

2015 Test beam Run Control

Generated by Doxygen 1.6.1

Wed Jul 8 15:10:18 2015

Contents

1	Module Index	1
1.1	Modules	1
2	Namespace Index	3
2.1	Namespace List	3
3	Data Structure Index	5
3.1	Class Hierarchy	5
4	Data Structure Index	7
4.1	Data Structures	7
5	Module Documentation	9
5.1	Socket communication objects	9
6	Namespace Documentation	11
6.1	VME Namespace Reference	11
6.1.1	Typedef Documentation	13
6.1.1.1	TDCEventCollection	13
6.1.2	Enumeration Type Documentation	13
6.1.2.1	AcquisitionMode	13
6.1.2.2	BridgeType	13
6.1.2.3	DetectionMode	14
6.1.2.4	FPGAUnitV1495Register	14
6.1.2.5	IOModuleV262Register	15
6.1.2.6	micro_handshake	15
6.1.2.7	TDCV1x90Register	15

6.1.2.8	trailead_edge_lsb	16
6.1.2.9	trig_conf	16
6.2	VME::TDCV1x90Opcodes Namespace Reference	17
6.2.1	Function Documentation	20
6.2.1.1	AUTOLOAD_DEF_CONFI	20
6.2.1.2	AUTOLOAD_USER_CONF	20
6.2.1.3	CLEAR_KEEP_TOKEN	20
6.2.1.4	CONT_STOR	20
6.2.1.5	DEFAULT_SETUP_REG	20
6.2.1.6	DIS_ALL_CHANNEL	20
6.2.1.7	DIS_CHANNEL	20
6.2.1.8	DIS_ERROR_BYPASS	20
6.2.1.9	DIS_ERROR_MARK	20
6.2.1.10	DIS_HEAD_TRAILER	20
6.2.1.11	DIS_SUB_TRG	20
6.2.1.12	DISABLE_TEST_MODE	20
6.2.1.13	EN_ALL_CHANNEL	20
6.2.1.14	EN_CHANNEL	20
6.2.1.15	EN_ERROR_BYPASS	20
6.2.1.16	EN_ERROR_MARK	20
6.2.1.17	EN_HEAD_TRAILER	20
6.2.1.18	EN_SUB_TRG	20
6.2.1.19	ENABLE_TEST_MODE	20
6.2.1.20	LOAD_DEF_CONFIG	20
6.2.1.21	LOAD_USER_CONFIG	20
6.2.1.22	READ_ACQ_MOD	20
6.2.1.23	READ_ADJUST_CH	20
6.2.1.24	READ_DEAD_TIME	20
6.2.1.25	READ_DETECTION	20
6.2.1.26	READ_DLL_LOCK	20
6.2.1.27	READ_EEPROM	20
6.2.1.28	READ_EN_PATTERN	20
6.2.1.29	READ_EN_PATTERN32	20
6.2.1.30	READ_ERROR_STATUS	20

6.2.1.31	READ_ERROR_TYPES	20
6.2.1.32	READ_EVENT_SIZE	20
6.2.1.33	READ_FIFO_SIZE	20
6.2.1.34	READ_GLOB_OFFS	20
6.2.1.35	READ_HEAD_TRAILER	20
6.2.1.36	READ_MICRO_REV	20
6.2.1.37	READ_RC_ADJ	20
6.2.1.38	READ_RES	20
6.2.1.39	READ_SETUP_REG	20
6.2.1.40	READ_SETUP_SCANPATH	20
6.2.1.41	READ_SPARE	20
6.2.1.42	READ_STATUS_STREAM	20
6.2.1.43	READ_TDC_ID	20
6.2.1.44	READ_TRG_CONF	20
6.2.1.45	RESET_DLL_PLL	20
6.2.1.46	REV_DATE_MICRO_FW	20
6.2.1.47	SAVE_RC_ADJ	20
6.2.1.48	SAVE_USER_CONFIG	20
6.2.1.49	SET_ADJUST_CH	20
6.2.1.50	SET_DEAD_TIME	20
6.2.1.51	SET_DETECTION	20
6.2.1.52	SET_DLL_CLOCK	20
6.2.1.53	SET_ERROR_TYPES	20
6.2.1.54	SET_EVENT_SIZE	20
6.2.1.55	SET_FIFO_SIZE	20
6.2.1.56	SET_GLOB_OFFS	20
6.2.1.57	SET_KEEP_TOKEN	20
6.2.1.58	SET_PAIR_RES	20
6.2.1.59	SET_RC_ADJ	20
6.2.1.60	SET_REJ_MARGIN	20
6.2.1.61	SET_SW_MARGIN	20
6.2.1.62	SET_TDC_TSET_OUTPUT	20
6.2.1.63	SET_TR_LEAD_LSB	20
6.2.1.64	SET_WIN_OFFS	20

6.2.1.65	SET_WIN_WIDTH	20
6.2.1.66	TRG_MATCH	20
6.2.1.67	UPDATE_SETUP_REG	20
6.2.1.68	UPDATE_SETUP_TDC	20
6.2.1.69	WRITE_EEPROM	20
6.2.1.70	WRITE_EN_PATTERN	20
6.2.1.71	WRITE_EN_PATTERN32	20
6.2.1.72	WRITE_SETUP_REG	20
6.2.1.73	WRITE_SPARE	20
7	Data Structure Documentation	21
7.1	VME::BridgeVx718 Class Reference	21
7.1.1	Detailed Description	22
7.1.2	Member Enumeration Documentation	22
7.1.2.1	IRQId	22
7.1.3	Constructor & Destructor Documentation	23
7.1.3.1	BridgeVx718	23
7.1.3.2	~BridgeVx718	23
7.1.4	Member Function Documentation	23
7.1.4.1	CheckConfiguration	23
7.1.4.2	GetHandle	23
7.1.4.3	GetIRQStatus	23
7.1.4.4	InputConf	23
7.1.4.5	InputRead	23
7.1.4.6	OutputConf	23
7.1.4.7	OutputOff	24
7.1.4.8	OutputOn	24
7.1.4.9	SetIRQ	24
7.1.4.10	SinglePulse	24
7.1.4.11	StartPulser	24
7.1.4.12	StopPulser	24
7.1.4.13	TestOutputs	24
7.1.4.14	WaitIRQ	24
7.1.5	Field Documentation	24

7.1.5.1	fHasIRQ	24
7.2	VME::BridgeVx718Control Class Reference	25
7.2.1	Constructor & Destructor Documentation	25
7.2.1.1	BridgeVx718Control	25
7.2.1.2	~BridgeVx718Control	25
7.2.2	Member Function Documentation	25
7.2.2.1	GetAddressIncrement	25
7.2.2.2	GetArbiterType	26
7.2.2.3	GetBusReqLevel	26
7.2.2.4	GetBusTimeout	26
7.2.2.5	GetInterruptReq	26
7.2.2.6	GetReleaseType	26
7.2.2.7	GetRequesterType	26
7.2.2.8	GetSysRes	27
7.2.3	Field Documentation	27
7.2.3.1	fWord	27
7.3	VME::BridgeVx718Status Class Reference	28
7.3.1	Constructor & Destructor Documentation	29
7.3.1.1	BridgeVx718Status	29
7.3.1.2	~BridgeVx718Status	29
7.3.2	Member Function Documentation	29
7.3.2.1	Dump	29
7.3.2.2	GetBERR	29
7.3.2.3	GetDipSwitch	29
7.3.2.4	GetDTACK	29
7.3.2.5	GetSystemControl	29
7.3.2.6	GetSystemReset	29
7.3.2.7	GetUSBType	29
7.3.3	Field Documentation	29
7.3.3.1	fWord	29
7.4	Client Class Reference	30
7.4.1	Detailed Description	31
7.4.2	Constructor & Destructor Documentation	31
7.4.2.1	Client	31

7.4.2.2	Client	31
7.4.2.3	~Client	31
7.4.3	Member Function Documentation	31
7.4.3.1	Announce	31
7.4.3.2	Connect	31
7.4.3.3	Disconnect	32
7.4.3.4	GetType	32
7.4.3.5	ParseMessage	32
7.4.3.6	Receive	32
7.4.3.7	Send	32
7.4.3.8	Send	32
7.4.3.9	SendAndReceive	32
7.4.4	Field Documentation	32
7.4.4.1	fClientId	32
7.4.4.2	fIsConnected	32
7.4.4.3	fType	32
7.5	Exception Class Reference	34
7.5.1	Detailed Description	34
7.5.2	Constructor & Destructor Documentation	35
7.5.2.1	Exception	35
7.5.2.2	Exception	35
7.5.2.3	~Exception	35
7.5.3	Member Function Documentation	35
7.5.3.1	Description	35
7.5.3.2	Dump	35
7.5.3.3	ErrorNumber	35
7.5.3.4	From	35
7.5.3.5	OneLine	35
7.5.3.6	Type	35
7.5.3.7	TypeString	35
7.5.4	Field Documentation	35
7.5.4.1	fDescription	35
7.5.4.2	fErrorNumber	35
7.5.4.3	fFrom	35

7.5.4.4	fType	35
7.6	file_header_t Struct Reference	37
7.6.1	Detailed Description	37
7.6.2	Field Documentation	37
7.6.2.1	acq_mode	37
7.6.2.2	det_mode	37
7.6.2.3	magic	37
7.6.2.4	num_hptdc	37
7.6.2.5	run_id	37
7.6.2.6	spill_id	37
7.7	FileReader Class Reference	38
7.7.1	Detailed Description	38
7.7.2	Constructor & Destructor Documentation	38
7.7.2.1	FileReader	38
7.7.2.2	~FileReader	39
7.7.3	Member Function Documentation	39
7.7.3.1	GetNextEvent	39
7.7.3.2	GetNextMeasurement	39
7.7.3.3	GetNumEvents	39
7.7.3.4	GetNumTDCs	39
7.7.4	Field Documentation	39
7.7.4.1	fFile	39
7.7.4.2	fHeader	39
7.7.4.3	fNumEvents	39
7.7.4.4	fReadoutMode	39
7.8	VME::FPGAUnitV1495 Class Reference	40
7.8.1	Detailed Description	41
7.8.2	Member Enumeration Documentation	41
7.8.2.1	TDCBits	41
7.8.3	Constructor & Destructor Documentation	41
7.8.3.1	FPGAUnitV1495	41
7.8.3.2	~FPGAUnitV1495	41
7.8.4	Member Function Documentation	41
7.8.4.1	CheckBoardVersion	41

7.8.4.2	DumpFWInformation	41
7.8.4.3	GetCAENFirmwareRevision	42
7.8.4.4	GetControl	42
7.8.4.5	GetGeoAddress	42
7.8.4.6	GetHardwareRevision	42
7.8.4.7	GetInternalClockPeriod	42
7.8.4.8	GetInternalTriggerPeriod	42
7.8.4.9	GetOutputPulser	42
7.8.4.10	GetSerialNumber	42
7.8.4.11	GetTDCBits	43
7.8.4.12	GetUserFirmwareRevision	43
7.8.4.13	PulseTDCBits	43
7.8.4.14	ResetFPGA	43
7.8.4.15	SetControl	43
7.8.4.16	SetInternalClockPeriod	43
7.8.4.17	SetInternalTriggerPeriod	44
7.8.4.18	SetOutputPulser	44
7.8.4.19	SetTDCBits	44
7.9	VME::FPGAUnitV1495Control Class Reference	45
7.9.1	Detailed Description	45
7.9.2	Member Enumeration Documentation	46
7.9.2.1	ClockSource	46
7.9.2.2	TriggerSource	46
7.9.3	Constructor & Destructor Documentation	46
7.9.3.1	FPGAUnitV1495Control	46
7.9.3.2	~FPGAUnitV1495Control	46
7.9.4	Member Function Documentation	46
7.9.4.1	GetClockSource	46
7.9.4.2	GetTriggerSource	46
7.9.4.3	GetWord	46
7.9.4.4	SetClockSource	46
7.9.4.5	SetTriggerSource	47
7.9.5	Field Documentation	47
7.9.5.1	fWord	47

7.10 VME::GenericBoard< Register, am > Class Template Reference . . .	48
7.10.1 Constructor & Destructor Documentation	49
7.10.1.1 GenericBoard	49
7.10.1.2 ~GenericBoard	49
7.10.2 Member Function Documentation	49
7.10.2.1 ReadRegister	49
7.10.2.2 ReadRegister	49
7.10.2.3 WriteRegister	49
7.10.2.4 WriteRegister	50
7.10.3 Field Documentation	50
7.10.3.1 fBaseAddr	50
7.10.3.2 fHandle	50
7.11 VME::GlobalOffset Struct Reference	51
7.11.1 Field Documentation	51
7.11.1.1 coarse	51
7.11.1.2 fine	51
7.12 HTTPMessage Class Reference	52
7.12.1 Detailed Description	52
7.12.2 Constructor & Destructor Documentation	52
7.12.2.1 HTTPMessage	52
7.12.2.2 HTTPMessage	53
7.12.3 Member Function Documentation	53
7.12.3.1 Decode	53
7.12.3.2 Dump	53
7.12.3.3 Encode	53
7.12.3.4 GetKey	53
7.12.4 Field Documentation	53
7.12.4.1 fOriginalString	53
7.12.4.2 fWS	53
7.13 VME::IOModuleV262 Class Reference	54
7.13.1 Constructor & Destructor Documentation	54
7.13.1.1 IOModuleV262	54
7.13.1.2 ~IOModuleV262	54
7.13.2 Member Function Documentation	54

7.13.2.1	GetIdentifier	54
7.13.2.2	GetManufacturerId	54
7.13.2.3	GetModuleType	54
7.13.2.4	GetModuleVersion	54
7.13.2.5	GetSerialNumber	55
7.14	Message Class Reference	56
7.14.1	Detailed Description	56
7.14.2	Constructor & Destructor Documentation	57
7.14.2.1	Message	57
7.14.2.2	Message	57
7.14.2.3	Message	57
7.14.2.4	~Message	57
7.14.3	Member Function Documentation	57
7.14.3.1	Dump	57
7.14.3.2	GetKey	57
7.14.3.3	GetString	57
7.14.3.4	IsFromWeb	57
7.14.4	Field Documentation	57
7.14.4.1	fString	57
7.15	Messenger Class Reference	59
7.15.1	Detailed Description	60
7.15.2	Constructor & Destructor Documentation	60
7.15.2.1	Messenger	60
7.15.2.2	Messenger	60
7.15.2.3	~Messenger	60
7.15.3	Member Function Documentation	61
7.15.3.1	AddClient	61
7.15.3.2	Broadcast	61
7.15.3.3	Connect	61
7.15.3.4	Disconnect	61
7.15.3.5	DisconnectClient	61
7.15.3.6	GetType	61
7.15.3.7	ProcessMessage	62
7.15.3.8	Receive	62

7.15.3.9	Send	62
7.15.3.10	StartAcquisition	62
7.15.3.11	StopAcquisition	62
7.15.3.12	SwitchClientType	62
7.15.4	Field Documentation	63
7.15.4.1	fNumAttempts	63
7.15.4.2	fPID	63
7.15.4.3	fStderrPipe	63
7.15.4.4	fStdoutPipe	63
7.15.4.5	fWS	63
7.16	Socket Class Reference	64
7.16.1	Detailed Description	65
7.16.2	Member Typedef Documentation	66
7.16.2.1	SocketCollection	66
7.16.3	Member Enumeration Documentation	66
7.16.3.1	SocketType	66
7.16.4	Constructor & Destructor Documentation	66
7.16.4.1	Socket	66
7.16.4.2	Socket	66
7.16.4.3	~Socket	66
7.16.5	Member Function Documentation	66
7.16.5.1	AcceptConnections	66
7.16.5.2	Bind	66
7.16.5.3	Configure	67
7.16.5.4	Create	67
7.16.5.5	DumpConnected	67
7.16.5.6	FetchMessage	67
7.16.5.7	GetPort	67
7.16.5.8	GetSocketId	67
7.16.5.9	GetSocketType	67
7.16.5.10	IsWebSocket	67
7.16.5.11	Listen	67
7.16.5.12	PrepareConnection	67
7.16.5.13	SelectConnections	68

7.16.5.14	SendMessage	68
7.16.5.15	SetPort	68
7.16.5.16	SetSocketId	68
7.16.5.17	Start	68
7.16.5.18	Stop	68
7.16.6	Field Documentation	68
7.16.6.1	fAddress	68
7.16.6.2	fBuffer	68
7.16.6.3	fMaster	68
7.16.6.4	fPort	68
7.16.6.5	fReadFds	68
7.16.6.6	fSocketId	69
7.16.6.7	fSocketsConnected	69
7.17	SocketMessage Class Reference	70
7.17.1	Detailed Description	71
7.17.2	Constructor & Destructor Documentation	71
7.17.2.1	SocketMessage	71
7.17.2.2	SocketMessage	71
7.17.2.3	SocketMessage	72
7.17.2.4	SocketMessage	72
7.17.2.5	SocketMessage	72
7.17.2.6	SocketMessage	72
7.17.2.7	SocketMessage	72
7.17.2.8	SocketMessage	72
7.17.2.9	SocketMessage	72
7.17.2.10	SocketMessage	72
7.17.2.11	SocketMessage	73
7.17.2.12	~SocketMessage	73
7.17.3	Member Function Documentation	73
7.17.3.1	Dump	73
7.17.3.2	GetIntValue	73
7.17.3.3	GetKey	73
7.17.3.4	GetString	73
7.17.3.5	GetValue	73

7.17.3.6	GetVectorValue	73
7.17.3.7	Object	74
7.17.3.8	SetKeyValue	74
7.17.3.9	SetKeyValue	74
7.17.3.10	SetKeyValue	74
7.17.3.11	SetKeyValue	74
7.17.3.12	String	74
7.17.4	Field Documentation	74
7.17.4.1	fMessage	74
7.18	VME::TDCErrorFlag Class Reference	75
7.18.1	Detailed Description	75
7.18.2	Constructor & Destructor Documentation	76
7.18.2.1	TDCErrorFlag	76
7.18.2.2	~TDCErrorFlag	76
7.18.3	Member Function Documentation	76
7.18.3.1	Dump	76
7.18.3.2	GetWord	76
7.18.3.3	HasGroupError	76
7.18.3.4	HasInternalChipError	76
7.18.3.5	HasL1BufferOverflow	76
7.18.3.6	HasReachedEventSizeLimit	76
7.18.3.7	HasReadoutFIFOOverflow	76
7.18.3.8	HasTriggerFIFOOverflow	76
7.18.4	Friends And Related Function Documentation	77
7.18.4.1	operator<<	77
7.18.5	Field Documentation	77
7.18.5.1	fWord	77
7.19	VME::TDCEvent Class Reference	78
7.19.1	Detailed Description	79
7.19.2	Member Enumeration Documentation	79
7.19.2.1	EventType	79
7.19.3	Constructor & Destructor Documentation	80
7.19.3.1	TDCEvent	80
7.19.3.2	TDCEvent	80

7.19.3.3	TDCEvent	80
7.19.3.4	~TDCEvent	80
7.19.4	Member Function Documentation	80
7.19.4.1	Dump	80
7.19.4.2	GetBunchId	80
7.19.4.3	GetChannelId	80
7.19.4.4	GetErrorFlags	80
7.19.4.5	GetETTT	80
7.19.4.6	GetEventCount	81
7.19.4.7	GetEventId	81
7.19.4.8	GetGeo	81
7.19.4.9	GetLeadingTime	81
7.19.4.10	GetStatus	81
7.19.4.11	GetTDCId	81
7.19.4.12	GetTrailingTime	81
7.19.4.13	GetType	81
7.19.4.14	GetWidth	82
7.19.4.15	GetWord	82
7.19.4.16	GetWordCount	82
7.19.4.17	IsTrailing	82
7.19.4.18	SetWord	82
7.19.5	Field Documentation	82
7.19.5.1	fWord	82
7.20	VME::TDCMeasurement Class Reference	83
7.20.1	Constructor & Destructor Documentation	83
7.20.1.1	TDCMeasurement	83
7.20.1.2	TDCMeasurement	83
7.20.1.3	~TDCMeasurement	83
7.20.2	Member Function Documentation	83
7.20.2.1	Dump	83
7.20.2.2	GetBunchId	84
7.20.2.3	GetChannelId	84
7.20.2.4	GetEventId	84
7.20.2.5	GetLeadingTime	84

7.20.2.6	GetTDCId	84
7.20.2.7	GetToT	84
7.20.2.8	GetTrailingTime	84
7.20.2.9	NumEvents	84
7.20.2.10	SetEventsCollection	84
7.20.3	Field Documentation	84
7.20.3.1	fEvents	84
7.20.3.2	fMap	84
7.21	VME::TDCV1x90 Class Reference	85
7.21.1	Detailed Description	87
7.21.2	Member Enumeration Documentation	87
7.21.2.1	DLLMode	87
7.21.3	Constructor & Destructor Documentation	87
7.21.3.1	TDCV1x90	87
7.21.3.2	~TDCV1x90	87
7.21.4	Member Function Documentation	87
7.21.4.1	abort	87
7.21.4.2	CheckConfiguration	87
7.21.4.3	DisableChannel	87
7.21.4.4	EnableChannel	87
7.21.4.5	FetchEvents	88
7.21.4.6	GetAcquisitionMode	88
7.21.4.7	GetBLTEventNumberRegister	88
7.21.4.8	GetChannelDeadTime	88
7.21.4.9	GetControl	88
7.21.4.10	GetDetectionMode	88
7.21.4.11	GetDLLClock	88
7.21.4.12	GetErrorMarks	88
7.21.4.13	GetETTT	88
7.21.4.14	GetEventCounter	88
7.21.4.15	GetEventStored	88
7.21.4.16	GetFIFOSize	89
7.21.4.17	GetFirmwareRevision	89
7.21.4.18	GetGlobalOffset	89

7.21.4.19 GetModel	89
7.21.4.20 GetOUI	89
7.21.4.21 GetPoI	89
7.21.4.22 GetRCAdjust	89
7.21.4.23 GetResolution	89
7.21.4.24 GetSerialNumber	89
7.21.4.25 GetStatus	89
7.21.4.26 GetTDCEncapsulation	90
7.21.4.27 GetTestMode	90
7.21.4.28 GetTriggerConfiguration	90
7.21.4.29 GetWindowOffset	90
7.21.4.30 GetWindowWidth	90
7.21.4.31 HardwareReset	90
7.21.4.32 ReadAcquisitionMode	90
7.21.4.33 ReadDetectionMode	90
7.21.4.34 SetAcquisitionMode	90
7.21.4.35 SetBLTEventNumberRegister	90
7.21.4.36 SetChannelDeadTime	90
7.21.4.37 SetContinuousStorage	90
7.21.4.38 SetControl	91
7.21.4.39 SetDetectionMode	91
7.21.4.40 SetDLLClock	91
7.21.4.41 SetErrorMarks	91
7.21.4.42 SetETTT	91
7.21.4.43 SetFIFOSize	91
7.21.4.44 SetGlobalOffset	91
7.21.4.45 SetLSBTraileadEdge	91
7.21.4.46 SetPairModeResolution	91
7.21.4.47 SetPoI	92
7.21.4.48 SetRCAdjust	92
7.21.4.49 SetStatus	92
7.21.4.50 SetTDCEncapsulation	92
7.21.4.51 SetTestMode	92
7.21.4.52 SetTriggerMatching	92

7.21.4.53	SetVerboseLevel	92
7.21.4.54	SetWindowOffset	92
7.21.4.55	SetWindowWidth	92
7.21.4.56	SoftwareClear	93
7.21.4.57	SoftwareReset	93
7.21.4.58	WaitMicro	93
7.21.5	Field Documentation	93
7.21.5.1	fAcquisitionMode	93
7.21.5.2	fBuffer	93
7.21.5.3	fDetectionMode	93
7.21.5.4	fErrorMarks	93
7.21.5.5	fVerb	93
7.21.5.6	fWindowWidth	93
7.21.5.7	gEnd	93
7.21.5.8	nchannels	93
7.21.5.9	pair_lead_res	93
7.21.5.10	pair_width_res	93
7.22	VME::TDCV1x90Control Class Reference	94
7.22.1	Detailed Description	94
7.22.2	Constructor & Destructor Documentation	95
7.22.2.1	TDCV1x90Control	95
7.22.2.2	~TDCV1x90Control	95
7.22.3	Member Function Documentation	95
7.22.3.1	Dump	95
7.22.3.2	GetAlign64	95
7.22.3.3	GetBusError	95
7.22.3.4	GetCompensation	95
7.22.3.5	GetEmptyEvent	95
7.22.3.6	GetETTT	95
7.22.3.7	GetEventFIFO	95
7.22.3.8	GetMEBAccess	95
7.22.3.9	GetSRAMCompensation	95
7.22.3.10	GetSWTermination	95
7.22.3.11	GetTermination	95

7.22.3.12	GetTestFIFO	95
7.22.3.13	GetValue	95
7.22.3.14	SetAlign64	95
7.22.3.15	SetBusError	96
7.22.3.16	SetCompensation	96
7.22.3.17	SetEmptyEvent	96
7.22.3.18	SetETTT	96
7.22.3.19	SetEventFIFO	96
7.22.3.20	SetMEBAccess	96
7.22.3.21	SetSRAMCompensation	96
7.22.3.22	SetSWTermination	96
7.22.3.23	SetTermination	96
7.22.3.24	SetTestFIFO	97
7.22.4	Field Documentation	97
7.22.4.1	fWord	97
7.23	VME::TDCV1x90Status Class Reference	98
7.23.1	Detailed Description	98
7.23.2	Member Enumeration Documentation	99
7.23.2.1	TDCResolution	99
7.23.3	Constructor & Destructor Documentation	99
7.23.3.1	TDCV1x90Status	99
7.23.3.2	~TDCV1x90Status	99
7.23.4	Member Function Documentation	99
7.23.4.1	AlmostFull	99
7.23.4.2	BusError	99
7.23.4.3	DataReady	99
7.23.4.4	Dump	99
7.23.4.5	Error	99
7.23.4.6	Error	100
7.23.4.7	Full	100
7.23.4.8	GetValue	100
7.23.4.9	HeadersEnabled	100
7.23.4.10	PairMode	100
7.23.4.11	Purged	100

7.23.4.12 Resolution	100
7.23.4.13 TerminationOn	100
7.23.4.14 TriggerLost	100
7.23.4.15 TriggerMatching	100
7.23.5 Field Documentation	100
7.23.5.1 fWord	100
7.24 VME::trailead_t Struct Reference	101
7.24.1 Field Documentation	101
7.24.1.1 ettt	101
7.24.1.2 event_count	101
7.24.1.3 leading	101
7.24.1.4 total_hits	101
7.24.1.5 trailing	101
7.25 VMEReader Class Reference	102
7.25.1 Detailed Description	103
7.25.2 Member Typedef Documentation	103
7.25.2.1 TDCCollection	103
7.25.3 Constructor & Destructor Documentation	103
7.25.3.1 VMEReader	103
7.25.3.2 ~VMEReader	103
7.25.4 Member Function Documentation	104
7.25.4.1 Abort	104
7.25.4.2 AddFPGAUnit	104
7.25.4.3 AddIOModule	104
7.25.4.4 AddTDC	104
7.25.4.5 GetFPGAUnit	104
7.25.4.6 GetIOModule	104
7.25.4.7 GetRunNumber	104
7.25.4.8 GetTDC	104
7.25.4.9 SendClear	104
7.25.4.10 SendPulse	105
7.25.4.11 StartPulser	105
7.25.4.12 StopPulser	105
7.25.5 Field Documentation	105

7.25.5.1	fBridge	105
7.25.5.2	fFPGA	105
7.25.5.3	fIsPulserStarted	105
7.25.5.4	fOnSocket	105
7.25.5.5	fSG	105
7.25.5.6	fTDCCollection	105

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:

VME	11
VME::TDCV1x90OpCodes	17

Chapter 3

Data Structure Index

3.1 Class Hierarchy

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

VME::BridgeVx718Control	25
VME::BridgeVx718Status	28
Exception	34
file_header_t	37
FileReader	38
VME::FPGAUnitV1495Control	45
VME::GenericBoard< Register, am >	48
VME::GenericBoard< CVRegisters, cvA32_U_DATA >	48
VME::BridgeVx718	21
VME::GenericBoard< FPGAUnitV1495Register, cvA32_U_DATA >	48
VME::FPGAUnitV1495	40
VME::GenericBoard< IOModuleV262Register, cvA24_U_DATA >	48
VME::IOModuleV262	54
VME::GenericBoard< TDCV1x90Register, cvA32_U_DATA >	48
VME::TDCV1x90	85
VME::GlobalOffset	51
Message	56
HTTPMessage	52
SocketMessage	70
Socket	64
Client	30
VMEReader	102
Messenger	59
VME::TDCErrorFlag	75
VME::TDCEvent	78
VME::TDCMeasurement	83
VME::TDCV1x90Control	94

VME::TDCV1x90Status	98
VME::trailead_t	101

Chapter 4

Data Structure Index

4.1 Data Structures

Here are the data structures with brief descriptions:

VME::BridgeVx718 (Class defining the VME bridge)	21
VME::BridgeVx718Control	25
VME::BridgeVx718Status	28
Client (Base client object for the socket)	30
Exception (A simple exception handler)	34
file_header_t (Header to the output files)	37
FileReader (Handler for a TDC output file readout)	38
VME::FPGAUnitV1495	40
VME::FPGAUnitV1495Control	45
VME::GenericBoard< Register , am >	48
VME::GlobalOffset	51
HTTPMessage (Message to be transmitted through a WebSocket protocol)	52
VME::IOModuleV262	54
Message (Base socket message type)	56
Messenger (Base master object for the socket)	59
Socket (Base socket object from which clients/master from a socket inherit)	64
SocketMessage (Socket-passed message type)	70
VME::TDCErrorFlag (Error flags handler)	75
VME::TDCEvent (HPTDC event parser)	78
VME::TDCMeasurement	83
VME::TDCV1x90	85
VME::TDCV1x90Control (TDC control register)	94
VME::TDCV1x90Status (TDC status register)	98
VME::trailead_t	101
VMEReader	102

Chapter 5

Module Documentation

5.1 Socket communication objects

Data Structures

- class [Client](#)
Base client object for the socket.
- class [HTTPMessage](#)
Message to be transmitted through a WebSocket protocol.
- class [Messenger](#)
Base master object for the socket.
- class [Socket](#)
Base socket object from which clients/master from a socket inherit.
- class [SocketMessage](#)
Socket-passed message type.

Chapter 6

Namespace Documentation

6.1 VME Namespace Reference

Namespaces

- namespace [TDCV1x90Opcodes](#)

Data Structures

- class [BridgeVx718Status](#)
- class [BridgeVx718Control](#)
- class [BridgeVx718](#)
class defining the VME bridge
- class [FPGAUnitV1495Control](#)
- class [FPGAUnitV1495](#)
- class [GenericBoard](#)
- class [IOModuleV262](#)
- class [TDCErrorFlag](#)
Error flags handler.
- class [TDCEvent](#)
HPTDC event parser.
- class [TDCMeasurement](#)
- struct [GlobalOffset](#)
- struct [trailead_t](#)
- class [TDCV1x90Status](#)
TDC status register.
- class [TDCV1x90Control](#)

TDC control register.

- class [TDCV1x90](#)

Typedefs

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

Enumerations

- enum [BridgeType](#) { [CAEN_V1718](#), [CAEN_V2718](#) }

Compatible bridge types.

- enum [FPGAUnitV1495Register](#) {
[kV1495UserFWRevision](#) = 0x100c, [kV1495TDCBoardInterface](#) = 0x1018,
[kV1495ClockSettings](#) = 0x101c, [kV1495Control](#) = 0x1020,
[kV1495TriggerSettings](#) = 0x1024, [kV1495OutputSettings](#) = 0x1028,
[kV1495GeoAddress](#) = 0x8008, [kV1495UserFPGAFlashMem](#) = 0x8014,
[kV1495UserFPGAConfig](#) = 0x8016, [kV1495ModuleReset](#) = 0x800a,
[kV1495FWRevision](#) = 0x800c, [kV1495ConfigurationROM](#) = 0x8100,
[kV1495OUI2](#) = 0x8124, [kV1495OUI1](#) = 0x8128, [kV1495OUI0](#) = 0x812c,
[kV1495Board2](#) = 0x8134,
[kV1495Board1](#) = 0x8138, [kV1495Board0](#) = 0x813c, [kV1495HWRevision3](#) =
0x8140, [kV1495HWRevision2](#) = 0x8144,
[kV1495HWRevision1](#) = 0x8148, [kV1495HWRevision0](#) = 0x814c,
[kV1495SerNum0](#) = 0x8180, [kV1495SerNum1](#) = 0x8184 }
- enum [IOModuleV262Register](#) {
[kECLLevelWrite](#) = 0x04, [kNIMLevelWrite](#) = 0x06, [kNIMPulseWrite](#) = 0x08,
[kNIMPulseRead](#) = 0x0a,
[kIdentifier](#) = 0xfa, [kBoardInfo0](#) = 0xfc, [kBoardInfo1](#) = 0xfe }
- enum [AcquisitionMode](#) { [CONT_STORAGE](#), [TRIG_MATCH](#) }
TDC acquisition mode.
- enum [DetectionMode](#) { [PAIR](#) = 0x0, [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_lsb](#) { [r800ps](#) = 0, [r200ps](#) = 1, [r100ps](#) = 2, [r25ps](#) = 3 }
- enum [micro_handshake](#) { [WRITE_OK](#) = 0, [READ_OK](#) = 1 }

- enum `TDCV1x90Register` {
 `kOutputBuffer` = 0x0000, `kControl` = 0x1000, `kStatus` = 0x1002, `kInterruptLevel` = 0x100a,
 `kInterruptVector` = 0x100c, `kGeoAddress` = 0x100e, `kMCSTBase` = 0x1010, `kMCSTControl` = 0x1012,
 `kModuleReset` = 0x1014, `kSoftwareClear` = 0x1016, `kEventCounter` = 0x101c, `kEventStored` = 0x1020,
 `kBLTEventNumber` = 0x1024, `kFirmwareRev` = 0x1026, `kMicro` = 0x102e, `kMicroHandshake` = 0x1030,
 `kEventFIFO` = 0x1038, `kEventFIFOStoredRegister` = 0x103c, `kEventFIFOStatusRegister` = 0x103e, `kROMOui2` = 0x4024,
 `kROMOui1` = 0x4028, `kROMOui0` = 0x402c, `kROMBoard2` = 0x4034, `kROMBoard1` = 0x4038,
 `kROMBoard0` = 0x403c, `kROMRevis3` = 0x4040, `kROMRevis2` = 0x4044, `kROMRevis1` = 0x4048,
 `kROMRevis0` = 0x404c, `kROMSerNum1` = 0x4080, `kROMSerNum0` = 0x4084
}

6.1.1 Typedef Documentation

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

6.1.2 Enumeration Type Documentation

6.1.2.1 enum VME::AcquisitionMode

TDC acquisition mode.

Author:

Laurent Forthomme <laurent.forthomme@cern.ch>

Enumerator:

CONT_STORAGE

TRIG_MATCH

6.1.2.2 enum VME::BridgeType

Compatible bridge types.

Enumerator:

CAEN_V1718

CAEN_V2718

6.1.2.3 enum VME::DetectionMode

Enumerator:

PAIR
OTRAILING
OLEADING
TRAILEAD

6.1.2.4 enum VME::FPGAUnitV1495Register

Enumerator:

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

6.1.2.5 enum VME::IOModuleV262Register

Enumerator:

- kECLLevelWrite*
- kNIMLevelWrite*
- kNIMPulseWrite*
- kNIMPulseRead*
- kIdentifier*
- kBoardInfo0*
- kBoardInfo1*

6.1.2.6 enum VME::micro_handshake

Enumerator:

- WRITE_OK* Is the TDC ready for writing?
- READ_OK* Is the TDC ready for reading?

6.1.2.7 enum VME::TDCV1x90Register

Enumerator:

- kOutputBuffer*
- kControl*
- kStatus*
- kInterruptLevel*
- kInterruptVector*
- kGeoAddress*
- kMCSTBase*
- kMCSTControl*
- kModuleReset*
- kSoftwareClear*
- kEventCounter*
- kEventStored*
- kBLTEventNumber*
- kFirmwareRev*
- kMicro*
- kMicroHandshake*
- kEventFIFO*
- kEventFIFOStoredRegister*

kEventFIFOStatusRegister

kROMOui2

kROMOui1

kROMOui0

kROMBoard2

kROMBoard1

kROMBoard0

kROMRevis3

kROMRevis2

kROMRevis1

kROMRevis0

kROMSerNum1

kROMSerNum0

6.1.2.8 enum VME::trailead_edge_lsb

Enumerator:

r800ps

r200ps

r100ps

r25ps

6.1.2.9 enum VME::trig_conf

Enumerator:

MATCH_WIN_WIDTH

WIN_OFFSET

EXTRA_SEARCH_WIN_WIDTH

REJECT_MARGIN

TRIG_TIME_SUB

6.2 VME::TDCV1x90Opcodes Namespace Reference

Functions

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

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

6.2.1 Function Documentation

- 6.2.1.1 Opcode VME::TDCV1x90Opcodes::AUTOLOAD_DEF_CONFI (0x0900)
- 6.2.1.2 Opcode VME::TDCV1x90Opcodes::AUTOLOAD_USER_CONF (0x0800)
- 6.2.1.3 Opcode VME::TDCV1x90Opcodes::CLEAR_KEEP_TOKEN (0x0400)
- 6.2.1.4 Opcode VME::TDCV1x90Opcodes::CONT_STOR (0x0100)
- 6.2.1.5 Opcode VME::TDCV1x90Opcodes::DEFAULT_SETUP_REG (0x7300)
- 6.2.1.6 Opcode VME::TDCV1x90Opcodes::DIS_ALL_CHANNEL (0x4300)
- 6.2.1.7 Opcode VME::TDCV1x90Opcodes::DIS_CHANNEL (0x4100)
- 6.2.1.8 Opcode VME::TDCV1x90Opcodes::DIS_ERROR_BYPASS (0x3800)
- 6.2.1.9 Opcode VME::TDCV1x90Opcodes::DIS_ERROR_MARK (0x3600)
- 6.2.1.10 Opcode VME::TDCV1x90Opcodes::DIS_HEAD_TRAILER (0x3100)
- 6.2.1.11 Opcode VME::TDCV1x90Opcodes::DIS_SUB_TRG (0x1500)
- 6.2.1.12 Opcode VME::TDCV1x90Opcodes::DISABLE_TEST_MODE (0xc600)
- 6.2.1.13 Opcode VME::TDCV1x90Opcodes::EN_ALL_CHANNEL (0x4200)
- 6.2.1.14 Opcode VME::TDCV1x90Opcodes::EN_CHANNEL (0x4000)
- 6.2.1.15 Opcode VME::TDCV1x90Opcodes::EN_ERROR_BYPASS (0x3700)
- 6.2.1.16 Opcode VME::TDCV1x90Opcodes::EN_ERROR_MARK (0x3500)
- 6.2.1.17 Opcode VME::TDCV1x90Opcodes::EN_HEAD_TRAILER (0x3000)
- 6.2.1.18 Opcode VME::TDCV1x90Opcodes::EN_SUB_TRG (0x1400)
- 6.2.1.19 Opcode VME::TDCV1x90Opcodes::ENABLE_TEST_MODE (0xc500)
- 6.2.1.20 Opcode VME::TDCV1x90Opcodes::LOAD_DEF_CONFIG (0x0500)
- 6.2.1.21 Opcode VME::TDCV1x90Opcodes::LOAD_USER_CONFIG (0x0700)
- 6.2.1.22 Opcode VME::TDCV1x90Opcodes::READ_ACQ_MOD (0x0200)
- 6.2.1.23 Opcode VME::TDCV1x90Opcodes::READ_ADJUST_CH (0x5200)
- 6.2.1.24 Opcode VME::TDCV1x90Opcodes::READ_DEAD_TIME (0x2900)
- 6.2.1.25 Opcode VME::TDCV1x90Opcodes::READ_DETECTION (0x2300)
- 6.2.1.26 Opcode VME::TDCV1x90Opcodes::READ_DLL_LOCK (0x7500)
- 6.2.1.27 Opcode VME::TDCV1x90Opcodes::READ_EEPROM (0xc100)

Chapter 7

Data Structure Documentation

7.1 VME::BridgeVx718 Class Reference

class defining the [VME](#) bridge

`#include <VME_BridgeVx718.h>`[Inheritance diagram](#) for [VME::BridgeVx718](#):[Collaboration diagram for VME::BridgeVx718](#):

Public Types

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

Public Member Functions

- [BridgeVx718](#) (const char *device, [BridgeType](#) type)
 Constructor.
- [~BridgeVx718](#) ()
 Destructor.
- int32_t [GetHandle](#) () const
 Bridge's handle value.
- void [CheckConfiguration](#) () const
- void [TestOutputs](#) () const
- void [SetIRQ](#) (unsigned int irq, bool enable=true)
- void [WaitIRQ](#) (unsigned int irq, unsigned long timeout=1000) const
- unsigned int [GetIRQStatus](#) () const
- void [OutputConf](#) (CVOutputSelect output) const

Set and control the output lines.

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

Set and read the input lines.

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

Private Attributes

- bool [fHasIRQ](#)

7.1.1 Detailed Description

class defining the [VME](#) bridge This class initializes the CAEN V1718 [VME](#) bridge in order to control the crate.

Author:

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

Date:

Jun 2010

7.1.2 Member Enumeration Documentation

7.1.2.1 enum VME::BridgeVx718::IRQId

Enumerator:

IRQ1
IRQ2
IRQ3
IRQ4
IRQ5
IRQ6
IRQ7

7.1.3 Constructor & Destructor Documentation

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

Constructor. Bridge class constructor

Parameters:

← *device* Device identifier on the [VME](#) crate

← *type* Device type (1718/2718)

Here is the call graph for this function:

7.1.3.2 VME::BridgeVx718::~~BridgeVx718 ()

Destructor. Bridge class destructor

7.1.4 Member Function Documentation

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

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

Bridge's handle value.

Returns:

Handle value

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

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

Set and read the input lines.

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

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

Set and control the output lines.

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

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

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

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

Here is the call graph for this function:

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

Here is the call graph for this function:

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

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

Here is the call graph for this function:

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

7.1.5 Field Documentation

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

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

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

7.2 VME::BridgeVx718Control Class Reference

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

Public Member Functions

- [BridgeVx718Control](#) (uint16_t word)
- virtual [~BridgeVx718Control](#) ()
- bool [GetArbiterType](#) () const
Arbiter type.
- bool [GetRequesterType](#) () const
Requester type.
- bool [GetReleaseType](#) () const
Release type.
- unsigned int [GetBusReqLevel](#) () const
- bool [GetInterruptReq](#) () const
- bool [GetSysRes](#) () const
- bool [GetBusTimeout](#) () const
VME bus timeout.
- bool [GetAddressIncrement](#) () const
Address Increment.

Private Attributes

- uint16_t [fWord](#)

7.2.1 Constructor & Destructor Documentation

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

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

7.2.2 Member Function Documentation

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

Address Increment.

Returns:

true if enabled, else false (FIFO mode)

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

Arbiter type.

Returns:

true if "Round Robin", else fixed priority

7.2.2.3 unsigned int VME::BridgeVx718Control::GetBusReqLevel () const [inline]**7.2.2.4 bool VME::BridgeVx718Control::GetBusTimeout () const [inline]**

VME bus timeout.

Returns:

true if 1400 us, else 50 us

7.2.2.5 bool VME::BridgeVx718Control::GetInterruptReq () const [inline]**7.2.2.6 bool VME::BridgeVx718Control::GetReleaseType () const [inline]**

Release type.

Returns:

true if release on request, else release when done

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

Requester type.

Returns:

true if demand, else fair

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

7.2.3 Field Documentation

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

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

- `include/VME_BridgeVx718.h`

7.3 VME::BridgeVx718Status Class Reference

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

Public Member Functions

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

Private Attributes

- uint16_t [fWord](#)

7.3.1 Constructor & Destructor Documentation

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

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

7.3.2 Member Function Documentation

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

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

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

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

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

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

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

7.3.3 Field Documentation

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

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

- include/VME_BridgeVx718.h

7.4 Client Class Reference

Base client object for the socket.

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

Public Member Functions

- [Client](#) ()
General void client constructor.
- [Client](#) (int port)
Bind a socket client to a given port.
- virtual [~Client](#) ()
- bool [Connect](#) (const [SocketType](#) &type=CLIENT)
Bind this client to the socket.
- void [Disconnect](#) ()
Unbind this client from the socket.
- void [Send](#) (const [Message](#) &m) const
Send a message to the master through the socket.
- void [Send](#) (const [Exception](#) &e) const
- [SocketMessage](#) [SendAndReceive](#) (const [SocketMessage](#) &m, const MessageKey &a) const
- void [Receive](#) ()
Receive a socket message from the master.
- virtual void [ParseMessage](#) (const [SocketMessage](#) &m)
Parse a [SocketMessage](#) received from the master.
- virtual [SocketType](#) [GetType](#) () const
[Socket](#) actor type retrieval method.

Private Member Functions

- void [Announce](#) ()
Announce our entry on the socket to its master.

Private Attributes

- int `fClientId`
- bool `fIsConnected`
- `SocketType` `fType`

7.4.1 Detailed Description

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

Author:

Laurent Forthomme <laurent.forthomme@cern.ch>

Date:

24 Mar 2015

7.4.2 Constructor & Destructor Documentation

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

General void client constructor.

7.4.2.2 `Client::Client (int port)`

Bind a socket client to a given port.

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

Here is the call graph for this function:

7.4.3 Member Function Documentation

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

Announce our entry on the socket to its master.

Here is the call graph for this function:

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

Bind this client to the socket.

Here is the call graph for this function:

7.4.3.3 void Client::Disconnect ()

Unbind this client from the socket.

Here is the call graph for this function:

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

[Socket](#) actor type retrieval method.

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

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

7.4.3.6 void Client::Receive ()

Receive a socket message from the master.

Here is the call graph for this function:

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

Here is the call graph for this function:

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

Send a message to the master through the socket.

Here is the call graph for this function:

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

Here is the call graph for this function:

7.4.4 Field Documentation

7.4.4.1 int Client::fClientId [private]

7.4.4.2 bool Client::fIsConnected [private]

7.4.4.3 SocketType Client::fType [private]

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

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

7.5 Exception Class Reference

A simple exception handler.

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

Public Member Functions

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

Private Attributes

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

7.5.1 Detailed Description

A simple exception handler.

Author:

Laurent Forthomme <laurent.forthomme@cern.ch>

Date:

24 Mar 2015

7.5.2 Constructor & Destructor Documentation

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

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

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

Here is the call graph for this function:

7.5.3 Member Function Documentation

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

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

Here is the call graph for this function:

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

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

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

Here is the call graph for this function:

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

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

Here is the call graph for this function:

7.5.4 Field Documentation

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

7.5.4.2 `int Exception::fErrorNumber [private]`

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

7.5.4.4 `ExceptionType Exception::fType [private]`

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

- include/Exception.h

7.6 file_header_t Struct Reference

Header to the output files.

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

Data Fields

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

7.6.1 Detailed Description

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

Author:

Laurent Forthomme <laurent.forthomme@cern.ch>

Date:

14 Apr 2015

7.6.2 Field Documentation

7.6.2.1 VME::AcquisitionMode file_header_t::acq_mode

7.6.2.2 VME::DetectionMode file_header_t::det_mode

7.6.2.3 uint32_t file_header_t::magic

7.6.2.4 uint8_t file_header_t::num_hptdc

7.6.2.5 uint32_t file_header_t::run_id

7.6.2.6 uint32_t file_header_t::spill_id

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

- include/FileConstants.h

7.7 FileReader Class Reference

Handler for a TDC output file readout.

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

Public Member Functions

- [FileReader](#) (std::string name)
Class constructor.
- [~FileReader](#) ()
- unsigned int [GetNumTDCs](#) () const
- unsigned long [GetNumEvents](#) () const
- bool [GetNextEvent](#) (VME::TDCEvent *)
- bool [GetNextMeasurement](#) (unsigned int channel_id, VME::TDCMeasurement *mc)
Fetch the next full measurement on a given channel.

Private Attributes

- std::ifstream [fFile](#)
- [file_header_t](#) [fHeader](#)
- VME::AcquisitionMode [fReadoutMode](#)
- unsigned long [fNumEvents](#)

7.7.1 Detailed Description

Handler for a TDC output file readout.

Author:

Laurent Forthomme <laurent.forthomme@cern.ch>

Date:

Jun 2015

7.7.2 Constructor & Destructor Documentation

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

Class constructor.

Parameters:

- ← **name** Path to the file to read
- ← **ro** Data readout mode (continuous storage or trigger matching)

7.7.2.2 FileReader::~~FileReader ()

7.7.3 Member Function Documentation

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

Here is the call graph for this function:

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

Fetch the next full measurement on a given channel.

Parameters:

- ← *channel_id* Unique identifier of the channel number to retrieve
- *m* A full measurement with leading, trailing times, ...

Returns:

A boolean stating the success of retrieval operation

Here is the call graph for this function:

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

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

7.7.4 Field Documentation

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

7.7.4.2 file_header_t FileReader::fHeader [private]

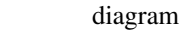

7.7.4.3 unsigned long FileReader::fNumEvents [private]

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

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

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

7.8 VME::FPGAUnitV1495 Class Reference

#include <VME_FPGAUnitV1495.h>  for
VME::FPGAUnitV1495: 

Public Types

- enum [TDCBits](#) { [kReset](#) = 0x1, [kTrigger](#) = 0x2, [kClear](#) = 0x4 }

Public Member Functions

- [FPGAUnitV1495](#) (int32_t bhandle, uint32_t baseaddr)
- [~FPGAUnitV1495](#) ()
- unsigned short [GetCAENFirmwareRevision](#) () const
- unsigned short [GetUserFirmwareRevision](#) () const
- unsigned int [GetHardwareRevision](#) () const
- unsigned short [GetSerialNumber](#) () const
- unsigned short [GetGeoAddress](#) () const
- void [CheckBoardVersion](#) () const
- void [ResetFPGA](#) () const
- void [DumpFWInformation](#) () const
- void [SetTDCBits](#) (unsigned short bits) const
Set a pattern of bits to be sent to all TDCs through the ECL mezzanine.
- void [PulseTDCBits](#) (unsigned short bits, unsigned int time_us=10) const
Send a pulse to TDCs' front panel.
- unsigned short [GetTDCBits](#) () const
Retrieve the current bits sent to TDCs' front panel.
- [FPGAUnitV1495Control](#) [GetControl](#) () const
Retrieve the user-defined control word.
- void [SetControl](#) (const [FPGAUnitV1495Control](#) &control) const
Set the user-defined control word.
- void [SetInternalClockPeriod](#) (uint32_t period) const
Set the internal clock period.
- uint32_t [GetInternalClockPeriod](#) () const
Retrieve the internal clock period.
- void [SetInternalTriggerPeriod](#) (uint32_t period) const
Set the internal trigger period.

- uint32_t [GetInternalTriggerPeriod](#) () const
Retrieve the internal trigger period.
- uint32_t [GetOutputPulser](#) () const
- void [SetOutputPulser](#) (unsigned short id, bool internal_trigger, bool enable=true) const

7.8.1 Detailed Description

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

Author:

Laurent Forthomme <laurent.forthomme@cern.ch>

Date:

25 Jun 2015

7.8.2 Member Enumeration Documentation

7.8.2.1 enum VME::FPGAUnitV1495::TDCBits

Enumerator:

kReset
kTrigger
kClear

7.8.3 Constructor & Destructor Documentation

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

Here is the call graph for this function:

7.8.3.2 VME::FPGAUnitV1495::~~FPGAUnitV1495 () [inline]

7.8.4 Member Function Documentation

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

Here is the call graph for this function:

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

Here is the call graph for this function:

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

Here is the call graph for this function:

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

Retrieve the user-defined control word.

Here is the call graph for this function:

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

Here is the call graph for this function:

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

Here is the call graph for this function:

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

Retrieve the internal clock period.

Returns:

Clock period (in units of 25 ns)

Here is the call graph for this function:

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

Retrieve the internal trigger period.

Returns:

Trigger period (in units of 50 ns)

Here is the call graph for this function:

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

Here is the call graph for this function:

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

Here is the call graph for this function:

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

Retrieve the current bits sent to TDCs' front panel.

Returns:

A 3-bit word PoI

Here is the call graph for this function:

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

Here is the call graph for this function:

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

Send a pulse to TDCs' front panel.

Parameters:

← *bits* The pattern to send (3 bits)

← *time_us* Pulse width (in us)

Here is the call graph for this function:

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

Here is the call graph for this function:

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

Set the user-defined control word.

Here is the call graph for this function:

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

Set the internal clock period.

Parameters:

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

Here is the call graph for this function:

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

Set the internal trigger period.

Parameters:

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

Here is the call graph for this function:

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

Here is the call graph for this function:

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

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

Here is the call graph for this function:

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

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

7.9 VME::FPGAUnitV1495Control Class Reference

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

Public Types

- enum [ClockSource](#) { [InternalClock](#) = 0x0, [ExternalClock](#) = 0x1 }
- enum [TriggerSource](#) { [InternalTrigger](#) = 0x0, [ExternalTrigger](#) = 0x1 }

Public Member Functions

- [FPGAUnitV1495Control](#) (uint32_t word)
- virtual [~FPGAUnitV1495Control](#) ()
- uint32_t [GetWord](#) () const
- [ClockSource](#) [GetClockSource](#) () const
Get the clock source.
- void [SetClockSource](#) (const [ClockSource](#) &cs)
Switch between internal and external clock source.
- [TriggerSource](#) [GetTriggerSource](#) () const
Get the trigger source.
- void [SetTriggerSource](#) (const [TriggerSource](#) &cs)
Switch between internal and external trigger source.

Private Attributes

- uint32_t [fWord](#)

7.9.1 Detailed Description

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

Author:

Laurent Forthomme <laurent.forthomme@cern.ch>

Date:

27 Jun 2015

7.9.2 Member Enumeration Documentation

7.9.2.1 enum VME::FPGAUnitV1495Control::ClockSource

Enumerator:

InternalClock

ExternalClock

7.9.2.2 enum VME::FPGAUnitV1495Control::TriggerSource

Enumerator:

InternalTrigger

ExternalTrigger

7.9.3 Constructor & Destructor Documentation

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

7.9.3.2 virtual VME::FPGAUnitV1495Control::~~FPGAUnitV1495Control () [inline, virtual]

7.9.4 Member Function Documentation

7.9.4.1 ClockSource VME::FPGAUnitV1495Control::GetClockSource () const [inline]

Get the clock source.

7.9.4.2 TriggerSource VME::FPGAUnitV1495Control::GetTriggerSource () const [inline]

Get the trigger source.

7.9.4.3 uint32_t VME::FPGAUnitV1495Control::GetWord () const [inline]

7.9.4.4 void VME::FPGAUnitV1495Control::SetClockSource (const ClockSource & cs) [inline]

Switch between internal and external clock source.

Here is the call graph for this function:

7.9.4.5 void VME::FPGAUnitV1495Control::SetTriggerSource (const TriggerSource & cs) [inline]

Switch between internal and external trigger source.

Here is the call graph for this function:

7.9.5 Field Documentation

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

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

- include/VME_FPGAUnitV1495.h

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

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

Public Member Functions

- [GenericBoard](#) (int32_t bhandle, uint32_t baseaddr)
- virtual [~GenericBoard](#) ()

Protected Member Functions

- void [WriteRegister](#) (const Register ®, const uint16_t &data) const

Write on register.

- void [WriteRegister](#) (const Register ®, const uint32_t &data) const

Write on register.

- void [ReadRegister](#) (const Register ®, uint16_t *data) const

Read on register.

- void [ReadRegister](#) (const Register ®, uint32_t *data) const

Read on register.

Protected Attributes

- int32_t [fHandle](#)
- uint32_t [fBaseAddr](#)

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

7.10.1 Constructor & Destructor Documentation

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

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

7.10.2 Member Function Documentation

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

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

Parameters:

← *addr* register
→ *data* word

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

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

Parameters:

← *addr* register
→ *data* word

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

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

Parameters:

← *addr* register
← *data* word

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

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

Parameters:

← *addr* register

← *data* word

7.10.3 Field Documentation

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

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

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

- include/VME_GenericBoard.h

7.12 HTTPMessage Class Reference

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

#include <HTTPMessage.h> Inheritance diagram for HTTPMessage:
 Collaboration diagram for HTTPMessage:

Public Member Functions

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

Private Attributes

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

7.12.1 Detailed Description

[Message](#) to be transmitted through a WebSocket protocol. Type of message compatible to the transmission through a WebSocket protocol. It enables a direct conversion of standards from any socket message format used elsewhere in this code using the *MessageAction* statement.

Author:

Laurent Forthomme <laurent.forthomme@cern.ch>

Date:

1 Apr 2015

7.12.2 Constructor & Destructor Documentation

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

Here is the call graph for this function:

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

Here is the call graph for this function:

7.12.3 Member Function Documentation

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

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

Reimplemented from [Message](#).

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

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

Placeholder for the MessageKey retrieval method.

Reimplemented from [Message](#).

7.12.4 Field Documentation

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

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

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

- `include/HTTPMessage.h`

7.13 VME::IOModuleV262 Class Reference

`#include <VME_IOModuleV262.h>` [Inheritance diagram](#) [for VME::IOModuleV262](#): [Collaboration diagram for VME::IOModuleV262](#):

Public Member Functions

- [IOModuleV262](#) (int32_t bhandle, uint32_t baseaddr)
- [~IOModuleV262](#) ()
- unsigned short [GetSerialNumber](#) () const
- unsigned short [GetModuleVersion](#) () const
- unsigned short [GetModuleType](#) () const
- unsigned short [GetManufacturerId](#) () const
- unsigned short [GetIdentifier](#) () const

7.13.1 Constructor & Destructor Documentation

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

Here is the call graph for this function:

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

7.13.2 Member Function Documentation

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

Here is the call graph for this function:

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

Here is the call graph for this function:

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

Here is the call graph for this function:

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

Here is the call graph for this function:

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

Here is the call graph for this function:

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

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

7.14 Message Class Reference

Base socket message type.

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

Public Member Functions

- [Message](#) ()
Void message constructor.
- [Message](#) (const char *msg)
Construct a message from a string.
- [Message](#) (std::string msg)
Construct a message from a string.
- virtual [~Message](#) ()
- MessageKey [GetKey](#) () const
Placeholder for the MessageKey retrieval method.
- std::string [GetString](#) () const
Retrieve the string carried by this message as a whole.
- bool [IsFromWeb](#) () const
Extract from any message its potential arrival from a WebSocket protocol.
- void [Dump](#) (std::ostream &os=std::cout) const

Protected Attributes

- std::string [fString](#)

7.14.1 Detailed Description

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

Author:

Laurent Forthomme <laurent.forthomme@cern.ch>

Date:

6 Apr 2015

7.14.2 Constructor & Destructor Documentation

7.14.2.1 Message::Message () [inline]

Void message constructor.

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

Construct a message from a string.

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

Construct a message from a string.

7.14.2.4 virtual Message::~Message () [inline, virtual]

7.14.3 Member Function Documentation

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

Reimplemented in [HTTPMessage](#), and [SocketMessage](#).

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

Placeholder for the MessageKey retrieval method.

Reimplemented in [HTTPMessage](#), and [SocketMessage](#).

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

Retrieve the string carried by this message as a whole.

Reimplemented in [SocketMessage](#).

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

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

7.14.4 Field Documentation

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

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

- include/Message.h

7.15 Messenger Class Reference

Base master object for the socket.

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

Public Member Functions

- [Messenger](#) ()
Build a void master object or socket actor.
- [Messenger](#) (int port)
Build a master object to control the socket.
- [~Messenger](#) ()
- bool [Connect](#) ()
Connect the master to the socket.
- void [Disconnect](#) ()
Remove the master and destroy the socket.
- void [Send](#) (const [Message](#) &m, int sid) const
Send any type of message to any client.
- void [Receive](#) ()
Handle a message reception from a client.
- void [Broadcast](#) (const [Message](#) &m) const
Emit a message to all clients connected through the socket.
- void [StartAcquisition](#) ()
Start the data acquisition.
- void [StopAcquisition](#) ()
- [SocketType GetType](#) () const
Socket actor type retrieval method.

Private Member Functions

- void [AddClient](#) ()
Add a client to listen to.
- void [DisconnectClient](#) (int sid, MessageKey key, bool force=false)
Disconnect a client.

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

Process a message received from the socket.

Private Attributes

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

7.15.1 Detailed Description

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

Author:

Laurent Forthomme <laurent.forthomme@cern.ch>

Date:

23 Mar 2015

7.15.2 Constructor & Destructor Documentation

7.15.2.1 `Messenger::Messenger ()`

Build a void master object or socket actor.

7.15.2.2 `Messenger::Messenger (int port)`

Build a master object to control the socket.

Here is the call graph for this function:

7.15.2.3 `Messenger::~~Messenger ()`

Here is the call graph for this function:

7.15.3 Member Function Documentation

7.15.3.1 void Messenger::AddClient () [private]

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

Here is the call graph for this function:

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

Emit a message to all clients connected through the socket.

Parameters:

← *m* Message to transmit

Here is the call graph for this function:

7.15.3.3 bool Messenger::Connect ()

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

Here is the call graph for this function:

7.15.3.4 void Messenger::Disconnect ()

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

Here is the call graph for this function:

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

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

Parameters:

← *sid* Unique identifier of the client to disconnect

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

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

Here is the call graph for this function:

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

Socket actor type retrieval method.

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

Process a message received from the socket.

Parameters:

← *Unique* identifier of the client sending the message

Here is the call graph for this function:

7.15.3.8 void Messenger::Receive ()

Handle a message reception from a client.

Here is the call graph for this function:

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

Send any type of message to any client.

Parameters:

← *m* Message to transmit

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

Here is the call graph for this function:

7.15.3.10 void Messenger::StartAcquisition ()

Start the data acquisition.

Here is the call graph for this function:

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

Here is the call graph for this function:

7.15.4 Field Documentation

7.15.4.1 `int Messenger::fNumAttempts` `[private]`

7.15.4.2 `pid_t Messenger::fPID` `[private]`

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

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

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

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

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

7.16 Socket Class Reference

Base socket object from which clients/master from a socket inherit.

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

Public Types

- enum [SocketType](#) {
[INVALID](#) = -1, [MASTER](#) = 0, [WEBSOCKET_CLIENT](#), [CLIENT](#),
[DETECTOR](#) }
Type of actor playing a role on the socket.
- typedef std::set< std::pair< int, [SocketType](#) > > [SocketCollection](#)

Public Member Functions

- [Socket](#) ()
- [Socket](#) (int port)
- virtual [~Socket](#) ()
- void [Stop](#) ()
Terminates the socket and all attached communications.
- void [SetPort](#) (int port)
- int [GetPort](#) () const
Retrieve the port used for this socket.
- void [AcceptConnections](#) ([Socket](#) &socket)
Accept connection from a client.
- void [SelectConnections](#) ()
- void [SetSocketId](#) (int sid)
- int [GetSocketId](#) () const
- [SocketType](#) [GetSocketType](#) (int sid) const
- bool [IsWebSocket](#) (int sid) const
- void [DumpConnected](#) () const

Protected Member Functions

- bool [Start](#) ()
Start the socket.
- void [Bind](#) ()
Bind a name to a socket.

- void [PrepareConnection](#) ()
- void [Listen](#) (int maxconn)
Listen to incoming messages.
- void [SendMessage](#) ([Message](#) message, int id=-1) const
Send a message on a socket.
- [Message](#) [FetchMessage](#) (int id=-1) const
Receive a message from a socket.

Protected Attributes

- int [fPort](#)
- char [fBuffer](#) [MAX_WORD_LENGTH]
- [SocketCollection](#) [fSocketsConnected](#)
- fd_set [fMaster](#)
Master file descriptor list.
- fd_set [fReadFds](#)
Temp file descriptor list for select().

Private Member Functions

- void [Create](#) ()
Create an endpoint for communication.
- void [Configure](#) ()
Configure the socket object for communication.

Private Attributes

- int [fSocketId](#)
- struct sockaddr_in [fAddress](#)

7.16.1 Detailed Description

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

Author:

Laurent Forthomme <laurent.forthomme@cern.ch>

Date:

23 Mar 2015

7.16.2 Member Typedef Documentation

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

7.16.3 Member Enumeration Documentation

7.16.3.1 `enum Socket::SocketType`

Type of actor playing a role on the socket.

Enumerator:

INVALID

MASTER

WEBSOCKET_CLIENT

CLIENT

DETECTOR

7.16.4 Constructor & Destructor Documentation

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

7.16.4.2 `Socket::Socket (int port)`

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

7.16.5 Member Function Documentation

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

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

Parameters:

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

Here is the call graph for this function:

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

Bind a name to a socket.

Returns:

Success of the operation

Here is the call graph for this function:

7.16.5.3 void Socket::Configure () [private]

Configure the socket object for communication.

7.16.5.4 void Socket::Create () [private]

Create an endpoint for communication.

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

Receive a message from a socket.

Returns:

Received message as a std::string

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

Retrieve the port used for this socket.

7.16.5.8 int Socket::GetSocketId () const [inline]**7.16.5.9 SocketType Socket::GetSocketType (int sid) const [inline]****7.16.5.10 bool Socket::IsWebSocket (int sid) const [inline]**

Here is the call graph for this function:

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

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

Here is the call graph for this function:

7.16.5.12 void Socket::PrepareConnection () [protected]

Here is the call graph for this function:

7.16.5.13 void Socket::SelectConnections ()

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

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

Send a message on a socket.

Here is the call graph for this function:

7.16.5.15 void Socket::SetPort (int *port*) [inline]**7.16.5.16 void Socket::SetSocketId (int *sid*) [inline]****7.16.5.17 bool Socket::Start () [protected]**

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

Returns:

Success of the operation

Here is the call graph for this function:

7.16.5.18 void Socket::Stop ()

Terminates the socket and all attached communications.

7.16.6 Field Documentation**7.16.6.1 struct sockaddr_in Socket::fAddress [read, private]****7.16.6.2 char Socket::fBuffer[MAX_WORD_LENGTH] [protected]****7.16.6.3 fd_set Socket::fMaster [protected]**

Master file descriptor list.

7.16.6.4 int Socket::fPort [protected]**7.16.6.5 fd_set Socket::fReadFds [protected]**

Temp file descriptor list for select().

7.16.6.6 int Socket::fSocketId [private]

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

7.16.6.7 SocketCollection Socket::fSocketsConnected [protected]

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

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

7.17 SocketMessage Class Reference

Socket-passed message type.

#include <SocketMessage.h> Inheritance diagram for SocketMessage: Collaboration diagram for SocketMessage:

Public Member Functions

- [SocketMessage](#) ()
- [SocketMessage](#) (const [Message](#) &msg)
- [SocketMessage](#) (const char *msg_s)
- [SocketMessage](#) (std::string msg_s)
- [SocketMessage](#) (const MessageKey &key)
Construct a socket message out of a key.
- [SocketMessage](#) (const MessageKey &key, const char *value)
Construct a socket message out of a key and a string-type value.
- [SocketMessage](#) (const MessageKey &key, std::string value)
Construct a socket message out of a key and a string-type value.
- [SocketMessage](#) (const MessageKey &key, const int value)
Construct a socket message out of a key and an integer-type value.
- [SocketMessage](#) (const MessageKey &key, const float value)
Construct a socket message out of a key and a float-type value.
- [SocketMessage](#) (const MessageKey &key, const double value)
Construct a socket message out of a key and a double precision-type value.
- [SocketMessage](#) (MessageMap msg_m)
Construct a socket message out of a map of key/string-type value.
- [~SocketMessage](#) ()
- void [SetKeyValue](#) (const MessageKey &key, const char *value)
String-valued message.
- void [SetKeyValue](#) (const MessageKey &key, int int_value)
Send an integer-valued message.
- void [SetKeyValue](#) (const MessageKey &key, float float_value)
Float-valued message.
- void [SetKeyValue](#) (const MessageKey &key, double double_value)
Double-valued message.

- std::string [GetString](#) () const
Extract the whole key:value message.
- MessageKey [GetKey](#) () const
Extract the message's key.
- std::string [GetValue](#) () const
Extract the message's string value.
- int [GetIntValue](#) () const
Extract the message's integer value.
- VectorValue [GetVectorValue](#) () const
Extract the message's vector of string value.
- void [Dump](#) (std::ostream &os=std::cout) const

Private Member Functions

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

Private Attributes

- MessageMap [fMessage](#)

7.17.1 Detailed Description

Socket-passed message type.

Author:

Laurent Forthomme <laurent.forthomme@cern.ch>

Date:

26 Mar 2015

7.17.2 Constructor & Destructor Documentation

7.17.2.1 SocketMessage::SocketMessage () [inline]

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

Here is the call graph for this function:

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

Here is the call graph for this function:

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

Here is the call graph for this function:

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

Construct a socket message out of a key.

Here is the call graph for this function:

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

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

Here is the call graph for this function:

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

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

Here is the call graph for this function:

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

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

Here is the call graph for this function:

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

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

Here is the call graph for this function:

7.17.2.10 SocketMessage::SocketMessage (const MessageKey & *key*, const double *value*) [inline]

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

Here is the call graph for this function:

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

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

7.17.2.12 SocketMessage::~~SocketMessage () [inline]

7.17.3 Member Function Documentation

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

Reimplemented from [Message](#).

Here is the call graph for this function:

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

Extract the message's integer value.

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

Extract the message's key.

Reimplemented from [Message](#).

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

Extract the whole key:value message.

Reimplemented from [Message](#).

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

Extract the message's string value.

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

Extract the message's vector of string value.

Here is the call graph for this function:

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

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

Double-valued message.

Here is the call graph for this function:

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

Float-valued message.

Here is the call graph for this function:

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

Send an integer-valued message.

Here is the call graph for this function:

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

String-valued message.

Here is the call graph for this function:

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

7.17.4 Field Documentation

7.17.4.1 `MessageMap SocketMessage::fMessage [private]`

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

- include/SocketMessage.h

7.18 VME::TDCErrorFlag Class Reference

Error flags handler.

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

Public Member Functions

- [TDCErrorFlag](#) (uint16_t ef)
- virtual [~TDCErrorFlag](#) ()
- uint16_t [GetWord](#) () const
- void [Dump](#) () const
- bool [HasReadoutFIFOOverflow](#) (unsigned int group_id) const
Check whether hits have been lost from read-out FIFO overflow in a given group.
- bool [HasL1BufferOverflow](#) (unsigned int group_id) const
Check whether hits have been lost from L1 buffer overflow in a given group.
- bool [HasGroupError](#) (unsigned int group_id) const
Check whether hits have been lost due to error in a given group.
- bool [HasReachedEventSizeLimit](#) () const
Hits rejected because of programmed event size limit.
- bool [HasTriggerFIFOOverflow](#) () const
Event lost (trigger FIFO overflow).
- bool [HasInternalChipError](#) () const
Internal fatal chip error has been detected.

Private Attributes

- uint16_t [fWord](#)

Friends

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

7.18.1 Detailed Description

Error flags handler.

Author:

Laurent Forthomme <laurent.forthomme@cern.ch>

Date:

22 Jun 2015

7.18.2 Constructor & Destructor Documentation

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

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

7.18.3 Member Function Documentation

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

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

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

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

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

Internal fatal chip error has been detected.

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

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

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

Hits rejected because of programmed event size limit.

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

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

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

Event lost (trigger FIFO overflow).

7.18.4 Friends And Related Function Documentation

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

7.18.5 Field Documentation

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

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

- `include/VME_TDCEvent.h`

7.19 VME::TDCEvent Class Reference

HPTDC event parser.

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

Public Types

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

Public Member Functions

- [TDCEvent](#) ()
- [TDCEvent](#) (const [TDCEvent](#) &ev)
- [TDCEvent](#) (const uint32_t &word)
- virtual [~TDCEvent](#) ()
- void [Dump](#) () const
- void [SetWord](#) (const uint32_t &word)
- uint32_t [GetWord](#) () const
- [EventType](#) [GetType](#) () const
Type of packet read out from the TDC.
- unsigned int [GetTDCId](#) () const
Programmed identifier of master TDC providing the event.
- uint16_t [GetEventId](#) () const
Event identifier from event counter.
- uint16_t [GetWordCount](#) () const
Total number of words in event (including headers and trailers).
- unsigned int [GetGeo](#) () const
- unsigned int [GetChannelId](#) () const
Channel number for.
- uint32_t [GetEventCount](#) () const
Total number of events.
- uint16_t [GetBunchId](#) () const
Bunch identifier of trigger (or trigger time tag).
- bool [IsTrailing](#) () const

Are we dealing with a trailing or a leading measurement?

- uint32_t [GetETTT](#) () const
Extended trigger time tag.
- uint32_t [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](#)

7.19.1 Detailed Description

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

Author:

Laurent Forthomme <laurent.forthomme@cern.ch>

Date:

4 May 2015

7.19.2 Member Enumeration Documentation

7.19.2.1 enum VME::TDCEvent::EventType

Enumerator:

TDCMeasurement

TDCHeader

TDCTrailer

TDCError

GlobalHeader

GlobalTrailer

ETTT

Filler

7.19.3 Constructor & Destructor Documentation

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

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

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

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

7.19.4 Member Function Documentation

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

Here is the call graph for this function:

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

Bunch identifier of trigger (or trigger time tag).

Here is the call graph for this function:

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

Channel number for.

Here is the call graph for this function:

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

Return error flags if an error condition has been detected.

Here is the call graph for this function:

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

Extended trigger time tag.

Here is the call graph for this function:

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

Total number of events.

Here is the call graph for this function:

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

Event identifier from event counter.

Here is the call graph for this function:

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

Here is the call graph for this function:

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

Leading edge measurement in programmed time resolution.

Parameters:

← *pair* Are we dealing with a pair measurement?

Here is the call graph for this function:

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

Here is the call graph for this function:

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

Programmed identifier of master TDC providing the event.

Here is the call graph for this function:

7.19.4.12 uint32_t VME::TDCEvent::GetTrailingTime () const [inline]

Trailing edge measurement in programmed time resolution.

Here is the call graph for this function:

7.19.4.13 EventType VME::TDCEvent::GetType () const [inline]

Type of packet read out from the TDC.

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

Width of pulse in programmed time resolution.

Here is the call graph for this function:

7.19.4.15 uint32_t VME::TDCEvent::GetWord () const [inline]**7.19.4.16 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:

7.19.4.17 bool VME::TDCEvent::IsTrailing () const [inline]

Are we dealing with a trailing or a leading measurement?

Here is the call graph for this function:

**7.19.4.18 void VME::TDCEvent::SetWord (const uint32_t & word)
[inline]****7.19.5 Field Documentation****7.19.5.1 uint32_t VME::TDCEvent::fWord [private]**

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

- include/VME_TDCEvent.h

7.20 VME::TDCMeasurement Class Reference

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

Public Member Functions

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

Private Attributes

- std::map< [TDCEvent::EventType](#), [TDCEvent](#) > [fMap](#)
- std::vector< std::pair< [TDCEvent](#), [TDCEvent](#) > > [fEvents](#)

7.20.1 Constructor & Destructor Documentation

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

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

Here is the call graph for this function:

7.20.1.3 VME::TDCMeasurement::~~TDCMeasurement () [[inline](#)]

7.20.2 Member Function Documentation

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

Here is the call graph for this function:

- 7.20.2.2 `uint16_t VME::TDCMeasurement::GetBunchId () [inline]`
- 7.20.2.3 `uint16_t VME::TDCMeasurement::GetChannelId (unsigned short event_id = 0) [inline]`
- 7.20.2.4 `uint16_t VME::TDCMeasurement::GetEventId () [inline]`
- 7.20.2.5 `uint32_t VME::TDCMeasurement::GetLeadingTime (unsigned short event_id = 0) [inline]`
- 7.20.2.6 `uint16_t VME::TDCMeasurement::GetTDCId () [inline]`
- 7.20.2.7 `uint16_t VME::TDCMeasurement::GetToT (unsigned short event_id = 0) [inline]`

Here is the call graph for this function:

- 7.20.2.8 `uint32_t VME::TDCMeasurement::GetTrailingTime (unsigned short event_id = 0) [inline]`
- 7.20.2.9 `size_t VME::TDCMeasurement::NumEvents () const [inline]`
- 7.20.2.10 `void VME::TDCMeasurement::SetEventsCollection (const std::vector< TDCEvent > & v) [inline]`

7.20.3 Field Documentation

- 7.20.3.1 `std::vector< std::pair<TDCEvent,TDCEvent> >`
`VME::TDCMeasurement::fEvents [private]`
- 7.20.3.2 `std::map<TDCEvent::EventType,TDCEvent>`
`VME::TDCMeasurement::fMap [private]`

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

- `include/VME_TDCMeasurement.h`

7.21 VME::TDCV1x90 Class Reference

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

Public Types

- enum [DLLMode](#) { [DLL_Direct_LowRes](#) = 0x0, [DLL_PLL_LowRes](#) = 0x1, [DLL_PLL_MedRes](#) = 0x2, [DLL_PLL_HighRes](#) = 0x3 }

Public Member Functions

- [TDCV1x90](#) (int32_t bhandle, uint32_t baseaddr)
- [~TDCV1x90](#) ()
- void [SetVerboseLevel](#) (unsigned short verb=1)
- void [SetTestMode](#) (bool en=true) const
- bool [GetTestMode](#) () const
- uint32_t [GetModel](#) () const
- uint32_t [GetOUI](#) () const
- uint32_t [GetSerialNumber](#) () const
- 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 occured triggers.
- uint16_t [GetEventStored](#) () const
Number of events currently stored in the output buffer.

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

Private Member Functions

- bool [WaitMicro](#) ([micro_handshake](#) mode) const
- void [ReadAcquisitionMode](#) ()
- void [ReadDetectionMode](#) ()

Private Attributes

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

7.21.1 Detailed Description

Author:

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

Date:

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

7.21.2 Member Enumeration Documentation

7.21.2.1 enum VME::TDCV1x90::DLLMode

Enumerator:

DLL_Direct_LowRes
DLL_PLL_LowRes
DLL_PLL_MedRes
DLL_PLL_HighRes

7.21.3 Constructor & Destructor Documentation

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

Here is the call graph for this function:

7.21.3.2 VME::TDCV1x90::~~TDCV1x90 ()

7.21.4 Member Function Documentation

7.21.4.1 void VME::TDCV1x90::abort ()

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

Here is the call graph for this function:

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

Here is the call graph for this function:

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

Here is the call graph for this function:

7.21.4.5 TDCEventCollection VME::TDCV1x90::FetchEvents ()

Here is the call graph for this function:

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

Here is the call graph for this function:

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

Here is the call graph for this function:

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

Here is the call graph for this function:

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

Here is the call graph for this function:

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

Here is the call graph for this function:

7.21.4.11 DLLMode VME::TDCV1x90::GetDLLClock () const**7.21.4.12 bool VME::TDCV1x90::GetErrorMarks () const [inline]****7.21.4.13 bool VME::TDCV1x90::GetETTT () const [inline]**

Here is the call graph for this function:

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

Number of 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:

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

Number of events currently stored in the output buffer.

Here is the call graph for this function:

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

Here is the call graph for this function:

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

Here is the call graph for this function:

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

Here is the call graph for this function:

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

Here is the call graph for this function:

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

Here is the call graph for this function:

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

Here is the call graph for this function:

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

Here is the call graph for this function:

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

Here is the call graph for this function:

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

Here is the call graph for this function:

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

Here is the call graph for this function:

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

Here is the call graph for this function:

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

Here is the call graph for this function:

7.21.4.29 int16_t VME::TDCV1x90::GetWindowOffset () const**7.21.4.30 uint16_t VME::TDCV1x90::GetWindowWidth () const [inline]****7.21.4.31 bool VME::TDCV1x90::HardwareReset () const****7.21.4.32 void VME::TDCV1x90::ReadAcquisitionMode () [private]**

Here is the call graph for this function:

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

Here is the call graph for this function:

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

Here is the call graph for this function:

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

Here is the call graph for this function:

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

Here is the call graph for this function:

7.21.4.37 void VME::TDCV1x90::SetContinuousStorage ()

Here is the call graph for this function:

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

Here is the call graph for this function:

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

Here is the call graph for this function:

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

Here is the call graph for this function:

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

Here is the call graph for this function:

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

Here is the call graph for this function:

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

Here is the call graph for this function:

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

Here is the call graph for this function:

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

Here is the call graph for this function:

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

Here is the call graph for this function:

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

Here is the call graph for this function:

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

Here is the call graph for this function:

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

Here is the call graph for this function:

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

Here is the call graph for this function:

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

Here is the call graph for this function:

7.21.4.52 `void VME::TDCV1x90::SetTriggerMatching ()`

Here is the call graph for this function:

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

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

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

Parameters:

← *Window* offset, in units of clock cycles

Here is the call graph for this function:

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

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

Parameters:

← *Window* width, in units of clock cycles

Here is the call graph for this function:

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

Here is the call graph for this function:

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

Here is the call graph for this function:

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

Here is the call graph for this function:

7.21.5 Field Documentation**7.21.5.1 AcquisitionMode VME::TDCV1x90::fAcquisitionMode [private]****7.21.5.2 uint32_t* VME::TDCV1x90::fBuffer [private]****7.21.5.3 DetectionMode VME::TDCV1x90::fDetectionMode [private]****7.21.5.4 bool VME::TDCV1x90::fErrorMarks [private]****7.21.5.5 unsigned short VME::TDCV1x90::fVerb [private]****7.21.5.6 uint16_t VME::TDCV1x90::fWindowWidth [private]****7.21.5.7 bool VME::TDCV1x90::gEnd [private]****7.21.5.8 uint32_t VME::TDCV1x90::nchannels [private]****7.21.5.9 std::string VME::TDCV1x90::pair_lead_res[8] [private]****7.21.5.10 std::string VME::TDCV1x90::pair_width_res[16] [private]**

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

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

7.22 VME::TDCV1x90Control Class Reference

TDC control register.

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

Public Member Functions

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

Private Attributes

- uint16_t [fWord](#)

7.22.1 Detailed Description

TDC control register.

Author:

Laurent Forthomme <laurent.forthomme@cern.ch>

Date:

Jun 2015

7.22.2 Constructor & Destructor Documentation

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

7.22.2.2 virtual VME::TDCV1x90Control::~~TDCV1x90Control () [inline, virtual]

7.22.3 Member Function Documentation

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

Here is the call graph for this function:

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

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

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

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

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

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

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

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

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

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

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

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

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

Here is the call graph for this function:

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

Here is the call graph for this function:

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

Here is the call graph for this function:

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

Here is the call graph for this function:

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

Here is the call graph for this function:

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

Here is the call graph for this function:

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

Here is the call graph for this function:

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

Here is the call graph for this function:

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

Here is the call graph for this function:

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

Here is the call graph for this function:

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

Here is the call graph for this function:

7.22.4 Field Documentation

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

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

- include/VME_TDCV1x90.h

7.23 VME::TDCV1x90Status Class Reference

TDC status register.

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

Public Types

- enum [TDCResolution](#) { [R_800ps](#) = 0x0, [R_200ps](#) = 0x1, [R_100ps](#) = 0x2, [R_25ps](#) = 0x3 }

Public Member Functions

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

Private Attributes

- uint16_t [fWord](#)

7.23.1 Detailed Description

TDC status register.

Author:

Laurent Forthomme <laurent.forthomme@cern.ch>

Date:

Jun 2015

7.23.2 Member Enumeration Documentation

7.23.2.1 enum VME::TDCV1x90Status::TDCResolution

Enumerator:

R_800ps

R_200ps

R_100ps

R_25ps

7.23.3 Constructor & Destructor Documentation

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

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

7.23.4 Member Function Documentation

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

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

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

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

Here is the call graph for this function:

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

Here is the call graph for this function:

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

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

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

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

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

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

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

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

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

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

7.23.5 Field Documentation

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

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

- `include/VME_TDCV1x90.h`

7.24 VME::trailead_t Struct Reference

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

Data Fields

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

7.24.1 Field Documentation

7.24.1.1 uint32_t VME::trailead_t::ettt

7.24.1.2 uint32_t VME::trailead_t::event_count

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

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

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

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

- include/VME_TDCV1x90.h

7.25 VMEReader Class Reference

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

Public Member Functions

- [VMEReader](#) (const char *device, [VME::BridgeType](#) type, bool on_socket=true)
- virtual [~VMEReader](#) ()
- void [AddTDC](#) (uint32_t address)
Add a TDC to handle.
- [VME::TDCV1x90](#) * [GetTDC](#) (uint32_t address)
Get a TDC on the [VME](#) bus Return a pointer to the TDC object, given its physical address on the [VME](#) bus.
- void [AddIOModule](#) (uint32_t address)
- [VME::IOModuleV262](#) * [GetIOModule](#) ()
- void [AddFPGAUnit](#) (uint32_t address)
- [VME::FPGAUnitV1495](#) * [GetFPGAUnit](#) ()
- unsigned int [GetRunNumber](#) ()
Ask the socket master a run number.
- void [StartPulser](#) (double period, double width, unsigned int num_pulses=0)
- void [StopPulser](#) ()
- void [SendPulse](#) (unsigned short output=0) const
- void [SendClear](#) () const
- void [Abort](#) ()
Abort data collection for all modules on the bus handled by the bridge.

Private Types

- typedef std::map< uint32_t, [VME::TDCV1x90](#) * > [TDCCollection](#)
Mapper from physical [VME](#) addresses to pointers to TDC objects.

Private Attributes

- [VME::BridgeVx718](#) * [fBridge](#)
The [VME](#) bridge object to handle.
- [TDCCollection](#) [fTDCCollection](#)
A set of pointers to TDC objects indexed by their physical [VME](#) address.

- [VME::IOModuleV262 * fSG](#)
Pointer to the [VME](#) input/output module object.
- [VME::FPGAUnitV1495 * fFPGA](#)
Pointer to the [VME](#) general purpose FPGA unit object.
- bool [fOnSocket](#)
Are we dealing with socket message passing?
- bool [fIsPulserStarted](#)

7.25.1 Detailed Description

[VME](#) reader object to fetch events on a HPTDC board

Author:

Laurent Forthomme <laurent.forthomme@cern.ch>

Date:

4 May 2015

7.25.2 Member Typedef Documentation

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

Mapper from physical [VME](#) addresses to pointers to TDC objects.

7.25.3 Constructor & Destructor Documentation

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

Parameters:

- ← *device* Path to the device (/dev/xxx)
- ← *type* Bridge model
- ← *on_socket* Are we trying to connect through the socket?

Here is the call graph for this function:

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

Here is the call graph for this function:

7.25.4 Member Function Documentation

7.25.4.1 void VMEReader::Abort ()

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

Here is the call graph for this function:

7.25.4.2 void VMEReader::AddFPGAUnit (uint32_t address)

Here is the call graph for this function:

7.25.4.3 void VMEReader::AddIOModule (uint32_t address)

Here is the call graph for this function:

7.25.4.4 void VMEReader::AddTDC (uint32_t address)

Add a TDC to handle.

Parameters:

← *address* 32-bit address of the TDC module on the [VME](#) bus Create a new TDC handler for the [VME](#) bus

Here is the call graph for this function:

7.25.4.5 VME::FPGAUnitV1495* VMEReader::GetFPGAUnit () [inline]

7.25.4.6 VME::IOModuleV262* VMEReader::GetIOModule () [inline]

7.25.4.7 unsigned int VMEReader::GetRunNumber ()

Ask the socket master a run number.

Here is the call graph for this function:

7.25.4.8 VME::TDCV1x90* VMEReader::GetTDC (uint32_t address) [inline]

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

7.25.4.9 void VMEReader::SendClear () const [inline]

Here is the call graph for this function:

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

Here is the call graph for this function:

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

Here is the call graph for this function:

7.25.4.12 void VMEReader::StopPulser () [inline]

Here is the call graph for this function:

7.25.5 Field Documentation

7.25.5.1 VME::BridgeVx718* VMEReader::fBridge [private]

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

7.25.5.2 VME::FPGAUnitV1495* VMEReader::fFPGA [private]

Pointer to the [VME](#) general purpose FPGA unit object.

7.25.5.3 bool VMEReader::fIsPulserStarted [private]

7.25.5.4 bool VMEReader::fOnSocket [private]

Are we dealing with socket message passing?

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

Pointer to the [VME](#) input/output module object.

7.25.5.6 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, [23](#)
- ~BridgeVx718Control
 - VME::BridgeVx718Control, [25](#)
- ~BridgeVx718Status
 - VME::BridgeVx718Status, [29](#)
- ~Client
 - Client, [31](#)
- ~Exception
 - Exception, [35](#)
- ~FPGAUnitV1495
 - VME::FPGAUnitV1495, [41](#)
- ~FPGAUnitV1495Control
 - VME::FPGAUnitV1495Control, [46](#)
- ~FileReader
 - FileReader, [38](#)
- ~GenericBoard
 - VME::GenericBoard, [49](#)
- ~IOModuleV262
 - VME::IOModuleV262, [54](#)
- ~Message
 - Message, [57](#)
- ~Messenger
 - Messenger, [60](#)
- ~Socket
 - Socket, [66](#)
- ~SocketMessage
 - SocketMessage, [73](#)
- ~TDCErrorFlag
 - VME::TDCErrorFlag, [76](#)
- ~TDCEvent
 - VME::TDCEvent, [80](#)
- ~TDCMeasurement
 - VME::TDCMeasurement, [83](#)
- ~TDCV1x90
 - VME::TDCV1x90, [87](#)
- ~TDCV1x90Control
 - VME::TDCV1x90Control, [95](#)
- ~TDCV1x90Status
 - VME::TDCV1x90Status, [99](#)
- ~VMEReader
 - VMEReader, [103](#)
- Abort
 - VMEReader, [104](#)
- abort
 - VME::TDCV1x90, [87](#)
- AcceptConnections
 - Socket, [66](#)
- acq_mode
 - file_header_t, [37](#)
- AcquisitionMode
 - VME, [13](#)
- AddClient
 - Messenger, [61](#)
- AddFPGAUnit
 - VMEReader, [104](#)
- AddIOModule
 - VMEReader, [104](#)
- AddTDC
 - VMEReader, [104](#)
- AlmostFull
 - VME::TDCV1x90Status, [99](#)
- Announce
 - Client, [31](#)
- AUTOLOAD_DEF_CONFI
 - VME::TDCV1x90Opcodes, [20](#)
- AUTOLOAD_USER_CONF
 - VME::TDCV1x90Opcodes, [20](#)
- Bind
 - Socket, [66](#)
- BridgeType
 - VME, [13](#)
- BridgeVx718
 - VME::BridgeVx718, [23](#)
- BridgeVx718Control
 - VME::BridgeVx718Control, [25](#)
- BridgeVx718Status
 - VME::BridgeVx718Status, [29](#)
- Broadcast
 - Messenger, [61](#)

- BusError
 - VME::TDCV1x90Status, [99](#)
- CAEN_V1718
 - VME, [13](#)
- CAEN_V2718
 - VME, [13](#)
- CheckBoardVersion
 - VME::FPGAUnitV1495, [41](#)
- CheckConfiguration
 - VME::BridgeVx718, [23](#)
 - VME::TDCV1x90, [87](#)
- CLEAR_KEEP_TOKEN
 - VME::TDCV1x90Opcodes, [20](#)
- CLIENT
 - Socket, [66](#)
- Client, [30](#)
 - ~Client, [31](#)
 - Announce, [31](#)
 - Client, [31](#)
 - Connect, [31](#)
 - Disconnect, [31](#)
 - fClientId, [32](#)
 - flsConnected, [32](#)
 - fType, [32](#)
 - GetType, [32](#)
 - ParseMessage, [32](#)
 - Receive, [32](#)
 - Send, [32](#)
 - SendAndReceive, [32](#)
- ClockSource
 - VME::FPGAUnitV1495Control, [46](#)
- coarse
 - VME::GlobalOffset, [51](#)
- Configure
 - Socket, [67](#)
- Connect
 - Client, [31](#)
 - Messenger, [61](#)
- CONT_STORAGE
 - VME, [13](#)
- CONT_STOR
 - VME::TDCV1x90Opcodes, [20](#)
- Create
 - Socket, [67](#)
- DataReady
 - VME::TDCV1x90Status, [99](#)
- Decode
 - HTTPMessage, [53](#)
- DEFAULT_SETUP_REG
 - VME::TDCV1x90Opcodes, [20](#)
- Description
 - Exception, [35](#)
- det_mode
 - file_header_t, [37](#)
- DetectionMode
 - VME, [13](#)
- DETECTOR
 - Socket, [66](#)
- DIS_ALL_CHANNEL
 - VME::TDCV1x90Opcodes, [20](#)
- DIS_CHANNEL
 - VME::TDCV1x90Opcodes, [20](#)
- DIS_ERROR_BYPASS
 - VME::TDCV1x90Opcodes, [20](#)
- DIS_ERROR_MARK
 - VME::TDCV1x90Opcodes, [20](#)
- DIS_HEAD_TRAILER
 - VME::TDCV1x90Opcodes, [20](#)
- DIS_SUB_TRG
 - VME::TDCV1x90Opcodes, [20](#)
- DISABLE_TEST_MODE
 - VME::TDCV1x90Opcodes, [20](#)
- DisableChannel
 - VME::TDCV1x90, [87](#)
- Disconnect
 - Client, [31](#)
 - Messenger, [61](#)
- DisconnectClient
 - Messenger, [61](#)
- DLL_Direct_LowRes
 - VME::TDCV1x90, [87](#)
- DLL_PLL_HighRes
 - VME::TDCV1x90, [87](#)
- DLL_PLL_LowRes
 - VME::TDCV1x90, [87](#)
- DLL_PLL_MedRes
 - VME::TDCV1x90, [87](#)
- DLLMode
 - VME::TDCV1x90, [87](#)
- Dump
 - Exception, [35](#)
 - HTTPMessage, [53](#)
 - Message, [57](#)
 - SocketMessage, [73](#)
 - VME::BridgeVx718Status, [29](#)
 - VME::TDCErrorFlag, [76](#)
 - VME::TDCEvent, [80](#)
 - VME::TDCMeasurement, [83](#)

- VME::TDCV1x90Control, 95
- VME::TDCV1x90Status, 99
- DumpConnected
 - Socket, 67
- DumpFWInformation
 - VME::FPGAUnitV1495, 41
- EN_ALL_CHANNEL
 - VME::TDCV1x90Opcodes, 20
- EN_CHANNEL
 - VME::TDCV1x90Opcodes, 20
- EN_ERROR_BYPASS
 - VME::TDCV1x90Opcodes, 20
- EN_ERROR_MARK
 - VME::TDCV1x90Opcodes, 20
- EN_HEAD_TRAILER
 - VME::TDCV1x90Opcodes, 20
- EN_SUB_TRG
 - VME::TDCV1x90Opcodes, 20
- ENABLE_TEST_MODE
 - VME::TDCV1x90Opcodes, 20
- EnableChannel
 - VME::TDCV1x90, 87
- Encode
 - HTTPMessage, 53
- Error
 - VME::TDCV1x90Status, 99
- ErrorNumber
 - Exception, 35
- ETTT
 - VME::TDCEvent, 80
- ett
 - VME::trailead_t, 101
- event_count
 - VME::trailead_t, 101
- EventType
 - VME::TDCEvent, 79
- Exception, 34
 - ~Exception, 35
 - Description, 35
 - Dump, 35
 - ErrorNumber, 35
 - Exception, 35
 - fDescription, 35
 - fErrorNumber, 35
 - fFrom, 35
 - From, 35
 - fType, 35
 - OneLine, 35
 - Type, 35
- TypeString, 35
- ExternalClock
 - VME::FPGAUnitV1495Control, 46
- ExternalTrigger
 - VME::FPGAUnitV1495Control, 46
- EXTRA_SEARCH_WIN_WIDTH
 - VME, 16
- fAcquisitionMode
 - VME::TDCV1x90, 93
- fAddress
 - Socket, 68
- fBaseAddr
 - VME::GenericBoard, 50
- fBridge
 - VMEReader, 105
- fBuffer
 - Socket, 68
 - VME::TDCV1x90, 93
- fClientId
 - Client, 32
- fDescription
 - Exception, 35
- fDetectionMode
 - VME::TDCV1x90, 93
- fErrorMarks
 - VME::TDCV1x90, 93
- fErrorNumber
 - Exception, 35
- FetchEvents
 - VME::TDCV1x90, 87
- FetchMessage
 - Socket, 67
- fEvents
 - VME::TDCMeasurement, 84
- fFile
 - FileReader, 39
- fFPGA
 - VMEReader, 105
- fFrom
 - Exception, 35
- fHandle
 - VME::GenericBoard, 50
- fHasIRQ
 - VME::BridgeVx718, 24
- fHeader
 - FileReader, 39
- file_header_t, 37
 - acq_mode, 37
 - det_mode, 37

- magic, 37
- num_hptdc, 37
- run_id, 37
- spill_id, 37
- FileReader, 38
 - ~FileReader, 38
 - fFile, 39
 - fHeader, 39
 - FileReader, 38
 - fNumEvents, 39
 - fReadoutMode, 39
 - GetNextEvent, 39
 - GetNextMeasurement, 39
 - GetNumEvents, 39
 - GetNumTDCs, 39
- Filler
 - VME::TDCEvent, 80
- fine
 - VME::GlobalOffset, 51
- fIsConnected
 - Client, 32
- fIsPulserStarted
 - VMEReader, 105
- fMap
 - VME::TDCMeasurement, 84
- fMaster
 - Socket, 68
- fMessage
 - SocketMessage, 74
- fNumAttempts
 - Messenger, 63
- fNumEvents
 - FileReader, 39
- fOnSocket
 - VMEReader, 105
- fOriginalString
 - HTTPMessage, 53
- FPGAUnitV1495
 - VME::FPGAUnitV1495, 41
- FPGAUnitV1495Control
 - VME::FPGAUnitV1495Control, 46
- FPGAUnitV1495Register
 - VME, 14
- fPID
 - Messenger, 63
- fPort
 - Socket, 68
- fReadFds
 - Socket, 68
- fReadoutMode
 - FileReader, 39
- From
 - Exception, 35
- fSG
 - VMEReader, 105
- fSocketId
 - Socket, 68
- fSocketsConnected
 - Socket, 69
- fStderrPipe
 - Messenger, 63
- fStdoutPipe
 - Messenger, 63
- fString
 - Message, 57
- fTDCCollection
 - VMEReader, 105
- fType
 - Client, 32
 - Exception, 35
- Full
 - VME::TDCV1x90Status, 100
- fVerb
 - VME::TDCV1x90, 93
- fWindowWidth
 - VME::TDCV1x90, 93
- fWord
 - VME::BridgeVx718Control, 27
 - VME::BridgeVx718Status, 29
 - VME::FPGAUnitV1495Control, 47
 - VME::TDCErrorFlag, 77
 - VME::TDCEvent, 82
 - VME::TDCV1x90Control, 97
 - VME::TDCV1x90Status, 100
- fWS
 - HTTPMessage, 53
 - Messenger, 63
- gEnd
 - VME::TDCV1x90, 93
- GenericBoard
 - VME::GenericBoard, 49
- GetAcquisitionMode
 - VME::TDCV1x90, 88
- GetAddressIncrement
 - VME::BridgeVx718Control, 25
- GetAlign64
 - VME::TDCV1x90Control, 95
- GetArbiterType
 - VME::BridgeVx718Control, 26

- GetBERR
 - VME::BridgeVx718Status, [29](#)
- GetBLTEventNumberRegister
 - VME::TDCV1x90, [88](#)
- GetBunchId
 - VME::TDCEvent, [80](#)
 - VME::TDCMeasurement, [83](#)
- GetBusError
 - VME::TDCV1x90Control, [95](#)
- GetBusReqLevel
 - VME::BridgeVx718Control, [26](#)
- GetBusTimeout
 - VME::BridgeVx718Control, [26](#)
- GetCAENFirmwareRevision
 - VME::FPGAUnitV1495, [41](#)
- GetChannelDeadTime
 - VME::TDCV1x90, [88](#)
- GetChannelId
 - VME::TDCEvent, [80](#)
 - VME::TDCMeasurement, [84](#)
- GetClockSource
 - VME::FPGAUnitV1495Control, [46](#)
- GetCompensation
 - VME::TDCV1x90Control, [95](#)
- GetControl
 - VME::FPGAUnitV1495, [42](#)
 - VME::TDCV1x90, [88](#)
- GetDetectionMode
 - VME::TDCV1x90, [88](#)
- GetDipSwitch
 - VME::BridgeVx718Status, [29](#)
- GetDLLClock
 - VME::TDCV1x90, [88](#)
- GetDTACK
 - VME::BridgeVx718Status, [29](#)
- GetEmptyEvent
 - VME::TDCV1x90Control, [95](#)
- GetErrorFlags
 - VME::TDCEvent, [80](#)
- GetErrorMarks
 - VME::TDCV1x90, [88](#)
- GetETTT
 - VME::TDCEvent, [80](#)
 - VME::TDCV1x90, [88](#)
 - VME::TDCV1x90Control, [95](#)
- GetEventCount
 - VME::TDCEvent, [80](#)
- GetEventCounter
 - VME::TDCV1x90, [88](#)
- GetEventFIFO
 - VME::TDCV1x90Control, [95](#)
- GetEventId
 - VME::TDCEvent, [81](#)
 - VME::TDCMeasurement, [84](#)
- GetEventStored
 - VME::TDCV1x90, [88](#)
- GetFIFOSize
 - VME::TDCV1x90, [89](#)
- GetFirmwareRevision
 - VME::TDCV1x90, [89](#)
- GetFPGAUnit
 - VMEReader, [104](#)
- GetGeo
 - VME::TDCEvent, [81](#)
- GetGeoAddress
 - VME::FPGAUnitV1495, [42](#)
- GetGlobalOffset
 - VME::TDCV1x90, [89](#)
- GetHandle
 - VME::BridgeVx718, [23](#)
- GetHardwareRevision
 - VME::FPGAUnitV1495, [42](#)
- GetIdentifier
 - VME::IOModuleV262, [54](#)
- GetInternalClockPeriod
 - VME::FPGAUnitV1495, [42](#)
- GetInternalTriggerPeriod
 - VME::FPGAUnitV1495, [42](#)
- GetInterruptReq
 - VME::BridgeVx718Control, [26](#)
- GetIntValue
 - SocketMessage, [73](#)
- GetIOModule
 - VMEReader, [104](#)
- GetIRQStatus
 - VME::BridgeVx718, [23](#)
- GetKey
 - HTTPMessage, [53](#)
 - Message, [57](#)
 - SocketMessage, [73](#)
- GetLeadingTime
 - VME::TDCEvent, [81](#)
 - VME::TDCMeasurement, [84](#)
- GetManufacturerId
 - VME::IOModuleV262, [54](#)
- GetMEBAccess
 - VME::TDCV1x90Control, [95](#)
- GetModel
 - VME::TDCV1x90, [89](#)
- GetModuleType

- VME::IOModuleV262, [54](#)
- GetModuleVersion
 - VME::IOModuleV262, [54](#)
- GetNextEvent
 - FileReader, [39](#)
- GetNextMeasurement
 - FileReader, [39](#)
- GetNumEvents
 - FileReader, [39](#)
- GetNumTDCs
 - FileReader, [39](#)
- GetOUI
 - VME::TDCV1x90, [89](#)
- GetOutputPulser
 - VME::FPGAUnitV1495, [42](#)
- GetPol
 - VME::TDCV1x90, [89](#)
- GetPort
 - Socket, [67](#)
- GetRCAdjust
 - VME::TDCV1x90, [89](#)
- GetReleaseType
 - VME::BridgeVx718Control, [26](#)
- GetRequesterType
 - VME::BridgeVx718Control, [26](#)
- GetResolution
 - VME::TDCV1x90, [89](#)
- GetRunNumber
 - VMEReader, [104](#)
- GetSerialNumber
 - VME::FPGAUnitV1495, [42](#)
 - VME::IOModuleV262, [54](#)
 - VME::TDCV1x90, [89](#)
- GetSocketId
 - Socket, [67](#)
- GetSocketType
 - Socket, [67](#)
- GetSRAMCompensation
 - VME::TDCV1x90Control, [95](#)
- GetStatus
 - VME::TDCEvent, [81](#)
 - VME::TDCV1x90, [89](#)
- GetString
 - Message, [57](#)
 - SocketMessage, [73](#)
- GetSWTermination
 - VME::TDCV1x90Control, [95](#)
- GetSysRes
 - VME::BridgeVx718Control, [26](#)
- GetSystemControl
 - VME::BridgeVx718Status, [29](#)
- GetSystemReset
 - VME::BridgeVx718Status, [29](#)
- GetTDC
 - VMEReader, [104](#)
- GetTDCBits
 - VME::FPGAUnitV1495, [42](#)
- GetTDCEncapsulation
 - VME::TDCV1x90, [89](#)
- GetTDCId
 - VME::TDCEvent, [81](#)
 - VME::TDCMeasurement, [84](#)
- GetTermination
 - VME::TDCV1x90Control, [95](#)
- GetTestFIFO
 - VME::TDCV1x90Control, [95](#)
- GetTestMode
 - VME::TDCV1x90, [90](#)
- GetToT
 - VME::TDCMeasurement, [84](#)
- GetTrailingTime
 - VME::TDCEvent, [81](#)
 - VME::TDCMeasurement, [84](#)
- GetTriggerConfiguration
 - VME::TDCV1x90, [90](#)
- GetTriggerSource
 - VME::FPGAUnitV1495Control, [46](#)
- GetType
 - Client, [32](#)
 - Messenger, [61](#)
 - VME::TDCEvent, [81](#)
- GetUSBType
 - VME::BridgeVx718Status, [29](#)
- GetUserFirmwareRevision
 - VME::FPGAUnitV1495, [43](#)
- GetValue
 - SocketMessage, [73](#)
 - VME::TDCV1x90Control, [95](#)
 - VME::TDCV1x90Status, [100](#)
- GetVectorValue
 - SocketMessage, [73](#)
- GetWidth
 - VME::TDCEvent, [81](#)
- GetWindowOffset
 - VME::TDCV1x90, [90](#)
- GetWindowWidth
 - VME::TDCV1x90, [90](#)
- GetWord
 - VME::FPGAUnitV1495Control, [46](#)
 - VME::TDCErrorFlag, [76](#)

- VME::TDCEvent, [82](#)
- GetWordCount
 - VME::TDCEvent, [82](#)
- GlobalHeader
 - VME::TDCEvent, [79](#)
- GlobalTrailer
 - VME::TDCEvent, [79](#)
- HardwareReset
 - VME::TDCV1x90, [90](#)
- HasGroupError
 - VME::TDCErrorFlag, [76](#)
- HasInternalChipError
 - VME::TDCErrorFlag, [76](#)
- HasL1BufferOverflow
 - VME::TDCErrorFlag, [76](#)
- HasReachedEventSizeLimit
 - VME::TDCErrorFlag, [76](#)
- HasReadoutFIFOOverflow
 - VME::TDCErrorFlag, [76](#)
- HasTriggerFIFOOverflow
 - VME::TDCErrorFlag, [76](#)
- HeadersEnabled
 - VME::TDCV1x90Status, [100](#)
- HTTPMessage, [52](#)
 - Decode, [53](#)
 - Dump, [53](#)
 - Encode, [53](#)
 - fOriginalString, [53](#)
 - fWS, [53](#)
 - GetKey, [53](#)
 - HTTPMessage, [52](#)
- InputConf
 - VME::BridgeVx718, [23](#)
- InputRead
 - VME::BridgeVx718, [23](#)
- InternalClock
 - VME::FPGAUnitV1495Control, [46](#)
- InternalTrigger
 - VME::FPGAUnitV1495Control, [46](#)
- INVALID
 - Socket, [66](#)
- IOModuleV262
 - VME::IOModuleV262, [54](#)
- IOModuleV262Register
 - VME, [14](#)
- IRQ1
 - VME::BridgeVx718, [22](#)
- IRQ2
 - VME::BridgeVx718, [22](#)
- IRQ3
 - VME::BridgeVx718, [22](#)
- IRQ4
 - VME::BridgeVx718, [22](#)
- IRQ5
 - VME::BridgeVx718, [22](#)
- IRQ6
 - VME::BridgeVx718, [22](#)
- IRQ7
 - VME::BridgeVx718, [22](#)
- IRQId
 - VME::BridgeVx718, [22](#)
- IsFromWeb
 - Message, [57](#)
- IsTrailing
 - VME::TDCEvent, [82](#)
- IsWebSocket
 - Socket, [67](#)
- kBLTEventNumber
 - VME, [15](#)
- kBoardInfo0
 - VME, [15](#)
- kBoardInfo1
 - VME, [15](#)
- kClear
 - VME::FPGAUnitV1495, [41](#)
- kControl
 - VME, [15](#)
- kECLLevelWrite
 - VME, [15](#)
- kEventCounter
 - VME, [15](#)
- kEventFIFO
 - VME, [15](#)
- kEventFIFOStatusRegister
 - VME, [15](#)
- kEventFIFOStoredRegister
 - VME, [15](#)
- kEventStored
 - VME, [15](#)
- kFirmwareRev
 - VME, [15](#)
- kGeoAddress
 - VME, [15](#)
- kIdentifier
 - VME, [15](#)
- kInterruptLevel
 - VME, [15](#)

- kInterruptVector
 - VME, [15](#)
- kMCSTBase
 - VME, [15](#)
- kMCSTControl
 - VME, [15](#)
- kMicro
 - VME, [15](#)
- kMicroHandshake
 - VME, [15](#)
- kModuleReset
 - VME, [15](#)
- kNIMLevelWrite
 - VME, [15](#)
- kNIMPulseRead
 - VME, [15](#)
- kNIMPulseWrite
 - VME, [15](#)
- kOutputBuffer
 - VME, [15](#)
- kReset
 - VME::FPGAUnitV1495, [41](#)
- kROMBoard0
 - VME, [16](#)
- kROMBoard1
 - VME, [16](#)
- kROMBoard2
 - VME, [16](#)
- kROMOui0
 - VME, [16](#)
- kROMOui1
 - VME, [16](#)
- kROMOui2
 - VME, [16](#)
- kROMRevis0
 - VME, [16](#)
- kROMRevis1
 - VME, [16](#)
- kROMRevis2
 - VME, [16](#)
- kROMRevis3
 - VME, [16](#)
- kROMSerNum0
 - VME, [16](#)
- kROMSerNum1
 - VME, [16](#)
- kSoftwareClear
 - VME, [15](#)
- kStatus
 - VME, [15](#)
- kTrigger
 - VME::FPGAUnitV1495, [41](#)
- kV1495Board0
 - VME, [14](#)
- kV1495Board1
 - VME, [14](#)
- kV1495Board2
 - VME, [14](#)
- kV1495ClockSettings
 - VME, [14](#)
- kV1495ConfigurationROM
 - VME, [14](#)
- kV1495Control
 - VME, [14](#)
- kV1495FWRevision
 - VME, [14](#)
- kV1495GeoAddress
 - VME, [14](#)
- kV1495HWRevision0
 - VME, [14](#)
- kV1495HWRevision1
 - VME, [14](#)
- kV1495HWRevision2
 - VME, [14](#)
- kV1495HWRevision3
 - VME, [14](#)
- kV1495ModuleReset
 - VME, [14](#)
- kV1495OUI0
 - VME, [14](#)
- kV1495OUI1
 - VME, [14](#)
- kV1495OUI2
 - VME, [14](#)
- kV1495OutputSettings
 - VME, [14](#)
- kV1495SerNum0
 - VME, [14](#)
- kV1495SerNum1
 - VME, [14](#)
- kV1495TDCBoardInterface
 - VME, [14](#)
- kV1495TriggerSettings
 - VME, [14](#)
- kV1495UserFPGAConfig
 - VME, [14](#)
- kV1495UserFPGAFlashMem
 - VME, [14](#)
- kV1495UserFWRevision
 - VME, [14](#)

- leading
 - VME::trailead_t, [101](#)
- Listen
 - Socket, [67](#)
- LOAD_DEF_CONFIG
 - VME::TDCV1x90Opcodes, [20](#)
- LOAD_USER_CONFIG
 - VME::TDCV1x90Opcodes, [20](#)
- magic
 - file_header_t, [37](#)
- MASTER
 - Socket, [66](#)
- MATCH_WIN_WIDTH
 - VME, [16](#)
- Message, [56](#)
 - ~Message, [57](#)
 - Dump, [57](#)
 - fString, [57](#)
 - GetKey, [57](#)
 - GetString, [57](#)
 - IsFromWeb, [57](#)
 - Message, [57](#)
- Messenger, [59](#)
 - ~Messenger, [60](#)
 - AddClient, [61](#)
 - Broadcast, [61](#)
 - Connect, [61](#)
 - Disconnect, [61](#)
 - DisconnectClient, [61](#)
 - fNumAttempts, [63](#)
 - fPID, [63](#)
 - fStderrPipe, [63](#)
 - fStdoutPipe, [63](#)
 - fWS, [63](#)
 - GetType, [61](#)
 - Messenger, [60](#)
 - ProcessMessage, [61](#)
 - Receive, [62](#)
 - Send, [62](#)
 - StartAcquisition, [62](#)
 - StopAcquisition, [62](#)
 - SwitchClientType, [62](#)
- micro_handshake
 - VME, [15](#)
- nchannels
 - VME::TDCV1x90, [93](#)
- num_hptdc
 - file_header_t, [37](#)
- NumEvents
 - VME::TDCMeasurement, [84](#)
- Object
 - SocketMessage, [73](#)
- OLEADING
 - VME, [14](#)
- OneLine
 - Exception, [35](#)
- operator<<
 - VME::TDCErrorFlag, [77](#)
- OTRAILING
 - VME, [14](#)
- OutputConf
 - VME::BridgeVx718, [23](#)
- OutputOff
 - VME::BridgeVx718, [23](#)
- OutputOn
 - VME::BridgeVx718, [24](#)
- PAIR
 - VME, [14](#)
- pair_lead_res
 - VME::TDCV1x90, [93](#)
- pair_width_res
 - VME::TDCV1x90, [93](#)
- PairMode
 - VME::TDCV1x90Status, [100](#)
- ParseMessage
 - Client, [32](#)
- PrepareConnection
 - Socket, [67](#)
- ProcessMessage
 - Messenger, [61](#)
- PulseTDCBits
 - VME::FPGAUnitV1495, [43](#)
- Purged
 - VME::TDCV1x90Status, [100](#)
- r100ps
 - VME, [16](#)
- r200ps
 - VME, [16](#)
- r25ps
 - VME, [16](#)
- r800ps
 - VME, [16](#)
- R_100ps
 - VME::TDCV1x90Status, [99](#)
- R_200ps

- VME::TDCV1x90Status, [99](#)
- R_25ps
 - VME::TDCV1x90Status, [99](#)
- R_800ps
 - VME::TDCV1x90Status, [99](#)
- READ_OK
 - VME, [15](#)
- READ_ACQ_MOD
 - VME::TDCV1x90Opcodes, [20](#)
- READ_ADJUST_CH
 - VME::TDCV1x90Opcodes, [20](#)
- READ_DEAD_TIME
 - VME::TDCV1x90Opcodes, [20](#)
- READ_DETECTION
 - VME::TDCV1x90Opcodes, [20](#)
- READ_DLL_LOCK
 - VME::TDCV1x90Opcodes, [20](#)
- READ_EEPROM
 - VME::TDCV1x90Opcodes, [20](#)
- READ_EN_PATTERN
 - VME::TDCV1x90Opcodes, [20](#)
- READ_EN_PATTERN32
 - VME::TDCV1x90Opcodes, [20](#)
- READ_ERROR_STATUS
 - VME::TDCV1x90Opcodes, [20](#)
- READ_ERROR_TYPES
 - VME::TDCV1x90Opcodes, [20](#)
- READ_EVENT_SIZE
 - VME::TDCV1x90Opcodes, [20](#)
- READ_FIFO_SIZE
 - VME::TDCV1x90Opcodes, [20](#)
- READ_GLOB_OFFS
 - VME::TDCV1x90Opcodes, [20](#)
- READ_HEAD_TRAILER
 - VME::TDCV1x90Opcodes, [20](#)
- READ_MICRO_REV
 - VME::TDCV1x90Opcodes, [20](#)
- READ_RC_ADJ
 - VME::TDCV1x90Opcodes, [20](#)
- READ_RES
 - VME::TDCV1x90Opcodes, [20](#)
- READ_SETUP_REG
 - VME::TDCV1x90Opcodes, [20](#)
- READ_SETUP_SCANPATH
 - VME::TDCV1x90Opcodes, [20](#)
- READ_SPARE
 - VME::TDCV1x90Opcodes, [20](#)
- READ_STATUS_STREAM
 - VME::TDCV1x90Opcodes, [20](#)
- READ_TDC_ID
 - VME::TDCV1x90Opcodes, [20](#)
- READ_TRG_CONF
 - VME::TDCV1x90Opcodes, [20](#)
- ReadAcquisitionMode
 - VME::TDCV1x90, [90](#)
- ReadDetectionMode
 - VME::TDCV1x90, [90](#)
- ReadRegister
 - VME::GenericBoard, [49](#)
- Receive
 - Client, [32](#)
 - Messenger, [62](#)
- REJECT_MARGIN
 - VME, [16](#)
- RESET_DLL_PLL
 - VME::TDCV1x90Opcodes, [20](#)
- ResetFPGA
 - VME::FPGAUnitV1495, [43](#)
- Resolution
 - VME::TDCV1x90Status, [100](#)
- REV_DATE_MICRO_FW
 - VME::TDCV1x90Opcodes, [20](#)
- run_id
 - file_header_t, [37](#)
- SAVE_RC_ADJ
 - VME::TDCV1x90Opcodes, [20](#)
- SAVE_USER_CONFIG
 - VME::TDCV1x90Opcodes, [20](#)
- SelectConnections
 - Socket, [67](#)
- Send
 - Client, [32](#)
 - Messenger, [62](#)
- SendAndReceive
 - Client, [32](#)
- SendClear
 - VMEReader, [104](#)
- SendMessage
 - Socket, [68](#)
- SendPulse
 - VMEReader, [104](#)
- SET_ADJUST_CH
 - VME::TDCV1x90Opcodes, [20](#)
- SET_DEAD_TIME
 - VME::TDCV1x90Opcodes, [20](#)
- SET_DETECTION
 - VME::TDCV1x90Opcodes, [20](#)
- SET_DLL_CLOCK
 - VME::TDCV1x90Opcodes, [20](#)

- SET_ERROR_TYPES
 - VME::TDCV1x90Opcodes, 20
- SET_EVENT_SIZE
 - VME::TDCV1x90Opcodes, 20
- SET_FIFO_SIZE
 - VME::TDCV1x90Opcodes, 20
- SET_GLOB_OFFS
 - VME::TDCV1x90Opcodes, 20
- SET_KEEP_TOKEN
 - VME::TDCV1x90Opcodes, 20
- SET_PAIR_RES
 - VME::TDCV1x90Opcodes, 20
- SET_RC_ADJ
 - VME::TDCV1x90Opcodes, 20
- SET_REJ_MARGIN
 - VME::TDCV1x90Opcodes, 20
- SET_SW_MARGIN
 - VME::TDCV1x90Opcodes, 20
- SET_TDC_TSET_OUTPUT
 - VME::TDCV1x90Opcodes, 20
- SET_TR_LEAD_LSB
 - VME::TDCV1x90Opcodes, 20
- SET_WIN_OFFS
 - VME::TDCV1x90Opcodes, 20
- SET_WIN_WIDTH
 - VME::TDCV1x90Opcodes, 20
- SetAcquisitionMode
 - VME::TDCV1x90, 90
- SetAlign64
 - VME::TDCV1x90Control, 95
- SetBLTEventNumberRegister
 - VME::TDCV1x90, 90
- SetBusError
 - VME::TDCV1x90Control, 95
- SetChannelDeadTime
 - VME::TDCV1x90, 90
- SetClockSource
 - VME::FPGAUnitV1495Control, 46
- SetCompensation
 - VME::TDCV1x90Control, 96
- SetContinuousStorage
 - VME::TDCV1x90, 90
- SetControl
 - VME::FPGAUnitV1495, 43
 - VME::TDCV1x90, 90
- SetDetectionMode
 - VME::TDCV1x90, 91
- SetDLLClock
 - VME::TDCV1x90, 91
- SetEmptyEvent
 - VME::TDCV1x90Control, 96
- SetErrorMarks
 - VME::TDCV1x90, 91
- SetETTT
 - VME::TDCV1x90, 91
 - VME::TDCV1x90Control, 96
- SetEventFIFO
 - VME::TDCV1x90Control, 96
- SetEventsCollection
 - VME::TDCMeasurement, 84
- SetFIFOSize
 - VME::TDCV1x90, 91
- SetGlobalOffset
 - VME::TDCV1x90, 91
- SetInternalClockPeriod
 - VME::FPGAUnitV1495, 43
- SetInternalTriggerPeriod
 - VME::FPGAUnitV1495, 43
- SetIRQ
 - VME::BridgeVx718, 24
- SetKeyValue
 - SocketMessage, 74
- SetLSBTraileadEdge
 - VME::TDCV1x90, 91
- SetMEBAccess
 - VME::TDCV1x90Control, 96
- SetOutputPulser
 - VME::FPGAUnitV1495, 44
- SetPairModeResolution
 - VME::TDCV1x90, 91
- SetPol
 - VME::TDCV1x90, 91
- SetPort
 - Socket, 68
- SetRCAdjust
 - VME::TDCV1x90, 92
- SetSocketId
 - Socket, 68
- SetSRAMCompensation
 - VME::TDCV1x90Control, 96
- SetStatus
 - VME::TDCV1x90, 92
- SetSWTermination
 - VME::TDCV1x90Control, 96
- SetTDCBits
 - VME::FPGAUnitV1495, 44
- SetTDCEncapsulation
 - VME::TDCV1x90, 92
- SetTermination
 - VME::TDCV1x90Control, 96

- SetTestFIFO
 - VME::TDCV1x90Control, 96
- SetTestMode
 - VME::TDCV1x90, 92
- SetTriggerMatching
 - VME::TDCV1x90, 92
- SetTriggerSource
 - VME::FPGAUnitV1495Control, 46
- SetVerboseLevel
 - VME::TDCV1x90, 92
- SetWindowOffset
 - VME::TDCV1x90, 92
- SetWindowWidth
 - VME::TDCV1x90, 92
- SetWord
 - VME::TDCEvent, 82
- SinglePulse
 - VME::BridgeVx718, 24
- Socket, 64
 - ~Socket, 66
 - AcceptConnections, 66
 - Bind, 66
 - CLIENT, 66
 - Configure, 67
 - Create, 67
 - DETECTOR, 66
 - DumpConnected, 67
 - fAddress, 68
 - fBuffer, 68
 - FetchMessage, 67
 - fMaster, 68
 - fPort, 68
 - fReadFds, 68
 - fSocketId, 68
 - fSocketsConnected, 69
 - GetPort, 67
 - GetSocketId, 67
 - GetSocketType, 67
 - INVALID, 66
 - IsWebSocket, 67
 - Listen, 67
 - MASTER, 66
 - PrepareConnection, 67
 - SelectConnections, 67
 - SendMessage, 68
 - SetPort, 68
 - SetSocketId, 68
 - Socket, 66
 - SocketCollection, 66
 - SocketType, 66
 - Start, 68
 - Stop, 68
 - WEBSOCKET_CLIENT, 66
- Socket communication objects, 9
- SocketCollection
 - Socket, 66
- SocketMessage, 70
 - ~SocketMessage, 73
 - Dump, 73
 - fMessage, 74
 - GetIntValue, 73
 - GetKey, 73
 - GetString, 73
 - GetValue, 73
 - GetVectorValue, 73
 - Object, 73
 - SetKeyValue, 74
 - SocketMessage, 71–73
 - String, 74
- SocketType
 - Socket, 66
- SoftwareClear
 - VME::TDCV1x90, 93
- SoftwareReset
 - VME::TDCV1x90, 93
- spill_id
 - file_header_t, 37
- Start
 - Socket, 68
- StartAcquisition
 - Messenger, 62
- StartPulser
 - VME::BridgeVx718, 24
 - VMEReader, 105
- Stop
 - Socket, 68
- StopAcquisition
 - Messenger, 62
- StopPulser
 - VME::BridgeVx718, 24
 - VMEReader, 105
- String
 - SocketMessage, 74
- SwitchClientType
 - Messenger, 62
- TDCBits
 - VME::FPGAUnitV1495, 41
- TDCCollection
 - VMEReader, 103

- TDCErrors
 - VME::TDCEvent, [79](#)
- TDCErrorsFlag
 - VME::TDCErrorsFlag, [76](#)
- TDCEvent
 - VME::TDCEvent, [80](#)
- TDCEventCollection
 - VME, [13](#)
- TDCHeader
 - VME::TDCEvent, [79](#)
- TDCMeasurement
 - VME::TDCEvent, [79](#)
 - VME::TDCMeasurement, [83](#)
- TDCResolution
 - VME::TDCV1x90Status, [99](#)
- TDCTrailer
 - VME::TDCEvent, [79](#)
- TDCV1x90
 - VME::TDCV1x90, [87](#)
- TDCV1x90Control
 - VME::TDCV1x90Control, [95](#)
- TDCV1x90Register
 - VME, [15](#)
- TDCV1x90Status
 - VME::TDCV1x90Status, [99](#)
- TerminationOn
 - VME::TDCV1x90Status, [100](#)
- TestOutputs
 - VME::BridgeVx718, [24](#)
- total_hits
 - VME::trailead_t, [101](#)
- TRAILEAD
 - VME, [14](#)
- trailead_edge_lsb
 - VME, [16](#)
- trailing
 - VME::trailead_t, [101](#)
- TRG_MATCH
 - VME::TDCV1x90OpCodes, [20](#)
- TRIG_MATCH
 - VME, [13](#)
- TRIG_TIME_SUB
 - VME, [16](#)
- trig_conf
 - VME, [16](#)
- TriggerLost
 - VME::TDCV1x90Status, [100](#)
- TriggerMatching
 - VME::TDCV1x90Status, [100](#)
- TriggerSource
 - VME::FPGAUnitV1495Control, [46](#)
- Type
 - Exception, [35](#)
- TypeString
 - Exception, [35](#)
- UPDATE_SETUP_REG
 - VME::TDCV1x90OpCodes, [20](#)
- UPDATE_SETUP_TDC
 - VME::TDCV1x90OpCodes, [20](#)
- VME, [11](#)
 - AcquisitionMode, [13](#)
 - BridgeType, [13](#)
 - CAEN_V1718, [13](#)
 - CAEN_V2718, [13](#)
 - CONT_STORAGE, [13](#)
 - DetectionMode, [13](#)
 - EXTRA_SEARCH_WIN_WIDTH, [16](#)
 - FPGAUnitV1495Register, [14](#)
 - IOModuleV262Register, [14](#)
 - kBLTEventNumber, [15](#)
 - kBoardInfo0, [15](#)
 - kBoardInfo1, [15](#)
 - kControl, [15](#)
 - kECLLevelWrite, [15](#)
 - kEventCounter, [15](#)
 - kEventFIFO, [15](#)
 - kEventFIFOStatusRegister, [15](#)
 - kEventFIFOStoredRegister, [15](#)
 - kEventStored, [15](#)
 - kFirmwareRev, [15](#)
 - kGeoAddress, [15](#)
 - kIdentifier, [15](#)
 - kInterruptLevel, [15](#)
 - kInterruptVector, [15](#)
 - kMCSTBase, [15](#)
 - kMCSTControl, [15](#)
 - kMicro, [15](#)
 - kMicroHandshake, [15](#)
 - kModuleReset, [15](#)
 - kNIMLevelWrite, [15](#)
 - kNIMPulseRead, [15](#)
 - kNIMPulseWrite, [15](#)
 - kOutputBuffer, [15](#)
 - kROMBoard0, [16](#)
 - kROMBoard1, [16](#)
 - kROMBoard2, [16](#)
 - kROMOui0, [16](#)

- kROMOui1, [16](#)
- kROMOui2, [16](#)
- kROMRevis0, [16](#)
- kROMRevis1, [16](#)
- kROMRevis2, [16](#)
- kROMRevis3, [16](#)
- kROMSerNum0, [16](#)
- kROMSerNum1, [16](#)
- kSoftwareClear, [15](#)
- kStatus, [15](#)
- kV1495Board0, [14](#)
- kV1495Board1, [14](#)
- kV1495Board2, [14](#)
- kV1495ClockSettings, [14](#)
- kV1495ConfigurationROM, [14](#)
- kV1495Control, [14](#)
- kV1495FWRevision, [14](#)
- kV1495GeoAddress, [14](#)
- kV1495HWRevision0, [14](#)
- kV1495HWRevision1, [14](#)
- kV1495HWRevision2, [14](#)
- kV1495HWRevision3, [14](#)
- kV1495ModuleReset, [14](#)
- kV1495OUI0, [14](#)
- kV1495OUI1, [14](#)
- kV1495OUI2, [14](#)
- kV1495OutputSettings, [14](#)
- kV1495SerNum0, [14](#)
- kV1495SerNum1, [14](#)
- kV1495TDCBoardInterface, [14](#)
- kV1495TriggerSettings, [14](#)
- kV1495UserFPGAConfig, [14](#)
- kV1495UserFPGAFlashMem, [14](#)
- kV1495UserFWRevision, [14](#)
- MATCH_WIN_WIDTH, [16](#)
- micro_handshake, [15](#)
- OLEADING, [14](#)
- OTRAILING, [14](#)
- PAIR, [14](#)
- r100ps, [16](#)
- r200ps, [16](#)
- r25ps, [16](#)
- r800ps, [16](#)
- READ_OK, [15](#)
- REJECT_MARGIN, [16](#)
- TDCEventCollection, [13](#)
- TDCV1x90Register, [15](#)
- TRAILHEAD, [14](#)
- trailead_edge_lsb, [16](#)
- TRIG_MATCH, [13](#)
- TRIG_TIME_SUB, [16](#)
- trig_conf, [16](#)
- WIN_OFFSET, [16](#)
- WRITE_OK, [15](#)
- VME::BridgeVx718, [21](#)
- ~BridgeVx718, [23](#)
- BridgeVx718, [23](#)
- CheckConfiguration, [23](#)
- fHasIRQ, [24](#)
- GetHandle, [23](#)
- GetIRQStatus, [23](#)
- InputConf, [23](#)
- InputRead, [23](#)
- IRQ1, [22](#)
- IRQ2, [22](#)
- IRQ3, [22](#)
- IRQ4, [22](#)
- IRQ5, [22](#)
- IRQ6, [22](#)
- IRQ7, [22](#)
- IRQId, [22](#)
- OutputConf, [23](#)
- OutputOff, [23](#)
- OutputOn, [24](#)
- SetIRQ, [24](#)
- SinglePulse, [24](#)
- StartPulser, [24](#)
- StopPulser, [24](#)
- TestOutputs, [24](#)
- WaitIRQ, [24](#)
- VME::BridgeVx718Control, [25](#)
- ~BridgeVx718Control, [25](#)
- BridgeVx718Control, [25](#)
- fWord, [27](#)
- GetAddressIncrement, [25](#)
- GetArbiterType, [26](#)
- GetBusReqLevel, [26](#)
- GetBusTimeout, [26](#)
- GetInterruptReq, [26](#)
- GetReleaseType, [26](#)
- GetRequesterType, [26](#)
- GetSysRes, [26](#)
- VME::BridgeVx718Status, [28](#)
- ~BridgeVx718Status, [29](#)
- BridgeVx718Status, [29](#)
- Dump, [29](#)
- fWord, [29](#)
- GetBERR, [29](#)
- GetDipSwitch, [29](#)
- GetDTACK, [29](#)

- GetSystemControl, [29](#)
- GetSystemReset, [29](#)
- GetUSBType, [29](#)
- VME::FPGAUnitV1495, [40](#)
 - ~FPGAUnitV1495, [41](#)
 - CheckBoardVersion, [41](#)
 - DumpFWInformation, [41](#)
 - FPGAUnitV1495, [41](#)
 - GetCAENFirmwareRevision, [41](#)
 - GetControl, [42](#)
 - GetGeoAddress, [42](#)
 - GetHardwareRevision, [42](#)
 - GetInternalClockPeriod, [42](#)
 - GetInternalTriggerPeriod, [42](#)
 - GetOutputPulser, [42](#)
 - GetSerialNumber, [42](#)
 - GetTDCBits, [42](#)
 - GetUserFirmwareRevision, [43](#)
 - kClear, [41](#)
 - kReset, [41](#)
 - kTrigger, [41](#)
 - PulseTDCBits, [43](#)
 - ResetFPGA, [43](#)
 - SetControl, [43](#)
 - SetInternalClockPeriod, [43](#)
 - SetInternalTriggerPeriod, [43](#)
 - SetOutputPulser, [44](#)
 - SetTDCBits, [44](#)
 - TDCBits, [41](#)
- VME::FPGAUnitV1495Control, [45](#)
 - ~FPGAUnitV1495Control, [46](#)
 - ClockSource, [46](#)
 - ExternalClock, [46](#)
 - ExternalTrigger, [46](#)
 - FPGAUnitV1495Control, [46](#)
 - fWord, [47](#)
 - GetClockSource, [46](#)
 - GetTriggerSource, [46](#)
 - GetWord, [46](#)
 - InternalClock, [46](#)
 - InternalTrigger, [46](#)
 - SetClockSource, [46](#)
 - SetTriggerSource, [46](#)
 - TriggerSource, [46](#)
- VME::GenericBoard, [48](#)
 - ~GenericBoard, [49](#)
 - fBaseAddr, [50](#)
 - fHandle, [50](#)
 - GenericBoard, [49](#)
 - ReadRegister, [49](#)
 - WriteRegister, [49](#)
- VME::GlobalOffset, [51](#)
 - coarse, [51](#)
 - fine, [51](#)
- VME::IOModuleV262, [54](#)
 - ~IOModuleV262, [54](#)
 - GetIdentifier, [54](#)
 - GetManufacturerId, [54](#)
 - GetModuleType, [54](#)
 - GetModuleVersion, [54](#)
 - GetSerialNumber, [54](#)
 - IOModuleV262, [54](#)
- VME::TDCErrorFlag, [75](#)
 - ~TDCErrorFlag, [76](#)
 - Dump, [76](#)
 - fWord, [77](#)
 - GetWord, [76](#)
 - HasGroupError, [76](#)
 - HasInternalChipError, [76](#)
 - HasL1BufferOverflow, [76](#)
 - HasReachedEventSizeLimit, [76](#)
 - HasReadoutFIFOOverflow, [76](#)
 - HasTriggerFIFOOverflow, [76](#)
 - operator<<, [77](#)
 - TDCErrorFlag, [76](#)
- VME::TDCEvent, [78](#)
 - ~TDCEvent, [80](#)
 - Dump, [80](#)
 - ETTT, [80](#)
 - EventType, [79](#)
 - Filler, [80](#)
 - fWord, [82](#)
 - GetBunchId, [80](#)
 - GetChannelId, [80](#)
 - GetErrorFlags, [80](#)
 - GetETTT, [80](#)
 - GetEventCount, [80](#)
 - GetEventId, [81](#)
 - GetGeo, [81](#)
 - GetLeadingTime, [81](#)
 - GetStatus, [81](#)
 - GetTDCId, [81](#)
 - GetTrailingTime, [81](#)
 - GetType, [81](#)
 - GetWidth, [81](#)
 - GetWord, [82](#)
 - GetWordCount, [82](#)
 - GlobalHeader, [79](#)
 - GlobalTrailer, [79](#)
 - IsTrailing, [82](#)

- SetWord, [82](#)
- TDCError, [79](#)
- TDCEvent, [80](#)
- TDCHeader, [79](#)
- TDCMeasurement, [79](#)
- TDCTrailer, [79](#)
- VME::TDCMeasurement, [83](#)
- ~TDCMeasurement, [83](#)
- Dump, [83](#)
- fEvents, [84](#)
- fMap, [84](#)
- GetBunchId, [83](#)
- GetChannelId, [84](#)
- GetEventId, [84](#)
- GetLeadingTime, [84](#)
- GetTDCId, [84](#)
- GetToT, [84](#)
- GetTrailingTime, [84](#)
- NumEvents, [84](#)
- SetEventsCollection, [84](#)
- TDCMeasurement, [83](#)
- VME::TDCV1x90, [85](#)
- ~TDCV1x90, [87](#)
- abort, [87](#)
- CheckConfiguration, [87](#)
- DisableChannel, [87](#)
- DLL_Direct_LowRes, [87](#)
- DLL_PLL_HighRes, [87](#)
- DLL_PLL_LowRes, [87](#)
- DLL_PLL_MedRes, [87](#)
- DLLMode, [87](#)
- EnableChannel, [87](#)
- fAcquisitionMode, [93](#)
- fBuffer, [93](#)
- fDetectionMode, [93](#)
- fErrorMarks, [93](#)
- FetchEvents, [87](#)
- fVerb, [93](#)
- fWindowWidth, [93](#)
- gEnd, [93](#)
- GetAcquisitionMode, [88](#)
- GetBLTEventNumberRegister, [88](#)
- GetChannelDeadTime, [88](#)
- GetControl, [88](#)
- GetDetectionMode, [88](#)
- GetDLLClock, [88](#)
- GetErrorMarks, [88](#)
- GetETTT, [88](#)
- GetEventCounter, [88](#)
- GetEventStored, [88](#)
- GetFIFOSize, [89](#)
- GetFirmwareRevision, [89](#)
- GetGlobalOffset, [89](#)
- GetModel, [89](#)
- GetOUI, [89](#)
- GetPoI, [89](#)
- GetRCAdjust, [89](#)
- GetResolution, [89](#)
- GetSerialNumber, [89](#)
- GetStatus, [89](#)
- GetTDCEncapsulation, [89](#)
- GetTestMode, [90](#)
- GetTriggerConfiguration, [90](#)
- GetWindowOffset, [90](#)
- GetWindowWidth, [90](#)
- HardwareReset, [90](#)
- nchannels, [93](#)
- pair_lead_res, [93](#)
- pair_width_res, [93](#)
- ReadAcquisitionMode, [90](#)
- ReadDetectionMode, [90](#)
- SetAcquisitionMode, [90](#)
- SetBLTEventNumberRegister, [90](#)
- SetChannelDeadTime, [90](#)
- SetContinuousStorage, [90](#)
- SetControl, [90](#)
- SetDetectionMode, [91](#)
- SetDLLClock, [91](#)
- SetErrorMarks, [91](#)
- SetETTT, [91](#)
- SetFIFOSize, [91](#)
- SetGlobalOffset, [91](#)
- SetLSBTrileadEdge, [91](#)
- SetPairModeResolution, [91](#)
- SetPoI, [91](#)
- SetRCAdjust, [92](#)
- SetStatus, [92](#)
- SetTDCEncapsulation, [92](#)
- SetTestMode, [92](#)
- SetTriggerMatching, [92](#)
- SetVerboseLevel, [92](#)
- SetWindowOffset, [92](#)
- SetWindowWidth, [92](#)
- SoftwareClear, [93](#)
- SoftwareReset, [93](#)
- TDCV1x90, [87](#)
- WaitMicro, [93](#)
- VME::TDCV1x90Control, [94](#)
- ~TDCV1x90Control, [95](#)
- Dump, [95](#)

- fWord, 97
- GetAlign64, 95
- GetBusError, 95
- GetCompensation, 95
- GetEmptyEvent, 95
- GetETTT, 95
- GetEventFIFO, 95
- GetMEBAccess, 95
- GetSRAMCompensation, 95
- GetSWTermination, 95
- GetTermination, 95
- GetTestFIFO, 95
- GetValue, 95
- SetAlign64, 95
- SetBusError, 95
- SetCompensation, 96
- SetEmptyEvent, 96
- SetETTT, 96
- SetEventFIFO, 96
- SetMEBAccess, 96
- SetSRAMCompensation, 96
- SetSWTermination, 96
- SetTermination, 96
- SetTestFIFO, 96
- TDCV1x90Control, 95
- VME::TDCV1x90OpCodes, 17
 - AUTOLOAD_DEF_CONFI, 20
 - AUTOLOAD_USER_CONF, 20
 - CLEAR_KEEP_TOKEN, 20
 - CONT_STOR, 20
 - DEFAULT_SETUP_REG, 20
 - DIS_ALL_CHANNEL, 20
 - DIS_CHANNEL, 20
 - DIS_ERROR_BYPASS, 20
 - DIS_ERROR_MARK, 20
 - DIS_HEAD_TRAILER, 20
 - DIS_SUB_TRG, 20
 - DISABLE_TEST_MODE, 20
 - EN_ALL_CHANNEL, 20
 - EN_CHANNEL, 20
 - EN_ERROR_BYPASS, 20
 - EN_ERROR_MARK, 20
 - EN_HEAD_TRAILER, 20
 - EN_SUB_TRG, 20
 - ENABLE_TEST_MODE, 20
 - LOAD_DEF_CONFIG, 20
 - LOAD_USER_CONFIG, 20
 - READ_ACQ_MOD, 20
 - READ_ADJUST_CH, 20
 - READ_DEAD_TIME, 20
 - READ_DETECTION, 20
 - READ_DLL_LOCK, 20
 - READ_EEPROM, 20
 - READ_EN_PATTERN, 20
 - READ_EN_PATTERN32, 20
 - READ_ERROR_STATUS, 20
 - READ_ERROR_TYPES, 20
 - READ_EVENT_SIZE, 20
 - READ_FIFO_SIZE, 20
 - READ_GLOB_OFFS, 20
 - READ_HEAD_TRAILER, 20
 - READ_MICRO_REV, 20
 - READ_RC_ADJ, 20
 - READ_RES, 20
 - READ_SETUP_REG, 20
 - READ_SETUP_SCANPATH, 20
 - READ_SPARE, 20
 - READ_STATUS_STREAM, 20
 - READ_TDC_ID, 20
 - READ_TRG_CONF, 20
 - RESET_DLL_PLL, 20
 - REV_DATE_MICRO_FW, 20
 - SAVE_RC_ADJ, 20
 - SAVE_USER_CONFIG, 20
 - SET_ADJUST_CH, 20
 - SET_DEAD_TIME, 20
 - SET_DETECTION, 20
 - SET_DLL_CLOCK, 20
 - SET_ERROR_TYPES, 20
 - SET_EVENT_SIZE, 20
 - SET_FIFO_SIZE, 20
 - SET_GLOB_OFFS, 20
 - SET_KEEP_TOKEN, 20
 - SET_PAIR_RES, 20
 - SET_RC_ADJ, 20
 - SET_REJ_MARGIN, 20
 - SET_SW_MARGIN, 20
 - SET_TDC_TSET_OUTPUT, 20
 - SET_TR_LEAD_LSB, 20
 - SET_WIN_OFFS, 20
 - SET_WIN_WIDTH, 20
 - TRG_MATCH, 20
 - UPDATE_SETUP_REG, 20
 - UPDATE_SETUP_TDC, 20
 - WRITE_EEPROM, 20
 - WRITE_EN_PATTERN, 20
 - WRITE_EN_PATTERN32, 20
 - WRITE_SETUP_REG, 20
 - WRITE_SPARE, 20
- VME::TDCV1x90Status, 98

- ~TDCV1x90Status, 99
- AlmostFull, 99
- BusError, 99
- DataReady, 99
- Dump, 99
- Error, 99
- Full, 100
- fWord, 100
- GetValue, 100
- HeadersEnabled, 100
- PairMode, 100
- Purged, 100
- R_100ps, 99
- R_200ps, 99
- R_25ps, 99
- R_800ps, 99
- Resolution, 100
- TDCResolution, 99
- TDCV1x90Status, 99
- TerminationOn, 100
- TriggerLost, 100
- TriggerMatching, 100
- VME::trailead_t, 101
 - ettt, 101
 - event_count, 101
 - leading, 101
 - total_hits, 101
 - trailing, 101
- VMEReader, 102
 - ~VMEReader, 103
 - Abort, 104
 - AddFPGAUnit, 104
 - AddIOModule, 104
 - AddTDC, 104
 - fBridge, 105
 - fFPGA, 105
 - fIsPulserStarted, 105
 - fOnSocket, 105
 - fSG, 105
 - fTDCCollection, 105
 - GetFPGAUnit, 104
 - GetIOModule, 104
 - GetRunNumber, 104
 - GetTDC, 104
 - SendClear, 104
 - SendPulse, 104
 - StartPulser, 105
 - StopPulser, 105
 - TDCCollection, 103
 - VMEReader, 103
 - WaitIRQ
 - VME::BridgeVx718, 24
 - WaitMicro
 - VME::TDCV1x90, 93
 - WEBSOCKET_CLIENT
 - Socket, 66
 - WIN_OFFSET
 - VME, 16
 - WRITE_OK
 - VME, 15
 - WRITE_EEPROM
 - VME::TDCV1x90Opcodes, 20
 - WRITE_EN_PATTERN
 - VME::TDCV1x90Opcodes, 20
 - WRITE_EN_PATTERN32
 - VME::TDCV1x90Opcodes, 20
 - WRITE_SETUP_REG
 - VME::TDCV1x90Opcodes, 20
 - WRITE_SPARE
 - VME::TDCV1x90Opcodes, 20
 - WriteRegister
 - VME::GenericBoard, 49