

2015 Test beam Run Control

Generated by Doxygen 1.6.1

Wed Jul 29 19:40:20 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	NIM Namespace Reference	11
6.1.1	Enumeration Type Documentation	12
6.1.1.1	HVModuleN470Opcodes	12
6.2	VME Namespace Reference	13
6.2.1	Typedef Documentation	15
6.2.1.1	CFDCollection	15
6.2.1.2	TDCCollection	15
6.2.1.3	TDCEventCollection	16
6.2.2	Enumeration Type Documentation	16
6.2.2.1	AcquisitionMode	16
6.2.2.2	BridgeType	16

6.2.2.3	CAENETControllerV288Answer	16
6.2.2.4	CAENETControllerV288Register	16
6.2.2.5	CFDV812Register	17
6.2.2.6	DetectionMode	17
6.2.2.7	FPGAUnitV1495Register	17
6.2.2.8	IOModuleV262Register	18
6.2.2.9	micro_handshake	18
6.2.2.10	TDCV1x90Register	18
6.2.2.11	trailead_edge_lsb	19
6.2.2.12	trig_conf	20
6.3	VME::TDCV1x90Opcodes Namespace Reference	21
6.3.1	Function Documentation	24
6.3.1.1	AUTOLOAD_DEF_CONFI	24
6.3.1.2	AUTOLOAD_USER_CONF	24
6.3.1.3	CLEAR_KEEP_TOKEN	24
6.3.1.4	CONT_STOR	24
6.3.1.5	DEFAULT_SETUP_REG	24
6.3.1.6	DIS_ALL_CHANNEL	24
6.3.1.7	DIS_CHANNEL	24
6.3.1.8	DIS_ERROR_BYPASS	24
6.3.1.9	DIS_ERROR_MARK	24
6.3.1.10	DIS_HEAD_TRAILER	24
6.3.1.11	DIS_SUB_TRG	24
6.3.1.12	DISABLE_TEST_MODE	24
6.3.1.13	EN_ALL_CHANNEL	24
6.3.1.14	EN_CHANNEL	24
6.3.1.15	EN_ERROR_BYPASS	24
6.3.1.16	EN_ERROR_MARK	24
6.3.1.17	EN_HEAD_TRAILER	24
6.3.1.18	EN_SUB_TRG	24
6.3.1.19	ENABLE_TEST_MODE	24
6.3.1.20	LOAD_DEF_CONFIG	24
6.3.1.21	LOAD_USER_CONFIG	24
6.3.1.22	READ_ACQ_MOD	24

6.3.1.23	READ_ADJUST_CH	24
6.3.1.24	READ_DEAD_TIME	24
6.3.1.25	READ_DETECTION	24
6.3.1.26	READ_DLL_LOCK	24
6.3.1.27	READ_EEPROM	24
6.3.1.28	READ_EN_PATTERN	24
6.3.1.29	READ_EN_PATTERN32	24
6.3.1.30	READ_ERROR_STATUS	24
6.3.1.31	READ_ERROR_TYPES	24
6.3.1.32	READ_EVENT_SIZE	24
6.3.1.33	READ_FIFO_SIZE	24
6.3.1.34	READ_GLOB_OFFS	24
6.3.1.35	READ_HEAD_TRAILER	24
6.3.1.36	READ_MICRO_REV	24
6.3.1.37	READ_RC_ADJ	24
6.3.1.38	READ_RES	24
6.3.1.39	READ_SETUP_REG	24
6.3.1.40	READ_SETUP_SCANPATH	24
6.3.1.41	READ_SPARE	24
6.3.1.42	READ_STATUS_STREAM	24
6.3.1.43	READ_TDC_ID	24
6.3.1.44	READ_TRG_CONF	24
6.3.1.45	RESET_DLL_PLL	24
6.3.1.46	REV_DATE_MICRO_FW	24
6.3.1.47	SAVE_RC_ADJ	24
6.3.1.48	SAVE_USER_CONFIG	24
6.3.1.49	SET_ADJUST_CH	24
6.3.1.50	SET_DEAD_TIME	24
6.3.1.51	SET_DETECTION	24
6.3.1.52	SET_DLL_CLOCK	24
6.3.1.53	SET_ERROR_TYPES	24
6.3.1.54	SET_EVENT_SIZE	24
6.3.1.55	SET_FIFO_SIZE	24
6.3.1.56	SET_GLOB_OFFS	24

6.3.1.57	SET_KEEP_TOKEN	24
6.3.1.58	SET_PAIR_RES	24
6.3.1.59	SET_RC_ADJ	24
6.3.1.60	SET_REJ_MARGIN	24
6.3.1.61	SET_SW_MARGIN	24
6.3.1.62	SET_TDC_TSET_OUTPUT	24
6.3.1.63	SET_TR_LEAD_LSB	24
6.3.1.64	SET_WIN_OFFS	24
6.3.1.65	SET_WIN_WIDTH	24
6.3.1.66	TRG_MATCH	24
6.3.1.67	UPDATE_SETUP_REG	24
6.3.1.68	UPDATE_SETUP_TDC	24
6.3.1.69	WRITE_EEPROM	24
6.3.1.70	WRITE_EN_PATTERN	24
6.3.1.71	WRITE_EN_PATTERN32	24
6.3.1.72	WRITE_SETUP_REG	24
6.3.1.73	WRITE_SPARE	24
7	Data Structure Documentation	25
7.1	VME::BridgeVx718 Class Reference	25
7.1.1	Detailed Description	26
7.1.2	Member Enumeration Documentation	26
7.1.2.1	IRQId	26
7.1.3	Constructor & Destructor Documentation	27
7.1.3.1	BridgeVx718	27
7.1.3.2	~BridgeVx718	27
7.1.4	Member Function Documentation	27
7.1.4.1	CheckConfiguration	27
7.1.4.2	CheckPCIInterface	27
7.1.4.3	GetHandle	27
7.1.4.4	GetIRQStatus	28
7.1.4.5	GetStatus	28
7.1.4.6	InputConf	28
7.1.4.7	InputRead	28

7.1.4.8	OutputConf	28
7.1.4.9	OutputOff	28
7.1.4.10	OutputOn	28
7.1.4.11	Reset	28
7.1.4.12	SetIRQ	28
7.1.4.13	SinglePulse	28
7.1.4.14	StartPulser	28
7.1.4.15	StopPulser	28
7.1.4.16	TestOutputs	28
7.1.4.17	WaitIRQ	28
7.1.5	Field Documentation	28
7.1.5.1	fHasIRQ	28
7.2	VME::BridgeVx718Control Class Reference	30
7.2.1	Constructor & Destructor Documentation	30
7.2.1.1	BridgeVx718Control	30
7.2.1.2	~BridgeVx718Control	30
7.2.2	Member Function Documentation	30
7.2.2.1	GetAddressIncrement	30
7.2.2.2	GetArbiterType	31
7.2.2.3	GetBusReqLevel	31
7.2.2.4	GetBusTimeout	31
7.2.2.5	GetInterruptReq	31
7.2.2.6	GetReleaseType	31
7.2.2.7	GetRequesterType	31
7.2.2.8	GetSysRes	32
7.2.3	Field Documentation	32
7.2.3.1	fWord	32
7.3	VME::BridgeVx718Status Class Reference	33
7.3.1	Constructor & Destructor Documentation	33
7.3.1.1	BridgeVx718Status	33
7.3.1.2	~BridgeVx718Status	33
7.3.2	Member Function Documentation	33
7.3.2.1	Dump	33
7.3.2.2	GetBERR	34

7.3.2.3	GetDipSwitch	34
7.3.2.4	GetDTACK	34
7.3.2.5	GetSystemControl	34
7.3.2.6	GetSystemReset	34
7.3.2.7	GetUSBType	34
7.3.3	Field Documentation	34
7.3.3.1	fWord	34
7.4	VME::CAENETControllerV288 Class Reference	35
7.4.1	Detailed Description	36
7.4.2	Constructor & Destructor Documentation	36
7.4.2.1	CAENETControllerV288	36
7.4.2.2	~CAENETControllerV288	36
7.4.3	Member Function Documentation	36
7.4.3.1	FetchBuffer	36
7.4.3.2	GetStatus	36
7.4.3.3	Reset	37
7.4.3.4	SendBuffer	37
7.4.3.5	WaitForResponse	37
7.4.4	Friends And Related Function Documentation	37
7.4.4.1	operator<<	37
7.4.4.2	operator>>	37
7.5	VME::CAENETControllerV288Status Class Reference	38
7.5.1	Member Enumeration Documentation	38
7.5.1.1	OperationStatus	38
7.5.2	Constructor & Destructor Documentation	39
7.5.2.1	CAENETControllerV288Status	39
7.5.2.2	~CAENETControllerV288Status	39
7.5.3	Member Function Documentation	39
7.5.3.1	GetOperationStatus	39
7.5.4	Field Documentation	39
7.5.4.1	fWord	39
7.6	VME::CFDV812 Class Reference	40
7.6.1	Detailed Description	41
7.6.2	Constructor & Destructor Documentation	41

7.6.2.1	CFDV812	41
7.6.2.2	~CFDV812	41
7.6.3	Member Function Documentation	41
7.6.3.1	CheckConfiguration	41
7.6.3.2	DeadTimeCalculator	42
7.6.3.3	GetFixedCode	42
7.6.3.4	GetManufacturerId	42
7.6.3.5	GetModuleType	42
7.6.3.6	GetModuleVersion	42
7.6.3.7	GetSerialNumber	42
7.6.3.8	OutputWidthCalculator	43
7.6.3.9	SetDeadTime	43
7.6.3.10	SetOutputWidth	43
7.6.3.11	SetPOI	43
7.6.3.12	SetThreshold	44
7.7	NIM::HVModuleN470Values::ChannelStatus Class Reference	45
7.7.1	Member Enumeration Documentation	46
7.7.1.1	SignalStandard	46
7.7.2	Constructor & Destructor Documentation	48
7.7.2.1	ChannelStatus	48
7.7.2.2	~ChannelStatus	48
7.7.3	Member Function Documentation	48
7.7.3.1	Alarm	48
7.7.3.2	Dump	48
7.7.3.3	Enabled	48
7.7.3.4	HVEnabled	48
7.7.3.5	Isel	48
7.7.3.6	Kill	48
7.7.3.7	MaxV	48
7.7.3.8	NonCalibrated	48
7.7.3.9	OVC	48
7.7.3.10	OVV	48
7.7.3.11	Polarity	48
7.7.3.12	RampDown	48

7.7.3.13	RampUp	48
7.7.3.14	Standard	48
7.7.3.15	Trip	48
7.7.3.16	UNV	48
7.7.3.17	Vsel	48
7.7.4	Friends And Related Function Documentation	48
7.7.4.1	operator<<	48
7.7.5	Field Documentation	48
7.7.5.1	fWord	48
7.8	Client Class Reference	50
7.8.1	Detailed Description	51
7.8.2	Constructor & Destructor Documentation	51
7.8.2.1	Client	51
7.8.2.2	Client	51
7.8.2.3	~Client	51
7.8.3	Member Function Documentation	51
7.8.3.1	Announce	51
7.8.3.2	Connect	52
7.8.3.3	Disconnect	52
7.8.3.4	GetType	53
7.8.3.5	ParseMessage	53
7.8.3.6	Receive	53
7.8.3.7	Receive	53
7.8.3.8	Send	54
7.8.3.9	Send	54
7.8.3.10	SendAndReceive	54
7.8.4	Field Documentation	55
7.8.4.1	fClientId	55
7.8.4.2	flsConnected	55
7.8.4.3	fType	55
7.9	file_header_t Struct Reference	56
7.9.1	Detailed Description	56
7.9.2	Field Documentation	56
7.9.2.1	acq_mode	56

7.9.2.2	det_mode	56
7.9.2.3	magic	56
7.9.2.4	num_hptdc	56
7.9.2.5	run_id	56
7.9.2.6	spill_id	56
7.10	FileReader Class Reference	57
7.10.1	Detailed Description	57
7.10.2	Constructor & Destructor Documentation	58
7.10.2.1	FileReader	58
7.10.2.2	FileReader	58
7.10.2.3	~FileReader	58
7.10.3	Member Function Documentation	58
7.10.3.1	Clear	58
7.10.3.2	Dump	58
7.10.3.3	GetNextEvent	58
7.10.3.4	GetNextMeasurement	58
7.10.3.5	GetNumEvents	59
7.10.3.6	GetNumTDCs	59
7.10.3.7	IsOpen	59
7.10.3.8	Open	59
7.10.4	Field Documentation	59
7.10.4.1	fFile	59
7.10.4.2	fHeader	59
7.10.4.3	fNumEvents	59
7.10.4.4	fReadoutMode	59
7.10.4.5	fWriteTime	59
7.11	VME::FPGAUnitV1495 Class Reference	60
7.11.1	Detailed Description	61
7.11.2	Member Enumeration Documentation	61
7.11.2.1	TDCBits	61
7.11.3	Constructor & Destructor Documentation	62
7.11.3.1	FPGAUnitV1495	62
7.11.3.2	~FPGAUnitV1495	62
7.11.4	Member Function Documentation	62

7.11.4.1	CheckBoardVersion	62
7.11.4.2	ClearOutputPulser	62
7.11.4.3	DumpFWInformation	63
7.11.4.4	GetCAENFirmwareRevision	63
7.11.4.5	GetControl	63
7.11.4.6	GetGeoAddress	63
7.11.4.7	GetHardwareRevision	64
7.11.4.8	GetInternalClockPeriod	64
7.11.4.9	GetInternalTriggerPeriod	64
7.11.4.10	GetOutputDelay	64
7.11.4.11	GetOutputPulser	65
7.11.4.12	GetScalerValue	65
7.11.4.13	GetSerialNumber	65
7.11.4.14	GetTDCBits	65
7.11.4.15	GetUserFirmwareRevision	65
7.11.4.16	PulseTDCBits	66
7.11.4.17	ResetFPGA	66
7.11.4.18	SetControl	66
7.11.4.19	SetInternalClockPeriod	66
7.11.4.20	SetInternalTriggerPeriod	67
7.11.4.21	SetOutputDelay	67
7.11.4.22	SetOutputPulser	67
7.11.4.23	SetOutputPulserPOI	67
7.11.4.24	SetTDCBits	68
7.11.4.25	StartScaler	68
7.11.4.26	StopScaler	68
7.11.5	Field Documentation	68
7.11.5.1	fScalerStarted	68
7.12	VME::FPGAUnitV1495Control Class Reference	69
7.12.1	Detailed Description	70
7.12.2	Member Enumeration Documentation	70
7.12.2.1	ClockSource	70
7.12.2.2	SignalSource	70
7.12.2.3	TriggerSource	70

7.12.3	Constructor & Destructor Documentation	71
7.12.3.1	FPGAUnitV1495Control	71
7.12.3.2	~FPGAUnitV1495Control	71
7.12.4	Member Function Documentation	71
7.12.4.1	Dump	71
7.12.4.2	GetBit	71
7.12.4.3	GetClockSource	71
7.12.4.4	GetScalerStatus	71
7.12.4.5	GetSignalSource	71
7.12.4.6	GetTriggerSource	72
7.12.4.7	GetWord	72
7.12.4.8	SetBit	72
7.12.4.9	SetClockSource	72
7.12.4.10	SetScalerReset	72
7.12.4.11	SetScalerStatus	73
7.12.4.12	SetSignalSource	73
7.12.4.13	SetTriggerSource	73
7.12.5	Field Documentation	73
7.12.5.1	fWord	73
7.13	VME::GenericBoard< Register, am > Class Template Reference . . .	74
7.13.1	Constructor & Destructor Documentation	75
7.13.1.1	GenericBoard	75
7.13.1.2	~GenericBoard	75
7.13.2	Member Function Documentation	75
7.13.2.1	ReadRegister	75
7.13.2.2	ReadRegister	75
7.13.2.3	WriteRegister	75
7.13.2.4	WriteRegister	76
7.13.3	Field Documentation	76
7.13.3.1	fBaseAddr	76
7.13.3.2	fHandle	76
7.14	VME::GlobalOffset Struct Reference	77
7.14.1	Field Documentation	77
7.14.1.1	coarse	77

7.14.1.2	fine	77
7.15	HTTPMessage Class Reference	78
7.15.1	Detailed Description	78
7.15.2	Constructor & Destructor Documentation	78
7.15.2.1	HTTPMessage	78
7.15.2.2	HTTPMessage	79
7.15.3	Member Function Documentation	79
7.15.3.1	Decode	79
7.15.3.2	Dump	79
7.15.3.3	Encode	79
7.15.3.4	GetKey	79
7.15.4	Field Documentation	79
7.15.4.1	fOriginalString	79
7.15.4.2	fWS	79
7.16	NIM::HVModuleN470 Class Reference	80
7.16.1	Constructor & Destructor Documentation	81
7.16.1.1	HVModuleN470	81
7.16.1.2	~HVModuleN470	81
7.16.2	Member Function Documentation	81
7.16.2.1	GetFWRevision	81
7.16.2.2	GetModuleId	81
7.16.2.3	ReadChannelValues	81
7.16.2.4	ReadMonitoringValues	82
7.16.2.5	ReadRegister	82
7.16.2.6	SetChannelI0	82
7.16.2.7	SetChannelI1	83
7.16.2.8	SetChannelV0	83
7.16.2.9	SetChannelV1	83
7.16.2.10	WriteRegister	83
7.16.2.11	WriteRegister	84
7.16.3	Field Documentation	84
7.16.3.1	fAddress	84
7.16.3.2	fController	84
7.17	NIM::HVModuleN470ChannelValues Class Reference	85

7.17.1 Detailed Description	85
7.17.2 Constructor & Destructor Documentation	86
7.17.2.1 HVModuleN470ChannelValues	86
7.17.2.2 ~HVModuleN470ChannelValues	86
7.17.3 Member Function Documentation	86
7.17.3.1 ChannelStatus	86
7.17.3.2 Dump	86
7.17.3.3 I0	87
7.17.3.4 I1	87
7.17.3.5 Imon	87
7.17.3.6 MaxV	87
7.17.3.7 RampDown	87
7.17.3.8 RampUp	87
7.17.3.9 Trip	87
7.17.3.10 V0	87
7.17.3.11 V1	87
7.17.3.12 Vmon	87
7.17.4 Field Documentation	87
7.17.4.1 fChannelId	87
7.17.4.2 fValues	87
7.18 NIM::HVModuleN470Values Class Reference	88
7.18.1 Detailed Description	88
7.18.2 Constructor & Destructor Documentation	89
7.18.2.1 HVModuleN470Values	89
7.18.2.2 ~HVModuleN470Values	89
7.18.3 Member Function Documentation	89
7.18.3.1 Dump	89
7.18.3.2 GetChannelStatus	89
7.18.3.3 Imon	89
7.18.3.4 Vmax	89
7.18.3.5 Vmon	89
7.18.4 Field Documentation	89
7.18.4.1 fValues	89
7.19 VME::IOModuleV262 Class Reference	90

7.19.1	Constructor & Destructor Documentation	90
7.19.1.1	IOModuleV262	90
7.19.1.2	~IOModuleV262	90
7.19.2	Member Function Documentation	90
7.19.2.1	GetIdentifier	90
7.19.2.2	GetManufacturerId	91
7.19.2.3	GetModuleType	91
7.19.2.4	GetModuleVersion	91
7.19.2.5	GetSerialNumber	91
7.20	Message Class Reference	92
7.20.1	Detailed Description	92
7.20.2	Constructor & Destructor Documentation	93
7.20.2.1	Message	93
7.20.2.2	Message	93
7.20.2.3	Message	93
7.20.2.4	~Message	93
7.20.3	Member Function Documentation	93
7.20.3.1	Dump	93
7.20.3.2	GetKey	93
7.20.3.3	GetString	93
7.20.3.4	IsFromWeb	93
7.20.4	Field Documentation	93
7.20.4.1	fString	93
7.21	Messenger Class Reference	95
7.21.1	Detailed Description	96
7.21.2	Constructor & Destructor Documentation	96
7.21.2.1	Messenger	96
7.21.2.2	Messenger	96
7.21.2.3	~Messenger	97
7.21.3	Member Function Documentation	97
7.21.3.1	AddClient	97
7.21.3.2	Broadcast	97
7.21.3.3	Connect	98
7.21.3.4	Disconnect	98

7.21.3.5	DisconnectClient	98
7.21.3.6	GetType	99
7.21.3.7	ProcessMessage	99
7.21.3.8	Receive	100
7.21.3.9	Send	100
7.21.3.10	SendAll	101
7.21.3.11	StartAcquisition	101
7.21.3.12	StopAcquisition	101
7.21.3.13	SwitchClientType	101
7.21.4	Field Documentation	102
7.21.4.1	fNumAttempts	102
7.21.4.2	fPID	102
7.21.4.3	fStderrPipe	102
7.21.4.4	fStdoutPipe	102
7.21.4.5	fWS	102
7.22	VME::PCIInterfaceA2818 Class Reference	103
7.22.1	Constructor & Destructor Documentation	103
7.22.1.1	PCIInterfaceA2818	103
7.22.1.2	~PCIInterfaceA2818	103
7.22.2	Member Function Documentation	103
7.22.2.1	GetFWRevision	103
7.22.3	Field Documentation	103
7.22.3.1	fHandle	103
7.23	Socket Class Reference	104
7.23.1	Detailed Description	105
7.23.2	Member Typedef Documentation	106
7.23.2.1	SocketCollection	106
7.23.3	Member Enumeration Documentation	106
7.23.3.1	SocketType	106
7.23.4	Constructor & Destructor Documentation	106
7.23.4.1	Socket	106
7.23.4.2	Socket	106
7.23.4.3	~Socket	106
7.23.5	Member Function Documentation	106

7.23.5.1	AcceptConnections	106
7.23.5.2	Bind	107
7.23.5.3	Configure	107
7.23.5.4	Create	107
7.23.5.5	DumpConnected	107
7.23.5.6	FetchMessage	107
7.23.5.7	GetPort	107
7.23.5.8	GetSocketId	107
7.23.5.9	GetSocketType	107
7.23.5.10	IsWebSocket	107
7.23.5.11	Listen	107
7.23.5.12	PrepareConnection	108
7.23.5.13	SelectConnections	108
7.23.5.14	SendMessage	108
7.23.5.15	SetPort	108
7.23.5.16	SetSocketId	108
7.23.5.17	Start	108
7.23.5.18	Stop	108
7.23.6	Field Documentation	109
7.23.6.1	fAddress	109
7.23.6.2	fBuffer	109
7.23.6.3	fMaster	109
7.23.6.4	fPort	109
7.23.6.5	fReadFds	109
7.23.6.6	fSocketId	109
7.23.6.7	fSocketsConnected	109
7.24	SocketMessage Class Reference	110
7.24.1	Detailed Description	111
7.24.2	Constructor & Destructor Documentation	112
7.24.2.1	SocketMessage	112
7.24.2.2	SocketMessage	112
7.24.2.3	SocketMessage	112
7.24.2.4	SocketMessage	112
7.24.2.5	SocketMessage	112

7.24.2.6	SocketMessage	112
7.24.2.7	SocketMessage	112
7.24.2.8	SocketMessage	112
7.24.2.9	SocketMessage	113
7.24.2.10	SocketMessage	113
7.24.2.11	SocketMessage	113
7.24.2.12	~SocketMessage	113
7.24.3	Member Function Documentation	113
7.24.3.1	Dump	113
7.24.3.2	GetCleanedValue	113
7.24.3.3	GetIntValue	113
7.24.3.4	GetKey	113
7.24.3.5	GetString	113
7.24.3.6	GetValue	114
7.24.3.7	GetVectorValue	114
7.24.3.8	Object	114
7.24.3.9	SetKeyValue	114
7.24.3.10	SetKeyValue	114
7.24.3.11	SetKeyValue	114
7.24.3.12	SetKeyValue	114
7.24.3.13	String	115
7.24.4	Field Documentation	115
7.24.4.1	fMessage	115
7.25	VME::TDCErrorFlag Class Reference	116
7.25.1	Detailed Description	116
7.25.2	Constructor & Destructor Documentation	117
7.25.2.1	TDCErrorFlag	117
7.25.2.2	~TDCErrorFlag	117
7.25.3	Member Function Documentation	117
7.25.3.1	Dump	117
7.25.3.2	GetWord	117
7.25.3.3	HasGroupError	117
7.25.3.4	HasInternalChipError	117
7.25.3.5	HasL1BufferOverflow	117

7.25.3.6	HasReachedEventSizeLimit	117
7.25.3.7	HasReadoutFIFOOverflow	117
7.25.3.8	HasTriggerFIFOOverflow	117
7.25.4	Friends And Related Function Documentation	118
7.25.4.1	operator<<	118
7.25.5	Field Documentation	118
7.25.5.1	fWord	118
7.26	VME::TDCEvent Class Reference	119
7.26.1	Detailed Description	120
7.26.2	Member Enumeration Documentation	120
7.26.2.1	EventType	120
7.26.3	Constructor & Destructor Documentation	121
7.26.3.1	TDCEvent	121
7.26.3.2	TDCEvent	121
7.26.3.3	TDCEvent	121
7.26.3.4	~TDCEvent	121
7.26.4	Member Function Documentation	121
7.26.4.1	Dump	121
7.26.4.2	GetBunchId	121
7.26.4.3	GetChannelId	121
7.26.4.4	GetErrorFlags	121
7.26.4.5	GetETTT	121
7.26.4.6	GetEventCount	122
7.26.4.7	GetEventId	122
7.26.4.8	GetGeo	122
7.26.4.9	GetStatus	122
7.26.4.10	GetTDCId	122
7.26.4.11	GetTime	122
7.26.4.12	GetType	122
7.26.4.13	GetWidth	123
7.26.4.14	GetWord	123
7.26.4.15	GetWordCount	123
7.26.4.16	IsTrailing	123
7.26.4.17	SetWord	123

7.26.5	Field Documentation	123
7.26.5.1	fWord	123
7.27	VME::TDCMeasurement Class Reference	124
7.27.1	Detailed Description	124
7.27.2	Constructor & Destructor Documentation	124
7.27.2.1	TDCMeasurement	124
7.27.2.2	TDCMeasurement	124
7.27.2.3	~TDCMeasurement	125
7.27.3	Member Function Documentation	125
7.27.3.1	Dump	125
7.27.3.2	GetBunchId	125
7.27.3.3	GetChannelId	125
7.27.3.4	GetETTT	125
7.27.3.5	GetEventId	125
7.27.3.6	GetLeadingTime	125
7.27.3.7	GetTDCId	125
7.27.3.8	GetToT	125
7.27.3.9	GetTrailingTime	126
7.27.3.10	NumErrors	126
7.27.3.11	NumEvents	126
7.27.3.12	SetEventsCollection	126
7.27.4	Field Documentation	126
7.27.4.1	fEvents	126
7.27.4.2	fMap	126
7.28	VME::TDCV1x90 Class Reference	127
7.28.1	Detailed Description	129
7.28.2	Member Enumeration Documentation	129
7.28.2.1	DLLMode	129
7.28.3	Constructor & Destructor Documentation	129
7.28.3.1	TDCV1x90	129
7.28.3.2	~TDCV1x90	130
7.28.4	Member Function Documentation	130
7.28.4.1	abort	130
7.28.4.2	CheckConfiguration	130

7.28.4.3	DisableChannel	130
7.28.4.4	EnableChannel	130
7.28.4.5	FetchEvents	130
7.28.4.6	GetAcquisitionMode	131
7.28.4.7	GetBLTEventNumberRegister	131
7.28.4.8	GetChannelDeadTime	131
7.28.4.9	GetControl	131
7.28.4.10	GetDetectionMode	132
7.28.4.11	GetDLLClock	132
7.28.4.12	GetErrorMarks	132
7.28.4.13	GetETTT	132
7.28.4.14	GetEventCounter	132
7.28.4.15	GetEventStored	132
7.28.4.16	GetFIFOSize	133
7.28.4.17	GetFirmwareRevision	133
7.28.4.18	GetGlobalOffset	133
7.28.4.19	GetModel	133
7.28.4.20	GetOUI	133
7.28.4.21	GetPoI	134
7.28.4.22	GetRCAdjust	134
7.28.4.23	GetResolution	134
7.28.4.24	GetSerialNumber	134
7.28.4.25	GetStatus	135
7.28.4.26	GetTDCEncapsulation	135
7.28.4.27	GetTestMode	135
7.28.4.28	GetTriggerConfiguration	135
7.28.4.29	GetWindowOffset	136
7.28.4.30	GetWindowWidth	136
7.28.4.31	HardwareReset	136
7.28.4.32	ReadAcquisitionMode	136
7.28.4.33	ReadDetectionMode	136
7.28.4.34	SetAcquisitionMode	136
7.28.4.35	SetBLTEventNumberRegister	137
7.28.4.36	SetChannelDeadTime	137

7.28.4.37 SetContinuousStorage	137
7.28.4.38 SetControl	137
7.28.4.39 SetDetectionMode	138
7.28.4.40 SetDLLClock	138
7.28.4.41 SetErrorMarks	138
7.28.4.42 SetETTT	138
7.28.4.43 SetFIFOSize	139
7.28.4.44 SetGlobalOffset	139
7.28.4.45 SetLSBTraileadEdge	139
7.28.4.46 SetPairModeResolution	139
7.28.4.47 SetPoI	140
7.28.4.48 SetRCAdjust	140
7.28.4.49 SetStatus	140
7.28.4.50 SetTDCEncapsulation	140
7.28.4.51 SetTestMode	141
7.28.4.52 SetTriggerMatching	141
7.28.4.53 SetVerboseLevel	141
7.28.4.54 SetWindowOffset	141
7.28.4.55 SetWindowWidth	142
7.28.4.56 SoftwareClear	142
7.28.4.57 SoftwareReset	142
7.28.4.58 WaitMicro	142
7.28.5 Field Documentation	143
7.28.5.1 fAcquisitionMode	143
7.28.5.2 fBuffer	143
7.28.5.3 fDetectionMode	143
7.28.5.4 fErrorMarks	143
7.28.5.5 fVerb	143
7.28.5.6 fWindowWidth	143
7.28.5.7 gEnd	143
7.28.5.8 nchannels	143
7.28.5.9 pair_lead_res	143
7.28.5.10 pair_width_res	143
7.29 VME::TDCV1x90Control Class Reference	144

7.29.1 Detailed Description	144
7.29.2 Constructor & Destructor Documentation	145
7.29.2.1 TDCV1x90Control	145
7.29.2.2 ~TDCV1x90Control	145
7.29.3 Member Function Documentation	145
7.29.3.1 Dump	145
7.29.3.2 GetAlign64	145
7.29.3.3 GetBusError	145
7.29.3.4 GetCompensation	145
7.29.3.5 GetEmptyEvent	145
7.29.3.6 GetETTT	145
7.29.3.7 GetEventFIFO	145
7.29.3.8 GetMEBAccess	145
7.29.3.9 GetSRAMCompensation	145
7.29.3.10 GetSWTermination	145
7.29.3.11 GetTermination	145
7.29.3.12 GetTestFIFO	145
7.29.3.13 GetValue	145
7.29.3.14 SetAlign64	145
7.29.3.15 SetBusError	146
7.29.3.16 SetCompensation	146
7.29.3.17 SetEmptyEvent	146
7.29.3.18 SetETTT	146
7.29.3.19 SetEventFIFO	146
7.29.3.20 SetMEBAccess	146
7.29.3.21 SetSRAMCompensation	146
7.29.3.22 SetSWTermination	146
7.29.3.23 SetTermination	146
7.29.3.24 SetTestFIFO	147
7.29.4 Field Documentation	147
7.29.4.1 fWord	147
7.30 VME::TDCV1x90Status Class Reference	148
7.30.1 Detailed Description	148
7.30.2 Member Enumeration Documentation	149

7.30.2.1	TDCResolution	149
7.30.3	Constructor & Destructor Documentation	149
7.30.3.1	TDCV1x90Status	149
7.30.3.2	~TDCV1x90Status	149
7.30.4	Member Function Documentation	149
7.30.4.1	AlmostFull	149
7.30.4.2	BusError	149
7.30.4.3	DataReady	149
7.30.4.4	Dump	149
7.30.4.5	Error	149
7.30.4.6	Error	150
7.30.4.7	Full	150
7.30.4.8	GetValue	150
7.30.4.9	HeadersEnabled	150
7.30.4.10	PairMode	150
7.30.4.11	Purged	150
7.30.4.12	Resolution	150
7.30.4.13	TerminationOn	150
7.30.4.14	TriggerLost	150
7.30.4.15	TriggerMatching	150
7.30.5	Field Documentation	150
7.30.5.1	fWord	150
7.31	VME::trailead_t Struct Reference	151
7.31.1	Field Documentation	151
7.31.1.1	ettt	151
7.31.1.2	event_count	151
7.31.1.3	leading	151
7.31.1.4	total_hits	151
7.31.1.5	trailing	151
7.32	VMEReader Class Reference	152
7.32.1	Detailed Description	154
7.32.2	Member Typedef Documentation	155
7.32.2.1	OutputFiles	155
7.32.3	Constructor & Destructor Documentation	155

7.32.3.1	VMERReader	155
7.32.3.2	~VMERReader	155
7.32.4	Member Function Documentation	156
7.32.4.1	Abort	156
7.32.4.2	AddCFD	156
7.32.4.3	AddFPGAUnit	156
7.32.4.4	AddHVModule	156
7.32.4.5	AddIOModule	157
7.32.4.6	AddTDC	157
7.32.4.7	GetCFD	157
7.32.4.8	GetCFDCollection	157
7.32.4.9	GetFPGAUnit	157
7.32.4.10	GetHVModule	158
7.32.4.11	GetIOModule	158
7.32.4.12	GetNumCFD	158
7.32.4.13	GetNumTDC	158
7.32.4.14	GetOutputFile	158
7.32.4.15	GetRunNumber	158
7.32.4.16	GetTDC	158
7.32.4.17	GetTDCCollection	158
7.32.4.18	ReadXML	158
7.32.4.19	ReadXML	159
7.32.4.20	SendClear	159
7.32.4.21	SendPulse	159
7.32.4.22	SetOutputFile	160
7.32.4.23	StartPulser	160
7.32.4.24	StopPulser	160
7.32.4.25	UseSocket	160
7.32.5	Field Documentation	160
7.32.5.1	fBridge	160
7.32.5.2	fCAENET	160
7.32.5.3	fCFDCollection	160
7.32.5.4	fFPGA	161
7.32.5.5	fHV	161

7.32.5.6	fIsPulserStarted	161
7.32.5.7	fOnSocket	161
7.32.5.8	fOutputFiles	161
7.32.5.9	fSG	161
7.32.5.10	fTDCCollection	161

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:

NIM	11
VME	13
VME::TDCV1x90Opcodes	21

Chapter 3

Data Structure Index

3.1 Class Hierarchy

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

VME::BridgeVx718Control	30
VME::BridgeVx718Status	33
VME::CAENETControllerV288Status	38
NIM::HVModuleN470Values::ChannelStatus	45
file_header_t	56
FileReader	57
VME::FPGAUnitV1495Control	69
VME::GenericBoard< Register, am >	74
VME::GenericBoard< CAENETControllerV288Register, cvA24_U_DATA >	74
VME::CAENETControllerV288	35
VME::GenericBoard< CFDV812Register, cvA24_U_DATA >	74
VME::CFDV812	40
VME::GenericBoard< CVRegisters, cvA32_U_DATA >	74
VME::BridgeVx718	25
VME::GenericBoard< FPGAUnitV1495Register, cvA32_U_DATA >	74
VME::FPGAUnitV1495	60
VME::GenericBoard< IOModuleV262Register, cvA24_U_DATA >	74
VME::IOModuleV262	90
VME::GenericBoard< TDCV1x90Register, cvA32_U_DATA >	74
VME::TDCV1x90	127
VME::GlobalOffset	77
NIM::HVModuleN470	80
NIM::HVModuleN470ChannelValues	85
NIM::HVModuleN470Values	88
Message	92
HTTPMessage	78
SocketMessage	110

VME::PCIInterfaceA2818	103
Socket	104
Client	50
VMEReader	152
Messenger	95
VME::TDCErrorFlag	116
VME::TDCEvent	119
VME::TDCMeasurement	124
VME::TDCV1x90Control	144
VME::TDCV1x90Status	148
VME::trailead_t	151

Chapter 4

Data Structure Index

4.1 Data Structures

Here are the data structures with brief descriptions:

VME::BridgeVx718 (Class defining the VME bridge)	25
VME::BridgeVx718Control	30
VME::BridgeVx718Status	33
VME::CAENETControllerV288 (Handler for a CAEN V288 CAENET controller)	35
VME::CAENETControllerV288Status	38
VME::CFDV812 (Controller for a CAEN V812 constant fraction discriminator)	40
NIM::HVModuleN470Values::ChannelStatus	45
Client (Base client object for the socket)	50
file_header_t (Header to the output files)	56
FileReader (Handler for a TDC output file readout)	57
VME::FPGAUnitV1495	60
VME::FPGAUnitV1495Control	69
VME::GenericBoard< Register, am >	74
VME::GlobalOffset	77
HTTPMessage (Message to be transmitted through a WebSocket protocol)	78
NIM::HVModuleN470	80
NIM::HVModuleN470ChannelValues (Single channel monitoring values for the HV power supply)	85
NIM::HVModuleN470Values (General monitoring values for the HV power supply)	88
VME::IOModuleV262	90
Message (Base socket message type)	92
Messenger (Base master object for the socket)	95
VME::PCIInterfaceA2818	103
Socket (Base socket object from which clients/master from a socket inherit)	104
SocketMessage (Socket-passed message type)	110
VME::TDCErrorFlag (Error flags handler)	116

VME::TDCEvent (HPTDC event parser)	119
VME::TDCMeasurement	124
VME::TDCV1x90	127
VME::TDCV1x90Control (TDC control register)	144
VME::TDCV1x90Status (TDC status register)	148
VME::trailead_t	151
VMEReader	152

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 NIM Namespace Reference

Data Structures

- class [HVModuleN470Values](#)

General monitoring values for the HV power supply.

- class [HVModuleN470ChannelValues](#)

Single channel monitoring values for the HV power supply.

- class [HVModuleN470](#)

Enumerations

- enum [HVModuleN470Opcodes](#) {

[kN470GeneralInfo](#) = 0x00, [kN470MonStatus](#) = 0x01, [kN470OperationalParams](#) = 0x02, [kN470V0Value](#) = 0x03,

[kN470I0Value](#) = 0x04, [kN470V1Value](#) = 0x05, [kN470I1Value](#) = 0x06, [kN470TripValue](#) = 0x07,

[kN470RampUpValue](#) = 0x08, [kN470RampDownValue](#) = 0x09, [kN470ChannelOn](#) = 0x0a, [kN470ChannelOff](#) = 0x0b,

[kN470KillAllChannels](#) = 0x0c, [kN470ClearAlarm](#) = 0x0d, [kN470EnableFrontPanel](#) = 0x0e, [kN470DisableFrontPanel](#) = 0x0f,

[kN470TTLLLevel](#) = 0x10, [kN470NIMLevel](#) = 0x11 }

6.1.1 Enumeration Type Documentation

6.1.1.1 enum NIM::HVModuleN470Opcodes

Enumerator:

- kN470GeneralInfo*
- kN470MonStatus*
- kN470OperationalParams*
- kN470V0Value*
- kN470I0Value*
- kN470V1Value*
- kN470I1Value*
- kN470TripValue*
- kN470RampUpValue*
- kN470RampDownValue*
- kN470ChannelOn*
- kN470ChannelOff*
- kN470KillAllChannels*
- kN470ClearAlarm*
- kN470EnableFrontPanel*
- kN470DisableFrontPanel*
- kN470TTLLLevel*
- kN470NIMLevel*

6.2 VME Namespace Reference

Namespaces

- namespace [TDCV1x90Opcodes](#)

Data Structures

- class [BridgeVx718Status](#)
- class [BridgeVx718Control](#)
- class [BridgeVx718](#)

class defining the VME bridge

- class [CAENETControllerV288Status](#)
- class [CAENETControllerV288](#)

Handler for a CAEN V288 CAENET controller.

- class [CFDV812](#)

Controller for a CAEN V812 constant fraction discriminator.

- class [FPGAUnitV1495Control](#)
- class [FPGAUnitV1495](#)
- class [GenericBoard](#)
- class [IOModuleV262](#)
- class [PCIInterfaceA2818](#)
- 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::map< uint32_t, [VME::CFDV812 *](#) > [CFDCollection](#)
Mapper from physical [VME](#) addresses to pointers to CFD objects.
- typedef std::vector< [TDCEvent](#) > [TDCEventCollection](#)
- typedef std::map< uint32_t, [VME::TDCV1x90 *](#) > [TDCCollection](#)
Mapper from physical [VME](#) addresses to pointers to TDC objects.

Enumerations

- enum [BridgeType](#) { [CAEN_V1718](#), [CAEN_V2718](#) }
Compatible bridge types.
- enum [CAENETControllerV288Register](#) {
[kV288DataBuffer](#) = 0x00, [kV288Status](#) = 0x02, [kV288Transmission](#) = 0x04,
[kV288ModuleReset](#) = 0x06,
[kV288IRQVector](#) = 0x08 }
- enum [CAENETControllerV288Answer](#) {
[cnSuccess](#) = 0x0000, [cnBusy](#) = 0xff00, [cnUnrecognizedCode](#) = 0xff01, [cnIncorrectValue](#) = 0xff02,
[cnNoData](#) = 0xffff, [cnIncorrectHCC](#) = 0xfffe, [cnWrongModuleAddress](#) = 0xffff
}
- enum [CFDV812Register](#) {
[kV812ThresholdChannel0](#) = 0x00, [kV812OutputWidthGroup0](#) = 0x40,
[kV812OutputWidthGroup1](#) = 0x42, [kV812DeadTimeGroup0](#) = 0x44,
[kV812DeadTimeGroup1](#) = 0x46, [kV812MajorityThreshold](#) = 0x48,
[kV812PatternOfInhibit](#) = 0x4a, [kV812TestPulse](#) = 0x4c,
[kV812FixedCode](#) = 0xfa, [kV812Info0](#) = 0xfc, [kV812Info1](#) = 0xfe }
- enum [FPGAUnitV1495Register](#) {
[kV1495ScalerCounter](#) = 0x100c, [kV1495DelaySettings](#) = 0x1010,
[kV1495UserFWRevision](#) = 0x1014, [kV1495TDCBoardInterface](#) = 0x1018,
[kV1495ClockSettings](#) = 0x101c, [kV1495Control](#) = 0x1020,
[kV1495TriggerSettings](#) = 0x1024, [kV1495OutputSettings](#) = 0x1028,
[kV1495GeoAddress](#) = 0x8008, [kV1495UserFPGAFlashMem](#) = 0x8014,
[kV1495UserFPGAConfig](#) = 0x8016, [kV1495ModuleReset](#) = 0x800a,
[kV1495FWRevision](#) = 0x800c, [kV1495ConfigurationROM](#) = 0x8100,
[kV1495OUI2](#) = 0x8124, [kV1495OUI1](#) = 0x8128,
[kV1495OUI0](#) = 0x812c, [kV1495Board2](#) = 0x8134, [kV1495Board1](#) = 0x8138,
[kV1495Board0](#) = 0x813c,
[kV1495HWRevision3](#) = 0x8140, [kV1495HWRevision2](#) = 0x8144,
[kV1495HWRevision1](#) = 0x8148, [kV1495HWRevision0](#) = 0x814c,
[kV1495SerNum0](#) = 0x8180, [kV1495SerNum1](#) = 0x8184 }

- enum `IOModuleV262Register` {
`kECLLevelWrite` = 0x04, `kNIMLevelWrite` = 0x06, `kNIMPulseWrite` = 0x08,
`kNIMPulseRead` = 0x0a,
`kIdentifier` = 0xfa, `kBoardInfo0` = 0xfc, `kBoardInfo1` = 0xfe }
- enum `AcquisitionMode` { `CONT_STORAGE`, `TRIG_MATCH` }
TDC acquisition mode.
- enum `DetectionMode` { `PAIR` = 0x0, `OTRILING` = 0x1, `OLEADING` = 0x2,
`TRILEAD` = 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, `kMi-`
`croHandshake` = 0x1030,
`kEventFIFO` = 0x1038, `kEventFIFOStoredRegister` = 0x103c, `kEventFIFOSta-`
`tusRegister` = 0x103e, `kROMOui2` = 0x4024,
`kROMOui1` = 0x4028, `kROMOui0` = 0x402c, `kROMBoard2` = 0x4034, `kROM-`
`Board1` = 0x4038,
`kROMBoard0` = 0x403c, `kROMRevis3` = 0x4040, `kROMRevis2` = 0x4044,
`kROMRevis1` = 0x4048,
`kROMRevis0` = 0x404c, `kROMSerNum1` = 0x4080, `kROMSerNum0` = 0x4084
}

6.2.1 Typedef Documentation

6.2.1.1 typedef std::map<uint32_t,VME::CFDV812*> VME::CFDCollection

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

6.2.1.2 typedef std::map<uint32_t,VME::TDCV1x90*> VME::TDCCollection

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

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

6.2.2 Enumeration Type Documentation

6.2.2.1 enum VME::AcquisitionMode

TDC acquisition mode.

Author:

Laurent Forthomme <laurent.forthomme@cern.ch>

Enumerator:

CONT_STORAGE

TRIG_MATCH

6.2.2.2 enum VME::BridgeType

Compatible bridge types.

Enumerator:

CAEN_V1718

CAEN_V2718

6.2.2.3 enum VME::CAENETControllerV288Answer

Enumerator:

cnSuccess

cnBusy

cnUnrecognizedCode

cnIncorrectValue

cnNoData

cnIncorrectHCC

cnWrongModuleAddress

6.2.2.4 enum VME::CAENETControllerV288Register

Enumerator:

kV288DataBuffer

kV288Status

kV288Transmission

kV288ModuleReset

kV288IRQVector

6.2.2.5 enum VME::CFDV812Register

Enumerator:

kV812ThresholdChannel0
kV812OutputWidthGroup0
kV812OutputWidthGroup1
kV812DeadTimeGroup0
kV812DeadTimeGroup1
kV812MajorityThreshold
kV812PatternOfInhibit
kV812TestPulse
kV812FixedCode
kV812Info0
kV812Info1

6.2.2.6 enum VME::DetectionMode

Enumerator:

PAIR
OTRAILING
OLEADING
TRAILEAD

6.2.2.7 enum VME::FPGAUnitV1495Register

Enumerator:

kV1495ScalerCounter
kV1495DelaySettings
kV1495UserFWRevision
kV1495TDCBoardInterface
kV1495ClockSettings
kV1495Control
kV1495TriggerSettings
kV1495OutputSettings
kV1495GeoAddress
kV1495UserFPGAFlashMem
kV1495UserFPGAConfig
kV1495ModuleReset

kV1495FWRevision
kV1495ConfigurationROM
kV1495OUI2
kV1495OUI1
kV1495OUI0
kV1495Board2
kV1495Board1
kV1495Board0
kV1495HWRevision3
kV1495HWRevision2
kV1495HWRevision1
kV1495HWRevision0
kV1495SerNum0
kV1495SerNum1

6.2.2.8 enum VME::IOModuleV262Register

Enumerator:

kECLLevelWrite
kNIMLevelWrite
kNIMPulseWrite
kNIMPulseRead
kIdentifier
kBoardInfo0
kBoardInfo1

6.2.2.9 enum VME::micro_handshake

Enumerator:

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

6.2.2.10 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

6.2.2.11 enum VME::trailead_edge_lsb

Enumerator:

r800ps
r200ps
r100ps
r25ps

6.2.2.12 enum VME::trig_conf**Enumerator:***MATCH_WIN_WIDTH**WIN_OFFSET**EXTRA_SEARCH_WIN_WIDTH**REJECT_MARGIN**TRIG_TIME_SUB*

6.3 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)

6.3.1 Function Documentation

- 6.3.1.1 Opcode VME::TDCV1x90Opcodes::AUTOLOAD_DEF_CONFI (0x0900)
- 6.3.1.2 Opcode VME::TDCV1x90Opcodes::AUTOLOAD_USER_CONF (0x0800)
- 6.3.1.3 Opcode VME::TDCV1x90Opcodes::CLEAR_KEEP_TOKEN (0x0400)
- 6.3.1.4 Opcode VME::TDCV1x90Opcodes::CONT_STOR (0x0100)
- 6.3.1.5 Opcode VME::TDCV1x90Opcodes::DEFAULT_SETUP_REG (0x7300)
- 6.3.1.6 Opcode VME::TDCV1x90Opcodes::DIS_ALL_CHANNEL (0x4300)
- 6.3.1.7 Opcode VME::TDCV1x90Opcodes::DIS_CHANNEL (0x4100)
- 6.3.1.8 Opcode VME::TDCV1x90Opcodes::DIS_ERROR_BYPASS (0x3800)
- 6.3.1.9 Opcode VME::TDCV1x90Opcodes::DIS_ERROR_MARK (0x3600)
- 6.3.1.10 Opcode VME::TDCV1x90Opcodes::DIS_HEAD_TRAILER (0x3100)
- 6.3.1.11 Opcode VME::TDCV1x90Opcodes::DIS_SUB_TRG (0x1500)
- 6.3.1.12 Opcode VME::TDCV1x90Opcodes::DISABLE_TEST_MODE (0xc600)
- 6.3.1.13 Opcode VME::TDCV1x90Opcodes::EN_ALL_CHANNEL (0x4200)
- 6.3.1.14 Opcode VME::TDCV1x90Opcodes::EN_CHANNEL (0x4000)
- 6.3.1.15 Opcode VME::TDCV1x90Opcodes::EN_ERROR_BYPASS (0x3700)
- 6.3.1.16 Opcode VME::TDCV1x90Opcodes::EN_ERROR_MARK (0x3500)
- 6.3.1.17 Opcode VME::TDCV1x90Opcodes::EN_HEAD_TRAILER (0x3000)
- 6.3.1.18 Opcode VME::TDCV1x90Opcodes::EN_SUB_TRG (0x1400)
- 6.3.1.19 Opcode VME::TDCV1x90Opcodes::ENABLE_TEST_MODE (0xc500)
- 6.3.1.20 Opcode VME::TDCV1x90Opcodes::LOAD_DEF_CONFIG (0x0500)
- 6.3.1.21 Opcode VME::TDCV1x90Opcodes::LOAD_USER_CONFIG (0x0700)
- 6.3.1.22 Opcode VME::TDCV1x90Opcodes::READ_ACQ_MOD (0x0200)
- 6.3.1.23 ~~Opcode VME::TDCV1x90Opcodes::READ_ADJUST_CH (0x5200)~~
- 6.3.1.24 Opcode VME::TDCV1x90Opcodes::READ_DEAD_TIME (0x2900)
- 6.3.1.25 Opcode VME::TDCV1x90Opcodes::READ_DETECTION (0x2300)
- 6.3.1.26 Opcode VME::TDCV1x90Opcodes::READ_DLL_LOCK (0x7500)
- 6.3.1.27 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 [CheckPCIInterface](#) (const char *device) const
- void [CheckConfiguration](#) () const
- void [TestOutputs](#) () const
- void [Reset](#) () const
Perform a system reset of the module.

- [BridgeVx718Status GetStatus](#) () const
- void [SetIRQ](#) (unsigned int irq, bool enable=true)
- void [WaitIRQ](#) (unsigned int irq, unsigned long timeout=1000) const
- unsigned int [GetIRQStatus](#) () const
- void [OutputConf](#) (CVOutputSelect output) const

Set and control the output lines.

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

Set and read the input lines.

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

Private Attributes

- bool [fHasIRQ](#)

7.1.1 Detailed Description

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

Author:

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

Date:

Jun 2010

7.1.2 Member Enumeration Documentation

7.1.2.1 enum VME::BridgeVx718::IRQId

Enumerator:

[IRQ1](#)
[IRQ2](#)
[IRQ3](#)
[IRQ4](#)
[IRQ5](#)
[IRQ6](#)
[IRQ7](#)

7.1.3 Constructor & Destructor Documentation

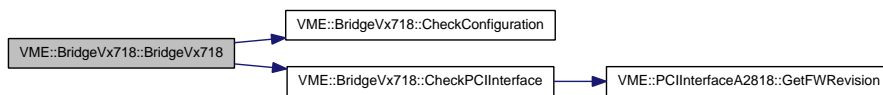
7.1.3.1 VME::BridgeVx718::BridgeVx718 (const char * *device*, BridgeType *type*)

Constructor. Bridge class constructor

Parameters:

- ← *device* Device identifier on the [VME](#) crate
- ← *type* Device type (1718/2718)

Here is the call graph for this function:



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

Destructor. Bridge class destructor

7.1.4 Member Function Documentation

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

7.1.4.2 void VME::BridgeVx718::CheckPCIInterface (const char * *device*) const

Here is the call graph for this function:



7.1.4.3 int32_t VME::BridgeVx718::GetHandle () const [inline]

Bridge's handle value.

Returns:

Handle value

7.1.4.4 `unsigned int VME::BridgeVx718::GetIRQStatus () const`

7.1.4.5 `BridgeVx718Status VME::BridgeVx718::GetStatus () const`

7.1.4.6 `void VME::BridgeVx718::InputConf (CVInputSelect input) const`

Set and read the input lines.

7.1.4.7 `void VME::BridgeVx718::InputRead (CVInputSelect input) const`

7.1.4.8 `void VME::BridgeVx718::OutputConf (CVOutputSelect output) const`

Set and control the output lines.

7.1.4.9 `void VME::BridgeVx718::OutputOff (unsigned short output) const`

7.1.4.10 `void VME::BridgeVx718::OutputOn (unsigned short output) const`

7.1.4.11 `void VME::BridgeVx718::Reset () const`

Perform a system reset of the module.

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

7.1.4.13 `void VME::BridgeVx718::SinglePulse (unsigned short channel) const`

Here is the call graph for this function:

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

7.1.4.15 `void VME::BridgeVx718::StopPulser () const`

7.1.4.16 `void VME::BridgeVx718::TestOutputs () const`

Here is the call graph for this function:

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

7.1.5 Field Documentation

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

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

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

7.2 VME::BridgeVx718Control Class Reference

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

Public Member Functions

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

Private Attributes

- uint16_t [fWord](#)

7.2.1 Constructor & Destructor Documentation

- 7.2.1.1 [VME::BridgeVx718Control::BridgeVx718Control \(uint16_t word\)](#)
[inline]
- 7.2.1.2 [virtual VME::BridgeVx718Control::~~BridgeVx718Control \(\)](#)
[inline, virtual]

7.2.2 Member Function Documentation

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

Address Increment.

Returns:

true if enabled, else false (FIFO mode)

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

Arbiter type.

Returns:

true if "Round Robin", else fixed priority

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

VME bus timeout.

Returns:

true if 1400 us, else 50 us

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

Release type.

Returns:

true if release on request, else release when done

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

Requester type.

Returns:

true if demand, else fair

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

7.2.3 Field Documentation

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

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

- `include/VME_BridgeVx718.h`

7.3 VME::BridgeVx718Status Class Reference

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

Public Member Functions

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

Private Attributes

- uint16_t [fWord](#)

7.3.1 Constructor & Destructor Documentation

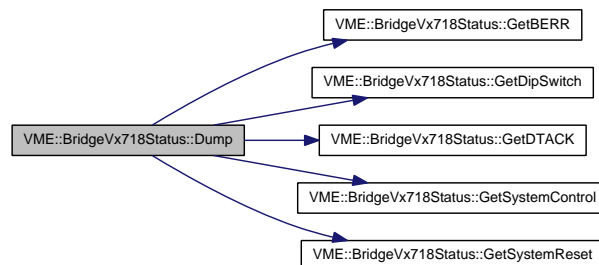
7.3.1.1 VME::BridgeVx718Status::BridgeVx718Status (uint16_t word) **[inline]**

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

7.3.2 Member Function Documentation

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

Here is the call graph for this function:



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]`

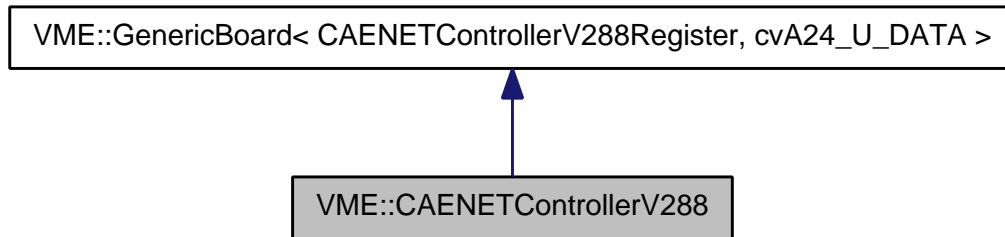
The documentation for this class was generated from the following file:

- `include/VME_BridgeVx718.h`

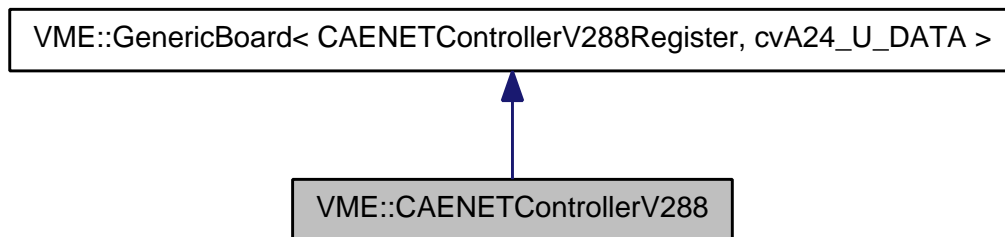
7.4 VME::CAENETControllerV288 Class Reference

Handler for a CAEN V288 CAENET controller.

#include <VME_CAENETControllerV288.h> Inheritance diagram for VME::CAENETControllerV288:



Collaboration diagram for VME::CAENETControllerV288:



Public Member Functions

- [CAENETControllerV288](#) (int32_t handle, uint32_t baseaddr)
- [~CAENETControllerV288](#) ()
- void [Reset](#) () const
- [CAENETControllerV288Status](#) [GetStatus](#) () const
- void [SendBuffer](#) () const
Send the whole buffer through the network.
- std::vector< uint16_t > [FetchBuffer](#) (unsigned int num_words) const
Retrieve the network buffer.
- bool [WaitForResponse](#) ([CAENETControllerV288Answer](#) *response, unsigned int max_trials=-1) const

Friends

- void [operator<<](#) (const [CAENETControllerV288](#) &cnt, uint16_t word)

Fill the buffer with an additional 16-bit word.

- `uint16_t & operator>> (const CAENETControllerV288 &cnt, uint16_t &word)`

Read back a 16-bit word from the buffer.

7.4.1 Detailed Description

Handler for a CAEN V288 CAENET controller.

Author:

Laurent Forthomme <laurent.forthomme@cern.ch>

Date:

23 Jul 2015

7.4.2 Constructor & Destructor Documentation

7.4.2.1 `VME::CAENETControllerV288::CAENETControllerV288 (int32_t handle, uint32_t baseaddr)`

7.4.2.2 `VME::CAENETControllerV288::~~CAENETControllerV288 ()`

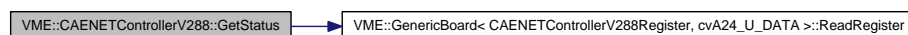
7.4.3 Member Function Documentation

7.4.3.1 `std::vector< uint16_t > VME::CAENETControllerV288::FetchBuffer (unsigned int num_words = 1) const`

Retrieve the network buffer.

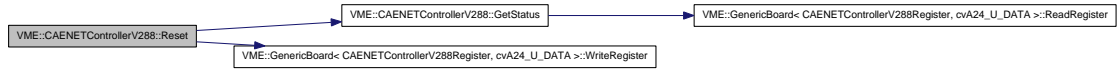
7.4.3.2 `CAENETControllerV288Status VME::CAENETControllerV288::GetStatus () const`

Here is the call graph for this function:



7.4.3.3 void VME::CAENETControllerV288::Reset () const

Here is the call graph for this function:



7.4.3.4 void VME::CAENETControllerV288::SendBuffer () const

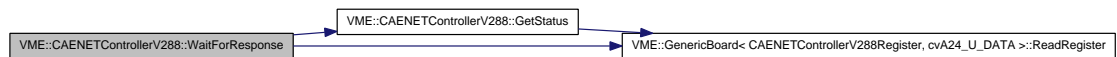
Send the whole buffer through the network.

Here is the call graph for this function:



7.4.3.5 bool VME::CAENETControllerV288::WaitForResponse (CAENETControllerV288Answer * response, unsigned int max_trials = -1) const

Here is the call graph for this function:



7.4.4 Friends And Related Function Documentation

7.4.4.1 void operator<< (const CAENETControllerV288 & cnt, uint16_t word) [friend]

Fill the buffer with an additional 16-bit word.

7.4.4.2 uint16_t& operator>> (const CAENETControllerV288 & cnt, uint16_t & word) [friend]

Read back a 16-bit word from the buffer.

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

- include/VME_CAENETControllerV288.h
- src/VME_CAENETControllerV288.cpp

7.5 VME::CAENETControllerV288Status Class Reference

```
#include <VME_CAENETControllerV288.h>
```

Public Types

- enum [OperationStatus](#) { [Valid](#) = 0x0, [Invalid](#) = 0x1 }

Public Member Functions

- [CAENETControllerV288Status](#) (uint16_t word)
- [~CAENETControllerV288Status](#) ()
- [OperationStatus](#) [GetOperationStatus](#) () const

Private Attributes

- uint16_t [fWord](#)

7.5.1 Member Enumeration Documentation

7.5.1.1 enum VME::CAENETControllerV288Status::OperationStatus

Enumerator:

Valid

Invalid

7.5.2 Constructor & Destructor Documentation

7.5.2.1 VME::CAENETControllerV288Status::CAENETControllerV288Status
(uint16_t *word*) [inline]

7.5.2.2 VME::CAENETControllerV288Status::~~CAENETControllerV288Status
() [inline]

7.5.3 Member Function Documentation

7.5.3.1 OperationStatus
VME::CAENETControllerV288Status::GetOperationStatus () const
[inline]

7.5.4 Field Documentation

7.5.4.1 uint16_t VME::CAENETControllerV288Status::fWord [private]

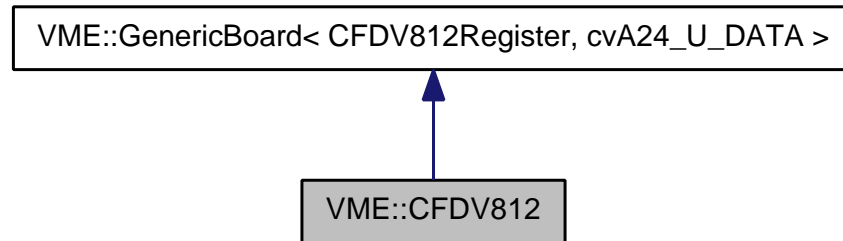
The documentation for this class was generated from the following file:

- include/VME_CAENETControllerV288.h

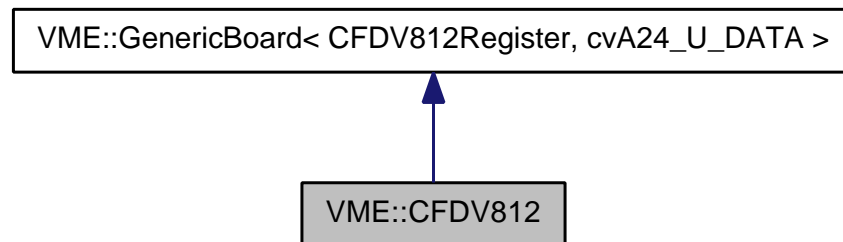
7.6 VME::CFDV812 Class Reference

Controller for a CAEN V812 constant fraction discriminator.

#include <VME_CFDV812.h> Inheritance diagram for VME::CFDV812:



Collaboration diagram for VME::CFDV812:



Public Member Functions

- [CFDV812](#) (int32_t bhandle, uint32_t baseaddr)
- [~CFDV812](#) ()
- void [CheckConfiguration](#) () const
- unsigned short [GetFixedCode](#) () const
- unsigned short [GetManufacturerId](#) () const
- unsigned short [GetModuleType](#) () const
- unsigned short [GetModuleVersion](#) () const
- unsigned short [GetSerialNumber](#) () const
- void [SetPOI](#) (unsigned short poi) const
Set the pattern of inhibit (list of enabled channels).
- void [SetThreshold](#) (unsigned short channel_id, unsigned short value) const
Set the threshold for one single channel, in units of 1 mV.
- void [SetOutputWidth](#) (unsigned short group_id, unsigned short value) const
Set the discriminated pulse output width for one group of 8 channels.

- void [SetDeadTime](#) (unsigned short group_id, unsigned short value) const
Set the discrimination dead time for one group of 8 channels.

Private Member Functions

- float [OutputWidthCalculator](#) (unsigned short value) const
- float [DeadTimeCalculator](#) (unsigned short value) const

7.6.1 Detailed Description

Controller for a CAEN V812 constant fraction discriminator.

Author:

Laurent Forthomme <laurent.forthomme@cern.ch>

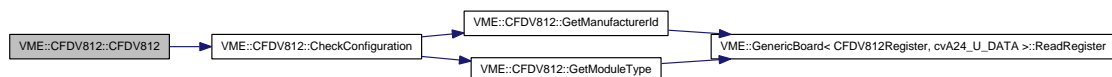
Date:

22 Jul 2015

7.6.2 Constructor & Destructor Documentation

7.6.2.1 VME::CFDV812::CFDV812 (int32_t bhandle, uint32_t baseaddr)

Here is the call graph for this function:

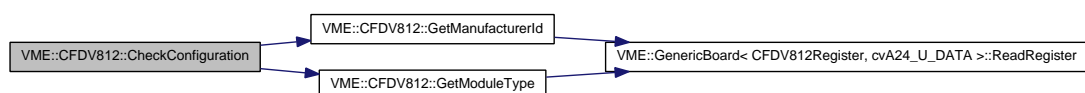


7.6.2.2 VME::CFDV812::~~CFDV812 () [inline]

7.6.3 Member Function Documentation

7.6.3.1 void VME::CFDV812::CheckConfiguration () const

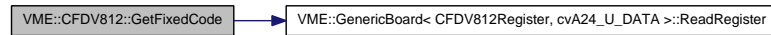
Here is the call graph for this function:



7.6.3.2 float VME::CFDV812::DeadTimeCalculator (unsigned short *value*) const [private]

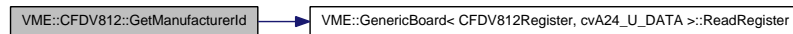
7.6.3.3 unsigned short VME::CFDV812::GetFixedCode () const

Here is the call graph for this function:



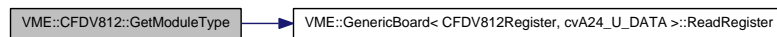
7.6.3.4 unsigned short VME::CFDV812::GetManufacturerId () const

Here is the call graph for this function:



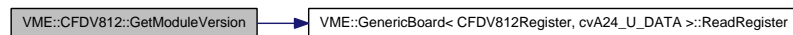
7.6.3.5 unsigned short VME::CFDV812::GetModuleType () const

Here is the call graph for this function:



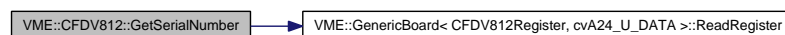
7.6.3.6 unsigned short VME::CFDV812::GetModuleVersion () const

Here is the call graph for this function:



7.6.3.7 unsigned short VME::CFDV812::GetSerialNumber () const

Here is the call graph for this function:



7.6.3.8 float VME::CFDV812::OutputWidthCalculator (unsigned short *value*)
const [private]

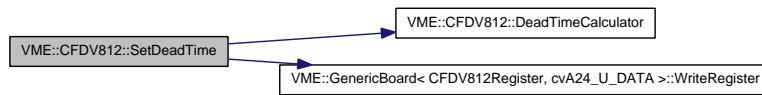
7.6.3.9 void VME::CFDV812::SetDeadTime (unsigned short *group_id*,
unsigned short *value*) const

Set the discrimination dead time for one group of 8 channels.

Parameters:

← *group_id* Group of 8 channels (either 0 for 0-7, or 1 for 8-15)

Here is the call graph for this function:



7.6.3.10 void VME::CFDV812::SetOutputWidth (unsigned short *group_id*,
unsigned short *value*) const

Set the discriminated pulse output width for one group of 8 channels.

Parameters:

← *group_id* Group of 8 channels (either 0 for 0-7, or 1 for 8-15)

Here is the call graph for this function:



7.6.3.11 void VME::CFDV812::SetPOI (unsigned short *poi*) const

Set the pattern of inhibit (list of enabled channels).

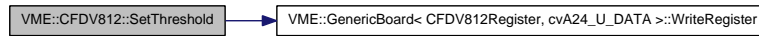
Here is the call graph for this function:



7.6.3.12 void VME::CFDV812::SetThreshold (unsigned short *channel_id*, unsigned short *value*) const

Set the threshold for one single channel, in units of 1 mV.

Here is the call graph for this function:



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

- include/VME_CFDV812.h
- src/VME_CFDV812.cpp

7.7 NIM::HVModuleN470Values::ChannelStatus Class Reference

```
#include <NIM_HVModuleN470.h>
```

Public Types

- enum [SignalStandard](#) { [NIM](#) = 0x0, [TTL](#) = 0x1 }

Public Member Functions

- [ChannelStatus](#) (unsigned short word)
- [~ChannelStatus](#) ()
- bool [Enabled](#) () const
- bool [OVC](#) () const
- bool [OVV](#) () const
- bool [UNV](#) () const
- bool [Trip](#) () const
- bool [RampUp](#) () const
- bool [RampDown](#) () const
- bool [MaxV](#) () const
- bool [Polarity](#) () const
- bool [Vsel](#) () const
- bool [Isel](#) () const
- bool [Kill](#) () const
- bool [HVEnabled](#) () const
- [SignalStandard Standard](#) () const
- bool [NonCalibrated](#) () const
- bool [Alarm](#) () const
- void [Dump](#) () const

Private Attributes

- unsigned short [fWord](#)

Friends

- std::ostream & [operator<<](#) (std::ostream &os, const [ChannelStatus](#) &cs)

7.7.1 Member Enumeration Documentation

7.7.1.1 enum NIM::HVModuleN470Values::ChannelStatus::SignalStandard

Enumerator:

NIM

TTL

7.7.2 Constructor & Destructor Documentation

7.7.2.1 `NIM::HVModuleN470Values::ChannelStatus::ChannelStatus`
(unsigned short word) [inline]

7.7.2.2 `NIM::HVModuleN470Values::ChannelStatus::~~ChannelStatus ()`
[inline]

7.7.3 Member Function Documentation

7.7.3.1 `bool NIM::HVModuleN470Values::ChannelStatus::Alarm () const`
[inline]

7.7.3.2 `void NIM::HVModuleN470Values::ChannelStatus::Dump () const`
[inline]

7.7.3.3 `bool NIM::HVModuleN470Values::ChannelStatus::Enabled () const`
[inline]

7.7.3.4 `bool NIM::HVModuleN470Values::ChannelStatus::HVEnabled () const`
[inline]

7.7.3.5 `bool NIM::HVModuleN470Values::ChannelStatus::Isel () const`
[inline]

7.7.3.6 `bool NIM::HVModuleN470Values::ChannelStatus::Kill () const`
[inline]

7.7.3.7 `bool NIM::HVModuleN470Values::ChannelStatus::MaxV () const`
[inline]

7.7.3.8 `bool NIM::HVModuleN470Values::ChannelStatus::NonCalibrated ()`
`const [inline]`

7.7.3.9 `bool NIM::HVModuleN470Values::ChannelStatus::OVC () const`
[inline]

7.7.3.10 `bool NIM::HVModuleN470Values::ChannelStatus::OVV () const`
[inline]

7.7.3.11 `bool NIM::HVModuleN470Values::ChannelStatus::Polarity () const`
[inline]

7.7.3.12 `bool NIM::HVModuleN470Values::ChannelStatus::RampDown ()`
`const [inline]`

7.7.3.13 `bool NIM::HVModuleN470Values::ChannelStatus::RampUp () const`
[inline]

7.7.3.14 `SignalStandard`

`NIM::HVModuleN470Values::ChannelStatus::Standard () const`
[inline]

7.7.3.15 `bool NIM::HVModuleN470Values::ChannelStatus::Trip () const`
[inline]

7.7.3.16 `bool NIM::HVModuleN470Values::ChannelStatus::UNV () const`
[inline]

- `include/NIM_HVModuleN470.h`

7.8 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.
- [SocketMessage Receive](#) (const MessageKey &key)
- 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`

7.8.1 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

7.8.2 Constructor & Destructor Documentation

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

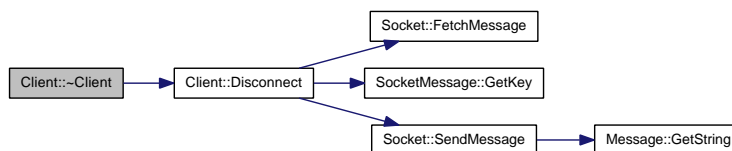
General void client constructor.

7.8.2.2 `Client::Client (int port)`

Bind a socket client to a given port.

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

Here is the call graph for this function:

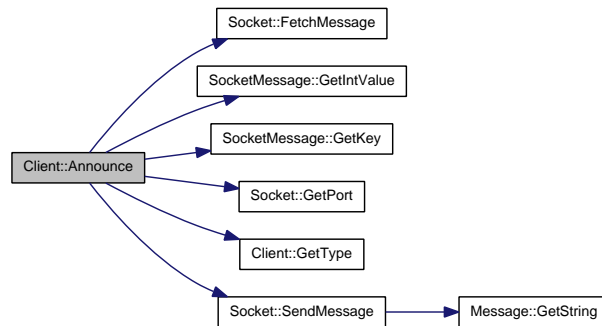


7.8.3 Member Function Documentation

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

Announce our entry on the socket to its master.

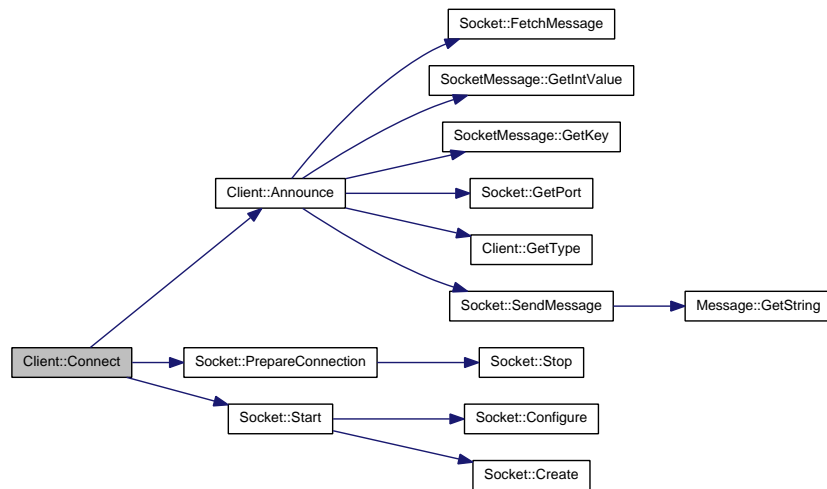
Here is the call graph for this function:



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

Bind this client to the socket.

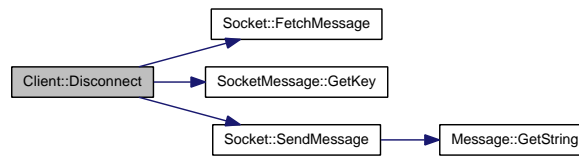
Here is the call graph for this function:



7.8.3.3 `void Client::Disconnect ()`

Unbind this client from the socket.

Here is the call graph for this function:



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

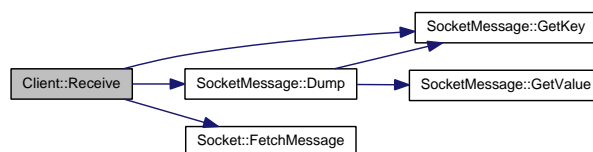
[Socket](#) actor type retrieval method.

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

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

7.8.3.6 SocketMessage Client::Receive (const MessageKey & key)

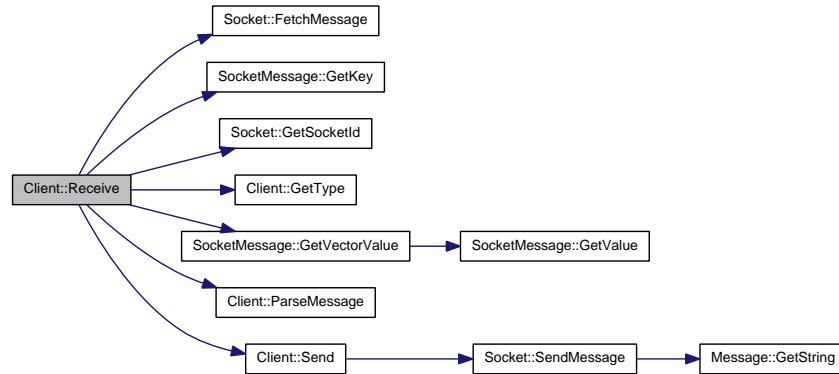
Here is the call graph for this function:



7.8.3.7 void Client::Receive ()

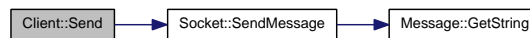
Receive a socket message from the master.

Here is the call graph for this function:



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

Here is the call graph for this function:



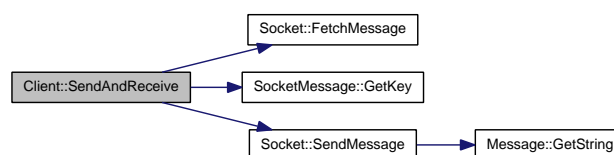
7.8.3.9 `void Client::Send (const Message & m) const [inline]`

Send a message to the master through the socket.

Here is the call graph for this function:

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

Here is the call graph for this function:



7.8.4 Field Documentation

7.8.4.1 `int Client::fClientId` `[private]`

7.8.4.2 `bool Client::fIsConnected` `[private]`

7.8.4.3 `SocketType Client::fType` `[private]`

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

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

7.9 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](#)

7.9.1 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

7.9.2 Field Documentation

7.9.2.1 VME::AcquisitionMode file_header_t::acq_mode

7.9.2.2 VME::DetectionMode file_header_t::det_mode

7.9.2.3 uint32_t file_header_t::magic

7.9.2.4 uint8_t file_header_t::num_hptdc

7.9.2.5 uint32_t file_header_t::run_id

7.9.2.6 uint32_t file_header_t::spill_id

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

- include/FileConstants.h

7.10 FileReader Class Reference

Handler for a TDC output file readout.

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

Public Member Functions

- [FileReader](#) ()
- [FileReader](#) (std::string name)
Class constructor.
- [~FileReader](#) ()
- void [Open](#) (std::string name)
- bool [IsOpen](#) () const
- void [Clear](#) ()
- void [Dump](#) () const
- 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](#)
- time_t [fWriteTime](#)
- unsigned long [fNumEvents](#)

7.10.1 Detailed Description

Handler for a TDC output file readout.

Author:

Laurent Forthomme <laurent.forthomme@cern.ch>

Date:

Jun 2015

7.10.2 Constructor & Destructor Documentation

7.10.2.1 FileReader::FileReader () [inline]

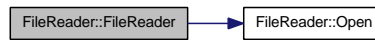
7.10.2.2 FileReader::FileReader (std::string *name*)

Class constructor.

Parameters:

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

Here is the call graph for this function:



7.10.2.3 FileReader::~~FileReader ()

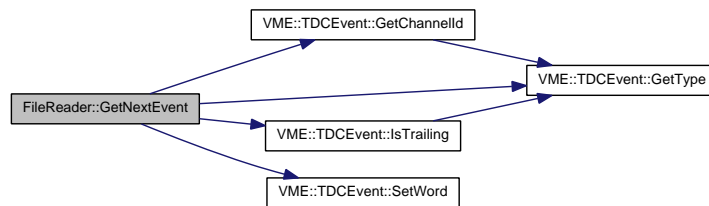
7.10.3 Member Function Documentation

7.10.3.1 void FileReader::Clear () [inline]

7.10.3.2 void FileReader::Dump () const

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

Here is the call graph for this function:



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

Fetch the next full measurement on a given channel.

Parameters:

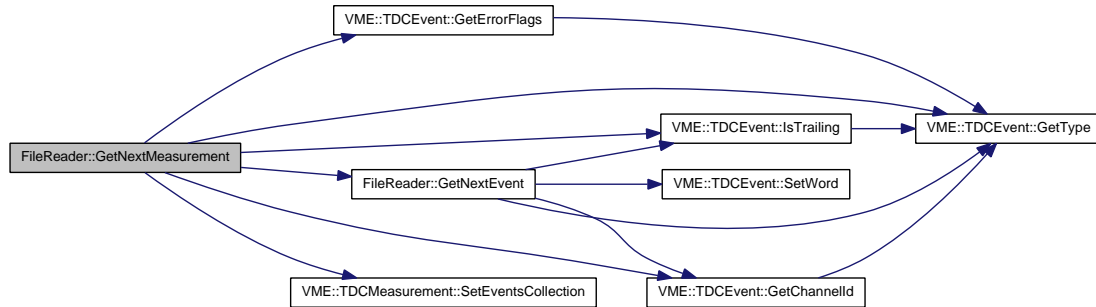
- ← ***channel_id*** Unique identifier of the channel number to retrieve

→ *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:



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

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

7.10.3.7 `bool FileReader::IsOpen () const [inline]`

7.10.3.8 `void FileReader::Open (std::string name)`

7.10.4 Field Documentation

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

7.10.4.2 `file_header_t FileReader::fHeader [private]`

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

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

7.10.4.5 `time_t FileReader::fWriteTime [private]`

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

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

7.11 VME::FPGAUnitV1495 Class Reference

`#include <VME_FPGAUnitV1495.h>` [Inheritance diagram](#) [for VME::FPGAUnitV1495](#): [Collaboration diagram for VME::FPGAUnitV1495](#):

Public Types

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

Public Member Functions

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

- uint32_t [GetInternalTriggerPeriod](#) () const
Retrieve the internal trigger period.
- uint32_t [GetOutputPulser](#) () const
- void [ClearOutputPulser](#) () const
- void [SetOutputPulser](#) (unsigned short id, bool enable=true) const
- void [SetOutputPulserPOI](#) (uint32_t poi) const
- uint32_t [GetOutputDelay](#) () const
- void [SetOutputDelay](#) (uint32_t delay) const
- void [StartScaler](#) ()
Start the inner triggers counter.
- void [StopScaler](#) ()
Stop the inner triggers counter.
- uint32_t [GetScalerValue](#) () const
Return the inner triggers counter value.

Private Attributes

- bool [fScalerStarted](#)

7.11.1 Detailed Description

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

Author:

Laurent Forthomme <laurent.forthomme@cern.ch>

Date:

25 Jun 2015

7.11.2 Member Enumeration Documentation

7.11.2.1 enum VME::FPGAUnitV1495::TDCBits

Enumerator:

kReset

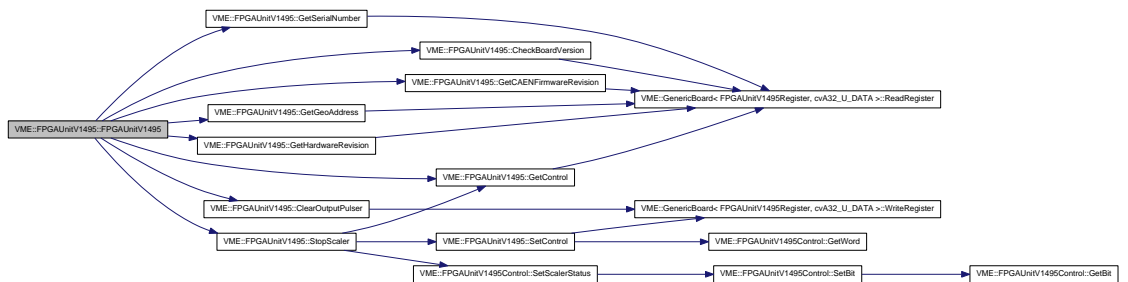
kTrigger

kClear

7.11.3 Constructor & Destructor Documentation

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

Here is the call graph for this function:



7.11.3.2 VME::FPGAUnitV1495::~~FPGAUnitV1495 ()

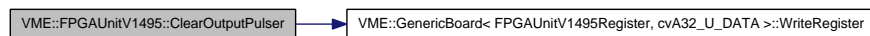
7.11.4 Member Function Documentation

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

Here is the call graph for this function:

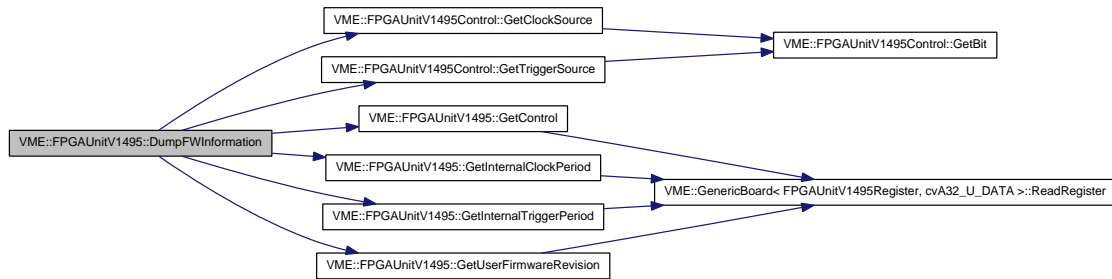
7.11.4.2 void VME::FPGAUnitV1495::ClearOutputPulser () const

Here is the call graph for this function:



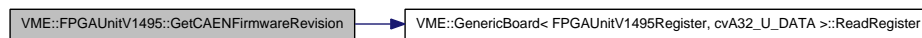
7.11.4.3 void VME::FPGAUnitV1495::DumpFWInformation () const

Here is the call graph for this function:



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

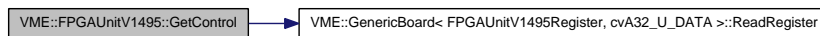
Here is the call graph for this function:



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

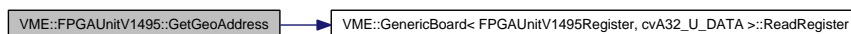
Retrieve the user-defined control word.

Here is the call graph for this function:



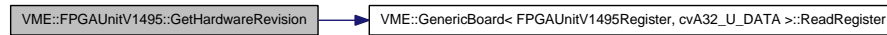
7.11.4.6 unsigned short VME::FPGAUnitV1495::GetGeoAddress () const

Here is the call graph for this function:



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

Here is the call graph for this function:



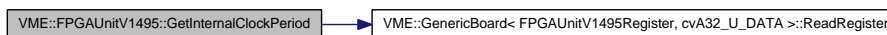
7.11.4.8 uint32_t VME::FPGAUnitV1495::GetInternalClockPeriod () const

Retrieve the internal clock period.

Returns:

Clock period (in units of 25 ns)

Here is the call graph for this function:



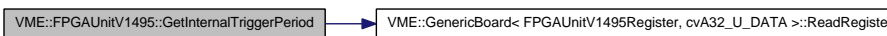
7.11.4.9 uint32_t VME::FPGAUnitV1495::GetInternalTriggerPeriod () const

Retrieve the internal trigger period.

Returns:

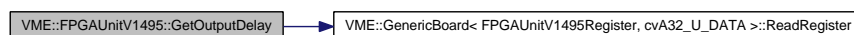
Trigger period (in units of 50 ns)

Here is the call graph for this function:



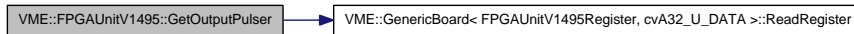
7.11.4.10 uint32_t VME::FPGAUnitV1495::GetOutputDelay () const

Here is the call graph for this function:



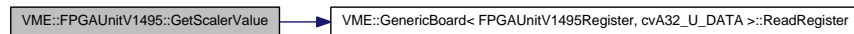
7.11.4.11 uint32_t VME::FPGAUnitV1495::GetOutputPulser () const

Here is the call graph for this function:

**7.11.4.12 uint32_t VME::FPGAUnitV1495::GetScalerValue () const**

Return the inner triggers counter value.

Here is the call graph for this function:

**7.11.4.13 unsigned short VME::FPGAUnitV1495::GetSerialNumber () const**

Here is the call graph for this function:

**7.11.4.14 unsigned short VME::FPGAUnitV1495::GetTDCBits () const**

Retrieve the current bits sent to TDCs' front panel.

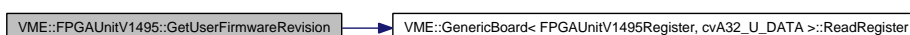
Returns:

A 3-bit word PoI

Here is the call graph for this function:

**7.11.4.15 unsigned short VME::FPGAUnitV1495::GetUserFirmwareRevision () const**

Here is the call graph for this function:



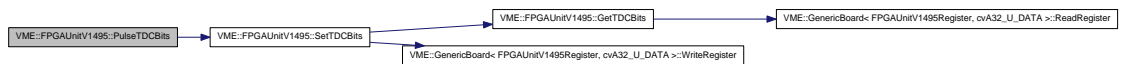
7.11.4.16 void VME::FPGAUnitV1495::PulseTDCBits (unsigned short *bits*, unsigned int *time_us* = 10) const

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:



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

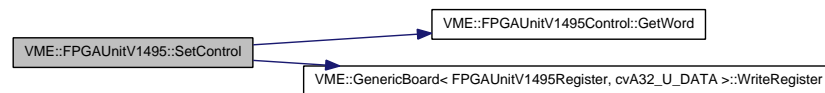
Here is the call graph for this function:



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

Set the user-defined control word.

Here is the call graph for this function:



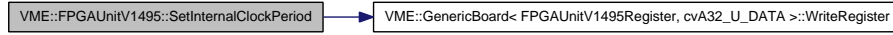
7.11.4.19 void VME::FPGAUnitV1495::SetInternalClockPeriod (uint32_t *period*) const

Set the internal clock period.

Parameters:

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

Here is the call graph for this function:



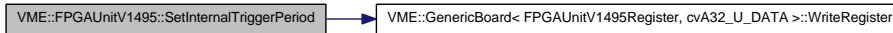
7.11.4.20 void VME::FPGAUnitV1495::SetInternalTriggerPeriod (uint32_t *period*) const

Set the internal trigger period.

Parameters:

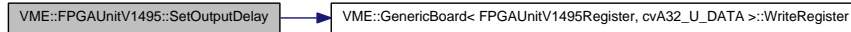
← *period* Trigger period (in units of 50 ns)

Here is the call graph for this function:



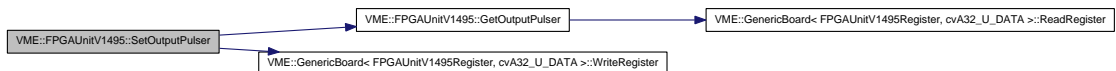
7.11.4.21 void VME::FPGAUnitV1495::SetOutputDelay (uint32_t *delay*) const

Here is the call graph for this function:



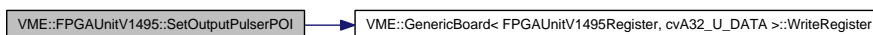
7.11.4.22 void VME::FPGAUnitV1495::SetOutputPulser (unsigned short *id*, bool *enable = true*) const

Here is the call graph for this function:



7.11.4.23 void VME::FPGAUnitV1495::SetOutputPulserPOI (uint32_t *poi*) const

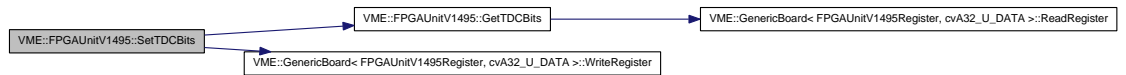
Here is the call graph for this function:



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

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

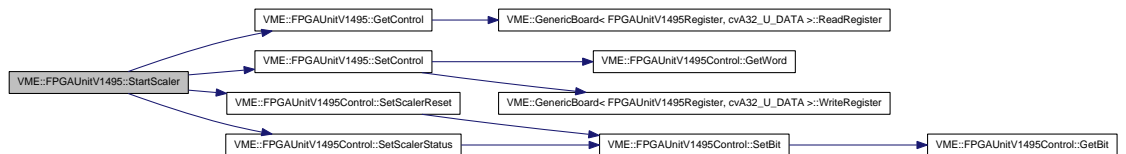
Here is the call graph for this function:



7.11.4.25 void VME::FPGAUnitV1495::StartScaler ()

Start the inner triggers counter.

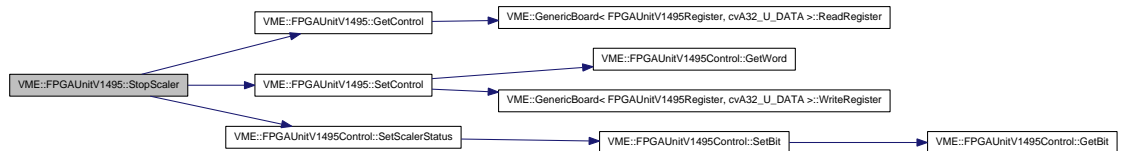
Here is the call graph for this function:



7.11.4.26 void VME::FPGAUnitV1495::StopScaler ()

Stop the inner triggers counter.

Here is the call graph for this function:



7.11.5 Field Documentation

7.11.5.1 bool VME::FPGAUnitV1495::fScalerStarted [private]

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

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

7.12 VME::FPGAUnitV1495Control Class Reference

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

Public Types

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

Public Member Functions

- [FPGAUnitV1495Control](#) (uint32_t word)
- virtual [~FPGAUnitV1495Control](#) ()
- void [Dump](#) () const
- uint32_t [GetWord](#) () const
- [ClockSource](#) [GetClockSource](#) () const
Get the clock source.
- void [SetClockSource](#) (const [ClockSource](#) &cs)
Switch between internal and external clock source.
- [TriggerSource](#) [GetTriggerSource](#) () const
Get the trigger source.
- void [SetTriggerSource](#) (const [TriggerSource](#) &cs)
Switch between internal and external trigger source.
- bool [GetScalerStatus](#) () const
- void [SetScalerStatus](#) (bool start=true)
- void [SetScalerReset](#) (bool reset=true)
- [SignalSource](#) [GetSignalSource](#) (unsigned short map_id) const
- void [SetSignalSource](#) (unsigned short map_id, const [SignalSource](#) &s)

Private Member Functions

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

Private Attributes

- uint32_t [fWord](#)

7.12.1 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

7.12.2 Member Enumeration Documentation

7.12.2.1 enum VME::FPGAUnitV1495Control::ClockSource

Enumerator:

InternalClock

ExternalClock

7.12.2.2 enum VME::FPGAUnitV1495Control::SignalSource

Enumerator:

InternalSignal

ExternalSignal

7.12.2.3 enum VME::FPGAUnitV1495Control::TriggerSource

Enumerator:

InternalTrigger

ExternalTrigger

7.12.3 Constructor & Destructor Documentation

7.12.3.1 VME::FPGAUnitV1495Control::FPGAUnitV1495Control (uint32_t *word*) [inline]

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

7.12.4 Member Function Documentation

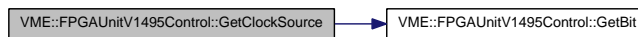
7.12.4.1 void VME::FPGAUnitV1495Control::Dump () const [inline]

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

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

Get the clock source.

Here is the call graph for this function:



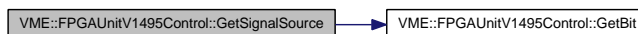
7.12.4.4 bool VME::FPGAUnitV1495Control::GetScalerStatus () const [inline]

Here is the call graph for this function:



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

Here is the call graph for this function:



7.12.4.6 **TriggerSource** VME::FPGAUnitV1495Control::GetTriggerSource () const [inline]

Get the trigger source.

Here is the call graph for this function:



7.12.4.7 **uint32_t** VME::FPGAUnitV1495Control::GetWord () const [inline]

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

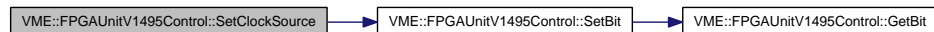
Here is the call graph for this function:



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

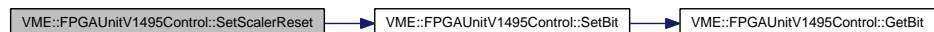
Switch between internal and external clock source.

Here is the call graph for this function:



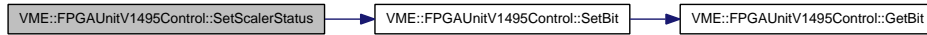
7.12.4.10 **void** VME::FPGAUnitV1495Control::SetScalerReset (bool *reset* = true) [inline]

Here is the call graph for this function:



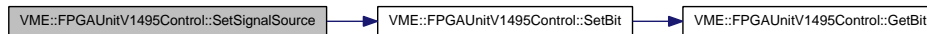
7.12.4.11 void VME::FPGAUnitV1495Control::SetScalerStatus (bool *start* = true) [inline]

Here is the call graph for this function:



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

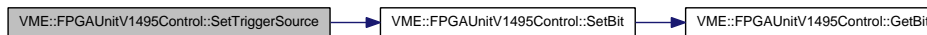
Here is the call graph for this function:



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

Switch between internal and external trigger source.

Here is the call graph for this function:



7.12.5 Field Documentation

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

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

- include/VME_FPGAUnitV1495.h

7.13 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 >
```

7.13.1 Constructor & Destructor Documentation

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

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

7.13.2 Member Function Documentation

7.13.2.1 `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

7.13.2.2 `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

7.13.2.3 `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

7.13.2.4 `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

7.13.3 Field Documentation

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

7.13.3.2 `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

7.15 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](#)

7.15.1 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

7.15.2 Constructor & Destructor Documentation

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

Here is the call graph for this function:

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

Here is the call graph for this function:

7.15.3 Member Function Documentation

7.15.3.1 void HTTPMessage::Decode () [inline]

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

Reimplemented from [Message](#).

7.15.3.3 void HTTPMessage::Encode () [inline]

7.15.3.4 MessageKey HTTPMessage::GetKey () const [inline]

Placeholder for the MessageKey retrieval method.

Reimplemented from [Message](#).

7.15.4 Field Documentation

7.15.4.1 std::string HTTPMessage::fOriginalString [private]

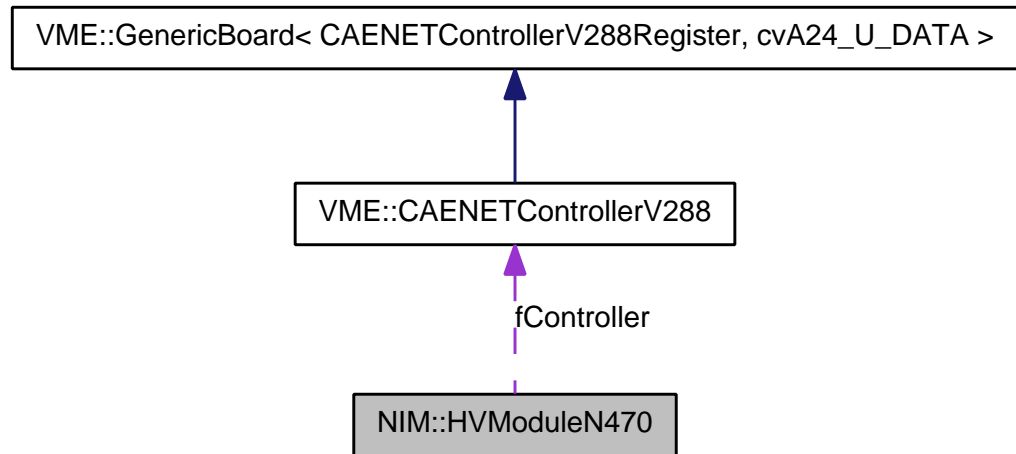
7.15.4.2 WebSocket* HTTPMessage::fWS [private]

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

- include/HTTPMessage.h

7.16 NIM::HVModuleN470 Class Reference

#include <NIM_HVModuleN470.h> Collaboration diagram for NIM::HVModuleN470:



Public Member Functions

- `HVModuleN470` (uint16_t addr, [VME::CAENETControllerV288](#) &cont)
- `~HVModuleN470` ()
- std::string [GetModuleId](#) () const
- unsigned short [GetFWRevision](#) () const
- [HVModuleN470Values](#) [ReadMonitoringValues](#) () const
- [HVModuleN470ChannelValues](#) [ReadChannelValues](#) (unsigned short ch_id) const
- void [SetChannelV0](#) (unsigned short ch_id, unsigned short v0) const
- void [SetChannelI0](#) (unsigned short ch_id, unsigned short i0) const
- void [SetChannelV1](#) (unsigned short ch_id, unsigned short v1) const
- void [SetChannelI1](#) (unsigned short ch_id, unsigned short i1) const

Private Member Functions

- void [ReadRegister](#) (const [HVModuleN470Opcodes](#) ®, std::vector< uint16_t > *data, unsigned int num_words=1) const
Read in register.
- void [WriteRegister](#) (const [HVModuleN470Opcodes](#) ®, const std::vector< uint16_t > &data) const
Write on register.

- void [WriteRegister](#) (const [HVModuleN470Opcodes](#) ®, const uint16_t &data)
const

Write on register.

Private Attributes

- [VME::CAENETControllerV288](#) fController
- uint16_t fAddress

7.16.1 Constructor & Destructor Documentation

7.16.1.1 [NIM::HVModuleN470::HVModuleN470](#) (uint16_t *addr*,
[VME::CAENETControllerV288](#) & *cont*)

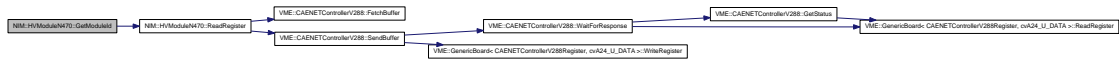
7.16.1.2 [NIM::HVModuleN470::~~HVModuleN470](#) () [[inline](#)]

7.16.2 Member Function Documentation

7.16.2.1 unsigned short [NIM::HVModuleN470::GetFWRevision](#) () const

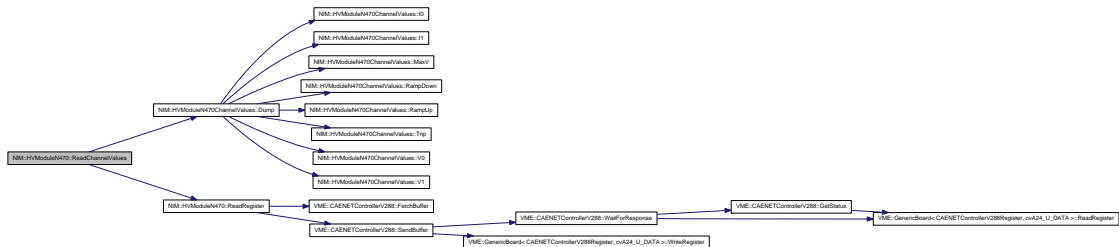
7.16.2.2 std::string [NIM::HVModuleN470::GetModuleId](#) () const

Here is the call graph for this function:



7.16.2.3 [HVModuleN470ChannelValues](#)
[NIM::HVModuleN470::ReadChannelValues](#)
(unsigned short *ch_id*) const

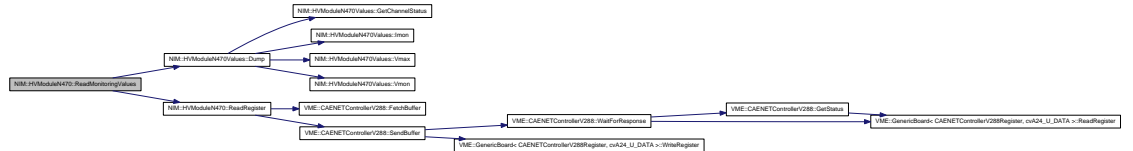
Here is the call graph for this function:



7.16.2.4 HVModuleN470Values

NIM::HVModuleN470::ReadMonitoringValues ()
const

Here is the call graph for this function:



7.16.2.5 void NIM::HVModuleN470::ReadRegister (const HVModuleN470Opcodes & reg, std::vector< uint16_t > * data, unsigned int num_words = 1) const [private]

Read in register. Read a vector of 16-bit words in the register

Parameters:

← *addr* register

→ *vector* of data words

Here is the call graph for this function:



7.16.2.6 void NIM::HVModuleN470::SetChannelI0 (unsigned short ch_id, unsigned short i0) const

Here is the call graph for this function:



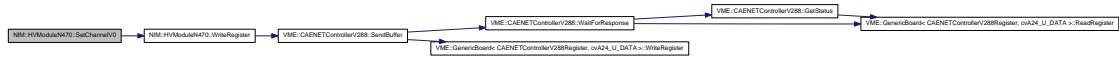
7.16.2.7 void NIM::HVModuleN470::SetChannelI1 (unsigned short *ch_id*, unsigned short *iI*) const

Here is the call graph for this function:



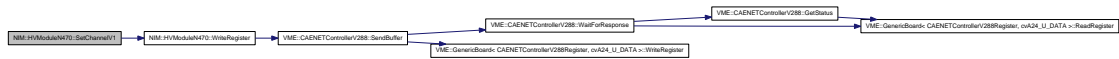
7.16.2.8 void NIM::HVModuleN470::SetChannelV0 (unsigned short *ch_id*, unsigned short *v0*) const

Here is the call graph for this function:



7.16.2.9 void NIM::HVModuleN470::SetChannelV1 (unsigned short *ch_id*, unsigned short *vI*) const

Here is the call graph for this function:



7.16.2.10 void NIM::HVModuleN470::WriteRegister (const HVModuleN470Opcodes & *reg*, const uint16_t & *data*) const [private]

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

Parameters:

← *addr* register

→ *data* word

Here is the call graph for this function:



7.16.2.11 `void NIM::HVModuleN470::WriteRegister (const HVModuleN470Opcodes & reg, const std::vector< uint16_t > & data) const [private]`

Write on register. Write a vector of 16-bit words in the register

Parameters:

← *addr* register

→ *data* word

Here is the call graph for this function:



7.16.3 Field Documentation

7.16.3.1 `uint16_t NIM::HVModuleN470::fAddress [private]`

7.16.3.2 `VME::CAENETControllerV288 NIM::HVModuleN470::fController [private]`

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

- include/NIM_HVModuleN470.h
- src/NIM_HVModuleN470.cpp

7.17 NIM::HVModuleN470ChannelValues Class Reference

Single channel monitoring values for the HV power supply.

```
#include <NIM_HVModuleN470.h>
```

Public Member Functions

- [HVModuleN470ChannelValues](#) (unsigned short ch_id, std::vector< unsigned short > vals)
- [~HVModuleN470ChannelValues](#) ()
- void [Dump](#) () const
- unsigned short [ChannelStatus](#) () const
- unsigned short [Vmon](#) () const
- unsigned short [Imon](#) () const
- unsigned short [V0](#) () const
- unsigned short [I0](#) () const
- unsigned short [V1](#) () const
- unsigned short [I1](#) () const
- unsigned short [Trip](#) () const
- unsigned short [RampUp](#) () const
- unsigned short [RampDown](#) () const
- unsigned short [MaxV](#) () const

Private Attributes

- unsigned short [fChannelId](#)
- std::vector< unsigned short > [fValues](#)

7.17.1 Detailed Description

Single channel monitoring values for the HV power supply.

Author:

Laurent Forthomme <laurent.forthomme@cern.ch>

Date:

24 Jul 2015

7.17.2 Constructor & Destructor Documentation

7.17.2.1 `NIM::HVModuleN470ChannelValues::HVModuleN470ChannelValues`
 (unsigned short *ch_id*, std::vector< unsigned short > *vals*)
 [inline]

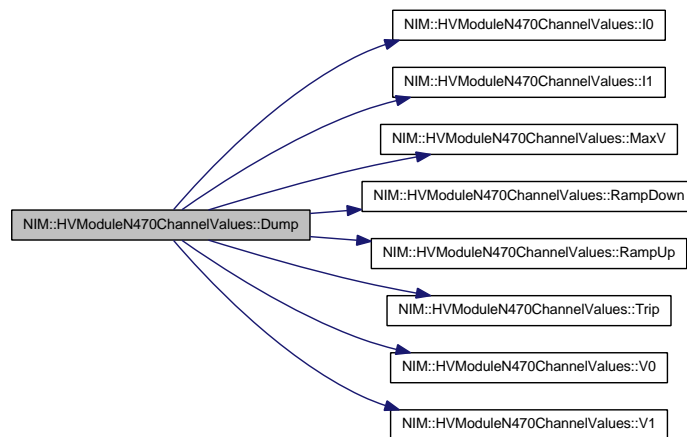
7.17.2.2 `NIM::HVModuleN470ChannelValues::~~HVModuleN470ChannelValues`
 () [inline]

7.17.3 Member Function Documentation

7.17.3.1 `unsigned short NIM::HVModuleN470ChannelValues::ChannelStatus`
 () const [inline]

7.17.3.2 `void NIM::HVModuleN470ChannelValues::Dump () const`
 [inline]

Here is the call graph for this function:



- 7.17.3.3 unsigned short NIM::HVModuleN470ChannelValues::I0 () const
[inline]
- 7.17.3.4 unsigned short NIM::HVModuleN470ChannelValues::I1 () const
[inline]
- 7.17.3.5 unsigned short NIM::HVModuleN470ChannelValues::Imon () const
[inline]
- 7.17.3.6 unsigned short NIM::HVModuleN470ChannelValues::MaxV () const
[inline]
- 7.17.3.7 unsigned short NIM::HVModuleN470ChannelValues::RampDown ()
const [inline]
- 7.17.3.8 unsigned short NIM::HVModuleN470ChannelValues::RampUp ()
const [inline]
- 7.17.3.9 unsigned short NIM::HVModuleN470ChannelValues::Trip () const
[inline]
- 7.17.3.10 unsigned short NIM::HVModuleN470ChannelValues::V0 () const
[inline]
- 7.17.3.11 unsigned short NIM::HVModuleN470ChannelValues::V1 () const
[inline]
- 7.17.3.12 unsigned short NIM::HVModuleN470ChannelValues::Vmon () const
[inline]

7.17.4 Field Documentation

- 7.17.4.1 unsigned short NIM::HVModuleN470ChannelValues::fChannelId
[private]
- 7.17.4.2 std::vector<unsigned short>
NIM::HVModuleN470ChannelValues::fValues
[private]

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

- include/NIM_HVModuleN470.h

7.18 NIM::HVModuleN470Values Class Reference

General monitoring values for the HV power supply.

```
#include <NIM_HVModuleN470.h>
```

Data Structures

- class [ChannelStatus](#)

Public Member Functions

- [HVModuleN470Values](#) (std::vector< unsigned short > vals)
- [~HVModuleN470Values](#) ()
- void [Dump](#) () const
- unsigned short [Vmon](#) () const
- unsigned short [Imon](#) () const
- unsigned short [Vmax](#) () const
- [ChannelStatus](#) [GetChannelStatus](#) (unsigned short ch_id) const

Private Attributes

- std::vector< unsigned short > [fValues](#)

7.18.1 Detailed Description

General monitoring values for the HV power supply.

Author:

Laurent Forthomme <laurent.forthomme@cern.ch>

Date:

24 Jul 2015

7.18.2 Constructor & Destructor Documentation

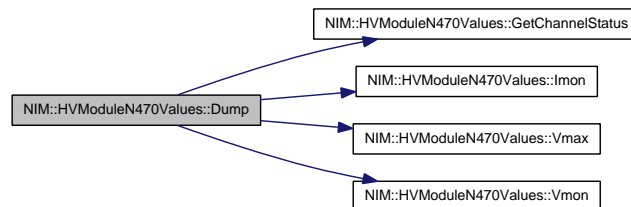
7.18.2.1 `NIM::HVModuleN470Values::HVModuleN470Values (std::vector< unsigned short > vals) [inline]`

7.18.2.2 `NIM::HVModuleN470Values::~~HVModuleN470Values () [inline]`

7.18.3 Member Function Documentation

7.18.3.1 `void NIM::HVModuleN470Values::Dump () const [inline]`

Here is the call graph for this function:



7.18.3.2 `ChannelStatus NIM::HVModuleN470Values::GetChannelStatus (unsigned short ch_id) const [inline]`

7.18.3.3 `unsigned short NIM::HVModuleN470Values::Imon () const [inline]`

7.18.3.4 `unsigned short NIM::HVModuleN470Values::Vmax () const [inline]`

7.18.3.5 `unsigned short NIM::HVModuleN470Values::Vmon () const [inline]`

7.18.4 Field Documentation

7.18.4.1 `std::vector<unsigned short> NIM::HVModuleN470Values::fValues [private]`

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

- `include/NIM_HVModuleN470.h`

7.19 VME::IOModuleV262 Class Reference

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

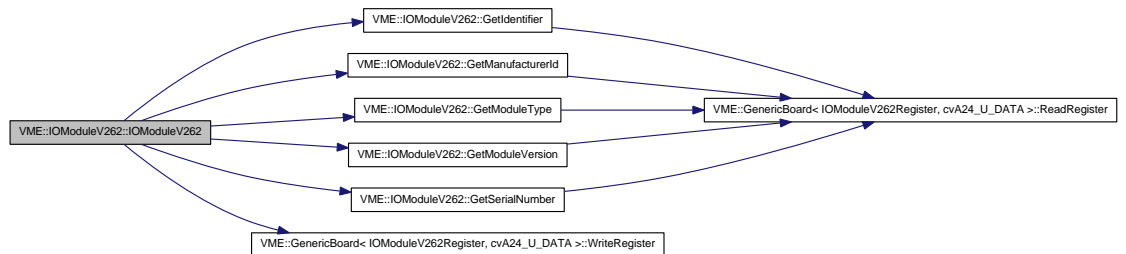
Public Member Functions

- [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

7.19.1 Constructor & Destructor Documentation

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

Here is the call graph for this function:

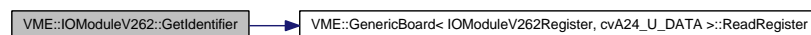


7.19.1.2 VME::IOModuleV262::~~IOModuleV262 () [inline]

7.19.2 Member Function Documentation

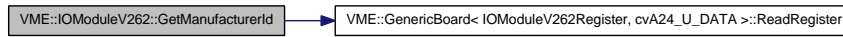
7.19.2.1 unsigned short VME::IOModuleV262::GetIdentifier () const

Here is the call graph for this function:



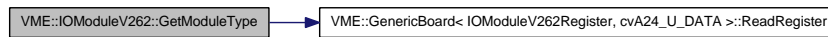
7.19.2.2 unsigned short VME::IOModuleV262::GetManufacturerId () const

Here is the call graph for this function:



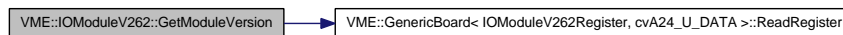
7.19.2.3 unsigned short VME::IOModuleV262::GetModuleType () const

Here is the call graph for this function:



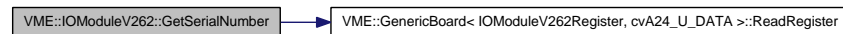
7.19.2.4 unsigned short VME::IOModuleV262::GetModuleVersion () const

Here is the call graph for this function:



7.19.2.5 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

7.20 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](#)

7.20.1 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

7.20.2 Constructor & Destructor Documentation

7.20.2.1 Message::Message () [inline]

Void message constructor.

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

Construct a message from a string.

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

Construct a message from a string.

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

7.20.3 Member Function Documentation

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

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

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

Placeholder for the MessageKey retrieval method.

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

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

Retrieve the string carried by this message as a whole.

Reimplemented in [SocketMessage](#).

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

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

7.20.4 Field Documentation

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

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

- `include/Message.h`

7.21 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 [SendAll](#) (const [Socket::SocketType](#) &type, const [Message](#) &m) const
Send any type of message to all clients of one type.
- 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]

7.21.1 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

7.21.2 Constructor & Destructor Documentation

7.21.2.1 [Messenger::Messenger \(\)](#)

Build a void master object or socket actor.

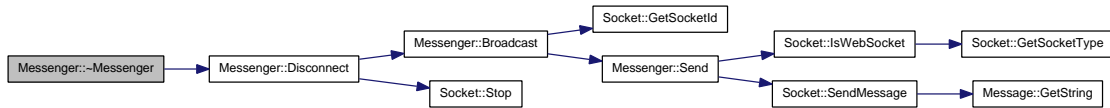
7.21.2.2 [Messenger::Messenger \(int port\)](#)

Build a master object to control the socket.

Here is the call graph for this function:

7.21.2.3 Messenger::~~Messenger ()

Here is the call graph for this function:

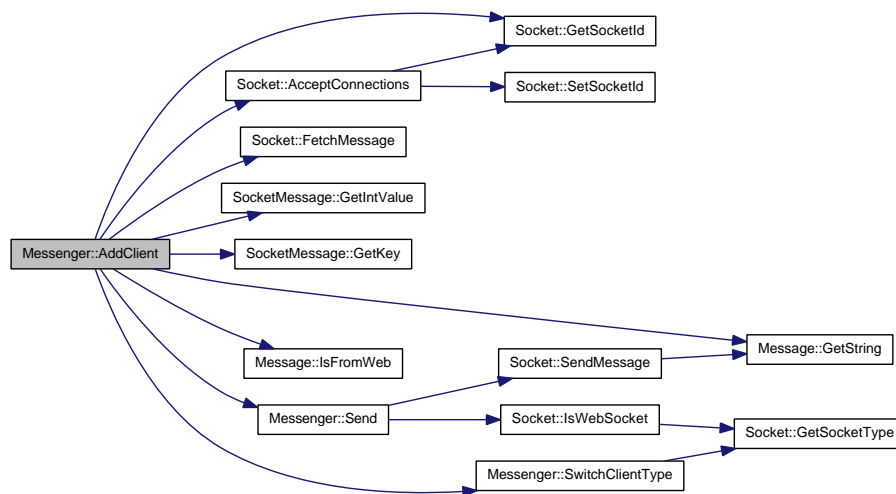


7.21.3 Member Function Documentation

7.21.3.1 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:



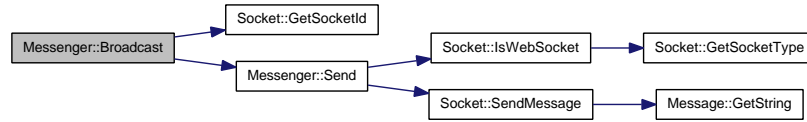
7.21.3.2 void Messenger::Broadcast (const Message & m) const

Emit a message to all clients connected through the socket.

Parameters:

← *m* **Message** to transmit

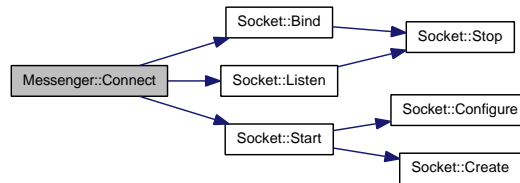
Here is the call graph for this function:



7.21.3.3 bool Messenger::Connect ()

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

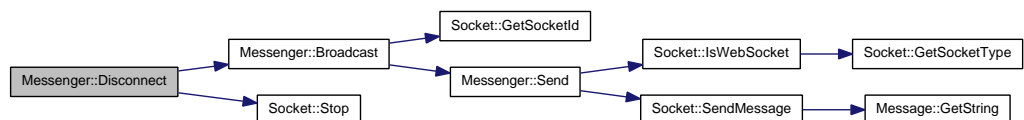
Here is the call graph for this function:



7.21.3.4 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:



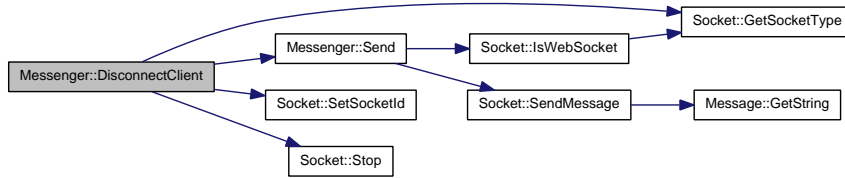
7.21.3.5 void Messenger::DisconnectClient (int sid, MessageKey key, bool force = false) [private]

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

Parameters:

- ← *sid* Unique identifier of the client to disconnect
- ← *key* Key to the message to transmit for disconnection
- ← *force* Do we need to force the client out of this socket ?

Here is the call graph for this function:



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

Socket actor type retrieval method.

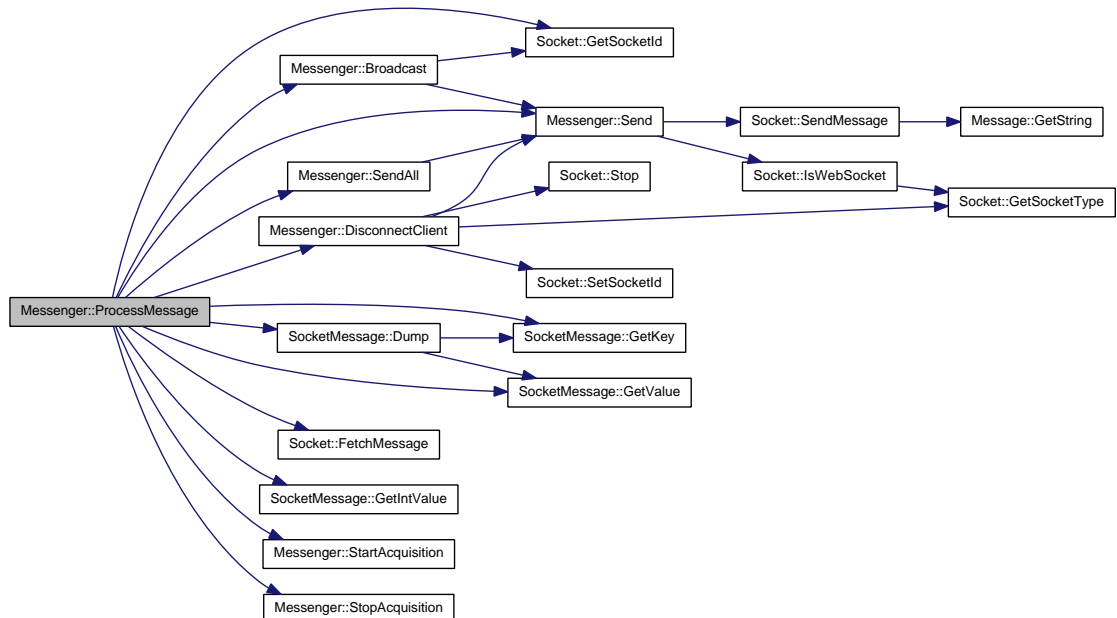
7.21.3.7 void Messenger::ProcessMessage (SocketMessage m, int sid) [private]

Process a message received from the socket.

Parameters:

← *Unique* identifier of the client sending the message

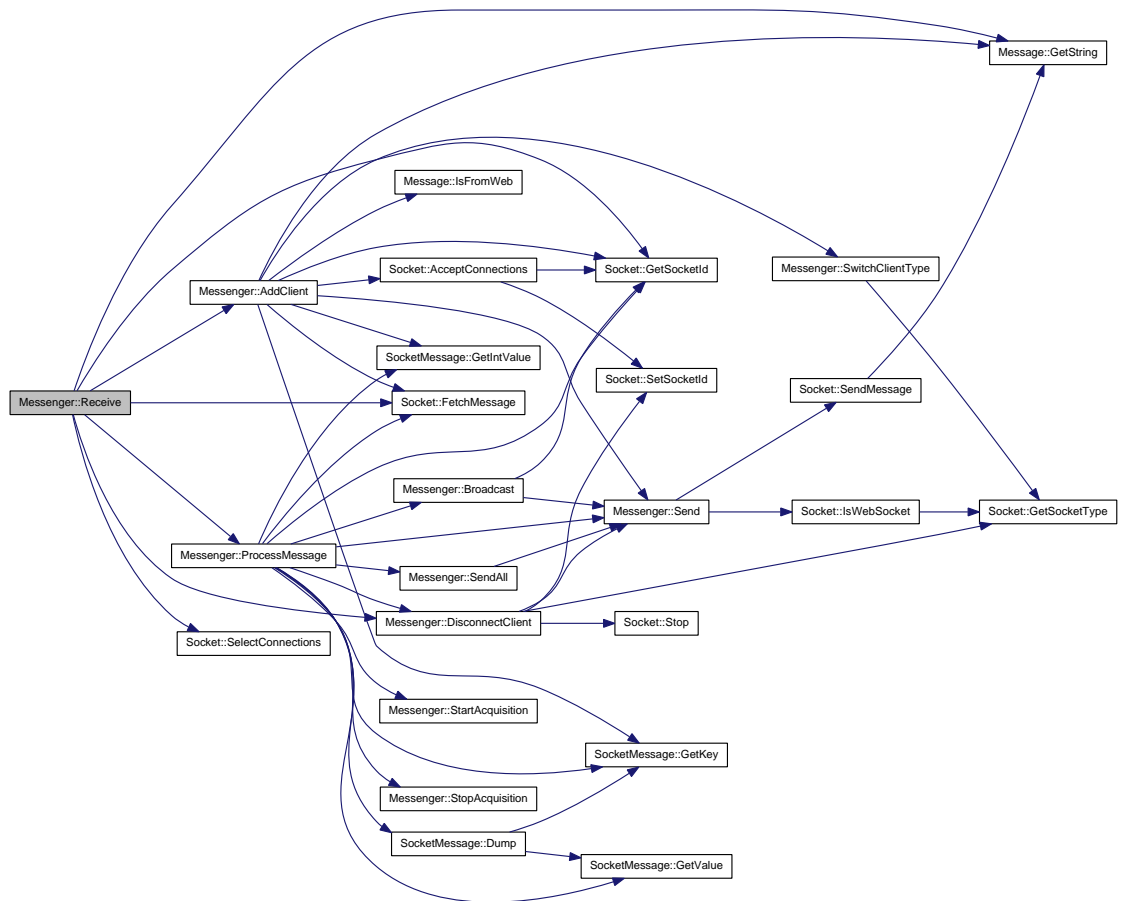
Here is the call graph for this function:



7.21.3.8 void Messenger::Receive ()

Handle a message reception from a client.

Here is the call graph for this function:



7.21.3.9 void Messenger::Send (const Message & m, int sid) const

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:



7.21.3.10 void Messenger::SendAll (const Socket::SocketType & type, const Message & m) const [inline]

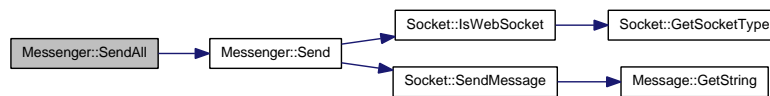
Send any type of message to all clients of one type.

Parameters:

← *type* Client type

← *m* Message to transmit

Here is the call graph for this function:



7.21.3.11 void Messenger::StartAcquisition ()

Start the data acquisition.

7.21.3.12 void Messenger::StopAcquisition ()

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

Here is the call graph for this function:



7.21.4 Field Documentation

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

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

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

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

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

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

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

7.22 VME::PCIInterfaceA2818 Class Reference

```
#include <VME_PCIInterfaceA2818.h>
```

Public Member Functions

- [PCIInterfaceA2818](#) (const char *device)
- virtual [~PCIInterfaceA2818](#) ()
- std::string [GetFWRevision](#) () const

Private Attributes

- int [fHandle](#)

7.22.1 Constructor & Destructor Documentation

7.22.1.1 VME::PCIInterfaceA2818::PCIInterfaceA2818 (const char * *device*)
[[inline](#)]

7.22.1.2 virtual VME::PCIInterfaceA2818::~~PCIInterfaceA2818 ()
[[inline](#), [virtual](#)]

7.22.2 Member Function Documentation

7.22.2.1 std::string VME::PCIInterfaceA2818::GetFWRevision () const
[[inline](#)]

7.22.3 Field Documentation

7.22.3.1 int VME::PCIInterfaceA2818::fHandle [[private](#)]

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

- include/VME_PCIInterfaceA2818.h

7.23 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](#), [DQM](#) }
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](#)

7.23.1 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

7.23.2 Member Typedef Documentation

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

7.23.3 Member Enumeration Documentation

7.23.3.1 `enum Socket::SocketType`

Type of actor playing a role on the socket.

Enumerator:

INVALID

MASTER

WEBSOCKET_CLIENT

CLIENT

DETECTOR

DQM

7.23.4 Constructor & Destructor Documentation

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

7.23.4.2 `Socket::Socket (int port)`

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

7.23.5 Member Function Documentation

7.23.5.1 `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:

7.23.5.2 void Socket::Bind () [protected]

Bind a name to a socket.

Returns:

Success of the operation

Here is the call graph for this function:

7.23.5.3 void Socket::Configure () [private]

Configure the socket object for communication.

7.23.5.4 void Socket::Create () [private]

Create an endpoint for communication.

7.23.5.5 void Socket::DumpConnected () const**7.23.5.6 Message Socket::FetchMessage (int *id* = -1) const [protected]**

Receive a message from a socket.

Returns:

Received message as a std::string

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

Retrieve the port used for this socket.

7.23.5.8 int Socket::GetSocketId () const [inline]**7.23.5.9 SocketType Socket::GetSocketType (int *sid*) const [inline]****7.23.5.10 bool Socket::IsWebSocket (int *sid*) const [inline]**

Here is the call graph for this function:

7.23.5.11 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:

7.23.5.12 void Socket::PrepareConnection () [protected]

Here is the call graph for this function:

7.23.5.13 void Socket::SelectConnections ()

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

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

Send a message on a socket.

Here is the call graph for this function:

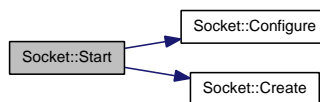
7.23.5.15 void Socket::SetPort (int *port*) [inline]**7.23.5.16 void Socket::SetSocketId (int *sid*) [inline]****7.23.5.17 bool Socket::Start () [protected]**

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:

**7.23.5.18 void Socket::Stop ()**

Terminates the socket and all attached communications.

7.23.6 Field Documentation

7.23.6.1 struct sockaddr_in Socket::fAddress [read, private]

7.23.6.2 char Socket::fBuffer[MAX_WORD_LENGTH] [protected]

7.23.6.3 fd_set Socket::fMaster [protected]

Master file descriptor list.

7.23.6.4 int Socket::fPort [protected]

7.23.6.5 fd_set Socket::fReadFds [protected]

Temp file descriptor list for select().

7.23.6.6 int Socket::fSocketId [private]

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

7.23.6.7 SocketCollection Socket::fSocketsConnected [protected]

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

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

7.24 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.
- std::string [GetCleanedValue](#) () const
Extract the message's string value (without the trailing endlines).
- 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](#)

7.24.1 Detailed Description

Socket-passed message type.

Author:

Laurent Forthomme <laurent.forthomme@cern.ch>

Date:

26 Mar 2015

7.24.2 Constructor & Destructor Documentation

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

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

Here is the call graph for this function:

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

Here is the call graph for this function:

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

Here is the call graph for this function:

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

Construct a socket message out of a key.

Here is the call graph for this function:

7.24.2.6 `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:

7.24.2.7 `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:

7.24.2.8 `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:

7.24.2.9 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:

7.24.2.10 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:

7.24.2.11 SocketMessage::SocketMessage (MessageMap msg_m) [inline]

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

7.24.2.12 SocketMessage::~~SocketMessage () [inline]**7.24.3 Member Function Documentation****7.24.3.1 void SocketMessage::Dump (std::ostream & os = std::cout) const [inline]**

Reimplemented from [Message](#).

Here is the call graph for this function:

7.24.3.2 std::string SocketMessage::GetCleanedValue () const [inline]

Extract the message's string value (without the trailing endlines).

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

Extract the message's integer value.

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

Extract the message's key.

Reimplemented from [Message](#).

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

Extract the whole key:value message.

Reimplemented from [Message](#).

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

Extract the message's string value.

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

Extract the message's vector of string value.

Here is the call graph for this function:

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

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

Double-valued message.

Here is the call graph for this function:

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

Float-valued message.

Here is the call graph for this function:

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

Send an integer-valued message.

Here is the call graph for this function:

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

String-valued message.

Here is the call graph for this function:

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

7.24.4 Field Documentation

7.24.4.1 `MessageMap SocketMessage::fMessage` `[private]`

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

- `include/SocketMessage.h`

7.25 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)

7.25.1 Detailed Description

Error flags handler.

Author:

Laurent Forthomme <laurent.forthomme@cern.ch>

Date:

22 Jun 2015

7.25.2 Constructor & Destructor Documentation**7.25.2.1** VME::TDCErrrorFlag::TDCErrrorFlag (uint16_t ef) [inline]**7.25.2.2** virtual VME::TDCErrrorFlag::~~TDCErrrorFlag () [inline, virtual]**7.25.3 Member Function Documentation****7.25.3.1** void VME::TDCErrrorFlag::Dump () const [inline]**7.25.3.2** uint16_t VME::TDCErrrorFlag::GetWord () const [inline]**7.25.3.3** bool VME::TDCErrrorFlag::HasGroupError (unsigned int group_id) const [inline]

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

7.25.3.4 bool VME::TDCErrrorFlag::HasInternalChipError () const [inline]

Internal fatal chip error has been detected.

7.25.3.5 bool VME::TDCErrrorFlag::HasL1BufferOverflow (unsigned int group_id) const [inline]

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

7.25.3.6 bool VME::TDCErrrorFlag::HasReachedEventSizeLimit () const [inline]

Hits rejected because of programmed event size limit.

7.25.3.7 bool VME::TDCErrrorFlag::HasReadoutFIFOOverflow (unsigned int group_id) const [inline]

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

7.25.3.8 bool VME::TDCErrrorFlag::HasTriggerFIFOOverflow () const [inline]

Event lost (trigger FIFO overflow).

7.25.4 Friends And Related Function Documentation

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

7.25.5 Field Documentation

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

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

- `include/VME_TDCEvent.h`

7.26 VME::TDCEvent Class Reference

HPTDC event parser.

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

Public Types

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

Public Member Functions

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

Are we dealing with a trailing or a leading measurement?

- uint32_t [GetETTT](#) () const
Extended trigger time tag.
- uint32_t [GetTime](#) (bool pair=false) const
Edge measurement in programmed time resolution.
- unsigned int [GetWidth](#) () const
Width of pulse in programmed time resolution.
- unsigned int [GetStatus](#) () const
- [TDCErrorFlag](#) [GetErrorFlags](#) () const
Return error flags if an error condition has been detected.

Private Attributes

- uint32_t [fWord](#)

7.26.1 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

7.26.2 Member Enumeration Documentation

7.26.2.1 enum VME::TDCEvent::EventType

Enumerator:

TDCMeasurement
TDCHeader
TDCTrailer
TDCError
GlobalHeader
GlobalTrailer
ETTT
Filler

7.26.3 Constructor & Destructor Documentation

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

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

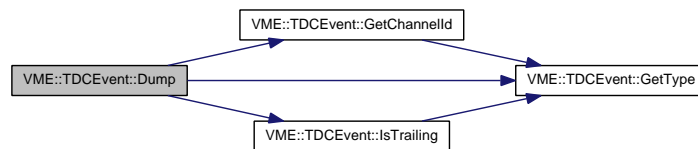
7.26.3.3 VME::TDCEvent::TDCEvent (const uint32_t & *word*) [inline]

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

7.26.4 Member Function Documentation

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

Here is the call graph for this function:



7.26.4.2 uint16_t VME::TDCEvent::GetBunchId () const [inline]

Bunch identifier of trigger (or trigger time tag).

Here is the call graph for this function:

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

Channel number for.

Here is the call graph for this function:

7.26.4.4 TDCErrorFlag VME::TDCEvent::GetErrorFlags () const [inline]

Return error flags if an error condition has been detected.

Here is the call graph for this function:

7.26.4.5 uint32_t VME::TDCEvent::GetETTT () const [inline]

Extended trigger time tag.

Here is the call graph for this function:

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

Total number of events.

Here is the call graph for this function:

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

Event identifier from event counter.

Here is the call graph for this function:

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

Here is the call graph for this function:

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

Here is the call graph for this function:

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

Programmed identifier of master TDC providing the event.

Here is the call graph for this function:

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

Edge measurement in programmed time resolution.

Parameters:

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

Here is the call graph for this function:

**7.26.4.12 EventType VME::TDCEvent::GetType () const [inline]**

Type of packet read out from the TDC.

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

Width of pulse in programmed time resolution.

Here is the call graph for this function:

7.26.4.14 uint32_t VME::TDCEvent::GetWord () const [inline]**7.26.4.15 uint16_t VME::TDCEvent::GetWordCount () const [inline]**

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

Here is the call graph for this function:

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

Are we dealing with a trailing or a leading measurement?

Here is the call graph for this function:

**7.26.4.17 void VME::TDCEvent::SetWord (const uint32_t & word)
[inline]****7.26.5 Field Documentation****7.26.5.1 uint32_t VME::TDCEvent::fWord [private]**

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

- include/VME_TDCEvent.h

7.27 VME::TDCMeasurement Class Reference

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

Public Member Functions

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

Private Attributes

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

7.27.1 Detailed Description

Author:

Laurent Forthomme <laurent.forthomme@cern.ch>

Date:

Jun 2015

7.27.2 Constructor & Destructor Documentation

7.27.2.1 VME::TDCMeasurement::TDCMeasurement () [inline]

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

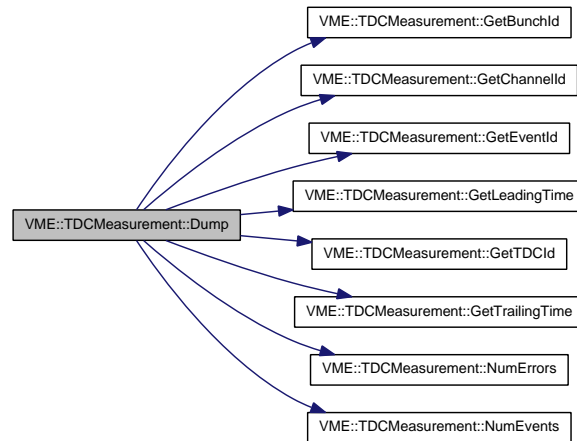
Here is the call graph for this function:

7.27.2.3 VME::TDCMeasurement::~~TDCMeasurement () [inline]

7.27.3 Member Function Documentation

7.27.3.1 void VME::TDCMeasurement::Dump () [inline]

Here is the call graph for this function:



7.27.3.2 uint16_t VME::TDCMeasurement::GetBunchId () [inline]

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

7.27.3.4 uint32_t VME::TDCMeasurement::GetETTT () [inline]

7.27.3.5 uint16_t VME::TDCMeasurement::GetEventId () [inline]

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

7.27.3.7 uint16_t VME::TDCMeasurement::GetTDCId () [inline]

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

Here is the call graph for this function:

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

7.27.3.10 `size_t VME::TDCMeasurement::NumErrors () const [inline]`

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

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

7.27.4 Field Documentation

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

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

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

- `include/VME_TDCMeasurement.h`

7.28 VME::TDCV1x90 Class Reference

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

Public Types

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

Public Member Functions

- [TDCV1x90](#) (int32_t bhandle, uint32_t baseaddr)
- [~TDCV1x90](#) ()
- void [SetVerboseLevel](#) (unsigned short verb=1)
- void [SetTestMode](#) (bool en=true) const
- bool [GetTestMode](#) () const
- uint32_t [GetModel](#) () const
- uint32_t [GetOUI](#) () const
- uint32_t [GetSerialNumber](#) () const
- uint16_t [GetFirmwareRevision](#) () const
- void [CheckConfiguration](#) () const
- void [EnableChannel](#) (short) const
- void [DisableChannel](#) (short) const
- void [SetPol](#) (uint16_t word1, uint16_t word2) const
- std::map< unsigned short, bool > [GetPol](#) () const
- void [SetLSBTraileadEdge](#) ([trailead_edge_lsb](#)) const
- void [SetAcquisitionMode](#) (const [AcquisitionMode](#) &)
- [AcquisitionMode](#) [GetAcquisitionMode](#) ()
- void [SetTriggerMatching](#) ()
- void [SetContinuousStorage](#) ()
- void [SetDetectionMode](#) (const [DetectionMode](#) &detm)
- [DetectionMode](#) [GetDetectionMode](#) ()
- void [SetDLLClock](#) (const [DLLMode](#) &dll) const
- [DLLMode](#) [GetDLLClock](#) () const
- void [SetGlobalOffset](#) (const [GlobalOffset](#) &) const
- [GlobalOffset](#) [GetGlobalOffset](#) () const
- void [SetRCAdjust](#) (int, uint16_t) const
- uint16_t [GetRCAdjust](#) (int) const
- uint32_t [GetEventCounter](#) () const
Number of occurred triggers.
- uint16_t [GetEventStored](#) () const
Number of events currently stored in the output buffer.

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

Private Member Functions

- bool [WaitMicro](#) (const [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]

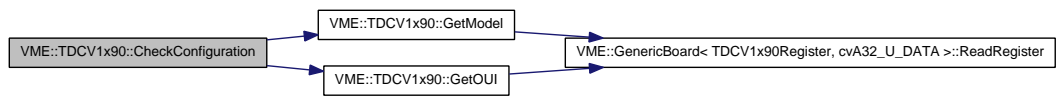
7.28.3.2 VME::TDCV1x90::~~TDCV1x90 ()

7.28.4 Member Function Documentation

7.28.4.1 void VME::TDCV1x90::abort ()

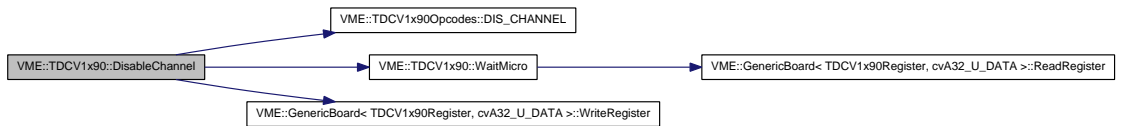
7.28.4.2 void VME::TDCV1x90::CheckConfiguration () const

Here is the call graph for this function:



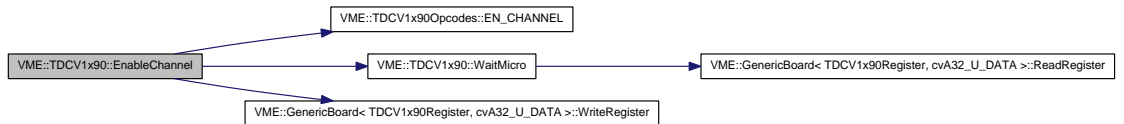
7.28.4.3 void VME::TDCV1x90::DisableChannel (short *channel_id*) const

Here is the call graph for this function:



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

Here is the call graph for this function:

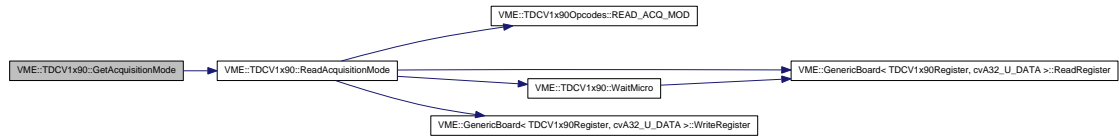


7.28.4.5 TDCEventCollection VME::TDCV1x90::FetchEvents ()

Here is the call graph for this function:

7.28.4.6 AcquisitionMode VME::TDCV1x90::GetAcquisitionMode () [inline]

Here is the call graph for this function:



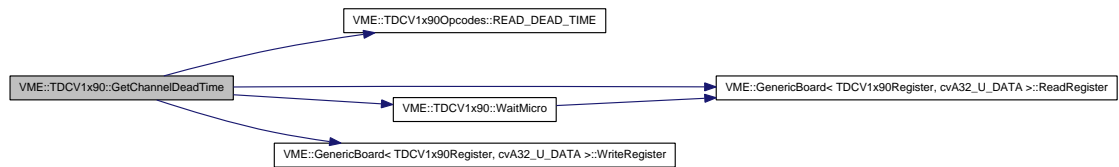
7.28.4.7 uint16_t VME::TDCV1x90::GetBLTEventNumberRegister () const

Here is the call graph for this function:



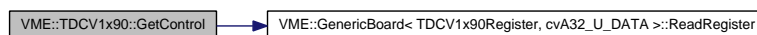
7.28.4.8 unsigned short VME::TDCV1x90::GetChannelDeadTime () const

Here is the call graph for this function:



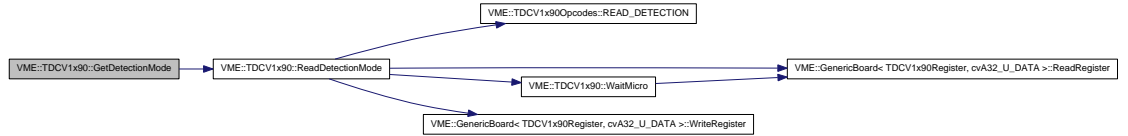
7.28.4.9 TDCV1x90Control VME::TDCV1x90::GetControl () const

Here is the call graph for this function:



7.28.4.10 DetectionMode VME::TDCV1x90::GetDetectionMode () [inline]

Here is the call graph for this function:

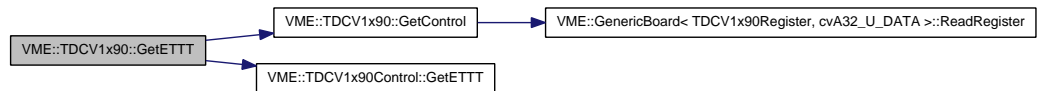


7.28.4.11 DLLMode VME::TDCV1x90::GetDLLClock () const

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

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

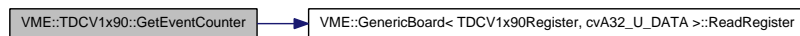
Here is the call graph for this function:



7.28.4.14 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.

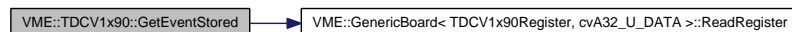
Here is the call graph for this function:



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

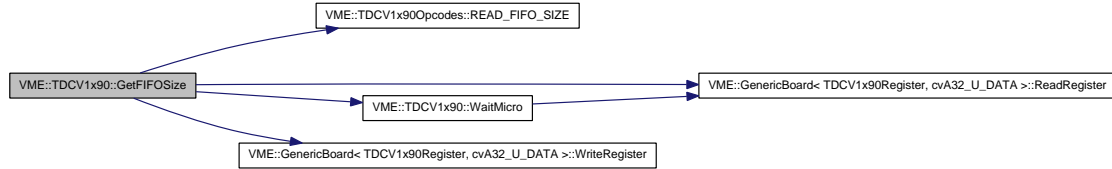
Number of events currently stored in the output buffer.

Here is the call graph for this function:

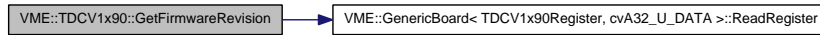


7.28.4.16 uint16_t VME::TDCV1x90::GetFIFOSize () const

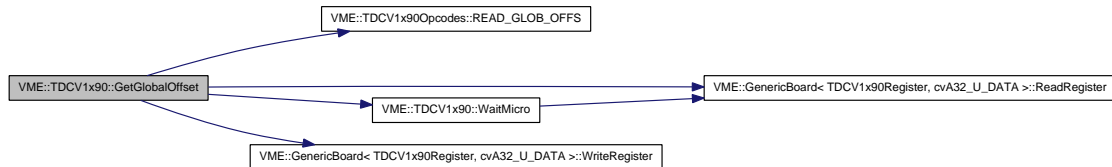
Here is the call graph for this function:

**7.28.4.17 uint16_t VME::TDCV1x90::GetFirmwareRevision () const**

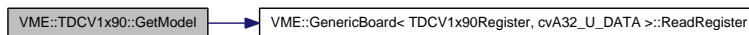
Here is the call graph for this function:

**7.28.4.18 GlobalOffset VME::TDCV1x90::GetGlobalOffset () const**

Here is the call graph for this function:

**7.28.4.19 uint32_t VME::TDCV1x90::GetModel () const**

Here is the call graph for this function:

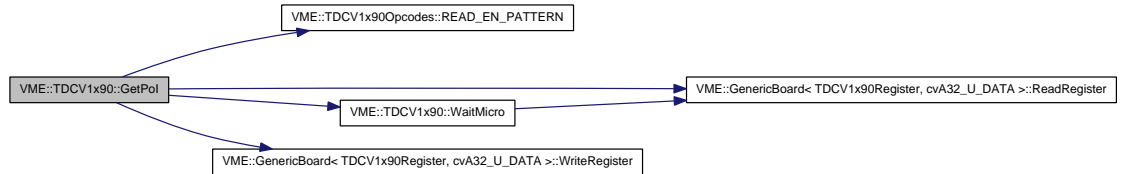
**7.28.4.20 uint32_t VME::TDCV1x90::GetOUI () const**

Here is the call graph for this function:



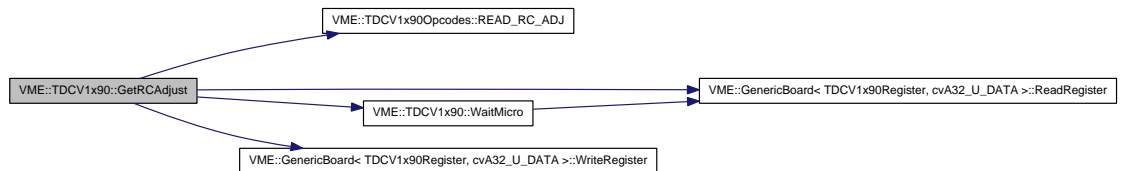
7.28.4.21 `std::map< unsigned short, bool > VME::TDCV1x90::GetPol () const`

Here is the call graph for this function:



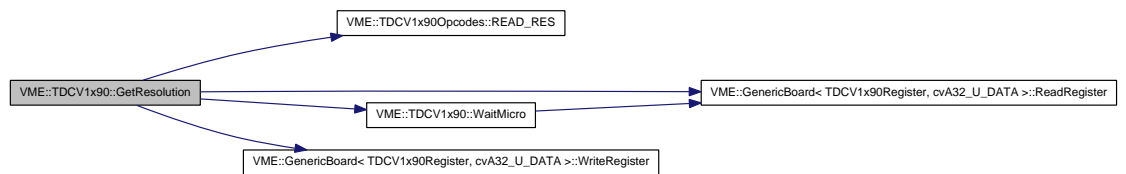
7.28.4.22 `uint16_t VME::TDCV1x90::GetRCAdjust (int tdc) const`

Here is the call graph for this function:



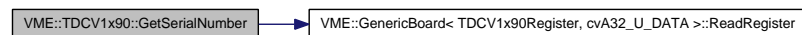
7.28.4.23 `uint16_t VME::TDCV1x90::GetResolution () const`

Here is the call graph for this function:



7.28.4.24 `uint32_t VME::TDCV1x90::GetSerialNumber () const`

Here is the call graph for this function:

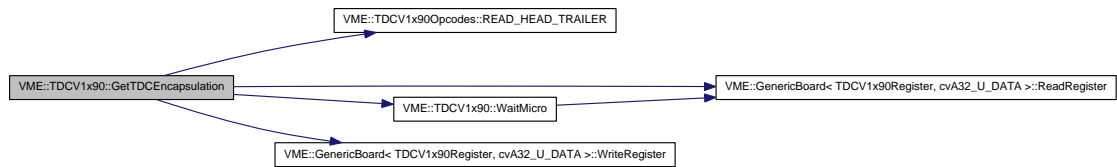


7.28.4.25 TDCV1x90Status VME::TDCV1x90::GetStatus () const

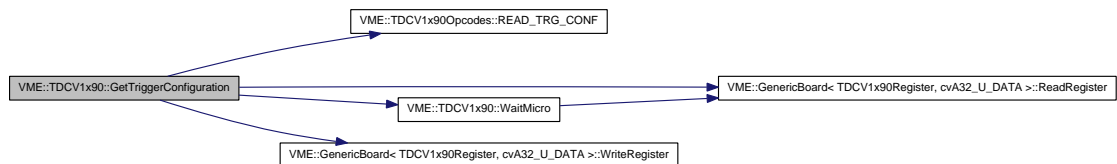
Here is the call graph for this function:

**7.28.4.26 bool VME::TDCV1x90::GetTDCEncapsulation () const**

Here is the call graph for this function:

**7.28.4.27 bool VME::TDCV1x90::GetTestMode () const****7.28.4.28 uint16_t VME::TDCV1x90::GetTriggerConfiguration (const trig_conf & type) const**

Here is the call graph for this function:



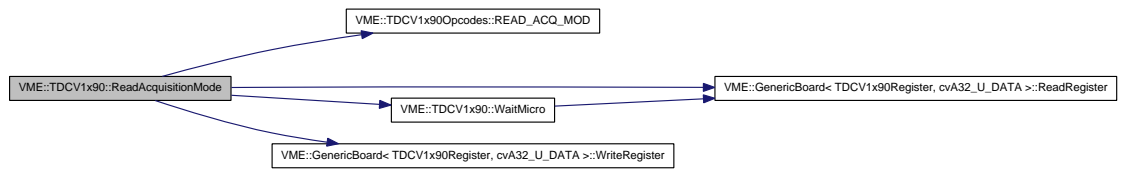
7.28.4.29 `int16_t VME::TDCV1x90::GetWindowOffset () const`

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

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

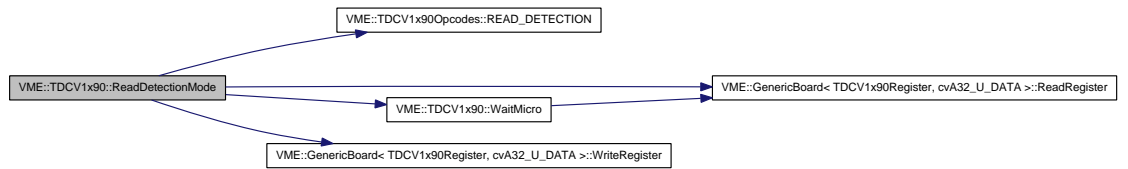
7.28.4.32 `void VME::TDCV1x90::ReadAcquisitionMode ()` `[private]`

Here is the call graph for this function:



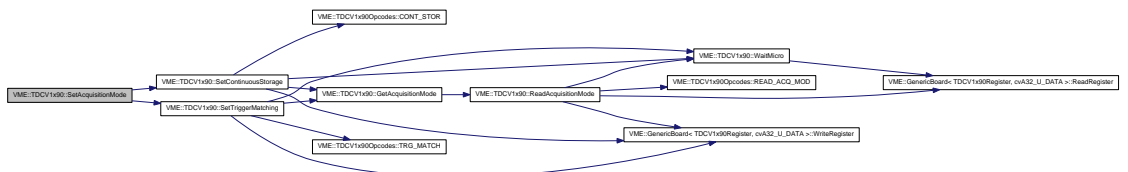
7.28.4.33 `void VME::TDCV1x90::ReadDetectionMode ()` `[private]`

Here is the call graph for this function:



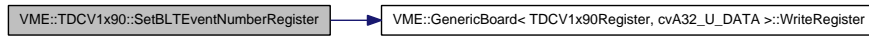
7.28.4.34 `void VME::TDCV1x90::SetAcquisitionMode (const AcquisitionMode & mode)`

Here is the call graph for this function:



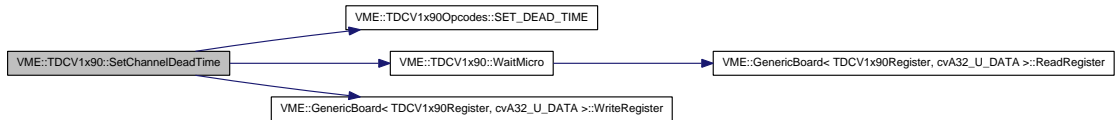
7.28.4.35 void VME::TDCV1x90::SetBLTEventNumberRegister (const uint16_t & value) const

Here is the call graph for this function:



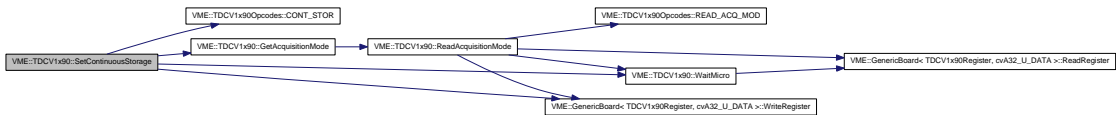
7.28.4.36 void VME::TDCV1x90::SetChannelDeadTime (unsigned short dt) const

Here is the call graph for this function:



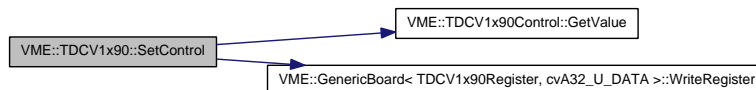
7.28.4.37 void VME::TDCV1x90::SetContinuousStorage ()

Here is the call graph for this function:



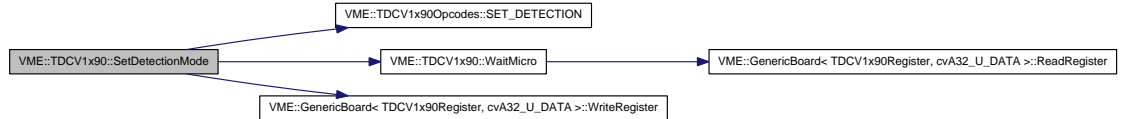
7.28.4.38 void VME::TDCV1x90::SetControl (const TDCV1x90Control & control) const

Here is the call graph for this function:



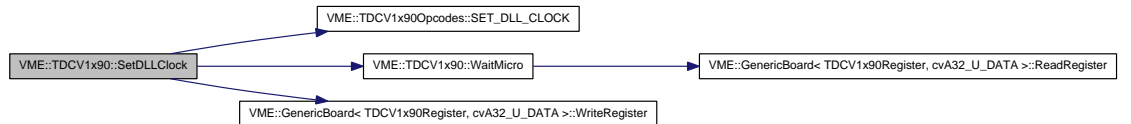
7.28.4.39 void VME::TDCV1x90::SetDetectionMode (const DetectionMode & *detm*)

Here is the call graph for this function:



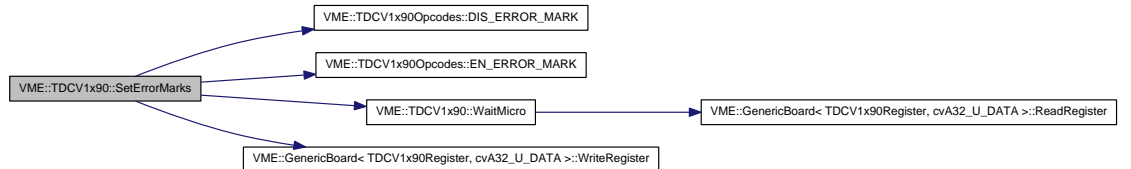
7.28.4.40 void VME::TDCV1x90::SetDLLClock (const DLLMode & *dll*) const

Here is the call graph for this function:



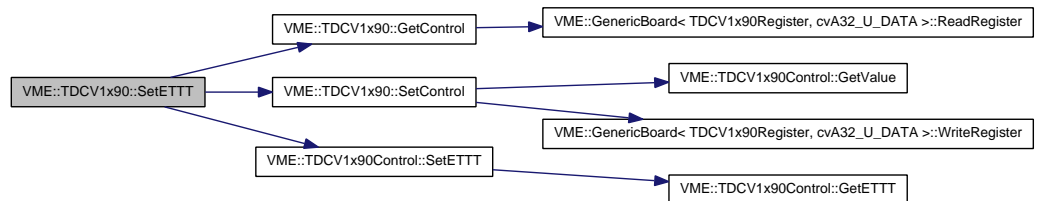
7.28.4.41 void VME::TDCV1x90::SetErrorMarks (bool *mode* = true)

Here is the call graph for this function:



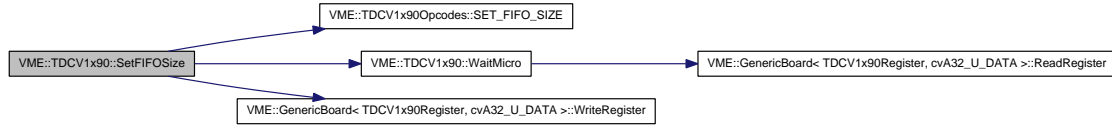
7.28.4.42 void VME::TDCV1x90::SetETTT (bool *ettt* = true) const [inline]

Here is the call graph for this function:

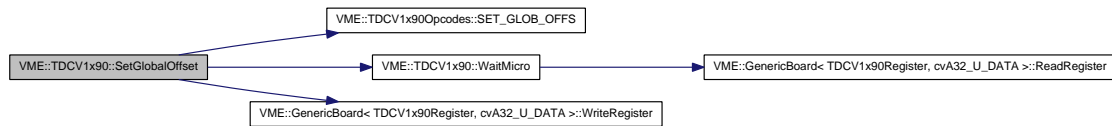


7.28.4.43 void VME::TDCV1x90::SetFIFOSize (const uint16_t & size) const

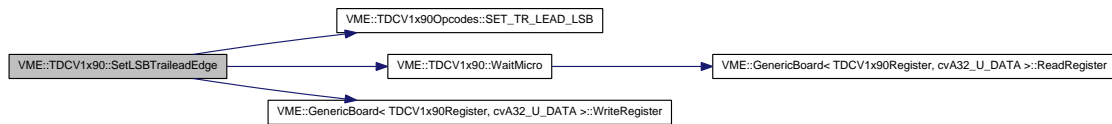
Here is the call graph for this function:

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

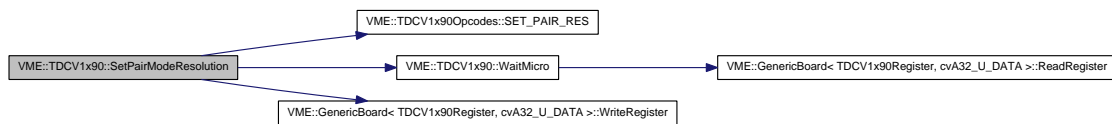
Here is the call graph for this function:

**7.28.4.45 void VME::TDCV1x90::SetLSBTrailEdge (trailead_edge_lsb conf) const**

Here is the call graph for this function:

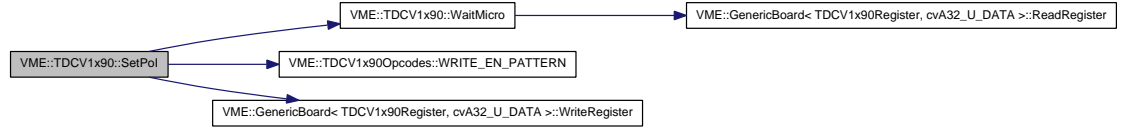
**7.28.4.46 void VME::TDCV1x90::SetPairModeResolution (int lead_time_res, int pulse_width_res) const**

Here is the call graph for this function:



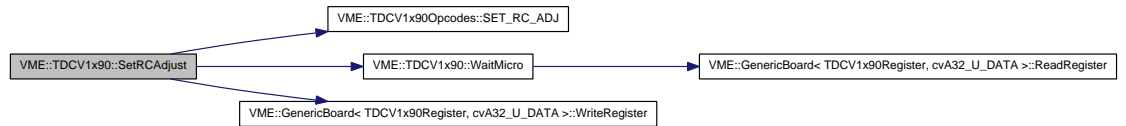
7.28.4.47 void VME::TDCV1x90::SetPol (uint16_t word1, uint16_t word2) const

Here is the call graph for this function:



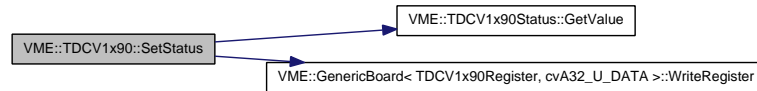
7.28.4.48 void VME::TDCV1x90::SetRCAdjust (int tdc, uint16_t value) const

Here is the call graph for this function:



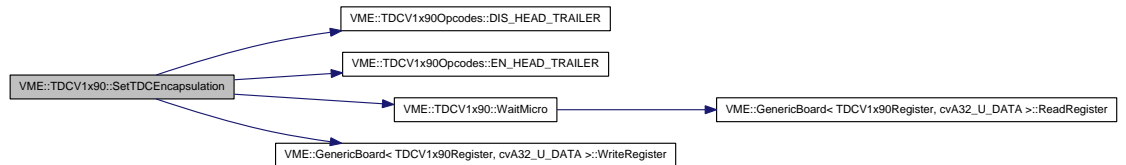
7.28.4.49 void VME::TDCV1x90::SetStatus (const TDCV1x90Status & status) const

Here is the call graph for this function:



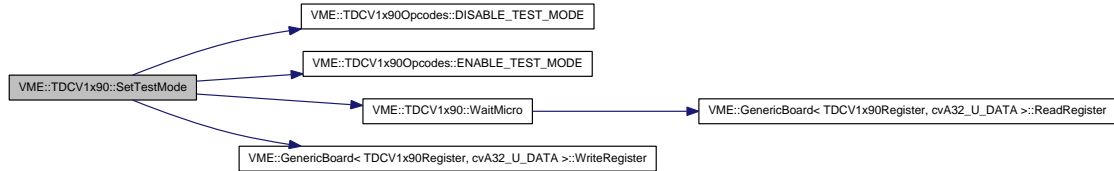
7.28.4.50 void VME::TDCV1x90::SetTDCEncapsulation (bool mode) const

Here is the call graph for this function:



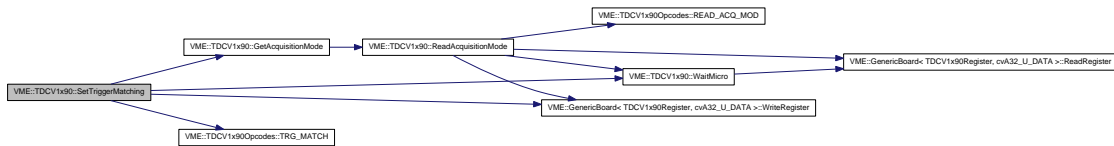
7.28.4.51 void VME::TDCV1x90::SetTestMode (bool *en* = true) const

Here is the call graph for this function:



7.28.4.52 void VME::TDCV1x90::SetTriggerMatching ()

Here is the call graph for this function:



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

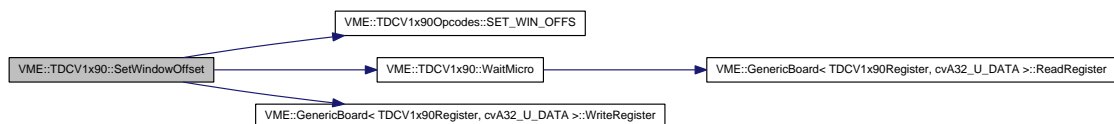
7.28.4.54 void VME::TDCV1x90::SetWindowOffset (const int16_t & *offs*) const

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:



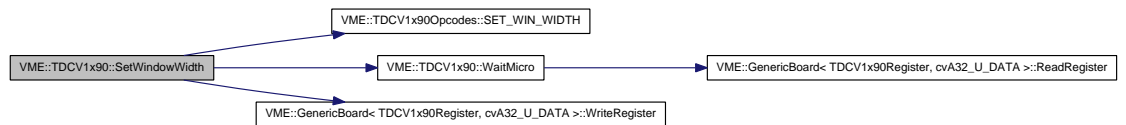
7.28.4.55 void VME::TDCV1x90::SetWindowWidth (const uint16_t & width)

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

Parameters:

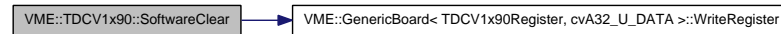
← **Window** width, in units of clock cycles

Here is the call graph for this function:



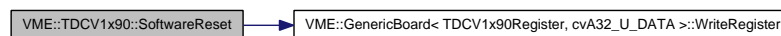
7.28.4.56 bool VME::TDCV1x90::SoftwareClear () const

Here is the call graph for this function:



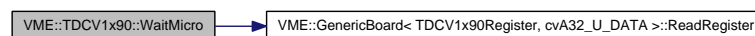
7.28.4.57 bool VME::TDCV1x90::SoftwareReset () const

Here is the call graph for this function:



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

Here is the call graph for this function:



7.28.5 Field Documentation

7.28.5.1 AcquisitionMode VME::TDCV1x90::fAcquisitionMode [private]

7.28.5.2 uint32_t* VME::TDCV1x90::fBuffer [private]

7.28.5.3 DetectionMode VME::TDCV1x90::fDetectionMode [private]

7.28.5.4 bool VME::TDCV1x90::fErrorMarks [private]

7.28.5.5 unsigned short VME::TDCV1x90::fVerb [private]

7.28.5.6 uint16_t VME::TDCV1x90::fWindowWidth [private]

7.28.5.7 bool VME::TDCV1x90::gEnd [private]

7.28.5.8 uint32_t VME::TDCV1x90::nchannels [private]

7.28.5.9 std::string VME::TDCV1x90::pair_lead_res[8] [private]

7.28.5.10 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

7.29 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](#)

7.29.1 Detailed Description

TDC control register.

Author:

Laurent Forthomme <laurent.forthomme@cern.ch>

Date:

Jun 2015

7.29.2 Constructor & Destructor Documentation

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

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

7.29.3 Member Function Documentation

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

Here is the call graph for this function:

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

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

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

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

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

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

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

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

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

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

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

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

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

Here is the call graph for this function:

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

Here is the call graph for this function:

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

Here is the call graph for this function:

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

Here is the call graph for this function:

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

Here is the call graph for this function:

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

Here is the call graph for this function:

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

Here is the call graph for this function:

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

Here is the call graph for this function:

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

Here is the call graph for this function:

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

Here is the call graph for this function:

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

Here is the call graph for this function:

7.29.4 Field Documentation

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

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

- include/VME_TDCV1x90.h

7.30 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](#)

7.30.1 Detailed Description

TDC status register.

Author:

Laurent Forthomme <laurent.forthomme@cern.ch>

Date:

Jun 2015

7.30.2 Member Enumeration Documentation

7.30.2.1 enum VME::TDCV1x90Status::TDCResolution

Enumerator:

R_800ps

R_200ps

R_100ps

R_25ps

7.30.3 Constructor & Destructor Documentation

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

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

7.30.4 Member Function Documentation

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

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

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

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

Here is the call graph for this function:

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

Here is the call graph for this function:

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

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

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

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

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

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

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

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

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

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

7.30.5 Field Documentation

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

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

- `include/VME_TDCV1x90.h`

7.31 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](#)

7.31.1 Field Documentation

7.31.1.1 uint32_t VME::trailead_t::ettt

7.31.1.2 uint32_t VME::trailead_t::event_count

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

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

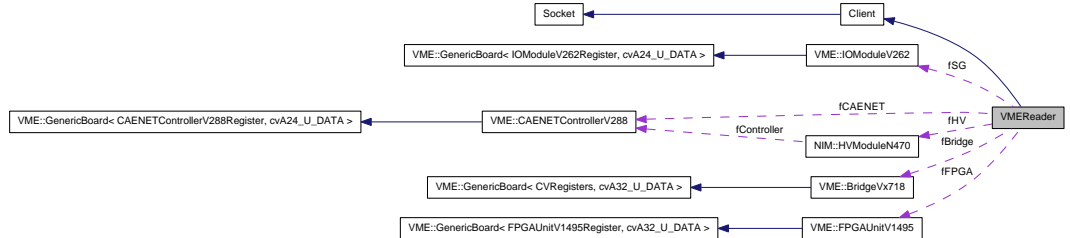
7.31.1.5 std::multimap<int32_t,int32_t> VME::trailead_t::trailing

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

- include/VME_TDCV1x90.h

7.32 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 **ReadXML** (const char *filename)
Load an XML configuration file.
- void **ReadXML** (std::string filename)
- 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.
- size_t **GetNumTDC** () const
- **VME::TDCCollection GetTDCCollection** ()
- void **AddIOModule** (uint32_t address)
- **VME::IOModuleV262 * GetIOModule** ()
- void **AddCFD** (uint32_t address)
Add a CFD to handle.
- **VME::CFDV812 * GetCFD** (uint32_t address)
Get a CFD on the VME bus Return a pointer to the CFD object, given its physical address on the VME bus.
- size_t **GetNumCFD** () const
- **VME::CFDCollection GetCFDCollection** ()
- void **AddFPGAUnit** (uint32_t address)
Add a multi-purposes FPGA board (CAEN V1495) to the crate controller.
- **VME::FPGAUnitV1495 * GetFPGAUnit** ()

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

- unsigned int [GetRunNumber](#) ()
Ask the socket master a run number.
- void [StartPulser](#) (double period, double width, unsigned int num_pulses=0)
Start the bridge's pulse generator [faulty].
- void [StopPulser](#) ()
Stop the bridge's pulse generator [faulty].
- void [SendPulse](#) (unsigned short output=0) const
Send a single pulse to the output register/plug connected to TDC boards.
- void [SendClear](#) () const
Send a clear signal to both the TDC boards.
- void [AddHVModule](#) (uint32_t vme_address, uint16_t nim_address)
Add a high voltage module (controlled by a VME-CAENET controller) to the DAQ.
- [NIM::HVModuleN470](#) * [GetHVModule](#) ()
Retrieve the [NIM](#) high voltage module.
- void [SetOutputFile](#) (uint32_t tdc_address, std::string filename)
Set the path to the output file where the DAQ is to write.
- std::string [GetOutputFile](#) (uint32_t tdc_address)
Return the path to the output file the DAQ is currently writing to.
- bool [UseSocket](#) () const
- void [Abort](#) ()
Abort data collection for all modules on the bus handled by the bridge.

Private Types

- typedef std::map< uint32_t, std::string > [OutputFiles](#)

Private Attributes

- [VME::BridgeVx718](#) * [fBridge](#)
The [VME](#) bridge object to handle.
- [VME::TDCCollection](#) [fTDCCollection](#)

A set of pointers to TDC objects indexed by their physical [VME](#) address.

- [VME::CFDCollection](#) [fCFDCollection](#)

A set of pointers to CFD 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.

- [VME::CAENETControllerV288](#) * [fCAENET](#)

Pointer to the [VME](#) CAENET controller.

- [NIM::HVModuleN470](#) * [fHV](#)

Pointer to the [NIM](#) high voltage module (passing through the CAENET controller).

- bool [fOnSocket](#)

Are we dealing with socket message passing?

- bool [fIsPulserStarted](#)

Is the bridge's pulser already started?

- [OutputFiles](#) [fOutputFiles](#)

7.32.1 Detailed Description

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

Author:

Laurent Forthomme <laurent.forthomme@cern.ch>

Date:

4 May 2015

7.32.2 Member Typedef Documentation

7.32.2.1 `typedef std::map<uint32_t, std::string> VMEReader::OutputFiles`
`[private]`

7.32.3 Constructor & Destructor Documentation

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

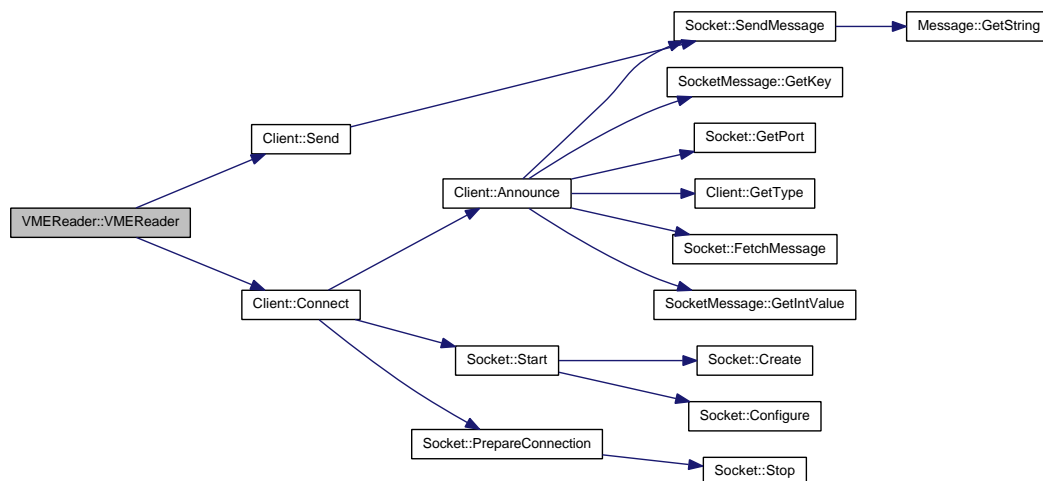
Parameters:

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

← *type* Bridge model

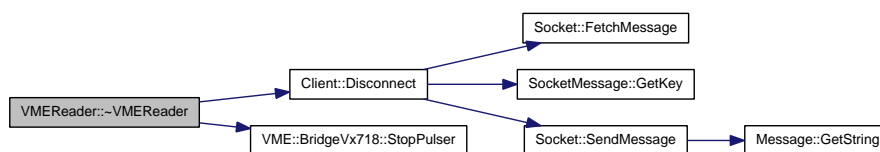
← *on_socket* Are we trying to connect through the socket?

Here is the call graph for this function:



7.32.3.2 VMEReader::~~VMEReader () [virtual]

Here is the call graph for this function:

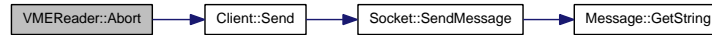


7.32.4 Member Function Documentation

7.32.4.1 void VMEReader::Abort ()

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

Here is the call graph for this function:



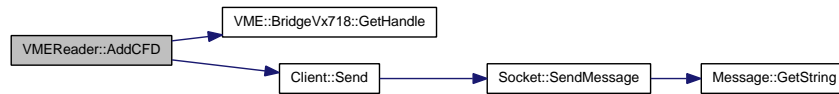
7.32.4.2 void VMEReader::AddCFD (uint32_t address)

Add a CFD to handle.

Parameters:

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

Here is the call graph for this function:



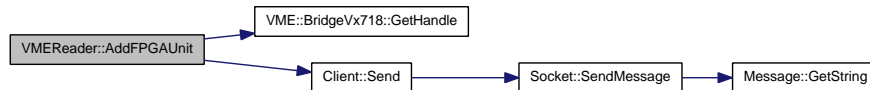
7.32.4.3 void VMEReader::AddFPGAUnit (uint32_t address)

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

Parameters:

← *address* 32-bit address of the TDC module on the [VME](#) bus

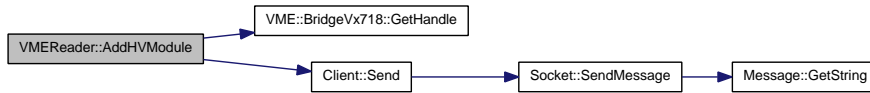
Here is the call graph for this function:



7.32.4.4 void VMEReader::AddHVModule (uint32_t vme_address, uint16_t nim_address)

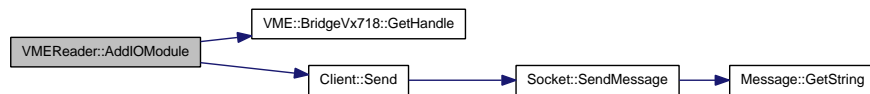
Add a high voltage module (controlled by a VME-CAENET controller) to the DAQ.

Here is the call graph for this function:



7.32.4.5 void VMEReader::AddIOModule (uint32_t address)

Here is the call graph for this function:



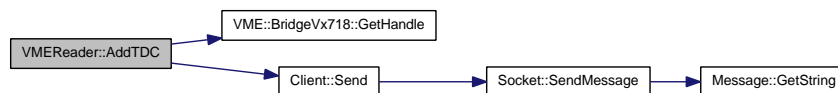
7.32.4.6 void VMEReader::AddTDC (uint32_t address)

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:



7.32.4.7 VME::CFDV812* VMEReader::GetCFD (uint32_t address) [inline]

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

7.32.4.8 VME::CFDCollection VMEReader::GetCFDCollection () [inline]

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

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

7.32.4.10 NIM::HVModuleN470* VMEReader::GetHVModule () [inline]

Retrieve the [NIM](#) high voltage module.

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

7.32.4.12 size_t VMEReader::GetNumCFD () const [inline]

7.32.4.13 size_t VMEReader::GetNumTDC () const [inline]

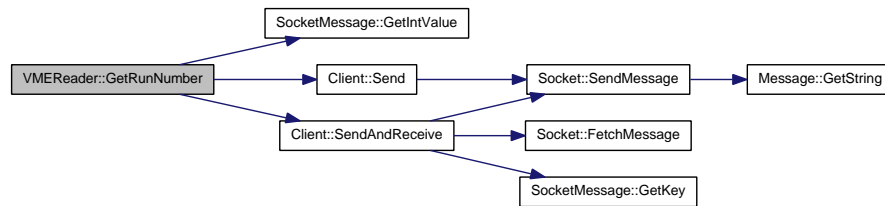
7.32.4.14 std::string VMEReader::GetOutputFile (uint32_t tdc_address) [inline]

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

7.32.4.15 unsigned int VMEReader::GetRunNumber ()

Ask the socket master a run number.

Here is the call graph for this function:



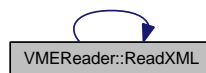
7.32.4.16 VME::TDCV1x90* VMEReader::GetTDC (uint32_t address) [inline]

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

7.32.4.17 VME::TDCCollection VMEReader::GetTDCCollection () [inline]

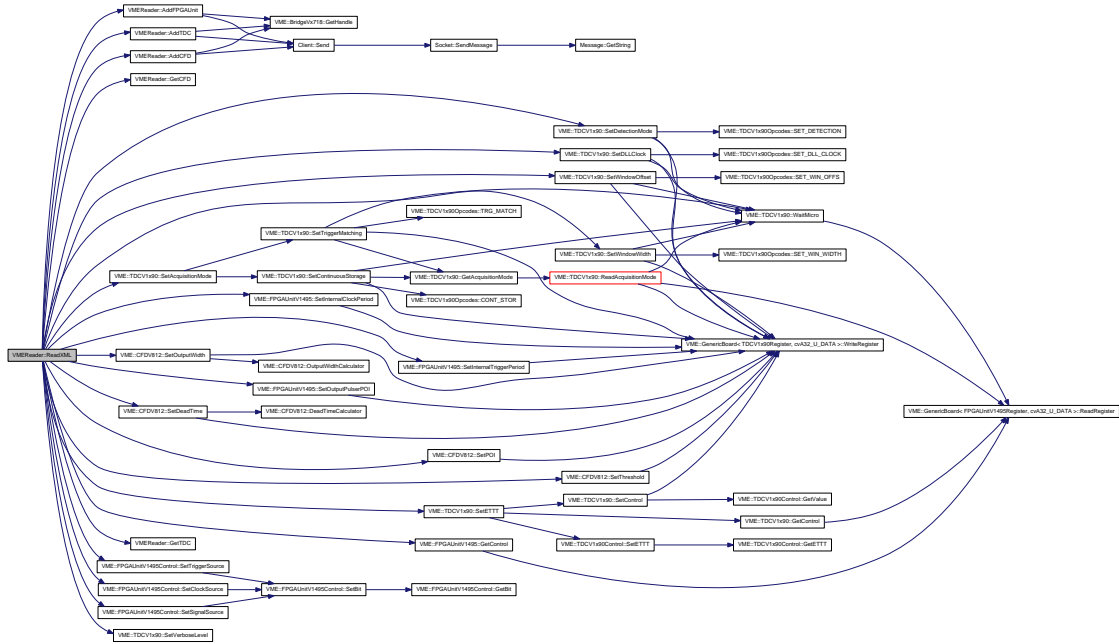
7.32.4.18 void VMEReader::ReadXML (std::string filename) [inline]

Here is the call graph for this function:



Load an XML configuration file.

Here is the call graph for this function:

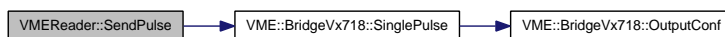


Send a clear signal to both the TDC boards.

Here is the call graph for this function:



Send a single pulse to the output register/plug connected to TDC boards.
Here is the call graph for this function:



7.32.4.22 void VMEReader::SetOutputFile (uint32_t *tdc_address*, std::string *filename*)

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

Here is the call graph for this function:



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

Start the bridge's pulse generator [faulty].

Here is the call graph for this function:



7.32.4.24 void VMEReader::StopPulser () [inline]

Stop the bridge's pulse generator [faulty].

Here is the call graph for this function:

7.32.4.25 bool VMEReader::UseSocket () const [inline]

7.32.5 Field Documentation

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

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

7.32.5.2 VME::CAENETControllerV288* VMEReader::fCAENET [private]

Pointer to the [VME](#) CAENET controller.

7.32.5.3 VME::CFDCollection VMEReader::fCFDCollection [private]

A set of pointers to CFD objects indexed by their physical [VME](#) address.

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

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

7.32.5.5 NIM::HVModuleN470* VMEReader::fHV [private]

Pointer to the [NIM](#) high voltage module (passing through the CAENET controller).

7.32.5.6 bool VMEReader::fIsPulserStarted [private]

Is the bridge's pulser already started?

7.32.5.7 bool VMEReader::fOnSocket [private]

Are we dealing with socket message passing?

7.32.5.8 OutputFiles VMEReader::fOutputFiles [private]

Path to the current output files the DAQ is writing to (indexed by the TDC id)

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

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

7.32.5.10 VME::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, [27](#)
- ~BridgeVx718Control
 - VME::BridgeVx718Control, [30](#)
- ~BridgeVx718Status
 - VME::BridgeVx718Status, [33](#)
- ~CAENETControllerV288
 - VME::CAENETControllerV288, [36](#)
- ~CAENETControllerV288Status
 - VME::CAENETControllerV288Status, [39](#)
- ~CFDV812
 - VME::CFDV812, [41](#)
- ~ChannelStatus
 - NIM::HVModuleN470Values::ChannelStatus, [48](#)
- ~Client
 - Client, [51](#)
- ~FPGAUnitV1495
 - VME::FPGAUnitV1495, [62](#)
- ~FPGAUnitV1495Control
 - VME::FPGAUnitV1495Control, [71](#)
- ~FileReader
 - FileReader, [58](#)
- ~GenericBoard
 - VME::GenericBoard, [75](#)
- ~HVModuleN470
 - NIM::HVModuleN470, [81](#)
- ~HVModuleN470ChannelValues
 - NIM::HVModuleN470ChannelValues, [86](#)
- ~HVModuleN470Values
 - NIM::HVModuleN470Values, [89](#)
- ~IOModuleV262
 - VME::IOModuleV262, [90](#)
- ~Message
 - Message, [93](#)
- ~Messenger
 - Messenger, [96](#)
- ~PCIInterfaceA2818
 - VME::PCIInterfaceA2818, [103](#)
- ~Socket
 - Socket, [106](#)
- ~SocketMessage
 - SocketMessage, [113](#)
- ~TDCErrorFlag
 - VME::TDCErrorFlag, [117](#)
- ~TDCEvent
 - VME::TDCEvent, [121](#)
- ~TDCMeasurement
 - VME::TDCMeasurement, [124](#)
- ~TDCV1x90
 - VME::TDCV1x90, [129](#)
- ~TDCV1x90Control
 - VME::TDCV1x90Control, [145](#)
- ~TDCV1x90Status
 - VME::TDCV1x90Status, [149](#)
- ~VMEReader
 - VMEReader, [155](#)
- Abort
 - VMEReader, [156](#)
- abort
 - VME::TDCV1x90, [130](#)
- AcceptConnections
 - Socket, [106](#)
- acq_mode
 - file_header_t, [56](#)
- AcquisitionMode
 - VME, [16](#)
- AddCFD
 - VMEReader, [156](#)
- AddClient
 - Messenger, [97](#)
- AddFPGAUnit
 - VMEReader, [156](#)
- AddHVModule
 - VMEReader, [156](#)
- AddIOModule
 - VMEReader, [157](#)
- AddTDC
 - VMEReader, [157](#)

- Alarm
 - NIM::HVModuleN470Values::ChannelStatus, [48](#)
- AlmostFull
 - VME::TDCV1x90Status, [149](#)
- Announce
 - Client, [51](#)
- AUTOLOAD_DEF_CONFI
 - VME::TDCV1x90OpCodes, [24](#)
- AUTOLOAD_USER_CONF
 - VME::TDCV1x90OpCodes, [24](#)
- Bind
 - Socket, [106](#)
- BridgeType
 - VME, [16](#)
- BridgeVx718
 - VME::BridgeVx718, [27](#)
- BridgeVx718Control
 - VME::BridgeVx718Control, [30](#)
- BridgeVx718Status
 - VME::BridgeVx718Status, [33](#)
- Broadcast
 - Messenger, [97](#)
- BusError
 - VME::TDCV1x90Status, [149](#)
- CAEN_V1718
 - VME, [16](#)
- CAEN_V2718
 - VME, [16](#)
- CAENETControllerV288
 - VME::CAENETControllerV288, [36](#)
- CAENETControllerV288Answer
 - VME, [16](#)
- CAENETControllerV288Register
 - VME, [16](#)
- CAENETControllerV288Status
 - VME::CAENETControllerV288Status, [39](#)
- CFDCollection
 - VME, [15](#)
- CFDV812
 - VME::CFDV812, [41](#)
- CFDV812Register
 - VME, [16](#)
- ChannelStatus
 - NIM::HVModuleN470ChannelValues, coarse [86](#)
 - NIM::HVModuleN470Values::ChannelStatus, [48](#)
- CheckBoardVersion
 - VME::FPGAUnitV1495, [62](#)
- CheckConfiguration
 - VME::BridgeVx718, [27](#)
 - VME::CFDV812, [41](#)
 - VME::TDCV1x90, [130](#)
- CheckPCIInterface
 - VME::BridgeVx718, [27](#)
- Clear
 - FileReader, [58](#)
- CLEAR_KEEP_TOKEN
 - VME::TDCV1x90OpCodes, [24](#)
- ClearOutputPulser
 - VME::FPGAUnitV1495, [62](#)
- CLIENT
 - Socket, [106](#)
- Client, [50](#)
 - ~Client, [51](#)
 - Announce, [51](#)
 - Client, [51](#)
 - Connect, [52](#)
 - Disconnect, [52](#)
 - fClientId, [55](#)
 - fIsConnected, [55](#)
 - fType, [55](#)
 - GetType, [53](#)
 - ParseMessage, [53](#)
 - Receive, [53](#)
 - Send, [54](#)
 - SendAndReceive, [54](#)
- ClockSource
 - VME::FPGAUnitV1495Control, [70](#)
- cnBusy
 - VME, [16](#)
- cnIncorrectHCC
 - VME, [16](#)
- cnIncorrectValue
 - VME, [16](#)
- cnNoData
 - VME, [16](#)
- cnSuccess
 - VME, [16](#)
- cnUnrecognizedCode
 - VME, [16](#)
- cnWrongModuleAddress
 - VME, [16](#)
- VME::GlobalOffset, [77](#)

- Configure
 - Socket, [107](#)
- Connect
 - Client, [52](#)
 - Messenger, [98](#)
- CONT_STORAGE
 - VME, [16](#)
- CONT_STOR
 - VME::TDCV1x90Opcodes, [24](#)
- Create
 - Socket, [107](#)
- DataReady
 - VME::TDCV1x90Status, [149](#)
- DeadTimeCalculator
 - VME::CFDV812, [41](#)
- Decode
 - HTTPMessage, [79](#)
- DEFAULT_SETUP_REG
 - VME::TDCV1x90Opcodes, [24](#)
- det_mode
 - file_header_t, [56](#)
- DetectionMode
 - VME, [17](#)
- DETECTOR
 - Socket, [106](#)
- DIS_ALL_CHANNEL
 - VME::TDCV1x90Opcodes, [24](#)
- DIS_CHANNEL
 - VME::TDCV1x90Opcodes, [24](#)
- DIS_ERROR_BYPASS
 - VME::TDCV1x90Opcodes, [24](#)
- DIS_ERROR_MARK
 - VME::TDCV1x90Opcodes, [24](#)
- DIS_HEAD_TRAILER
 - VME::TDCV1x90Opcodes, [24](#)
- DIS_SUB_TRG
 - VME::TDCV1x90Opcodes, [24](#)
- DISABLE_TEST_MODE
 - VME::TDCV1x90Opcodes, [24](#)
- DisableChannel
 - VME::TDCV1x90, [130](#)
- Disconnect
 - Client, [52](#)
 - Messenger, [98](#)
- DisconnectClient
 - Messenger, [98](#)
- DLL_Direct_LowRes
 - VME::TDCV1x90, [129](#)
- DLL_PLL_HighRes
 - VME::TDCV1x90, [129](#)
- DLL_PLL_LowRes
 - VME::TDCV1x90, [129](#)
- DLL_PLL_MedRes
 - VME::TDCV1x90, [129](#)
- DLLMode
 - VME::TDCV1x90, [129](#)
- DQM
 - Socket, [106](#)
- Dump
 - FileReader, [58](#)
 - HTTPMessage, [79](#)
 - Message, [93](#)
 - NIM::HVModuleN470ChannelValues, [86](#)
 - NIM::HVModuleN470Values, [89](#)
 - NIM::HVModuleN470Values::ChannelStatus, [48](#)
 - SocketMessage, [113](#)
 - VME::BridgeVx718Status, [33](#)
 - VME::FPGAUnitV1495Control, [71](#)
 - VME::TDCErrorFlag, [117](#)
 - VME::TDCEvent, [121](#)
 - VME::TDCMeasurement, [125](#)
 - VME::TDCV1x90Control, [145](#)
 - VME::TDCV1x90Status, [149](#)
- DumpConnected
 - Socket, [107](#)
- DumpFWInformation
 - VME::FPGAUnitV1495, [62](#)
- EN_ALL_CHANNEL
 - VME::TDCV1x90Opcodes, [24](#)
- EN_CHANNEL
 - VME::TDCV1x90Opcodes, [24](#)
- EN_ERROR_BYPASS
 - VME::TDCV1x90Opcodes, [24](#)
- EN_ERROR_MARK
 - VME::TDCV1x90Opcodes, [24](#)
- EN_HEAD_TRAILER
 - VME::TDCV1x90Opcodes, [24](#)
- EN_SUB_TRG
 - VME::TDCV1x90Opcodes, [24](#)
- ENABLE_TEST_MODE
 - VME::TDCV1x90Opcodes, [24](#)
- EnableChannel
 - VME::TDCV1x90, [130](#)
- Enabled
 - NIM::HVModuleN470Values::ChannelStatus, [48](#)

- Encode
 - HTTPMessage, 79
- Error
 - VME::TDCV1x90Status, 149
- ETTT
 - VME::TDCEvent, 120
- ettt
 - VME::trailead_t, 151
- event_count
 - VME::trailead_t, 151
- EventType
 - VME::TDCEvent, 120
- ExternalClock
 - VME::FPGAUnitV1495Control, 70
- ExternalSignal
 - VME::FPGAUnitV1495Control, 70
- ExternalTrigger
 - VME::FPGAUnitV1495Control, 70
- EXTRA_SEARCH_WIN_WIDTH
 - VME, 20
- fAcquisitionMode
 - VME::TDCV1x90, 143
- fAddress
 - NIM::HVModuleN470, 84
 - Socket, 109
- fBaseAddr
 - VME::GenericBoard, 76
- fBridge
 - VMEReader, 160
- fBuffer
 - Socket, 109
 - VME::TDCV1x90, 143
- fCAENET
 - VMEReader, 160
- fCFDCollection
 - VMEReader, 160
- fChannelId
 - NIM::HVModuleN470ChannelValues, 87
- fClientId
 - Client, 55
- fController
 - NIM::HVModuleN470, 84
- fDetectionMode
 - VME::TDCV1x90, 143
- fErrorMarks
 - VME::TDCV1x90, 143
- FetchBuffer
 - VME::CAENETControllerV288, 36
- FetchEvents
 - VME::TDCV1x90, 130
- FetchMessage
 - Socket, 107
- fEvents
 - VME::TDCMeasurement, 126
- fFile
 - FileReader, 59
- fFPGA
 - VMEReader, 160
- fHandle
 - VME::GenericBoard, 76
 - VME::PCIInterfaceA2818, 103
- fHasIRQ
 - VME::BridgeVx718, 28
- fHeader
 - FileReader, 59
- fHV
 - VMEReader, 161
- file_header_t, 56
 - acq_mode, 56
 - det_mode, 56
 - magic, 56
 - num_hptdc, 56
 - run_id, 56
 - spill_id, 56
- FileReader, 57
 - ~FileReader, 58
 - Clear, 58
 - Dump, 58
 - fFile, 59
 - fHeader, 59
 - FileReader, 58
 - fNumEvents, 59
 - fReadoutMode, 59
 - fWriteTime, 59
 - GetNextEvent, 58
 - GetNextMeasurement, 58
 - GetNumEvents, 59
 - GetNumTDCs, 59
 - IsOpen, 59
 - Open, 59
- Filler
 - VME::TDCEvent, 120
- fine
 - VME::GlobalOffset, 77
- fIsConnected
 - Client, 55
- fIsPulserStarted
 - VMEReader, 161

- fMap
 - VME::TDCMeasurement, [126](#)
- fMaster
 - Socket, [109](#)
- fMessage
 - SocketMessage, [115](#)
- fNumAttempts
 - Messenger, [102](#)
- fNumEvents
 - FileReader, [59](#)
- fOnSocket
 - VMEReader, [161](#)
- fOriginalString
 - HTTPMessage, [79](#)
- fOutputFiles
 - VMEReader, [161](#)
- FPGAUnitV1495
 - VME::FPGAUnitV1495, [62](#)
- FPGAUnitV1495Control
 - VME::FPGAUnitV1495Control, [71](#)
- FPGAUnitV1495Register
 - VME, [17](#)
- fPID
 - Messenger, [102](#)
- fPort
 - Socket, [109](#)
- fReadFds
 - Socket, [109](#)
- fReadoutMode
 - FileReader, [59](#)
- fScalerStarted
 - VME::FPGAUnitV1495, [68](#)
- fSG
 - VMEReader, [161](#)
- fSocketId
 - Socket, [109](#)
- fSocketsConnected
 - Socket, [109](#)
- fStderrPipe
 - Messenger, [102](#)
- fStdoutPipe
 - Messenger, [102](#)
- fString
 - Message, [93](#)
- fTDCCollection
 - VMEReader, [161](#)
- fType
 - Client, [55](#)
- Full
 - VME::TDCV1x90Status, [150](#)
- fValues
 - NIM::HVModuleN470ChannelValues, [87](#)
 - NIM::HVModuleN470Values, [89](#)
- fVerb
 - VME::TDCV1x90, [143](#)
- fWindowWidth
 - VME::TDCV1x90, [143](#)
- fWord
 - NIM::HVModuleN470Values::ChannelStatus, [48](#)
 - VME::BridgeVx718Control, [32](#)
 - VME::BridgeVx718Status, [34](#)
 - VME::CAENETControllerV288Status, [39](#)
 - VME::FPGAUnitV1495Control, [73](#)
 - VME::TDCErrorFlag, [118](#)
 - VME::TDCEvent, [123](#)
 - VME::TDCV1x90Control, [147](#)
 - VME::TDCV1x90Status, [150](#)
- fWriteTime
 - FileReader, [59](#)
- fWS
 - HTTPMessage, [79](#)
 - Messenger, [102](#)
- gEnd
 - VME::TDCV1x90, [143](#)
- GenericBoard
 - VME::GenericBoard, [75](#)
- GetAcquisitionMode
 - VME::TDCV1x90, [130](#)
- GetAddressIncrement
 - VME::BridgeVx718Control, [30](#)
- GetAlign64
 - VME::TDCV1x90Control, [145](#)
- GetArbiterType
 - VME::BridgeVx718Control, [31](#)
- GetBERR
 - VME::BridgeVx718Status, [33](#)
- GetBit
 - VME::FPGAUnitV1495Control, [71](#)
- GetBLTEventNumberRegister
 - VME::TDCV1x90, [131](#)
- GetBunchId
 - VME::TDCEvent, [121](#)
 - VME::TDCMeasurement, [125](#)
- GetBusError
 - VME::TDCV1x90Control, [145](#)
- GetBusReqLevel

- VME::BridgeVx718Control, [31](#)
- GetBusTimeout
 - VME::BridgeVx718Control, [31](#)
- GetCAENFirmwareRevision
 - VME::FPGAUnitV1495, [63](#)
- GetCFD
 - VMEReader, [157](#)
- GetCFDCollection
 - VMEReader, [157](#)
- GetChannelDeadTime
 - VME::TDCV1x90, [131](#)
- GetChannelId
 - VME::TDCEvent, [121](#)
 - VME::TDCMeasurement, [125](#)
- GetChannelStatus
 - NIM::HVModuleN470Values, [89](#)
- GetCleanedValue
 - SocketMessage, [113](#)
- GetClockSource
 - VME::FPGAUnitV1495Control, [71](#)
- GetCompensation
 - VME::TDCV1x90Control, [145](#)
- GetControl
 - VME::FPGAUnitV1495, [63](#)
 - VME::TDCV1x90, [131](#)
- GetDetectionMode
 - VME::TDCV1x90, [131](#)
- GetDipSwitch
 - VME::BridgeVx718Status, [34](#)
- GetDLLClock
 - VME::TDCV1x90, [132](#)
- GetDTACK
 - VME::BridgeVx718Status, [34](#)
- GetEmptyEvent
 - VME::TDCV1x90Control, [145](#)
- GetErrorFlags
 - VME::TDCEvent, [121](#)
- GetErrorMarks
 - VME::TDCV1x90, [132](#)
- GetETTT
 - VME::TDCEvent, [121](#)
 - VME::TDCMeasurement, [125](#)
 - VME::TDCV1x90, [132](#)
 - VME::TDCV1x90Control, [145](#)
- GetEventCount
 - VME::TDCEvent, [121](#)
- GetEventCounter
 - VME::TDCV1x90, [132](#)
- GetEventFIFO
 - VME::TDCV1x90Control, [145](#)
- GetEventId
 - VME::TDCEvent, [122](#)
 - VME::TDCMeasurement, [125](#)
- GetEventStored
 - VME::TDCV1x90, [132](#)
- GetFIFOSize
 - VME::TDCV1x90, [132](#)
- GetFirmwareRevision
 - VME::TDCV1x90, [133](#)
- GetFixedCode
 - VME::CFDV812, [42](#)
- GetFPGAUnit
 - VMEReader, [157](#)
- GetFWRevision
 - NIM::HVModuleN470, [81](#)
 - VME::PCInterfaceA2818, [103](#)
- GetGeo
 - VME::TDCEvent, [122](#)
- GetGeoAddress
 - VME::FPGAUnitV1495, [63](#)
- GetGlobalOffset
 - VME::TDCV1x90, [133](#)
- GetHandle
 - VME::BridgeVx718, [27](#)
- GetHardwareRevision
 - VME::FPGAUnitV1495, [63](#)
- GetHVModule
 - VMEReader, [157](#)
- GetIdentifier
 - VME::IOModuleV262, [90](#)
- GetInternalClockPeriod
 - VME::FPGAUnitV1495, [64](#)
- GetInternalTriggerPeriod
 - VME::FPGAUnitV1495, [64](#)
- GetInterruptReq
 - VME::BridgeVx718Control, [31](#)
- GetIntValue
 - SocketMessage, [113](#)
- GetIOModule
 - VMEReader, [158](#)
- GetIRQStatus
 - VME::BridgeVx718, [27](#)
- GetKey
 - HTTPMessage, [79](#)
 - Message, [93](#)
 - SocketMessage, [113](#)
- GetLeadingTime
 - VME::TDCMeasurement, [125](#)
- GetManufacturerId
 - VME::CFDV812, [42](#)

- VME::IOModuleV262, 90
- GetMEBAccess
 - VME::TDCV1x90Control, 145
- GetModel
 - VME::TDCV1x90, 133
- GetModuleId
 - NIM::HVModuleN470, 81
- GetModuleType
 - VME::CFDV812, 42
 - VME::IOModuleV262, 91
- GetModuleVersion
 - VME::CFDV812, 42
 - VME::IOModuleV262, 91
- GetNextEvent
 - FileReader, 58
- GetNextMeasurement
 - FileReader, 58
- GetNumCFD
 - VMEReader, 158
- GetNumEvents
 - FileReader, 59
- GetNumTDC
 - VMEReader, 158
- GetNumTDCs
 - FileReader, 59
- GetOperationStatus
 - VME::CAENETControllerV288Status, 39
- GetOUI
 - VME::TDCV1x90, 133
- GetOutputDelay
 - VME::FPGAUnitV1495, 64
- GetOutputFile
 - VMEReader, 158
- GetOutputPulser
 - VME::FPGAUnitV1495, 64
- GetPoI
 - VME::TDCV1x90, 134
- GetPort
 - Socket, 107
- GetRCAdjust
 - VME::TDCV1x90, 134
- GetReleaseType
 - VME::BridgeVx718Control, 31
- GetRequesterType
 - VME::BridgeVx718Control, 31
- GetResolution
 - VME::TDCV1x90, 134
- GetRunNumber
 - VMEReader, 158
- GetScalerStatus
 - VME::FPGAUnitV1495Control, 71
- GetScalerValue
 - VME::FPGAUnitV1495, 65
- GetSerialNumber
 - VME::CFDV812, 42
 - VME::FPGAUnitV1495, 65
 - VME::IOModuleV262, 91
 - VME::TDCV1x90, 134
- GetSignalSource
 - VME::FPGAUnitV1495Control, 71
- GetSocketId
 - Socket, 107
- GetSocketType
 - Socket, 107
- GetSRAMCompensation
 - VME::TDCV1x90Control, 145
- GetStatus
 - VME::BridgeVx718, 28
 - VME::CAENETControllerV288, 36
 - VME::TDCEvent, 122
 - VME::TDCV1x90, 134
- GetString
 - Message, 93
 - SocketMessage, 113
- GetSWTermination
 - VME::TDCV1x90Control, 145
- GetSysRes
 - VME::BridgeVx718Control, 31
- GetSystemControl
 - VME::BridgeVx718Status, 34
- GetSystemReset
 - VME::BridgeVx718Status, 34
- GetTDC
 - VMEReader, 158
- GetTDCBits
 - VME::FPGAUnitV1495, 65
- GetTDCCollection
 - VMEReader, 158
- GetTDCEncapsulation
 - VME::TDCV1x90, 135
- GetTDCId
 - VME::TDCEvent, 122
 - VME::TDCMeasurement, 125
- GetTermination
 - VME::TDCV1x90Control, 145
- GetTestFIFO
 - VME::TDCV1x90Control, 145
- GetTestMode
 - VME::TDCV1x90, 135

- GetTime
 - VME::TDCEvent, [122](#)
- GetToT
 - VME::TDCMeasurement, [125](#)
- GetTrailingTime
 - VME::TDCMeasurement, [125](#)
- GetTriggerConfiguration
 - VME::TDCV1x90, [135](#)
- GetTriggerSource
 - VME::FPGAUnitV1495Control, [71](#)
- GetType
 - Client, [53](#)
 - Messenger, [99](#)
 - VME::TDCEvent, [122](#)
- GetUSBType
 - VME::BridgeVx718Status, [34](#)
- GetUserFirmwareRevision
 - VME::FPGAUnitV1495, [65](#)
- GetValue
 - SocketMessage, [114](#)
 - VME::TDCV1x90Control, [145](#)
 - VME::TDCV1x90Status, [150](#)
- GetVectorValue
 - SocketMessage, [114](#)
- GetWidth
 - VME::TDCEvent, [122](#)
- GetWindowOffset
 - VME::TDCV1x90, [135](#)
- GetWindowWidth
 - VME::TDCV1x90, [136](#)
- GetWord
 - VME::FPGAUnitV1495Control, [72](#)
 - VME::TDCErrorFlag, [117](#)
 - VME::TDCEvent, [123](#)
- GetWordCount
 - VME::TDCEvent, [123](#)
- GlobalHeader
 - VME::TDCEvent, [120](#)
- GlobalTrailer
 - VME::TDCEvent, [120](#)
- HardwareReset
 - VME::TDCV1x90, [136](#)
- HasGroupError
 - VME::TDCErrorFlag, [117](#)
- HasInternalChipError
 - VME::TDCErrorFlag, [117](#)
- HasL1BufferOverflow
 - VME::TDCErrorFlag, [117](#)
- HasReachedEventSizeLimit
 - VME::TDCErrorFlag, [117](#)
- HasReadoutFIFOOverflow
 - VME::TDCErrorFlag, [117](#)
- HasTriggerFIFOOverflow
 - VME::TDCErrorFlag, [117](#)
- HeadersEnabled
 - VME::TDCV1x90Status, [150](#)
- HTTPMessage, [78](#)
 - Decode, [79](#)
 - Dump, [79](#)
 - Encode, [79](#)
 - fOriginalString, [79](#)
 - fWS, [79](#)
 - GetKey, [79](#)
 - HTTPMessage, [78](#)
- HVEnabled
 - NIM::HVModuleN470Values::ChannelStatus, [48](#)
- HVModuleN470
 - NIM::HVModuleN470, [81](#)
- HVModuleN470ChannelValues
 - NIM::HVModuleN470ChannelValues, [86](#)
- HVModuleN470Opcodes
 - NIM, [12](#)
- HVModuleN470Values
 - NIM::HVModuleN470Values, [89](#)
- I0
 - NIM::HVModuleN470ChannelValues, [86](#)
- I1
 - NIM::HVModuleN470ChannelValues, [87](#)
- Imon
 - NIM::HVModuleN470ChannelValues, [87](#)
 - NIM::HVModuleN470Values, [89](#)
- InputConf
 - VME::BridgeVx718, [28](#)
- InputRead
 - VME::BridgeVx718, [28](#)
- InternalClock
 - VME::FPGAUnitV1495Control, [70](#)
- InternalSignal
 - VME::FPGAUnitV1495Control, [70](#)
- InternalTrigger
 - VME::FPGAUnitV1495Control, [70](#)
- INVALID
 - Socket, [106](#)

- Invalid
 - VME::CAENETControllerV288Status,kEventFIFOStatusRegister
38
- IOModuleV262
 - VME::IOModuleV262, 90
- IOModuleV262Register
 - VME, 18
- IRQ1
 - VME::BridgeVx718, 26
- IRQ2
 - VME::BridgeVx718, 26
- IRQ3
 - VME::BridgeVx718, 26
- IRQ4
 - VME::BridgeVx718, 26
- IRQ5
 - VME::BridgeVx718, 26
- IRQ6
 - VME::BridgeVx718, 26
- IRQ7
 - VME::BridgeVx718, 26
- IRQId
 - VME::BridgeVx718, 26
- Isel
 - NIM::HVModuleN470Values::ChannelStatus,
48
- IsFromWeb
 - Message, 93
- IsOpen
 - FileReader, 59
- IsTrailing
 - VME::TDCEvent, 123
- IsWebSocket
 - Socket, 107
- kBLTEventNumber
 - VME, 19
- kBoardInfo0
 - VME, 18
- kBoardInfo1
 - VME, 18
- kClear
 - VME::FPGAUnitV1495, 61
- kControl
 - VME, 18
- kECLLevelWrite
 - VME, 18
- kEventCounter
 - VME, 19
- kEventFIFO
 - VME, 19
- kEventFIFOStoredRegister
 - VME, 19
- kEventStored
 - VME, 19
- kFirmwareRev
 - VME, 19
- kGeoAddress
 - VME, 19
- kIdentifier
 - VME, 18
- Kill
 - NIM::HVModuleN470Values::ChannelStatus,
48
- kInterruptLevel
 - VME, 19
- kInterruptVector
 - VME, 19
- kMCSTBase
 - VME, 19
- kMCSTControl
 - VME, 19
- kMicroHandshake
 - VME, 19
- kModuleReset
 - VME, 19
- kN470ChannelOff
 - NIM, 12
- kN470ChannelOn
 - NIM, 12
- kN470ClearAlarm
 - NIM, 12
- kN470DisableFrontPanel
 - NIM, 12
- kN470EnableFrontPanel
 - NIM, 12
- kN470GeneralInfo
 - NIM, 12
- kN470I0Value
 - NIM, 12
- kN470I1Value
 - NIM, 12
- kN470KillAllChannels
 - NIM, 12
- kN470MonStatus
 - NIM, 12

- kN470NIMLevel
NIM, [12](#)
- kN470OperationalParams
NIM, [12](#)
- kN470RampDownValue
NIM, [12](#)
- kN470RampUpValue
NIM, [12](#)
- kN470TripValue
NIM, [12](#)
- kN470TTLLevel
NIM, [12](#)
- kN470V0Value
NIM, [12](#)
- kN470V1Value
NIM, [12](#)
- kNIMLevelWrite
VME, [18](#)
- kNIMPulseRead
VME, [18](#)
- kNIMPulseWrite
VME, [18](#)
- kOutputBuffer
VME, [18](#)
- kReset
VME::FPGAUnitV1495, [61](#)
- kROMBoard0
VME, [19](#)
- kROMBoard1
VME, [19](#)
- kROMBoard2
VME, [19](#)
- kROMOui0
VME, [19](#)
- kROMOui1
VME, [19](#)
- kROMOui2
VME, [19](#)
- kROMRevis0
VME, [19](#)
- kROMRevis1
VME, [19](#)
- kROMRevis2
VME, [19](#)
- kROMRevis3
VME, [19](#)
- kROMSerNum0
VME, [19](#)
- kROMSerNum1
VME, [19](#)
- kSoftwareClear
VME, [19](#)
- kStatus
VME, [18](#)
- kTrigger
VME::FPGAUnitV1495, [61](#)
- kV1495Board0
VME, [18](#)
- kV1495Board1
VME, [18](#)
- kV1495Board2
VME, [18](#)
- kV1495ClockSettings
VME, [17](#)
- kV1495ConfigurationROM
VME, [18](#)
- kV1495Control
VME, [17](#)
- kV1495DelaySettings
VME, [17](#)
- kV1495FWRevision
VME, [17](#)
- kV1495GeoAddress
VME, [17](#)
- kV1495HWRevision0
VME, [18](#)
- kV1495HWRevision1
VME, [18](#)
- kV1495HWRevision2
VME, [18](#)
- kV1495HWRevision3
VME, [18](#)
- kV1495ModuleReset
VME, [17](#)
- kV1495OUI0
VME, [18](#)
- kV1495OUI1
VME, [18](#)
- kV1495OUI2
VME, [18](#)
- kV1495OutputSettings
VME, [17](#)
- kV1495ScalerCounter
VME, [17](#)
- kV1495SerNum0
VME, [18](#)
- kV1495SerNum1
VME, [18](#)
- kV1495TDCBoardInterface
VME, [17](#)

- kV1495TriggerSettings
 - VME, [17](#)
- kV1495UserFPGAConfig
 - VME, [17](#)
- kV1495UserFPGAFlashMem
 - VME, [17](#)
- kV1495UserFWRevision
 - VME, [17](#)
- kV288DataBuffer
 - VME, [16](#)
- kV288IRQVector
 - VME, [16](#)
- kV288ModuleReset
 - VME, [16](#)
- kV288Status
 - VME, [16](#)
- kV288Transmission
 - VME, [16](#)
- kV812DeadTimeGroup0
 - VME, [17](#)
- kV812DeadTimeGroup1
 - VME, [17](#)
- kV812FixedCode
 - VME, [17](#)
- kV812Info0
 - VME, [17](#)
- kV812Info1
 - VME, [17](#)
- kV812MajorityThreshold
 - VME, [17](#)
- kV812OutputWidthGroup0
 - VME, [17](#)
- kV812OutputWidthGroup1
 - VME, [17](#)
- kV812PatternOfInhibit
 - VME, [17](#)
- kV812TestPulse
 - VME, [17](#)
- kV812ThresholdChannel0
 - VME, [17](#)
- leading
 - VME::trailead_t, [151](#)
- Listen
 - Socket, [107](#)
- LOAD_DEF_CONFIG
 - VME::TDCV1x900pcodes, [24](#)
- LOAD_USER_CONFIG
 - VME::TDCV1x900pcodes, [24](#)
- magic
 - file_header_t, [56](#)
- MASTER
 - Socket, [106](#)
- MATCH_WIN_WIDTH
 - VME, [20](#)
- MaxV
 - NIM::HVModuleN470ChannelValues, [87](#)
 - NIM::HVModuleN470Values::ChannelStatus, [48](#)
- Message, [92](#)
 - ~Message, [93](#)
 - Dump, [93](#)
 - fString, [93](#)
 - GetKey, [93](#)
 - GetString, [93](#)
 - IsFromWeb, [93](#)
 - Message, [93](#)
- Messenger, [95](#)
 - ~Messenger, [96](#)
 - AddClient, [97](#)
 - Broadcast, [97](#)
 - Connect, [98](#)
 - Disconnect, [98](#)
 - DisconnectClient, [98](#)
 - fNumAttempts, [102](#)
 - fPID, [102](#)
 - fStderrPipe, [102](#)
 - fStdoutPipe, [102](#)
 - fWS, [102](#)
 - GetType, [99](#)
 - Messenger, [96](#)
 - ProcessMessage, [99](#)
 - Receive, [99](#)
 - Send, [100](#)
 - SendAll, [101](#)
 - StartAcquisition, [101](#)
 - StopAcquisition, [101](#)
 - SwitchClientType, [101](#)
- micro_handshake
 - VME, [18](#)
- nchannels
 - VME::TDCV1x90, [143](#)
- NIM, [11](#)
 - HVModuleN470Opcodes, [12](#)
 - kN470ChannelOff, [12](#)
 - kN470ChannelOn, [12](#)
 - kN470ClearAlarm, [12](#)

- kN470DisableFrontPanel, 12
- kN470EnableFrontPanel, 12
- kN470GeneralInfo, 12
- kN470IOValue, 12
- kN470I1Value, 12
- kN470KillAllChannels, 12
- kN470MonStatus, 12
- kN470NIMLevel, 12
- kN470OperationalParams, 12
- kN470RampDownValue, 12
- kN470RampUpValue, 12
- kN470TripValue, 12
- kN470TTLLLevel, 12
- kN470V0Value, 12
- kN470V1Value, 12
- NIM::HVModuleN470Values::ChannelStatus, 46
- NIM::HVModuleN470, 80
 - ~HVModuleN470, 81
 - fAddress, 84
 - fController, 84
 - GetFWRevision, 81
 - GetModuleId, 81
 - HVModuleN470, 81
 - ReadChannelValues, 81
 - ReadMonitoringValues, 81
 - ReadRegister, 82
 - SetChannelI0, 82
 - SetChannelI1, 82
 - SetChannelV0, 83
 - SetChannelV1, 83
 - WriteRegister, 83
- NIM::HVModuleN470ChannelValues, 85
 - ~HVModuleN470ChannelValues, 86
 - ChannelStatus, 86
 - Dump, 86
 - fChannelId, 87
 - fValues, 87
 - HVModuleN470ChannelValues, 86
 - I0, 86
 - I1, 87
 - Imon, 87
 - MaxV, 87
 - RampDown, 87
 - RampUp, 87
 - Trip, 87
 - V0, 87
 - V1, 87
 - Vmon, 87
- NIM::HVModuleN470Values, 88
 - ~HVModuleN470Values, 89
 - Dump, 89
 - fValues, 89
 - GetChannelStatus, 89
 - HVModuleN470Values, 89
 - Imon, 89
 - Vmax, 89
 - Vmon, 89
- NIM::HVModuleN470Values::ChannelStatus, 45
 - ~ChannelStatus, 48
 - Alarm, 48
 - ChannelStatus, 48
 - Dump, 48
 - Enabled, 48
 - fWord, 48
 - HVEnabled, 48
 - Isel, 48
 - Kill, 48
 - MaxV, 48
 - NIM, 46
 - NonCalibrated, 48
 - operator<<, 48
 - OVC, 48
 - OVV, 48
 - Polarity, 48
 - RampDown, 48
 - RampUp, 48
 - SignalStandard, 46
 - Standard, 48
 - Trip, 48
 - TTL, 46
 - UNV, 48
 - Vsel, 48
- NonCalibrated
 - NIM::HVModuleN470Values::ChannelStatus, 48
- num_hptdc
 - file_header_t, 56
- NumErrors
 - VME::TDCMeasurement, 126
- NumEvents
 - VME::TDCMeasurement, 126
- Object
 - SocketMessage, 114
- OLEADING
 - VME, 17
- Open

- FileReader, 59
- OperationStatus
 - VME::CAENETControllerV288Status, 38
- operator<<
 - NIM::HVModuleN470Values::ChannelStatus, 48
 - VME::CAENETControllerV288, 37
 - VME::TDCErrorFlag, 118
- operator>>
 - VME::CAENETControllerV288, 37
- OTRAILING
 - VME, 17
- OutputConf
 - VME::BridgeVx718, 28
- OutputFiles
 - VMEReader, 155
- OutputOff
 - VME::BridgeVx718, 28
- OutputOn
 - VME::BridgeVx718, 28
- OutputWidthCalculator
 - VME::CFDV812, 42
- OVC
 - NIM::HVModuleN470Values::ChannelStatus, 48
- OVV
 - NIM::HVModuleN470Values::ChannelStatus, 48
- PAIR
 - VME, 17
- pair_lead_res
 - VME::TDCV1x90, 143
- pair_width_res
 - VME::TDCV1x90, 143
- PairMode
 - VME::TDCV1x90Status, 150
- ParseMessage
 - Client, 53
- PCIInterfaceA2818
 - VME::PCIInterfaceA2818, 103
- Polarity
 - NIM::HVModuleN470Values::ChannelStatus, 48
- PrepareConnection
 - Socket, 107
- ProcessMessage
 - Messenger, 99
- PulseTDCBits
 - VME::FPGAUnitV1495, 65
 - Purged
 - VME::TDCV1x90Status, 150
 - r100ps
 - VME, 19
 - r200ps
 - VME, 19
 - r25ps
 - VME, 19
 - r800ps
 - VME, 19
 - R_100ps
 - VME::TDCV1x90Status, 149
 - R_200ps
 - VME::TDCV1x90Status, 149
 - R_25ps
 - VME::TDCV1x90Status, 149
 - R_800ps
 - VME::TDCV1x90Status, 149
 - RampDown
 - NIM::HVModuleN470ChannelValues, 87
 - NIM::HVModuleN470Values::ChannelStatus, 48
 - RampUp
 - NIM::HVModuleN470ChannelValues, 87
 - NIM::HVModuleN470Values::ChannelStatus, 48
 - READ_OK
 - VME, 18
 - READ_ACQ_MOD
 - VME::TDCV1x90Opcodes, 24
 - READ_ADJUST_CH
 - VME::TDCV1x90Opcodes, 24
 - READ_DEAD_TIME
 - VME::TDCV1x90Opcodes, 24
 - READ_DETECTION
 - VME::TDCV1x90Opcodes, 24
 - READ_DLL_LOCK
 - VME::TDCV1x90Opcodes, 24
 - READ_EEPROM
 - VME::TDCV1x90Opcodes, 24
 - READ_EN_PATTERN
 - VME::TDCV1x90Opcodes, 24
 - READ_EN_PATTERN32
 - VME::TDCV1x90Opcodes, 24
 - READ_ERROR_STATUS
 - VME::TDCV1x90Opcodes, 24

- READ_ERROR_TYPES
 - VME::TDCV1x90Opcodes, [24](#)
- READ_EVENT_SIZE
 - VME::TDCV1x90Opcodes, [24](#)
- READ_FIFO_SIZE
 - VME::TDCV1x90Opcodes, [24](#)
- READ_GLOB_OFFS
 - VME::TDCV1x90Opcodes, [24](#)
- READ_HEAD_TRAILER
 - VME::TDCV1x90Opcodes, [24](#)
- READ_MICRO_REV
 - VME::TDCV1x90Opcodes, [24](#)
- READ_RC_ADJ
 - VME::TDCV1x90Opcodes, [24](#)
- READ_RES
 - VME::TDCV1x90Opcodes, [24](#)
- READ_SETUP_REG
 - VME::TDCV1x90Opcodes, [24](#)
- READ_SETUP_SCANPATH
 - VME::TDCV1x90Opcodes, [24](#)
- READ_SPARE
 - VME::TDCV1x90Opcodes, [24](#)
- READ_STATUS_STREAM
 - VME::TDCV1x90Opcodes, [24](#)
- READ_TDC_ID
 - VME::TDCV1x90Opcodes, [24](#)
- READ_TRG_CONF
 - VME::TDCV1x90Opcodes, [24](#)
- ReadAcquisitionMode
 - VME::TDCV1x90, [136](#)
- ReadChannelValues
 - NIM::HVModuleN470, [81](#)
- ReadDetectionMode
 - VME::TDCV1x90, [136](#)
- ReadMonitoringValues
 - NIM::HVModuleN470, [81](#)
- ReadRegister
 - NIM::HVModuleN470, [82](#)
 - VME::GenericBoard, [75](#)
- ReadXML
 - VMEReader, [158](#)
- Receive
 - Client, [53](#)
 - Messenger, [99](#)
- REJECT_MARGIN
 - VME, [20](#)
- Reset
 - VME::BridgeVx718, [28](#)
 - VME::CAENETControllerV288, [36](#)
- RESET_DLL_PLL
 - VME::TDCV1x90Opcodes, [24](#)
- ResetFPGA
 - VME::FPGAUnitV1495, [66](#)
- Resolution
 - VME::TDCV1x90Status, [150](#)
- REV_DATE_MICRO_FW
 - VME::TDCV1x90Opcodes, [24](#)
- run_id
 - file_header_t, [56](#)
- SAVE_RC_ADJ
 - VME::TDCV1x90Opcodes, [24](#)
- SAVE_USER_CONFIG
 - VME::TDCV1x90Opcodes, [24](#)
- SelectConnections
 - Socket, [108](#)
- Send
 - Client, [54](#)
 - Messenger, [100](#)
- SendAll
 - Messenger, [101](#)
- SendAndReceive
 - Client, [54](#)
- SendBuffer
 - VME::CAENETControllerV288, [37](#)
- SendClear
 - VMEReader, [159](#)
- SendMessage
 - Socket, [108](#)
- SendPulse
 - VMEReader, [159](#)
- SET_ADJUST_CH
 - VME::TDCV1x90Opcodes, [24](#)
- SET_DEAD_TIME
 - VME::TDCV1x90Opcodes, [24](#)
- SET_DETECTION
 - VME::TDCV1x90Opcodes, [24](#)
- SET_DLL_CLOCK
 - VME::TDCV1x90Opcodes, [24](#)
- SET_ERROR_TYPES
 - VME::TDCV1x90Opcodes, [24](#)
- SET_EVENT_SIZE
 - VME::TDCV1x90Opcodes, [24](#)
- SET_FIFO_SIZE
 - VME::TDCV1x90Opcodes, [24](#)
- SET_GLOB_OFFS
 - VME::TDCV1x90Opcodes, [24](#)
- SET_KEEP_TOKEN
 - VME::TDCV1x90Opcodes, [24](#)
- SET_PAIR_RES

- VME::TDCV1x90OpCodes, [24](#)
- SET_RC_ADJ
 - VME::TDCV1x90OpCodes, [24](#)
- SET_REJ_MARGIN
 - VME::TDCV1x90OpCodes, [24](#)
- SET_SW_MARGIN
 - VME::TDCV1x90OpCodes, [24](#)
- SET_TDC_TSET_OUTPUT
 - VME::TDCV1x90OpCodes, [24](#)
- SET_TR_LEAD_LSB
 - VME::TDCV1x90OpCodes, [24](#)
- SET_WIN_OFFS
 - VME::TDCV1x90OpCodes, [24](#)
- SET_WIN_WIDTH
 - VME::TDCV1x90OpCodes, [24](#)
- SetAcquisitionMode
 - VME::TDCV1x90, [136](#)
- SetAlign64
 - VME::TDCV1x90Control, [145](#)
- SetBit
 - VME::FPGAUnitV1495Control, [72](#)
- SetBLTEventNumberRegister
 - VME::TDCV1x90, [136](#)
- SetBusError
 - VME::TDCV1x90Control, [145](#)
- SetChannelDeadTime
 - VME::TDCV1x90, [137](#)
- SetChannelIO
 - NIM::HVModuleN470, [82](#)
- SetChannelI1
 - NIM::HVModuleN470, [82](#)
- SetChannelV0
 - NIM::HVModuleN470, [83](#)
- SetChannelV1
 - NIM::HVModuleN470, [83](#)
- SetClockSource
 - VME::FPGAUnitV1495Control, [72](#)
- SetCompensation
 - VME::TDCV1x90Control, [146](#)
- SetContinuousStorage
 - VME::TDCV1x90, [137](#)
- SetControl
 - VME::FPGAUnitV1495, [66](#)
 - VME::TDCV1x90, [137](#)
- SetDeadTime
 - VME::CFDV812, [43](#)
- SetDetectionMode
 - VME::TDCV1x90, [137](#)
- SetDLLClock
 - VME::TDCV1x90, [138](#)
- SetEmptyEvent
 - VME::TDCV1x90Control, [146](#)
- SetErrorMarks
 - VME::TDCV1x90, [138](#)
- SetETTT
 - VME::TDCV1x90, [138](#)
 - VME::TDCV1x90Control, [146](#)
- SetEventFIFO
 - VME::TDCV1x90Control, [146](#)
- SetEventsCollection
 - VME::TDCMeasurement, [126](#)
- SetFIFOSize
 - VME::TDCV1x90, [139](#)
- SetGlobalOffset
 - VME::TDCV1x90, [139](#)
- SetInternalClockPeriod
 - VME::FPGAUnitV1495, [66](#)
- SetInternalTriggerPeriod
 - VME::FPGAUnitV1495, [67](#)
- SetIRQ
 - VME::BridgeVx718, [28](#)
- SetKeyValue
 - SocketMessage, [114](#)
- SetLSBTraileadEdge
 - VME::TDCV1x90, [139](#)
- SetMEBAccess
 - VME::TDCV1x90Control, [146](#)
- SetOutputDelay
 - VME::FPGAUnitV1495, [67](#)
- SetOutputFile
 - VMEReader, [159](#)
- SetOutputPulser
 - VME::FPGAUnitV1495, [67](#)
- SetOutputPulserPOI
 - VME::FPGAUnitV1495, [67](#)
- SetOutputWidth
 - VME::CFDV812, [43](#)
- SetPairModeResolution
 - VME::TDCV1x90, [139](#)
- SetPOI
 - VME::CFDV812, [43](#)
- SetPoI
 - VME::TDCV1x90, [139](#)
- SetPort
 - Socket, [108](#)
- SetRCAdjust
 - VME::TDCV1x90, [140](#)
- SetScalerReset
 - VME::FPGAUnitV1495Control, [72](#)
- SetScalerStatus

- VME::FPGAUnitV1495Control, 72
- SetSignalSource
 - VME::FPGAUnitV1495Control, 73
- SetSocketId
 - Socket, 108
- SetSRAMCompensation
 - VME::TDCV1x90Control, 146
- SetStatus
 - VME::TDCV1x90, 140
- SetSWTermination
 - VME::TDCV1x90Control, 146
- SetTDCBits
 - VME::FPGAUnitV1495, 67
- SetTDCEncapsulation
 - VME::TDCV1x90, 140
- SetTermination
 - VME::TDCV1x90Control, 146
- SetTestFIFO
 - VME::TDCV1x90Control, 146
- SetTestMode
 - VME::TDCV1x90, 140
- SetThreshold
 - VME::CFDV812, 43
- SetTriggerMatching
 - VME::TDCV1x90, 141
- SetTriggerSource
 - VME::FPGAUnitV1495Control, 73
- SetVerboseLevel
 - VME::TDCV1x90, 141
- SetWindowOffset
 - VME::TDCV1x90, 141
- SetWindowWidth
 - VME::TDCV1x90, 141
- SetWord
 - VME::TDCEvent, 123
- SignalSource
 - VME::FPGAUnitV1495Control, 70
- SignalStandard
 - NIM::HVModuleN470Values::ChannelStatus, 46
- SinglePulse
 - VME::BridgeVx718, 28
- Socket, 104
 - ~Socket, 106
 - AcceptConnections, 106
 - Bind, 106
 - CLIENT, 106
 - Configure, 107
 - Create, 107
 - DETECTOR, 106
 - DQM, 106
 - DumpConnected, 107
 - fAddress, 109
 - fBuffer, 109
 - FetchMessage, 107
 - fMaster, 109
 - fPort, 109
 - fReadFds, 109
 - fSocketId, 109
 - fSocketsConnected, 109
 - GetPort, 107
 - GetSocketId, 107
 - GetSocketType, 107
 - INVALID, 106
 - IsWebSocket, 107
 - Listen, 107
 - MASTER, 106
 - PrepareConnection, 107
 - SelectConnections, 108
 - SendMessage, 108
 - SetPort, 108
 - SetSocketId, 108
 - Socket, 106
 - SocketCollection, 106
 - SocketType, 106
 - Start, 108
 - Stop, 108
 - WEBSOCKET_CLIENT, 106
- Socket communication objects, 9
- SocketCollection
 - Socket, 106
- SocketMessage, 110
 - ~SocketMessage, 113
 - Dump, 113
 - fMessage, 115
 - GetCleanedValue, 113
 - GetIntValue, 113
 - GetKey, 113
 - GetString, 113
 - GetValue, 114
 - GetVectorValue, 114
 - Object, 114
 - SetKeyValue, 114
 - SocketMessage, 112, 113
 - String, 114
- SocketType
 - Socket, 106
- SoftwareClear
 - VME::TDCV1x90, 142
- SoftwareReset

- VME::TDCV1x90, 142
- spill_id
 - file_header_t, 56
- Standard
 - NIM::HVModuleN470Values::ChannelStatus, 48
- Start
 - Socket, 108
- StartAcquisition
 - Messenger, 101
- StartPulser
 - VME::BridgeVx718, 28
 - VMEReader, 160
- StartScaler
 - VME::FPGAUnitV1495, 68
- Stop
 - Socket, 108
- StopAcquisition
 - Messenger, 101
- StopPulser
 - VME::BridgeVx718, 28
 - VMEReader, 160
- StopScaler
 - VME::FPGAUnitV1495, 68
- String
 - SocketMessage, 114
- SwitchClientType
 - Messenger, 101
- TDCBits
 - VME::FPGAUnitV1495, 61
- TDCCollection
 - VME, 15
- TDCError
 - VME::TDCEvent, 120
- TDCErrorFlag
 - VME::TDCEvent, 117
- TDCEvent
 - VME::TDCEvent, 121
- TDCEventCollection
 - VME, 15
- TDCHeader
 - VME::TDCEvent, 120
- TDCMeasurement
 - VME::TDCEvent, 120
 - VME::TDCMeasurement, 124
- TDCResolution
 - VME::TDCV1x90Status, 149
- TDCTrailer
 - VME::TDCEvent, 120
- TDCV1x90
 - VME::TDCV1x90, 129
- TDCV1x90Control
 - VME::TDCV1x90Control, 145
- TDCV1x90Register
 - VME, 18
- TDCV1x90Status
 - VME::TDCV1x90Status, 149
- TerminationOn
 - VME::TDCV1x90Status, 150
- TestOutputs
 - VME::BridgeVx718, 28
- total_hits
 - VME::trailead_t, 151
- TRIALEAD
 - VME, 17
- trailead_edge_lsb
 - VME, 19
- trailing
 - VME::trailead_t, 151
- TRG_MATCH
 - VME::TDCV1x90Opcodes, 24
- TRIG_MATCH
 - VME, 16
- TRIG_TIME_SUB
 - VME, 20
- trig_conf
 - VME, 19
- TriggerLost
 - VME::TDCV1x90Status, 150
- TriggerMatching
 - VME::TDCV1x90Status, 150
- TriggerSource
 - VME::FPGAUnitV1495Control, 70
- Trip
 - NIM::HVModuleN470ChannelValues, 87
 - NIM::HVModuleN470Values::ChannelStatus, 48
- TTL
 - NIM::HVModuleN470Values::ChannelStatus, 46
- UNV
 - NIM::HVModuleN470Values::ChannelStatus, 48
- UPDATE_SETUP_REG
 - VME::TDCV1x90Opcodes, 24
- UPDATE_SETUP_TDC
 - VME::TDCV1x90Opcodes, 24

-
- UseSocket
 - VMERReader, [160](#)
 - V0
 - NIM::HVModuleN470ChannelValues, [87](#)
 - V1
 - NIM::HVModuleN470ChannelValues, [87](#)
 - Valid
 - VME::CAENETControllerV288Status, [38](#)
 - Vmax
 - NIM::HVModuleN470Values, [89](#)
 - VME, [13](#)
 - AcquisitionMode, [16](#)
 - BridgeType, [16](#)
 - CAEN_V1718, [16](#)
 - CAEN_V2718, [16](#)
 - CAENETControllerV288Answer, [16](#)
 - CAENETControllerV288Register, [16](#)
 - CFDCollection, [15](#)
 - CFDV812Register, [16](#)
 - cnBusy, [16](#)
 - cnIncorrectHCC, [16](#)
 - cnIncorrectValue, [16](#)
 - cnNoData, [16](#)
 - cnSuccess, [16](#)
 - cnUnrecognizedCode, [16](#)
 - cnWrongModuleAddress, [16](#)
 - CONT_STORAGE, [16](#)
 - DetectionMode, [17](#)
 - EXTRA_SEARCH_WIN_WIDTH, [20](#)
 - FPGAUnitV1495Register, [17](#)
 - IOModuleV262Register, [18](#)
 - kBLTEventNumber, [19](#)
 - kBoardInfo0, [18](#)
 - kBoardInfo1, [18](#)
 - kControl, [18](#)
 - kECLLevelWrite, [18](#)
 - kEventCounter, [19](#)
 - kEventFIFO, [19](#)
 - kEventFIFOStatusRegister, [19](#)
 - kEventFIFOStoredRegister, [19](#)
 - kEventStored, [19](#)
 - kFirmwareRev, [19](#)
 - kGeoAddress, [19](#)
 - kIdentifier, [18](#)
 - kInterruptLevel, [19](#)
 - kInterruptVector, [19](#)
 - kMCSTBase, [19](#)
 - kMCSTControl, [19](#)
 - kMicro, [19](#)
 - kMicroHandshake, [19](#)
 - kModuleReset, [19](#)
 - kNIMLevelWrite, [18](#)
 - kNIMPulseRead, [18](#)
 - kNIMPulseWrite, [18](#)
 - kOutputBuffer, [18](#)
 - kROMBoard0, [19](#)
 - kROMBoard1, [19](#)
 - kROMBoard2, [19](#)
 - kROMOui0, [19](#)
 - kROMOui1, [19](#)
 - kROMOui2, [19](#)
 - kROMRevis0, [19](#)
 - kROMRevis1, [19](#)
 - kROMRevis2, [19](#)
 - kROMRevis3, [19](#)
 - kROMSerNum0, [19](#)
 - kROMSerNum1, [19](#)
 - kSoftwareClear, [19](#)
 - kStatus, [18](#)
 - kV1495Board0, [18](#)
 - kV1495Board1, [18](#)
 - kV1495Board2, [18](#)
 - kV1495ClockSettings, [17](#)
 - kV1495ConfigurationROM, [18](#)
 - kV1495Control, [17](#)
 - kV1495DelaySettings, [17](#)
 - kV1495FWRevision, [17](#)
 - kV1495GeoAddress, [17](#)
 - kV1495HWRevision0, [18](#)
 - kV1495HWRevision1, [18](#)
 - kV1495HWRevision2, [18](#)
 - kV1495HWRevision3, [18](#)
 - kV1495ModuleReset, [17](#)
 - kV1495OUI0, [18](#)
 - kV1495OUI1, [18](#)
 - kV1495OUI2, [18](#)
 - kV1495OutputSettings, [17](#)
 - kV1495ScalerCounter, [17](#)
 - kV1495SerNum0, [18](#)
 - kV1495SerNum1, [18](#)
 - kV1495TDCBoardInterface, [17](#)
 - kV1495TriggerSettings, [17](#)
 - kV1495UserFPGAConfig, [17](#)
-

- kV1495UserFPGAFlashMem, [17](#)
- kV1495UserFWRevision, [17](#)
- kV288DataBuffer, [16](#)
- kV288IRQVector, [16](#)
- kV288ModuleReset, [16](#)
- kV288Status, [16](#)
- kV288Transmission, [16](#)
- kV812DeadTimeGroup0, [17](#)
- kV812DeadTimeGroup1, [17](#)
- kV812FixedCode, [17](#)
- kV812Info0, [17](#)
- kV812Info1, [17](#)
- kV812MajorityThreshold, [17](#)
- kV812OutputWidthGroup0, [17](#)
- kV812OutputWidthGroup1, [17](#)
- kV812PatternOfInhibit, [17](#)
- kV812TestPulse, [17](#)
- kV812ThresholdChannel0, [17](#)
- MATCH_WIN_WIDTH, [20](#)
- micro_handshake, [18](#)
- OLEADING, [17](#)
- OTRAILING, [17](#)
- PAIR, [17](#)
- r100ps, [19](#)
- r200ps, [19](#)
- r25ps, [19](#)
- r800ps, [19](#)
- READ_OK, [18](#)
- REJECT_MARGIN, [20](#)
- TDCCollection, [15](#)
- TDCEventCollection, [15](#)
- TDCV1x90Register, [18](#)
- TRAILHEAD, [17](#)
- trailead_edge_lsb, [19](#)
- TRIG_MATCH, [16](#)
- TRIG_TIME_SUB, [20](#)
- trig_conf, [19](#)
- WIN_OFFSET, [20](#)
- WRITE_OK, [18](#)
- VME::BridgeVx718, [25](#)
- ~BridgeVx718, [27](#)
- BridgeVx718, [27](#)
- CheckConfiguration, [27](#)
- CheckPCIInterface, [27](#)
- fHasIRQ, [28](#)
- GetHandle, [27](#)
- GetIRQStatus, [27](#)
- GetStatus, [28](#)
- InputConf, [28](#)
- InputRead, [28](#)
- IRQ1, [26](#)
- IRQ2, [26](#)
- IRQ3, [26](#)
- IRQ4, [26](#)
- IRQ5, [26](#)
- IRQ6, [26](#)
- IRQ7, [26](#)
- IRQId, [26](#)
- OutputConf, [28](#)
- OutputOff, [28](#)
- OutputOn, [28](#)
- Reset, [28](#)
- SetIRQ, [28](#)
- SinglePulse, [28](#)
- StartPulser, [28](#)
- StopPulser, [28](#)
- TestOutputs, [28](#)
- WaitIRQ, [28](#)
- VME::BridgeVx718Control, [30](#)
- ~BridgeVx718Control, [30](#)
- BridgeVx718Control, [30](#)
- fWord, [32](#)
- GetAddressIncrement, [30](#)
- GetArbiterType, [31](#)
- GetBusReqLevel, [31](#)
- GetBusTimeout, [31](#)
- GetInterruptReq, [31](#)
- GetReleaseType, [31](#)
- GetRequesterType, [31](#)
- GetSysRes, [31](#)
- VME::BridgeVx718Status, [33](#)
- ~BridgeVx718Status, [33](#)
- BridgeVx718Status, [33](#)
- Dump, [33](#)
- fWord, [34](#)
- GetBERR, [33](#)
- GetDipSwitch, [34](#)
- GetDTACK, [34](#)
- GetSystemControl, [34](#)
- GetSystemReset, [34](#)
- GetUSBType, [34](#)
- VME::CAENETControllerV288, [35](#)
- ~CAENETControllerV288, [36](#)
- CAENETControllerV288, [36](#)
- FetchBuffer, [36](#)
- GetStatus, [36](#)
- operator<<, [37](#)
- operator>>, [37](#)
- Reset, [36](#)
- SendBuffer, [37](#)

- WaitForResponse, 37
- VME::CAENETControllerV288Status, 38
 - ~CAENETControllerV288Status, 39
 - CAENETControllerV288Status, 39
 - fWord, 39
 - GetOperationStatus, 39
 - Invalid, 38
 - OperationStatus, 38
 - Valid, 38
- VME::CFDV812, 40
 - ~CFDV812, 41
 - CFDV812, 41
 - CheckConfiguration, 41
 - DeadTimeCalculator, 41
 - GetFixedCode, 42
 - GetManufacturerId, 42
 - GetModuleType, 42
 - GetModuleVersion, 42
 - GetSerialNumber, 42
 - OutputWidthCalculator, 42
 - SetDeadTime, 43
 - SetOutputWidth, 43
 - SetPOI, 43
 - SetThreshold, 43
- VME::FPGAUnitV1495, 60
 - ~FPGAUnitV1495, 62
 - CheckBoardVersion, 62
 - ClearOutputPulser, 62
 - DumpFWInformation, 62
 - FPGAUnitV1495, 62
 - fScalerStarted, 68
 - GetCAENFirmwareRevision, 63
 - GetControl, 63
 - GetGeoAddress, 63
 - GetHardwareRevision, 63
 - GetInternalClockPeriod, 64
 - GetInternalTriggerPeriod, 64
 - GetOutputDelay, 64
 - GetOutputPulser, 64
 - GetScalerValue, 65
 - GetSerialNumber, 65
 - GetTDCBits, 65
 - GetUserFirmwareRevision, 65
 - kClear, 61
 - kReset, 61
 - kTrigger, 61
 - PulseTDCBits, 65
 - ResetFPGA, 66
 - SetControl, 66
 - SetInternalClockPeriod, 66
 - SetInternalTriggerPeriod, 67
 - SetOutputDelay, 67
 - SetOutputPulser, 67
 - SetOutputPulserPOI, 67
 - SetTDCBits, 67
 - StartScaler, 68
 - StopScaler, 68
 - TDCBits, 61
- VME::FPGAUnitV1495Control, 69
 - ~FPGAUnitV1495Control, 71
 - ClockSource, 70
 - Dump, 71
 - ExternalClock, 70
 - ExternalSignal, 70
 - ExternalTrigger, 70
 - FPGAUnitV1495Control, 71
 - fWord, 73
 - GetBit, 71
 - GetClockSource, 71
 - GetScalerStatus, 71
 - GetSignalSource, 71
 - GetTriggerSource, 71
 - GetWord, 72
 - InternalClock, 70
 - InternalSignal, 70
 - InternalTrigger, 70
 - SetBit, 72
 - SetClockSource, 72
 - SetScalerReset, 72
 - SetScalerStatus, 72
 - SetSignalSource, 73
 - SetTriggerSource, 73
 - SignalSource, 70
 - TriggerSource, 70
- VME::GenericBoard, 74
 - ~GenericBoard, 75
 - fBaseAddr, 76
 - fHandle, 76
 - GenericBoard, 75
 - ReadRegister, 75
 - WriteRegister, 75
- VME::GlobalOffset, 77
 - coarse, 77
 - fine, 77
- VME::IOModuleV262, 90
 - ~IOModuleV262, 90
 - GetIdentifier, 90
 - GetManufacturerId, 90

- GetModuleType, 91
- GetModuleVersion, 91
- GetSerialNumber, 91
- IOModuleV262, 90
- VME::PCIInterfaceA2818, 103
- ~PCIInterfaceA2818, 103
- fHandle, 103
- GetFWRevision, 103
- PCIInterfaceA2818, 103
- VME::TDCErrorFlag, 116
- ~TDCErrorFlag, 117
- Dump, 117
- fWord, 118
- GetWord, 117
- HasGroupError, 117
- HasInternalChipError, 117
- HasL1BufferOverflow, 117
- HasReachedEventSizeLimit, 117
- HasReadoutFIFOOverflow, 117
- HasTriggerFIFOOverflow, 117
- operator<=, 118
- TDCErrorFlag, 117
- VME::TDCEvent, 119
- ~TDCEvent, 121
- Dump, 121
- ETTT, 120
- EventType, 120
- Filler, 120
- fWord, 123
- GetBunchId, 121
- GetChannelId, 121
- GetErrorFlags, 121
- GetETTT, 121
- GetEventCount, 121
- GetEventId, 122
- GetGeo, 122
- GetStatus, 122
- GetTDCId, 122
- GetTime, 122
- GetType, 122
- GetWidth, 122
- GetWord, 123
- GetWordCount, 123
- GlobalHeader, 120
- GlobalTrailer, 120
- IsTrailing, 123
- SetWord, 123
- TDCError, 120
- TDCEvent, 121
- TDCHeader, 120
- TDCMeasurement, 120
- TDCTrailer, 120
- VME::TDCMeasurement, 124
- ~TDCMeasurement, 124
- Dump, 125
- fEvents, 126
- fMap, 126
- GetBunchId, 125
- GetChannelId, 125
- GetETTT, 125
- GetEventId, 125
- GetLeadingTime, 125
- GetTDCId, 125
- GetToT, 125
- GetTrailingTime, 125
- NumErrors, 126
- NumEvents, 126
- SetEventsCollection, 126
- TDCMeasurement, 124
- VME::TDCV1x90, 127
- ~TDCV1x90, 129
- abort, 130
- CheckConfiguration, 130
- DisableChannel, 130
- DLL_Direct_LowRes, 129
- DLL_PLL_HighRes, 129
- DLL_PLL_LowRes, 129
- DLL_PLL_MedRes, 129
- DLLMode, 129
- EnableChannel, 130
- fAcquisitionMode, 143
- fBuffer, 143
- fDetectionMode, 143
- fErrorMarks, 143
- FetchEvents, 130
- fVerb, 143
- fWindowWidth, 143
- gEnd, 143
- GetAcquisitionMode, 130
- GetBLTEventNumberRegister, 131
- GetChannelDeadTime, 131
- GetControl, 131
- GetDetectionMode, 131
- GetDLLClock, 132
- GetErrorMarks, 132
- GetETTT, 132
- GetEventCounter, 132
- GetEventStored, 132
- GetFIFOSize, 132
- GetFirmwareRevision, 133

- GetGlobalOffset, [133](#)
- GetModel, [133](#)
- GetOUI, [133](#)
- GetPoI, [134](#)
- GetRCAdjust, [134](#)
- GetResolution, [134](#)
- GetSerialNumber, [134](#)
- GetStatus, [134](#)
- GetTDCEncapsulation, [135](#)
- GetTestMode, [135](#)
- GetTriggerConfiguration, [135](#)
- GetWindowOffset, [135](#)
- GetWindowWidth, [136](#)
- HardwareReset, [136](#)
- nchannels, [143](#)
- pair_lead_res, [143](#)
- pair_width_res, [143](#)
- ReadAcquisitionMode, [136](#)
- ReadDetectionMode, [136](#)
- SetAcquisitionMode, [136](#)
- SetBLTEventNumberRegister, [136](#)
- SetChannelDeadTime, [137](#)
- SetContinuousStorage, [137](#)
- SetControl, [137](#)
- SetDetectionMode, [137](#)
- SetDLLClock, [138](#)
- SetErrorMarks, [138](#)
- SetETTT, [138](#)
- SetFIFOSize, [139](#)
- SetGlobalOffset, [139](#)
- SetLSBTrailEdge, [139](#)
- SetPairModeResolution, [139](#)
- SetPoI, [139](#)
- SetRCAdjust, [140](#)
- SetStatus, [140](#)
- SetTDCEncapsulation, [140](#)
- SetTestMode, [140](#)
- SetTriggerMatching, [141](#)
- SetVerboseLevel, [141](#)
- SetWindowOffset, [141](#)
- SetWindowWidth, [141](#)
- SoftwareClear, [142](#)
- SoftwareReset, [142](#)
- TDCV1x90, [129](#)
- WaitMicro, [142](#)
- VME::TDCV1x90Control, [144](#)
 - ~TDCV1x90Control, [145](#)
 - Dump, [145](#)
 - fWord, [147](#)
 - GetAlign64, [145](#)
 - GetBusError, [145](#)
 - GetCompensation, [145](#)
 - GetEmptyEvent, [145](#)
 - GetETTT, [145](#)
 - GetEventFIFO, [145](#)
 - GetMEBAccess, [145](#)
 - GetSRAMCompensation, [145](#)
 - GetSWTermination, [145](#)
 - GetTermination, [145](#)
 - GetTestFIFO, [145](#)
 - GetValue, [145](#)
 - SetAlign64, [145](#)
 - SetBusError, [145](#)
 - SetCompensation, [146](#)
 - SetEmptyEvent, [146](#)
 - SetETTT, [146](#)
 - SetEventFIFO, [146](#)
 - SetMEBAccess, [146](#)
 - SetSRAMCompensation, [146](#)
 - SetSWTermination, [146](#)
 - SetTermination, [146](#)
 - SetTestFIFO, [146](#)
 - TDCV1x90Control, [145](#)
- VME::TDCV1x90Opcodes, [21](#)
 - AUTOLOAD_DEF_CONF, [24](#)
 - AUTOLOAD_USER_CONF, [24](#)
 - CLEAR_KEEP_TOKEN, [24](#)
 - CONT_STOR, [24](#)
 - DEFAULT_SETUP_REG, [24](#)
 - DIS_ALL_CHANNEL, [24](#)
 - DIS_CHANNEL, [24](#)
 - DIS_ERROR_BYPASS, [24](#)
 - DIS_ERROR_MARK, [24](#)
 - DIS_HEAD_TRAILER, [24](#)
 - DIS_SUB_TRG, [24](#)
 - DISABLE_TEST_MODE, [24](#)
 - EN_ALL_CHANNEL, [24](#)
 - EN_CHANNEL, [24](#)
 - EN_ERROR_BYPASS, [24](#)
 - EN_ERROR_MARK, [24](#)
 - EN_HEAD_TRAILER, [24](#)
 - EN_SUB_TRG, [24](#)
 - ENABLE_TEST_MODE, [24](#)
 - LOAD_DEF_CONFIG, [24](#)
 - LOAD_USER_CONFIG, [24](#)
 - READ_ACQ_MOD, [24](#)
 - READ_ADJUST_CH, [24](#)
 - READ_DEAD_TIME, [24](#)
 - READ_DETECTION, [24](#)
 - READ_DLL_LOCK, [24](#)

- READ_EEPROM, [24](#)
- READ_EN_PATTERN, [24](#)
- READ_EN_PATTERN32, [24](#)
- READ_ERROR_STATUS, [24](#)
- READ_ERROR_TYPES, [24](#)
- READ_EVENT_SIZE, [24](#)
- READ_FIFO_SIZE, [24](#)
- READ_GLOB_OFFSETS, [24](#)
- READ_HEAD_TRAILER, [24](#)
- READ_MICRO_REV, [24](#)
- READ_RC_ADJ, [24](#)
- READ_RES, [24](#)
- READ_SETUP_REG, [24](#)
- READ_SETUP_SCANPATH, [24](#)
- READ_SPARE, [24](#)
- READ_STATUS_STREAM, [24](#)
- READ_TDC_ID, [24](#)
- READ_TRG_CONF, [24](#)
- RESET_DLL_PLL, [24](#)
- REV_DATE_MICRO_FW, [24](#)
- SAVE_RC_ADJ, [24](#)
- SAVE_USER_CONFIG, [24](#)
- SET_ADJUST_CH, [24](#)
- SET_DEAD_TIME, [24](#)
- SET_DETECTION, [24](#)
- SET_DLL_CLOCK, [24](#)
- SET_ERROR_TYPES, [24](#)
- SET_EVENT_SIZE, [24](#)
- SET_FIFO_SIZE, [24](#)
- SET_GLOB_OFFSETS, [24](#)
- SET_KEEP_TOKEN, [24](#)
- SET_PAIR_RES, [24](#)
- SET_RC_ADJ, [24](#)
- SET_REJ_MARGIN, [24](#)
- SET_SW_MARGIN, [24](#)
- SET_TDC_TSET_OUTPUT, [24](#)
- SET_TR_LEAD_LSB, [24](#)
- SET_WIN_OFFSETS, [24](#)
- SET_WIN_WIDTH, [24](#)
- TRG_MATCH, [24](#)
- UPDATE_SETUP_REG, [24](#)
- UPDATE_SETUP_TDC, [24](#)
- WRITE_EEPROM, [24](#)
- WRITE_EN_PATTERN, [24](#)
- WRITE_EN_PATTERN32, [24](#)
- WRITE_SETUP_REG, [24](#)
- WRITE_SPARE, [24](#)
- VME::TDCV1x90Status, [148](#)
- ~TDCV1x90Status, [149](#)
- AlmostFull, [149](#)
- BusError, [149](#)
- DataReady, [149](#)
- Dump, [149](#)
- Error, [149](#)
- Full, [150](#)
- fWord, [150](#)
- GetValue, [150](#)
- HeadersEnabled, [150](#)
- PairMode, [150](#)
- Purged, [150](#)
- R_100ps, [149](#)
- R_200ps, [149](#)
- R_25ps, [149](#)
- R_800ps, [149](#)
- Resolution, [150](#)
- TDCResolution, [149](#)
- TDCV1x90Status, [149](#)
- TerminationOn, [150](#)
- TriggerLost, [150](#)
- TriggerMatching, [150](#)
- VME::trailead_t, [151](#)
- ettt, [151](#)
- event_count, [151](#)
- leading, [151](#)
- total_hits, [151](#)
- trailing, [151](#)
- VMEReader, [152](#)
- ~VMEReader, [155](#)
- Abort, [156](#)
- AddCFD, [156](#)
- AddFPGAUnit, [156](#)
- AddHVModule, [156](#)
- AddIOModule, [157](#)
- AddTDC, [157](#)
- fBridge, [160](#)
- fCAENET, [160](#)
- fCFDCollection, [160](#)
- fFPGA, [160](#)
- fHV, [161](#)
- fIsPulserStarted, [161](#)
- fOnSocket, [161](#)
- fOutputFiles, [161](#)
- fSG, [161](#)
- fTDCCollection, [161](#)
- GetCFD, [157](#)
- GetCFDCollection, [157](#)
- GetFPGAUnit, [157](#)
- GetHVModule, [157](#)
- GetIOModule, [158](#)
- GetNumCFD, [158](#)

- GetNumTDC, [158](#)
- GetOutputFile, [158](#)
- GetRunNumber, [158](#)
- GetTDC, [158](#)
- GetTDCCollection, [158](#)
- OutputFiles, [155](#)
- ReadXML, [158](#)
- SendClear, [159](#)
- SendPulse, [159](#)
- SetOutputFile, [159](#)
- StartPulser, [160](#)
- StopPulser, [160](#)
- UseSocket, [160](#)
- VMEReader, [155](#)
- Vmon
 - NIM::HVModuleN470ChannelValues, [87](#)
 - NIM::HVModuleN470Values, [89](#)
- Vsel
 - NIM::HVModuleN470Values::ChannelStatus, [48](#)
- WaitForResponse
 - VME::CAENETControllerV288, [37](#)
- WaitIRQ
 - VME::BridgeVx718, [28](#)
- WaitMicro
 - VME::TDCV1x90, [142](#)
- WEBSOCKET_CLIENT
 - Socket, [106](#)
- WIN_OFFSET
 - VME, [20](#)
- WRITE_OK
 - VME, [18](#)
- WRITE_EEPROM
 - VME::TDCV1x90OpCodes, [24](#)
- WRITE_EN_PATTERN
 - VME::TDCV1x90OpCodes, [24](#)
- WRITE_EN_PATTERN32
 - VME::TDCV1x90OpCodes, [24](#)
- WRITE_SETUP_REG
 - VME::TDCV1x90OpCodes, [24](#)
- WRITE_SPARE
 - VME::TDCV1x90OpCodes, [24](#)
- WriteRegister
 - NIM::HVModuleN470, [83](#)
 - VME::GenericBoard, [75](#)