the call graph for this function:



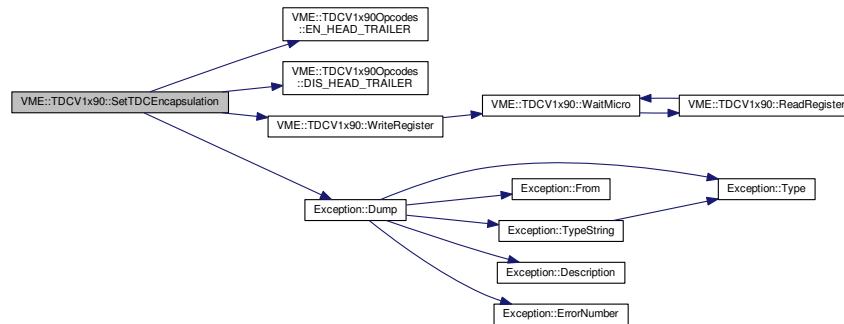
## 7.17.4.51 void VME::TDCV1x90::SetStatus ( const TDCV1x90Status &amp; status ) const

Here is the call graph for this function:



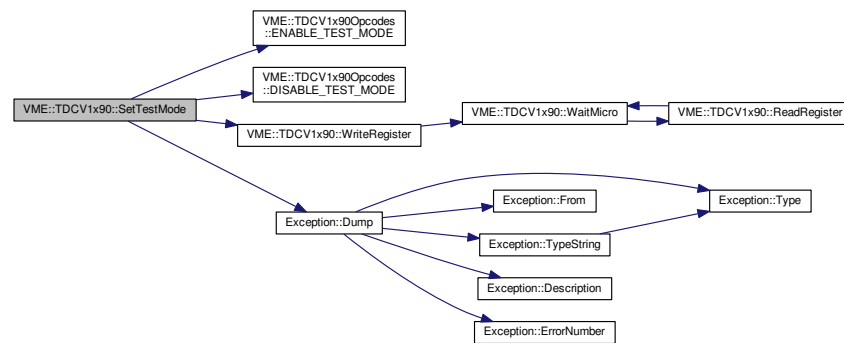
#### 7.17.4.52 void VME::TDCV1x90::SetTDCEncapsulation ( bool mode ) const

Here is the call graph for this function:



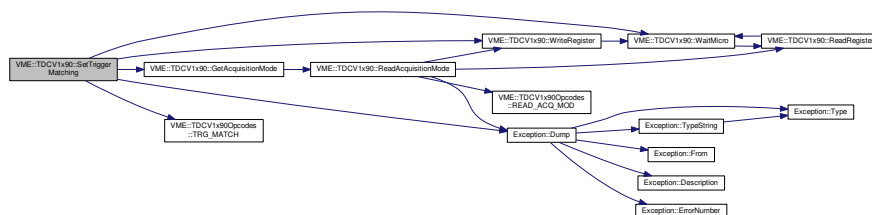
#### 7.17.4.53 void VME::TDCV1x90::SetTestMode ( bool en = true ) const

Here is the call graph for this function:



#### 7.17.4.54 void VME::TDCV1x90::SetTriggerMatching ( )

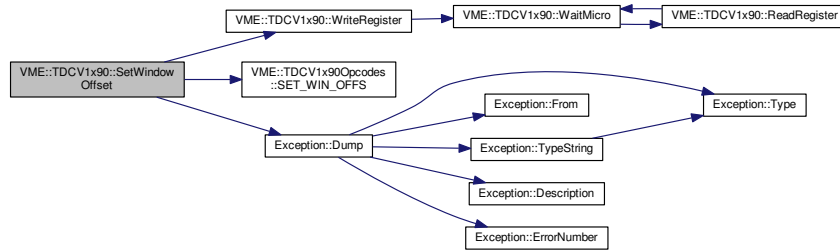
Here is the call graph for this function:



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

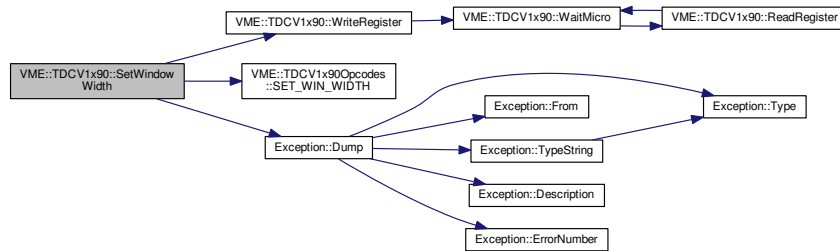
7.17.4.56 void VME::TDCV1x90::SetWindowOffset ( const int16\_t & *offs* ) const

Here is the call graph for this function:



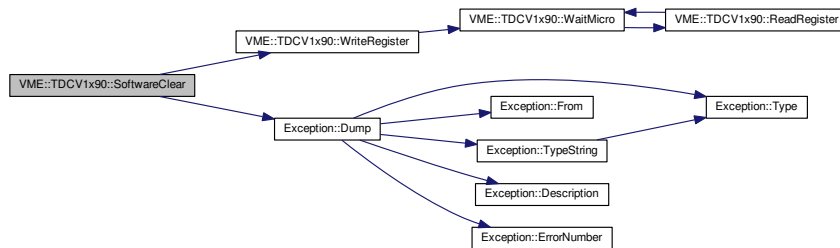
7.17.4.57 void VME::TDCV1x90::SetWindowWidth ( const uint16\_t & *width* )

Here is the call graph for this function:



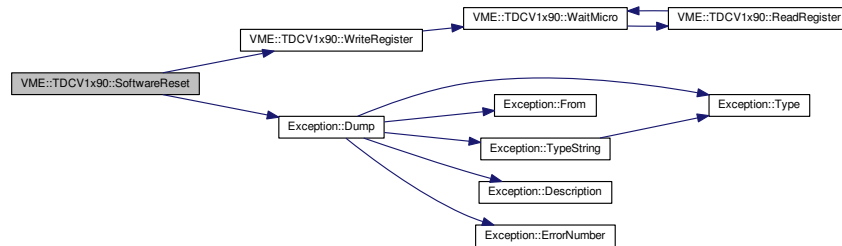
7.17.4.58 bool VME::TDCV1x90::SoftwareClear ( ) const

Here is the call graph for this function:



#### 7.17.4.59 bool VME::TDCV1x90::SoftwareReset ( ) const

Here is the call graph for this function:



#### 7.17.4.60 bool VME::TDCV1x90::WaitMicro ( micro\_handshake mode ) const [private]

Here is the call graph for this function:



#### 7.17.4.61 void VME::TDCV1x90::WriteRegister ( mod\_reg addr, const uint16\_t & data ) const [private]

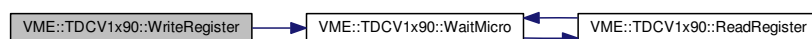
Write on register.

Write a 16-bit word in the register

##### Parameters

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

Here is the call graph for this function:



#### 7.17.4.62 void VME::TDCV1x90::WriteRegister ( mod\_reg addr, const uint32\_t & data ) const [private]

Write on register.

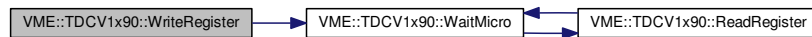
Write a 32-bit word in the register



## Parameters

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

Here is the call graph for this function:



## 7.17.5 Field Documentation

- 7.17.5.1 CVAddressModifier VME::TDCV1x90::am [private]
- 7.17.5.2 CVAddressModifier VME::TDCV1x90::am\_blt [private]
- 7.17.5.3 AcquisitionMode VME::TDCV1x90::fAcquisitionMode [private]
- 7.17.5.4 uint32\_t VME::TDCV1x90::fBaseAddr [private]
- 7.17.5.5 uint32\_t\* VME::TDCV1x90::fBuffer [private]
- 7.17.5.6 DetectionMode VME::TDCV1x90::fDetectionMode [private]
- 7.17.5.7 bool VME::TDCV1x90::fErrorMarks [private]
- 7.17.5.8 int32\_t VME::TDCV1x90::fHandle [private]
- 7.17.5.9 unsigned short VME::TDCV1x90::fVerb [private]
- 7.17.5.10 uint16\_t VME::TDCV1x90::fWindowWidth [private]
- 7.17.5.11 bool VME::TDCV1x90::gEnd [private]
- 7.17.5.12 uint32\_t VME::TDCV1x90::nchannels [private]
- 7.17.5.13 std::string VME::TDCV1x90::pair\_lead\_res[8] [private]
- 7.17.5.14 std::string VME::TDCV1x90::pair\_width\_res[16] [private]
- 7.17.5.15 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

## 7.18 VME::TDCV1x90Control Class Reference

TDC control register.

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

## Public Member Functions

- [TDCV1x90Control](#) (const uint16\_t &word)
- virtual [~TDCV1x90Control](#) ()
- void [Dump](#) () const
- uint16\_t [GetValue](#) () const
- bool [GetBusError](#) () const
- void [SetBusError](#) (bool sw)
- bool [GetTermination](#) () const
- void [SetTermination](#) (bool sw)
- bool [GetSWTermination](#) () const
- void [SetSWTermination](#) (bool sw)
- bool [GetEmptyEvent](#) () const
- void [SetEmptyEvent](#) (bool sw)
- bool [GetAlign64](#) () const
- void [SetAlign64](#) (bool sw)
- bool [GetCompensation](#) () const
- void [SetCompensation](#) (bool sw)
- bool [GetTestFIFO](#) () const
- void [SetTestFIFO](#) (bool sw)
- bool [GetSRAMCompensation](#) () const
- void [SetSRAMCompensation](#) (bool sw)
- bool [GetEventFIFO](#) () const
- void [SetEventFIFO](#) (bool sw)
- bool [GetETTT](#) () const
- void [SetETTT](#) (bool sw)
- bool [GetMEBAccess](#) () const
- void [SetMEBAccess](#) (bool sw)

## Private Attributes

- uint16\_t [fWord](#)

### 7.18.1 Detailed Description

TDC control register.

#### Author

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

#### Date

Jun 2015

### 7.18.2 Constructor & Destructor Documentation

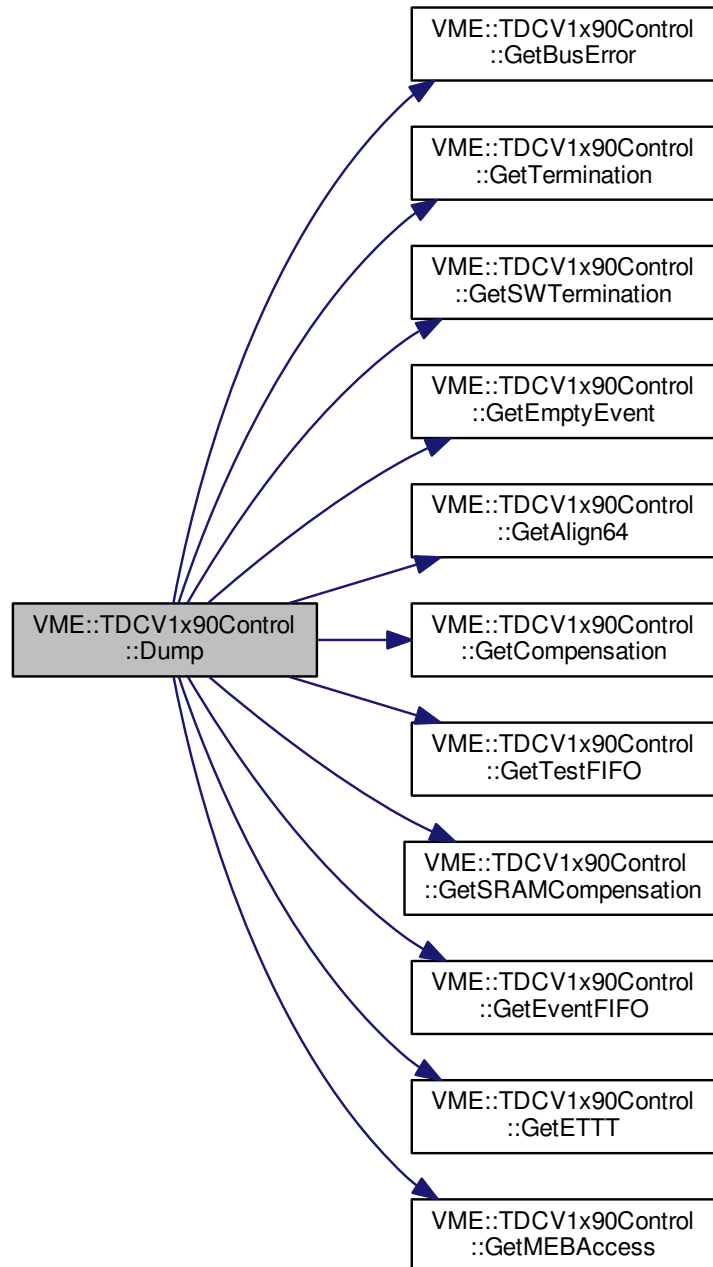
7.18.2.1 `VME::TDCV1x90Control::TDCV1x90Control ( const uint16_t & word ) [inline]`

7.18.2.2 `virtual VME::TDCV1x90Control::~~TDCV1x90Control ( ) [inline],[virtual]`

### 7.18.3 Member Function Documentation

### 7.18.3.1 void VME::TDCV1x90Control::Dump ( ) const [inline]

Here is the call graph for this function:



### 7.18.3.2 bool VME::TDCV1x90Control::GetAlign64 ( ) const [inline]

### 7.18.3.3 bool VME::TDCV1x90Control::GetBusError ( ) const [inline]

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

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

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

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

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

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

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

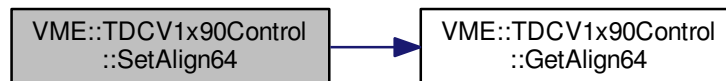
7.18.3.11 `bool VME::TDCV1x90Control::GetTermination ( ) const [inline]`

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

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

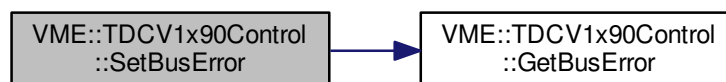
7.18.3.14 `void VME::TDCV1x90Control::SetAlign64 ( bool sw ) [inline]`

Here is the call graph for this function:



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

Here is the call graph for this function:



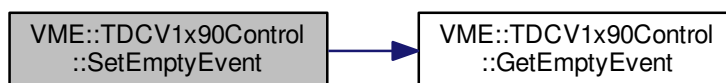
7.18.3.16 void VME::TDCV1x90Control::SetCompensation ( bool *sw* ) [inline]

Here is the call graph for this function:



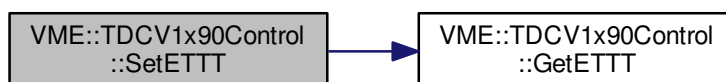
7.18.3.17 void VME::TDCV1x90Control::SetEmptyEvent ( bool *sw* ) [inline]

Here is the call graph for this function:



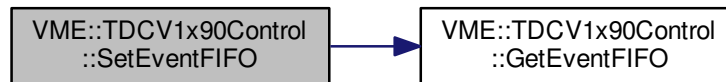
7.18.3.18 void VME::TDCV1x90Control::SetETTT ( bool *sw* ) [inline]

Here is the call graph for this function:



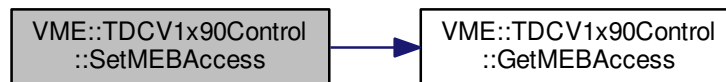
7.18.3.19 void VME::TDCV1x90Control::SetEventFIFO ( bool *sw* ) [inline]

Here is the call graph for this function:



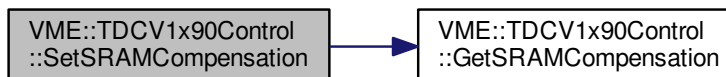
7.18.3.20 void VME::TDCV1x90Control::SetMEBAccess ( bool *sw* ) [inline]

Here is the call graph for this function:



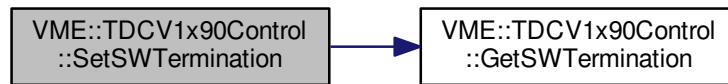
7.18.3.21 void VME::TDCV1x90Control::SetSRAMCompensation ( bool *sw* ) [inline]

Here is the call graph for this function:



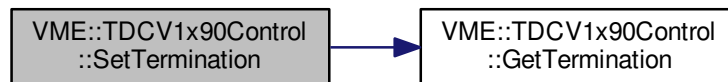
7.18.3.22 `void VME::TDCV1x90Control::SetSWTermination ( bool sw ) [inline]`

Here is the call graph for this function:



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

Here is the call graph for this function:



7.18.3.24 `void VME::TDCV1x90Control::SetTestFIFO ( bool sw ) [inline]`

Here is the call graph for this function:



## 7.18.4 Field Documentation

7.18.4.1 `uint16_t VME::TDCV1x90Control::fWord [private]`

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

- `include/VME_TDCV1x90.h`

## 7.19 VME::TDCV1x90Status Class Reference

TDC status register.

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

### Public Types

- enum [TDCResolution](#) { [R\\_800ps](#) = 0x0, [R\\_200ps](#) = 0x1, [R\\_100ps](#) = 0x2, [R\\_25ps](#) = 0x3 }

### Public Member Functions

- [TDCV1x90Status](#) (const uint16\_t &word)
- virtual [~TDCV1x90Status](#) ()
- void [Dump](#) () const
- uint16\_t [GetValue](#) () const
- bool [DataReady](#) () const
- bool [AlmostFull](#) () const
- bool [Full](#) () const
- bool [TriggerMatching](#) () const
- bool [HeadersEnabled](#) () const
- bool [TerminationOn](#) () const
- bool [Error](#) (const unsigned int &id) const
- bool [Error](#) () const
- bool [BusError](#) () const
- bool [Purged](#) () const
- [TDCResolution Resolution](#) () const
- bool [PairMode](#) () const
- bool [TriggerLost](#) () const

### Private Attributes

- uint16\_t [fWord](#)

### 7.19.1 Detailed Description

TDC status register.

Author

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

Date

Jun 2015

### 7.19.2 Member Enumeration Documentation

#### 7.19.2.1 enum VME::TDCV1x90Status::TDCResolution

Enumerator

***R\_800ps***

***R\_200ps***

***R\_100ps***

***R\_25ps***



### 7.19.3 Constructor & Destructor Documentation

7.19.3.1 VME::TDCV1x90Status::TDCV1x90Status ( const uint16\_t & *word* ) [inline]

7.19.3.2 virtual VME::TDCV1x90Status::~~TDCV1x90Status ( ) [inline],[virtual]

### 7.19.4 Member Function Documentation

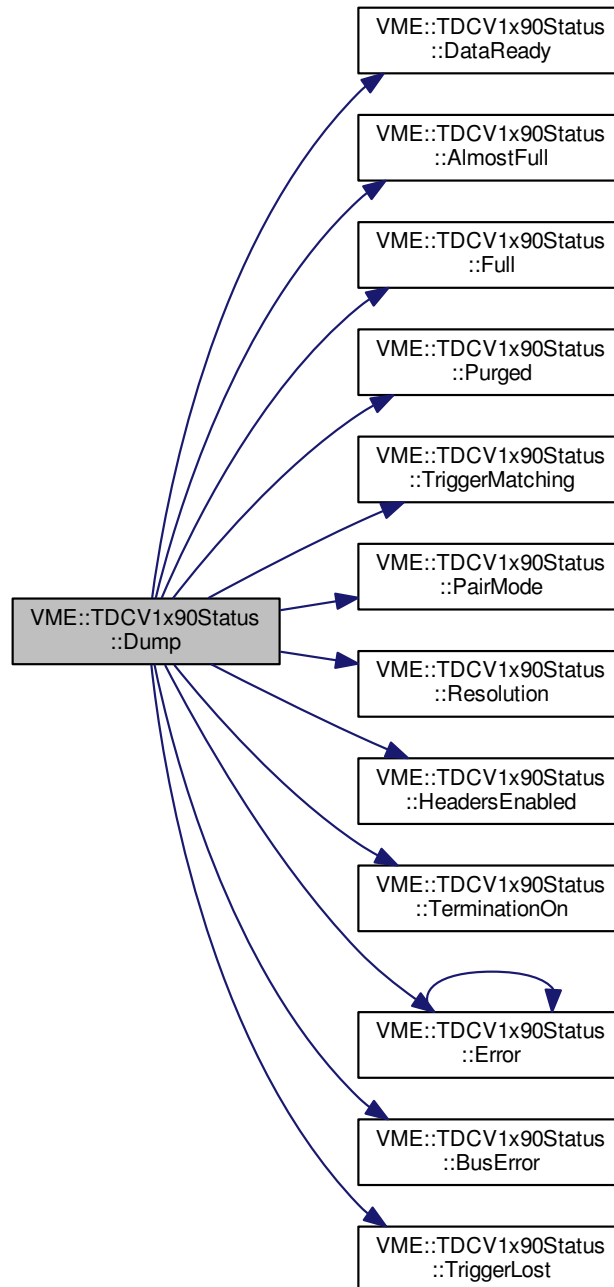
7.19.4.1 bool VME::TDCV1x90Status::AlmostFull ( ) const [inline]

7.19.4.2 bool VME::TDCV1x90Status::BusError ( ) const [inline]

7.19.4.3 bool VME::TDCV1x90Status::DataReady ( ) const [inline]

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

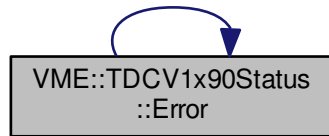
Here is the call graph for this function:



7.19.4.5 `bool VME::TDCV1x90Status::Error ( const unsigned int & id ) const [inline]`

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

Here is the call graph for this function:



7.19.4.7 `bool VME::TDCV1x90Status::Full ( ) const [inline]`

7.19.4.8 `uint16_t VME::TDCV1x90Status::GetValue ( ) const [inline]`

7.19.4.9 `bool VME::TDCV1x90Status::HeadersEnabled ( ) const [inline]`

7.19.4.10 `bool VME::TDCV1x90Status::PairMode ( ) const [inline]`

7.19.4.11 `bool VME::TDCV1x90Status::Purged ( ) const [inline]`

7.19.4.12 `TDCResolution VME::TDCV1x90Status::Resolution ( ) const [inline]`

7.19.4.13 `bool VME::TDCV1x90Status::TerminationOn ( ) const [inline]`

7.19.4.14 `bool VME::TDCV1x90Status::TriggerLost ( ) const [inline]`

7.19.4.15 `bool VME::TDCV1x90Status::TriggerMatching ( ) const [inline]`

## 7.19.5 Field Documentation

7.19.5.1 `uint16_t VME::TDCV1x90Status::fWord [private]`

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

- `include/VME_TDCV1x90.h`

## 7.20 VME::trailead\_t Struct Reference

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

### Data Fields

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

### 7.20.1 Field Documentation

7.20.1.1 `uint32_t VME::trailead_t::ettt`

7.20.1.2 `uint32_t VME::trailead_t::event_count`

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

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

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

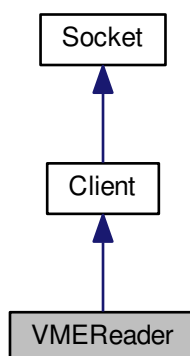
The documentation for this struct was generated from the following file:

- `include/VME_TDCV1x90.h`

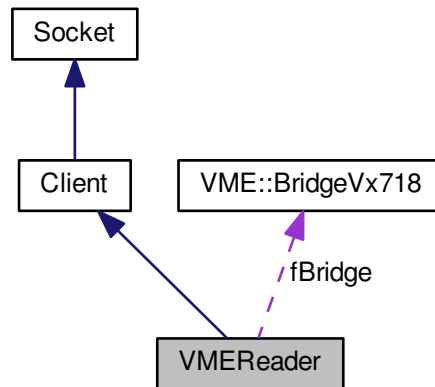
## 7.21 VMEReader Class Reference

```
#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 [StartPulser](#) (double period, double width, unsigned char num\_pulses=0)
- void [StopPulser](#) ()
- 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?*
- bool [fIsPulserStarted](#)

## Additional Inherited Members

### 7.21.1 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

### 7.21.2 Member Typedef Documentation

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

Mapper from physical VME addresses to pointers to TDC objects.

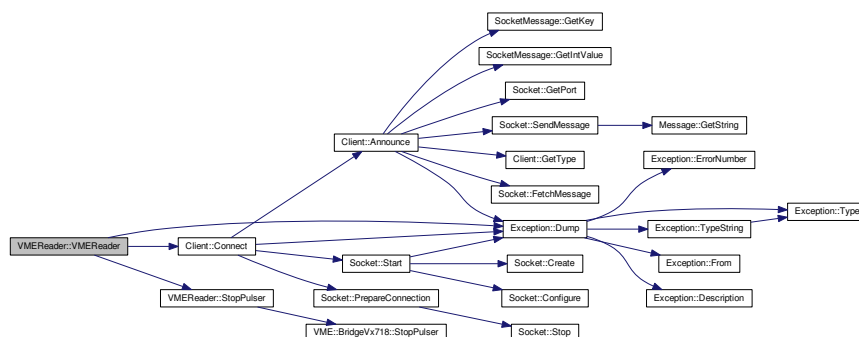
### 7.21.3 Constructor & Destructor Documentation

7.21.3.1 `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:



## 7.21.3.2 VMEReader::~~VMEReader ( ) [virtual]

Here is the call graph for this function:



## 7.21.4 Member Function Documentation

## 7.21.4.1 void VMEReader::Abort ( )

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

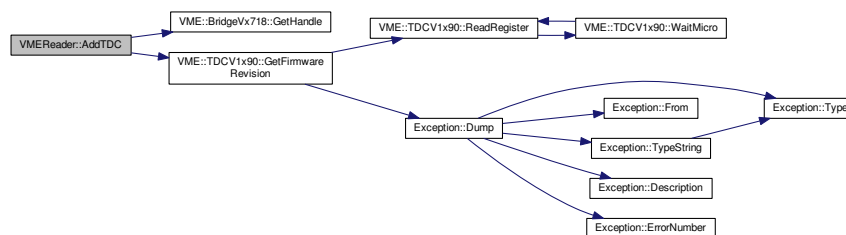
## 7.21.4.2 void VMEReader::AddTDC ( uint32\_t address )

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
----	----------------	--

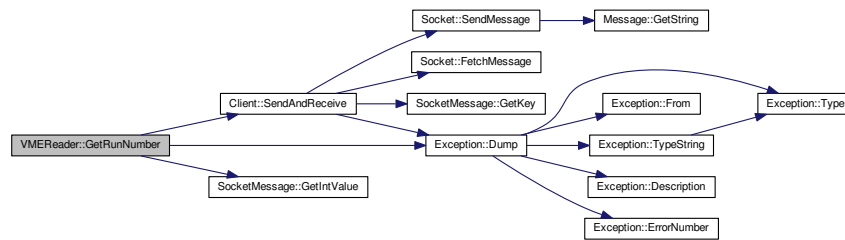
Here is the call graph for this function:



## 7.21.4.3 unsigned int VMEReader::GetRunNumber ( )

Ask the socket master a run number.

Here is the call graph for this function:

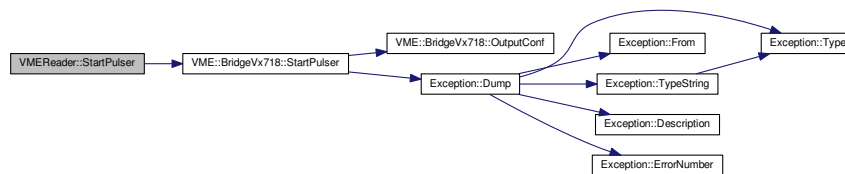


#### 7.21.4.4 VME::TDCV1x90\* VMEReaders::GetTDC ( uint32\_t address ) [inline]

Get a TDC on the [VME](#) bus Return a pointer to the TDC object, given its physical address on the [VME](#) bus.

#### 7.21.4.5 void VMEReaders::StartPulser ( double period, double width, unsigned char num\_pulses = 0 ) [inline]

Here is the call graph for this function:



#### 7.21.4.6 void VMEReaders::StopPulser ( ) [inline]

Here is the call graph for this function:



## 7.21.5 Field Documentation

### 7.21.5.1 VME::BridgeVx718\* VMEReaders::fBridge [private]

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



7.21.5.2 `bool VMEReader::fIsPulserStarted` [private]

7.21.5.3 `bool VMEReader::fOnSocket` [private]

Are we dealing with socket message passing?

7.21.5.4 `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](#)
- ~BridgeVx718Control
  - VME::BridgeVx718Control, [22](#)
- ~BridgeVx718Status
  - VME::BridgeVx718Status, [24](#)
- ~Client
  - Client, [26](#)
- ~Exception
  - Exception, [31](#)
- ~FileReader
  - FileReader, [35](#)
- ~Message
  - Message, [40](#)
- ~Messenger
  - Messenger, [43](#)
- ~Socket
  - Socket, [51](#)
- ~SocketMessage
  - SocketMessage, [60](#)
- ~TDCErrorFlag
  - VME::TDCErrorFlag, [63](#)
- ~TDCEvent
  - VME::TDCEvent, [65](#)
- ~TDCMeasurement
  - VME::TDCMeasurement, [71](#)
- ~TDCV1x90
  - VME::TDCV1x90, [76](#)
- ~TDCV1x90Control
  - VME::TDCV1x90Control, [96](#)
- ~TDCV1x90Status
  - VME::TDCV1x90Status, [103](#)
- ~VMEReader
  - VMEReader, [108](#)
- AUTOLOAD\_DEF\_CONFI
  - VME::TDCV1x90Opcodes, [14](#)
- AUTOLOAD\_USER\_CONF
  - VME::TDCV1x90Opcodes, [14](#)
- Abort
  - VMEReader, [109](#)
- abort
  - VME::TDCV1x90, [77](#)
- AcceptConnections
  - Socket, [51](#)
- acq\_mode
  - file\_header\_t, [33](#)
- AcquisitionMode
  - VME, [12](#)
- AddClient
  - Messenger, [43](#)
- AddTDC
  - VMEReader, [109](#)
- AlmostFull
  - VME::TDCV1x90Status, [103](#)
- am
  - VME::TDCV1x90, [95](#)
- am\_blt
  - VME::TDCV1x90, [95](#)
- Announce
  - Client, [27](#)
- Bind
  - Socket, [52](#)
- BridgeType
  - VME, [12](#)
- BridgeVx718
  - VME::BridgeVx718, [20](#)
- BridgeVx718Control
  - VME::BridgeVx718Control, [22](#)
- BridgeVx718Status
  - VME::BridgeVx718Status, [24](#)
- Broadcast
  - Messenger, [43](#)
- BusError
  - VME::TDCV1x90Status, [103](#)
- CAEN\_V1718
  - VME, [12](#)
- CAEN\_V2718
  - VME, [12](#)
- CLEAR\_KEEP\_TOKEN
  - VME::TDCV1x90Opcodes, [14](#)
- CLIENT
  - Socket communication objects, [9](#)
- CONT\_STOR
  - VME::TDCV1x90Opcodes, [14](#)
- CONT\_STORAGE
  - VME, [12](#)
- CheckConfiguration
  - VME::BridgeVx718, [20](#)
  - VME::TDCV1x90, [77](#)
- Client, [24](#)
  - ~Client, [26](#)
  - Announce, [27](#)
  - Client, [26](#)
  - Connect, [27](#)
  - Disconnect, [28](#)
  - fClientId, [30](#)
  - flsConnected, [30](#)

- GetType, 28
  - ParseMessage, 28
  - Receive, 28
  - Send, 29
  - SendAndReceive, 29
- coarse
  - VME::GlobalOffset, 36
- Configure
  - Socket, 52
- Connect
  - Client, 27
  - Messenger, 44
- Create
  - Socket, 52
- DEFAULT\_SETUP\_REG
  - VME::TDCV1x90Opcodes, 15
- DETECTOR
  - Socket communication objects, 9
- DIS\_ALL\_CHANNEL
  - VME::TDCV1x90Opcodes, 15
- DIS\_CHANNEL
  - VME::TDCV1x90Opcodes, 15
- DIS\_ERROR\_BYPASS
  - VME::TDCV1x90Opcodes, 15
- DIS\_ERROR\_MARK
  - VME::TDCV1x90Opcodes, 15
- DIS\_HEAD\_TRAILER
  - VME::TDCV1x90Opcodes, 15
- DIS\_SUB\_TRG
  - VME::TDCV1x90Opcodes, 15
- DISABLE\_TEST\_MODE
  - VME::TDCV1x90Opcodes, 15
- DLL\_Direct\_LowRes
  - VME::TDCV1x90, 75
- DLL\_PLL\_HighRes
  - VME::TDCV1x90, 75
- DLL\_PLL\_LowRes
  - VME::TDCV1x90, 75
- DLL\_PLL\_MedRes
  - VME::TDCV1x90, 75
- DLLMode
  - VME::TDCV1x90, 75
- DataReady
  - VME::TDCV1x90Status, 103
- Decode
  - HTTPMessage, 38
- Description
  - Exception, 31
- det\_mode
  - file\_header\_t, 33
- DetectionMode
  - VME, 12
- DisableChannel
  - VME::TDCV1x90, 77
- Disconnect
  - Client, 28
  - Messenger, 44
- DisconnectClient
  - Messenger, 45
- Dump
  - Exception, 31
  - HTTPMessage, 38
  - Message, 40
  - SocketMessage, 60
  - VME::BridgeVx718Status, 24
  - VME::TDCErrorFlag, 63
  - VME::TDCEvent, 66
  - VME::TDCMeasurement, 71
  - VME::TDCV1x90Control, 96
  - VME::TDCV1x90Status, 103
- DumpConnected
  - Socket, 52
- EN\_ALL\_CHANNEL
  - VME::TDCV1x90Opcodes, 15
- EN\_CHANNEL
  - VME::TDCV1x90Opcodes, 15
- EN\_ERROR\_BYPASS
  - VME::TDCV1x90Opcodes, 15
- EN\_ERROR\_MARK
  - VME::TDCV1x90Opcodes, 15
- EN\_HEAD\_TRAILER
  - VME::TDCV1x90Opcodes, 15
- EN\_SUB\_TRG
  - VME::TDCV1x90Opcodes, 15
- ENABLE\_TEST\_MODE
  - VME::TDCV1x90Opcodes, 15
- ETTT
  - VME::TDCEvent, 65
  - VME::TDCMeasurement, 71
- EXTRA\_SEARCH\_WIN\_WIDTH
  - VME, 13
- EnableChannel
  - VME::TDCV1x90, 77
- Encode
  - HTTPMessage, 38
- Error
  - VME::TDCV1x90Status, 104
- ErrorNumber
  - Exception, 32
- ettt
  - VME::trailead\_t, 106
- event\_count
  - VME::trailead\_t, 106
- EventType
  - VME::TDCEvent, 65
- Exception, 30
  - ~Exception, 31
  - Description, 31
  - Dump, 31
  - ErrorNumber, 32
  - Exception, 31
  - fDescription, 32
  - fErrorNumber, 32
  - fFrom, 32
  - fType, 32
  - From, 32

- Type, 32
- TypeString, 32
- fAcquisitionMode
  - VME::TDCV1x90, 95
- fAddress
  - Socket, 55
- fBaseAddr
  - VME::BridgeVx718, 21
  - VME::TDCV1x90, 95
- fBridge
  - VMEReader, 110
- fBuffer
  - Socket, 55
  - VME::TDCV1x90, 95
- fClientId
  - Client, 30
- fDescription
  - Exception, 32
- fDetectionMode
  - VME::TDCV1x90, 95
- fErrorMarks
  - VME::TDCV1x90, 95
- fErrorNumber
  - Exception, 32
- fFile
  - FileReader, 36
- fFrom
  - Exception, 32
- fHandle
  - VME::BridgeVx718, 21
  - VME::TDCV1x90, 95
- fHeader
  - FileReader, 36
- fIsConnected
  - Client, 30
- fIsPulserStarted
  - VMEReader, 110
- fMap
  - VME::TDCMeasurement, 72
- fMaster
  - Socket, 55
- fMessage
  - SocketMessage, 62
- fNumAttempts
  - Messenger, 49
- fOnSocket
  - VMEReader, 111
- fOriginalString
  - HTTPMessage, 39
- fPID
  - Messenger, 49
- fPort
  - Socket, 55
- fReadFds
  - Socket, 55
- fReadoutMode
  - FileReader, 36
- fSocketId
  - Socket, 55
- fSocketsConnected
  - Socket, 55
- fString
  - Message, 40
- fTDCCollection
  - VMEReader, 111
- fType
  - Exception, 32
- fVerb
  - VME::TDCV1x90, 95
- fWS
  - HTTPMessage, 39
  - Messenger, 49
- fWindowWidth
  - VME::TDCV1x90, 95
- fWord
  - VME::BridgeVx718Control, 23
  - VME::BridgeVx718Status, 24
  - VME::TDCErrorFlag, 64
  - VME::TDCEvent, 70
  - VME::TDCV1x90Control, 101
  - VME::TDCV1x90Status, 105
- FetchEvents
  - VME::TDCV1x90, 78
- FetchMessage
  - Socket, 52
- file\_header\_t, 33
  - acq\_mode, 33
  - det\_mode, 33
  - magic, 33
  - num\_hptdc, 33
  - run\_id, 33
  - spill\_id, 33
- FileReader, 33
  - ~FileReader, 35
  - fFile, 36
  - fHeader, 36
  - fReadoutMode, 36
  - FileReader, 34
  - GetNextEvent, 35
  - GetNextMeasurement, 35
  - GetNumTDCs, 35
- Filler
  - VME::TDCEvent, 65
- fine
  - VME::GlobalOffset, 36
- From
  - Exception, 32
- Full
  - VME::TDCV1x90Status, 105
- gEnd
  - VME::TDCV1x90, 95
- GetAcquisitionMode
  - VME::TDCV1x90, 78
- GetAddressIncrement
  - VME::BridgeVx718Control, 22
- GetAlign64

- VME::TDCV1x90Control, 97
- GetArbiterType
  - VME::BridgeVx718Control, 22
- GetBERR
  - VME::BridgeVx718Status, 24
- GetBLTEventNumberRegister
  - VME::TDCV1x90, 78
- GetBunchId
  - VME::TDCEvent, 66
  - VME::TDCMeasurement, 72
- GetBusError
  - VME::TDCV1x90Control, 97
- GetBusReqLevel
  - VME::BridgeVx718Control, 22
- GetBusTimeout
  - VME::BridgeVx718Control, 23
- GetChannelDeadTime
  - VME::TDCV1x90, 78
- GetChannelId
  - VME::TDCEvent, 66
  - VME::TDCMeasurement, 72
- GetCompensation
  - VME::TDCV1x90Control, 97
- GetControl
  - VME::TDCV1x90, 79
- GetDLLClock
  - VME::TDCV1x90, 79
- GetDTACK
  - VME::BridgeVx718Status, 24
- GetDetectionMode
  - VME::TDCV1x90, 79
- GetDipSwitch
  - VME::BridgeVx718Status, 24
- GetETTT
  - VME::TDCEvent, 67
  - VME::TDCV1x90, 80
  - VME::TDCV1x90Control, 98
- GetEmptyEvent
  - VME::TDCV1x90Control, 98
- GetErrorFlags
  - VME::TDCEvent, 66
- GetErrorMarks
  - VME::TDCV1x90, 79
- GetEventCount
  - VME::TDCEvent, 67
- GetEventCounter
  - VME::TDCV1x90, 80
- GetEventFIFO
  - VME::TDCV1x90Control, 98
- GetEventId
  - VME::TDCEvent, 67
  - VME::TDCMeasurement, 72
- GetEventStored
  - VME::TDCV1x90, 80
- GetFIFOSize
  - VME::TDCV1x90, 81
- GetFirmwareRevision
  - VME::TDCV1x90, 81
- GetGeo
  - VME::TDCEvent, 68
- GetGlobalOffset
  - VME::TDCV1x90, 81
- GetHandle
  - VME::BridgeVx718, 20
- GetIntValue
  - SocketMessage, 60
- GetInterruptReq
  - VME::BridgeVx718Control, 23
- GetKey
  - HTTPMessage, 38
  - Message, 40
  - SocketMessage, 60
- GetLeadingTime
  - VME::TDCEvent, 68
  - VME::TDCMeasurement, 72
- GetMEBAccess
  - VME::TDCV1x90Control, 98
- GetModel
  - VME::TDCV1x90, 81
- GetNextEvent
  - FileReader, 35
- GetNextMeasurement
  - FileReader, 35
- GetNumTDCs
  - FileReader, 35
- GetOUI
  - VME::TDCV1x90, 82
- GetPol
  - VME::TDCV1x90, 82
- GetPort
  - Socket, 52
- GetRCAdjust
  - VME::TDCV1x90, 82
- GetReleaseType
  - VME::BridgeVx718Control, 23
- GetRequesterType
  - VME::BridgeVx718Control, 23
- GetResolution
  - VME::TDCV1x90, 83
- GetRunNumber
  - VMEReader, 109
- GetSRAMCompensation
  - VME::TDCV1x90Control, 98
- GetSWTermination
  - VME::TDCV1x90Control, 98
- GetSerialNumber
  - VME::TDCV1x90, 83
- GetSocketId
  - Socket, 53
- GetSocketType
  - Socket, 53
- GetStatus
  - VME::TDCEvent, 68
  - VME::TDCV1x90, 83
- GetString
  - Message, 40

- SocketMessage, 60
- GetSysRes
  - VME::BridgeVx718Control, 23
- GetSystemControl
  - VME::BridgeVx718Status, 24
- GetSystemReset
  - VME::BridgeVx718Status, 24
- GetTDC
  - VMEReader, 110
- GetTDCEncapsulation
  - VME::TDCV1x90, 84
- GetTDCId
  - VME::TDCEvent, 69
  - VME::TDCMeasurement, 72
- GetTermination
  - VME::TDCV1x90Control, 98
- GetTestFIFO
  - VME::TDCV1x90Control, 98
- GetTestMode
  - VME::TDCV1x90, 84
- GetTrailingTime
  - VME::TDCEvent, 69
  - VME::TDCMeasurement, 72
- GetTriggerConfiguration
  - VME::TDCV1x90, 84
- GetType
  - Client, 28
  - Messenger, 45
  - VME::TDCEvent, 69
- GetUSBType
  - VME::BridgeVx718Status, 24
- GetValue
  - SocketMessage, 60
  - VME::TDCV1x90Control, 98
  - VME::TDCV1x90Status, 105
- GetVectorValue
  - SocketMessage, 60
- GetWidth
  - VME::TDCEvent, 69
- GetWindowOffset
  - VME::TDCV1x90, 84
- GetWindowWidth
  - VME::TDCV1x90, 85
- GetWord
  - VME::TDCErrorFlag, 63
  - VME::TDCEvent, 69
- GetWordCount
  - VME::TDCEvent, 69
- GlobalHeader
  - VME::TDCEvent, 65
  - VME::TDCMeasurement, 71
- GlobalTrailer
  - VME::TDCEvent, 65
  - VME::TDCMeasurement, 71
- HTTPMessage, 36
  - Decode, 38
  - Dump, 38
  - Encode, 38
  - fOriginalString, 39
  - fWS, 39
  - GetKey, 38
  - HTTPMessage, 38
- HardwareReset
  - VME::TDCV1x90, 85
- HasGroupError
  - VME::TDCErrorFlag, 63
- HasInternalChipError
  - VME::TDCErrorFlag, 63
- HasL1BufferOverflow
  - VME::TDCErrorFlag, 63
- HasReachedEventSizeLimit
  - VME::TDCErrorFlag, 63
- HasReadoutFIFOOverflow
  - VME::TDCErrorFlag, 63
- HasTriggerFIFOOverflow
  - VME::TDCErrorFlag, 63
- HeadersEnabled
  - VME::TDCV1x90Status, 105
- INVALID
  - Socket communication objects, 9
- InputConf
  - VME::BridgeVx718, 20
- InputRead
  - VME::BridgeVx718, 21
- IsFromWeb
  - Message, 40
- IsTrailing
  - VME::TDCEvent, 70
- IsWebSocket
  - Socket, 53
- kBLTEventNumber
  - VME::TDCV1x90, 75
- kControl
  - VME::TDCV1x90, 75
- kEventCounter
  - VME::TDCV1x90, 75
- kEventFIFO
  - VME::TDCV1x90, 76
- kEventFIFOStatusRegister
  - VME::TDCV1x90, 76
- kEventFIFOStoredRegister
  - VME::TDCV1x90, 76
- kEventStored
  - VME::TDCV1x90, 75
- kFirmwareRev
  - VME::TDCV1x90, 75
- kGeoAddress
  - VME::TDCV1x90, 75
- kInterruptLevel
  - VME::TDCV1x90, 75
- kInterruptVector
  - VME::TDCV1x90, 75
- kMCSTBase
  - VME::TDCV1x90, 75
- kMCSTControl

- VME::TDCV1x90, 75
- kMicro
  - VME::TDCV1x90, 76
- kMicroHandshake
  - VME::TDCV1x90, 76
- kModuleReset
  - VME::TDCV1x90, 75
- kOutputBuffer
  - VME::TDCV1x90, 75
- kROMBoard0
  - VME::TDCV1x90, 76
- kROMBoard1
  - VME::TDCV1x90, 76
- kROMBoard2
  - VME::TDCV1x90, 76
- kROMOui0
  - VME::TDCV1x90, 76
- kROMOui1
  - VME::TDCV1x90, 76
- kROMOui2
  - VME::TDCV1x90, 76
- kROMRevis0
  - VME::TDCV1x90, 76
- kROMRevis1
  - VME::TDCV1x90, 76
- kROMRevis2
  - VME::TDCV1x90, 76
- kROMRevis3
  - VME::TDCV1x90, 76
- kROMSerNum0
  - VME::TDCV1x90, 76
- kROMSerNum1
  - VME::TDCV1x90, 76
- kSoftwareClear
  - VME::TDCV1x90, 75
- kStatus
  - VME::TDCV1x90, 75
- LOAD\_DEF\_CONFIG
  - VME::TDCV1x90Opcodes, 15
- LOAD\_USER\_CONFIG
  - VME::TDCV1x90Opcodes, 15
- leading
  - VME::trailead\_t, 106
- LeadingEdge
  - VME::TDCMeasurement, 71
- Listen
  - Socket, 53
- MASTER
  - Socket communication objects, 9
- MATCH\_WIN\_WIDTH
  - VME, 13
- magic
  - file\_header\_t, 33
- Message, 39
  - ~Message, 40
  - Dump, 40
  - fString, 40
  - GetKey, 40
  - GetString, 40
  - IsFromWeb, 40
  - Message, 40
- Messenger, 41
  - ~Messenger, 43
  - AddClient, 43
  - Broadcast, 43
  - Connect, 44
  - Disconnect, 44
  - DisconnectClient, 45
  - fNumAttempts, 49
  - fPID, 49
  - fWS, 49
  - GetType, 45
  - Messenger, 42
  - ProcessMessage, 45
  - Receive, 46
  - Send, 46
  - StartAcquisition, 48
  - StopAcquisition, 48
  - SwitchClientType, 48
- micro\_handshake
  - VME, 12
- mod\_reg
  - VME::TDCV1x90, 75
- nchannels
  - VME::TDCV1x90, 95
- num\_hptdc
  - file\_header\_t, 33
- OLEADING
  - VME, 12
- OTRILING
  - VME, 12
- Object
  - SocketMessage, 61
- operator<<
  - VME::TDCErrrorFlag, 64
- OutputConf
  - VME::BridgeVx718, 21
- OutputOff
  - VME::BridgeVx718, 21
- OutputOn
  - VME::BridgeVx718, 21
- PAIR
  - VME, 12
- pair\_lead\_res
  - VME::TDCV1x90, 95
- pair\_width\_res
  - VME::TDCV1x90, 95
- PairMode
  - VME::TDCV1x90Status, 105
- ParseMessage
  - Client, 28
- PrepareConnection
  - Socket, 53



- ProcessMessage
  - Messenger, [45](#)
- Purged
  - VME::TDCV1x90Status, [105](#)
- r100ps
  - VME, [13](#)
- r200ps
  - VME, [13](#)
- r25ps
  - VME, [13](#)
- r800ps
  - VME, [13](#)
- R\_100ps
  - VME::TDCV1x90Status, [102](#)
- R\_200ps
  - VME::TDCV1x90Status, [102](#)
- R\_25ps
  - VME::TDCV1x90Status, [102](#)
- R\_800ps
  - VME::TDCV1x90Status, [102](#)
- READ\_ACQ\_MOD
  - VME::TDCV1x90Opcodes, [15](#)
- READ\_ADJUST\_CH
  - VME::TDCV1x90Opcodes, [15](#)
- READ\_DEAD\_TIME
  - VME::TDCV1x90Opcodes, [15](#)
- READ\_DETECTION
  - VME::TDCV1x90Opcodes, [15](#)
- READ\_DLL\_LOCK
  - VME::TDCV1x90Opcodes, [15](#)
- READ\_EEPROM
  - VME::TDCV1x90Opcodes, [15](#)
- READ\_EN\_PATTERN
  - VME::TDCV1x90Opcodes, [15](#)
- READ\_EN\_PATTERN32
  - VME::TDCV1x90Opcodes, [15](#)
- READ\_ERROR\_STATUS
  - VME::TDCV1x90Opcodes, [15](#)
- READ\_ERROR\_TYPES
  - VME::TDCV1x90Opcodes, [15](#)
- READ\_EVENT\_SIZE
  - VME::TDCV1x90Opcodes, [15](#)
- READ\_FIFO\_SIZE
  - VME::TDCV1x90Opcodes, [16](#)
- READ\_GLOB\_OFFS
  - VME::TDCV1x90Opcodes, [16](#)
- READ\_HEAD\_TRAILER
  - VME::TDCV1x90Opcodes, [16](#)
- READ\_MICRO\_REV
  - VME::TDCV1x90Opcodes, [16](#)
- READ\_OK
  - VME, [12](#)
- READ\_RC\_ADJ
  - VME::TDCV1x90Opcodes, [16](#)
- READ\_RES
  - VME::TDCV1x90Opcodes, [16](#)
- READ\_SETUP\_REG
  - VME::TDCV1x90Opcodes, [16](#)
- READ\_SETUP\_SCANPATH
  - VME::TDCV1x90Opcodes, [16](#)
- READ\_SPARE
  - VME::TDCV1x90Opcodes, [16](#)
- READ\_STATUS\_STREAM
  - VME::TDCV1x90Opcodes, [16](#)
- READ\_TDC\_ID
  - VME::TDCV1x90Opcodes, [16](#)
- READ\_TRG\_CONF
  - VME::TDCV1x90Opcodes, [16](#)
- REJECT\_MARGIN
  - VME, [13](#)
- RESET\_DLL\_PLL
  - VME::TDCV1x90Opcodes, [16](#)
- REV\_DATE\_MICRO\_FW
  - VME::TDCV1x90Opcodes, [16](#)
- ReadAcquisitionMode
  - VME::TDCV1x90, [85](#)
- ReadDetectionMode
  - VME::TDCV1x90, [85](#)
- ReadRegister
  - VME::BridgeVx718, [21](#)
  - VME::TDCV1x90, [85](#), [86](#)
- Receive
  - Client, [28](#)
  - Messenger, [46](#)
- Resolution
  - VME::TDCV1x90Status, [105](#)
- run\_id
  - file\_header\_t, [33](#)
- SAVE\_RC\_ADJ
  - VME::TDCV1x90Opcodes, [16](#)
- SAVE\_USER\_CONFIG
  - VME::TDCV1x90Opcodes, [16](#)
- SET\_ADJUST\_CH
  - VME::TDCV1x90Opcodes, [16](#)
- SET\_DEAD\_TIME
  - VME::TDCV1x90Opcodes, [16](#)
- SET\_DETECTION
  - VME::TDCV1x90Opcodes, [16](#)
- SET\_DLL\_CLOCK
  - VME::TDCV1x90Opcodes, [16](#)
- SET\_ERROR\_TYPES
  - VME::TDCV1x90Opcodes, [16](#)
- SET\_EVENT\_SIZE
  - VME::TDCV1x90Opcodes, [16](#)
- SET\_FIFO\_SIZE
  - VME::TDCV1x90Opcodes, [16](#)
- SET\_GLOB\_OFFS
  - VME::TDCV1x90Opcodes, [16](#)
- SET\_KEEP\_TOKEN
  - VME::TDCV1x90Opcodes, [16](#)
- SET\_PAIR\_RES
  - VME::TDCV1x90Opcodes, [16](#)
- SET\_RC\_ADJ
  - VME::TDCV1x90Opcodes, [16](#)
- SET\_REJ\_MARGIN
  - VME::TDCV1x90Opcodes, [16](#)

- SET\_SW\_MARGIN
  - VME::TDCV1x90Opcodes, [17](#)
- SET\_TDC\_TSET\_OUTPUT
  - VME::TDCV1x90Opcodes, [17](#)
- SET\_TR\_LEAD\_LSB
  - VME::TDCV1x90Opcodes, [17](#)
- SET\_WIN\_OFFS
  - VME::TDCV1x90Opcodes, [17](#)
- SET\_WIN\_WIDTH
  - VME::TDCV1x90Opcodes, [17](#)
- SelectConnections
  - Socket, [53](#)
- Send
  - Client, [29](#)
  - Messenger, [46](#)
- SendAndReceive
  - Client, [29](#)
- SendMessage
  - Socket, [54](#)
- SetAcquisitionMode
  - VME::TDCV1x90, [86](#)
- SetAlign64
  - VME::TDCV1x90Control, [98](#)
- SetBLTEventNumberRegister
  - VME::TDCV1x90, [86](#)
- SetBusError
  - VME::TDCV1x90Control, [98](#)
- SetChannelDeadTime
  - VME::TDCV1x90, [87](#)
- SetCompensation
  - VME::TDCV1x90Control, [98](#)
- SetContinuousStorage
  - VME::TDCV1x90, [87](#)
- SetControl
  - VME::TDCV1x90, [87](#)
- SetDLLClock
  - VME::TDCV1x90, [88](#)
- SetDetectionMode
  - VME::TDCV1x90, [88](#)
- SetETTT
  - VME::TDCV1x90, [89](#)
  - VME::TDCV1x90Control, [99](#)
- SetEmptyEvent
  - VME::TDCV1x90Control, [99](#)
- SetErrorMarks
  - VME::TDCV1x90, [88](#)
- SetEventFIFO
  - VME::TDCV1x90Control, [99](#)
- SetEventsCollection
  - VME::TDCMeasurement, [72](#)
- SetFIFOSize
  - VME::TDCV1x90, [89](#)
- SetGlobalOffset
  - VME::TDCV1x90, [89](#)
- SetKeyValue
  - SocketMessage, [61](#), [62](#)
- SetLSBTraileadEdge
  - VME::TDCV1x90, [90](#)
- SetMEBAccess
  - VME::TDCV1x90Control, [100](#)
- SetPairModeResolution
  - VME::TDCV1x90, [90](#)
- SetPol
  - VME::TDCV1x90, [90](#)
- SetPort
  - Socket, [54](#)
- SetRCAdjust
  - VME::TDCV1x90, [91](#)
- SetSRAMCompensation
  - VME::TDCV1x90Control, [100](#)
- SetSWTermination
  - VME::TDCV1x90Control, [100](#)
- SetSocketId
  - Socket, [54](#)
- SetStatus
  - VME::TDCV1x90, [91](#)
- SetTDCEncapsulation
  - VME::TDCV1x90, [91](#)
- SetTermination
  - VME::TDCV1x90Control, [101](#)
- SetTestFIFO
  - VME::TDCV1x90Control, [101](#)
- SetTestMode
  - VME::TDCV1x90, [92](#)
- SetTriggerMatching
  - VME::TDCV1x90, [92](#)
- SetVerboseLevel
  - VME::TDCV1x90, [92](#)
- SetWindowOffset
  - VME::TDCV1x90, [93](#)
- SetWindowWidth
  - VME::TDCV1x90, [93](#)
- SetWord
  - VME::TDCEvent, [70](#)
- Socket, [49](#)
  - ~Socket, [51](#)
  - AcceptConnections, [51](#)
  - Bind, [52](#)
  - Configure, [52](#)
  - Create, [52](#)
  - DumpConnected, [52](#)
  - fAddress, [55](#)
  - fBuffer, [55](#)
  - fMaster, [55](#)
  - fPort, [55](#)
  - fReadFds, [55](#)
  - fSocketId, [55](#)
  - fSocketsConnected, [55](#)
  - FetchMessage, [52](#)
  - GetPort, [52](#)
  - GetSocketId, [53](#)
  - GetSocketType, [53](#)
  - IsWebSocket, [53](#)
  - Listen, [53](#)
  - PrepareConnection, [53](#)
  - SelectConnections, [53](#)

- SendMessage, [54](#)
- SetPort, [54](#)
- SetSocketId, [54](#)
- Socket, [51](#)
- SocketCollection, [51](#)
- Start, [54](#)
- Stop, [54](#)
- Socket communication objects, [9](#)
  - CLIENT, [9](#)
  - DETECTOR, [9](#)
  - INVALID, [9](#)
  - MASTER, [9](#)
  - SocketType, [9](#)
  - WEBSOCKET\_CLIENT, [9](#)
- SocketCollection
  - Socket, [51](#)
- SocketMessage, [55](#)
  - ~SocketMessage, [60](#)
  - Dump, [60](#)
  - fMessage, [62](#)
  - GetIntValue, [60](#)
  - GetKey, [60](#)
  - GetString, [60](#)
  - GetValue, [60](#)
  - GetVectorValue, [60](#)
  - Object, [61](#)
  - SetKeyValue, [61](#), [62](#)
  - SocketMessage, [57–59](#)
  - String, [62](#)
- SocketType
  - Socket communication objects, [9](#)
- SoftwareClear
  - VME::TDCV1x90, [93](#)
- SoftwareReset
  - VME::TDCV1x90, [93](#)
- spill\_id
  - file\_header\_t, [33](#)
- Start
  - Socket, [54](#)
- StartAcquisition
  - Messenger, [48](#)
- StartPulser
  - VME::BridgeVx718, [21](#)
  - VMEReader, [110](#)
- Stop
  - Socket, [54](#)
- StopAcquisition
  - Messenger, [48](#)
- StopPulser
  - VME::BridgeVx718, [21](#)
  - VMEReader, [110](#)
- String
  - SocketMessage, [62](#)
- SwitchClientType
  - Messenger, [48](#)
- TDCCollection
  - VMEReader, [108](#)
- TDCError
  - VME::TDCEvent, [65](#)
- TDCErrorFlag
  - VME::TDCErrorFlag, [63](#)
- TDCEvent
  - VME::TDCEvent, [65](#)
- TDCEventCollection
  - VME, [12](#)
- TDCHeader
  - VME::TDCEvent, [65](#)
  - VME::TDCMeasurement, [71](#)
- TDCMeasurement
  - VME::TDCEvent, [65](#)
  - VME::TDCMeasurement, [71](#)
- TDCResolution
  - VME::TDCV1x90Status, [102](#)
- TDCTrailer
  - VME::TDCEvent, [65](#)
  - VME::TDCMeasurement, [71](#)
- TDCV1x90
  - VME::TDCV1x90, [76](#)
- TDCV1x90Control
  - VME::TDCV1x90Control, [96](#)
- TDCV1x90Status
  - VME::TDCV1x90Status, [103](#)
- TRAILHEAD
  - VME, [12](#)
- TRG\_MATCH
  - VME::TDCV1x90Opcodes, [17](#)
- TRIG\_MATCH
  - VME, [12](#)
- TRIG\_TIME\_SUB
  - VME, [13](#)
- TerminationOn
  - VME::TDCV1x90Status, [105](#)
- total\_hits
  - VME::trailead\_t, [106](#)
- trailead\_edge\_lsb
  - VME, [12](#)
- trailead\_edge\_res
  - VME::TDCV1x90, [95](#)
- trailing
  - VME::trailead\_t, [106](#)
- TrailingEdge
  - VME::TDCMeasurement, [71](#)
- trig\_conf
  - VME, [13](#)
- TriggerLost
  - VME::TDCV1x90Status, [105](#)
- TriggerMatching
  - VME::TDCV1x90Status, [105](#)
- Type
  - Exception, [32](#)
  - VME::TDCMeasurement, [71](#)
- TypeString
  - Exception, [32](#)
- UPDATE\_SETUP\_REG
  - VME::TDCV1x90Opcodes, [17](#)
- UPDATE\_SETUP\_TDC

- VME::TDCV1x90Opcodes, 17
- VME, 11
  - AcquisitionMode, 12
  - BridgeType, 12
  - CAEN\_V1718, 12
  - CAEN\_V2718, 12
  - CONT\_STORAGE, 12
  - DetectionMode, 12
  - EXTRA\_SEARCH\_WIN\_WIDTH, 13
  - MATCH\_WIN\_WIDTH, 13
  - micro\_handshake, 12
  - OLEADING, 12
  - OTRILING, 12
  - PAIR, 12
  - r100ps, 13
  - r200ps, 13
  - r25ps, 13
  - r800ps, 13
  - READ\_OK, 12
  - REJECT\_MARGIN, 13
  - TDCEventCollection, 12
  - TRAILEAD, 12
  - TRIG\_MATCH, 12
  - TRIG\_TIME\_SUB, 13
  - trailead\_edge\_lsb, 12
  - trig\_conf, 13
  - WIN\_OFFSET, 13
  - WRITE\_OK, 12
- VME::BridgeVx718, 19
  - ~BridgeVx718, 20
  - BridgeVx718, 20
  - CheckConfiguration, 20
  - fBaseAddr, 21
  - fHandle, 21
  - GetHandle, 20
  - InputConf, 20
  - InputRead, 21
  - OutputConf, 21
  - OutputOff, 21
  - OutputOn, 21
  - ReadRegister, 21
  - StartPulser, 21
  - StopPulser, 21
  - WriteRegister, 21
- VME::BridgeVx718Control, 22
  - ~BridgeVx718Control, 22
  - BridgeVx718Control, 22
  - fWord, 23
  - GetAddressIncrement, 22
  - GetArbiterType, 22
  - GetBusReqLevel, 22
  - GetBusTimeout, 23
  - GetInterruptReq, 23
  - GetReleaseType, 23
  - GetRequesterType, 23
  - GetSysRes, 23
- VME::BridgeVx718Status, 23
  - ~BridgeVx718Status, 24
  - BridgeVx718Status, 24
  - Dump, 24
  - fWord, 24
  - GetBERR, 24
  - GetDTACK, 24
  - GetDipSwitch, 24
  - GetSystemControl, 24
  - GetSystemReset, 24
  - GetUSBType, 24
- VME::GlobalOffset, 36
  - coarse, 36
  - fine, 36
- VME::TDCErrorFlag, 62
  - ~TDCErrorFlag, 63
  - Dump, 63
  - fWord, 64
  - GetWord, 63
  - HasGroupError, 63
  - HasInternalChipError, 63
  - HasL1BufferOverflow, 63
  - HasReachedEventSizeLimit, 63
  - HasReadoutFIFOOverflow, 63
  - HasTriggerFIFOOverflow, 63
  - operator<<, 64
  - TDCErrorFlag, 63
- VME::TDCEvent, 64
  - ~TDCEvent, 65
  - Dump, 66
  - ETTT, 65
  - EventType, 65
  - fWord, 70
  - Filler, 65
  - GetBunchId, 66
  - GetChannelId, 66
  - GetETTT, 67
  - GetErrorFlags, 66
  - GetEventCount, 67
  - GetEventId, 67
  - GetGeo, 68
  - GetLeadingTime, 68
  - GetStatus, 68
  - GetTDCId, 69
  - GetTrailingTime, 69
  - GetType, 69
  - GetWidth, 69
  - GetWord, 69
  - GetWordCount, 69
  - GlobalHeader, 65
  - GlobalTrailer, 65
  - IsTrailing, 70
  - SetWord, 70
  - TDCError, 65
  - TDCEvent, 65
  - TDCHHeader, 65
  - TDCMeasurement, 65
  - TDCTrailer, 65
- VME::TDCMeasurement, 70
  - ~TDCMeasurement, 71

- Dump, [71](#)
- ETTT, [71](#)
- fMap, [72](#)
- GetBunchId, [72](#)
- GetChannelId, [72](#)
- GetEventId, [72](#)
- GetLeadingTime, [72](#)
- GetTDCId, [72](#)
- GetTrailingTime, [72](#)
- GlobalHeader, [71](#)
- GlobalTrailer, [71](#)
- LeadingEdge, [71](#)
- SetEventsCollection, [72](#)
- TDCHeader, [71](#)
- TDCMeasurement, [71](#)
- TDCTrailer, [71](#)
- TrailingEdge, [71](#)
- Type, [71](#)
- VME::TDCV1x90, [73](#)
- ~TDCV1x90, [76](#)
- abort, [77](#)
- am, [95](#)
- am\_blt, [95](#)
- CheckConfiguration, [77](#)
- DLL\_Direct\_LowRes, [75](#)
- DLL\_PLL\_HighRes, [75](#)
- DLL\_PLL\_LowRes, [75](#)
- DLL\_PLL\_MedRes, [75](#)
- DLLMode, [75](#)
- DisableChannel, [77](#)
- EnableChannel, [77](#)
- fAcquisitionMode, [95](#)
- fBaseAddr, [95](#)
- fBuffer, [95](#)
- fDetectionMode, [95](#)
- fErrorMarks, [95](#)
- fHandle, [95](#)
- fVerb, [95](#)
- fWindowWidth, [95](#)
- FetchEvents, [78](#)
- gEnd, [95](#)
- GetAcquisitionMode, [78](#)
- GetBLTEventNumberRegister, [78](#)
- GetChannelDeadTime, [78](#)
- GetControl, [79](#)
- GetDLLClock, [79](#)
- GetDetectionMode, [79](#)
- GetETTT, [80](#)
- GetErrorMarks, [79](#)
- GetEventCounter, [80](#)
- GetEventStored, [80](#)
- GetFIFOSize, [81](#)
- GetFirmwareRevision, [81](#)
- GetGlobalOffset, [81](#)
- GetModel, [81](#)
- GetOUI, [82](#)
- GetPol, [82](#)
- GetRCAdjust, [82](#)
- GetResolution, [83](#)
- GetSerialNumber, [83](#)
- GetStatus, [83](#)
- GetTDCEncapsulation, [84](#)
- GetTestMode, [84](#)
- GetTriggerConfiguration, [84](#)
- GetWindowOffset, [84](#)
- GetWindowWidth, [85](#)
- HardwareReset, [85](#)
- kBLTEventNumber, [75](#)
- kControl, [75](#)
- kEventCounter, [75](#)
- kEventFIFO, [76](#)
- kEventFIFOStatusRegister, [76](#)
- kEventFIFOStoredRegister, [76](#)
- kEventStored, [75](#)
- kFirmwareRev, [75](#)
- kGeoAddress, [75](#)
- kInterruptLevel, [75](#)
- kInterruptVector, [75](#)
- kMCSTBase, [75](#)
- kMCSTControl, [75](#)
- kMicro, [76](#)
- kMicroHandshake, [76](#)
- kModuleReset, [75](#)
- kOutputBuffer, [75](#)
- kROMBoard0, [76](#)
- kROMBoard1, [76](#)
- kROMBoard2, [76](#)
- kROMOui0, [76](#)
- kROMOui1, [76](#)
- kROMOui2, [76](#)
- kROMRevis0, [76](#)
- kROMRevis1, [76](#)
- kROMRevis2, [76](#)
- kROMRevis3, [76](#)
- kROMSerNum0, [76](#)
- kROMSerNum1, [76](#)
- kSoftwareClear, [75](#)
- kStatus, [75](#)
- mod\_reg, [75](#)
- nchannels, [95](#)
- pair\_lead\_res, [95](#)
- pair\_width\_res, [95](#)
- ReadAcquisitionMode, [85](#)
- ReadDetectionMode, [85](#)
- ReadRegister, [85, 86](#)
- SetAcquisitionMode, [86](#)
- SetBLTEventNumberRegister, [86](#)
- SetChannelDeadTime, [87](#)
- SetContinuousStorage, [87](#)
- SetControl, [87](#)
- SetDLLClock, [88](#)
- SetDetectionMode, [88](#)
- SetETTT, [89](#)
- SetErrorMarks, [88](#)
- SetFIFOSize, [89](#)
- SetGlobalOffset, [89](#)

- SetLSBTraileadEdge, 90
- SetPairModeResolution, 90
- SetPol, 90
- SetRCAdjust, 91
- SetStatus, 91
- SetTDCEncapsulation, 91
- SetTestMode, 92
- SetTriggerMatching, 92
- SetVerboseLevel, 92
- SetWindowOffset, 93
- SetWindowWidth, 93
- SoftwareClear, 93
- SoftwareReset, 93
- TDCV1x90, 76
- trailead\_edge\_res, 95
- WaitMicro, 94
- WriteRegister, 94
- VME::TDCV1x90Control, 95
  - ~TDCV1x90Control, 96
  - Dump, 96
  - fWord, 101
  - GetAlign64, 97
  - GetBusError, 97
  - GetCompensation, 97
  - GetETTT, 98
  - GetEmptyEvent, 98
  - GetEventFIFO, 98
  - GetMEBAccess, 98
  - GetSRAMCompensation, 98
  - GetSWTermination, 98
  - GetTermination, 98
  - GetTestFIFO, 98
  - GetValue, 98
  - SetAlign64, 98
  - SetBusError, 98
  - SetCompensation, 98
  - SetETTT, 99
  - SetEmptyEvent, 99
  - SetEventFIFO, 99
  - SetMEBAccess, 100
  - SetSRAMCompensation, 100
  - SetSWTermination, 100
  - SetTermination, 101
  - SetTestFIFO, 101
  - TDCV1x90Control, 96
- VME::TDCV1x90Opcodes, 13
  - AUTOLOAD\_DEF\_CONF, 14
  - AUTOLOAD\_USER\_CONF, 14
  - CLEAR\_KEEP\_TOKEN, 14
  - CONT\_STOR, 14
  - DEFAULT\_SETUP\_REG, 15
  - DIS\_ALL\_CHANNEL, 15
  - DIS\_CHANNEL, 15
  - DIS\_ERROR\_BYPASS, 15
  - DIS\_ERROR\_MARK, 15
  - DIS\_HEAD\_TRAILER, 15
  - DIS\_SUB\_TRG, 15
  - DISABLE\_TEST\_MODE, 15
  - EN\_ALL\_CHANNEL, 15
  - EN\_CHANNEL, 15
  - EN\_ERROR\_BYPASS, 15
  - EN\_ERROR\_MARK, 15
  - EN\_HEAD\_TRAILER, 15
  - EN\_SUB\_TRG, 15
  - ENABLE\_TEST\_MODE, 15
  - LOAD\_DEF\_CONFIG, 15
  - LOAD\_USER\_CONFIG, 15
  - READ\_ACQ\_MOD, 15
  - READ\_ADJUST\_CH, 15
  - READ\_DEAD\_TIME, 15
  - READ\_DETECTION, 15
  - READ\_DLL\_LOCK, 15
  - READ\_EEPROM, 15
  - READ\_EN\_PATTERN, 15
  - READ\_EN\_PATTERN32, 15
  - READ\_ERROR\_STATUS, 15
  - READ\_ERROR\_TYPES, 15
  - READ\_EVENT\_SIZE, 15
  - READ\_FIFO\_SIZE, 16
  - READ\_GLOB\_OFFS, 16
  - READ\_HEAD\_TRAILER, 16
  - READ\_MICRO\_REV, 16
  - READ\_RC\_ADJ, 16
  - READ\_RES, 16
  - READ\_SETUP\_REG, 16
  - READ\_SETUP\_SCANPATH, 16
  - READ\_SPARE, 16
  - READ\_STATUS\_STREAM, 16
  - READ\_TDC\_ID, 16
  - READ\_TRG\_CONF, 16
  - RESET\_DLL\_PLL, 16
  - REV\_DATE\_MICRO\_FW, 16
  - SAVE\_RC\_ADJ, 16
  - SAVE\_USER\_CONFIG, 16
  - SET\_ADJUST\_CH, 16
  - SET\_DEAD\_TIME, 16
  - SET\_DETECTION, 16
  - SET\_DLL\_CLOCK, 16
  - SET\_ERROR\_TYPES, 16
  - SET\_EVENT\_SIZE, 16
  - SET\_FIFO\_SIZE, 16
  - SET\_GLOB\_OFFS, 16
  - SET\_KEEP\_TOKEN, 16
  - SET\_PAIR\_RES, 16
  - SET\_RC\_ADJ, 16
  - SET\_REJ\_MARGIN, 16
  - SET\_SW\_MARGIN, 17
  - SET\_TDC\_TSET\_OUTPUT, 17
  - SET\_TR\_LEAD\_LSB, 17
  - SET\_WIN\_OFFS, 17
  - SET\_WIN\_WIDTH, 17
  - TRG\_MATCH, 17
  - UPDATE\_SETUP\_REG, 17
  - UPDATE\_SETUP\_TDC, 17
  - WRITE\_EEPROM, 17
  - WRITE\_EN\_PATTERN, 17

- WRITE\_EN\_PATTERN32, [17](#)
- WRITE\_SETUP\_REG, [17](#)
- WRITE\_SPARE, [17](#)
- VME::TDCV1x90Status, [102](#)
  - ~TDCV1x90Status, [103](#)
  - AlmostFull, [103](#)
  - BusError, [103](#)
  - DataReady, [103](#)
  - Dump, [103](#)
  - Error, [104](#)
  - fWord, [105](#)
  - Full, [105](#)
  - GetValue, [105](#)
  - HeadersEnabled, [105](#)
  - PairMode, [105](#)
  - Purged, [105](#)
  - R\_100ps, [102](#)
  - R\_200ps, [102](#)
  - R\_25ps, [102](#)
  - R\_800ps, [102](#)
  - Resolution, [105](#)
  - TDCResolution, [102](#)
  - TDCV1x90Status, [103](#)
  - TerminationOn, [105](#)
  - TriggerLost, [105](#)
  - TriggerMatching, [105](#)
- VME::trailead\_t, [105](#)
  - ettt, [106](#)
  - event\_count, [106](#)
  - leading, [106](#)
  - total\_hits, [106](#)
  - trailing, [106](#)
- VMEReader, [106](#)
  - ~VMEReader, [108](#)
  - Abort, [109](#)
  - AddTDC, [109](#)
  - fBridge, [110](#)
  - flsPulserStarted, [110](#)
  - fOnSocket, [111](#)
  - fTDCCollection, [111](#)
  - GetRunNumber, [109](#)
  - GetTDC, [110](#)
  - StartPulser, [110](#)
  - StopPulser, [110](#)
  - TDCCollection, [108](#)
  - VMEReader, [108](#)
- WEBSOCKET\_CLIENT
  - Socket communication objects, [9](#)
- WIN\_OFFSET
  - VME, [13](#)
- WRITE\_EEPROM
  - VME::TDCV1x90Opcodes, [17](#)
- WRITE\_EN\_PATTERN
  - VME::TDCV1x90Opcodes, [17](#)
- WRITE\_EN\_PATTERN32
  - VME::TDCV1x90Opcodes, [17](#)
- WRITE\_OK
  - VME, [12](#)
- WRITE\_SETUP\_REG
  - VME::TDCV1x90Opcodes, [17](#)
- WRITE\_SPARE
  - VME::TDCV1x90Opcodes, [17](#)
- WaitMicro
  - VME::TDCV1x90, [94](#)
- WriteRegister
  - VME::BridgeVx718, [21](#)
  - VME::TDCV1x90, [94](#)