TCNOpen TRDP Light V2.0.3

Generated by Doxygen 1.8.13

Contents

1	The	TRDP Light Library API Specification 1						
	1.1	General Information	1					
		1.1.1 Purpose	1					
		1.1.2 Scope	1					
		1.1.3 Related documents	1					
		1.1.4 Abbreviations and Definitions	1					
	1.2	Terminology	2					
	1.3	Use Cases	2					
	1.4	Conventions of the API	5					
2	Data	a Structure Index	7					
	2.1	Data Structures	7					
3	File	Index	9					
	3.1	File List	9					
4	Data	a Structure Documentation	13					
	4.1	DNS_HEADER Struct Reference	13					
		4.1.1 Detailed Description	13					
	4.2	GNU_PACKED Struct Reference	13					
		4.2.1 Detailed Description	21					
		4.2.2 Field Documentation	22					
		4.2.2.1 callBack	22					
		4.2.2.2 comld	22					
		4.2.2.3 confVehCnt	23					

ii CONTENTS

4.2.2.4	confVehList	23
4.2.2.5	cstList	23
4.2.2.6	cstUUID	23
4.2.2.7	datasetLength	23
4.2.2.8	defQos	24
4.2.2.9	defTtl	24
4.2.2.10	destAddr	24
4.2.2.11	deviceName	24
4.2.2.12	etbld	24
4.2.2.13	etbTopoCnt	24
4.2.2.14	filterAddr	25
4.2.2.15	inhibit	25
4.2.2.16	isLead	25
4.2.2.17	leadDir	25
4.2.2.18	leadVehOfCst	25
4.2.2.19	lifesign	25
4.2.2.20	msgType	26
4.2.2.21	numCrcErr	26
4.2.2.22	numMissed	26
4.2.2.23	numProtErr	26
4.2.2.24	numRcv	26
4.2.2.25	numRecv	26
4.2.2.26	numSend	27
4.2.2.27	numTopoErr	27
4.2.2.28	opCstList	27
4.2.2.29	opTrnDirState	27
4.2.2.30	opTrnTopoCnt	27
4.2.2.31	opVehList	28
4.2.2.32	ownOpCstNo	28
4.2.2.33	protocolVersion	28

CONTENTS

	4.2.2.34 reservedur [1/2]	20
	4.2.2.35 reserved01 [2/2]	28
	4.2.2.36 reserved02 [1/2]	28
	4.2.2.37 reserved02 [2/2]	29
	4.2.2.38 reserved03	29
	4.2.2.39 reserved04	29
	4.2.2.40 reserved06	29
	4.2.2.41 safetyTrail	29
	4.2.2.42 serviceEntry	29
	4.2.2.43 timeout	30
	4.2.2.44 toBehav	30
	4.2.2.45 trnCstNo	30
	4.2.2.46 trnDirState	30
	4.2.2.47 trnld	30
	4.2.2.48 trnNetDir	30
	4.2.2.49 trnOperator	31
	4.2.2.50 trnTopoCnt	31
	4.2.2.51 trnVehNo	31
	4.2.2.52 vehld	31
	4.2.2.53 vehOrient	31
	4.2.2.54 version	32
4.3	hp_slot Struct Reference	32
	4.3.1 Detailed Description	33
4.4	hp_slots Struct Reference	33
	4.4.1 Detailed Description	34
4.5	PD_ELE Struct Reference	34
	4.5.1 Detailed Description	35
	4.5.2 Field Documentation	35
	4.5.2.1 pFrame	36
4.6	service_info Struct Reference	36

iv CONTENTS

	4.6.1	Detailed Description	36
	4.6.2	Field Documentation	37
		4.6.2.1 fctDev	37
4.7	srv_inf	o_req Struct Reference	37
	4.7.1	Detailed Description	37
4.8	TAU_N	MARSHALL_INFO_T Struct Reference	38
	4.8.1	Detailed Description	38
4.9	TCN_U	JRI Struct Reference	38
	4.9.1	Detailed Description	39
4.10	TRDP_	CLTR_CST_INFO_T Struct Reference	39
	4.10.1	Detailed Description	39
4.11	TRDP_	COM_PARAM_T Struct Reference	39
	4.11.1	Detailed Description	40
4.12	TRDP_	COMID_DSID_MAP_T Struct Reference	40
	4.12.1	Detailed Description	40
4.13	TRDP_	CONSIST_INFO_T Struct Reference	41
	4.13.1	Detailed Description	42
	4.13.2	Field Documentation	42
		4.13.2.1 cstld	42
		4.13.2.2 cstOwner	42
4.14	TRDP_	_DATASET Struct Reference	43
	4.14.1	Detailed Description	43
4.15	TRDP_	DATASET_ELEMENT_T Struct Reference	44
	4.15.1	Detailed Description	44
4.16	TRDP_	_DBG_CONFIG_T Struct Reference	45
	4.16.1	Detailed Description	45
4.17	TRDP_	_DNS_REPLY Struct Reference	45
	4.17.1	Detailed Description	46
	4.17.2	Field Documentation	46
		4.17.2.1 tcnUriCnt	46

CONTENTS

4.18 TRDP_DNS_REQUEST Struct Reference	47
4.18.1 Detailed Description	47
4.18.2 Field Documentation	47
4.18.2.1 tcnUriCnt	48
4.19 TRDP_ETB_INFO_T Struct Reference	48
4.19.1 Detailed Description	48
4.19.2 Field Documentation	48
4.19.2.1 cnCnt	48
4.20 TRDP_FUNCTION_INFO_T Struct Reference	49
4.20.1 Detailed Description	49
4.20.2 Field Documentation	49
4.20.2.1 cnld	49
4.20.2.2 cstVehNo	49
4.20.2.3 etbld	50
4.20.2.4 fctld	50
4.21 TRDP_HANDLE Struct Reference	50
4.21.1 Detailed Description	51
4.22 TRDP_IDX_TABLE_T Struct Reference	51
4.22.1 Detailed Description	51
4.22.2 Field Documentation	51
4.22.2.1 maxNoOfExtPublishers	52
4.22.2.2 maxNoOfHighCatPublishers	52
4.22.2.3 maxNoOfHighCatSubscriptions	52
4.22.2.4 maxNoOfLowCatPublishers	52
4.22.2.5 maxNoOfLowCatSubscriptions	52
4.22.2.6 maxNoOfMidCatPublishers	52
4.22.2.7 maxNoOfMidCatSubscriptions	53
4.23 TRDP_MARSHALL_CONFIG_T Struct Reference	53
4.23.1 Detailed Description	53
4.24 TRDP_MD_CONFIG_T Struct Reference	54

vi

	4.24.1 Detailed Description	54
4.25	TRDP_MD_INFO_T Struct Reference	55
	4.25.1 Detailed Description	56
4.26	TRDP_MEM_CONFIG_T Struct Reference	56
	4.26.1 Detailed Description	56
4.27	TRDP_PD_CONFIG_T Struct Reference	57
	4.27.1 Detailed Description	57
4.28	TRDP_PD_INFO_T Struct Reference	57
	4.28.1 Detailed Description	58
4.29	TRDP_PROCESS_CONFIG_T Struct Reference	58
	4.29.1 Detailed Description	59
4.30	TRDP_PROP_T Struct Reference	59
	4.30.1 Detailed Description	59
4.31	TRDP_SDT_PAR_T Struct Reference	60
	4.31.1 Detailed Description	60
4.32	TRDP_SEQ_CNT_ENTRY_T Struct Reference	60
	4.32.1 Detailed Description	61
4.33	TRDP_SESSION Struct Reference	61
	4.33.1 Detailed Description	62
4.34	TRDP_SOCKET_TCP Struct Reference	62
	4.34.1 Detailed Description	63
4.35	TRDP_SOCKETS Struct Reference	63
	4.35.1 Detailed Description	64
	4.35.2 Field Documentation	64
	4.35.2.1 usage	64
4.36	TRDP_VEHICLE_INFO_T Struct Reference	65
	4.36.1 Detailed Description	65
	4.36.2 Field Documentation	65
	4.36.2.1 vehld	66
4.37	TRDP_XML_DOC_HANDLE_T Struct Reference	66
	4.37.1 Detailed Description	66
4.38	VOS_SOCK_OPT_T Struct Reference	66
	4.38.1 Detailed Description	67
4.39	VOS_VERSION_T Struct Reference	67
	4.39.1 Detailed Description	67

CONTENTS vii

5	File	Docum	entation		69
	5.1	iec613	75-2-3.h Fi	le Reference	69
		5.1.1	Detailed I	Description	73
		5.1.2	Macro De	efinition Documentation	73
			5.1.2.1	ETB_CTRL_COMID	73
			5.1.2.2	TRDP_ETBCTRL_DSID	74
			5.1.2.3	TRDP_MAX_FILE_NAME_LEN	74
			5.1.2.4	TRDP_MAX_LABEL_LEN	74
			5.1.2.5	TRDP_MAX_MD_DATA_SIZE	74
			5.1.2.6	TRDP_MAX_URI_HOST_LEN	74
			5.1.2.7	TRDP_MAX_URI_LEN	74
			5.1.2.8	TRDP_MAX_URI_USER_LEN	75
			5.1.2.9	TRDP_MD_DEFAULT_REPLY_TIMEOUT	75
			5.1.2.10	TRDP_MD_INFINITE_TIME	75
			5.1.2.11	TRDP_MIN_PD_HEADER_SIZE	75
			5.1.2.12	TRDP_MSG_PD	75
			5.1.2.13	TRDP_PD_UDP_PORT	75
			5.1.2.14	TRDP_PROCESS_DEFAULT_CYCLE_TIME	76
			5.1.2.15	TRDP_USR_URI_SIZE	76
			5.1.2.16	TTDB_NET_DIR_REQ_COMID	76
			5.1.2.17	TTDB_OP_DIR_INFO_COMID	76
			5.1.2.18	TTDB_STAT_CST_REQ_COMID	76
			5.1.2.19	TTDB_TRN_DIR_REQ_COMID	76
	5.2	tau_cs	tinfo.c File	Reference	77
		5.2.1	Detailed I	Description	78
		5.2.2	Function	Documentation	78
			5.2.2.1	cstInfoGetPropSize()	78
	5.3	tau_ctr	l.c File Ref	erence	79
		5.3.1	Detailed I	Description	80
		5.3.2	Function	Documentation	81

viii CONTENTS

		5.3.2.1	tau_getEcspStat()	 81
		5.3.2.2	tau_initEcspCtrl()	 81
		5.3.2.3	tau_requestEcspConfirm()	 82
		5.3.2.4	tau_setEcspCtrl()	 82
		5.3.2.5	tau_terminateEcspCtrl()	 83
5.4	tau_ctr	rl.h File Re	eference	 83
	5.4.1	Detailed	Description	 85
	5.4.2	Function	n Documentation	 86
		5.4.2.1	tau_getEcspStat()	 86
		5.4.2.2	tau_initEcspCtrl()	 86
		5.4.2.3	tau_requestEcspConfirm()	 87
		5.4.2.4	tau_setEcspCtrl()	 87
		5.4.2.5	tau_terminateEcspCtrl()	 88
5.5	tau_ctr	rl_types.h	File Reference	 88
	5.5.1	Detailed	Description	 90
5.6	tau_dn	nr.c File Re	eference	 91
	5.6.1	Detailed	Description	 92
	5.6.2	Function	n Documentation	 93
		5.6.2.1	tau_addr2Uri()	 93
		5.6.2.2	tau_deInitDnr()	 93
		5.6.2.3	tau_DNRstatus()	 94
		5.6.2.4	tau_getOwnAddr()	 94
		5.6.2.5	tau_initDnr()	 94
		5.6.2.6	tau_uri2Addr()	 95
5.7	tau_dn	nr.h File Re	eference	 96
	5.7.1	Detailed	Description	 98
	5.7.2	Enumera	ation Type Documentation	 98
		5.7.2.1	TRDP_DNR_OPTS	 98
	5.7.3	Function	n Documentation	 98
		5.7.3.1	tau_addr2Uri()	 98

CONTENTS

		5.7.3.2	tau_deInitDnr()	99
		5.7.3.3	tau_DNRstatus()	100
		5.7.3.4	tau_getOwnAddr()	100
		5.7.3.5	tau_initDnr()	101
		5.7.3.6	tau_uri2Addr()	102
5.8	tau_dn	r_types.h	File Reference	103
	5.8.1	Detailed	Description	104
5.9	tau_ma	arshall.c Fi	le Reference	105
	5.9.1	Detailed	Description	106
	5.9.2	Function	Documentation	106
		5.9.2.1	tau_calcDatasetSize()	106
		5.9.2.2	tau_calcDatasetSizeByComId()	107
		5.9.2.3	tau_initMarshall()	108
		5.9.2.4	tau_marshall()	108
		5.9.2.5	tau_marshallDs()	109
		5.9.2.6	tau_unmarshall()	110
		5.9.2.7	tau_unmarshallDs()	110
5.10	tau_ma	arshall.h Fi	ile Reference	111
	5.10.1	Detailed	Description	113
	5.10.2	Function	Documentation	113
		5.10.2.1	tau_calcDatasetSize()	113
		5.10.2.2	tau_calcDatasetSizeByComId()	114
		5.10.2.3	tau_initMarshall()	115
		5.10.2.4	tau_marshall()	116
		5.10.2.5	tau_marshallDs()	117
		5.10.2.6	tau_unmarshall()	118
		5.10.2.7	tau_unmarshallDs()	120
5.11	tau_so	_if.c File R	Reference	121
	5.11.1	Detailed	Description	123
	5.11.2	Function	Documentation	123

CONTENTS

	5.11.2.1	tau_addService()	. 123
	5.11.2.2	tau_delService()	. 124
	5.11.2.3	tau_freeServicesList()	. 124
	5.11.2.4	tau_getServicesList()	. 124
	5.11.2.5	tau_updService()	. 125
5.12 tau	_so_if.h File F	Reference	. 126
5.12	2.1 Detailed	Description	. 127
5.12	2.2 Function	Documentation	. 128
	5.12.2.1	tau_addService()	. 128
	5.12.2.2	tau_delService()	. 128
	5.12.2.3	tau_freeServicesList()	. 129
	5.12.2.4	tau_getServicesList()	. 129
	5.12.2.5	tau_updService()	. 130
5.13 tau_	_tti.c File Refe	erence	. 130
5.13	3.1 Detailed	Description	. 132
5.10	3.2 Macro Do	efinition Documentation	. 133
	5.13.2.1	TTI_CACHED_CONSISTS	. 133
5.10	3.3 Function	Documentation	. 133
	5.13.3.1	tau_deInitTTI()	. 133
	5.13.3.2	tau_getCstFctCnt()	. 133
	5.13.3.3	tau_getCstFctInfo()	. 134
	5.13.3.4	tau_getCstInfo()	. 134
	5.13.3.5	tau_getCstVehCnt()	. 135
	5.13.3.6	tau_getOpTrDirectory()	. 135
	5.13.3.7	tau_getOpTrnDirectoryStatusInfo()	. 136
	5.13.3.8	tau_getOwnlds()	. 136
	5.13.3.9	tau_getOwnOpCstNo()	. 137
	5.13.3.10	O tau_getOwnTrnCstNo()	. 137
	5.13.3.11	1 tau_getStaticCstInfo()	. 137
	5.13.3.12	2 tau_getTrDirectory()	. 138

CONTENTS xi

	5.13.3.13 tau_getTrnCstCnt()	138
	5.13.3.14 tau_getTrnVehCnt()	139
	5.13.3.15 tau_getTTI()	139
	5.13.3.16 tau_getVehInfo()	139
	5.13.3.17 tau_getVehOrient()	140
	5.13.3.18 tau_initTTlaccess()	140
5.14 tau_tti.	h File Reference	141
5.14.1	Detailed Description	144
5.14.2	Function Documentation	144
	5.14.2.1 tau_deInitTTI()	144
	5.14.2.2 tau_getCstFctCnt()	145
	5.14.2.3 tau_getCstFctInfo()	145
	5.14.2.4 tau_getCstInfo()	146
	5.14.2.5 tau_getCstVehCnt()	146
	5.14.2.6 tau_getOpTrDirectory()	147
	5.14.2.7 tau_getOpTrnDirectoryStatusInfo()	148
	5.14.2.8 tau_getOwnlds()	148
	5.14.2.9 tau_getOwnOpCstNo()	149
	5.14.2.10 tau_getOwnTrnCstNo()	149
	5.14.2.11 tau_getStaticCstInfo()	150
	5.14.2.12 tau_getTrDirectory()	150
	5.14.2.13 tau_getTrnCstCnt()	151
	5.14.2.14 tau_getTrnVehCnt()	151
	5.14.2.15 tau_getTTI()	152
	5.14.2.16 tau_getVehInfo()	152
	5.14.2.17 tau_getVehOrient()	153
	5.14.2.18 tau_initTTlaccess()	154
5.15 tau_tti_	_types.h File Reference	155
5.15.1	Detailed Description	157
5.16 tau_xm	nl.c File Reference	158

xii CONTENTS

	5.16.1	Detailed Description
	5.16.2	Macro Definition Documentation
		5.16.2.1 TRDP_SDT_DEFAULT_CMTHR
		5.16.2.2 TRDP_SDT_DEFAULT_LMIMAX
	5.16.3	Function Documentation
		5.16.3.1 tau_freeTelegrams()
		5.16.3.2 tau_freeXmlDatasetConfig()
		5.16.3.3 tau_freeXmlDoc()
		5.16.3.4 tau_prepareXmlDoc()
		5.16.3.5 tau_prepareXmlMem()
		5.16.3.6 tau_readXmlDatasetConfig()
		5.16.3.7 tau_readXmlDeviceConfig()
		5.16.3.8 tau_readXmlInterfaceConfig()
		5.16.3.9 tau_readXmlMappedDeviceConfig()
		5.16.3.10 tau_readXmlMappedDevices()
		5.16.3.11 tau_readXmlMappedInterfaceConfig()
		5.16.3.12 tau_readXmlServiceConfig()
5.17	tau_xm	I.h File Reference
	5.17.1	Detailed Description
	5.17.2	Macro Definition Documentation
		5.17.2.1 TRDP_DBG_DEFAULT
	5.17.3	Enumeration Type Documentation
		5.17.3.1 TRDP_EXCHG_OPTION_T
	5.17.4	Function Documentation
		5.17.4.1 tau_freeTelegrams()
		5.17.4.2 tau_freeXmlDatasetConfig()
		5.17.4.3 tau_freeXmlDoc()
		5.17.4.4 tau_prepareXmlDoc()
		5.17.4.5 tau_prepareXmlMem()
		5.17.4.6 tau_readXmlDatasetConfig()

CONTENTS xiii

	5.17.4.7 tau_readXmlDeviceConfig()	172
	5.17.4.8 tau_readXmlInterfaceConfig()	173
	5.17.4.9 tau_readXmlMappedDeviceConfig()	174
	5.17.4.10 tau_readXmlMappedDevices()	174
	5.17.4.11 tau_readXmlMappedInterfaceConfig()	176
	5.17.4.12 tau_readXmlServiceConfig()	176
5.18 tlc_if.c	File Reference	178
5.18.1	Detailed Description	180
5.18.2	Function Documentation	181
	5.18.2.1 tlc_closeSession()	181
	5.18.2.2 tlc_configSession()	181
	5.18.2.3 tlc_getETBTopoCount()	182
	5.18.2.4 tlc_getInterval()	182
	5.18.2.5 tlc_getOpTrainTopoCount()	183
	5.18.2.6 tlc_getOwnlpAddress()	183
	5.18.2.7 tlc_getVersion()	183
	5.18.2.8 tlc_getVersionString()	184
	5.18.2.9 tlc_init()	184
	5.18.2.10 tlc_openSession()	185
	5.18.2.11 tlc_presetIndexSession()	185
	5.18.2.12 tlc_process()	186
	5.18.2.13 tlc_reinitSession()	186
	5.18.2.14 tlc_setETBTopoCount()	187
	5.18.2.15 tlc_setOpTrainTopoCount()	187
	5.18.2.16 tlc_terminate()	188
	5.18.2.17 tlc_updateSession()	188
	5.18.2.18 trdp_getAccess()	189
	5.18.2.19 trdp_isValidSession()	189
	5.18.2.20 trdp_releaseAccess()	189
	5.18.2.21 trdp_sessionQueue()	190

xiv CONTENTS

5.19	tlc_if.h	File Refere	ence				 	190						
	5.19.1	Detailed [Desc	ription			 	192						
	5.19.2	Function	Docu	ımentati	ion .		 	192						
		5.19.2.1	trdp	_isValid	Sessi	ion()	 	192						
		5.19.2.2	trdp	_sessio	nQue	eue() .	 	192						
5.20	tlm_if.c	File Refer	rence				 	193						
	5.20.1	Detailed [Desc	ription			 	194						
	5.20.2	Function	Docu	ımentati	ion .		 	195						
		5.20.2.1	tlm_	_abortSe	essior	า()	 	195						
		5.20.2.2	tlm_	_addList	ener()	 	195						
		5.20.2.3	tlm_	_confirm	ı()		 	196						
		5.20.2.4	tlm_	_delListe	ener()		 	197						
		5.20.2.5	tlm_	_getInte	rval()		 	197						
		5.20.2.6	tlm_	_notify()			 	198						
		5.20.2.7	tlm_	_process	s() .		 	199						
		5.20.2.8	tlm_	_readdLi	istene	er() .	 	199						
		5.20.2.9	tlm_	reply()			 	200						
		5.20.2.10	tlm_	_replyQı	uery()		 	201						
		5.20.2.11	tlm_	_request	t ()		 	201						
5.21	tlp_if.c	File Refere	ence				 	202						
	5.21.1	Detailed [Desc	ription			 	204						
	5.21.2	Function	Docu	ımentati	ion .		 	205						
		5.21.2.1	tlp_	get() .			 	205						
		5.21.2.2	tlp_	getInter	val()		 	205						
		5.21.2.3	tlp_	getRedı	undar	nt() .	 	206						
		5.21.2.4	tlp_	process	Rece	ive()	 	206						
		5.21.2.5	tlp_	process	Send	()	 	207						
		5.21.2.6	tlp_	publish()		 	207						
		5.21.2.7	tlp_	put() .			 	208						
		5.21.2.8	tlp_i	putlmm	ediate	∍()	 	209						

CONTENTS xv

5.21.2.9 tlp_republish()	 209
5.21.2.10 tlp_request()	 210
5.21.2.11 tlp_resubscribe()	 211
5.21.2.12 tlp_setRedundant()	 212
5.21.2.13 tlp_subscribe()	 212
5.21.2.14 tlp_unpublish()	 213
5.21.2.15 tlp_unsubscribe()	 214
5.22 trdp_dllmain.c File Reference	 214
5.22.1 Detailed Description	 214
5.23 trdp_if_light.h File Reference	 215
5.23.1 Detailed Description	 219
5.23.2 Function Documentation	 219
5.23.2.1 tlc_closeSession()	 219
5.23.2.2 tlc_configSession()	 219
5.23.2.3 tlc_getETBTopoCount()	 220
5.23.2.4 tlc_getInterval()	 220
5.23.2.5 tlc_getJoinStatistics()	 221
5.23.2.6 tlc_getOpTrainTopoCount()	 222
5.23.2.7 tlc_getOwnlpAddress()	 222
5.23.2.8 tlc_getPubStatistics()	 222
5.23.2.9 tlc_getRedStatistics()	 223
5.23.2.10 tlc_getStatistics()	 223
5.23.2.11 tlc_getSubsStatistics()	 224
5.23.2.12 tlc_getVersion()	 224
5.23.2.13 tlc_getVersionString()	 225
5.23.2.14 tlc_init()	 225
5.23.2.15 tlc_openSession()	 225
5.23.2.16 tlc_presetIndexSession()	 226
5.23.2.17 tlc_process()	 227
5.23.2.18 tlc_reinitSession()	 227

xvi CONTENTS

5.23.2.19 tlc_resetStatistics()
5.23.2.20 tlc_setETBTopoCount()
5.23.2.21 tlc_setOpTrainTopoCount()
5.23.2.22 tlc_terminate()
5.23.2.23 tlc_updateSession()
5.23.2.24 tlm_abortSession()
5.23.2.25 tlm_addListener()
5.23.2.26 tlm_confirm()
5.23.2.27 tlm_delListener()
5.23.2.28 tlm_getInterval()
5.23.2.29 tlm_notify()
5.23.2.30 tlm_process()
5.23.2.31 tlm_readdListener()
5.23.2.32 tlm_reply()
5.23.2.33 tlm_replyQuery()
5.23.2.34 tlm_request()
5.23.2.35 tlp_get()
5.23.2.36 tlp_getInterval()
5.23.2.37 tlp_getRedundant()
5.23.2.38 tlp_processReceive()
5.23.2.39 tlp_processSend()
5.23.2.40 tlp_publish()
5.23.2.41 tlp_put()
5.23.2.42 tlp_putImmediate()
5.23.2.43 tlp_republish()
5.23.2.44 tlp_request()
5.23.2.45 tlp_resubscribe()
5.23.2.46 tlp_setRedundant()
5.23.2.47 tlp_subscribe()
5.23.2.48 tlp_unpublish()

CONTENTS xvii

		5.23.2.49	tlp_unsubscri	be()		 	 	 	 	 	. 247
5.24	trdp_m	dcom.c Fil	e Reference .			 	 	 	 	 	. 247
	5.24.1	Detailed I	Description .			 	 	 	 	 	. 249
	5.24.2	Function	Documentation	ı		 	 	 	 	 	. 249
		5.24.2.1	trdp_mdCall()			 	 	 	 	 	. 249
		5.24.2.2	trdp_mdChec	kListenSc	ocks() .	 	 	 	 	 	. 250
		5.24.2.3	trdp_mdChec	kPending	()	 	 	 	 	 	. 251
		5.24.2.4	trdp_mdChec	kTimeout	s()	 	 	 	 	 	. 251
		5.24.2.5	trdp_mdConfi	rm()		 	 	 	 	 	. 251
		5.24.2.6	trdp_mdFreeS	Session()		 	 	 	 	 	. 252
		5.24.2.7	trdp_mdGetT	CPSocke	t()	 	 	 	 	 	. 252
		5.24.2.8	trdp_mdReply	/()		 	 	 	 	 	. 253
		5.24.2.9	trdp_mdSend	()		 	 	 	 	 	. 253
5.25	trdp_m	dcom.h Fil	le Reference .			 	 	 	 	 	. 254
	5.25.1	Detailed I	Description .			 	 	 	 	 	. 255
	5.25.2	Function	Documentation	ı		 	 	 	 	 	. 256
		5.25.2.1	trdp_mdCall()			 	 	 	 	 	. 256
		5.25.2.2	trdp_mdChec	kListenSc	ocks() .	 	 	 	 	 	. 257
		5.25.2.3	trdp_mdChec	kPending	()	 	 	 	 	 	. 257
		5.25.2.4	trdp_mdChec	kTimeout	:s()	 	 	 	 	 	. 258
		5.25.2.5	trdp_mdConfi	rm()		 	 	 	 	 	. 258
		5.25.2.6	trdp_mdFreeS	Session()		 	 	 	 	 	. 259
		5.25.2.7	trdp_mdGetT	CPSocke	t()	 	 	 	 	 	. 259
		5.25.2.8	trdp_mdReply	/()		 	 	 	 	 	. 259
		5.25.2.9	trdp_mdSend	()		 	 	 	 	 	. 260
5.26	trdp_pc	dcom.c File	e Reference .			 	 	 	 	 	. 260
	5.26.1	Detailed I	Description .			 	 	 	 	 	. 262
	5.26.2	Function	Documentation	ı		 	 	 	 	 	. 262
		5.26.2.1	trdp_handleT	imeout()		 	 	 	 	 	. 262
		5.26.2.2	trdp_pdCheck	(()		 	 	 	 	 	. 263

xviii CONTENTS

		5.26.2.3	tro	dp_pdC	heck	kListe	enSo	ocks(() .	 	 	 	 	 			263
		5.26.2.4	tro	dp_pdC	heck	kPen	nding	J() .		 	 	 	 	 			264
		5.26.2.5	tro	dp_pdE)istrib	oute(()			 	 	 	 	 			264
		5.26.2.6	tro	dp_pdF	landl	leTin	neOı	uts()		 	 	 	 	 			265
		5.26.2.7	tro	dp_pdlr	nit()					 	 	 	 	 			266
		5.26.2.8	tro	dp_pdF	ut()					 	 	 	 	 			266
		5.26.2.9	tro	dp_pdF	lecei	ive()				 	 	 	 	 			267
		5.26.2.10	0 tr	dp_pdS	end(() .				 	 	 	 	 			267
		5.26.2.11	1 tro	dp_pdS	endE	Elem	nent()		 	 	 	 	 			268
		5.26.2.12	2 tr	dp_pdS	endl	lmme	ediat	te()		 	 	 	 	 			269
		5.26.2.13	3 tr	dp_pdS	end(Queu	u <mark>ed</mark> ())		 	 	 	 	 			269
		5.26.2.14	4 tr	dp_pdL	Jpdat	te()				 	 	 	 	 			270
5.27	trdp_pc	dcom.h File	ile F	Reference	ce .					 	 	 	 	 			270
	5.27.1	Detailed	De	scriptio	n .					 	 	 	 	 			272
	5.27.2	Function	n Do	ocumen	tatior	n .				 	 	 	 	 			273
		5.27.2.1	tro	dp_han	dleTi	imed	out()			 	 	 	 	 			273
		5.27.2.2	tro	dp_pdC	heck	k()				 	 	 	 	 			273
		5.27.2.3	tro	dp_pdC	heck	kListe	enSo	ocks(() .	 	 	 	 	 			274
		5.27.2.4	tro	dp_pdC	heck	kPen	nding	J () .		 	 	 	 	 			274
		5.27.2.5	tro	dp_pdE)istrib	oute(()			 	 	 	 	 			275
		5.27.2.6	tro	dp_pdF	landl	leTin	neOı	uts()		 	 	 	 	 			275
		5.27.2.7	tro	dp_pdIr	nit()					 	 	 	 	 			276
		5.27.2.8	tro	dp_pdF	ut()					 	 	 	 	 			277
		5.27.2.9	tro	dp_pdF	lecei [,]	ive()				 	 	 	 	 			277
		5.27.2.10	0 tr	dp_pdS	end(() .				 	 	 	 	 			278
		5.27.2.11	1 tr	dp_pdS	endE	Elem	nent()		 	 	 	 	 			278
		5.27.2.12	2 tr	dp_pdS	endl	lmme	ediat	te()		 	 	 	 	 			279
		5.27.2.13	3 tr	dp_pdS	end(Quei	ued())		 	 	 	 	 			280
		5.27.2.14	4 tr	dp_pdL	Jpdat	te()				 	 	 	 	 			281
5.28	trdp_pc	dindex.c Fi	-ile	Referer	ice.					 	 	 	 	 			281

CONTENTS xix

	5.28.1	Detailed I	Descript	ion .						 	 	 	 	 	282
5.29	trdp_pc	dindex.h Fi	ile Refer	ence						 	 	 	 	 	282
	5.29.1	Detailed I	Descript	ion .						 	 	 	 	 	284
	5.29.2	Macro De	efinition I	Docum	entatio	n .				 	 	 	 	 	284
		5.29.2.1	TRDP_	DEFA	ULT_IN	NDEX	(_SIZI	ES .		 	 	 	 	 	284
5.30	trdp_pr	ivate.h File	e Refere	nce .						 	 	 	 	 	285
	5.30.1	Detailed I	Descript	ion .						 	 	 	 	 	287
	5.30.2	Macro De	efinition I	Docum	entatio	n .				 	 	 	 	 	288
		5.30.2.1	TRDP_	_MAX_I	PD_SC	OCKE	T_CN	IT .		 	 	 	 	 	288
	5.30.3	Enumerat	tion Type	e Docu	mentat	tion				 	 	 	 	 	288
		5.30.3.1	TRDP_	MD_E	LE_ST	_T				 	 	 	 	 	288
		5.30.3.2	TRDP_	SOCK	_TYPE	<u>=_</u> T				 	 	 	 	 	288
5.31	trdp_se	erviceRegis	stry.h Fil	e Refe	rence					 	 	 	 	 	289
	5.31.1	Detailed I	Descript	ion .						 	 	 	 	 	292
	5.31.2	Macro De	efinition I	Docum	entatio	n .				 	 	 	 	 	292
		5.31.2.1	SOA_S	SAME_	SERVI	CEID)			 	 	 	 	 	292
		5.31.2.2	SRM_S	SERVIC	CE_RE	AD_I	REQ_	СОМ	ID .	 	 	 	 	 	292
		5.31.2.3	SRM_S	SRVINE	=O_NC	TIFY	_COI	MID		 	 	 	 	 	292
5.32	trdp_st	ats.c File F	Referenc	e						 	 	 	 	 	293
	5.32.1	Detailed I	Descript	ion .						 	 	 	 	 	294
	5.32.2	Function	Docume	ntation	١					 	 	 	 	 	294
		5.32.2.1	tlc_get	JoinSta	ıtistics(0 .				 	 	 	 	 	294
		5.32.2.2	tlc_getI	⊃ubSta	ıtistics() .				 	 	 	 	 	295
		5.32.2.3	tlc_getI	RedSta	ıtistics(0 .				 	 	 	 	 	295
		5.32.2.4	tlc_get	Statistic	cs()					 	 	 	 	 	296
		5.32.2.5	tlc_gets	SubsSt	atistics	s() .				 	 	 	 	 	296
		5.32.2.6	tlc_rese	etStatis	stics() .					 	 	 	 	 	297
		5.32.2.7	trdp_in	itStats()					 	 	 	 	 	297
		5.32.2.8	trdp_pc	dPrepa	reStats	s() .				 	 	 	 	 	298
		5.32.2.9	trdp_U _I	pdateS	tats().					 	 	 	 	 	299

CONTENTS

5.33 trdp_sta	ats.h File R	Reference	 . 299
5.33.1	Detailed D	Description	 . 300
5.33.2	Function [Documentation	 . 300
	5.33.2.1	trdp_initStats()	 . 300
	5.33.2.2	trdp_pdPrepareStats()	 . 301
5.34 trdp_tsr	n_def.h File	e Reference	 . 302
5.34.1	Detailed D	Description	 . 302
5.34.2	Macro De	finition Documentation	 . 303
	5.34.2.1	TRDP_MIN_PD2_HEADER_SIZE	 . 303
	5.34.2.2	TRDP_MSG_TSN_PD	 . 303
	5.34.2.3	TRDP_PD_DEFAULT_QOS	 . 303
5.35 trdp_typ	oes.h File F	Reference	 . 303
5.35.1	Detailed D	Description	 . 308
5.35.2	Macro De	finition Documentation	 . 309
	5.35.2.1	TRDP_FLAGS_DEFAULT	 . 309
5.35.3	Typedef D	Documentation	 . 309
	5.35.3.1	TRDP_IP_ADDR_T	 . 309
	5.35.3.2	TRDP_MARSHALL_T	 . 309
	5.35.3.3	TRDP_MD_CALLBACK_T	 . 310
	5.35.3.4	TRDP_PD_CALLBACK_T	 . 310
	5.35.3.5	TRDP_PRINT_DBG_T	 . 310
	5.35.3.6	TRDP_TIME_T	 . 310
	5.35.3.7	TRDP_UNMARSHALL_T	 . 311
5.35.4	Enumerat	ion Type Documentation	 . 311
	5.35.4.1	TRDP_DATA_TYPE_T	 . 311
	5.35.4.2	TRDP_ERR_T	 . 312
	5.35.4.3	TRDP_RED_STATE_T	 . 313
	5.35.4.4	TRDP_REPLY_STATUS_T	 . 313
	5.35.4.5	TRDP_TO_BEHAVIOR_T	 . 313
5.36 trdp_uti	ls.c File Re	eference	 . 314

CONTENTS xxi

5.36.1	Detailed D	Des	crip	tion							 		 		 				•	315
5.36.2	Function D	Doc	cum	enta	ation						 	 	 		 					316
	5.36.2.1	pri	ntS	ocke	etUs	age) ()				 	 	 		 					316
	5.36.2.2	trd	p _c	hecl	kSed	que	nce	;Coı	unte	er()			 		 					316
	5.36.2.3	trd	p_fi	indN	1Cjoi	ins() .				 		 	-	 					317
	5.36.2.4	trd	p_fi	ndS	ubA	ddr	() .				 		 		 					317
	5.36.2.5	trd	p_g	jetSe	eqCı	nt()					 		 		 					318
	5.36.2.6	trd	p_iı	nitSc	ocke	ts()					 		 		 					318
	5.36.2.7	trd	p_is	sAdo	dres	sed	() .				 		 		 					319
	5.36.2.8	trd	p_is	sInIF	Pran	ge()) .				 		 		 					319
	5.36.2.9	trd	p_p	acke	etSiz	zeM	1D())			 		 		 					319
	5.36.2.10	trd	p_p	acke	etSiz	zeP	'D()				 		 		 					320
	5.36.2.11	trd	p_c	lueu	еАр	pLa	ıst()) .			 		 		 					320
	5.36.2.12	trd	p_c	lueu	eDe	lEle	eme	ent())		 		 		 					320
	5.36.2.13	trd	p_c	lueu	eFin	ndC	om	ld()			 		 		 					321
	5.36.2.14	trd	p_c	lueu	eFin	ndE	xist	ings	Sub	o ()	 		 		 					321
	5.36.2.15	trd	p_c	lueu	eFin	ndPı	ubA	∖ddr	r()		 	 	 		 					322
	5.36.2.16	trd	p_c	lueu	eFin	ndSı	ubA	∖ddr	r()		 		 		 					322
	5.36.2.17	trd	p_c	lueu	elns	Fire	st()				 		 		 					323
	5.36.2.18	trd	p_r	elea	seS	ock	et()	١			 	 	 		 					323
	5.36.2.19	trd	p_r	eque	estS	ock	et())			 		 		 					323
	5.36.2.20	trd	p_r	eset	Seq	uen	ıce(Cou	ınte	r()	 		 		 					324
	5.36.2.21	trd	p_8	3ock	Add	Joir	า()				 		 		 					325
	5.36.2.22	trd	p_8	Sock	Del	Join	() .				 		 		 					325
	5.36.2.23	trd	p_8	Sock	IsJo	ine	d()				 		 		 					325
	5.36.2.24	trd	p_v	'alid	Торс	Со	unt	ers(()		 		 		 					326
trdp_uti	ils.h File Re	efe	rend	Эе							 		 		 					326
5.37.1	Detailed D	Des	crip	tion							 		 		 					328
5.37.2	Function D	Doc	cum	enta	ation						 		 		 					329
	5.37.2.1	pri	ntS	ocke	etUs	age) ()				 		 		 					329

5.37

xxii CONTENTS

	5.37.2.2 trdp_checkSequenceCounter()	29
	5.37.2.3 trdp_findMCjoins()	30
	5.37.2.4 trdp_findSubAddr()	30
	5.37.2.5 trdp_getSeqCnt()	31
	5.37.2.6 trdp_initSockets()	31
	5.37.2.7 trdp_isAddressed()	32
	5.37.2.8 trdp_isInIPrange()	32
	5.37.2.9 trdp_packetSizeMD()	32
	5.37.2.10 trdp_packetSizePD()	33
	5.37.2.11 trdp_queueAppLast()	33
	5.37.2.12 trdp_queueDelElement()	33
	5.37.2.13 trdp_queueFindComId()	34
	5.37.2.14 trdp_queueFindExistingSub()	34
	5.37.2.15 trdp_queueFindPubAddr()	35
	5.37.2.16 trdp_queueFindSubAddr()	35
	5.37.2.17 trdp_queueInsFirst()	36
	5.37.2.18 trdp_releaseSocket()	36
	5.37.2.19 trdp_requestSocket()	36
	5.37.2.20 trdp_resetSequenceCounter()	37
	5.37.2.21 trdp_SockAddJoin()	38
	5.37.2.22 trdp_SockDelJoin()	38
	5.37.2.23 trdp_SockIsJoined()	38
	5.37.2.24 trdp_validTopoCounters()	39
5.38 trdp_x	ml.c File Reference	39
5.38.1	Detailed Description	41
5.38.2	Function Documentation	41
	5.38.2.1 trdp_XMLClose()	41
	5.38.2.2 trdp_XMLCountStartTag()	41
	5.38.2.3 trdp_XMLEnter()	42
	5.38.2.4 trdp_XMLGetAttribute()	42
		_

CONTENTS xxiii

	5.38.2.5 trdp_XMLLeave()
	5.38.2.6 trdp_XMLMemOpen()
	5.38.2.7 trdp_XMLOpen()
	5.38.2.8 trdp_XMLRewind()
	5.38.2.9 trdp_XMLSeekStartTag()
	5.38.2.10 trdp_XMLSeekStartTagAny()
5.39 trdp_x	ml.h File Reference
5.39.1	Detailed Description
5.39.2	Function Documentation
	5.39.2.1 trdp_XMLClose()
	5.39.2.2 trdp_XMLCountStartTag()
	5.39.2.3 trdp_XMLEnter()
	5.39.2.4 trdp_XMLGetAttribute()
	5.39.2.5 trdp_XMLLeave()
	5.39.2.6 trdp_XMLMemOpen()
	5.39.2.7 trdp_XMLOpen()
	5.39.2.8 trdp_XMLRewind()
	5.39.2.9 trdp_XMLSeekStartTag()
	5.39.2.10 trdp_XMLSeekStartTagAny()
5.40 vos_m	nem.c File Reference
5.40.1	Detailed Description
5.40.2	Function Documentation
	5.40.2.1 vos_bsearch()
	5.40.2.2 vos_memAlloc()
	5.40.2.3 vos_memCount()
	5.40.2.4 vos_memDelete()
	5.40.2.5 vos_memFree()
	5.40.2.6 vos_memInit()
	5.40.2.7 vos_qsort()
	5.40.2.8 vos_queueCreate()

xxiv CONTENTS

	5.40.2.9 vos_queueDestroy()	57
	5.40.2.10 vos_queueReceive()	57
	5.40.2.11 vos_queueSend()	58
	5.40.2.12 vos_strncat()	58
	5.40.2.13 vos_strncpy()	30
	5.40.2.14 vos_strnicmp()	30
5.41 vos_m	em.h File Reference	31
5.41.1	Detailed Description	33
5.41.2	Macro Definition Documentation	33
	5.41.2.1 VOS_MEM_MAX_PREALLOCATE	33
	5.41.2.2 VOS_MEM_PREALLOCATE	33
5.41.3	Function Documentation	33
	5.41.3.1 vos_bsearch()	33
	5.41.3.2 vos_memAlloc()	34
	5.41.3.3 vos_memCount()	34
	5.41.3.4 vos_memDelete()	35
	5.41.3.5 vos_memFree()	35
	5.41.3.6 vos_memInit()	36
	5.41.3.7 vos_qsort()	36
	5.41.3.8 vos_queueCreate()	37
	5.41.3.9 vos_queueDestroy()	38
	5.41.3.10 vos_queueReceive()	38
	5.41.3.11 vos_queueSend()	39
	5.41.3.12 vos_strncat()	39
	5.41.3.13 vos_strncpy()	39
	5.41.3.14 vos_strnicmp()	70
5.42 vos_sh	nared_mem.h File Reference	70
5.42.1	Detailed Description	71
5.42.2	Function Documentation	72
	5.42.2.1 vos_sharedClose()	72

CONTENTS xxv

	5.42.2.2 vos_sharedOpen()
5.43 vos_s	ock.h File Reference
5.43.1	Detailed Description
5.43.2	Macro Definition Documentation
	5.43.2.1 VOS_MAX_SOCKET_CNT
	5.43.2.2 VOS_TTL_MULTICAST
5.43.3	Function Documentation
	5.43.3.1 vos_determineBindAddr()
	5.43.3.2 vos_dottedIP()
	5.43.3.3 vos_getInterfaces()
	5.43.3.4 vos_htonl()
	5.43.3.5 vos_htonll()
	5.43.3.6 vos_htons()
	5.43.3.7 vos_ipDotted()
	5.43.3.8 vos_isMulticast()
	5.43.3.9 vos_netlfUp()
	5.43.3.10 vos_ntohl()
	5.43.3.11 vos_ntohll()
	5.43.3.12 vos_ntohs()
	5.43.3.13 vos_select()
	5.43.3.14 vos_sockAccept()
	5.43.3.15 vos_sockBind()
	5.43.3.16 vos_sockClose()
	5.43.3.17 vos_sockConnect()
	5.43.3.18 vos_sockGetMAC()
	5.43.3.19 vos_sockInit()
	5.43.3.20 vos_sockJoinMC()
	5.43.3.21 vos_sockLeaveMC()
	5.43.3.22 vos_sockListen()
	5.43.3.23 vos_sockOpenTCP()

xxvi CONTENTS

	5.43.3.24 vos_sockOpenUDP()	386
	5.43.3.25 vos_sockReceiveTCP()	386
	5.43.3.26 vos_sockReceiveUDP()	387
	5.43.3.27 vos_sockSendTCP()	388
	5.43.3.28 vos_sockSendUDP()	388
	5.43.3.29 vos_sockSetMulticastIf()	389
	5.43.3.30 vos_sockSetOptions()	389
	5.43.3.31 vos_sockTerm()	390
5.44 vos_th	read.h File Reference	390
5.44.1	Detailed Description	392
5.44.2	Function Documentation	393
	5.44.2.1 vos_addTime()	393
	5.44.2.2 vos_clearTime()	393
	5.44.2.3 vos_cmpTime()	393
	5.44.2.4 vos_divTime()	394
	5.44.2.5 vos_getRealTime()	394
	5.44.2.6 vos_getTime()	394
	5.44.2.7 vos_getTimeStamp()	395
	5.44.2.8 vos_getUuid()	395
	5.44.2.9 vos_mulTime()	395
	5.44.2.10 vos_mutexCreate()	396
	5.44.2.11 vos_mutexDelete()	396
	5.44.2.12 vos_mutexLock()	396
	5.44.2.13 vos_mutexTryLock()	397
	5.44.2.14 vos_mutexUnlock()	397
	5.44.2.15 vos_semaCreate()	398
	5.44.2.16 vos_semaDelete()	398
	5.44.2.17 vos_semaGive()	398
	5.44.2.18 vos_semaTake()	399
	5.44.2.19 vos_subTime()	399

CONTENTS xxvii

		5.44.2.20	vos_threadCreate()	 399
		5.44.2.21	vos_threadCreateSync()	 400
		5.44.2.22	vos_threadDelay()	 401
		5.44.2.23	vos_threadInit()	 401
		5.44.2.24	vos_threadIsActive()	 402
		5.44.2.25	vos_threadSelf()	 402
		5.44.2.26	vos_threadTerm()	 402
		5.44.2.27	vos_threadTerminate()	 403
5.45	vos_typ	es.h File R	Reference	 403
	5.45.1	Detailed D	Description	 405
	5.45.2	Typedef D	ocumentation	 405
		5.45.2.1	VOS_PRINT_DBG_T	 405
		5.45.2.2	VOS_TIMEVAL_T	 405
	5.45.3	Enumerati	on Type Documentation	 406
		5.45.3.1	VOS_ERR_T	 406
		5.45.3.2	VOS_LOG_T	 406
5.46	vos_uti	ls.c File Re	ference	 407
	5.46.1	Detailed D	Description	 408
	5.46.2	Function [Documentation	 408
		5.46.2.1	vos_crc32()	 408
		5.46.2.2	vos_getErrorString()	 409
		5.46.2.3	vos_getVersion()	 409
		5.46.2.4	vos_getVersionString()	 409
		5.46.2.5	vos_hostIsBigEndian()	 410
		5.46.2.6	vos_init()	 410
		5.46.2.7	vos_sc32()	 410
		5.46.2.8	vos_terminate()	 411
5.47	vos_uti	ls.h File Re	ference	 411
	5.47.1	Detailed D	Description	 413
	5.47.2	Macro Det	finition Documentation	 413
		5.47.2.1	INITFCS	 413
		5.47.2.2	VOS_MAX_ERR_STR_SIZE	 413
		5.47.2.3	VOS_MAX_FRMT_SIZE	 413
		5.47.2.4	VOS_MAX_PRNT_STR_SIZE	 414
	5.47.3	Function [Documentation	 414
		5.47.3.1	vos_crc32()	 414
		5.47.3.2	vos_getErrorString()	 415
		5.47.3.3	vos_getVersion()	 415
			vos_getVersionString()	
		5.47.3.5	vos_hostIsBigEndian()	 416
		5.47.3.6	vos_init()	 416
		5.47.3.7	vos_sc32()	 417
		5.47.3.8	vos_terminate()	 417

•••	001177170
XXVIII	CONTENTS
AVIII	OCITIENTS

Index 419

Chapter 1

The TRDP Light Library API Specification



1.1 General Information

1.1.1 Purpose

The TRDP protocol has been defined as the standard communication protocol in IP-enabled trains. It allows communication via process data (periodically transmitted data using UDP/IP) and message data (client - server messaging using UDP/IP or TCP/IP) This document describes the light API of the TRDP Library.

1.1.2 Scope

The intended audience of this document is the developers and project members of the TRDP project. TRDP Client Applications are programs using the TRDP protocol library to access the services of TRDP. Programmers developing such applications are the main target audience for this documentation.

1.1.3 Related documents

TCN-TRDP2-D-BOM-004-01 IEC61375-2-3_CD_ANNEXA Protocol definition of the TRDP standard TCN-TRDP2-D-BOM-011-32 TRDP User's Manual

1.1.4 Abbreviations and Definitions

- -API Application Programming Interface -ECN Ethernet Consist Network -TRDP Train Real-time Data Protocol
- -TCMS Train Control Management System

1.2 Terminology

The API documented here is mainly concerned with three bodies of code:

- TRDP Client Applications (or 'client applications' for short): These are programs using the API to access the services of TRDP. Programmers developing such applications are the main target audience for this documentation.
- TRDP Light Implementations (or just 'TRDP implementation'): These are libraries realising the API as documented here. Programmers developing such implementations will find useful definitions about syntax and semantics of the API wihtin this documentation.
- VOS Subsystem (Virtual Operating System): An OS and hardware abstraction layer which offers memory, networking, threading, queues and debug functions. The VOS API is documented here.

1.3 Use Cases

The following diagram shows how these pieces of software are interrelated. Single threaded flow:

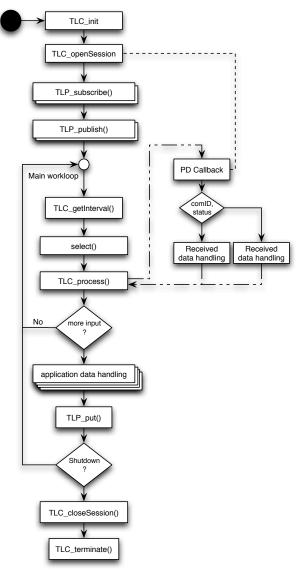


Figure 1.1 Sample client workflow (Single Thread)

1.3 Use Cases 3

Usage of the separate process handling (separate threads for PD and MD):

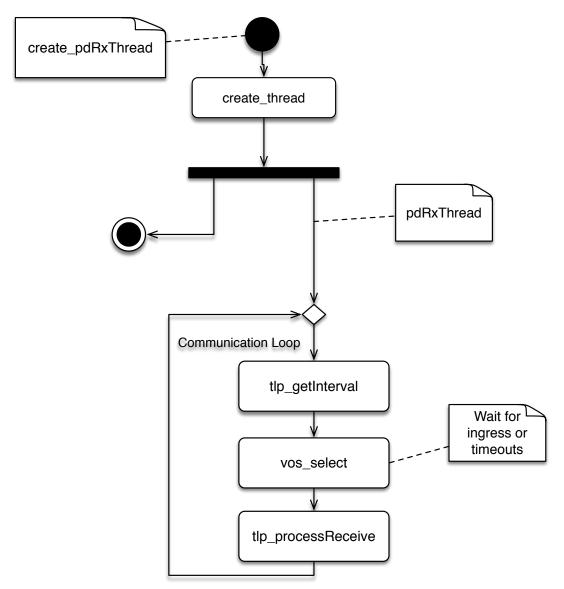


Figure 1.2 Multi-threaded processing of PD Reception

The transmit thread should be a cyclic thread. Cycle times down to 1ms are supported:

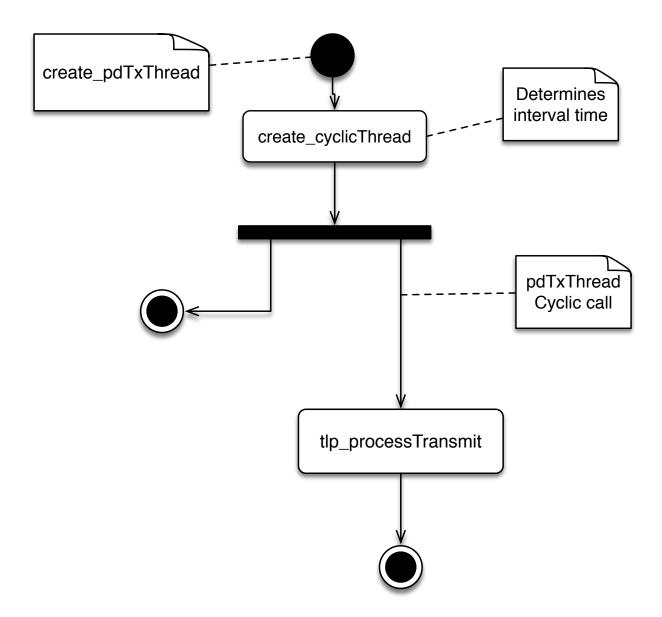


Figure 1.3 Multi-threaded processing of PD Transmit

1.4 Conventions of the API 5

If Message Data support is needed (MD_SUPPORT=1):

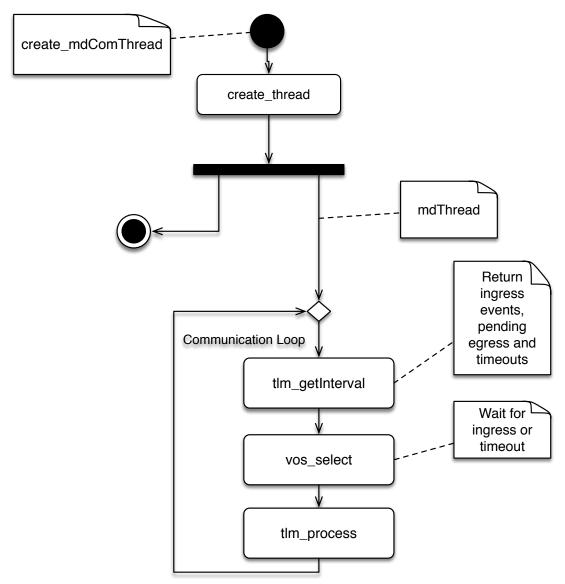


Figure 1.4 Multi-threaded processing of MD

Note: Mixed usage of the single threaded call tlc_process() with the multi-threaded calls tlm_process/tlp_process Transmit/tlp_processReceive is not supported!

1.4 Conventions of the API

The API comprises a set of C header files that can also be used from client applications written in C++. These header files are contained in a directory named trdp/api and a subdirectory called trdp/vos/api with declarations not topical to TRDP but needed by the stack. Client applications shall include these header files like:

#include "trdp_if_light.h"

and, if VOS functions are needed, also the corresponding headers:

#include "vos_thread.h"

for example.

The subdirectory trdp/doc contains files needed for the API documentation.

Generally client application source code including API headers will only compile if the parent directory of the trdp directory is part of the include path of the used compiler. No other subdirectories of the API should be added to the compiler's include path.

The client API doesn't support a "catch-all" header file that includes all declarations in one step; rather the client application has to include individual headers for each feature set it wants to use.

Further description of the API and the usage of the TRDP protocol stack can be found in the TCNOpen TRDP User's Manual (at tcnopen.eu).

Chapter 2

Data Structure Index

2.1 Data Structures

Here are the data structures with brief descriptions:

DNS_HEADER	
DNS header structure	13
GNU_PACKED	
Types for ETB control	13
hp_slot	
Low cycle-time slots	32
hp_slots	
Entry for the application session	33
PD_ELE	
Queue element for PD packets to send or receive	34
service_info	٠,
Preliminary definition of a service info entry	36
srv_info_req	0-
Preliminary definition of a service info request	37
TAU_MARSHALL_INFO_T	20
Marshalling info, used to and from wire	38
TCN_URI	38
TCN-DNS simplified header structures	SC
Closed train consists information	39
TRDP COM PARAM T	Je
Quality/type of service, time to live , no	39
TRDP_COMID_DSID_MAP_T	U
Comld - data set mapping element definition	40
TRDP CONSIST INFO T	-
Consist information structure	41
TRDP DATASET	
Dataset definition	43
TRDP_DATASET_ELEMENT_T	
Dataset element definition	44
TRDP DBG CONFIG T	
Control for debug output device/file on application level	45
TRDP DNS REPLY	
TCN-DNS Reply telegram TCN_DNS_REP_DS	45
TRDP_DNS_REQUEST	
TCN-DNS Request telegram TCN DNS REQ DS	47

8 Data Structure Index

TRDP_ETB_INFO_T	
Types for train configuration information	48
Function/device information structure	49
TRDP_HANDLE	
Hidden handle definition, used as unique addressing item	50
TRDP_IDX_TABLE_T	
Settings for pre-allocation of index tables for application session initialization	51
Marshaling/unmarshalling configuration	53
TRDP MD CONFIG T	•
Default MD configuration	54
TRDP_MD_INFO_T	•
Message data info from received telegram; allows the application to generate responses	55
TRDP_MEM_CONFIG_T	
Enumeration type for memory pre-fragmentation, reuse of VOS definition	56
TRDP_PD_CONFIG_T	
Default PD configuration	57
TRDP PD INFO T	
Process data info from received telegram; allows the application to generate responses	57
TRDP PROCESS CONFIG T	
Various flags/general TRDP options for library initialization	58
TRDP_PROP_T	
Application defined properties	59
TRDP_SDT_PAR_T	
Types to read out the XML configuration	60
TRDP_SEQ_CNT_ENTRY_T	
Tuples of last received sequence counter per comId	60
TRDP_SESSION	
Session/application variables store	61
TRDP_SOCKET_TCP	
TCP parameters	62
TRDP_SOCKETS	
Socket item	63
TRDP_VEHICLE_INFO_T	
Vehicle information structure	65
TRDP_XML_DOC_HANDLE_T	
Parsed XML document handle	66
VOS_SOCK_OPT_T	
Common socket options	66
VOS_VERSION_T	
Version information	67

Chapter 3

File Index

3.1 File List

Here is a list of all documented files with brief descriptions:

iec61375-2-3.h	
All definitions from IEC 61375-2-3	69
tau_cstinfo.c	
Functions for consist information access	77
tau_ctrl.c	
	79
tau_ctrl.h	
•	83
tau_ctrl_types.h TRDP utility interface definitions	88
tau_dnr.c	,
	91
tau_dnr.h	
TRDP utility interface definitions	96
tau_dnr_types.h	
TRDP utility interface definitions)3
tau_marshall.c	
Marshalling functions for TRDP)5
tau_marshall.h	4 -
TRDP utility interface definitions	11
Access to service oriented functions of the SRM	21
tau_so_if.h	- '
Access to the Service Registry	26
tau_tti.c	
Functions for train topology information access	30
tau_tti.h	
TRDP utility interface definitions	41
tau_tti_types.h	
TRDP utility interface definitions	55
tau_xml.c	
Functions for XML file parsing	36
tau_xml.h TRDP utility interface definitions	26
tlc_if.c	,
	_,

10 File Index

tlc_if.h		
tlm if o	Typedefs for TRDP communication	190
tlm_if.c	Functions for Message Data Communication	193
tlp_if.c	Functions for Process Data Communication	202
trdp dllm		202
	Windows DLL main function	214
trdp_if_li	ght.h TRDP Light interface functions (API)	215
trdp_mdo		
trdp_mdo	Functions for MD communication	247
trup_mut	Functions for MD communication	254
trdp_pdc	om.c	
	Functions for PD communication	260
trdp_pdc	om.n Functions for PD communication	270
trdp_pdir		
	Functions for indexed PD communication	281
trdp_pdir	ndex.h Functions for indexed PD communication	282
trdp_priv		202
	Typedefs for TRDP communication	285
trdp_serv	viceRegistry.h Additional definitions for IEC 61375-2-3 (Service Discovery) The definitions herein are prelimi-	000
trdp_stat	nary and will change with the next major release of the IEC 61375-2-3 standard	289
up_ u	Statistics functions for TRDP communication	293
trdp_stat	s.h Statistics for TRDP communication	299
trdp_tsn_		
	Additional definitions for TSN	302
trdp_type	ss.n Typedefs for TRDP communication	303
trdp_utils	s.c	
	Helper functions for TRDP communication	314
trdp_utils	s.h Common utilities for TRDP communication	326
trdp_xml		320
. –	Simple XML parser	339
trdp_xml		245
vos men	Simple XML parser	345
	Memory functions	351
vos_men		
voe char	Memory and queue functions for OS abstraction	361
V03_311a1	Shared Memory functions for OS abstraction	370
vos_sock	c.h	
vaa thra	Typedefs for OS abstraction	373
vos_thre	ad.n Threading functions for OS abstraction	390
vos_type	s.h	
	Typedefs for OS abstraction	403
vos_utils	.c Common functions for VOS	407

3.1 File List	44
3. I FIIH I ISI	■

vos_utils.h	
Typedefs for OS abstraction	 41

12 File Index

Chapter 4

Data Structure Documentation

4.1 DNS_HEADER Struct Reference

DNS header structure.

4.1.1 Detailed Description

DNS header structure.

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

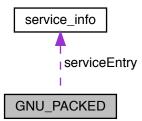
• tau_dnr.c

4.2 GNU_PACKED Struct Reference

Types for ETB control.

#include <trdp_private.h>

Collaboration diagram for GNU_PACKED:



Data Fields

UINT8 trnVehNo

vehicle sequence number within the train with vehicle 01 being the first vehicle in ETB reference direction 1 as defined in IEC61375-2-5 value range: 0..63 a value of 0 indicates that this vehicle has been inserted by correction

ANTIVALENT8 isLead

vehicle is leading

• UINT8 leadDir

vehicle leading direction 0 = not relevant 1 = leading direction 1 2 = leading direction 2

UINT8 vehOrient

 $vehicle \ orientation \ 0 = not \ known \ (corrected \ vehicle) \ 1 = same \ as \ operational \ train \ direction \ 2 = inverse \ to \ operational \ train \ direction$

• TRDP_SHORT_VERSION_T version

telegram version information, main_version = 1, sub_version = 0

• UINT16 reserved01

reserved (=0)

UINT8 trnCstNo

own TCN consist number (= 1..32)

UINT8 reserved02

reserved (=0)

UINT8 ownOpCstNo

own operational address (= 1..32) = 0 if unknown (e.g.

• UINT8 reserved03

reserved (=0)

UINT32 cstTopoCount

Consist topology counter.

UINT32 trnTopoCount

Train directory topology counter.

UINT32 opTrnTopoCount

Operational Train topology counter.

ANTIVALENT8 wasLead

consist was leading, '01'B = false, '10'B = true

ANTIVALENT8 reqLead

leading request, '01'B = false, '10'B = true

UINT8 reqLeadDir

(request) leading direction, '01'B = consist direction 1, '10'B = consist direction 2

ANTIVALENT8 accLead

accept remote leading request, '01'B = false/not accepted, '10'B = true/accepted

ANTIVALENT8 clearConfComp

clear confirmed composition, '01'B = false, '10'B = true

ANTIVALENT8 corrRequest

request confirmation, '01'B = false, '10'B = true

ANTIVALENT8 corrInfoSet

correction info set, '01'B = false, '10'B = true

ANTIVALENT8 compStored

corrected composition stored, '01'B = false, '10'B = true

• ANTIVALENT8 sleepRequest

request sleep mode, '01'B = false, '10'B = true

UINT8 leadVehOfCst

position of leading vehicle in consist, 0..31 (1: first vehicle in consist in Direction 1, 2: second vehicle, etc.)

• UINT8 reserved04

reserved (=0)

UINT16 reserved05

reserved (=0)

• UINT8 reserved06

reserved (=0)

UINT8 confVehCnt

number of confirmed vehicles in train (1..63)

TRDP_CONF_VEHICLE_T confVehList [TRDP_MAX_VEH_CNT]

dynamic ordered list of confirmed vehicles in train, starting with vehicle at train head, see sub-clause 5.3.3.2.6

TRDP_ETB_CTRL_VDP_T safetyTrail

ETBCTRL-VDP trailer, completely set to 0 == not used.

UINT8 reserved01

reserved (=0)

• TRDP_NET_LABEL_T deviceName

function device of ECSC which sends the telegram

UINT8 inhibit

inauguration inhibit 0 = no inhibit request 1 = inhibit request

UINT8 leadingReq

leading request 0 = no leading request 1 = leading request

UINT8 leadingDir

leading direction 0 = no leading request 1 = leading request direction 1 2 = leading request direction 2

UINT8 sleepReq

sleep request 0 = no sleep request 1 = sleep request

· UINT16 lifesign

wrap-around counter, incremented with each produced datagram.

UINT8 ecspState

ECSP state indication 0 = ECSP not operational(initial value) 1 = ECSP in operation.

· UINT8 etbInhibit

inauguration inhibit indication 0 = n/a (default) 1 = inhibit not requested on ETB 2 = inhibit set on local ETBN 3 = inhibit set on remote ETBN 4 = inhibit set on local and remote ETBN

· UINT8 etbLength

indicates train lengthening in case train inauguration is inhibit 0 = no lengthening (default) 1 = lengthening detected

UINT8 etbShort

indicates train shortening in case train inauguration is inhibit 0 = no shortening (default) 1 = shortening detected

• UINT16 reserved02

reserved (=0)

UINT8 etbLeadState

indication of local consist leadership 5 = consist not leading (initial value) 6 = consist is leading requesting 9 = consist is leading 10 = leading conflict other values are not allowed

UINT8 etbLeadDir

direction of the leading end car in the local consist 0 = unknown (default) 1 = TCN direction 1 2 = TCN direction 2 other values are not allowed

UINT8 ttdbSrvState

TTDB server state indication 0 = n/a (initial value) 1 = Leader (default) 2 = Follower 3 = Error.

UINT8 dnsSrvState

DNS server state indication 0 = n/a (initial value) 1 = Leader (default) 2 = Follower 3 = Error.

UINT8 trnDirState

train directory state 1 = UNCONFIRMED 2 = CONFIRMED other values are not allowed

UINT8 opTrnDirState

train directory state 1 = INVALID 2 = VALID 4 = SHARED other values are not allowed

UINT8 sleepCtrlState

sleep control state (option) 0 = option not available 1 = RegularOperation 2 = WaitForSleepMode 3 = PrepareFor SleepMode

UINT8 sleepReqCnt

number of sleep requests (option) value range: 0..63, not used = 0

UINT32 opTrnTopoCnt

operational train topology counter

UINT8 command

confirmation order 1 = confirmation/correction request 2 = un-confirmation request

UINT16 confVehCnt

number of confirmed vehicles in the train (1..63).

TRDP OP VEHICLE T confVehList [TRDP MAX VEH CNT]

ordered list of confirmed vehicles in the train, starting with vehicle at train head, see chapter 5.3.3.2.10.

UINT8 status

status of storing correction info 0 = correctly stored 1 = not stored

UINT32 regSafetyCode

SC-32 value of the request message.

UINT8 byPassCtrl

ETBN bypass control 0 = no action (keep old state) 1 = no bypass 2 = activate bypass.

UINT8 txCtrl

ETBN transmission control 0 = no action (keep old state) 1 = activate sending on ETB (default) 2 = stop sending on ETB.

UINT8 slCtrl

sleep mode control (option) 0 = no action (keep old state) 1 = deactivate sleep mode 2 = activate sleep mode (line activity sensing)

· UINT8 etbnState

state indication of the (active) ETBN 0 = ETBN not operational(initial value) 1 = ETBN in operation

UINT8 etbnInaugState

ETBN inauguration state as defined in IEC61375-2-5 0 = init 1 = not inaugurated 2 = inaugurated 3 = ready for inauguration.

· UINT8 etbnPosition

position of the ETBN 0 = unknown (default) 1 = single node 2 = middle node 3 = end node TCN direction <math>1 = single node 2 = middle node 3 = end node TCN direction <math>1 = single node 2 = middle node 3 = end node TCN direction <math>1 = single node 2 = middle node 3 = end node TCN direction <math>1 = single node 2 = middle node 3 = end node TCN direction <math>1 = single node 2 = middle node 3 = end node TCN direction <math>1 = single node 2 = middle node 3 = end node TCN direction <math>1 = single node 2 = middle node 3 = end node TCN direction <math>1 = single node 2 = middle node 3 = end node TCN direction <math>1 = single node 2 = middle node 3 = end node TCN direction <math>1 = single node 3 = end node 3 = e

UINT8 etbnRole

ETBN node role as defined in IEC61375-2-5 0 = undefined 1 = master (redundancy leader) 2 = backup (redundancy follower) 3 = not redundant.

BITSET8 etbLineState

indication of ETB line status (FALSE == not trusted, TRUE == trusted) bit0 = line A ETBN direction 1 bit1 = line B ETBN direction 1 bit2 = line C ETBN direction 1 bit3 = line D ETBN direction 1 bit4 = line A ETBN direction 2 bit5 = line B ETBN direction 2 bit6 = line C ETBN direction 2 bit7 = line D ETBN direction 2

UINT8 byPassState

state of bypass function 0 = bypass disabled 1 = bypass enabled

UINT8 slState

sleep mode state (option) 0 = no sleep mode 1 = sleep mode active (line activity sensing)

UINT32 etbTopoCnt

ETB topography counter.

• TRDP_TRAIN_NET_DIR_T trnNetDir

dynamic train info

• UINT8 ver

Version - incremented for incompatible changes.

UINT8 rel

Release - incremented for compatible changes.

UINT32 reserved01

reserved (=0)

TRDP_SHORT_VERSION_T userDataVersion

version of the vital ETBCTRL telegram mainVersion = 1, subVersion = 0

UINT32 safeSegCount

safe sequence counter, as defined in B.9

UINT32 safetyCode

checksum, as defined in B.9

TRDP_UUID_T cstUUID

UUID of the consist, provided by ETBN (TrainNetworkDirectory) Reference to static consist attributes 0 if not available (e.g.

UINT32 cstTopoCnt

consist topology counter provided with the CSTINFO 0 if no CSTINFO available

UINT8 cstOrient

consist orientation '01'B = same as train direction '10'B = inverse to train direction

UINT8 cstCnt

number of consists in train; range: 1..63

• TRDP_CONSIST_T cstList [TRDP_MAX_CST_CNT]

consist list.

UINT32 trnTopoCnt

trnTopoCnt value ctrlType == 0: actual value ctrlType == 1: set to 0

· UINT8 etbld

identification of the ETB the TTDB is computed for bit0: ETB0 (operational network) bit1: ETB1 (multimedia network) bit2: ETB2 (other network) bit3: ETB3 (other network)

· TRDP NET LABEL T vehId

Unique vehicle identifier, application defined (e.g.

UINT8 opVehNo

operational vehicle sequence number in train value range 1..63

UINT8 opCstNo

operational consist number in train (1..63)

UINT8 opCstOrient

consist orientation '00'B = not known (corrected vehicle) '01'B = same as operational train direction '10'B = inverse to operational train direction

TRDP_NET_LABEL_T trnld

train identifier, application defined (e.g.

• TRDP NET LABEL T trnOperator

train operator, e.g.

UINT32 crc

sc-32 computed over record (seed value: 'FFFFFFF'H)

UINT8 opTrnOrient

operational train orientation '00'B = unknown '01'B = same as train direction '10'B = inverse to train direction

UINT8 opCstCnt

number of consists in train (1..63)

TRDP_OP_CONSIST_T opCstList [TRDP_MAX_CST_CNT]

operational consist list starting with op.

• UINT8 reserved05

reserved for future use (= 0)

UINT8 opVehCnt

number of vehicles in train (1..63)

TRDP_OP_VEHICLE_T opVehList [TRDP_MAX_VEH_CNT]

operational vehicle list starting with op.

• TRDP_OP_TRAIN_DIR_STATE_T state

operational state of the train

UINT32 cstNetProp

consist network properties bit0..1: consist orientation bit2..7: 0 bit8..13: ETBN Id bit14..15: 0 bit16..21: subnet Id bit24..29: CN Id bit30..31: 0

UINT16 entryCnt

number of entries in train network directory

TRDP_TRAIN_NET_DIR_ENTRY_T trnNetDir [TRDP_MAX_CST_CNT]

train network directory

TRDP_OP_TRAIN_DIR_T opTrnDir

operational directory

TRDP_TRAIN_DIR_T trnDir

train directory

• UINT16 noOfEntries

number of entries in array

• SRM_SERVICE_INFO_T serviceEntry [1]

var.

• UINT32 comId

Comld to request: 35...41.

UINT32 total

total memory size

UINT32 free

free memory size

UINT32 minFree

minimal free memory size in statistics interval

UINT32 numAllocBlocks

allocated memory blocks

UINT32 numAllocErr

allocation errors

UINT32 numFreeErr

free errors

UINT32 blockSize [VOS_MEM_NBLOCKSIZES]

preallocated memory blocks

• UINT32 usedBlockSize [VOS_MEM_NBLOCKSIZES]

used memory blocks

• UINT32 defQos

default QoS for PD

UINT32 defTtl

default TTL for PD

UINT32 defTimeout

default timeout in us for PD

UINT32 numSubs

number of subscribed Comld's

UINT32 numPub

number of published ComId's

UINT32 numRcv

number of received PD packets

UINT32 numCrcErr

number of received PD packets with CRC err

UINT32 numProtErr

number of received PD packets with protocol err

UINT32 numTopoErr

number of received PD packets with wrong topo count

UINT32 numNoSubs

number of received PD push packets without subscription

UINT32 numNoPub

number of received PD pull packets without publisher

UINT32 numTimeout

number of PD timeouts

UINT32 numSend

number of sent PD packets

UINT32 numMissed

number of packets skipped

· UINT32 defReplyTimeout

default reply timeout in us for MD

UINT32 defConfirmTimeout

default confirm timeout in us for MD

UINT32 numList

number of listeners

• UINT32 numNoListener

number of received MD packets without listener

UINT32 numReplyTimeout

number of reply timeouts

UINT32 numConfirmTimeout

number of confirm timeouts

UINT32 version

TRDP version.

UINT64 timeStamp

actual time stamp

UINT32 upTime

time in sec since last initialisation

UINT32 statisticTime

time in sec since last reset of statistics

• TRDP_NET_LABEL_T hostName

host name

• TRDP_NET_LABEL_T leaderName

leader host name

• TRDP_IP_ADDR_T ownIpAddr

own IP address

• TRDP_IP_ADDR_T leaderlpAddr

leader IP address

UINT32 processPrio

priority of TRDP process

UINT32 processCycle

cycle time of TRDP process in microseconds

UINT32 numJoin

number of joins

UINT32 numRed

number of redundancy groups

• TRDP_MEM_STATISTICS_T mem

memory statistics

TRDP_PD_STATISTICS_T pd

pd statistics

• TRDP_MD_STATISTICS_T udpMd

UDP md statistics.

• TRDP MD STATISTICS T tcpMd

TCP md statistics.

• TRDP_IP_ADDR_T joinedAddr

Joined IP address.

• TRDP_IP_ADDR_T filterAddr

Filter IP address, i.e IP address of the sender for this subscription, 0.0.0.0 in case all senders.

UINT32 callBack

call back function if used

UINT32 userRef

User reference if used.

UINT32 timeout

Time-out value in us.

• UINT32 status

Receive status information TRDP_NO_ERR, TRDP_TIMEOUT_ERR.

UINT32 toBehav

Behavior at time-out.

UINT32 numRecv

Number of packets received for this subscription.

TRDP_IP_ADDR_T destAddr

IP address of destination for this publishing.

· UINT32 cycle

Publishing cycle in us.

UINT32 redId

Redundancy group id.

UINT32 redState

Redundant state.Leader or Follower.

UINT32 numPut

Number of packet updates.

CHAR8 uri [32]

URI user part to listen to.

• UINT32 queue

Queue reference if used.

UINT32 id

Redundant Id.

UINT32 state

Redundant state.Leader or Follower.

UINT32 sequenceCounter

Unique counter (autom incremented)

• UINT16 protocolVersion

fix value for compatibility (set by the API)

UINT16 msgType

of datagram: PD Request (0x5072) or PD_MSG (0x5064)

• UINT32 datasetLength

length of the data to transmit 0...1432

UINT32 reserved

reserved for ServiceID/InstanceID support

UINT32 replyComId

used in PD request

• UINT32 replylpAddress

used for PD request

• UINT32 frameCheckSum

CRC32 of header.

INT32 replyStatus

0 = OK

UINT8 sessionID [16u]

UUID as a byte stream.

UINT32 replyTimeout

in us

• UINT8 sourceURI [32u]

User part of URI.

UINT8 destinationURI [32u]

User part of URI.

PD_HEADER_T frameHead

Packet header in network byte order.

UINT8 data [TRDP_MAX_PD_DATA_SIZE]

data ready to be sent or received

4.2.1 Detailed Description

Types for ETB control.

TRDP PD packet.

TRDP message data header - network order and alignment.

TRDP process data header - network order and alignment.

A table containing PD redundant group information.

Information about a particular MD listener.

Table containing particular PD publishing information.

Table containing particular PD subscription information.

Structure containing all general memory, PD and MD statistics information.

Structure containing all general MD statistics information.

Structure containing all general PD statistics information.

Structure containing all general memory statistics information.

TRDP statistics type definitions.

Complete TTDB structure.

Train network directory structure.

Train network directory entry structure acc.

Operational Train directory status info structure.

Operational train structure.

Operational train directory state.

Operational consist structure.

Operational vehicle structure.

TCN train directory.

CSTINFO Control telegram.

TCN consist structure.

Version information for communication buffers.

to IEC61375-2-5

Statistical data regarding the former info provided via SNMP the following information was left out/can be implemented additionally using MD:

- PD subscr table: Comld, sourcelpAddr, destlpAddr, cbFct?, timout, toBehavior, counter
- PD publish table: Comld, destlpAddr, redld, redState cycle, ttl, qos, counter
- · PD join table: joined MC address table
- MD listener table: ComId destIpAddr, destUri, cbFct?, counter
- · Memory usageStructure containing comld for MD statistics request (Comld 32).

4.2.2 Field Documentation

4.2.2.1 callBack

UINT32 GNU_PACKED::callBack

call back function if used

Call back function if used.

4.2.2.2 comld

UINT32 GNU_PACKED::comId

ComId to request: 35...41.

set by user: unique id

ComId to listen to.

Published Comld.

Subscribed Comld.

4.2.2.3 confVehCnt

```
UINT16 GNU_PACKED::confVehCnt
```

number of confirmed vehicles in the train (1..63).

4.2.2.4 confVehList

```
TRDP_OP_VEHICLE_T GNU_PACKED::confVehList[TRDP_MAX_VEH_CNT]
```

ordered list of confirmed vehicles in the train, starting with vehicle at train head, see chapter 5.3.3.2.10.

Parameters 'isLead' and 'leadDir' to be set to 0

4.2.2.5 cstList

```
TRDP_CONSIST_T GNU_PACKED::cstList
```

consist list.

consist list ordered list starting with trnCstNo == 1 Note: This is a variable size array, only opCstCnt array elements are present on the network and for crc computation

If trnCstNo > 0 this shall be an ordered list starting with trnCstNo == 1 (exactly the same as in structure TRAIN \leftarrow _DIRECTORY). If trnCstNo == 0 it is not mandatory to list all consists (only consists which should send CSTINFO telegram). The parameters 'trnCstNo' and 'cstOrient' are optional and can be set to 0.

4.2.2.6 cstUUID

```
TRDP_UUID_T GNU_PACKED::cstUUID
```

UUID of the consist, provided by ETBN (TrainNetworkDirectory) Reference to static consist attributes 0 if not available (e.g.

unique consist identifier

Reference to static consist attributes, 0 if not available (e.g.

correction)

4.2.2.7 datasetLength

UINT32 GNU_PACKED::datasetLength

length of the data to transmit 0...1432

defined by user: length of data to transmit

4.2.2.8 defQos

UINT32 GNU_PACKED::defQos

default QoS for PD

default QoS for MD

4.2.2.9 defTtl

UINT32 GNU_PACKED::defTtl

default TTL for PD

default TTL for MD

4.2.2.10 destAddr

```
TRDP_IP_ADDR_T GNU_PACKED::destAddr
```

IP address of destination for this publishing.

4.2.2.11 deviceName

```
TRDP_NET_LABEL_T GNU_PACKED::deviceName
```

function device of ECSC which sends the telegram

function device of ED which sends the telegram

4.2.2.12 etbld

```
UINT8 GNU_PACKED::etbId
```

identification of the ETB the TTDB is computed for bit0: ETB0 (operational network) bit1: ETB1 (multimedia network) bit2: ETB2 (other network) bit3: ETB3 (other network)

identification of the ETB the TTDB is computed for 0: ETB0 (operational network) 1: ETB1 (multimedia network) 2: ETB2 (other network) 3: ETB3 (other network)

4.2.2.13 etbTopoCnt

UINT32 GNU_PACKED::etbTopoCnt

ETB topography counter.

set by user: ETB to use, '0' for consist local traffic

train network directory CRC

4.2.2.14 filterAddr

```
TRDP_IP_ADDR_T GNU_PACKED::filterAddr
```

Filter IP address, i.e IP address of the sender for this subscription, 0.0.0.0 in case all senders.

4.2.2.15 inhibit

```
UINT8 GNU_PACKED::inhibit
```

inauguration inhibit 0 = no inhibit request 1 = inhibit request

ETBN inhibit 0 = no action (keep old state) 1 = no inhibit request 2 = inhibit request.

4.2.2.16 isLead

```
ANTIVALENT8 GNU_PACKED::isLead
```

vehicle is leading

consist contains leading vehicle, '01'B = false, '10'B = true

4.2.2.17 leadDir

```
UINT8 GNU_PACKED::leadDir
```

vehicle leading direction 0 = not relevant 1 = leading direction 1 2 = leading direction 2

'vehicle leading direction 0 = not relevant 1 = leading direction 1 2 = leading direction 2

4.2.2.18 leadVehOfCst

```
UINT8 GNU_PACKED::leadVehOfCst
```

position of leading vehicle in consist, 0..31 (1: first vehicle in consist in Direction 1, 2: second vehicle, etc.)

position of leading vehicle in consist range 0...32 0 = not defined 1 = first vehicle in consist in direction 1 2 = second vehicle etc.

4.2.2.19 lifesign

```
UINT16 GNU_PACKED::lifesign
```

wrap-around counter, incremented with each produced datagram.

4.2.2.20 msgType

UINT16 GNU_PACKED::msgType

of datagram: PD Request (0x5072) or PD_MSG (0x5064)

of datagram: Mn, Mr, Mp, Mq, Mc or Me

4.2.2.21 numCrcErr

UINT32 GNU_PACKED::numCrcErr

number of received PD packets with CRC err

number of received MD packets with CRC err

4.2.2.22 numMissed

UINT32 GNU_PACKED::numMissed

number of packets skipped

number of packets skipped for this subscription

4.2.2.23 numProtErr

UINT32 GNU_PACKED::numProtErr

number of received PD packets with protocol err

number of received MD packets with protocol err

4.2.2.24 numRcv

UINT32 GNU_PACKED::numRcv

number of received PD packets

number of received MD packets

4.2.2.25 numRecv

UINT32 GNU_PACKED::numRecv

Number of packets received for this subscription.

Number of received packets.

4.2.2.26 numSend UINT32 GNU_PACKED::numSend number of sent PD packets Number of packets sent out. number of sent MD packets 4.2.2.27 numTopoErr

UINT32 GNU_PACKED::numTopoErr

number of received PD packets with wrong topo count

number of received MD packets with wrong topo count

4.2.2.28 opCstList

```
TRDP_OP_CONSIST_T GNU_PACKED::opCstList[TRDP_MAX_CST_CNT]
```

operational consist list starting with op.

consist #1 Note: This is a variable size array, only opCstCnt array elements are present

4.2.2.29 opTrnDirState

```
UINT8 GNU_PACKED::opTrnDirState
```

train directory state 1 = INVALID 2 = VALID 4 = SHARED other values are not allowed

Operational train directory status: '01'B == invalid, '10'B == valid, '100'B == shared.

4.2.2.30 opTrnTopoCnt

```
UINT32 GNU_PACKED::opTrnTopoCnt
```

operational train topology counter

set by user: direction/side critical, '0' if ignored

operational train topology counter computed as defined in 5.3.3.2.16 (seed value : trnTopoCnt)

operational train topology counter set to 0 if opTrnDirState == invalid

operational train topocounter value of the operational train directory the correction is based on

reserved for future use (= 0)

```
4.2.2.31 opVehList
TRDP_OP_VEHICLE_T GNU_PACKED::opVehList[TRDP_MAX_VEH_CNT]
operational vehicle list starting with op.
vehicle #1 Note: This is a variable size array, only opCstCnt array elements are present
4.2.2.32 ownOpCstNo
UINT8 GNU_PACKED::ownOpCstNo
own operational address (= 1..32) = 0 if unknown (e.g.
operational consist number the vehicle belongs to
after Inauguration)
4.2.2.33 protocolVersion
UINT16 GNU_PACKED::protocolVersion
fix value for compatibility (set by the API)
fix value for compatibility
4.2.2.34 reserved01 [1/2]
UINT16 GNU_PACKED::reserved01
reserved (=0)
reserved for future use (= 0)
4.2.2.35 reserved01 [2/2]
UINT8 GNU_PACKED::reserved01
reserved (=0)
reserved for future use (= 0)
4.2.2.36 reserved02 [1/2]
UINT16 GNU_PACKED::reserved02
reserved (=0)
reserved (= 0)
```

```
4.2.2.37 reserved02 [2/2]
UINT16 GNU_PACKED::reserved02
reserved (=0)
reserved (= 0)
4.2.2.38 reserved03
UINT8 GNU_PACKED::reserved03
reserved (=0)
reserved for future use (= 0)
4.2.2.39 reserved04
UINT8 GNU_PACKED::reserved04
reserved (=0)
reserved for future use (= 0)
4.2.2.40 reserved06
UINT8 GNU_PACKED::reserved06
reserved (=0)
reserved for future use (= 0)
4.2.2.41 safetyTrail
TRDP_ETB_CTRL_VDP_T GNU_PACKED::safetyTrail
ETBCTRL-VDP trailer, completely set to 0 == not used.
ETBCTRL-VDP trailer, parameter 'safeSequCount' == 0 completely set to 0 == not used.
ETBCTRL-VDP trailer, parameter 'safeSequCount' == 0 completely set to 0 == SDTv2 not used.
ETBCTRL-VDP trailer, completely set to 0 == SDTv2 not used.
4.2.2.42 serviceEntry
SRM_SERVICE_INFO_T GNU_PACKED::serviceEntry[1]
var.
number of entries
```

4.2.2.43 timeout

UINT32 GNU_PACKED::timeout

Time-out value in us.

0 = No time-out supervision

4.2.2.44 toBehav

UINT32 GNU_PACKED::toBehav

Behavior at time-out.

Set data to zero / keep last value

4.2.2.45 trnCstNo

UINT8 GNU_PACKED::trnCstNo

own TCN consist number (= 1..32)

sequence number of consist in train with vehicle 01 being the first vehicle in ETB reference direction 1 as defined in IEC61375-2-5, value range: 1..63, 0 = inserted by correction

train consist number telegram control type 0 = with trnTopoCnt tracking 1 = without trnTopoCnt tracking

Sequence number of consist in train (1..63)

4.2.2.46 trnDirState

UINT8 GNU_PACKED::trnDirState

train directory state 1 = UNCONFIRMED 2 = CONFIRMED other values are not allowed

TTDB status: '01'B == unconfirmed, '10'B == confirmed.

4.2.2.47 trnld

TRDP_NET_LABEL_T GNU_PACKED::trnId

train identifier, application defined (e.g.

'ICE75', 'IC346'), informal

4.2.2.48 trnNetDir

TRDP_TRAIN_NET_DIR_T GNU_PACKED::trnNetDir

dynamic train info

network directory

4.2.2.49 trnOperator

TRDP_NET_LABEL_T GNU_PACKED::trnOperator

train operator, e.g.

'trenitalia.it', informal

4.2.2.50 trnTopoCnt

UINT32 GNU_PACKED::trnTopoCnt

trnTopoCnt value ctrlType == 0: actual value ctrlType == 1: set to 0

computed as defined in 5.3.3.2.16 (seed value: etbTopoCnt)

4.2.2.51 trnVehNo

UINT8 GNU_PACKED::trnVehNo

vehicle sequence number within the train with vehicle 01 being the first vehicle in ETB reference direction 1 as defined in IEC61375-2-5 value range: 0..63 a value of 0 indicates that this vehicle has been inserted by correction

vehicle sequence number within the train with vehicle 01 being the first vehicle in ETB reference direction 1 as defined in IEC61375-2-5, value range: 1..63, a value of 0 indicates that this vehicle has been inserted by correction

4.2.2.52 vehld

TRDP_NET_LABEL_T GNU_PACKED::vehId

Unique vehicle identifier, application defined (e.g.

UIC Identifier)

4.2.2.53 vehOrient

UINT8 GNU_PACKED::vehOrient

vehicle orientation 0 = not known (corrected vehicle) 1 = same as operational train direction 2 = inverse to operational train direction

vehicle orientation, '00'B = not known (corrected vehicle) '01'B = same as operational train direction '10'B = inverse to operational train direction

4.2.2.54 version

TRDP_SHORT_VERSION_T GNU_PACKED::version

telegram version information, main_version = 1, sub_version = 0

1.0 telegram version

Train info structure version.

TrainDirectoryState data structure version parameter 'mainVersion' shall be set to 1.

TrainDirectory data structure version parameter 'mainVersion' shall be set to 1.

Consist Info Control structure version parameter 'mainVersion' shall be set to 1.

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

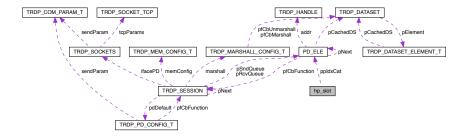
- tau_ctrl_types.h
- tau_tti_types.h
- trdp_serviceRegistry.h
- trdp_types.h
- · trdp private.h

4.3 hp_slot Struct Reference

Low cycle-time slots.

#include <trdp_pdindex.h>

Collaboration diagram for hp_slot:



Data Fields

UINT32 slotCycle

cycle time with which each slot will be called (us)

• UINT8 noOfTxEntries

no of slots == first array dimension

• UINT8 depthOfTxEntries

depth of slots == second array dimension

const PD_ELE_T ** ppldxCat

pointer to an array of PD_ELE_T* (dim[depth][slot])

· UINT32 allocatedTableSize

real allocated size

4.3.1 Detailed Description

Low cycle-time slots.

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

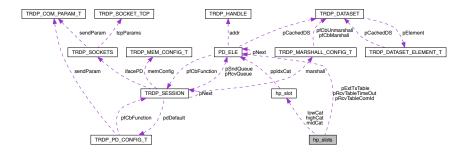
· trdp_pdindex.h

4.4 hp_slots Struct Reference

entry for the application session

#include <trdp_pdindex.h>

Collaboration diagram for hp_slots:



Data Fields

UINT32 processCycle

system cycle time with which lowest array will be called

UINT32 currentCycle

the current cycle of the send loop

TRDP_HP_CAT_SLOT_T lowCat

array dim[slot][depth]

TRDP_HP_CAT_SLOT_T midCat

array dim[slot][depth]

TRDP_HP_CAT_SLOT_T highCat

array dim[slot][depth]

UINT32 noOfRxEntries

number of subscribed PDs to be handled

PD_ELE_T ** pRcvTableComId

Pointer to sorted array of PDs to be handled.

PD_ELE_T ** pRcvTableTimeOut

Pointer to sorted array of PDs to be handled.

UINT32 allocatedRcvTableSize

real allocated size

UINT8 noOfExtTxEntries

number of 'special' PDs to be handled

PD_ELE_T ** pExtTxTable

Pointer to array of PDs to be handled.

UINT32 allocatedExtTxTableSize

real allocated size

4.4.1 Detailed Description

entry for the application session

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

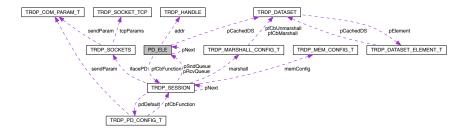
· trdp_pdindex.h

4.5 PD_ELE Struct Reference

Queue element for PD packets to send or receive.

```
#include <trdp_private.h>
```

Collaboration diagram for PD_ELE:



Data Fields

struct PD_ELE * pNext

pointer to next element or NULL

UINT32 magic

prevent acces through dangeling pointer

• TRDP_ADDRESSES_T addr

handle of publisher/subscriber

TRDP_IP_ADDR_T lastSrcIP

last source IP a subscribed packet was received from

• TRDP_IP_ADDR_T pullipAddress

In case of pulling a PD this is the requested Ip.

UINT32 redId

Redundancy group ID or zero.

UINT32 curSeqCnt

the last sent or received sequence counter

UINT32 curSeqCnt4Pull

the last sent sequence counter for PULL

TRDP_SEQ_CNT_LIST_T * pSeqCntList

pointer to list of received sequence numbers per comld

UINT32 numRxTx

Counter for received packets (statistics)

UINT32 updPkts

Counter for updated packets (statistics)

UINT32 getPkts

Counter for read packets (statistics)

UINT32 numMissed

Counter for skipped sequence number (statistics)

• TRDP_ERR_T lastErr

Last error (timeout)

• TRDP_PRIV_FLAGS_T privFlags

private flags

TRDP_FLAGS_T pktFlags

flags

TRDP_TIME_T interval

time out value for received packets or interval for packets to send (set from ms)

TRDP_TIME_T timeToGo

next time this packet must be sent/rcv

• TRDP_TO_BEHAVIOR_T toBehavior

timeout behavior for packets

UINT32 dataSize

net data size

UINT32 grossSize

complete packet size (header, data)

UINT32 sendSize

data size sent out

• TRDP_DATASET_T * pCachedDS

Pointer to dataset element if known.

INT32 socketldx

index into the socket list

const void * pUserRef

from subscribe()

• TRDP_PD_CALLBACK_T pfCbFunction

Pointer to PD callback function.

• PD_PACKET_T * pFrame

header ...

4.5.1 Detailed Description

Queue element for PD packets to send or receive.

4.5.2 Field Documentation

4.5.2.1 pFrame

```
PD_PACKET_T* PD_ELE::pFrame
```

header ...

data + FCS...

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

• trdp_private.h

4.6 service info Struct Reference

Preliminary definition of a service info entry.

```
#include <trdp_serviceRegistry.h>
```

Data Fields

- TRDP_NET_LABEL_T srvName
 - service short name as defined in X
- UINT32 serviceId
 - High Byte = serviceInstanceId Low 24 Bits = serviceTypeId
- TRDP SHORT VERSION T srvVers
 - service version
- UINT8 srvFlags
 - Flags Bit0: 0 = non safety related; 1 = safety related Bit1: 0 = global service 1 = local service Bit3: 0 = complete service list 1 = service list update Bit4: 0 = add service (update only) 1 = delete service (update only) Bit2-7: reserved for future use (= 0)
- UINT8 reserved01
 - reserved for future use (= 0)
- TIMEDATE64 srvTTL
 - Time to Live (us or ns?)
- TRDP_NET_LABEL_T fctDev
 - host identification of the function device the service is located on.
- UINT8 cstVehNo
 - sequence number of the vehicle within the consist (1..32)
- UINT8 cstNo
 - sequence number of the consist (1..63)
- UINT16 reserved03
 - reserved for future use (= 0)
- UINT32 addInfo [3]
 - service specific information

4.6.1 Detailed Description

Preliminary definition of a service info entry.

4.6.2 Field Documentation

4.6.2.1 fctDev

```
TRDP_NET_LABEL_T service_info::fctDev
```

- host identification of the function device the service is located on.

Defined in IEC 61375-2-3.

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

· trdp serviceRegistry.h

4.7 srv_info_req Struct Reference

Preliminary definition of a service info request.

```
#include <trdp_serviceRegistry.h>
```

Data Fields

- TRDP_SHORT_VERSION_T version
 - version of the telegram mainVersion = 1 subversion = 0
- UINT16 reserved01
 - reserved for future use (= 0)
- UINT32 trnTopoCnt
 - trnTopoCnt value
- UINT16 reserved02
 - reserved for future use (= 0)
- UINT8 reserved03
 - reserved for future use (= 0)
- UINT8 cstCnt
 - number of consists in list if set to 255 all consists are requested to resend their SRVINFO telegram if set to >0 and <64 only consists with different srvTopoCnt value are requested to resend their SRVINFO telegram
- UINT32 srvTcList []
 - list of srvTopoCnt values obtained from all consists set to 0 if unknown ordered list starting with trnCstNo = 1

4.7.1 Detailed Description

Preliminary definition of a service info request.

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

trdp_serviceRegistry.h

4.8 TAU_MARSHALL_INFO_T Struct Reference

Marshalling info, used to and from wire.

Data Fields

• INT32 level

track recursive level

UINT8 * pSrc

source pointer

UINT8 * pSrcEnd

last source

UINT8 * pDst

destination pointer

UINT8 * pDstEnd

last destination

4.8.1 Detailed Description

Marshalling info, used to and from wire.

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

· tau_marshall.c

4.9 TCN_URI Struct Reference

TCN-DNS simplified header structures.

```
#include <tau_dnr_types.h>
```

Data Fields

• CHAR8 tcnUriStr [80]

if != 0 use TCN DNS as resolver

• INT16 resolvState

on request: reserved (= 0), on reply: -1 unknown, 0 OK

UINT32 tcnUrilpAddr

IP address of URI.

UINT32 tcnUrilpAddr2

if != 0, end IP address of range

4.9.1 Detailed Description

TCN-DNS simplified header structures.

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

· tau_dnr_types.h

4.10 TRDP_CLTR_CST_INFO_T Struct Reference

Closed train consists information.

```
#include <tau_tti_types.h>
```

Data Fields

TRDP_UUID_T cltrCstUUID

closed train consist UUID

UINT8 cltrCstOrient

closed train consist orientation '01'B = same as closed train direction '10'B = inverse to closed train direction

UINT8 cltrCstNo

sequence number of the consist within the closed train, value range 1..32

• UINT16 reserved01

reserved for future use (= 0)

4.10.1 Detailed Description

Closed train consists information.

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

• tau_tti_types.h

4.11 TRDP_COM_PARAM_T Struct Reference

Quality/type of service, time to live , no.

```
#include <trdp_types.h>
```

Data Fields

• UINT8 qos

Quality of service (default should be 2 for PD and 2 for MD, TSN priority >= 3)

UINT8 ttl

Time to live (default should be 64)

UINT8 retries

MD Retries from XML file.

BOOL8 tsn

if TRUE, do not schedule packet but use TSN socket

UINT16 vlan

VLAN Id to be used.

4.11.1 Detailed Description

Quality/type of service, time to live, no.

of retries, TSN flag and VLAN ID

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

• trdp_types.h

4.12 TRDP_COMID_DSID_MAP_T Struct Reference

Comld - data set mapping element definition.

```
#include <trdp_types.h>
```

Data Fields

UINT32 comld

comld

UINT32 datasetId

corresponding dataset Id

4.12.1 Detailed Description

Comld - data set mapping element definition.

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

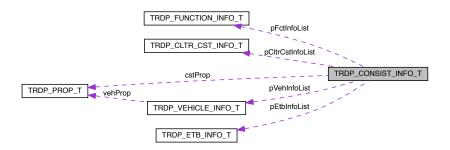
trdp_types.h

4.13 TRDP_CONSIST_INFO_T Struct Reference

consist information structure

#include <tau_tti_types.h>

Collaboration diagram for TRDP CONSIST INFO T:



Data Fields

• TRDP_SHORT_VERSION_T version

 $ConsistInfo\ data\ structure\ version,\ application\ defined\ mainVersion = 1,\ subVersion = 0.$

UINT8 cstClass

consist info classification 1 = (single) consist 2 = closed train 3 = closed train consist

UINT8 reserved01

reserved for future use (= 0)

• TRDP_NET_LABEL_T cstld

application defined consist identifier, e.g.

TRDP_NET_LABEL_T cstType

consist type, application defined

• TRDP_NET_LABEL_T cstOwner

consist owner, e.g.

TRDP_UUID_T cstUUID

consist UUID

UINT32 reserved02

reserved for future use (= 0)

TRDP_PROP_T cstProp

static consist properties

• UINT16 reserved03

reserved for future use (= 0)

UINT16 etbCnt

number of ETB's, range: 1..4

• TRDP_ETB_INFO_T * pEtbInfoList

ETB information list for the consist Ordered list starting with lowest etbld.

• UINT16 reserved04

reserved for future use (= 0)

UINT16 vehCnt

number of vehicles in consist 1..32

• TRDP_VEHICLE_INFO_T * pVehInfoList

vehicle info list for the vehicles in the consist Ordered list starting with cstVehNo==1

UINT16 reserved05

reserved for future use (= 0)

UINT16 fctCnt

number of consist functions value range 0..1024

• TRDP_FUNCTION_INFO_T * pFctInfoList

function info list for the functions in consist lexicographical ordered by fctName

• UINT16 reserved06

reserved for future use (= 0)

UINT16 cltrCstCnt

number of original consists in closed train value range: 0..32, 0 = consist is no closed train

• TRDP_CLTR_CST_INFO_T * pCltrCstInfoList

info on closed train composition Ordered list starting with cltrCstNo == 1

UINT32 cstTopoCnt

consist topology counter computed as defined in 5.3.3.2.16, seed value: 'FFFFFFF'H

4.13.1 Detailed Description

consist information structure

4.13.2 Field Documentation

```
4.13.2.1 cstld
```

```
TRDP_NET_LABEL_T TRDP_CONSIST_INFO_T::cstId
```

application defined consist identifier, e.g.

UIC identifier

4.13.2.2 cstOwner

```
TRDP_NET_LABEL_T TRDP_CONSIST_INFO_T::cstOwner
```

consist owner, e.g.

"trenitalia.it", "sncf.fr", "db.de"

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

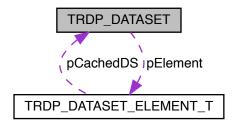
tau_tti_types.h

4.14 TRDP_DATASET Struct Reference

Dataset definition.

#include <trdp_types.h>

Collaboration diagram for TRDP_DATASET:



Data Fields

• UINT32 id

dataset identifier > 1000

• UINT16 reserved1

Reserved for future use, must be zero.

UINT16 numElement

Number of elements.

TRDP_DATASET_ELEMENT_T pElement []

Pointer to a dataset element, used as array.

4.14.1 Detailed Description

Dataset definition.

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

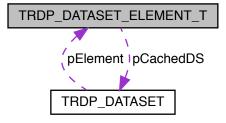
trdp_types.h

4.15 TRDP_DATASET_ELEMENT_T Struct Reference

Dataset element definition.

#include <trdp_types.h>

Collaboration diagram for TRDP_DATASET_ELEMENT_T:



Data Fields

· UINT32 type

Data type (TRDP_DATA_TYPE_T 1...99) or dataset id > 1000.

UINT32 size

Number of items or TRDP_VAR_SIZE (0)

• CHAR8 * name

Name param, on special request (Ticket #211)

· CHAR8 * unit

Unit text for visualisation.

• REAL32 scale

Factor for visualisation.

INT32 offset

Offset for visualisation (val = scale * x + offset)

• struct TRDP_DATASET * pCachedDS

Used internally for marshalling speed-up.

4.15.1 Detailed Description

Dataset element definition.

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

trdp_types.h

4.16 TRDP_DBG_CONFIG_T Struct Reference

Control for debug output device/file on application level.

```
#include <tau_xml.h>
```

Data Fields

- TRDP_DBG_OPTION_T option
 Debug printout options for application use.
- UINT32 maxFileSize

Maximal file size.

• TRDP_FILE_NAME_T fileName

Debug file name and path.

4.16.1 Detailed Description

Control for debug output device/file on application level.

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

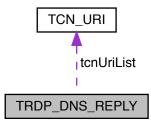
· tau xml.h

4.17 TRDP_DNS_REPLY Struct Reference

TCN-DNS Reply telegram TCN_DNS_REP_DS.

```
#include <tau_dnr_types.h>
```

Collaboration diagram for TRDP_DNS_REPLY:



• TRDP_SHORT_VERSION_T version

1.0

• TRDP_NET_LABEL_T deviceName

function device of ED which sends the telegram

UINT32 etbTopoCnt

ETB topography counter.

UINT32 opTrnTopoCnt

operational train topography counter needed for TCN-URIs related to the operational train view = 0 if not used

UINT8 etbld

identification of the related ETB 0 = ETB0 (operational network) 1 = ETB1 (multimedia network) 2 = ETB2 (other network) 3 = ETB3 (other network) 255 = don't care (for access to local DNS server)

INT8 dnsStatus

0 = OK -1 = DNS Server not ready -2 = Inauguration in progress

UINT8 tcnUriCnt

number of TCN-URIs to be resolved value range: 0.

TCN_URI_T tcnUriList [255]

defined for max size

• TRDP_ETB_CTRL_VDP_T safetyTrail

SDT trailer.

4.17.1 Detailed Description

TCN-DNS Reply telegram TCN_DNS_REP_DS.

4.17.2 Field Documentation

4.17.2.1 tcnUriCnt

```
UINT8 TRDP_DNS_REPLY::tcnUriCnt
```

number of TCN-URIs to be resolved value range: 0 .

. 255

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

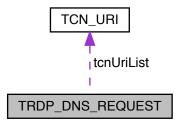
tau_dnr_types.h

4.18 TRDP_DNS_REQUEST Struct Reference

TCN-DNS Request telegram TCN_DNS_REQ_DS.

#include <tau_dnr_types.h>

Collaboration diagram for TRDP_DNS_REQUEST:



Data Fields

• TRDP_SHORT_VERSION_T version

1.0

TRDP_NET_LABEL_T deviceName

function device of ED which sends the telegram

UINT32 etbTopoCnt

ETB topography counter.

UINT32 opTrnTopoCnt

operational train topography counter needed for TCN-URIs related to the operational train view = 0 if not used

UINT8 etbld

identification of the related ETB 0 = ETB0 (operational network) 1 = ETB1 (multimedia network) 2 = ETB2 (other network) 3 = ETB3 (other network) 255 = don't care (for access to local DNS server)

UINT8 tcnUriCnt

number of TCN-URIs to be resolved value range: 0.

• TCN_URI_T tcnUriList [255]

defined for max size

• TRDP_ETB_CTRL_VDP_T safetyTrail

SDT trailer.

4.18.1 Detailed Description

TCN-DNS Request telegram TCN_DNS_REQ_DS.

4.18.2 Field Documentation

4.18.2.1 tcnUriCnt

```
UINT8 TRDP_DNS_REQUEST::tcnUriCnt
```

number of TCN-URIs to be resolved value range: 0.

. 255

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

· tau_dnr_types.h

4.19 TRDP_ETB_INFO_T Struct Reference

Types for train configuration information.

```
#include <tau_tti_types.h>
```

Data Fields

• UINT8 etbld

identification of train backbone; value range: 0..3

UINT8 cnCnt

number of CNs within consist connected to this ETB value range 1..16 referring to cnld 0..15 acc.

• UINT16 reserved01

reserved for future use (= 0)

4.19.1 Detailed Description

Types for train configuration information.

ETB information

4.19.2 Field Documentation

4.19.2.1 cnCnt

```
UINT8 TRDP_ETB_INFO_T::cnCnt
```

number of CNs within consist connected to this ETB value range 1..16 referring to cnld 0..15 acc.

IEC61375-2-5

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

tau_tti_types.h

4.20 TRDP_FUNCTION_INFO_T Struct Reference

function/device information structure

```
#include <tau_tti_types.h>
```

Data Fields

· TRDP NET LABEL T fctName

function device or group label

· UINT16 fctld

host identification of the function device or group as defined in IEC 61375-2-5, application defined.

BOOL8 grp

is a function group and will be resolved as IP multicast address

UINT8 reserved01

reserved for future use (= 0)

UINT8 cstVehNo

Sequence number of the vehicle in the consist the function belongs to.

UINT8 etbld

number of connected train backbone.

UINT8 cnld

identifier of connected consist network in the consist, related to the etbld.

• UINT8 reserved02

reserved for future use (= 0)

4.20.1 Detailed Description

function/device information structure

4.20.2 Field Documentation

```
4.20.2.1 cnld
```

```
UINT8 TRDP_FUNCTION_INFO_T::cnId
```

identifier of connected consist network in the consist, related to the etbld.

Value range: 0..31

4.20.2.2 cstVehNo

```
UINT8 TRDP_FUNCTION_INFO_T::cstVehNo
```

Sequence number of the vehicle in the consist the function belongs to.

Value range: 1..16, 0 = not defined

4.20.2.3 etbld

```
UINT8 TRDP_FUNCTION_INFO_T::etbId
```

number of connected train backbone.

Value range: 0..3

4.20.2.4 fctld

```
UINT16 TRDP_FUNCTION_INFO_T::fctId
```

host identification of the function device or group as defined in IEC 61375-2-5, application defined.

Value range: 1..16383 (device), 256..16383 (group)

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

· tau_tti_types.h

4.21 TRDP_HANDLE Struct Reference

Hidden handle definition, used as unique addressing item.

```
#include <trdp_private.h>
```

Data Fields

UINT32 comld

comld for packets to send/receive

• TRDP_IP_ADDR_T srclpAddr

source IP for PD/MD

• TRDP_IP_ADDR_T srclpAddr2

second source IP for PD/MD

TRDP_IP_ADDR_T destIpAddr

destination IP for PD

TRDP_IP_ADDR_T mcGroup

multicast group to join for PD

UINT32 etbTopoCnt

etb topocount belongs to addressing item

UINT32 opTrnTopoCnt

opTrn topocount belongs to addressing item

UINT32 serviceId

group of services this packet belongs to

4.21.1 Detailed Description

Hidden handle definition, used as unique addressing item.

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

• trdp_private.h

4.22 TRDP_IDX_TABLE_T Struct Reference

Settings for pre-allocation of index tables for application session initialization.

```
#include <trdp_types.h>
```

Data Fields

UINT32 maxNoOfLowCatSubscriptions

Мах.

UINT32 maxNoOfMidCatSubscriptions

Мах

UINT32 maxNoOfHighCatSubscriptions

Max

UINT32 maxNoOfLowCatPublishers

Мах.

UINT32 maxDepthOfLowCatPublishers

depth / overlapped publishers with intervals <= 100ms

UINT32 maxNoOfMidCatPublishers

Мах.

• UINT32 maxDepthOfMidCatPublishers

depth / overlapped publishers with intervals <= 1000ms

UINT32 maxNoOfHighCatPublishers

Мах.

• UINT32 maxDepthOfHighCatPublishers

depth / overlapped publishers with intervals <= 10000ms

UINT32 maxNoOfExtPublishers

Мах.

4.22.1 Detailed Description

Settings for pre-allocation of index tables for application session initialization.

4.22.2 Field Documentation

4.22.2.1 maxNoOfExtPublishers

UINT32 TRDP_IDX_TABLE_T::maxNoOfExtPublishers

Max.

number of expected publishers with intervals > 10000ms

4.22.2.2 maxNoOfHighCatPublishers

UINT32 TRDP_IDX_TABLE_T::maxNoOfHighCatPublishers

Max.

number of expected publishers with intervals <= 10000ms

4.22.2.3 maxNoOfHighCatSubscriptions

UINT32 TRDP_IDX_TABLE_T::maxNoOfHighCatSubscriptions

Max.

number of expected subscriptions with intervals > 1000ms

4.22.2.4 maxNoOfLowCatPublishers

UINT32 TRDP_IDX_TABLE_T::maxNoOfLowCatPublishers

Max.

number of expected publishers with intervals <= 100ms

4.22.2.5 maxNoOfLowCatSubscriptions

UINT32 TRDP_IDX_TABLE_T::maxNoOfLowCatSubscriptions

Max.

number of expected subscriptions with intervals \leq = 100ms

4.22.2.6 maxNoOfMidCatPublishers

UINT32 TRDP_IDX_TABLE_T::maxNoOfMidCatPublishers

Max.

number of expected publishers with intervals \leq = 1000ms

4.22.2.7 maxNoOfMidCatSubscriptions

UINT32 TRDP_IDX_TABLE_T::maxNoOfMidCatSubscriptions

Max.

number of expected subscriptions with intervals <= 1000ms

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

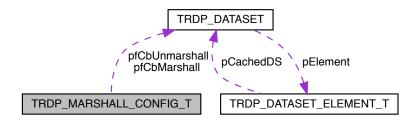
trdp_types.h

4.23 TRDP_MARSHALL_CONFIG_T Struct Reference

Marshaling/unmarshalling configuration.

```
#include <trdp_types.h>
```

Collaboration diagram for TRDP_MARSHALL_CONFIG_T:



Data Fields

TRDP_MARSHALL_T pfCbMarshall

Pointer to marshall callback function.

TRDP_UNMARSHALL_T pfCbUnmarshall

Pointer to unmarshall callback function.

void * pRefCon

Pointer to user context for call back.

4.23.1 Detailed Description

Marshaling/unmarshalling configuration.

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

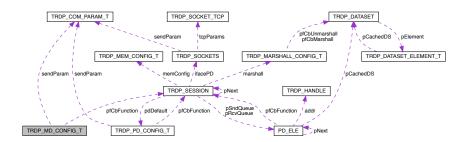
trdp_types.h

4.24 TRDP_MD_CONFIG_T Struct Reference

Default MD configuration.

#include <trdp_types.h>

Collaboration diagram for TRDP_MD_CONFIG_T:



Data Fields

TRDP_MD_CALLBACK_T pfCbFunction

Pointer to MD callback function.

void * pRefCon

Pointer to user context for call back.

TRDP_SEND_PARAM_T sendParam

Default send parameters.

TRDP_FLAGS_T flags

Default flags for MD packets.

UINT32 replyTimeout

Default reply timeout in us.

UINT32 confirmTimeout

Default confirmation timeout in us.

UINT32 connectTimeout

Default connection timeout in us.

UINT32 sendingTimeout

Default sending timeout in us.

UINT16 udpPort

Port to be used for UDP MD communication (default: 17225)

UINT16 tcpPort

Port to be used for TCP MD communication (default: 17225)

• UINT32 maxNumSessions

Maximal number of replier sessions.

4.24.1 Detailed Description

Default MD configuration.

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

trdp_types.h

4.25 TRDP_MD_INFO_T Struct Reference

Message data info from received telegram; allows the application to generate responses.

```
#include <trdp_types.h>
```

Data Fields

• TRDP_IP_ADDR_T srclpAddr

source IP address for filtering

TRDP_IP_ADDR_T destlpAddr

destination IP address for filtering

UINT32 seqCount

sequence counter

UINT16 protVersion

Protocol version.

TRDP_MSG_T msgType

Protocol ('PD', 'MD', ...)

UINT32 comld

ComID.

UINT32 etbTopoCnt

received topocount

UINT32 opTrnTopoCnt

received topocount

BOOL8 aboutToDie

session is about to die

UINT32 numRepliesQuery

number of ReplyQuery received

UINT32 numConfirmSent

number of Confirm sent

UINT32 numConfirmTimeout

number of Confirm Timeouts (incremented by listeners

• UINT16 userStatus

error code, user stat

TRDP_REPLY_STATUS_T replyStatus

reply status

· TRDP UUID T sessionId

for response

UINT32 replyTimeout

reply timeout in us given with the request

TRDP_URI_USER_T srcUserURI

source URI user part from MD header

TRDP_URI_HOST_T srcHostURI

source URI host part (unused)

TRDP_URI_USER_T destUserURI

destination URI user part from MD header

TRDP_URI_HOST_T destHostURI

destination URI host part (unused)

UINT32 numExpReplies

number of expected replies, 0 if unknown

UINT32 numReplies

actual number of replies for the request

const void * pUserRef

User reference given with the local call.

 TRDP_ERR_T resultCode error code

4.25.1 Detailed Description

Message data info from received telegram; allows the application to generate responses.

Note: Not all fields are relevant for each message type!

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

• trdp_types.h

4.26 TRDP_MEM_CONFIG_T Struct Reference

Enumeration type for memory pre-fragmentation, reuse of VOS definition.

```
#include <trdp_types.h>
```

Data Fields

UINT8 * p

pointer to static or allocated memory

• UINT32 size

size of static or allocated memory

UINT32 prealloc [VOS_MEM_NBLOCKSIZES]

memory block structure

4.26.1 Detailed Description

Enumeration type for memory pre-fragmentation, reuse of VOS definition.

Structure describing memory (and its pre-fragmentation)

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

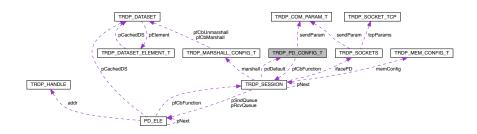
trdp_types.h

4.27 TRDP_PD_CONFIG_T Struct Reference

Default PD configuration.

#include <trdp_types.h>

Collaboration diagram for TRDP_PD_CONFIG_T:



Data Fields

TRDP_PD_CALLBACK_T pfCbFunction

Pointer to PD callback function.

void * pRefCon

Pointer to user context for call back.

• TRDP_SEND_PARAM_T sendParam

Default send parameters.

• TRDP_FLAGS_T flags

Default flags for PD packets.

UINT32 timeout

Default timeout in us.

• TRDP_TO_BEHAVIOR_T toBehavior

Default timeout behavior.

UINT16 port

Port to be used for PD communication (default: 17224)

4.27.1 Detailed Description

Default PD configuration.

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

· trdp_types.h

4.28 TRDP_PD_INFO_T Struct Reference

Process data info from received telegram; allows the application to generate responses.

#include <trdp_types.h>

• TRDP_IP_ADDR_T srclpAddr

source IP address for filtering

• TRDP_IP_ADDR_T destlpAddr

destination IP address for filtering

UINT32 seqCount

sequence counter

UINT16 protVersion

Protocol version.

TRDP_MSG_T msgType

Protocol ('PD', 'MD', ...)

UINT32 comld

ComID.

UINT32 etbTopoCnt

received ETB topocount

UINT32 opTrnTopoCnt

received operational train directory topocount

UINT32 replyComId

ComID for reply (request only)

• TRDP_IP_ADDR_T replyIpAddr

IP address for reply (request only)

const void * pUserRef

User reference given with the local subscribe.

• TRDP ERR T resultCode

error code

TRDP_URI_HOST_T srcHostURI

source URI host part (unused)

TRDP_URI_HOST_T destHostURI

destination URI host part (unused)

TRDP_TO_BEHAVIOR_T toBehavior

callback can decide about handling of data on timeout

UINT32 serviceId

the reserved field of the PD header

4.28.1 Detailed Description

Process data info from received telegram; allows the application to generate responses.

Note: Not all fields are relevant for each message type!

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

• trdp_types.h

4.29 TRDP_PROCESS_CONFIG_T Struct Reference

Various flags/general TRDP options for library initialization.

#include <trdp_types.h>

• TRDP_LABEL_T hostName

Host name.

• TRDP_LABEL_T leaderName

Leader name dependant on redundancy concept.

UINT32 cycleTime

TRDP main process cycle time in us.

· UINT32 priority

TRDP main process priority (0-255, 0=default, 255=highest)

• TRDP_OPTION_T options

TRDP options.

4.29.1 Detailed Description

Various flags/general TRDP options for library initialization.

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

• trdp_types.h

4.30 TRDP_PROP_T Struct Reference

Application defined properties.

```
#include <tau_tti_types.h>
```

Data Fields

TRDP_SHORT_VERSION_T ver

properties version information, application defined

• UINT16 len

properties length in number of octets, application defined, must be a multiple of 4 octets for alignment reasons value range: 0..32768

• UINT8 prop [1]

properties, application defined

4.30.1 Detailed Description

Application defined properties.

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

tau_tti_types.h

4.31 TRDP_SDT_PAR_T Struct Reference

Types to read out the XML configuration.

```
#include <tau_xml.h>
```

Data Fields

UINT32 smi1

Safe message identifier - unique for this message at consist level.

UINT32 smi2

Safe message identifier - unique for this message at consist level.

UINT32 cmThr

Channel monitoring threshold.

UINT16 udv

User data version.

UINT16 rxPeriod

Sink cycle time.

UINT16 txPeriod

Source cycle time.

UINT16 nGuard

Initial timeout cycles.

UINT8 nrxSafe

Timout cycles.

UINT8 reserved1

Reserved for future use.

UINT16 ImiMax

Latency monitoring cycles.

4.31.1 Detailed Description

Types to read out the XML configuration.

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

• tau_xml.h

4.32 TRDP_SEQ_CNT_ENTRY_T Struct Reference

Tuples of last received sequence counter per comld.

```
#include <trdp_private.h>
```

UINT32 lastSeqCnt

Sequence counter value for comld.

• TRDP_IP_ADDR_T srclpAddr

Source IP address.

TRDP_MSG_T msgType

message type

4.32.1 Detailed Description

Tuples of last received sequence counter per comld.

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

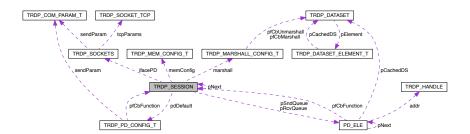
• trdp_private.h

4.33 TRDP_SESSION Struct Reference

Session/application variables store.

#include <trdp_private.h>

Collaboration diagram for TRDP_SESSION:



Data Fields

struct TRDP SESSION * pNext

Pointer to next session.

VOS_MUTEX_T mutex

protect this session

VOS_MUTEX_T mutexTxPD

protect the sending queue

VOS_MUTEX_T mutexRxPD

protect the receiving queue

TRDP_IP_ADDR_T realIP

Real IP address.

• TRDP_IP_ADDR_T virtualIP

Virtual IP address.

UINT32 etbTopoCnt

current valid topocount or zero

UINT32 opTrnTopoCnt

current valid topocount or zero

TRDP TIME T nextJob

Store for next select interval.

• TRDP_PRINT_DBG_T pPrintDebugString

Pointer to function to print debug information.

TRDP_MARSHALL_CONFIG_T marshall

Marshalling(unMarshalling configuration.

• TRDP_PD_CONFIG_T pdDefault

Default configuration for process data.

• TRDP_MEM_CONFIG_T memConfig

Internal memory handling configuration.

• TRDP_OPTION_T option

Stack behavior options.

TRDP_SOCKETS_T ifacePD [TRDP_MAX_PD_SOCKET_CNT]

Collection of sockets to use.

• PD ELE T * pSndQueue

pointer to first element of send queue

• PD_ELE_T * pRcvQueue

pointer to first element of rcv queue

PD_PACKET_T * pNewFrame

pointer to received PD frame

• TRDP_TIME_T initTime

initialization time of session

· TRDP STATISTICS T stats

statistics of this session

4.33.1 Detailed Description

Session/application variables store.

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

• trdp_private.h

4.34 TRDP_SOCKET_TCP Struct Reference

TCP parameters.

#include <trdp_private.h>

• TRDP_IP_ADDR_T cornerlp

The other TCP corner Ip.

BOOL8 notSend

If the message has been sent uncompleted.

• TRDP_TIME_T connectionTimeout

TCP socket connection Timeout.

BOOL8 sendNotOk

The sending timeout will be start.

• TRDP_TIME_T sendingTimeout

The timeout sending the message.

• BOOL8 addFileDesc

Ready to add the socket in the fd.

• BOOL8 morituri

about to die

4.34.1 Detailed Description

TCP parameters.

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

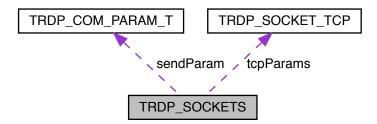
• trdp_private.h

4.35 TRDP_SOCKETS Struct Reference

Socket item.

#include <trdp_private.h>

Collaboration diagram for TRDP_SOCKETS:



SOCKET sock

vos socket descriptor to use

• TRDP_IP_ADDR_T bindAddr

Defines the interface to use.

• TRDP_IP_ADDR_T srcAddr

Defines the source interface to use.

• TRDP_SEND_PARAM_T sendParam

Send parameters.

• TRDP_SOCK_TYPE_T type

Usage of this socket.

BOOL8 rcvMostly

Used for receiving.

• INT16 usage

No

• TRDP_SOCKET_TCP_T tcpParams

Params used for TCP.

TRDP_IP_ADDR_T mcGroups [VOS_MAX_MULTICAST_CNT]

List of multicast addresses for this socket.

4.35.1 Detailed Description

Socket item.

4.35.2 Field Documentation

4.35.2.1 usage

INT16 TRDP_SOCKETS::usage

No.

of current users of this socket

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

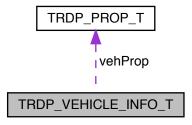
trdp_private.h

4.36 TRDP_VEHICLE_INFO_T Struct Reference

vehicle information structure

#include <tau_tti_types.h>

Collaboration diagram for TRDP_VEHICLE_INFO_T:



Data Fields

• TRDP_NET_LABEL_T vehId

vehicle identifier label, application defined (e.g.

• TRDP_NET_LABEL_T vehType

vehicle type,application defined

UINT8 vehOrient

vehicle orientation '01'B = same as consist direction '10'B = inverse to consist direction

UINT8 cstVehNo

Sequence number of vehicle in consist(1..16)

ANTIVALENT8 tractVeh

vehicle is a traction vehicle '01'B = vehicle is not a traction vehicle '10'B = vehicle is a traction vehicle

UINT8 reserved01

for future use (= 0)

TRDP_PROP_T vehProp

static vehicle properties

4.36.1 Detailed Description

vehicle information structure

4.36.2 Field Documentation

4.36.2.1 vehld

```
TRDP_NET_LABEL_T TRDP_VEHICLE_INFO_T::vehId
```

vehicle identifier label, application defined (e.g.

UIC vehicle identification number) vehId of vehicle with vehNo==1 is used also as cstId

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

• tau_tti_types.h

4.37 TRDP_XML_DOC_HANDLE_T Struct Reference

Parsed XML document handle.

```
#include <tau_xml.h>
```

Data Fields

struct XML_HANDLE * pXmlDocument
 XML document context.

4.37.1 Detailed Description

Parsed XML document handle.

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

• tau_xml.h

4.38 VOS_SOCK_OPT_T Struct Reference

Common socket options.

#include <vos_sock.h>

UINT8 gos

quality/type of service 0...7

• UINT8 ttl

time to live for unicast (default 64)

UINT8 ttl_multicast

time to live for multicast

BOOL8 reuseAddrPort

allow reuse of address and port

· BOOL8 nonBlocking

use non blocking calls

• BOOL8 no_mc_loop

no multicast loop back

BOOL8 no_udp_crc

supress udp crc computation

BOOL8 txTime

use transmit time on send, if available

BOOL8 raw

use raw socket, not for receiver!

CHAR8 ifName [VOS_MAX_IF_NAME_SIZE]

interface name if available

4.38.1 Detailed Description

Common socket options.

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

· vos_sock.h

4.39 VOS_VERSION_T Struct Reference

Version information.

```
#include <vos_types.h>
```

Data Fields

• UINT8 ver

Version - incremented for incompatible changes.

UINT8 rel

Release - incremented for compatible changes.

UINT8 upd

Update - incremented for bug fixes.

UINT8 evo

Evolution - incremented for build.

4.39.1 Detailed Description

Version information.

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

· vos_types.h

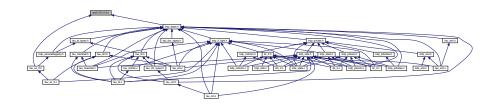
Chapter 5

File Documentation

5.1 iec61375-2-3.h File Reference

All definitions from IEC 61375-2-3.

This graph shows which files directly or indirectly include this file:



Macros

- #define ETB_WAIT_TIMER_VALUE 5u /* Compute train dir. IEC61375-2-3 Ch. 5.3.2.3 */
 Time out values (in seconds)
- #define TRDP_PD_UDP_PORT 17224u

TRDP defines (from former trpd_proto.h)

#define TRDP MD UDP PORT 17225u

IANA assigned message data UDP port.

#define TRDP_MD_TCP_PORT 17225u

IANA assigned message data TCP port.

• #define TRDP_PROTO_VER 0x0100u

Protocol version.

• #define TRDP_PROTOCOL_VERSION_CHECK_MASK 0xFF00u

Version check, two digits are relevant.

• #define TRDP_SESS_ID_SIZE 16u

Session ID (UUID) size in MD header.

• #define TRDP_USR_URI_SIZE 32u

тах.

• #define TRDP_MD_INFINITE_TIME (0)

Definitions for time out behaviour accd.

70 File Documentation

• #define TRDP_MD_DEFAULT_REPLY_TIMEOUT 5000000u Default MD communication parameters. #define TRDP MD DEFAULT CONFIRM TIMEOUT 1000000u [us] default confirm time out 1s #define TRDP_MD_DEFAULT_CONNECTION_TIMEOUT 60000000u [us] Socket connection time out 1min #define TRDP MD DEFAULT SENDING TIMEOUT 5000000u [us] Socket sending time out 5s #define TRDP PD DEFAULT QOS 5u Default PD communication parameters. #define TRDP PD DEFAULT TIMEOUT 100000u [us] 100ms default PD timeout #define TRDP_PROCESS_DEFAULT_CYCLE_TIME 10000u Default TRDP process options. #define TRDP PROCESS DEFAULT PRIORITY 64u Default priority of TRDP process. #define TRDP_PROCESS_DEFAULT_OPTIONS TRDP_OPTION_TRAFFIC_SHAPING Default options for TRDP process. #define TRDP_MIN_PD_HEADER_SIZE sizeof(PD_HEADER_T) PD packet properties. #define TRDP_MAX_PD_DATA_SIZE 1432u PD data. #define TRDP MAX MD DATA SIZE 65388u MD packet properties. #define TRDP_MAX_MD_RETRIES 2u Maximum values. • #define TRDP MAX LABEL LEN 16u label length incl. #define TRDP_MAX_URI_USER_LEN (2u * TRDP_MAX_LABEL_LEN) URI user part incl. #define TRDP_MAX_URI_HOST_LEN (5u * TRDP_MAX_LABEL_LEN) URI host part incl. • #define TRDP_MAX_URI_LEN (7u * TRDP_MAX_LABEL_LEN) URI length incl. #define TRDP MAX FILE NAME LEN 128u path and file name length incl. • #define TRDP VAR SIZE 0u Variable size dataset. #define TRDP_MSG_PD 0x5064u Message Types. #define TRDP MSG PP 0x5070u 'Pp' PD Data (Pull Reply) #define TRDP_MSG_PR 0x5072u 'Pr' PD Request • #define TRDP MSG PE 0x5065u 'Pe' PD Error #define TRDP MSG MN 0x4D6Eu 'Mn' MD Notification (Request w/o reply) #define TRDP MSG MR 0x4D72u 'Mr' MD Request with reply

#define TRDP_MSG_MP 0x4D70u

'Mp' MD Reply without confirmation #define TRDP_MSG_MQ 0x4D71u 'Mq' MD Reply with confirmation • #define TRDP MSG MC 0x4D63u 'Mc' MD Confirm #define TRDP MSG ME 0x4D65u 'Me' MD Error #define ETB0 ALL END DEVICES IP "239.193.0.0" from Table 22 • #define ETB_CTRL_COMID 1u Reserved COMIDs in the range 1 ... • #define ETB_CTRL_CYCLE 500000u [us] 0.5s • #define ETB_CTRL_TO_US 300000u [us] 3s #define TRDP ETBCTRL COMID ETB CTRL COMID alternative name #define CSTINFO COMID 2u Consist Info telegram (Message data notification 'Mn') #define TRDP CSTINFO COMID CSTINFO COMID alternative name #define CSTINFOCTRL_COMID 3u Consist Info control/request telegram (Message data notification 'Mn') #define TRDP_CSTINFOCTRL_COMID CSTINFOCTRL_COMID alternative name • #define TRDP COMID ECHO 10u Reserved in Annex D & E. #define TRDP_STATISTICS_PULL_COMID 31u reserved in Table A.2 #define TRDP GLOBAL STATS REPLY COMID 31u reserved in D.3 #define TTDB STATUS COMID 100u TTDB manager telegram PD. #define TTDB_STATUS_CYCLE 1000000u [us] 1s Push #define TTDB_STATUS_TO_US 5000000u • #define TTDB_OP_DIR_INFO_COMID 101u TTDB manager telegram MD: Push the OP_TRAIN_DIRECTORY. #define TTDB_OP_DIR_INFO_DS "TTDB_OP_TRAIN_DIRECTORY_INFO" OP_TRAIN_DIRECTORY. • #define TTDB TRN DIR REQ COMID 102u TTDB manager telegram MD: Get the TRAIN_DIRECTORY. #define TTDB_TRN_DIR_REQ_TO_US 3000000u 3s timeout #define TTDB_TRN_DIR_REP_COMID 103u

#define TTDB_TRN_DIR_REP_DS "TTDB_TRAIN_DIRECTORY_INFO_REPLY"

TTDB manager telegram MD: Get the static consist information.

Generated by Doxygen

TRAIN_DIRECTORY.

• #define TTDB_STAT_CST_REQ_COMID 104u

72 File Documentation

```
    #define TTDB_STAT_CST_REQ_TO_US 3000000u

    [us] 3s timeout
• #define TTDB STAT CST REP DS "TTDB STATIC CONSIST INFO REPLY"
    CONSIST INFO.
• #define TTDB_NET_DIR_REQ_COMID 106u
     TTDB manager telegram MD: Get the NETWORK_TRAIN_DIRECTORY.

    #define TTDB NET DIR REQ TO US 3000000u

    [us] 3s timeout

    #define TTDB_NET_DIR_REP_COMID 107u

    MD reply.

    #define TTDB NET DIR REP DS "TTDB TRAIN NETWORK DIRECTORY INFO REPLY"

     TRAIN_NETWORK_DIRECTORY.

    #define TTDB_OP_DIR_INFO_REQ_COMID 108u

     TTDB manager telegram MD: Get the OP_TRAIN_DIRECTORY.
• #define TTDB OP DIR INFO REQ TO US 3000000u
    [us] 3s timeout

    #define TTDB_OP_DIR_INFO_REP_DS "TTDB_OP_TRAIN_DIR_INFO"

    OP TRAIN DIRECTORY.
• #define TTDB_READ_CMPLT_REQ_COMID 110u
     TTDB manager telegram MD: Get the TTDB.

    #define TTDB_READ_CMPLT_REQ_DS "TTDB_READ_COMPLETE_REQUEST"

    FTBx.

    #define TTDB READ CMPLT REQ TO US 3000000u

    [us] 3s timeout

    #define TTDB_READ_CMPLT_REP_COMID 111u

    MD reply.

    #define TTDB READ CMPLT REP DS "TTDB READ COMPLETE REPLY"

     TRDP_READ_COMPLETE_REPLY_T.
• #define ECSP_CTRL_COMID 120u
    ECSP Control telegram.

    #define ECSP_CTRL_CYCLE 1000000u

    [us] 1s
• #define ECSP_CTRL_TO_US 5000000u
    [us] 5s

    #define ECSP CTRL DEST URI "devECSP.anyVeh.ICst.ICITrn.ITrn"

     10.0.0.1

    #define TRDP ECSP CTRL COMID ECSP CTRL COMID

    Etb control message.

    #define ECSP_STATUS_COMID 121u

    ECSP status telegram.

    #define ECSP STATUS CYCLE 1000000u

    [us] 1s

    #define ECSP_STATUS_TO_US 5000000u

    #define ECSP STATUS DEST URI "devECSC.anyVeh.ICst.ICITrn.ITrn"

     10.0.0.100

    #define ECSP CONF REQ COMID 122u

    ECSP Confirmation Request telegram MD:

    #define ECSP CONF REQ TO US 3000000u
```

#define ECSP_CONF_REQ_URI "devECSP.anyVeh.ICst.ICITrn.ITrn"

10.0.0.1

• #define ECSP_CONF_REP_TO_US 3000000u

[usi

#define ETBN_CTRL_REQ_COMID 130u

ETBN Control & Status Telegram MD.

#define ETBN_CTRL_REQ_DS "ETBN_CTRL"

• #define ETBN_CTRL_REQ_TO_US 3000000u

[us] 3s timeout

• #define ETBN_CTRL_REP_DS "ETBN_STATUS"

ETBN status reply.

#define ETBN_TRN_NET_DIR_REQ_COMID 132u

ETBN Control Telegram MD.

#define ETBN_TRN_NET_DIR_REQ_TO_US 3000000u

[us] 3s timeout

• #define TCN_DNS_REQ_COMID 140u

TCN-DNS Request Telegram MD.

#define TCN_DNS_REQ_TO_US 3000000u

[us] 3s timeout

• #define TRDP_ETBCTRL_DSID 1u

TRDP reserved data set ids in the range 1 ...

5.1.1 Detailed Description

All definitions from IEC 61375-2-3.

Note

Project: TCNOpen TRDP

Author

Bernd Loehr, NewTec GmbH, 2015-09-11

Remarks

This Source Code Form is subject to the terms of the Mozilla Public License, v. 2.0. If a copy of the MPL was not distributed with this file, You can obtain one at http://mozilla.org/MPL/2.0/.

5.1.2 Macro Definition Documentation

5.1.2.1 ETB_CTRL_COMID

#define ETB_CTRL_COMID 1u

Reserved COMIDs in the range 1 ...

1000 ETB Control telegram

74 File Documentation

```
5.1.2.2 TRDP_ETBCTRL_DSID
#define TRDP_ETBCTRL_DSID 1u
TRDP reserved data set ids in the range 1 ...
1000
5.1.2.3 TRDP_MAX_FILE_NAME_LEN
#define TRDP_MAX_FILE_NAME_LEN 128u
path and file name length incl.
terminating '0'
5.1.2.4 TRDP_MAX_LABEL_LEN
#define TRDP_MAX_LABEL_LEN 16u
label length incl.
terminating '0'
5.1.2.5 TRDP_MAX_MD_DATA_SIZE
#define TRDP_MAX_MD_DATA_SIZE 65388u
MD packet properties.
MD payload size
5.1.2.6 TRDP_MAX_URI_HOST_LEN
#define TRDP_MAX_URI_HOST_LEN (5u * TRDP_MAX_LABEL_LEN)
URI host part incl.
terminating '0'
5.1.2.7 TRDP_MAX_URI_LEN
#define TRDP_MAX_URI_LEN (7u * TRDP_MAX_LABEL_LEN)
```

URI length incl.

'.', '@' and terminating '0'

5.1.2.8 TRDP_MAX_URI_USER_LEN

#define TRDP_MAX_URI_USER_LEN (2u * TRDP_MAX_LABEL_LEN)

URI user part incl.

'.' and terminating '0'

5.1.2.9 TRDP_MD_DEFAULT_REPLY_TIMEOUT

#define TRDP_MD_DEFAULT_REPLY_TIMEOUT 5000000u

Default MD communication parameters.

[us] default reply timeout 5s

5.1.2.10 TRDP_MD_INFINITE_TIME

#define TRDP_MD_INFINITE_TIME (0)

Definitions for time out behaviour accd.

table A.18

5.1.2.11 TRDP_MIN_PD_HEADER_SIZE

#define TRDP_MIN_PD_HEADER_SIZE sizeof(PD_HEADER_T)

PD packet properties.

PD header size with FCS

5.1.2.12 TRDP_MSG_PD

#define TRDP_MSG_PD 0x5064u

Message Types.

'Pd' PD Data

5.1.2.13 TRDP_PD_UDP_PORT

#define TRDP_PD_UDP_PORT 17224u

TRDP defines (from former trpd_proto.h)

IANA assigned process data UDP port

76 File Documentation

```
5.1.2.14 TRDP_PROCESS_DEFAULT_CYCLE_TIME
```

#define TRDP_PROCESS_DEFAULT_CYCLE_TIME 10000u

Default TRDP process options.

[us] 10ms cycle time for TRDP process

5.1.2.15 TRDP_USR_URI_SIZE

#define TRDP_USR_URI_SIZE 32u

max.

User URI size in MD header

5.1.2.16 TTDB_NET_DIR_REQ_COMID

#define TTDB_NET_DIR_REQ_COMID 106u

TTDB manager telegram MD: Get the NETWORK_TRAIN_DIRECTORY.

MD request

5.1.2.17 TTDB_OP_DIR_INFO_COMID

#define TTDB_OP_DIR_INFO_COMID 101u

TTDB manager telegram MD: Push the OP_TRAIN_DIRECTORY.

MD notification

5.1.2.18 TTDB_STAT_CST_REQ_COMID

#define TTDB_STAT_CST_REQ_COMID 104u

TTDB manager telegram MD: Get the static consist information.

MD request

5.1.2.19 TTDB_TRN_DIR_REQ_COMID

#define TTDB_TRN_DIR_REQ_COMID 102u

TTDB manager telegram MD: Get the TRAIN_DIRECTORY.

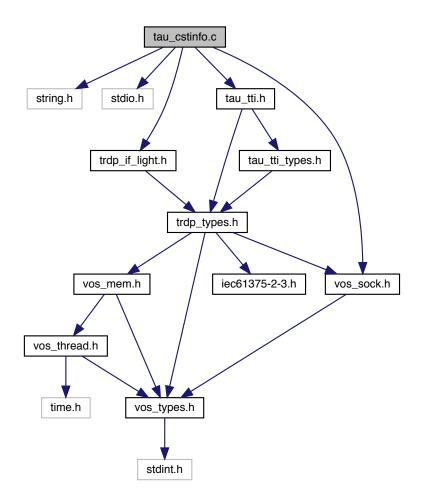
MD request

5.2 tau_cstinfo.c File Reference

Functions for consist information access.

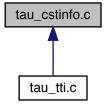
```
#include <string.h>
#include <stdio.h>
#include "trdp_if_light.h"
#include "tau_tti.h"
#include "vos_sock.h"
```

Include dependency graph for tau_cstinfo.c:



78 File Documentation

This graph shows which files directly or indirectly include this file:



Functions

• UINT16 cstInfoGetPropSize (TRDP_CONSIST_INFO_T *pCstInfo)

Getter function to retrieve a value from the consist info telegram value.

5.2.1 Detailed Description

Functions for consist information access.

Note

Project: TCNOpen TRDP prototype stack

Author

B. Loehr (initial version)

Remarks

This Source Code Form is subject to the terms of the Mozilla Public License, v. 2.0. If a copy of the MPL was not distributed with this file, You can obtain one at http://mozilla.org/MPL/2.0/. Copyright Bombardier Transportation Inc. or its subsidiaries and others, 2015. All rights reserved.

5.2.2 Function Documentation

5.2.2.1 cstInfoGetPropSize()

Getter function to retrieve a value from the consist info telegram value.

Parameters

in	pCstInfo	pointer to packed consist info in network byte order
----	----------	--

Return values



Here is the call graph for this function:

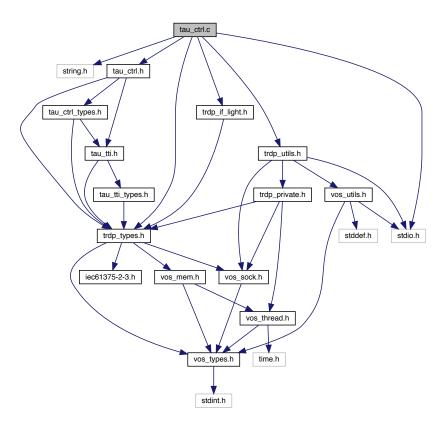


5.3 tau_ctrl.c File Reference

Functions for train switch control.

```
#include <string.h>
#include <stdio.h>
#include "trdp_types.h"
#include "trdp_utils.h"
#include "trdp_if_light.h"
#include "tau_ctrl.h"
```

Include dependency graph for tau_ctrl.c:



Functions

• EXT_DECL TRDP_ERR_T tau_initEcspCtrl (TRDP_APP_SESSION_T appHandle, TRDP_IP_ADDR_← TecsplpAddr)

Function to init ECSP control interface.

• EXT_DECL TRDP_ERR_T tau_terminateEcspCtrl (TRDP_APP_SESSION_T appHandle)

Function to close ECSP control interface.

EXT_DECL TRDP_ERR_T tau_setEcspCtrl (TRDP_APP_SESSION_T appHandle, TRDP_ECSP_CTRL_T *pEcspCtrl)

Function to set ECSP control information.

• EXT_DECL TRDP_ERR_T tau_getEcspStat (TRDP_APP_SESSION_T appHandle, TRDP_ECSP_STAT_T *pEcspStat, TRDP_PD_INFO_T *pPdInfo)

Function to get ECSP status information.

• EXT_DECL TRDP_ERR_T tau_requestEcspConfirm (TRDP_APP_SESSION_T appHandle, const void *pUserRef, TRDP_MD_CALLBACK_T pfCbFunction, TRDP_ECSP_CONF_REQUEST_T *pEcspConf← Request)

Function for ECSP confirmation/correction request, reply will be received via call back.

5.3.1 Detailed Description

Functions for train switch control.

Note

Project: TCNOpen TRDP prototype stack

Author

Armin-H. Weiss (initial version)

Remarks

This Source Code Form is subject to the terms of the Mozilla Public License, v. 2.0. If a copy of the MPL was not distributed with this file, You can obtain one at http://mozilla.org/MPL/2.0/. Copyright Bombardier Transportation Inc. or its subsidiaries and others, 2013-2019. All rights reserved.

5.3.2 Function Documentation

5.3.2.1 tau_getEcspStat()

Function to get ECSP status information.

Parameters

in	appHandle	Application handle
in,out	pEcspStat	Pointer to the ECSP status structure
in,out	pPdInfo	Pointer to PD status information

Return values

TRDP_NO_ERR	no error
TRDP_NOINIT_ERR	module not initialised
TRDP_PARAM_ERR	Parameter error

5.3.2.2 tau_initEcspCtrl()

Function to init ECSP control interface.

Parameters

in	appHandle	Application handle
in	ecsplpAddr	ECSP address

Return values

TRDP_NO_ERR	no error
TRDP_INIT_ERR	initialisation error

5.3.2.3 tau_requestEcspConfirm()

Function for ECSP confirmation/correction request, reply will be received via call back.

Parameters

in	appHandle	Application Handle
in	pUserRef	user reference returned with reply
in	pfCbFunction	Pointer to callback function, NULL for default
in	pEcspConfRequest	Pointer to confirmation data

Return values

TRDP_NO_ERR	no error
TRDP_NOINIT_ERR	module not initialised
TRDP_PARAM_ERR	Parameter error

5.3.2.4 tau_setEcspCtrl()

Function to set ECSP control information.

Parameters

in	appHandle	Application handle
in	pEcspCtrl	Pointer to the ECSP control structure

Return values

TRDP_NO_ERR	no error
TRDP_NOINIT_ERR	module not initialised
TRDP_PARAM_ERR	Parameter error

5.3.2.5 tau_terminateEcspCtrl()

Function to close ECSP control interface.

Parameters

in appHandle	Application handle
--------------	--------------------

Return values

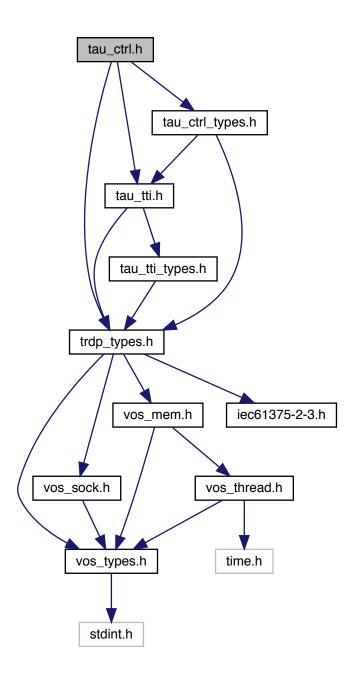
TRDP_NO_ERR	no error
TRDP_NOINIT_ERR	module not initialised
TRDP_UNKNOWN_ERR	undefined error

5.4 tau_ctrl.h File Reference

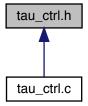
TRDP utility interface definitions.

```
#include "trdp_types.h"
#include "tau_tti.h"
#include "tau_ctrl_types.h"
```

Include dependency graph for tau_ctrl.h:



This graph shows which files directly or indirectly include this file:



Functions

• EXT_DECL TRDP_ERR_T tau_initEcspCtrl (TRDP_APP_SESSION_T appHandle, TRDP_IP_ADDR_← TecsplpAddr)

Function to init ECSP control interface.

• EXT_DECL TRDP_ERR_T tau_terminateEcspCtrl (TRDP_APP_SESSION_T appHandle)

Function to close ECSP control interface.

• EXT_DECL TRDP_ERR_T tau_setEcspCtrl (TRDP_APP_SESSION_T appHandle, TRDP_ECSP_CTRL_T *pEcspCtrl)

Function to set ECSP control information.

EXT_DECL TRDP_ERR_T tau_getEcspStat (TRDP_APP_SESSION_T appHandle, TRDP_ECSP_STAT_T *pEcspStat, TRDP_PD_INFO_T *pPdInfo)

Function to get ECSP status information.

• EXT_DECL TRDP_ERR_T tau_requestEcspConfirm (TRDP_APP_SESSION_T appHandle, const void *pUserRef, TRDP_MD_CALLBACK_T pfCbFunction, TRDP_ECSP_CONF_REQUEST_T *pEcspConf← Request)

Function for ECSP confirmation/correction request, reply will be received via call back.

5.4.1 Detailed Description

TRDP utility interface definitions.

This module provides the interface to the following utilities

· ETB control

Note

Project: TCNOpen TRDP prototype stack

Author

Armin-H. Weiss (initial version)

Remarks

This Source Code Form is subject to the terms of the Mozilla Public License, v. 2.0. If a copy of the MPL was not distributed with this file, You can obtain one at http://mozilla.org/MPL/2.0/. Copyright Bombardier Transportation Inc. or its subsidiaries and others, 2013. All rights reserved.

5.4.2 Function Documentation

5.4.2.1 tau_getEcspStat()

Function to get ECSP status information.

Parameters

in	appHandle	Application Handle
in,out	pEcspStat	Pointer to the ECSP status structure
in,out	pPdInfo	Pointer to PD status information

Return values

TRDP_NO_ERR	no error
TRDP_NOINIT_ERR	module not initialised
TRDP_PARAM_ERR	Parameter error

Parameters

in	appHandle	Application handle
in,out	pEcspStat	Pointer to the ECSP status structure
in,out	pPdInfo	Pointer to PD status information

Return values

TRDP_NO_ERR	no error
TRDP_NOINIT_ERR	module not initialised
TRDP_PARAM_ERR	Parameter error

5.4.2.2 tau_initEcspCtrl()

Function to init ECSP control interface.

Parameters

in	appHandle	Application handle
in	ecsplpAddr	ECSP address

Return values

TRDP_NO_ERR	no error
TRDP_INIT_ERR	initialisation error

5.4.2.3 tau_requestEcspConfirm()

Function for ECSP confirmation/correction request, reply will be received via call back.

Parameters

in	appHandle	Application Handle
in	pUserRef	user reference returned with reply
in	pfCbFunction	Pointer to callback function, NULL for default
in	pEcspConfRequest	Pointer to confirmation data

Return values

TRDP_NO_ERR	no error
TRDP_NOINIT_ERR	module not initialised
TRDP_PARAM_ERR	Parameter error

5.4.2.4 tau_setEcspCtrl()

Function to set ECSP control information.

Parameters

in	appHandle	Application handle
in	pEcspCtrl	Pointer to the ECSP control structure

Return values

TRDP_NO_ERR	no error
TRDP_NOINIT_ERR	module not initialised
TRDP_PARAM_ERR	Parameter error

5.4.2.5 tau_terminateEcspCtrl()

Function to close ECSP control interface.

Parameters

in	appHandle	Application handle
----	-----------	--------------------

Return values

TRDP_NO_ERR	no error
TRDP_UNKNOWN_ERR	undefined error

Parameters

in	appHandle	Application handle
----	-----------	--------------------

Return values

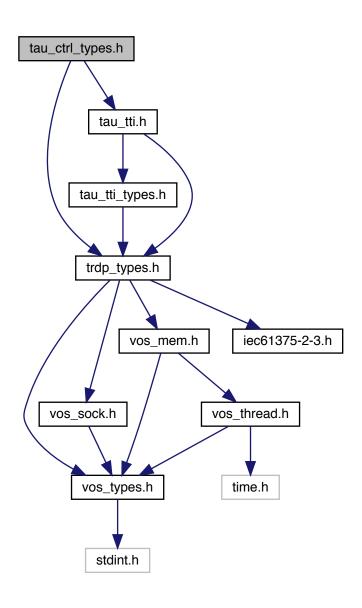
TRDP_NO_ERR	no error
TRDP_NOINIT_ERR	module not initialised
TRDP_UNKNOWN_ERR	undefined error

5.5 tau_ctrl_types.h File Reference

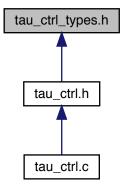
TRDP utility interface definitions.

```
#include "trdp_types.h"
#include "tau_tti.h"
```

Include dependency graph for tau_ctrl_types.h:



This graph shows which files directly or indirectly include this file:



Data Structures

struct GNU_PACKED

Types for ETB control.

• struct GNU_PACKED

Types for ETB control.

struct GNU_PACKED

Types for ETB control.

5.5.1 Detailed Description

TRDP utility interface definitions.

This module provides the interface to the following

• ETB control type definitions acc. to IEC61375-2-3

Note

Project: TCNOpen TRDP prototype stack

Author

Armin-H. Weiss (initial version)

Remarks

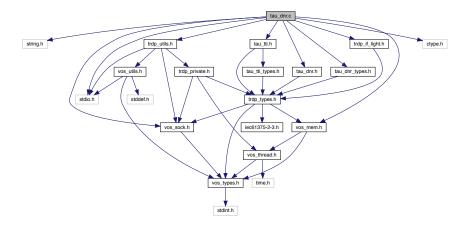
This Source Code Form is subject to the terms of the Mozilla Public License, v. 2.0. If a copy of the MPL was not distributed with this file, You can obtain one at http://mozilla.org/MPL/2.0/. Copyright Bombardier Transportation Inc. or its subsidiaries and others, 2013. All rights reserved.

5.6 tau_dnr.c File Reference

Functions for domain name resolution.

```
#include <string.h>
#include <stdio.h>
#include <ctype.h>
#include "tau_tti.h"
#include "tau_dnr.h"
#include "tau_dnr_types.h"
#include "trdp_utils.h"
#include "trdp_if_light.h"
#include "vos_mem.h"
#include "vos_sock.h"
```

Include dependency graph for tau_dnr.c:



Data Structures

• struct DNS_HEADER

DNS header structure.

Macros

• #define TAU MAX NO IF 4u

Default interface should be in the first 4.

· #define TAU DNS TIME OUT LONG 10u

Timeout in seconds for DNS server reply, if no hosts file provided.

#define TAU_DNS_TIME_OUT_SHORT 1u

Timeout in seconds for DNS server reply, if hosts file was provided.

Typedefs

typedef struct DNS_HEADER TAU_DNS_HEADER_T

DNS header structure.

Functions

• EXT_DECL TRDP_ERR_T tau_initDnr (TRDP_APP_SESSION_T appHandle, TRDP_IP_ADDR_T dnsIp← Addr, UINT16 dnsPort, const CHAR8 *pHostsFileName, TRDP_DNR_OPTS_T dnsOptions, BOOL8 wait← ForDnr)

Function to init the DNR subsystem Initialize the DNR resolver.

EXT_DECL void tau_deInitDnr (TRDP_APP_SESSION_T appHandle)

Function to deinit DNR.

• EXT_DECL TRDP_DNR_STATE_T tau_DNRstatus (TRDP_APP_SESSION_T appHandle)

Function to get the status of DNR.

EXT_DECL TRDP_IP_ADDR_T tau_getOwnAddr (TRDP_APP_SESSION_T appHandle)

Function to get the own IP address.

EXT_DECL TRDP_ERR_T tau_uri2Addr (TRDP_APP_SESSION_T appHandle, TRDP_IP_ADDR_T *p
 — Addr, const TRDP_URI_T pUri)

Function to convert a URI to an IP address.

• EXT_DECL TRDP_ERR_T tau_addr2Uri (TRDP_APP_SESSION_T appHandle, TRDP_URI_HOST_T pUri, TRDP_IP_ADDR_T addr)

Function to convert an IP address to a URI.

5.6.1 Detailed Description

Functions for domain name resolution.

Note

Project: TCNOpen TRDP prototype stack

Author

B. Loehr (initial version)

Remarks

This Source Code Form is subject to the terms of the Mozilla Public License, v. 2.0. If a copy of the MPL was not distributed with this file, You can obtain one at http://mozilla.org/MPL/2.0/. Copyright Bombardier Transportation Inc. or its subsidiaries and others, 2013. All rights reserved.

5.6.2 Function Documentation

5.6.2.1 tau_addr2Uri()

Function to convert an IP address to a URI.

Receives an IP-Address and translates it into the host part of the corresponding URI. Both unicast and multicast addresses are accepted.

Parameters

in	appHandle	Handle returned by tlc_openSession()
out	pUri	Pointer to a string to return the URI host part
in	addr	IP address, 0==own address

Return values

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	Parameter error

5.6.2.2 tau_delnitDnr()

Function to deinit DNR.

Release any resources allocated by DNR.

Parameters

	in	appHandle	Handle returned by tlc_openSession()	
--	----	-----------	--------------------------------------	--

Return values

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	Parameter error

5.6.2.3 tau_DNRstatus()

Function to get the status of DNR.

Parameters

	in	appHandle	Handle returned by tlc_openSession())
--	----	-----------	--------------------------------------	---

Return values

TRDP_DNR_NOT_AVAILABLE	no error
TRDP_DNR_UNKNOWN	enabled, but cache is empty
TRDP_DNR_ACTIVE	enabled, cache has values
TRDP_DNR_HOSTSFILE	enabled, hostsfile used (static mode)

5.6.2.4 tau_getOwnAddr()

Function to get the own IP address.

Returns the IP address set by openSession. If it was 0 (INADDR_ANY), the address of the default adapter will be returned.

Parameters

in	annHandle	Handle returned by tlc_openSession()
T11	αρρι ιαπαιε	rialidie returned by tio_openoession()

Return values

```
own IP address
```

5.6.2.5 tau_initDnr()

```
EXT_DECL TRDP_ERR_T tau_initDnr (

TRDP_APP_SESSION_T appHandle,

TRDP_IP_ADDR_T dnsIpAddr,

UINT16 dnsPort,

const CHAR8 * pHostsFileName,

TRDP_DNR_OPTS_T dnsOptions,

BOOL8 waitForDnr )
```

Function to init the DNR subsystem Initialize the DNR resolver.

Function to init DNR.

Depending on the supplied options, three operational modes are supported:

- 1. TRDP_DNR_COMMON_THREAD (default) Expect tlc_process running in a different, separate thread
- 2. TRDP_DNR_OWN_THREAD For single threaded systems only! Internally call tlc_process()
- 3. TRDP_DNR_STANDARD_DNS Use standard DNS instead of TCN-DNS. Default dnsPort (= 0) for TCN-DNS is 17225, for standard DNS it is 53.

Parameters

in	appHandle	Handle returned by tlc_openSession().	
in	dnslpAddr	InslpAddr DNS/ECSP IP address.	
in	dnsPort	DNS port number.	
in	pHostsFileName	Optional host file name as ECSP replacement/addition.	
in	dnsOptions	Use existing thread (recommended), use own tlc_process loop or use standard DNS	
in	waitForDnr	Waits for DNR if true(recommended), doesn't wait for DNR if false(for testing).	

Return values

TRDP_NO_ERR	no error
TRDP_INIT_ERR	initialisation error

< default DNR/ECSP settings

5.6.2.6 tau_uri2Addr()

Function to convert a URI to an IP address.

Receives an URI as input variable and translates this URI to an IP-Address. The URI may specify either a unicast or a multicast IP-Address.

Parameters

in	appHandle	Handle returned by tlc_openSession()
out	pAddr	Pointer to return the IP address
in	pUri	Pointer to an URI or an IP Address string, NULL==own URI

Return values

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	Parameter error
TRDP_UNRESOLVED_ERR	Could not resolve error

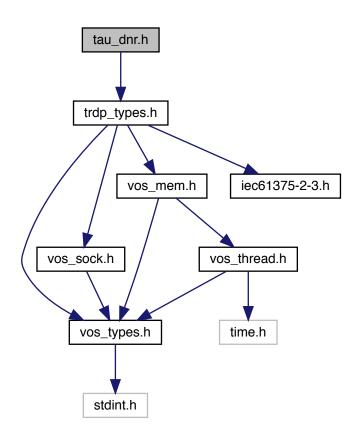
Return values

TRDP TOPO ERR	Cache/DB entry is invalid

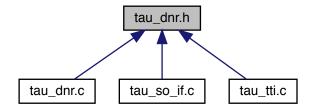
5.7 tau_dnr.h File Reference

TRDP utility interface definitions.

#include "trdp_types.h"
Include dependency graph for tau_dnr.h:



This graph shows which files directly or indirectly include this file:



Typedefs

- typedef enum TRDP_DNR_STATE_T DNR state.
- typedef enum TRDP_DNR_OPTS TRDP_DNR_OPTS_T DNR options.

Enumerations

- enum TRDP_DNR_STATE
 - DNR state.
- enum TRDP_DNR_OPTS { , TRDP_DNR_OWN_THREAD = 1 }
 DNR options.

Functions

EXT_DECL TRDP_ERR_T tau_initDnr (TRDP_APP_SESSION_T appHandle, TRDP_IP_ADDR_T dnsIp←
Addr, UINT16 dnsPort, const CHAR8 *pHostsFileName, TRDP_DNR_OPTS_T dnsOptions, BOOL8 wait←
ForDnr)

Function to init DNR.

• EXT_DECL void tau_deInitDnr (TRDP_APP_SESSION_T appHandle)

Release any resources allocated by DNR.

• EXT_DECL TRDP_DNR_STATE_T tau_DNRstatus (TRDP_APP_SESSION_T appHandle)

Function to get the status of DNR.

• EXT_DECL TRDP_IP_ADDR_T tau_getOwnAddr (TRDP_APP_SESSION_T appHandle)

Function to get the own IP address.

• EXT_DECL TRDP_ERR_T tau_uri2Addr (TRDP_APP_SESSION_T appHandle, TRDP_IP_ADDR_T *p↔ Addr, const TRDP_URI_T pUri)

Function to convert a URI to an IP address.

• EXT_DECL TRDP_ERR_T tau_addr2Uri (TRDP_APP_SESSION_T appHandle, TRDP_URI_HOST_T pUri, TRDP_IP_ADDR_T addr)

Function to convert an IP address to a URI.

5.7.1 Detailed Description

TRDP utility interface definitions.

This module provides the interface to the following utilities

• IP - URI address translation

Note

Project: TCNOpen TRDP prototype stack

Author

Armin-H. Weiss (initial version)

Remarks

This Source Code Form is subject to the terms of the Mozilla Public License, v. 2.0. If a copy of the MPL was not distributed with this file, You can obtain one at http://mozilla.org/MPL/2.0/. Copyright Bombardier Transportation Inc. or its subsidiaries and others, 2013. All rights reserved.

5.7.2 Enumeration Type Documentation

5.7.2.1 TRDP_DNR_OPTS

```
enum TRDP_DNR_OPTS
```

DNR options.

Enumerator

TRDP_DNR_OWN_THREAD	For single threaded systems only! Internally call tlc_process()
---------------------	---

5.7.3 Function Documentation

5.7.3.1 tau_addr2Uri()

```
EXT_DECL TRDP_ERR_T tau_addr2Uri (

TRDP_APP_SESSION_T appHandle,
```

```
TRDP_URI_HOST_T pUri,
TRDP_IP_ADDR_T addr )
```

Function to convert an IP address to a URI.

Receives an IP-Address and translates it into the host part of the corresponding URI. Both unicast and multicast addresses are accepted.

Parameters

in	appHandle	Handle returned by tlc_openSession().
out	pUri	Pointer to a string to return the URI host part
in	addr	IP address, 0==own address

Return values

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	Parameter error

Receives an IP-Address and translates it into the host part of the corresponding URI. Both unicast and multicast addresses are accepted.

Parameters

in	appHandle	Handle returned by tlc_openSession()
out	pUri	Pointer to a string to return the URI host part
in	addr	IP address, 0==own address

Return values

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	Parameter error

5.7.3.2 tau_delnitDnr()

Release any resources allocated by DNR.

Parameters

in	appHandle	Handle returned by tlc_openSession().
----	-----------	---------------------------------------

Return values

none	Release any resources allocated by DNR.

Parameters

	in	appHandle	Handle returned by tlc_openSession()]
--	----	-----------	--------------------------------------	---

Return values

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	Parameter error

5.7.3.3 tau_DNRstatus()

```
EXT_DECL TRDP_DNR_STATE_T tau_DNRstatus (

TRDP_APP_SESSION_T appHandle )
```

Function to get the status of DNR.

Parameters

in	appHandle	Handle returned by tlc_openSession()
----	-----------	--------------------------------------

Return values

TRDP_DNR_NOT_AVAILABLE	no error
TRDP_DNR_UNKNOWN	enabled, but cache is empty
TRDP_DNR_ACTIVE	enabled, cache has values
TRDP_DNR_HOSTSFILE	enabled, hostsfile used (static mode)

5.7.3.4 tau_getOwnAddr()

Function to get the own IP address.

Parameters

in	appHandle	Handle returned by tlc_openSession().
----	-----------	---------------------------------------

Return values

own	IP address
-----	------------

Returns the IP address set by openSession. If it was 0 (INADDR_ANY), the address of the default adapter will be returned.

Parameters

in	appHandle	Handle returned by tlc_openSession()
----	-----------	--------------------------------------

Return values

own	IP address
-----	------------

5.7.3.5 tau_initDnr()

```
EXT_DECL TRDP_ERR_T tau_initDnr (

TRDP_APP_SESSION_T appHandle,

TRDP_IP_ADDR_T dnsIpAddr,

UINT16 dnsPort,

const CHAR8 * pHostsFileName,

TRDP_DNR_OPTS_T dnsOptions,

BOOL8 waitForDnr )
```

Function to init DNR.

Parameters

in	appHandle	Handle returned by tlc_openSession().
in	dnslpAddr	DNS/ECSP IP address.
in	dnsPort	DNS port number.
in	pHostsFileName	Optional host file name as ECSP replacement/addition.
in	dnsOptions	Use existing thread (recommended), use own tlc_process loop or use standard DNS
in	waitForDnr	Waits for DNR if true(recommended), doesn't wait for DNR if false(for testing).

Return values

TRDP_NO_ERR	no error
TRDP_INIT_ERR	initialisation error

Function to init DNR.

Depending on the supplied options, three operational modes are supported:

- 1. TRDP_DNR_COMMON_THREAD (default) Expect tlc_process running in a different, separate thread
- 2. TRDP_DNR_OWN_THREAD For single threaded systems only! Internally call tlc_process()
- 3. TRDP_DNR_STANDARD_DNS Use standard DNS instead of TCN-DNS. Default dnsPort (= 0) for TCN-DNS is 17225, for standard DNS it is 53.

Parameters

in	appHandle	Handle returned by tlc_openSession().
in	dnslpAddr	DNS/ECSP IP address.

Parameters

in	dnsPort	DNS port number.
in	pHostsFileName	Optional host file name as ECSP replacement/addition.
in	dnsOptions	Use existing thread (recommended), use own tlc_process loop or use standard DNS
in	waitForDnr	Waits for DNR if true(recommended), doesn't wait for DNR if false(for testing).

Return values

TRDP_NO_ERR	no error
TRDP_INIT_ERR	initialisation error

< default DNR/ECSP settings

5.7.3.6 tau_uri2Addr()

Function to convert a URI to an IP address.

Receives a URI as input variable and translates this URI to an IP-Address. The URI may specify either a unicast or a multicast IP-Address. The caller may specify a topographic counter, which will be checked.

Parameters

in	appHandle	Handle returned by tlc_openSession().
out	pAddr	Pointer to return the IP address
in	pUri	Pointer to a URI or an IP Address string, NULL==own URI

Return values

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	Parameter error

Receives an URI as input variable and translates this URI to an IP-Address. The URI may specify either a unicast or a multicast IP-Address.

Parameters

in	appHandle	Handle returned by tlc_openSession()
out	pAddr	Pointer to return the IP address
in	pUri	Pointer to an URI or an IP Address string, NULL==own URI

Return values

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	Parameter error
TRDP_UNRESOLVED_ERR	Could not resolve error

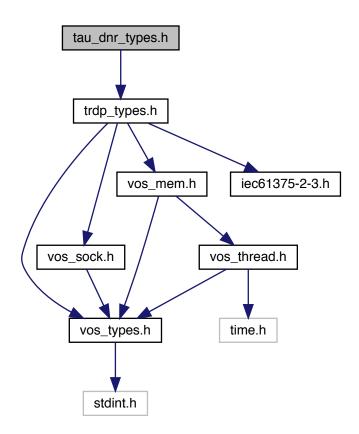
Return values

TRDP TOPO ERR	Cache/DB entry is invalid

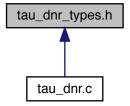
5.8 tau_dnr_types.h File Reference

TRDP utility interface definitions.

#include "trdp_types.h"
Include dependency graph for tau_dnr_types.h:



This graph shows which files directly or indirectly include this file:



Data Structures

• struct TCN_URI

TCN-DNS simplified header structures.

struct TRDP DNS REQUEST

TCN-DNS Request telegram TCN_DNS_REQ_DS.

struct TRDP_DNS_REPLY

TCN-DNS Reply telegram TCN_DNS_REP_DS.

Typedefs

typedef struct TCN_URI TCN_URI_T

TCN-DNS simplified header structures.

typedef struct TRDP_DNS_REQUEST_T

TCN-DNS Request telegram TCN_DNS_REQ_DS.

typedef struct TRDP_DNS_REPLY_T

TCN-DNS Reply telegram TCN_DNS_REP_DS.

5.8.1 Detailed Description

TRDP utility interface definitions.

This module provides typedefs to the following utilities

• IP - URI address translation

Note

Project: TCNOpen TRDP prototype stack

Author

Bernd Loehr (initial version)

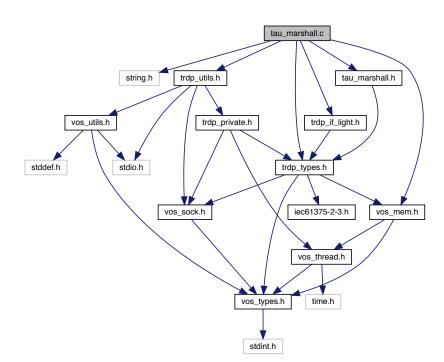
Remarks

This Source Code Form is subject to the terms of the Mozilla Public License, v. 2.0. If a copy of the MPL was not distributed with this file, You can obtain one at http://mozilla.org/MPL/2.0/. Copyright NewTec GmbH, 2017. All rights reserved.

5.9 tau_marshall.c File Reference

Marshalling functions for TRDP.

```
#include <string.h>
#include "trdp_types.h"
#include "trdp_if_light.h"
#include "trdp_utils.h"
#include "vos_mem.h"
#include "tau_marshall.h"
Include dependency graph for tau_marshall.c:
```



Data Structures

struct TAU_MARSHALL_INFO_T

Marshalling info, used to and from wire.

Functions

- EXT_DECL TRDP_ERR_T tau_initMarshall (void **ppRefCon, UINT32 numComId, TRDP_COMID_DSID
 _MAP_T *pComIdDsIdMap, UINT32 numDataSet, TRDP_DATASET_T *pDataset[])
 Function to initialise the marshalling/unmarshalling.
- EXT_DECL TRDP_ERR_T tau_marshall (void *pRefCon, UINT32 comId, UINT8 *pSrc, UINT32 srcSize, UINT8 *pDest, UINT32 *pDestSize, TRDP_DATASET_T **ppDSPointer)
 marshall function.
- EXT_DECL TRDP_ERR_T tau_unmarshall (void *pRefCon, UINT32 comId, UINT8 *pSrc, UINT32 srcSize, UINT8 *pDest, UINT32 *pDestSize, TRDP_DATASET_T **ppDSPointer)

unmarshall function.

 EXT_DECL TRDP_ERR_T tau_marshallDs (void *pRefCon, UINT32 dsld, UINT8 *pSrc, UINT32 srcSize, UINT8 *pDest, UINT32 *pDestSize, TRDP_DATASET_T **ppDSPointer)

marshall data set function.

 EXT_DECL TRDP_ERR_T tau_unmarshallDs (void *pRefCon, UINT32 dsld, UINT8 *pSrc, UINT32 srcSize, UINT8 *pDest, UINT32 *pDestSize, TRDP_DATASET_T **ppDSPointer)

unmarshall data set function.

EXT_DECL TRDP_ERR_T tau_calcDatasetSize (void *pRefCon, UINT32 dsld, UINT8 *pSrc, UINT32 src
 Size, UINT32 *pDestSize, TRDP_DATASET_T **ppDSPointer)

Calculate data set size by given data set id.

EXT_DECL TRDP_ERR_T tau_calcDatasetSizeByComId (void *pRefCon, UINT32 comId, UINT8 *pSrc, U
 INT32 srcSize, UINT32 *pDestSize, TRDP_DATASET_T **ppDSPointer)

Calculate data set size by given Comld.

5.9.1 Detailed Description

Marshalling functions for TRDP.

Note

Project: TCNOpen TRDP prototype stack

Author

Bernd Loehr, NewTec GmbH

Remarks

This Source Code Form is subject to the terms of the Mozilla Public License, v. 2.0. If a copy of the MPL was not distributed with this file, You can obtain one at http://mozilla.org/MPL/2.0/. Copyright Bombardier Transportation Inc. or its subsidiaries and others, 2013. All rights reserved.

5.9.2 Function Documentation

5.9.2.1 tau_calcDatasetSize()

Calculate data set size by given data set id.

Parameters

in	pRefCon	Pointer to user context
in	dsld	Dataset id to identify the structure out of a configuration
in	pSrc	Pointer to received original message
in	srcSize	size of the source buffer
out	pDestSize	Pointer to the size of the data set
in,out	ppDSPointer	pointer to pointer to cached dataset, set NULL if not used, set content NULL if
		unknown

Return values

TRDP_INIT_ERR	marshalling not initialised
TRDP_NO_ERR	no error
TRDP_MEM_ERR	provided buffer to small
TRDP_PARAM_ERR	Parameter error
TRDP_STATE_ERR	Too deep recursion
TRDP_COMID_ERR	comid not existing
TRDP_MARSHALLING_ERR	dataset/source size mismatch

5.9.2.2 tau_calcDatasetSizeByComId()

Calculate data set size by given Comld.

Parameters

in	pRefCon	Pointer to user context
in	comld	Comld id to identify the structure out of a configuration
in	pSrc	Pointer to received original message
in	srcSize	size of the source buffer
out	pDestSize	Pointer to the size of the data set
in,out	ppDSPointer	pointer to pointer to cached dataset, set NULL if not used, set content NULL if
		unknown

Return values

TRDP_INIT_ERR	marshalling not initialised
TRDP_NO_ERR	no error
TRDP_MEM_ERR	provided buffer to small
TRDP_PARAM_ERR	Parameter error
TRDP_STATE_ERR	Too deep recursion

Return values

TRDP_COMID_ERR	comid not existing
TRDP_MARSHALLING_ERR	dataset/source size mismatch

5.9.2.3 tau_initMarshall()

Function to initialise the marshalling/unmarshalling.

Types for marshalling / unmarshalling.

The supplied array must be sorted by Comlds. The array must exist during the use of the marshalling functions (until tlc_terminate()).

Parameters

in,out	ppRefCon	Returns a pointer to be used for the reference context of marshalling/unmarshalling
in	numComId	Number of datasets found in the configuration
in	pComldDsldMap	Pointer to an array of structures of type TRDP_DATASET_T
in	numDataSet	Number of datasets found in the configuration
in	pDataset	Pointer to an array of pointers to structures of type TRDP_DATASET_T

Return values

TRDP_NO_ERR	no error
TRDP_MEM_ERR	provided buffer to small
TRDP_PARAM_ERR	Parameter error

5.9.2.4 tau_marshall()

marshall function.

Parameters

in	pRefCon	pointer to user context
in	comId	Comld to identify the structure out of a configuration
in	pSrc	pointer to received original message
in	srcSize	size of the source buffer
in	pDest	pointer to a buffer for the treated message
in,out	pDestSize	size of the provide buffer / size of the treated message
in,out	ppDSPointer	pointer to pointer to cached dataset set NULL if not used, set content NULL if unknown

Return values

TRDP_NO_ERR	no error
TRDP_MEM_ERR	provided buffer to small
TRDP_PARAM_ERR	Parameter error
TRDP_STATE_ERR	Too deep recursion
TRDP_MARSHALLING_ERR	dataset/source size mismatch

5.9.2.5 tau_marshallDs()

marshall data set function.

Parameters

in	pRefCon	pointer to user context
in	dsld	Data set id to identify the structure out of a configuration
in	pSrc	pointer to received original message
in	srcSize	size of the source buffer
in	pDest	pointer to a buffer for the treated message
in, out	pDestSize	size of the provide buffer / size of the treated message
in, out	ppDSPointer	pointer to pointer to cached dataset set NULL if not used, set content NULL if unknown

Return values

TRDP_INIT_ERR	marshalling not initialised
TRDP_NO_ERR	no error
TRDP_MEM_ERR	provided buffer to small
TRDP_PARAM_ERR	Parameter error
TRDP_STATE_ERR	Too deep recursion
TRDP_COMID_ERR	comid not existing

Return values

TRDP_MARSHALLING_ERR	dataset/source size mismatch
----------------------	------------------------------

5.9.2.6 tau_unmarshall()

unmarshall function.

Parameters

in	pRefCon	pointer to user context
in	comId	Comld to identify the structure out of a configuration
in	pSrc	pointer to received original message
in	srcSize	size of the source buffer
in	pDest	pointer to a buffer for the treated message
in,out	pDestSize	size of the provide buffer / size of the treated message
in,out	ppDSPointer	pointer to pointer to cached dataset set NULL if not used, set content NULL if unknown

Return values

TRDP_INIT_ERR	marshalling not initialised
TRDP_NO_ERR	no error
TRDP_MEM_ERR	provided buffer to small
TRDP_PARAM_ERR	Parameter error
TRDP_STATE_ERR	Too deep recursion
TRDP_COMID_ERR	comid not existing
TRDP_MARSHALLING_ERR	dataset/source size mismatch

5.9.2.7 tau_unmarshallDs()

```
EXT_DECL TRDP_ERR_T tau_unmarshallDs (
    void * pRefCon,
    UINT32 dsId,
    UINT8 * pSrc,
    UINT32 srcSize,
    UINT8 * pDest,
```

```
UINT32 * pDestSize,
TRDP_DATASET_T ** ppDSPointer )
```

unmarshall data set function.

Parameters

in	pRefCon	pointer to user context
in	dsld	Data set id to identify the structure out of a configuration
in	pSrc	pointer to received original message
in	srcSize	size of the source buffer
in	pDest	pointer to a buffer for the treated message
in,out	pDestSize	size of the provide buffer / size of the treated message
in,out	ppDSPointer	pointer to pointer to cached dataset set NULL if not used, set content NULL if unknown

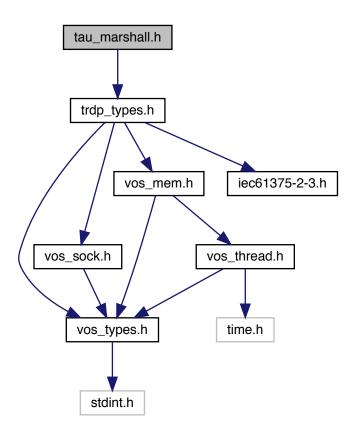
Return values

TRDP_INIT_ERR	marshalling not initialised
TRDP_NO_ERR	no error
TRDP_MEM_ERR	provided buffer to small
TRDP_PARAM_ERR	Parameter error
TRDP_STATE_ERR	Too deep recursion
TRDP_COMID_ERR	comid not existing
TRDP_MARSHALLING_ERR	dataset/source size mismatch

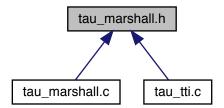
5.10 tau_marshall.h File Reference

TRDP utility interface definitions.

#include "trdp_types.h"
Include dependency graph for tau_marshall.h:



This graph shows which files directly or indirectly include this file:



Functions

marshall function.

Types for marshalling / unmarshalling.

- EXT_DECL TRDP_ERR_T tau_marshall (void *pRefCon, UINT32 comId, UINT8 *pSrc, UINT32 srcSize, UINT8 *pDest, UINT32 *pDestSize, TRDP_DATASET_T **ppDSPointer)
- EXT_DECL TRDP_ERR_T tau_marshallDs (void *pRefCon, UINT32 dsld, UINT8 *pSrc, UINT32 srcSize, UINT8 *pDest, UINT32 *pDestSize, TRDP_DATASET_T **ppDSPointer)

marshall data set function.

- EXT_DECL TRDP_ERR_T tau_unmarshall (void *pRefCon, UINT32 comId, UINT8 *pSrc, UINT32 srcSize, UINT8 *pDest, UINT32 *pDestSize, TRDP_DATASET_T **ppDSPointer)
 unmarshall function.
- EXT_DECL TRDP_ERR_T tau_unmarshallDs (void *pRefCon, UINT32 dsld, UINT8 *pSrc, UINT32 srcSize, UINT8 *pDest, UINT32 *pDestSize, TRDP_DATASET_T **ppDSPointer)
 unmarshall data set function.
- EXT_DECL TRDP_ERR_T tau_calcDatasetSize (void *pRefCon, UINT32 dsld, UINT8 *pSrc, UINT32 src
 Size, UINT32 *pDestSize, TRDP_DATASET_T **ppDSPointer)

Calculate data set size by given data set id.

EXT_DECL TRDP_ERR_T tau_calcDatasetSizeByComld (void *pRefCon, UINT32 comld, UINT8 *pSrc, U
 INT32 srcSize, UINT32 *pDestSize, TRDP_DATASET_T **ppDSPointer)

Calculate data set size by given Comld.

5.10.1 Detailed Description

TRDP utility interface definitions.

This module provides the interface to the following utilities

· marshalling/unmarshalling

Note

Project: TCNOpen TRDP prototype stack

Author

Armin-H. Weiss

Remarks

This Source Code Form is subject to the terms of the Mozilla Public License, v. 2.0. If a copy of the MPL was not distributed with this file, You can obtain one at http://mozilla.org/MPL/2.0/. Copyright Bombardier Transportation Inc. or its subsidiaries and others, 2013. All rights reserved.

5.10.2 Function Documentation

5.10.2.1 tau_calcDatasetSize()

Calculate data set size by given data set id.

Parameters

in	pRefCon	Pointer to user context
in	dsld	Dataset id to identify the structure out of a configuration
in	pSrc	Pointer to received original message
in	srcSize	size of the source buffer
out	pDestSize	Pointer to the size of the data set
in,out	ppDSPointer	pointer to pointer to cached dataset, set NULL if not used, set content NULL if
		unknown

Return values

TRDP_NO_ERR	no error
TRDP_INIT_ERR	marshalling not initialised
TRDP_PARAM_ERR	data set id not existing

Parameters

in	pRefCon	Pointer to user context
in	dsld	Dataset id to identify the structure out of a configuration
in	pSrc	Pointer to received original message
in	srcSize	size of the source buffer
out	pDestSize	Pointer to the size of the data set
in,out	ppDSPointer	pointer to pointer to cached dataset, set NULL if not used, set content NULL if
		unknown

Return values

TRDP_INIT_ERR	marshalling not initialised
TRDP_NO_ERR	no error
TRDP_MEM_ERR	provided buffer to small
TRDP_PARAM_ERR	Parameter error
TRDP_STATE_ERR	Too deep recursion
TRDP_COMID_ERR	comid not existing
TRDP_MARSHALLING_ERR	dataset/source size mismatch

5.10.2.2 tau_calcDatasetSizeByComId()

Calculate data set size by given Comld.

Parameters

in	pRefCon	Pointer to user context
in	comld	Comld id to identify the structure out of a configuration
in	pSrc	Pointer to received original message
in	srcSize	size of the source buffer
out	pDestSize	Pointer to the size of the data set
in,out	ppDSPointer	pointer to pointer to cached dataset, set NULL if not used, set content NULL if
		unknown

Return values

TRDP_NO_ERR	no error
TRDP_INIT_ERR	marshalling not initialised
TRDP_PARAM_ERR	data set id not existing

Parameters

in	pRefCon	Pointer to user context
in	comId	Comld id to identify the structure out of a configuration
in	pSrc	Pointer to received original message
in	srcSize	size of the source buffer
out	pDestSize	Pointer to the size of the data set
in,out	ppDSPointer	pointer to pointer to cached dataset, set NULL if not used, set content NULL if
		unknown

Return values

TRDP_INIT_ERR	marshalling not initialised
TRDP_NO_ERR	no error
TRDP_MEM_ERR	provided buffer to small
TRDP_PARAM_ERR	Parameter error
TRDP_STATE_ERR	Too deep recursion
TRDP_COMID_ERR	comid not existing
TRDP_MARSHALLING_ERR	dataset/source size mismatch

5.10.2.3 tau_initMarshall()

Types for marshalling / unmarshalling.

Function to initialise the marshalling/unmarshalling.

Parameters

in, out	ppRefCon	Returns a pointer to be used for the reference context of marshalling/unmarshalling
in	numComId	Number of datasets found in the configuration
in	pComldDsldMap	Pointer to an array of structures of type TRDP_DATASET_T
in	numDataSet	Number of datasets found in the configuration
in	pDataset	Pointer to an array of pointers to structures of type TRDP_DATASET_T

Return values

TRDP_NO_ERR	no error
TRDP_MEM_ERR	provided buffer to small
TRDP_PARAM_ERR	Parameter error

Types for marshalling / unmarshalling.

The supplied array must be sorted by Comlds. The array must exist during the use of the marshalling functions (until tlc_terminate()).

Parameters

in,out	ppRefCon	Returns a pointer to be used for the reference context of marshalling/unmarshalling
in	numComId	Number of datasets found in the configuration
in	pComldDsldMap	Pointer to an array of structures of type TRDP_DATASET_T
in	numDataSet	Number of datasets found in the configuration
in	pDataset	Pointer to an array of pointers to structures of type TRDP_DATASET_T

Return values

TRDP_NO_ERR	no error
TRDP_MEM_ERR	provided buffer to small
TRDP_PARAM_ERR	Parameter error

5.10.2.4 tau_marshall()

marshall function.

Parameters

in	pRefCon	pointer to user context
in	comId	Comld to identify the structure out of a configuration
in	pSrc	pointer to received original message
in	srcSize	size of the source buffer
in	pDest	pointer to a buffer for the treated message
in,out	pDestSize	size of the provide buffer / size of the treated message
in,out	ppDSPointer	pointer to pointer to cached dataset set NULL if not used, set content NULL if unknown

Return values

TRDP_NO_ERR	no error
TRDP_MEM_ERR	provided buffer to small
TRDP_INIT_ERR	marshalling not initialised
TRDP_COMID_ERR	comid not existing
TRDP_PARAM_ERR	Parameter error

Parameters

in	pRefCon	pointer to user context
in	comld	Comld to identify the structure out of a configuration
in	pSrc	pointer to received original message
in	srcSize	size of the source buffer
in	pDest	pointer to a buffer for the treated message
in,out	pDestSize	size of the provide buffer / size of the treated message
in,out	ppDSPointer	pointer to pointer to cached dataset set NULL if not used, set content NULL if unknown

Return values

TRDP_NO_ERR	no error
TRDP_MEM_ERR	provided buffer to small
TRDP_PARAM_ERR	Parameter error
TRDP_STATE_ERR	Too deep recursion
TRDP_MARSHALLING_ERR	dataset/source size mismatch

5.10.2.5 tau_marshallDs()

marshall data set function.

Parameters

in	pRefCon	pointer to user context
in	dsld	Data set id to identify the structure out of a configuration
in	pSrc	pointer to received original message
in	srcSize	size of the source buffer
in	pDest	pointer to a buffer for the treated message
in,out	pDestSize	size of the provide buffer / size of the treated message
in,out	ppDSPointer	pointer to pointer to cached dataset set NULL if not used, set content NULL if unknown

Return values

TRDP_NO_ERR	no error
TRDP_MEM_ERR	provided buffer to small
TRDP_INIT_ERR	marshalling not initialised
TRDP_COMID_ERR	comid not existing
TRDP_PARAM_ERR	Parameter error

Parameters

in	pRefCon	pointer to user context
in	dsld	Data set id to identify the structure out of a configuration
in	pSrc	pointer to received original message
in	srcSize	size of the source buffer
in	pDest	pointer to a buffer for the treated message
in,out	pDestSize	size of the provide buffer / size of the treated message
in,out	ppDSPointer	pointer to pointer to cached dataset set NULL if not used, set content NULL if unknown

Return values

TRDP_INIT_ERR	marshalling not initialised
TRDP_NO_ERR	no error
TRDP_MEM_ERR	provided buffer to small
TRDP_PARAM_ERR	Parameter error
TRDP_STATE_ERR	Too deep recursion
TRDP_COMID_ERR	comid not existing
TRDP_MARSHALLING_ERR	dataset/source size mismatch

5.10.2.6 tau_unmarshall()

```
UINT32 * pDestSize,
TRDP_DATASET_T ** ppDSPointer )
```

unmarshall function.

Parameters

in	pRefCon	pointer to user context
in	comId	Comld to identify the structure out of a configuration
in	pSrc	pointer to received original message
in	srcSize	size of the source buffer
in	pDest	pointer to a buffer for the treated message
in,out	pDestSize	size of the provide buffer / size of the treated message
in,out	ppDSPointer	pointer to pointer to cached dataset set NULL if not used, set content NULL if unknown

Return values

TRDP_NO_ERR	no error
TRDP_MEM_ERR	provided buffer to small
TRDP_INIT_ERR	marshalling not initialised
TRDP_COMID_ERR	comid not existing

Parameters

in	pRefCon	pointer to user context
in	comId	Comld to identify the structure out of a configuration
in	pSrc	pointer to received original message
in	srcSize	size of the source buffer
in	pDest	pointer to a buffer for the treated message
in,out	pDestSize	size of the provide buffer / size of the treated message
in,out	ppDSPointer	pointer to pointer to cached dataset set NULL if not used, set content NULL if unknown

Return values

TRDP_INIT_ERR	marshalling not initialised
TRDP_NO_ERR	no error
TRDP_MEM_ERR	provided buffer to small
TRDP_PARAM_ERR	Parameter error
TRDP_STATE_ERR	Too deep recursion
TRDP_COMID_ERR	comid not existing
TRDP_MARSHALLING_ERR	dataset/source size mismatch

5.10.2.7 tau_unmarshallDs()

unmarshall data set function.

Parameters

in	pRefCon	pointer to user context
in	dsld	Data set id to identify the structure out of a configuration
in	pSrc	pointer to received original message
in	srcSize	size of the source buffer
in	pDest	pointer to a buffer for the treated message
in,out	pDestSize	size of the provide buffer / size of the treated message
in,out	ppDSPointer	pointer to pointer to cached dataset set NULL if not used, set content NULL if unknown

Return values

TRDP_NO_ERR	no error
TRDP_MEM_ERR	provided buffer to small
TRDP_INIT_ERR	marshalling not initialised
TRDP_COMID_ERR	comid not existing

Parameters

in	pRefCon	pointer to user context
in	dsld	Data set id to identify the structure out of a configuration
in	pSrc	pointer to received original message
in	srcSize	size of the source buffer
in	pDest	pointer to a buffer for the treated message
in,out	pDestSize	size of the provide buffer / size of the treated message
in,out	ppDSPointer	pointer to pointer to cached dataset set NULL if not used, set content NULL if unknown

Return values

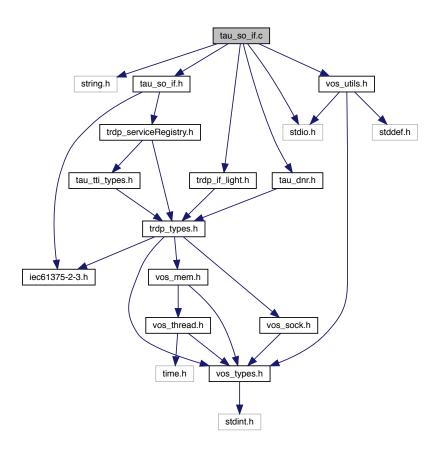
TRDP_INIT_ERR	marshalling not initialised
TRDP_NO_ERR	no error
TRDP_MEM_ERR	provided buffer to small
TRDP_PARAM_ERR	Parameter error
TRDP_STATE_ERR	Too deep recursion
TRDP_COMID_ERR	comid not existing
TRDP_MARSHALLING_ERR	dataset/source size mismatch

5.11 tau_so_if.c File Reference

Access to service oriented functions of the SRM.

```
#include <string.h>
#include <stdio.h>
#include "trdp_if_light.h"
#include "tau_dnr.h"
#include "tau_so_if.h"
```

#include "vos_utils.h"
Include dependency graph for tau_so_if.c:



Functions

EXT_DECL TRDP_ERR_T tau_addService (TRDP_APP_SESSION_T appHandle, SRM_SERVICE_INF

 O_T *pServiceToAdd, BOOL8 waitForCompletion)

Function to add to the service registry of the consist-local SRM.

EXT_DECL TRDP_ERR_T tau_delService (TRDP_APP_SESSION_T appHandle, SRM_SERVICE_INFO
 — T *pServiceToRemove, BOOL8 waitForCompletion)

Remove the defined service from the service registry of the consist-local SRM.

EXT_DECL TRDP_ERR_T tau_updService (TRDP_APP_SESSION_T appHandle, SRM_SERVICE_INF
 — O_T *pServiceToUpdate, BOOL8 waitForCompletion)

Register an update a service.

• EXT_DECL TRDP_ERR_T tau_getServicesList (TRDP_APP_SESSION_T appHandle, SRM_SERVICE_← ENTRIES_T **ppServicesListBuffer, UINT32 *pNoOfServices, SRM_SERVICE_ENTRIES_T *pFilterEntry)

Get a list of the services known by the service registry of the local TTDB / SRM.

• EXT_DECL void tau_freeServicesList (SRM_SERVICE_ENTRIES_T *pServicesListBuffer)

Release the memory of a list received by tau_getServiceList.

5.11.1 Detailed Description

Access to service oriented functions of the SRM.

Because of the asynchronous behavior of the TTI subsystem, the source functions (add/upd/del) will return TRD← P_NODATA_ERR if called with the the no-wait option.

Note

Project: TCNOpen TRDP prototype stack

Author

B. Loehr (initial version)

Remarks

This Source Code Form is subject to the terms of the Mozilla Public License, v. 2.0. If a copy of the MPL was not distributed with this file, You can obtain one at http://mozilla.org/MPL/2.0/. Copyright NewTec GmbH 2019. All rights reserved.

5.11.2 Function Documentation

5.11.2.1 tau_addService()

Function to add to the service registry of the consist-local SRM.

Note: If waitForCompletion == TRUE, this function will block until completion (or timeout).

Parameters

in	appHandle	Handle returned by tlc_openSession().
in,out	pServiceToAdd	Pointer to a service registry structure to be set and/or updated (returned)
in	waitForCompletion	if true, block for reply

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	Parameter error
TRDP_NODATA_ERR	Data currently not available, try again later
TRDP_TIMEOUT_ERR	Reply timed out
TRDP_SEMA_ERR	Semaphore could not be aquired

5.11.2.2 tau_delService()

Remove the defined service from the service registry of the consist-local SRM.

Note: waitForCompletion is currently ignored, this function does not block.

Parameters

in	appHandle	Handle returned by tlc_openSession().
in,out	pServiceToRemove	Pointer to a service registry structure to be set and/or updated (returned)
in	waitForCompletion	if true, block for reply

Return values

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	Parameter error
TRDP_NODATA_ERR	Data currently not available, try again later
TRDP_TIMEOUT_ERR	Reply timed out
TRDP_SEMA_ERR	Semaphore could not be aquired

5.11.2.3 tau_freeServicesList()

```
\begin{tabular}{ll} EXT\_DECL \ void \ tau\_freeServicesList \ ( \\ SRM\_SERVICE\_ENTRIES\_T \ * \ pServicesListBuffer \ ) \end{tabular}
```

Release the memory of a list received by tau_getServiceList.

Parameters

in	pServicesListBuffer	Pointer to list aquired by getServiceList.
----	---------------------	--

Return values

none

5.11.2.4 tau_getServicesList()

```
SRM_SERVICE_ENTRIES_T ** ppServicesListBuffer,
UINT32 * pNoOfServices,
SRM_SERVICE_ENTRIES_T * pFilterEntry )
```

Get a list of the services known by the service registry of the local TTDB / SRM.

Note: This function will block until completion (or timeout). The buffer must be provided by the caller.

Parameters 4 8 1

in	appHandle	Handle returned by tlc_openSession().
out	ppServicesListBuffer	Pointer to pointer containing the list. Has to be vos_memfree'd by user
out	pNoOfServices	Pointer to no. of services in returned list
in	pFilterEntry	Pointer to entry for filtering

Return values

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	Parameter error
TRDP_NODATA_ERR	Data currently not available, try again later
TRDP_TIMEOUT_ERR	Reply timed out

5.11.2.5 tau_updService()

Register an update a service.

Same as addService. Note: If waitForCompletion == TRUE, this function will block until completion (or timeout).

Parameters

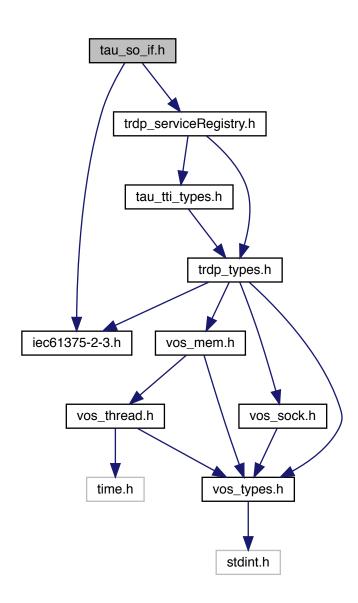
in	appHandle	Handle returned by tlc_openSession().
in, out	pServiceToUpdate	Pointer to a service registry structure to be updated
in	waitForCompletion	if true, block for reply

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	Parameter error
TRDP_NODATA_ERR	Data currently not available, try again later
TRDP_TIMEOUT_ERR	Reply timed out
TRDP_SEMA_ERR	Semaphore could not be aquired

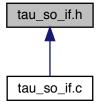
5.12 tau_so_if.h File Reference

Access to the Service Registry.

```
#include "iec61375-2-3.h"
#include "trdp_serviceRegistry.h"
Include dependency graph for tau_so_if.h:
```



This graph shows which files directly or indirectly include this file:



Functions

EXT_DECL TRDP_ERR_T tau_addService (TRDP_APP_SESSION_T appHandle, SRM_SERVICE_INF
 — O_T *pServiceToAdd, BOOL8 waitForCompletion)

Function to add to the service registry of the consist-local SRM.

EXT_DECL TRDP_ERR_T tau_delService (TRDP_APP_SESSION_T appHandle, SRM_SERVICE_INFO
 — T *pServiceToAdd, BOOL8 waitForCompletion)

Remove the defined service from the service registry of the consist-local SRM.

Register an update a service.

- EXT_DECL TRDP_ERR_T tau_getServicesList (TRDP_APP_SESSION_T appHandle, SRM_SERVICE_← ENTRIES_T **ppServicesToAdd, UINT32 *pNoOfServices, SRM_SERVICE_ENTRIES_T *pFilterEntry)

 Get a list of the services known by the service registry of the local TTDB / SRM.
- EXT_DECL void tau_freeServicesList (SRM_SERVICE_ENTRIES_T *pServicesListBuffer)

 Release the memory of a list received by tau_getServiceList.

5.12.1 Detailed Description

Access to the Service Registry.

This header file defines the proposed extensions and additions to access the service interface (proposed as extension to the TTDB defined in IEC61375-2-3:2017

Note

Project: TCNOpen TRDP prototype stack & FDF/DbD

Author

Bernd Loehr, NewTec GmbH, 2019-06-17

Remarks

Copyright 2019, NewTec GmbH

ld

tau_so_if.h 2091 2019-10-15 08:48:18Z s-bender

5.12.2 Function Documentation

5.12.2.1 tau_addService()

Function to add to the service registry of the consist-local SRM.

Note: If waitForCompletion == TRUE, this function will block until completion (or timeout).

Parameters

in	appHandle	Handle returned by tlc_openSession().
in,out	pServiceToAdd	Pointer to a service registry structure to be set and/or updated (returned)
in	waitForCompletion	if true, block for reply

Return values

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	Parameter error
TRDP_NODATA_ERR	Data currently not available, try again later
TRDP_TIMEOUT_ERR	Reply timed out
TRDP_SEMA_ERR	Semaphore could not be aquired

5.12.2.2 tau_delService()

Remove the defined service from the service registry of the consist-local SRM.

Note: waitForCompletion is currently ignored, this function does not block.

Parameters

in	appHandle	Handle returned by tlc_openSession().
in,out	pServiceToRemove	Pointer to a service registry structure to be set and/or updated (returned)
in	waitForCompletion	if true, block for reply

Return values

TRDP_NO_ERR	no error	
TRDP_PARAM_ERR	Parameter error	
TRDP_NODATA_ERR	Data currently not available, try again later	
TRDP_TIMEOUT_ERR	Reply timed out	
TRDP_SEMA_ERR	Semaphore could not be aquired	

5.12.2.3 tau_freeServicesList()

Release the memory of a list received by tau_getServiceList.

Parameters

in	pServicesListBuffer	Pointer to list aquired by getServiceList.
----	---------------------	--

Return values

none

5.12.2.4 tau_getServicesList()

Get a list of the services known by the service registry of the local TTDB / SRM.

Note: This function will block until completion (or timeout). The buffer must be provided by the caller.

Parameters

in	appHandle	Handle returned by tlc_openSession().	
out	ppServicesListBuffer Pointer to pointer containing the list. Has to be vos_memfree'd by use		
out	pNoOfServices Pointer to no. of services in returned list		
in	pFilterEntry Pointer to entry for filtering		

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	Parameter error

Return values

TRDP_NODATA_ERR	Data currently not available, try again later
TRDP_TIMEOUT_ERR	Reply timed out

5.12.2.5 tau_updService()

Register an update a service.

Same as addService. Note: If waitForCompletion == TRUE, this function will block until completion (or timeout).

Parameters

in	appHandle	Handle returned by tlc_openSession().	
in,out	pServiceToUpdate	Pointer to a service registry structure to be updated	
in	waitForCompletion	if true, block for reply	

Return values

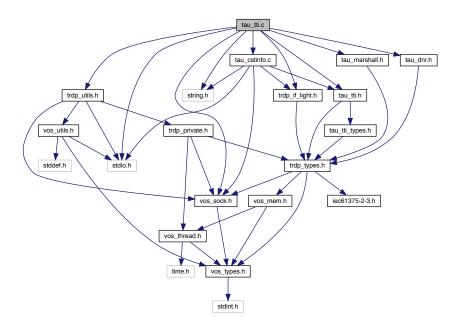
TRDP_NO_ERR	no error	
TRDP_PARAM_ERR	Parameter error	
TRDP_NODATA_ERR	Data currently not available, try again later	
TRDP_TIMEOUT_ERR	Reply timed out	
TRDP_SEMA_ERR	Semaphore could not be aquired	

5.13 tau_tti.c File Reference

Functions for train topology information access.

```
#include <string.h>
#include "trdp_if_light.h"
#include "trdp_utils.h"
#include "tau_marshall.h"
#include "tau_tti.h"
#include "vos_sock.h"
#include "tau_dnr.h"
#include "tau_cstinfo.c"
```

Include dependency graph for tau_tti.c:



Macros

#define TTI_CACHED_CONSISTS 8u
 We hold this number of consist infos (ca.

Functions

• EXT_DECL TRDP_ERR_T tau_initTTlaccess (TRDP_APP_SESSION_T appHandle, VOS_SEMA_T user ← Action, TRDP_IP_ADDR_T ecsplpAddr, CHAR8 *hostsFileName)

Function to init TTI access.

• EXT DECL void tau delnitTTI (TRDP APP SESSION T appHandle)

Release any resources allocated by TTI Must be called before closing the session.

• EXT_DECL TRDP_ERR_T tau_getOpTrDirectory (TRDP_APP_SESSION_T appHandle, TRDP_OP_TRA⊷ IN_DIR_STATE_T *pOpTrnDirState, TRDP_OP_TRAIN_DIR_T *pOpTrnDir)

Function to retrieve the operational train directory state.

• EXT_DECL TRDP_ERR_T tau_getOpTrnDirectoryStatusInfo (TRDP_APP_SESSION_T appHandle, TRD← P_OP_TRAIN_DIR_STATUS_INFO_T *pOpTrnDirStatusInfo)

Function to retrieve the operational train directory state info.

EXT_DECL TRDP_ERR_T tau_getTrDirectory (TRDP_APP_SESSION_T appHandle, TRDP_TRAIN_DIR
 — T *pTrnDir)

Function to retrieve the train directory.

• EXT_DECL TRDP_ERR_T tau_getStaticCstInfo (TRDP_APP_SESSION_T appHandle, TRDP_CONSIST → INFO T *pCstInfo, TRDP_UUID T const cstUUID)

Function to retrieve the consist info.

• EXT_DECL TRDP_ERR_T tau_getTTI (TRDP_APP_SESSION_T appHandle, TRDP_OP_TRAIN_DIR_S

TATE_T *pOpTrnDirState, TRDP_OP_TRAIN_DIR_T *pOpTrnDir, TRDP_TRAIN_DIR_T *pTrnDir, TRDP

_TRAIN_NET_DIR_T *pTrnNetDir)

Function to retrieve the operational train directory.

• EXT_DECL TRDP_ERR_T tau_getTrnCstCnt (TRDP_APP_SESSION_T appHandle, UINT16 *pTrnCstCnt) Function to retrieve the total number of consists in the train.

- EXT_DECL TRDP_ERR_T tau_getTrnVehCnt (TRDP_APP_SESSION_T appHandle, UINT16 *pTrnVehCnt) Function to retrieve the total number of vehicles in the train.
- EXT_DECL TRDP_ERR_T tau_getCstVehCnt (TRDP_APP_SESSION_T appHandle, UINT16 *pCstVehCnt, const TRDP_LABEL_T pCstLabel)

Function to retrieve the total number of vehicles in a consist.

EXT_DECL TRDP_ERR_T tau_getCstFctCnt (TRDP_APP_SESSION_T appHandle, UINT16 *pCstFctCnt, const TRDP_LABEL_T pCstLabel)

Function to retrieve the total number of functions in a consist.

EXT_DECL TRDP_ERR_T tau_getCstFctInfo (TRDP_APP_SESSION_T appHandle, TRDP_FUNCTION_
 —
 INFO_T *pFctInfo, const TRDP_LABEL_T pCstLabel, UINT16 maxFctCnt)

Function to retrieve the function information of the consist.

EXT_DECL TRDP_ERR_T tau_getVehInfo (TRDP_APP_SESSION_T appHandle, TRDP_VEHICLE_INF

 O_T *pVehInfo, const TRDP_LABEL_T pVehLabel, const TRDP_LABEL_T pCstLabel)

Function to retrieve the vehicle information of a consist's vehicle.

EXT_DECL TRDP_ERR_T tau_getCstInfo (TRDP_APP_SESSION_T appHandle, TRDP_CONSIST_INF

 O_T *pCstInfo, const TRDP_LABEL_T pCstLabel)

Function to retrieve the consist information of a train's consist.

EXT_DECL_TRDP_ERR_T tau_getVehOrient (TRDP_APP_SESSION_T appHandle, UINT8 *pVehOrient, UINT8 *pCstOrient, TRDP_LABEL_T pVehLabel, TRDP_LABEL_T pCstLabel)

Function to retrieve the orientation of the given vehicle.

Who am I?.

EXT_DECL UINT8 tau_getOwnOpCstNo (TRDP_APP_SESSION_T appHandle)

Get own operational consist number.

EXT_DECL UINT8 tau_getOwnTrnCstNo (TRDP_APP_SESSION_T appHandle)

Get own train consist number.

5.13.1 Detailed Description

Functions for train topology information access.

The TTI subsystem maintains a pointer to the TAU_TTDB struct in the TRDP session struct. That TAU_TTDB struct keeps the subscription and listener handles, the current TTDB directories and a pointer list to consist infos (in network format). On init, most TTDB data is requested from the ECSP plus the own consist info. This data is automatically updated if an inauguration is detected. Additional consist infos are requested on demand, only. Because of the asynchronous behavior of the TTI subsystem, most functions in tau_tti.c may return TRDP_N← ODATA_ERR on first invocation. They should be called again after 1...3 seconds (3s is the timeout for most MD replies).

Note

Project: TCNOpen TRDP prototype stack

Author

B. Loehr (initial version)

Remarks

This Source Code Form is subject to the terms of the Mozilla Public License, v. 2.0. If a copy of the MPL was not distributed with this file, You can obtain one at http://mozilla.org/MPL/2.0/. Copyright Bombardier Transportation Inc. or its subsidiaries and others, 2016-2019. All rights reserved.

5.13.2 Macro Definition Documentation

5.13.2.1 TTI_CACHED_CONSISTS

```
#define TTI_CACHED_CONSISTS 8u
```

We hold this number of consist infos (ca.

105kB)

5.13.3 Function Documentation

5.13.3.1 tau_delnitTTI()

Release any resources allocated by TTI Must be called before closing the session.

Function to terminate TTI access.

Parameters

in	appHandle	Handle returned by tlc_openSession().
----	-----------	---------------------------------------

Return values

none

5.13.3.2 tau_getCstFctCnt()

Function to retrieve the total number of functions in a consist.

Parameters

in	appHandle	Handle returned by tlc_openSession().	
out	pCstFctCnt	Pointer to the number of functions to be returned	
in pCstLabel Pointer to a consist label. NULL means own con-		Pointer to a consist label. NULL means own consist.	

Return values

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	Parameter error

5.13.3.3 tau_getCstFctInfo()

Function to retrieve the function information of the consist.

Parameters

in	appHandle	Handle returned by tlc_openSession().
out	pFctInfo	Pointer to function info list to be returned. Memory needs to be provided by application. Set NULL if not used.
in	pCstLabel	Pointer to a consist label. NULL means own consist.
in	maxFctCnt	Maximal number of functions to be returned in provided buffer.

Return values

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	Parameter error

5.13.3.4 tau_getCstInfo()

```
EXT_DECL TRDP_ERR_T tau_getCstInfo (

TRDP_APP_SESSION_T appHandle,

TRDP_CONSIST_INFO_T * pCstInfo,

const TRDP_LABEL_T pCstLabel )
```

Function to retrieve the consist information of a train's consist.

Parameters

in	appHandle	Handle returned by tlc_openSession().
out	pCstInfo	Pointer to the consist info to be returned.
in	pCstLabel	Pointer to a consist label. NULL means own consist.

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	Parameter error

5.13.3.5 tau_getCstVehCnt()

Function to retrieve the total number of vehicles in a consist.

Parameters

in	appHandle	Handle returned by tlc_openSession().
out	pCstVehCnt	Pointer to the number of vehicles to be returned
in	pCstLabel	Pointer to a consist label. NULL means own consist.

Return values

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	Parameter error
TRDP_NODATA_ERR	Try again

5.13.3.6 tau_getOpTrDirectory()

Function to retrieve the operational train directory state.

Parameters

in	appHandle	Handle returned by tlc_openSession().
out	pOpTrnDirState	Pointer to an operational train directory state structure to be returned.
out	pOpTrnDir	Pointer to an operational train directory structure to be returned.

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	Parameter error
TRDP_NODATA_ERR	Data currently not available, try again later

5.13.3.7 tau_getOpTrnDirectoryStatusInfo()

Function to retrieve the operational train directory state info.

Return a copy of the last received PD 100 telegram. Note: The values are in host endianess! When validating (v2), network endianess must be ensured.

Parameters

in	appHandle	Handle returned by tlc_openSession().
out	pOpTrnDirStatusInfo	Pointer to an operational train directory state structure to be returned.

Return values

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	Parameter error

5.13.3.8 tau_getOwnIds()

```
EXT_DECL TRDP_ERR_T tau_getOwnIds (

TRDP_APP_SESSION_T appHandle,

TRDP_LABEL_T * pDevId,

TRDP_LABEL_T * pVehId,

TRDP_LABEL_T * pCstId )
```

Who am I?.

Realizes a kind of 'Who am I' function. It is used to determine the own identifiers (i.e. the own labels), which may be used as host part of the own fully qualified domain name.

Parameters

in	appHandle	Handle returned by tlc_openSession()
out	pDevld	Returns the device label (host name)
out	pVehld	Returns the vehicle label
out	pCstld	Returns the consist label

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	Parameter error
TRDP_NODATA_ERR	Data currently not available, call again

5.13.3.9 tau_getOwnOpCstNo()

Get own operational consist number.

Parameters

i	n	appHandle	The handle returned by tlc_init
---	---	-----------	---------------------------------

Return values

ownOpCstNo	own operational consist number value 0 on error
------------	---

5.13.3.10 tau_getOwnTrnCstNo()

Get own train consist number.

Parameters

in	appHandle	The handle returned by tlc_init
----	-----------	---------------------------------

Return values

ownTrnCstNo	own train consist number value 0 on error
-------------	---

5.13.3.11 tau_getStaticCstInfo()

Function to retrieve the consist info.

Function to retrieve the operational train directory.

Parameters

in	appHandle	Handle returned by tlc_openSession().
out	pCstInfo	Pointer to a consist info structure to be returned.
in	cstUUID	UUID of the consist the consist info is rquested for.

Return values

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	Parameter error

5.13.3.12 tau_getTrDirectory()

Function to retrieve the train directory.

Parameters

	in	appHandle	Handle returned by tlc_openSession().
-	out	pTrnDir	Pointer to a train directory structure to be returned.

Return values

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	Parameter error
TRDP_NODATA_ERR	Try later

5.13.3.13 tau_getTrnCstCnt()

Function to retrieve the total number of consists in the train.

Parameters

in	appHandle	Handle returned by tlc_openSession().
out	pTrnCstCnt	Pointer to the number of consists to be returned

TRDP NO ERR	no error
TRDP_PARAM_ERR	Parameter error
TRDP_NODATA_ERR	Try again

5.13.3.14 tau_getTrnVehCnt()

Function to retrieve the total number of vehicles in the train.

Parameters

in	appHandle	Handle returned by tlc_openSession().
out	pTrnVehCnt	Pointer to the number of vehicles to be returned

Return values

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	Parameter error
TRDP_NODATA_ERR	Try again

5.13.3.15 tau_getTTI()

Function to retrieve the operational train directory.

Parameters

in	appHandle	Handle returned by tlc_openSession().
out	pOpTrnDirState	Pointer to an operational train directory state structure to be returned.
out	pOpTrnDir	Pointer to an operational train directory structure to be returned.
out	pTrnDir	Pointer to a train directory structure to be returned.
out	pTrnNetDir	Pointer to a train network directory structure to be returned.

Return values

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	Parameter error

5.13.3.16 tau_getVehInfo()

```
TRDP_VEHICLE_INFO_T * pVehInfo,
const TRDP_LABEL_T pVehLabel,
const TRDP_LABEL_T pCstLabel)
```

Function to retrieve the vehicle information of a consist's vehicle.

Parameters

in	appHandle	Handle returned by tlc_openSession().
out	pVehInfo	Pointer to the vehicle info to be returned.
in	pVehLabel	Pointer to a vehicle label. NULL means own vehicle if cstLabel refers to own consist.
in	pCstLabel	Pointer to a consist label. NULL means own consist.

Return values

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	Parameter error

5.13.3.17 tau_getVehOrient()

Function to retrieve the orientation of the given vehicle.

Parameters

in	appHandle	Handle returned by tlc_openSession().	
out	pVehOrient	Pointer to the vehicle orientation to be returned '00'B = not known (corrected vehicle)	
		'01'B = same as operational train direction '10'B = inverse to operational train direction	
out	pCstOrient	Pointer to the consist orientation to be returned '00'B = not known (corrected vehicle)	
		'01'B = same as operational train direction '10'B = inverse to operational train direction	
in	pVehLabel	vehLabel = NULL means own vehicle if cstLabel == NULL, currently ignored.	
in	pCstLabel	cstLabel = NULL means own consist	

Return values

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	Parameter error

5.13.3.18 tau_initTTlaccess()

```
EXT_DECL TRDP_ERR_T tau_initTTIaccess (
```

```
TRDP_APP_SESSION_T appHandle,
VOS_SEMA_T userAction,
TRDP_IP_ADDR_T ecspIpAddr,
CHAR8 * hostsFileName )
```

Function to init TTI access.

Subscribe to necessary process data for correct ECSP handling, further calls need DNS!

Parameters

in	appHandle	Handle returned by tlc_openSession().
in	userAction	Semaphore to fire if inauguration took place.
in	ecsplpAddr	ECSP IP address. Currently not used.
in	hostsFileName	Optional host file name as ECSP replacement. Currently not implemented.

Return values

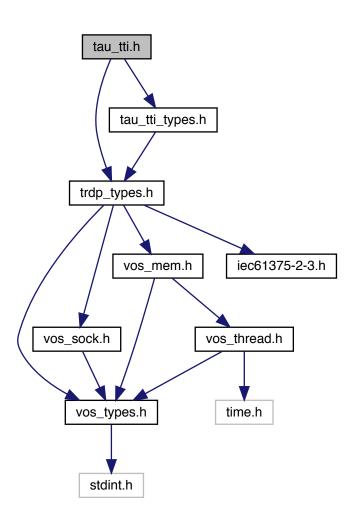
TRDP_NO_ERR	no error
TRDP_INIT_ERR	initialisation error

5.14 tau_tti.h File Reference

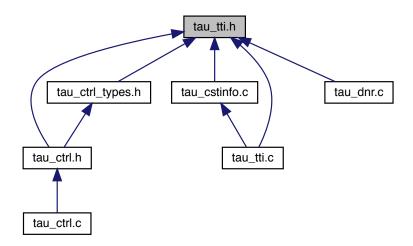
TRDP utility interface definitions.

```
#include "trdp_types.h"
#include "tau_tti_types.h"
```

Include dependency graph for tau_tti.h:



This graph shows which files directly or indirectly include this file:



Functions

EXT_DECL TRDP_ERR_T tau_initTTlaccess (TRDP_APP_SESSION_T appHandle, VOS_SEMA_T user

 Action, TRDP_IP_ADDR_T ecsplpAddr, CHAR8 *hostsFileName)

Function to init TTI access.

EXT_DECL void tau_delnitTTI (TRDP_APP_SESSION_T appHandle)

Function to terminate TTI access.

EXT_DECL TRDP_ERR_T tau_getOpTrDirectory (TRDP_APP_SESSION_T appHandle, TRDP_OP_TRA
 — IN_DIR_STATE_T *pOpTrnDirState, TRDP_OP_TRAIN_DIR_T *pOpTrnDir)

Function to retrieve the operational train directory state.

EXT_DECL TRDP_ERR_T tau_getOpTrnDirectoryStatusInfo (TRDP_APP_SESSION_T appHandle, TRD
 — P_OP_TRAIN_DIR_STATUS_INFO_T *pOpTrnDirStatusInfo)

Function to retrieve the operational train directory state info.

Function to retrieve the train directory.

EXT_DECL TRDP_ERR_T tau_getStaticCstInfo (TRDP_APP_SESSION_T appHandle, TRDP_CONSIST
 — INFO_T *pCstInfo, TRDP_UUID_T const cstUUID)

Function to retrieve the operational train directory.

• EXT_DECL TRDP_ERR_T tau_getTTI (TRDP_APP_SESSION_T appHandle, TRDP_OP_TRAIN_DIR_S
TATE_T *pOpTrnDirState, TRDP_OP_TRAIN_DIR_T *pOpTrnDir, TRDP_TRAIN_DIR_T *pTrnDir, TRDP
__TRAIN_NET_DIR_T *pTrnNetDir)

Function to retrieve the operational train directory.

- EXT_DECL TRDP_ERR_T tau_getTrnCstCnt (TRDP_APP_SESSION_T appHandle, UINT16 *pTrnCstCnt) Function to retrieve the total number of consists in the train.
- EXT_DECL TRDP_ERR_T tau_getTrnVehCnt (TRDP_APP_SESSION_T appHandle, UINT16 *pTrnVehCnt) Function to retrieve the total number of vehicles in the train.
- EXT_DECL TRDP_ERR_T tau_getCstVehCnt (TRDP_APP_SESSION_T appHandle, UINT16 *pCstVehCnt, const TRDP_LABEL_T pCstLabel)

Function to retrieve the total number of vehicles in a consist.

EXT_DECL TRDP_ERR_T tau_getCstFctCnt (TRDP_APP_SESSION_T appHandle, UINT16 *pCstFctCnt, const TRDP_LABEL_T pCstLabel)

Function to retrieve the total number of functions in a consist.

• EXT_DECL TRDP_ERR_T tau_getCstFctInfo (TRDP_APP_SESSION_T appHandle, TRDP_FUNCTION_

INFO_T *pFctInfo, const TRDP_LABEL_T pCstLabel, UINT16 maxFctCnt)

Function to retrieve the function information of the consist.

EXT_DECL TRDP_ERR_T tau_getVehInfo (TRDP_APP_SESSION_T appHandle, TRDP_VEHICLE_INF

 O_T *pVehInfo, const TRDP_LABEL_T pVehLabel, const TRDP_LABEL_T pCstLabel)

Function to retrieve the vehicle information of a consist's vehicle.

EXT_DECL TRDP_ERR_T tau_getCstInfo (TRDP_APP_SESSION_T appHandle, TRDP_CONSIST_INF
 O T *pCstInfo, const TRDP_LABEL_T pCstLabel)

Function to retrieve the consist information of a train's consist.

• EXT_DECL_TRDP_ERR_T_tau_getVehOrient (TRDP_APP_SESSION_T appHandle, UINT8 *pVehOrient, UINT8 *pCstOrient, TRDP_LABEL_T pVehLabel, TRDP_LABEL_T pCstLabel)

Function to retrieve the orientation of the given vehicle.

Who am I?.

EXT_DECL UINT8 tau_getOwnOpCstNo (TRDP_APP_SESSION_T appHandle)

Get own operational consist number.

• EXT_DECL UINT8 tau_getOwnTrnCstNo (TRDP_APP_SESSION_T appHandle)

Get own train consist number.

5.14.1 Detailed Description

TRDP utility interface definitions.

This module provides the interface to the following utilities

· train topology information access

Note

Project: TCNOpen TRDP prototype stack

Author

Armin-H. Weiss (initial version)

Remarks

This Source Code Form is subject to the terms of the Mozilla Public License, v. 2.0. If a copy of the MPL was not distributed with this file, You can obtain one at http://mozilla.org/MPL/2.0/. Copyright Bombardier Transportation Inc. or its subsidiaries and others, 2014. All rights reserved.

5.14.2 Function Documentation

5.14.2.1 tau_deInitTTI()

Function to terminate TTI access.

Parameters

	in	appHandle	Handle returned by tlc_openSession().	
--	----	-----------	---------------------------------------	--

Return values

Parameters

	in	appHandle	Handle returned by tlc_openSession().
--	----	-----------	---------------------------------------

Return values

```
none
```

5.14.2.2 tau_getCstFctCnt()

Function to retrieve the total number of functions in a consist.

Parameters

in	appHandle	Handle returned by tlc_openSession().
out	pCstFctCnt	Pointer to the number of functions to be returned
in	pCstLabel	Pointer to a consist label. NULL means own consist.

Return values

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	Parameter error

5.14.2.3 tau_getCstFctInfo()

Function to retrieve the function information of the consist.

Parameters

in	appHandle	Handle returned by tlc_openSession().	
out	pFctInfo	Pointer to function info list to be returned. Memory needs to be provided by application.	
	Set NULL if not used.		
in	pCstLabel	Pointer to a consist label. NULL means own consist.	
in	maxFctCnt	Maximal number of functions to be returned in provided buffer.	

Return values

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	Parameter error

5.14.2.4 tau_getCstInfo()

Function to retrieve the consist information of a train's consist.

Parameters

in	appHandle	Handle returned by tlc_openSession().
out	pCstInfo	Pointer to the consist info to be returned.
in	pCstLabel	Pointer to a consist label. NULL means own consist.

Return values

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	Parameter error

5.14.2.5 tau_getCstVehCnt()

Function to retrieve the total number of vehicles in a consist.

Parameters

i	.n	appHandle	Handle returned by tlc_openSession().
0	ut	pCstVehCnt	Pointer to the number of vehicles to be returned
i	.n	pCstLabel	Pointer to a consist label. NULL means own consist.

Return values

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	Parameter error

Parameters

in	appHandle	Handle returned by tlc_openSession().
out	pCstVehCnt	Pointer to the number of vehicles to be returned
in	pCstLabel	Pointer to a consist label. NULL means own consist.

Return values

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	Parameter error
TRDP_NODATA_ERR	Try again

5.14.2.6 tau_getOpTrDirectory()

Function to retrieve the operational train directory state.

Parameters

in	appHandle	Handle returned by tlc_openSession().
out	pOpTrnDirState	Pointer to an operational train directory state structure to be returned.
out	pOpTrnDir	Pointer to an operational train directory structure to be returned.

Return values

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	Parameter error

Parameters

in	appHandle	Handle returned by tlc_openSession().
out	pOpTrnDirState	Pointer to an operational train directory state structure to be returned.
out	pOpTrnDir	Pointer to an operational train directory structure to be returned.

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	Parameter error
TRDP_NODATA_ERR	Data currently not available, try again later

5.14.2.7 tau_getOpTrnDirectoryStatusInfo()

Function to retrieve the operational train directory state info.

Return a copy of the last received PD 100 telegram. Note: The values are in host endianess! When validating (SDTv2), network endianess must be ensured.

Parameters

-	in	appHandle	Handle returned by tlc_openSession().
(out	pOpTrnDirStatusInfo	Pointer to an operational train directory state structure to be returned.

Return values

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	Parameter error

Return a copy of the last received PD 100 telegram. Note: The values are in host endianess! When validating (v2), network endianess must be ensured.

Parameters

in	appHandle	Handle returned by tlc_openSession().
out	pOpTrnDirStatusInfo	Pointer to an operational train directory state structure to be returned.

Return values

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	Parameter error

5.14.2.8 tau_getOwnlds()

```
EXT_DECL TRDP_ERR_T tau_getOwnIds (

TRDP_APP_SESSION_T appHandle,

TRDP_LABEL_T * pDevId,

TRDP_LABEL_T * pVehId,

TRDP_LABEL_T * pCstId )
```

Who am I?.

Realizes a kind of 'Who am I' function. It is used to determine the own identifiers (i.e. the own labels), which may be used as host part of the own fully qualified domain name.

Parameters

in	appHandle	Handle returned by tlc_openSession()
out	pDevld	Returns the device label (host name)
out	pVehld	Returns the vehicle label
out	pCstld	Returns the consist label

Return values

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	Parameter error
TRDP_NODATA_ERR	Data currently not available, call again

5.14.2.9 tau_getOwnOpCstNo()

Get own operational consist number.

Parameters

in	appHandle	The handle returned by tlc_init
----	-----------	---------------------------------

Return values

ownOp	CstNo	own operational consist number value 0 on error
-------	-------	---

5.14.2.10 tau_getOwnTrnCstNo()

Get own train consist number.

Parameters

in	appHandle	The handle returned by tlc_init
----	-----------	---------------------------------

own IrnCstNo own train consist number value 0 on erro	ownTrnCstNo	own train consist number value 0 on error
---	-------------	---

5.14.2.11 tau_getStaticCstInfo()

```
EXT_DECL TRDP_ERR_T tau_getStaticCstInfo (

TRDP_APP_SESSION_T appHandle,

TRDP_CONSIST_INFO_T * pCstInfo,

TRDP_UUID_T const cstUUID )
```

Function to retrieve the operational train directory.

Parameters

in	appHandle	Handle returned by tlc_openSession().
out	pCstInfo	Pointer to a consist info structure to be returned.
in	cstUUID	UUID of the consist the consist info is rquested for.

Return values

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	Parameter error

Function to retrieve the operational train directory.

Parameters

in	appHandle	Handle returned by tlc_openSession().
out	pCstInfo	Pointer to a consist info structure to be returned.
in	cstUUID	UUID of the consist the consist info is rquested for.

Return values

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	Parameter error

5.14.2.12 tau_getTrDirectory()

Function to retrieve the train directory.

Parameters

in	appHandle	Handle returned by tlc_openSession().
out	pTrnDir	Pointer to a train directory structure to be returned.

TRDP_NO_ERR	no error

Return values

TRDP_PARAM_ERR	Parameter error
TRDP_NODATA_ERR	Try later

5.14.2.13 tau_getTrnCstCnt()

Function to retrieve the total number of consists in the train.

Parameters

in	appHandle	Handle returned by tlc_openSession().
out	pTrnCstCnt	Pointer to the number of consists to be returned

Return values

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	Parameter error

Parameters

in	appHandle	Handle returned by tlc_openSession().
out	pTrnCstCnt	Pointer to the number of consists to be returned

Return values

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	Parameter error
TRDP_NODATA_ERR	Try again

5.14.2.14 tau_getTrnVehCnt()

Function to retrieve the total number of vehicles in the train.

in	appHandle	Handle returned by tlc_openSession().
out	pTrnVehCnt	Pointer to the number of vehicles to be returned

Return values

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	Parameter error

Parameters

i	n	appHandle	Handle returned by tlc_openSession().
01	ut	pTrnVehCnt	Pointer to the number of vehicles to be returned

Return values

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	Parameter error
TRDP_NODATA_ERR	Try again

5.14.2.15 tau_getTTI()

```
EXT_DECL TRDP_ERR_T tau_getTTI (

TRDP_APP_SESSION_T appHandle,

TRDP_OP_TRAIN_DIR_STATE_T * pOpTrnDirState,

TRDP_OP_TRAIN_DIR_T * pOpTrnDir,

TRDP_TRAIN_DIR_T * pTrnDir,

TRDP_TRAIN_NET_DIR_T * pTrnNetDir )
```

Function to retrieve the operational train directory.

Parameters

in	appHandle	Handle returned by tlc_openSession().	
out	pOpTrnDirState Pointer to an operational train directory state structure to be returned.		
out	pOpTrnDir	Pointer to an operational train directory structure to be returned.	
out	pTrnDir Pointer to a train directory structure to be returned.		
out	pTrnNetDir	Pointer to a train network directory structure to be returned.	

Return values

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	Parameter error

5.14.2.16 tau_getVehInfo()

```
const TRDP_LABEL_T pVehLabel,
const TRDP_LABEL_T pCstLabel )
```

Function to retrieve the vehicle information of a consist's vehicle.

Parameters

in	appHandle	Handle returned by tlc_openSession().
out	pVehInfo	Pointer to the vehicle info to be returned.
in	pVehLabel	Pointer to a vehicle label. NULL means own vehicle if cstLabel refers to own consist.
in	pCstLabel	Pointer to a consist label. NULL means own consist.

Return values

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	Parameter error

5.14.2.17 tau_getVehOrient()

Function to retrieve the orientation of the given vehicle.

Parameters

in	appHandle	Handle returned by tlc_openSession().
out	pVehOrient	Pointer to the vehicle orientation to be returned '00'B = not known (corrected vehicle)
		'01'B = same as operational train direction '10'B = inverse to operational train direction
out	pCstOrient	Pointer to the consist orientation to be returned '00'B = not known (corrected vehicle)
		'01'B = same as operational train direction '10'B = inverse to operational train direction
in	pVehLabel	vehLabel = NULL means own vehicle if cstLabel == NULL
in	pCstLabel	cstLabel = NULL means own consist

Return values

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	Parameter error

in	appHandle	Handle returned by tlc_openSession().	
out	pVehOrient	Pointer to the vehicle orientation to be returned '00'B = not known (corrected vehicle) '01'B = same as operational train direction '10'B = inverse to operational train direction	
out	Pointer to the consist orientation to be returned '00'B = not known (corrected vehicle) '01'B = same as operational train direction '10'B = inverse to operational train direction		

Parameters

	in	pVehLabel	vehLabel = NULL means own vehicle if cstLabel == NULL, currently ignored.
ſ	in	pCstLabel	cstLabel = NULL means own consist

Return values

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	Parameter error

5.14.2.18 tau_initTTlaccess()

Function to init TTI access.

Parameters

in	appHandle	Handle returned by tlc_openSession().
in	userAction	Semaphore to fire if inauguration took place.
in	ecsplpAddr	ECSP IP address.
in	hostsFileName	Optional host file name as ECSP replacement.

Return values

TRDP_NO_ERR	no error
TRDP_INIT_ERR	initialisation error

Subscribe to necessary process data for correct ECSP handling, further calls need DNS!

Parameters

in	appHandle	Handle returned by tlc_openSession().
in	userAction	Semaphore to fire if inauguration took place.
in	ecsplpAddr	ECSP IP address. Currently not used.
in	hostsFileName	Optional host file name as ECSP replacement. Currently not implemented.

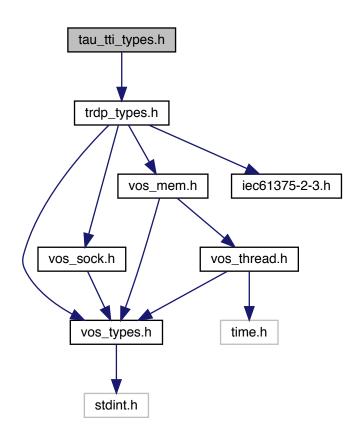
Return values

TRDP_NO_ERR	no error
TRDP_INIT_ERR	initialisation error

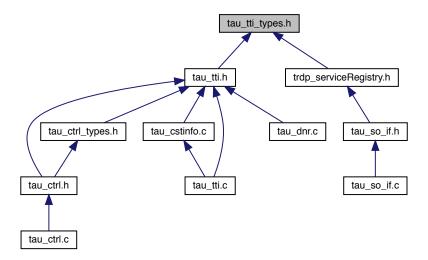
5.15 tau_tti_types.h File Reference

TRDP utility interface definitions.

#include "trdp_types.h"
Include dependency graph for tau_tti_types.h:



This graph shows which files directly or indirectly include this file:



Data Structures

struct GNU_PACKED

Types for ETB control.

struct TRDP_ETB_INFO_T

Types for train configuration information.

struct TRDP_CLTR_CST_INFO_T

Closed train consists information.

struct TRDP_PROP_T

Application defined properties.

struct TRDP_FUNCTION_INFO_T

function/device information structure

struct TRDP_VEHICLE_INFO_T

vehicle information structure

struct TRDP_CONSIST_INFO_T

consist information structure

struct GNU_PACKED

Types for ETB control.

struct GNU_PACKED

Types for ETB control.

struct GNU_PACKED

Types for ETB control.

struct GNU PACKED

Types for ETB control.

struct GNU_PACKED

Types for ETB control.

• struct GNU_PACKED

Types for ETB control.

struct GNU_PACKED

Types for ETB control.

• struct GNU_PACKED

Types for ETB control.

Macros

• #define TRDP_MAX_CST_CNT 63u

max number of consists per train

• #define TRDP_MAX_VEH_CNT 63u

max number of vehicles per train

5.15.1 Detailed Description

TRDP utility interface definitions.

This module provides the interface to the following utilities

• train topology information access type definitions acc. to IEC61375-2-3

Note

Project: TCNOpen TRDP prototype stack

Author

Armin-H. Weiss (initial version)

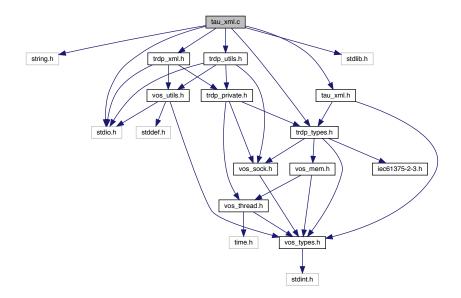
Remarks

This Source Code Form is subject to the terms of the Mozilla Public License, v. 2.0. If a copy of the MPL was not distributed with this file, You can obtain one at http://mozilla.org/MPL/2.0/. Copyright Bombardier Transportation Inc. or its subsidiaries and others, 2014. All rights reserved.

5.16 tau_xml.c File Reference

Functions for XML file parsing.

```
#include <string.h>
#include <stdio.h>
#include <stdlib.h>
#include "trdp_types.h"
#include "trdp_utils.h"
#include "tau_xml.h"
#include "trdp_xml.h"
Include dependency graph for tau xml.c:
```



Macros

• #define TRDP_SDT_DEFAULT_SMI2 0u

Default SDT safe message identifier.

• #define TRDP_SDT_DEFAULT_NRXSAFE 3u

Default SDT timeout cycles.

#define TRDP_SDT_DEFAULT_NGUARD 100u

Default SDT initial timeout cycles.

• #define TRDP_SDT_DEFAULT_CMTHR 10u

Default SDT chan.

#define TRDP_SDT_DEFAULT_LMIMAX (11u*TRDP_SDT_DEFAULT_NRXSAFE)

Default SDT chan.

Functions

• EXT_DECL TRDP_ERR_T tau_prepareXmlDoc (const CHAR8 *pFileName, TRDP_XML_DOC_HANDLE ← _ T *pDocHnd)

Open XML file, prepare XPath context.

EXT_DECL TRDP_ERR_T tau_prepareXmlMem (char *pBuffer, size_t bufSize, TRDP_XML_DOC_HAND

 LE T *pDocHnd)

Open XML stream, prepare XPath context.

EXT_DECL void tau_freeXmlDoc (TRDP_XML_DOC_HANDLE_T *pDocHnd)

Free all the memory allocated by tau_prepareXmlDoc.

EXT_DECL TRDP_ERR_T tau_readXmlInterfaceConfig (const TRDP_XML_DOC_HANDLE_T *pDocHnd, const CHAR8 *plfName, TRDP_PROCESS_CONFIG_T *pProcessConfig, TRDP_PD_CONFIG_T *p← PdConfig, TRDP_MD_CONFIG_T *pMdConfig, UINT32 *pNumExchgPar, TRDP_EXCHG_PAR_T **pp← ExchgPar)

Read the interface relevant telegram parameters (except data set configuration) out of the configuration file .

- EXT_DECL void tau_freeTelegrams (UINT32 numExchgPar, TRDP_EXCHG_PAR_T *pExchgPar)

 Free array of telegram configurations allocated by tau_readXmlInterfaceConfig.
- EXT_DECL_TRDP_ERR_T tau_readXmlDeviceConfig (const_TRDP_XML_DOC_HANDLE_T *pDocHnd, TRDP_MEM_CONFIG_T *pMemConfig, TRDP_DBG_CONFIG_T *pDbgConfig, UINT32 *pNumComPar, TRDP_COM_PAR_T **ppComPar, UINT32 *pNumIfConfig, TRDP_IF_CONFIG_T **pplfConfig)

Function to read the TRDP device configuration parameters out of the XML configuration file.

 EXT_DECL TRDP_ERR_T tau_readXmlMappedDevices (const TRDP_XML_DOC_HANDLE_T *pDocHnd, UINT32 *pNumProcConfig, TRDP_PROCESS_CONFIG_T **ppProcessConfig)

Function to read the TRDP mapped devices out of the XML configuration file.

• EXT_DECL TRDP_ERR_T tau_readXmlMappedDeviceConfig (const TRDP_XML_DOC_HANDLE_T *p↔ DocHnd, const CHAR8 *pHostname, UINT32 *pNumlfConfig, TRDP_IF_CONFIG_T **pplfConfig)

Function to read the TRDP mapped device configuration parameters for a particular host out of the XML configuration file.

EXT_DECL TRDP_ERR_T tau_readXmlMappedInterfaceConfig (const TRDP_XML_DOC_HANDLE_T *p
 — DocHnd, const CHAR8 *pHostname, const CHAR8 *plfName, UINT32 *pNumExchgPar, TRDP_EXCHG
 — PAR_T **ppExchgPar)

Read the interface relevant mapped telegram parameters for a particular host and it's interface out of the configuration file

 EXT_DECL TRDP_ERR_T tau_readXmlDatasetConfig (const TRDP_XML_DOC_HANDLE_T *pDocHnd, UINT32 *pNumComId, TRDP_COMID_DSID_MAP_T **ppComIdDsIdMap, UINT32 *pNumDataset, ap TRDP_DATASET_T *apDataset)

Function to read the DataSet configuration out of the XML configuration file.

EXT_DECL void tau_freeXmlDatasetConfig (UINT32 numComId, TRDP_COMID_DSID_MAP_T *pComId←DsIdMap, UINT32 numDataset, TRDP_DATASET_T **ppDataset)

Function to free the memory for the DataSet configuration.

 EXT_DECL_TRDP_ERR_T tau_readXmlServiceConfig (const_TRDP_XML_DOC_HANDLE_T *pDocHnd, UINT32 *pNumServiceDefs, TRDP_SERVICE_DEF_T **ppServiceDefs)

Function to read the TRDP device service definitions out of the XML configuration file.

5.16.1 Detailed Description

Functions for XML file parsing.

SOX parsing of XML configuration file

Note

Project: TCNOpen TRDP prototype stack

Author

B. Loehr, NewTec GmbH, Tomas Svoboda, UniControls a.s.

Remarks

This Source Code Form is subject to the terms of the Mozilla Public License, v. 2.0. If a copy of the MPL was not distributed with this file, You can obtain one at http://mozilla.org/MPL/2.0/. Copyright NewTec GmbH, 2016-2019. All rights reserved.

5.16.2 Macro Definition Documentation

5.16.2.1 TRDP_SDT_DEFAULT_CMTHR

```
#define TRDP_SDT_DEFAULT_CMTHR 10u
```

Default SDT chan.

monitoring threshold

5.16.2.2 TRDP_SDT_DEFAULT_LMIMAX

```
#define TRDP_SDT_DEFAULT_LMIMAX (11u*TRDP_SDT_DEFAULT_NRXSAFE)
```

Default SDT chan.

latency monitoring cycles

5.16.3 Function Documentation

5.16.3.1 tau_freeTelegrams()

Free array of telegram configurations allocated by tau_readXmlInterfaceConfig.

Parameters

in	numExchgPar	Number of telegram configurations in the array
in	pExchgPar	Pointer to array of telegram configurations

5.16.3.2 tau_freeXmlDatasetConfig()

Function to free the memory for the DataSet configuration.

Free the memory for the DataSet configuration which was allocated when parsing the XML configuration file.

Parameters

in	numComId	The number of entries in the Comld DatasetId mapping list
in	pComldDsldMap	Pointer to an array of structures of type TRDP_COMID_DSID_MAP_T
in	numDataset	The number of datasets found in the configuration
in	ppDataset	Pointer to an array of pointers to a structures of type TRDP_DATASET_T

Return values

```
none
```

5.16.3.3 tau_freeXmlDoc()

Free all the memory allocated by tau_prepareXmlDoc.

Parameters

in pDocHnd Handle of the parsed XML file
--

5.16.3.4 tau_prepareXmlDoc()

Open XML file, prepare XPath context.

Load XML file into DOM tree, prepare XPath context.

Parameters

in	pFileName	Path and filename of the xml configuration file
out	pDocHnd	Handle of the parsed XML file

Return values

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	File does not exist

5.16.3.5 tau_prepareXmlMem()

Open XML stream, prepare XPath context.

Parameters

in	pBuffer	Pointer to the xml configuration stream buffer
in	bufSize	Size of the xml configuration stream buffer
out	pDocHnd	Pointer to the handle of the parsed XML file

Return values

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	File does not exist

5.16.3.6 tau_readXmlDatasetConfig()

Function to read the DataSet configuration out of the XML configuration file.

Parameters

in	pDocHnd	Handle of the XML document prepared by tau_prepareXmlDoc
out	pNumComId	Pointer to the number of entries in the Comld DatasetId mapping list
out	ppComIdDsIdMap	Pointer to an array of a structures of type TRDP_COMID_DSID_MAP_T
out	pNumDataset	Pointer to the number of datasets found in the configuration
out	apDataset	Pointer to an array of pointers to a structure of type TRDP_DATASET_T

Return values

TRDP_NO_ERR	no error
TRDP_MEM_ERR	provided buffer to small
TRDP_PARAM_ERR	File not existing

5.16.3.7 tau_readXmlDeviceConfig()

Function to read the TRDP device configuration parameters out of the XML configuration file.

The user must release the memory for ppComPar and pplfConfig (using vos_memFree)

Parameters

in	pDocHnd	Handle of the XML document prepared by tau_prepareXmlDoc	
out	pMemConfig	Memory configuration	
out	pDbgConfig	Debug printout configuration for application use	
out	pNumComPar	Number of configured com parameters	
out	ppComPar	Pointer to array of com parameters	
out	pNumlfConfig	Number of configured interfaces	
out	pplfConfig	Pointer to an array of interface parameter sets	

Return values

TRDP_NO_ERR	no error
TRDP_MEM_ERR	provided buffer to small
TRDP_PARAM_ERR	File not existing

5.16.3.8 tau_readXmlInterfaceConfig()

Read the interface relevant telegram parameters (except data set configuration) out of the configuration file .

in	pDocHnd	Handle of the XML document prepared by tau_prepareXmlDoc	
in	plfName	Interface name	
out	pProcessConfig	TRDP process (session) configuration for the interface	
out	pPdConfig	PD default configuration for the interface	
out	pMdConfig	MD default configuration for the interface	
Generated	ырЫыggEnxchgPar	Number of configured telegrams	
out	ppExchgPar	Pointer to array of telegram configurations	

Return values

TRDP_NO_ERR	no error
TRDP_MEM_ERR	provided buffer to small
TRDP_PARAM_ERR	File not existing

5.16.3.9 tau_readXmlMappedDeviceConfig()

Function to read the TRDP mapped device configuration parameters for a particular host out of the XML configuration file.

Parameters

in	pDocHnd	Handle of the XML document prepared by tau_prepareXmlDoc	
in	pHostname	Host name for which interface config is to be read	
out	pNumIfConfig	Number of configured interfaces for this host	
out	pplfConfig	Pointer to an array of interface parameter sets	

Return values

TRDP_NO_ERR	no error
TRDP_MEM_ERR	provided buffer to small
TRDP_PARAM_ERR	File not existing

5.16.3.10 tau_readXmlMappedDevices()

Function to read the TRDP mapped devices out of the XML configuration file.

in	pDocHnd	Handle of the XML document prepared by tau_prepareXmlDoc
out	pNumProcConfig	Number of configured mapped devices
out	ppProcessConfig	Pointer to an array of mapped devices configuration

Return values

TRDP_NO_ERR	no error
TRDP_MEM_ERR	provided buffer to small
TRDP_PARAM_ERR	File not existing

5.16.3.11 tau_readXmlMappedInterfaceConfig()

Read the interface relevant mapped telegram parameters for a particular host and it's interface out of the configuration file .

Parameters

in	pDocHnd	Handle of the XML document prepared by tau_prepareXmlDoc
in	pHostname	Host name
in	plfName	Interface name
out	pNumExchgPar	Number of configured telegrams
out	ppExchgPar	Pointer to array of telegram configurations

Return values

TRDP_NO_ERR	no error
TRDP_MEM_ERR	provided buffer to small
TRDP_PARAM_ERR	File not existing

5.16.3.12 tau_readXmlServiceConfig()

Function to read the TRDP device service definitions out of the XML configuration file.

The user must release the memory for pServiceDefs (using vos_memFree)

Parameters

in	pDocHnd	Handle of the XML document prepared by tau_prepareXmlDoc
out	pNumServiceDefs	Pointer to number of defined Services
out	ppServiceDefs	Pointer to pointer of the defined Services

Generated by Doxygen

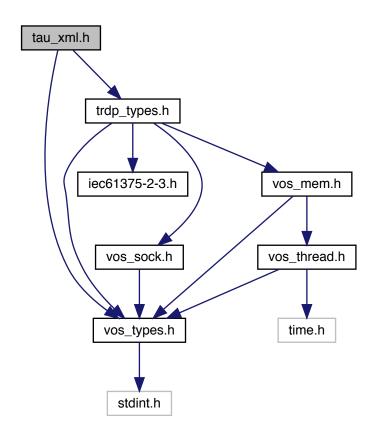
Return values

TRDP_NO_ERR	no error
TRDP_MEM_ERR	provided buffer to small
TRDP_PARAM_ERR	File not existing

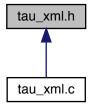
5.17 tau_xml.h File Reference

TRDP utility interface definitions.

```
#include "vos_types.h"
#include "trdp_types.h"
Include dependency graph for tau_xml.h:
```



This graph shows which files directly or indirectly include this file:



Data Structures

struct TRDP_SDT_PAR_T

Types to read out the XML configuration.

struct TRDP_DBG_CONFIG_T

Control for debug output device/file on application level.

• struct TRDP_XML_DOC_HANDLE_T

Parsed XML document handle.

Macros

• #define TRDP_DBG_DEFAULT 0

Control for debug output format on application level.

• #define TRDP_DBG_OFF 0x01

Printout off.

• #define TRDP_DBG_ERR 0x02

Printout error.

#define TRDP_DBG_WARN 0x04

Printout warning and error.

• #define TRDP DBG INFO 0x08

Printout info, warning and error.

• #define TRDP_DBG_DBG 0x10

Printout debug, info, warning and error.

• #define TRDP DBG TIME 0x20

Printout timestamp.

#define TRDP_DBG_LOC 0x40

Printout file name and line.

• #define TRDP_DBG_CAT 0x80

Printout category (DBG, INFO, WARN, ERR)

Enumerations

```
    enum TRDP_EXCHG_OPTION_T {
        TRDP_EXCHG_UNSET = 0,
        TRDP_EXCHG_SOURCE = 1,
        TRDP_EXCHG_SINK = 2,
        TRDP_EXCHG_SOURCESINK = 3 }
```

Type attribute for telegrams.

Functions

EXT_DECL TRDP_ERR_T tau_prepareXmlDoc (const CHAR8 *pFileName, TRDP_XML_DOC_HANDLE
 — T *pDocHnd)

Load XML file into DOM tree, prepare XPath context.

EXT_DECL TRDP_ERR_T tau_prepareXmlMem (char *pBuffer, size_t bufSize, TRDP_XML_DOC_HAND
 LE_T *pDocHnd)

Open XML stream, prepare XPath context.

EXT_DECL void tau_freeXmlDoc (TRDP_XML_DOC_HANDLE_T *pDocHnd)

Free all the memory allocated by tau_prepareXmlDoc.

• EXT_DECL_TRDP_ERR_T_tau_readXmlDeviceConfig (const_TRDP_XML_DOC_HANDLE_T_*pDocHnd, TRDP_MEM_CONFIG_T *pMemConfig, TRDP_DBG_CONFIG_T *pDbgConfig, UINT32 *pNumComPar, TRDP_COM_PAR_T **ppComPar, UINT32 *pNumIfConfig, TRDP_IF_CONFIG_T **ppIfConfig)

Function to read the TRDP device configuration parameters out of the XML configuration file.

EXT_DECL TRDP_ERR_T tau_readXmlInterfaceConfig (const TRDP_XML_DOC_HANDLE_T *pDocHnd, const CHAR8 *plfName, TRDP_PROCESS_CONFIG_T *pProcessConfig, TRDP_PD_CONFIG_T *p← PdConfig, TRDP_MD_CONFIG_T *pMdConfig, UINT32 *pNumExchgPar, TRDP_EXCHG_PAR_T **pp← ExchgPar)

Read the interface relevant telegram parameters (except data set configuration) out of the configuration file .

 EXT_DECL_TRDP_ERR_T tau_readXmlDatasetConfig (const_TRDP_XML_DOC_HANDLE_T *pDocHnd, UINT32 *pNumComId, TRDP_COMID_DSID_MAP_T **ppComIdDsIdMap, UINT32 *pNumDataset, pap← TRDP_DATASET_T papDataset)

Function to read the DataSet configuration out of the XML configuration file.

Function to free the memory for the DataSet configuration.

EXT_DECL void tau_freeTelegrams (UINT32 numExchgPar, TRDP_EXCHG_PAR_T *pExchgPar)

Free array of telegram configurations allocated by tau_readXmlInterfaceConfig.

 EXT_DECL TRDP_ERR_T tau_readXmlServiceConfig (const TRDP_XML_DOC_HANDLE_T *pDocHnd, UINT32 *pNumServiceDefs, TRDP SERVICE DEF T **ppServiceDefs)

Function to read the TRDP device service definitions out of the XML configuration file.

• EXT_DECL TRDP_ERR_T tau_readXmlMappedDevices (const TRDP_XML_DOC_HANDLE_T *pDocHnd, UINT32 *pNumProcConfig, TRDP PROCESS CONFIG T **ppProcessConfig)

Function to read the TRDP mapped devices out of the XML configuration file.

• EXT_DECL TRDP_ERR_T tau_readXmlMappedDeviceConfig (const TRDP_XML_DOC_HANDLE_T *p↔ DocHnd, const CHAR8 *pHostname, UINT32 *pNumlfConfig, TRDP_IF_CONFIG_T **pplfConfig)

Function to read the TRDP mapped device configuration parameters for a particular host out of the XML configuration file

EXT_DECL TRDP_ERR_T tau_readXmlMappedInterfaceConfig (const TRDP_XML_DOC_HANDLE_T *p↔
 DocHnd, const CHAR8 *pHostname, const CHAR8 *pIfName, UINT32 *pNumExchgPar, TRDP_EXCHG↔
 _PAR_T **ppExchgPar)

Read the interface relevant mapped telegram parameters for a particular host and it's interface out of the configuration

5.17.1 Detailed Description

TRDP utility interface definitions.

This module provides the interface to the following utilities

read xml configuration interpreter

Note

Project: TCNOpen TRDP prototype stack

Author

Armin-H. Weiss (initial version)

Remarks

This Source Code Form is subject to the terms of the Mozilla Public License, v. 2.0. If a copy of the MPL was not distributed with this file, You can obtain one at http://mozilla.org/MPL/2.0/. Copyright Bombardier Transportation Inc. or its subsidiaries and others, 2013. All rights reserved.

5.17.2 Macro Definition Documentation

5.17.2.1 TRDP_DBG_DEFAULT

```
#define TRDP_DBG_DEFAULT 0
```

Control for debug output format on application level.

Printout default

5.17.3 Enumeration Type Documentation

5.17.3.1 TRDP_EXCHG_OPTION_T

```
enum TRDP_EXCHG_OPTION_T
```

Type attribute for telegrams.

Enumerator

TRDP_EXCHG_UNSET	default, direction is not defined
TRDP_EXCHG_SOURCE	telegram shall be published
TRDP_EXCHG_SINK	telegram shall be subscribed
TRDP_EXCHG_SOURCESINK	telegram shall be published and subscribed

5.17.4 Function Documentation

5.17.4.1 tau_freeTelegrams()

Free array of telegram configurations allocated by tau_readXmlInterfaceConfig.

Parameters

in	numExchgPar	Number of telegram configurations in the array
in	pExchgPar	Pointer to array of telegram configurations

5.17.4.2 tau_freeXmlDatasetConfig()

Function to free the memory for the DataSet configuration.

Free the memory for the DataSet configuration which was allocated when parsing the XML configuration file.

Parameters

in	numComId	The number of entries in the Comld DatasetId mapping list
in	pComIdDsIdMap	Pointer to an array of structures of type TRDP_COMID_DSID_MAP_T
in	numDataset	The number of datasets found in the configuration
in	ppDataset	Pointer to an array of pointers to a structures of type TRDP_DATASET_T

Return values

none

5.17.4.3 tau_freeXmlDoc()

Free all the memory allocated by tau_prepareXmlDoc.

1			
ĺ	in	pDocHnd	Handle of the parsed XML file

5.17.4.4 tau_prepareXmlDoc()

Load XML file into DOM tree, prepare XPath context.

Parameters

in	pFileName	Path and filename of the xml configuration file
out	pDocHnd	Handle of the parsed XML file

Return values

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	File does not exist

Load XML file into DOM tree, prepare XPath context.

Parameters

in	pFileName	Path and filename of the xml configuration file
out	pDocHnd	Handle of the parsed XML file

Return values

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	File does not exist

5.17.4.5 tau_prepareXmlMem()

Open XML stream, prepare XPath context.

in	pBuffer	Pointer to the xml configuration stream buffer
in	bufSize	Size of the xml configuration stream buffer
out	pDocHnd	Pointer to the handle of the parsed XML file

Return values

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	File does not exist

5.17.4.6 tau_readXmlDatasetConfig()

Function to read the DataSet configuration out of the XML configuration file.

Parameters

in	pDocHnd	Handle of the XML document prepared by tau_prepareXmlDoc
out	pNumComId	Pointer to the number of entries in the Comld DatasetId mapping list
out	ppComldDsldMap	Pointer to an array of a structures of type TRDP_COMID_DSID_MAP_T
out	pNumDataset	Pointer to the number of datasets found in the configuration
out	papDataset	Pointer to an array of pointers to a structures of type TRDP_DATASET_T

Return values

TRDP_NO_ERR	no error
TRDP_MEM_ERR	provided buffer to small
TRDP_PARAM_ERR	File not existing

5.17.4.7 tau_readXmlDeviceConfig()

Function to read the TRDP device configuration parameters out of the XML configuration file.

in	pDocHnd	Handle of the XML document prepared by tau_prepareXmlDoc	
out	pMemConfig	Memory configuration	

Parameters

out	pDbgConfig	Debug printout configuration for application use
out	pNumComPar	Number of configured com parameters
out	ppComPar	Pointer to array of com parameters
out	pNumIfConfig	Number of configured interfaces
out	pplfConfig	Pointer to an array of interface parameter sets

Return values

TRDP_NO_ERR	no error
TRDP_MEM_ERR	provided buffer to small
TRDP_PARAM_ERR	File not existing

The user must release the memory for ppComPar and pplfConfig (using vos_memFree)

Parameters

in	pDocHnd	Handle of the XML document prepared by tau_prepareXmlDoc
out	pMemConfig	Memory configuration
out	pDbgConfig	Debug printout configuration for application use
out	pNumComPar	Number of configured com parameters
out	ppComPar	Pointer to array of com parameters
out	pNumlfConfig	Number of configured interfaces
out	pplfConfig	Pointer to an array of interface parameter sets

Return values

TRDP_NO_ERR	no error
TRDP_MEM_ERR	provided buffer to small
TRDP_PARAM_ERR	File not existing

5.17.4.8 tau_readXmlInterfaceConfig()

Read the interface relevant telegram parameters (except data set configuration) out of the configuration file .

in	pDocHnd	Handle of the XML document prepared by tau_prepareXmlDoc

Parameters

in	plfName	Interface name
out	pProcessConfig	TRDP process (session) configuration for the interface
out	pPdConfig	PD default configuration for the interface
out	pMdConfig	MD default configuration for the interface
out	pNumExchgPar	Number of configured telegrams
out	ppExchgPar	Pointer to array of telegram configurations

Return values

TRDP_NO_ERR	no error
TRDP_MEM_ERR	provided buffer to small
TRDP_PARAM_ERR	File not existing

5.17.4.9 tau_readXmlMappedDeviceConfig()

Function to read the TRDP mapped device configuration parameters for a particular host out of the XML configuration file.

Parameters

in	pDocHnd	Handle of the XML document prepared by tau_prepareXmlDoc
in	pHostname	Host name for which interface config is to be read
out	pNumIfConfig	Number of configured interfaces for this host
out	pplfConfig	Pointer to an array of interface parameter sets

Return values

TRDP_NO_ERR	no error
TRDP_MEM_ERR	provided buffer to small
TRDP_PARAM_ERR	File not existing

5.17.4.10 tau_readXmlMappedDevices()

Function to read the TRDP mapped devices out of the XML configuration file.

Parameters

	in	pDocHnd	Handle of the XML document prepared by tau_prepareXmlDoc	
	out	pNumProcConfig	Number of configured mapped devices	
Ī	out	ppProcessConfig	Pointer to an array of mapped devices configuration	

Return values

TRDP_NO_ERR	no error
TRDP_MEM_ERR	provided buffer to small
TRDP_PARAM_ERR	File not existing

5.17.4.11 tau_readXmlMappedInterfaceConfig()

Read the interface relevant mapped telegram parameters for a particular host and it's interface out of the configuration file .

Parameters

in	pDocHnd	Handle of the XML document prepared by tau_prepareXmlDoc
in	pHostname	Host name
in	plfName	Interface name
out	pNumExchgPar	Number of configured telegrams
out	ppExchgPar	Pointer to array of telegram configurations

Return values

TRDP_NO_ERR	no error
TRDP_MEM_ERR	provided buffer to small
TRDP_PARAM_ERR	File not existing

5.17.4.12 tau_readXmlServiceConfig()

Function to read the TRDP device service definitions out of the XML configuration file.

The user must release the memory for pServiceDefs (using vos_memFree)		

Parameters

	in	pDocHnd	Handle of the XML document prepared by tau_prepareXmlDoc	
	out	pNumServiceDefs	Number of defined Services	
ĺ	out	ppServiceDefs	Pointer to pointer of the defined Services	

Return values

TRDP_NO_ERR	no error
TRDP_MEM_ERR	provided buffer to small
TRDP_PARAM_ERR	File not existing

The user must release the memory for pServiceDefs (using vos_memFree)

Parameters

in	pDocHnd	Handle of the XML document prepared by tau_prepareXmlDoc
out	pNumServiceDefs	Pointer to number of defined Services
out	ppServiceDefs	Pointer to pointer of the defined Services

Return values

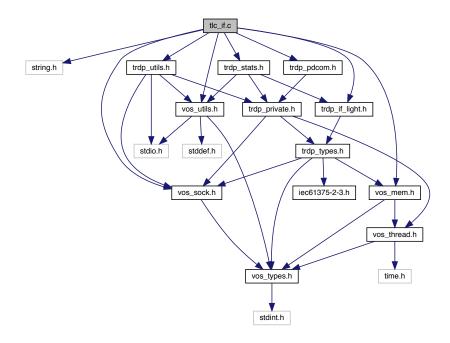
TRDP_NO_ERR	no error
TRDP_MEM_ERR	provided buffer to small
TRDP_PARAM_ERR	File not existing

5.18 tlc_if.c File Reference

Functions for ECN communication.

```
#include <string.h>
#include "trdp_if_light.h"
#include "trdp_utils.h"
#include "trdp_pdcom.h"
#include "trdp_stats.h"
#include "vos_sock.h"
#include "vos_mem.h"
#include "vos_utils.h"
```

Include dependency graph for tlc_if.c:



Functions

• BOOL8 trdp_isValidSession (TRDP_APP_SESSION_T pSessionHandle)

Check if the session handle is valid.

• TRDP_APP_SESSION_T * trdp_sessionQueue (void)

Get the session queue head pointer.

• TRDP_ERR_T trdp_getAccess (TRDP_APP_SESSION_T appHandle, int force)

Get mutual access to the session Take all mutexes of that session.

void trdp_releaseAccess (TRDP_APP_SESSION_T appHandle)

Release access to the session.

 $\bullet \ \ \mathsf{EXT_DECL} \ \mathsf{TRDP_IP_ADDR_T} \ \mathsf{tlc_getOwnIpAddress} \ (\mathsf{TRDP_APP_SESSION_T} \ \mathsf{appHandle})$

Get the interface address.

• EXT_DECL TRDP_ERR_T tlc_init (const TRDP_PRINT_DBG_T pPrintDebugString, void *pRefCon, const TRDP_MEM_CONFIG_T *pMemConfig)

Initialize the TRDP stack.

EXT_DECL TRDP_ERR_T tlc_openSession (TRDP_APP_SESSION_T *pAppHandle, TRDP_IP_ADDR
 — T ownlpAddr, TRDP_IP_ADDR_T leaderlpAddr, const TRDP_MARSHALL_CONFIG_T *pMarshall, const
 TRDP_PD_CONFIG_T *pPdDefault, const TRDP_MD_CONFIG_T *pMdDefault, const TRDP_PROCES
 — S_CONFIG_T *pProcessConfig)

Open a session with the TRDP stack.

 EXT_DECL TRDP_ERR_T tlc_configSession (TRDP_APP_SESSION_T appHandle, const TRDP_MAR← SHALL_CONFIG_T *pMarshall, const TRDP_PD_CONFIG_T *pPdDefault, const TRDP_MD_CONFIG_T *pMdDefault, const TRDP_PROCESS_CONFIG_T *pProcessConfig)

(Re-)configure a session.

• EXT DECL TRDP ERR Ttlc updateSession (TRDP APP SESSION TappHandle)

Update a session.

EXT_DECL TRDP_ERR_T tlc_presetIndexSession (TRDP_APP_SESSION_T appHandle, TRDP_IDX_T

 ABLE_T *pIndexTableSizes)

Preset the index table sizes of a session.

• EXT DECL TRDP ERR T tlc closeSession (TRDP APP SESSION T appHandle)

Close a session.

EXT_DECL TRDP_ERR_T tlc_terminate (void)

Un-Initialize.

• EXT_DECL TRDP_ERR_T tlc_reinitSession (TRDP_APP_SESSION_T appHandle)

Re-Initialize.

• EXT_DECL TRDP_ERR_T tlc_getInterval (TRDP_APP_SESSION_T appHandle, TRDP_TIME_T *pInterval, TRDP_FDS_T *pFileDesc, INT32 *pNoDesc)

Get the lowest time interval for PDs.

EXT_DECL TRDP_ERR_T tlc_process (TRDP_APP_SESSION_T appHandle, TRDP_FDS_T *pRfds, INT32 *pCount)

Work loop of the TRDP handler.

• EXT_DECL const char * tlc_getVersionString (void)

Return a human readable version representation.

• EXT_DECL const TRDP_VERSION_T * tlc_getVersion (void)

Return version

EXT_DECL TRDP_ERR_T tlc_setETBTopoCount (TRDP_APP_SESSION_T appHandle, UINT32 etbTopo

 Cnt)

Set new topocount for trainwide communication.

Set new operational train topocount for direction/orientation sensitive communication.

• EXT_DECL UINT32 tlc_getETBTopoCount (TRDP_APP_SESSION_T appHandle)

Set new topocount for trainwide communication.

EXT_DECL UINT32 tlc_getOpTrainTopoCount (TRDP_APP_SESSION_T appHandle)

Set new operational train topocount for direction/orientation sensitive communication.

5.18.1 Detailed Description

Functions for ECN communication.

API implementation of TRDP Light

Note

Project: TCNOpen TRDP prototype stack

Author

Bernd Loehr, NewTec GmbH

Remarks

This Source Code Form is subject to the terms of the Mozilla Public License, v. 2.0. If a copy of the MPL was not distributed with this file, You can obtain one at http://mozilla.org/MPL/2.0/. Copyright Bombardier Transportation Inc. or its subsidiaries and others, 2013-2019. All rights reserved.

5.18.2 Function Documentation

5.18.2.1 tlc_closeSession()

Close a session.

Clean up and release all resources of that session

Parameters

i	appHandle	The handle returned by tlc_openSession
---	-----------	--

Return values

TRDP_NO_ERR	no error
TRDP_NOINIT_ERR	handle invalid
TRDP_PARAM_ERR	handle NULL

5.18.2.2 tlc_configSession()

```
EXT_DECL TRDP_ERR_T tlc_configSession (

TRDP_APP_SESSION_T appHandle,

const TRDP_MARSHALL_CONFIG_T * pMarshall,

const TRDP_PD_CONFIG_T * pPdDefault,

const TRDP_MD_CONFIG_T * pMdDefault,

const TRDP_PROCESS_CONFIG_T * pProcessConfig )
```

(Re-)configure a session.

tlc_configSession is called by openSession, but may also be called later on to change the defaults. Only the supplied settings (pointer != NULL) will be evaluated.

Parameters

in	appHandle	A handle for further calls to the trdp stack
in	n pMarshall Pointer to marshalling configuration	
in	pPdDefault	Pointer to default PD configuration
in	pMdDefault	Pointer to default MD configuration
in	pProcessConfig	Pointer to process configuration only option parameter is used here to define session
		behavior all other parameters are only used to feed statistics

Return values

TRDP_NO_ERR	no error
-------------	----------

Return values

TRDP_INIT_ERR	not yet inited
TRDP_PARAM_ERR	parameter error

5.18.2.3 tlc_getETBTopoCount()

Set new topocount for trainwide communication.

This value is used for validating outgoing and incoming packets only!

Parameters

in	appHandle	the handle returned by tlc_openSession
----	-----------	--

Return values

etbTopoCnt

5.18.2.4 tlc_getInterval()

Get the lowest time interval for PDs.

Return the maximum time interval suitable for 'select()' so that we can send due PD packets in time. If the PD send queue is empty, return zero time

Parameters

in	appHandle	pHandle The handle returned by tlc_openSession	
out	pInterval	pointer to needed interval	
in,out	pFileDesc	pointer to file descriptor set	
out	pNoDesc	pointer to put no of highest used descriptors (for select())	

Return values

TRDP_NO_ERR	no error
TRDP_NOINIT_ERR	handle invalid

5.18.2.5 tlc_getOpTrainTopoCount()

Set new operational train topocount for direction/orientation sensitive communication.

This value is used for validating outgoing and incoming packets only!

Parameters

	in	appHandle	The handle returned by tlc_openSession
--	----	-----------	--

Return values

opTrnTopoCnt	New operational topocount value
--------------	---------------------------------

5.18.2.6 tlc_getOwnlpAddress()

Get the interface address.

Parameters

out	appHandle	A handle for further calls to the trdp stack
-----	-----------	--

Return values



5.18.2.7 tlc_getVersion()

Return version.

Return pointer to version structure

Return values

```
TRDP_VERSION↔
_T
```

5.18.2.8 tlc_getVersionString()

Return a human readable version representation.

Return string in the form 'v.r.u.b'

Return values

```
const string
```

5.18.2.9 tlc_init()

Initialize the TRDP stack.

Support for message data can only be excluded during compile time!

tlc_init initializes the memory subsystem and takes a function pointer to an output function for logging.

Parameters

in	pPrintDebugString	Pointer to debug print function
in	pRefCon	user context
in	pMemConfig	Pointer to memory configuration

Return values

TRDP_NO_ERR	no error
TRDP_MEM_ERR	memory allocation failed
TRDP PARAM ERR	initialization error

5.18.2.10 tlc_openSession()

```
EXT_DECL TRDP_ERR_T tlc_openSession (

TRDP_APP_SESSION_T * pAppHandle,

TRDP_IP_ADDR_T ownIpAddr,

TRDP_IP_ADDR_T leaderIpAddr,

const TRDP_MARSHALL_CONFIG_T * pMarshall,

const TRDP_PD_CONFIG_T * pPdDefault,

const TRDP_MD_CONFIG_T * pMdDefault,

const TRDP_PROCESS_CONFIG_T * pProcessConfig )
```

Open a session with the TRDP stack.

tlc_openSession returns in pAppHandle a unique handle to be used in further calls to the stack.

Parameters

out	pAppHandle	A handle for further calls to the trdp stack
in	ownlpAddr	Own IP address, can be different for each process in multihoming systems, if zero,
		the default interface / IP will be used.
in	leaderlpAddr	IP address of redundancy leader
in	pMarshall	Pointer to marshalling configuration
in	pPdDefault	Pointer to default PD configuration
in	pMdDefault	Pointer to default MD configuration
in	pProcessConfig	Pointer to process configuration only option parameter is used here to define session
		behavior all other parameters are only used to feed statistics

Return values

TRDP_NO_ERR	no error
TRDP_INIT_ERR	not yet inited
TRDP_PARAM_ERR	parameter error
TRDP_SOCK_ERR	socket error

5.18.2.11 tlc_presetIndexSession()

Preset the index table sizes of a session.

tlc_presetIndexSession allows to preallocate the table sizes in HIGH_PERF_INDEXED mode. If no table sizes are provided, the default sizes are used. In normal mode, this is a no-op. This function should be called during initialisation stage, e.g. right after a session has been opened.

in	appHandle	The handle returned by tlc_openSession
in	pIndexTableSizes	Pointer to a table of sizes to reserve the memory

Return values

TRDP_NO_ERR	no error
TRDP_INIT_ERR	not yet inited
TRDP_PARAM_ERR	parameter error

5.18.2.12 tlc_process()

Work loop of the TRDP handler.

Search the queue for pending PDs and MDs to be sent Search the receive queue for pending PDs and MDs (time out)

Note: If using tlc_process(), do not use tlp_process*() and tlm_process() calls at the same time! Single thread usage -> use tlc_getInterval(), vos_select(), tlc_process() Multiple threads -> thread 1: use tlp_getInterval(), vos_select(), tlp_processReceive() -> thread 2: cyclically call tlp_processSend() -> thread 3: use tlm_getInterval(), vos_select(), tlm_process() for message data

Also see User Manual.

Parameters

in	appHandle	The handle returned by tlc_openSession
in	pRfds	pointer to set of ready descriptors
in,out	pCount	pointer to number of ready descriptors

Return values

TRDP_NO_ERR	no error
TRDP_NOINIT_ERR	handle invalid

5.18.2.13 tlc_reinitSession()

Re-Initialize.

Should be called by the application when a link-down/link-up event has occured during normal operation. We need to re-join the multicast groups...

Parameters

	in <i>appHand</i>	The handle returned by tlc_openSession
--	-------------------	--

Return values

TRDP_NO_ERR	no error
TRDP_NOINIT_ERR	handle invalid
TRDP_PARAM_ERR	handle NULL

5.18.2.14 tlc_setETBTopoCount()

Set new topocount for trainwide communication.

This value is used for validating outgoing and incoming packets only!

Parameters

in	appHandle	the handle returned by tlc_openSession
in <i>etbTopoCnt</i>		New etbTopoCnt value

Return values

TRDP_NO_ERR	no error
TRDP_NOINIT_ERR	handle invalid

5.18.2.15 tlc_setOpTrainTopoCount()

Set new operational train topocount for direction/orientation sensitive communication.

This value is used for validating outgoing and incoming packets only!

Parameters

in	appHandle	The handle returned by tlc_openSession
in	opTrnTopoCnt	New operational topocount value

Return values

TRDP_NO_ERR	no error
TRDP_NOINIT_ERR	handle invalid

5.18.2.16 tlc_terminate()

Un-Initialize.

Clean up and close all sessions. Mainly used for debugging/test runs. No further calls to library allowed

Return values

TRDP_NO_ERR	no error
TRDP_INIT_ERR	no error
TRDP_MEM_ERR	TrafficStore nothing
TRDP_MUTEX_ERR	TrafficStore mutex err

5.18.2.17 tlc_updateSession()

Update a session.

tlc_updateSession signals the end of the set-up phase to the stack. It shall be called after the last publisher and subscriber was added and will create and compute the index tables to be used by the high-performance targets. This function is currently a no-op on standard targets.

Parameters

in	appHandle	The handle returned by tlc_openSession
----	-----------	--

Return values

TRDP_NO_ERR	no error
TRDP_INIT_ERR	not yet inited
TRDP_PARAM_ERR	parameter error

5.18.2.18 trdp_getAccess()

Get mutual access to the session Take all mutexes of that session.

Parameters

in	appHandle	A handle for further calls to the trdp stack
in force If TRUE, access the session even if we cannot get the m		If TRUE, access the session even if we cannot get the mutex.

Return values

TRDP_NO_ERR	
TRDP_INIT_ERR	
TRDP_MUTEX_ERR	

5.18.2.19 trdp_isValidSession()

```
BOOL8 trdp_isValidSession (

TRDP_APP_SESSION_T pSessionHandle )
```

Check if the session handle is valid.

Parameters

in	pSessionHandle	pointer to packet data (dataset)
----	----------------	----------------------------------

Return values

TRUE	is valid
FALSE	is invalid

5.18.2.20 trdp_releaseAccess()

Release access to the session.

Parameters

	in	appHandle	A handle for further calls to the trdp stack
--	----	-----------	--

Return values

real⊷	
ΙP	

Here is the call graph for this function:



5.18.2.21 trdp_sessionQueue()

Get the session queue head pointer.

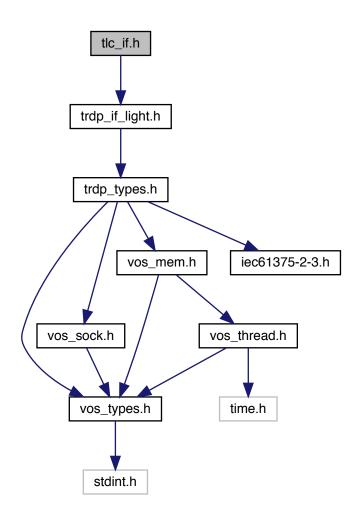
Return values

&sSession

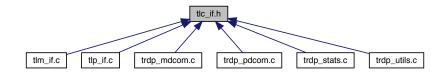
5.19 tlc_if.h File Reference

Typedefs for TRDP communication.

#include "trdp_if_light.h"
Include dependency graph for tlc_if.h:



This graph shows which files directly or indirectly include this file:



Functions

• BOOL8 trdp_isValidSession (TRDP_APP_SESSION_T pSessionHandle)

Check if the session handle is valid.

• TRDP_APP_SESSION_T * trdp_sessionQueue (void)

Get the session queue head pointer.

5.19.1 Detailed Description

Typedefs for TRDP communication.

Note

Project: TCNOpen TRDP prototype stack

Author

Bernd Loehr, NewTec GmbH

Remarks

This Source Code Form is subject to the terms of the Mozilla Public License, v. 2.0. If a copy of the MPL was not distributed with this file, You can obtain one at http://mozilla.org/MPL/2.0/. Copyright Bombardier Transportation Inc. or its subsidiaries and others, 2013-2019. All rights reserved.

5.19.2 Function Documentation

5.19.2.1 trdp_isValidSession()

Check if the session handle is valid.

Parameters

	in	pSessionHandle	pointer to packet data (dataset)
--	----	----------------	----------------------------------

Return values

TRUE	is valid
FALSE	is invalid

5.19.2.2 trdp_sessionQueue()

Get the session queue head pointer.

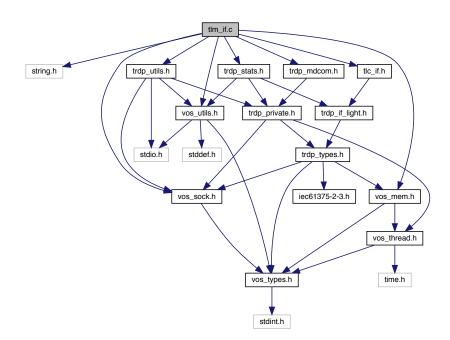
Return values

&sSession

5.20 tlm_if.c File Reference

Functions for Message Data Communication.

```
#include <string.h>
#include "tlc_if.h"
#include "trdp_utils.h"
#include "trdp_mdcom.h"
#include "trdp_stats.h"
#include "vos_sock.h"
#include "vos_mem.h"
#include "vos_utils.h"
Include dependency graph for tlm_if.c:
```



Functions

• EXT_DECL_TRDP_ERR_T_tlm_getInterval (TRDP_APP_SESSION_T appHandle, TRDP_TIME_T *p↔ Interval, TRDP_FDS_T *pFileDesc, INT32 *pNoDesc)

Get the lowest time interval for MDs.

• EXT_DECL TRDP_ERR_T tlm_process (TRDP_APP_SESSION_T appHandle, TRDP_FDS_T *pRfds, IN ← T32 *pCount)

Message Data Work loop of the TRDP handler.

EXT_DECL TRDP_ERR_T tlm_notify (TRDP_APP_SESSION_T appHandle, const void *pUserRef, TRD← P_MD_CALLBACK_T pfCbFunction, UINT32 comld, UINT32 etbTopoCnt, UINT32 opTrnTopoCnt, TRDP_← IP_ADDR_T srclpAddr, TRDP_IP_ADDR_T destlpAddr, TRDP_FLAGS_T pktFlags, const TRDP_SEND_← PARAM_T *pSendParam, const UINT8 *pData, UINT32 dataSize, const TRDP_URI_USER_T sourceURI, const TRDP_URI_USER_T destURI)

Initiate sending MD notification message.

EXT_DECL TRDP_ERR_T tlm_request (TRDP_APP_SESSION_T appHandle, const void *pUserRef, T← RDP_MD_CALLBACK_T pfCbFunction, TRDP_UUID_T *pSessionId, UINT32 comId, UINT32 etbTopo← Cnt, UINT32 opTrnTopoCnt, TRDP_IP_ADDR_T srcIpAddr, TRDP_IP_ADDR_T destIpAddr, TRDP_FLA← GS_T pktFlags, UINT32 numReplies, UINT32 replyTimeout, const TRDP_SEND_PARAM_T *pSendParam, const UINT8 *pData, UINT32 dataSize, const TRDP_URI_USER_T sourceURI, const TRDP_URI_USER_T destURI)

Initiate sending MD request message.

• EXT_DECL TRDP_ERR_T tlm_addListener (TRDP_APP_SESSION_T appHandle, TRDP_LIS_T *pListen← Handle, const void *pUserRef, TRDP_MD_CALLBACK_T pfCbFunction, BOOL8 comldListener, UINT32 comld, UINT32 etbTopoCnt, UINT32 opTrnTopoCnt, TRDP_IP_ADDR_T srclpAddr1, TRDP_IP_ADDR_← T srclpAddr2, TRDP_IP_ADDR_T mcDestlpAddr, TRDP_FLAGS_T pktFlags, const TRDP_URI_USER_T srcURI, const TRDP_URI_USER_T destURI)

Subscribe to MD messages.

• EXT_DECL TRDP_ERR_T tlm_delListener (TRDP_APP_SESSION_T appHandle, TRDP_LIS_T listen → Handle)

Remove Listener.

EXT_DECL TRDP_ERR_T tlm_readdListener (TRDP_APP_SESSION_T appHandle, TRDP_LIS_T listen
 Handle, UINT32 etbTopoCnt, UINT32 opTrnTopoCnt, TRDP_IP_ADDR_T srclpAddr1, TRDP_IP_ADDR_T
 srclpAddr2, TRDP_IP_ADDR_T mcDestlpAddr)

Resubscribe to MD messages.

EXT_DECL_TRDP_ERR_T tlm_reply (TRDP_APP_SESSION_T appHandle, const_TRDP_UUID_T *p
 SessionId, UINT32 comId, UINT16 userStatus, const_TRDP_SEND_PARAM_T *pSendParam, const_UINT8 *pData, UINT32 dataSize)

Send a MD reply message.

Send a MD reply query message.

• EXT_DECL TRDP_ERR_T tlm_confirm (TRDP_APP_SESSION_T appHandle, const TRDP_UUID_T *p↔ SessionId, UINT16 userStatus, const TRDP_SEND_PARAM_T *pSendParam)

Initiate sending MD confirm message.

EXT_DECL TRDP_ERR_T tlm_abortSession (TRDP_APP_SESSION_T appHandle, const TRDP_UUID_T *pSessionId)

Cancel an open session.

5.20.1 Detailed Description

Functions for Message Data Communication.

API implementation of TRDP Light

Note

Project: TCNOpen TRDP prototype stack

Author

Bernd Loehr, NewTec GmbH

Remarks

This Source Code Form is subject to the terms of the Mozilla Public License, v. 2.0. If a copy of the MPL was not distributed with this file, You can obtain one at http://mozilla.org/MPL/2.0/. Copyright Bombardier Transportation Inc. or its subsidiaries and others, 2013-2019. All rights reserved.

5.20.2 Function Documentation

5.20.2.1 tlm_abortSession()

Cancel an open session.

Abort an open session; any pending messages will be dropped

Parameters

i	ln	appHandle	the handle returned by tlc_openSession
i	n	p⇔	Session ID returned by request
		SessionId	

Return values

TRDP_NO_ERR	no error
TRDP_NOSESSION_ERR	no such session
TRDP_NOINIT_ERR	handle invalid

5.20.2.2 tlm_addListener()

```
EXT_DECL TRDP_ERR_T tlm_addListener (

TRDP_APP_SESSION_T appHandle,

TRDP_LIS_T * pListenHandle,

const void * pUserRef,

TRDP_MD_CALLBACK_T pfCbFunction,

BOOL8 comIdListener,

UINT32 comId,

UINT32 etbTopoCnt,

UINT32 opTrnTopoCnt,

TRDP_IP_ADDR_T srcIpAddr1,
```

```
TRDP_IP_ADDR_T srcIpAddr2,
TRDP_IP_ADDR_T mcDestIpAddr,
TRDP_FLAGS_T pktFlags,
const TRDP_URI_USER_T srcURI,
const TRDP_URI_USER_T destURI)
```

Subscribe to MD messages.

Add a listener to TRDP to get notified when messages are received

Parameters

	•.•		
in	appHandle	the handle returned by tlc_openSession	
out	pListenHandle	Handle for this listener returned	
in	pUserRef	user supplied value returned with received message	
in	pfCbFunction	Pointer to listener specific callback function, NULL to use default function	
in	comldListener	set TRUE if comld shall be observed	
in	comld	comld to be observed	
in	etbTopoCnt	ETB topocount to use, 0 if consist local communication	
in	opTrnTopoCnt	operational topocount, != 0 for orientation/direction sensitive communication	
in	srclpAddr1	Source IP address, lower address in case of address range, set to 0 if not used	
in	srclpAddr2	upper address in case of address range, set to 0 if not used	
in	mcDestlpAddr	multicast group to listen on	
in	pktFlags	OPTION: TRDP_FLAGS_DEFAULT, TRDP_FLAGS_MARSHALL	
in	srcURI	only functional group of source URI, set to NULL if not used	
in	destURI	only functional group of destination URI, set to NULL if not used	

Return values

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	parameter error
TRDP_MEM_ERR	out of memory
TRDP_NOINIT_ERR	handle invalid

5.20.2.3 tlm_confirm()

Initiate sending MD confirm message.

Send a MD confirmation message User reference, source and destination IP addresses as well as topo counts and packet flags are taken from the session

Parameters

in	appHandle	the handle returned by tlc_openSession	
in	pSessionId	Session ID returned by request	
in	userStatus	Info for requester about application errors	
in	pSendParam	Pointer to send parameters, NULL to use default send parameters	

Generated by Doxygen

Return values

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	parameter error
TRDP_MEM_ERR	out of memory
TRDP_NOSESSION_ERR	no such session
TRDP_NOINIT_ERR	handle invalid

5.20.2.4 tlm_delListener()

Remove Listener.

Parameters

in	appHandle	the handle returned by tlc_openSession
out	listenHandle	Handle for this listener

Return values

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	parameter error
TRDP_NOINIT_ERR	handle invalid

5.20.2.5 tlm_getInterval()

Get the lowest time interval for MDs.

Return the maximum time interval suitable for 'select()' so that we can report time outs to the higher layer.

Parameters

in	appHandle	The handle returned by tlc_openSession
out	pInterval pointer to needed interval	
in, out pFileDesc pointer to file descriptor set		
out	pNoDesc	pointer to put no of highest used descriptors (for select())

Return values

TRDP_NO_ERR	no error
TRDP_NOINIT_ERR	handle invalid

5.20.2.6 tlm_notify()

```
EXT_DECL TRDP_ERR_T tlm_notify (

TRDP_APP_SESSION_T appHandle,
const void * pUserRef,
TRDP_MD_CALLBACK_T pfCbFunction,
UINT32 comId,
UINT32 etbTopoCnt,
UINT32 opTrnTopoCnt,
TRDP_IP_ADDR_T srcIpAddr,
TRDP_IP_ADDR_T destIpAddr,
TRDP_FLAGS_T pktFlags,
const TRDP_SEND_PARAM_T * pSendParam,
const UINT8 * pData,
UINT32 dataSize,
const TRDP_URI_USER_T sourceURI,
const TRDP_URI_USER_T destURI)
```

Initiate sending MD notification message.

Send a MD notification message

Parameters

in	appHandle	the handle returned by tlc_openSession	
in	pUserRef	user supplied value returned with reply	
in	pfCbFunction	Pointer to listener specific callback function, NULL to use default function	
in	comld	comld of packet to be sent	
in	etbTopoCnt	ETB topocount to use, 0 if consist local communication	
in	opTrnTopoCnt	operational topocount, != 0 for orientation/direction sensitive communication	
in	srclpAddr	own IP address, 0 - srcIP will be set by the stack	
in	destlpAddr	where to send the packet to	
in	pktFlags	OPTION: TRDP_FLAGS_DEFAULT, TRDP_FLAGS_NONE, TRDP_FLAGS_MARSHALL,	
		TRDP_FLAGS_CALLBACK	
in	pSendParam	optional pointer to send parameter, NULL - default parameters are used	
in	pData	pointer to packet data / dataset	
in	dataSize	size of packet data	
in	sourceURI	only functional group of source URI	
in	destURI	only functional group of destination URI	

Return values

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	parameter error
TRDP_MEM_ERR	out of memory

Return values

TRDP_NOINIT_ERR	handle invalid
-----------------	----------------

5.20.2.7 tlm_process()

Message Data Work loop of the TRDP handler.

Search the queue for pending MDs to be sent Search the receive queue for pending MDs (replies, time outs) and incoming requests

Parameters

	in	appHandle	The handle returned by tlc_openSession
	in	pRfds	pointer to set of ready descriptors
Ī	in,out	pCount	pointer to number of ready descriptors

Return values

TRDP_NO_ERR	no error
TRDP_NOINIT_ERR	handle invalid

5.20.2.8 tlm_readdListener()

```
EXT_DECL TRDP_ERR_T tlm_readdListener (

TRDP_APP_SESSION_T appHandle,

TRDP_LIS_T listenHandle,

UINT32 etbTopoCnt,

UINT32 opTrnTopoCnt,

TRDP_IP_ADDR_T srcIpAddr1,

TRDP_IP_ADDR_T srcIpAddr2,

TRDP_IP_ADDR_T mcDestIpAddr )
```

Resubscribe to MD messages.

Readd a listener after topoCount changes to get notified when messages are received

Parameters

in	appHandle	the handle returned by tlc_openSession	
out	listenHandle	e Handle for this listener	
in	etbTopoCnt	t ETB topocount to use, 0 if consist local communication	

Parameters

in	opTrnTopoCnt	operational topocount, != 0 for orientation/direction sensitive communication
in	srclpAddr1 Source IP address, lower address in case of address range, set to 0 if not used	
in	srclpAddr2	upper address in case of address range, set to 0 if not used
in	mcDestlpAddr	multicast group to listen on

Return values

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	parameter error
TRDP_MEM_ERR	out of memory
TRDP_NOINIT_ERR	handle invalid

5.20.2.9 tlm_reply()

```
EXT_DECL TRDP_ERR_T tlm_reply (

TRDP_APP_SESSION_T appHandle,

const TRDP_UUID_T * pSessionId,

UINT32 comId,

UINT16 userStatus,

const TRDP_SEND_PARAM_T * pSendParam,

const UINT8 * pData,

UINT32 dataSize )
```

Send a MD reply message.

Send a MD reply message after receiving an request User reference, source and destination IP addresses as well as topo counts and packet flags are taken from the session

Parameters

in	appHandle	the handle returned by tlc_openSession	
in	pSessionId Session ID returned by indication		
in	comld	comld of packet to be sent	
in	userStatus	serStatus Info for requester about application errors	
in	pSendParam Pointer to send parameters, NULL to use default send parameters		
in	pData	pointer to packet data / dataset	
in	dataSize	size of packet data	

Return values

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	parameter error
TRDP_MEM_ERR	Out of memory
TRDP_NO_SESSION_ERR	no such session
TRDP_NOINIT_ERR	handle invalid

5.20.2.10 tlm_replyQuery()

```
EXT_DECL TRDP_ERR_T tlm_replyQuery (

TRDP_APP_SESSION_T appHandle,

const TRDP_UUID_T * pSessionId,

UINT32 comId,

UINT16 userStatus,

UINT32 confirmTimeout,

const TRDP_SEND_PARAM_T * pSendParam,

const UINT3 * pData,

UINT32 dataSize )
```

Send a MD reply query message.

Send a MD reply query message after receiving a request and ask for confirmation. User reference, source and destination IP addresses as well as topo counts and packet flags are taken from the session

Parameters

in	appHandle	the handle returned by tlc_openSession	
in	pSessionId	Session ID returned by indication	
in	comld	comld of packet to be sent	
in	userStatus	Info for requester about application errors	
in	confirmTimeout	timeout for confirmation	
in	pSendParam	Pointer to send parameters, NULL to use default send parameters	
in	pData	pointer to packet data / dataset	
in	dataSize	size of packet data	

Return values

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	parameter error
TRDP_MEM_ERR	out of memory
TRDP_NO_SESSION_ERR	no such session
TRDP_NOINIT_ERR	handle invalid

5.20.2.11 tlm_request()

```
TRDP_IP_ADDR_T destIpAddr,

TRDP_FLAGS_T pktFlags,

UINT32 numReplies,

UINT32 replyTimeout,

const TRDP_SEND_PARAM_T * pSendParam,

const UINT8 * pData,

UINT32 dataSize,

const TRDP_URI_USER_T sourceURI,

const TRDP_URI_USER_T destURI)
```

Initiate sending MD request message.

Send a MD request message

Parameters

in	appHandle	the handle returned by tlc_openSession	
in	pUserRef	user supplied value returned with reply	
in	pfCbFunction	Pointer to listener specific callback function, NULL to use default function	
out	pSessionId	return session ID	
in	comld	comld of packet to be sent	
in	etbTopoCnt	ETB topocount to use, 0 if consist local communication	
in	opTrnTopoCnt	operational topocount, != 0 for orientation/direction sensitive communication	
in	srclpAddr	own IP address, 0 - srcIP will be set by the stack	
in	destlpAddr	where to send the packet to	
in	pktFlags	OPTION: TRDP_FLAGS_DEFAULT, TRDP_FLAGS_NONE, TRDP_FLAGS_MARSHALL	
in	numReplies	numReplies number of expected replies, 0 if unknown	
in	replyTimeout timeout for reply		
in	pSendParam Pointer to send parameters, NULL to use default send parameters		
in	pData	pData pointer to packet data / dataset	
in	dataSize	size of packet data	
in	sourceURI	eURI only functional group of source URI	
in	destURI	only functional group of destination URI	

Return values

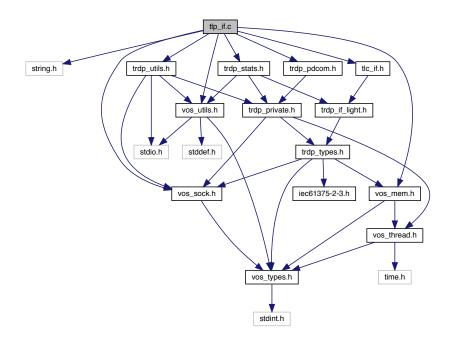
TRDP_NO_ERR	no error
TRDP_PARAM_ERR	parameter error
TRDP_MEM_ERR	out of memory
TRDP_NOINIT_ERR	handle invalid

5.21 tlp_if.c File Reference

Functions for Process Data Communication.

```
#include <string.h>
#include "tlc_if.h"
#include "trdp_utils.h"
#include "trdp_pdcom.h"
#include "trdp_stats.h"
```

```
#include "vos_sock.h"
#include "vos_mem.h"
#include "vos_utils.h"
Include dependency graph for tlp_if.c:
```



Functions

EXT_DECL TRDP_ERR_T tlp_getInterval (TRDP_APP_SESSION_T appHandle, TRDP_TIME_T *pInterval, TRDP FDS T *pFileDesc, INT32 *pNoDesc)

Get the lowest time interval for PDs.

• EXT_DECL TRDP_ERR_T tlp_processReceive (TRDP_APP_SESSION_T appHandle, TRDP_FDS_T *p↔ Rfds, INT32 *pCount)

Work loop of the TRDP handler.

- EXT_DECL TRDP_ERR_T tlp_processSend (TRDP_APP_SESSION_T appHandle) Work loop of the TRDP handler.
- EXT_DECL TRDP_ERR_T tlp_setRedundant (TRDP_APP_SESSION_T appHandle, UINT32 redId, BOOL8 leader)

Do not send non-redundant PDs when we are follower.

• EXT_DECL TRDP_ERR_T tlp_getRedundant (TRDP_APP_SESSION_T appHandle, UINT32 redId, BOOL8 *pLeader)

Get status of redundant Comlds.

• EXT_DECL TRDP_ERR_T tlp_publish (TRDP_APP_SESSION_T appHandle, TRDP_PUB_T *pPubHandle, const void *pUserRef, TRDP_PD_CALLBACK_T pfCbFunction, UINT32 serviceld, UINT32 comld, UINT32 etbTopoCnt, UINT32 opTrnTopoCnt, TRDP_IP_ADDR_T srclpAddr, TRDP_IP_ADDR_T destlpAddr, UINT32 interval, UINT32 redId, TRDP_FLAGS_T pktFlags, const TRDP_SEND_PARAM_T *pSendParam, const U← INT8 *pData, UINT32 dataSize)

Prepare for sending PD messages.

• EXT_DECL TRDP_ERR_T tlp_republish (TRDP_APP_SESSION_T appHandle, TRDP_PUB_T pubHandle, UINT32 etbTopoCnt, UINT32 opTrnTopoCnt, TRDP_IP_ADDR_T srclpAddr, TRDP_IP_ADDR_T destlpAddr)

Prepare for sending PD messages.

• EXT_DECL TRDP_ERR_T tlp_unpublish (TRDP_APP_SESSION_T appHandle, TRDP_PUB_T pubHandle) Stop sending PD messages.

• EXT_DECL TRDP_ERR_T tlp_put (TRDP_APP_SESSION_T appHandle, TRDP_PUB_T pubHandle, const UINT8 *pData, UINT32 dataSize)

Update the process data to send.

• EXT_DECL TRDP_ERR_T tlp_putImmediate (TRDP_APP_SESSION_T appHandle, TRDP_PUB_T pub↔ Handle, const UINT8 *pData, UINT32 dataSize, VOS_TIMEVAL_T *pTxTime)

Update and send process data.

• EXT_DECL TRDP_ERR_T tlp_request (TRDP_APP_SESSION_T appHandle, TRDP_SUB_T subHandle, UINT32 serviceld, UINT32 comId, UINT32 etbTopoCnt, UINT32 opTrnTopoCnt, TRDP_IP_ADDR_T src← lpAddr, TRDP_IP_ADDR_T destlpAddr, UINT32 redld, TRDP_FLAGS_T pktFlags, const TRDP_SEND_P← ARAM_T *pSendParam, const UINT8 *pData, UINT32 dataSize, UINT32 replyComId, TRDP_IP_ADDR_T replyIpAddr)

Initiate sending PD messages (PULL).

EXT_DECL TRDP_ERR_T tlp_subscribe (TRDP_APP_SESSION_T appHandle, TRDP_SUB_T *pSub← Handle, const void *pUserRef, TRDP_PD_CALLBACK_T pfCbFunction, UINT32 serviceld, UINT32 comld, UINT32 etbTopoCnt, UINT32 opTrnTopoCnt, TRDP_IP_ADDR_T srclpAddr1, TRDP_IP_ADDR_T srclp← Addr2, TRDP_IP_ADDR_T destlpAddr, TRDP_FLAGS_T pktFlags, const TRDP_COM_PARAM_T *pRec← Params, UINT32 timeout, TRDP_TO_BEHAVIOR_T toBehavior)

Prepare for receiving PD messages.

• EXT_DECL TRDP_ERR_T tlp_unsubscribe (TRDP_APP_SESSION_T appHandle, TRDP_SUB_T sub↔ Handle)

Stop receiving PD messages.

EXT_DECL_TRDP_ERR_T tlp_resubscribe (TRDP_APP_SESSION_T appHandle, TRDP_SUB_T sub
 Handle, UINT32 etbTopoCnt, UINT32 opTrnTopoCnt, TRDP_IP_ADDR_T srclpAddr1, TRDP_IP_ADDR_T
 srclpAddr2, TRDP_IP_ADDR_T destIpAddr)

Reprepare for receiving PD messages.

Get the last valid PD message.

5.21.1 Detailed Description

Functions for Process Data Communication.

API implementation of TRDP Light

Note

Project: TCNOpen TRDP prototype stack

Author

Bernd Loehr, NewTec GmbH

Remarks

This Source Code Form is subject to the terms of the Mozilla Public License, v. 2.0. If a copy of the MPL was not distributed with this file, You can obtain one at http://mozilla.org/MPL/2.0/. Copyright Bombardier Transportation Inc. or its subsidiaries and others, 2013-2019. All rights reserved.

5.21.2 Function Documentation

5.21.2.1 tlp_get()

```
EXT_DECL TRDP_ERR_T tlp_get (
          TRDP_APP_SESSION_T appHandle,
          TRDP_SUB_T subHandle,
          TRDP_PD_INFO_T * pPdInfo,
          UINT8 * pData,
          UINT32 * pDataSize )
```

Get the last valid PD message.

This allows polling of PDs instead of event driven handling by callbacks

Parameters

in	appHandle	the handle returned by tlc_openSession
in	subHandle	the handle returned by subscription
in,out	pPdInfo	pointer to application's info buffer
in,out	pData	pointer to application's data buffer
in,out	pDataSize	in: size of buffer, out: size of data

Return values

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	parameter error
TRDP_SUB_ERR	not subscribed
TRDP_TIMEOUT_ERR	packet timed out
TRDP_NOINIT_ERR	handle invalid
TRDP_COMID_ERR	ComID not found when marshalling

5.21.2.2 tlp_getInterval()

Get the lowest time interval for PDs.

Return the maximum time interval suitable for 'select()' so that we can send due PD packets in time. If the PD send queue is empty, return zero time

Parameters

in	appHandle	The handle returned by tlc_openSession	
out	pInterval	pointer to needed interval	
in,out	pFileDesc	pointer to file descriptor set	
out	pNoDesc	pointer to put no of highest used descriptors (for select())	

Return values

TRDP_NO_ERR	no error
TRDP_NOINIT_ERR	handle invalid

5.21.2.3 tlp_getRedundant()

Get status of redundant Comlds.

Only the status of the first found redundancy group entry will be returned!

Parameters

in	appHandle	the handle returned by tlc_openSession
in	redId	will be returned for all ComID's with the given redId
in, out	pLeader	TRUE if we're sending this redundancy group (leader)

Return values

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	redld invalid or not existing
TRDP_NOINIT_ERR	handle invalid

5.21.2.4 tlp_processReceive()

Work loop of the TRDP handler.

Check the sockets for incoming PD telegrams. Search the receive queue for pending PDs (time out) and report them, either by informing the higher layer via the callback mechanism or just by marking the subscriber as timed-out

Parameters

in	appHandle	The handle returned by tlc_openSession
in	pRfds	pointer to set of ready descriptors
in,out	pCount	pointer to number of ready descriptors

Return values

TRDP_NO_ERR	no error
TRDP_NOINIT_ERR	handle invalid

5.21.2.5 tlp_processSend()

Work loop of the TRDP handler.

Search the queue for pending PDs to be sent

Parameters

in	appHandle	The handle returned by tlc_openSession
----	-----------	--

Return values

TRDP_NO_ERR	no error
TRDP_NOINIT_ERR	handle invalid

5.21.2.6 tlp_publish()

```
EXT_DECL TRDP_ERR_T tlp_publish (

TRDP_APP_SESSION_T appHandle,

TRDP_PUB_T * pPubHandle,

const void * pUserRef,

TRDP_PD_CALLBACK_T pfCbFunction,

UINT32 serviceId,

UINT32 comId,

UINT32 etbTopoCnt,

UINT32 opTrnTopoCnt,

TRDP_IP_ADDR_T srcIpAddr,

TRDP_IP_ADDR_T destIpAddr,

UINT32 interval,

UINT32 redId,

TRDP_FLAGS_T pktFlags,

const TRDP_SEND_PARAM_T * pSendParam,
```

```
const UINT8 * pData,
UINT32 dataSize )
```

Prepare for sending PD messages.

Queue a PD message, it will be send when tlc_publish has been called

Parameters

aramet	CIS	
in	appHandle	the handle returned by tlc_openSession
out	pPubHandle	returned handle for related re/unpublish
in	pUserRef	user supplied value returned within the info structure of callback function
in	pfCbFunction	Pointer to pre-send callback function, NULL if not used
in	serviceld	optional serviceld this telegram belongs to (default = 0)
in	comld	comld of packet to send
in	etbTopoCnt	ETB topocount to use, 0 if consist local communication
in	opTrnTopoCnt	operational topocount, != 0 for orientation/direction sensitive communication
in	srclpAddr	own IP address, 0 - srcIP will be set by the stack
in	destlpAddr	where to send the packet to
in	interval	frequency of PD packet (>= 10ms) in usec
in	redId	0 - Non-redundant, > 0 valid redundancy group
in	pktFlags	OPTION: TRDP_FLAGS_DEFAULT, TRDP_FLAGS_NONE, TRDP_FLAGS_MARSHALL, TRDP_FLAGS_CALLBACK
in	pSendParam	optional pointer to send parameter, NULL - default parameters are used
in	pData	optional pointer to data packet / dataset, NULL if sending starts later with tlp_put()
in	dataSize	size of data packet >= 0 and <= TRDP_MAX_PD_DATA_SIZE

Return values

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	parameter error
TRDP_MEM_ERR	could not insert (out of memory)
TRDP_NOINIT_ERR	handle invalid

5.21.2.7 tlp_put()

```
EXT_DECL TRDP_ERR_T tlp_put (
          TRDP_APP_SESSION_T appHandle,
          TRDP_PUB_T pubHandle,
          const UINT8 * pData,
          UINT32 dataSize )
```

Update the process data to send.

Update previously published data. The new telegram will be sent earliest when tlc_process is called.

Parameters

in	appHandle	the handle returned by tlc_openSession	
in	pubHandle	the handle returned by publish	
in,out	pData	pointer to application's data buffer	Generated by I
in,out	dataSize	size of data	delicitied by E

Return values

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	parameter error on uninitialized parameter or changed dataSize compared to
	published one
TRDP_NOPUB_ERR	not published
TRDP_NOINIT_ERR	handle invalid
TRDP_COMID_ERR	ComID not found when marshalling

5.21.2.8 tlp_putlmmediate()

```
EXT_DECL TRDP_ERR_T tlp_putImmediate (
          TRDP_APP_SESSION_T appHandle,
          TRDP_PUB_T pubHandle,
          const UINT8 * pData,
          UINT32 dataSize,
          VOS_TIMEVAL_T * pTxTime )
```

Update and send process data.

Update previously published data. The new telegram will be sent immediatly or at txTime, if txTime != 0 and TSN == 1 Should be used if application (or higher layer, e.g. ara::com and acyclic events) needs full control over process data schedule.

Note: For TSN this function is not protected by any mutexes and should not be called while adding or removing any publishers, subscribers or even sessions! Also: Marshalling is not supported!

Parameters

in	appHandle	the handle returned by tlc_openSession
in	pubHandle	the handle returned by publish
in,out	pData	pointer to application's data buffer
in,out	dataSize	size of data
in	pTxTime	when to send (absolute time), optional for TSN only

Return values

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	parameter error on uninitialized parameter or changed dataSize compared to
	published one
TRDP_NOPUB_ERR	not published
TRDP_NOINIT_ERR	handle invalid

5.21.2.9 tlp_republish()

```
TRDP_PUB_T pubHandle,
UINT32 etbTopoCnt,
UINT32 opTrnTopoCnt,
TRDP_IP_ADDR_T srcIpAddr,
TRDP_IP_ADDR_T destIpAddr)
```

Prepare for sending PD messages.

Reinitialize and queue a PD message, it will be send when tlc_publish has been called

Parameters

in	appHandle	the handle returned by tlc_openSession
in	pubHandle	handle for related unpublish
in	etbTopoCnt	ETB topocount to use, 0 if consist local communication
in	opTrnTopoCnt	operational topocount, != 0 for orientation/direction sensitive communication
in	srclpAddr	own IP address, 0 - srcIP will be set by the stack
in	destlpAddr	where to send the packet to

Return values

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	parameter error
TRDP_MEM_ERR	could not insert (out of memory)
TRDP_NOINIT_ERR	handle invalid

5.21.2.10 tlp_request()

```
EXT_DECL TRDP_ERR_T tlp_request (
             TRDP_APP_SESSION_T appHandle,
             TRDP_SUB_T subHandle,
             UINT32 serviceId,
             UINT32 comId,
             UINT32 etbTopoCnt,
             UINT32 opTrnTopoCnt,
             TRDP_IP_ADDR_T srcIpAddr,
             TRDP_IP_ADDR_T destIpAddr,
             UINT32 redId,
             TRDP_FLAGS_T pktFlags,
             const TRDP_SEND_PARAM_T * pSendParam,
             const UINT8 * pData,
             UINT32 dataSize,
             UINT32 replyComId,
             TRDP_IP_ADDR_T replyIpAddr )
```

Initiate sending PD messages (PULL).

Send a PD request message

Parameters

in	appHandle	the handle returned by tlc_openSession
----	-----------	--

Parameters

in	subHandle	handle from related subscribe
in	serviceld	optional serviceld this telegram belongs to (default = 0)
in	comld	comld of packet to be sent
in	etbTopoCnt	ETB topocount to use, 0 if consist local communication
in	opTrnTopoCnt	operational topocount, != 0 for orientation/direction sensitive communication
in	srclpAddr	own IP address, 0 - srcIP will be set by the stack
in	destlpAddr	where to send the packet to
in	redId	0 - Non-redundant, > 0 valid redundancy group
in	pktFlags	OPTION: TRDP_FLAGS_DEFAULT, TRDP_FLAGS_NONE, TRDP_FLAGS_MARSHALL, TRDP_FLAGS_CALLBACK
in	pSendParam	optional pointer to send parameter, NULL - default parameters are used
in	pData	pointer to packet data / dataset
in	dataSize	size of packet data
in	replyComId	comld of reply (default comID of subscription)
in	replylpAddr	IP for reply

Return values

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	parameter error
TRDP_MEM_ERR	could not insert (out of memory)
TRDP_NOINIT_ERR	handle invalid
TRDP_NOSUB_ERR	no matching subscription found

5.21.2.11 tlp_resubscribe()

Reprepare for receiving PD messages.

Resubscribe to a specific PD ComID and source IP

Parameters

in	appHandle	the handle returned by tlc_openSession
in	subHandle	handle for this subscription
in	etbTopoCnt	ETB topocount to use, 0 if consist local communication
in	opTrnTopoCnt	operational topocount, != 0 for orientation/direction sensitive communication
in	srclpAddr1	Source IP address, lower address in case of address range, set to 0 if not used
in	srclpAddr2	upper address in case of address range, set to 0 if not used
in	destlpAddr	IP address to join

Generated by Doxygen

Return values

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	parameter error
TRDP_MEM_ERR	could not reserve memory (out of memory)
TRDP_NOINIT_ERR	handle invalid
TRDP_SOCK_ERR	Resource (socket) not available, subscription canceled

5.21.2.12 tlp_setRedundant()

Do not send non-redundant PDs when we are follower.

Parameters

in	appHandle	the handle returned by tlc_openSession will be set for all ComID's with the given redId, 0 to change for all redId	
in	redId		
in	leader	TRUE if we send	

Return values

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	parameter error / redld not existing
TRDP_NOINIT_ERR	handle invalid

5.21.2.13 tlp_subscribe()

```
EXT_DECL TRDP_ERR_T tlp_subscribe (
             TRDP_APP_SESSION_T appHandle,
             TRDP_SUB_T * pSubHandle,
             const void * pUserRef,
             TRDP_PD_CALLBACK_T pfCbFunction,
             UINT32 serviceId,
             UINT32 comId,
             UINT32 etbTopoCnt,
             UINT32 opTrnTopoCnt,
             TRDP_IP_ADDR_T srcIpAddr1,
             TRDP_IP_ADDR_T srcIpAddr2,
             TRDP_IP_ADDR_T destIpAddr,
             TRDP_FLAGS_T pktFlags,
             const TRDP_COM_PARAM_T * pRecParams,
             UINT32 timeout,
             TRDP_TO_BEHAVIOR_T toBehavior )
```

Prepare for receiving PD messages.

Subscribe to a specific PD ComID and source IP.

Parameters

in	appHandle	the handle returned by tlc_openSession	
out	pSubHandle	return a handle for this subscription	
in	pUserRef user supplied value returned within the info structure		
in	pfCbFunction	Pointer to subscriber specific callback function, NULL to use default function	
in	serviceld	optional serviceld this telegram belongs to (default = 0)	
in	comld	comld of packet to receive	
in	etbTopoCnt	ETB topocount to use, 0 if consist local communication	
in	opTrnTopoCnt	operational topocount, != 0 for orientation/direction sensitive communication	
in	srclpAddr1	Source IP address, lower address in case of address range, set to 0 if not used	
in	srclpAddr2	upper address in case of address range, set to 0 if not used	
in	destlpAddr	IP address to join	
in	pktFlags	OPTION: TRDP_FLAGS_DEFAULT, TRDP_FLAGS_NONE, TRDP_FLAGS_MARSHALL, TRDP_FLAGS_CALLBACK	
in	pRecParams	optional pointer to send parameter, NULL - default parameters are used	
in	timeout	timeout (>= 10ms) in usec	
in	toBehavior	timeout behavior	

Return values

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	parameter error
TRDP_MEM_ERR	could not reserve memory (out of memory)
TRDP_NOINIT_ERR	handle invalid

5.21.2.14 tlp_unpublish()

Stop sending PD messages.

Parameters

in	appHandle	the handle returned by tlc_openSession
in	pubHandle	the handle returned by prepare

Return values

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	parameter error
TRDP_NOPUB_ERR	not published

Return values

TRDP_NOINIT_ERR	handle invalid
-----------------	----------------

5.21.2.15 tlp_unsubscribe()

Stop receiving PD messages.

Unsubscribe to a specific PD ComID

Parameters

in	appHandle	the handle returned by tlc_openSession
in	subHandle	the handle for this subscription

Return values

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	parameter error
TRDP_NOSUB_ERR	not subscribed
TRDP_NOINIT_ERR	handle invalid

5.22 trdp_dllmain.c File Reference

Windows DLL main function.

5.22.1 Detailed Description

Windows DLL main function.

Note

Project: TCNOpen TRDP prototype stack

Author

Armin-H. Weiss, Bombardier

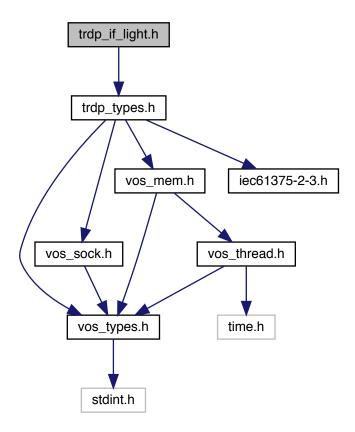
Remarks

This Source Code Form is subject to the terms of the Mozilla Public License, v. 2.0. If a copy of the MPL was not distributed with this file, You can obtain one at http://mozilla.org/MPL/2.0/. Copyright Bombardier Transportation Inc. or its subsidiaries and others, 2013. All rights reserved.

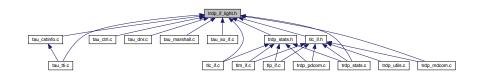
5.23 trdp_if_light.h File Reference

TRDP Light interface functions (API)

#include "trdp_types.h"
Include dependency graph for trdp_if_light.h:



This graph shows which files directly or indirectly include this file:



Functions

• EXT_DECL TRDP_ERR_T tlc_init (const TRDP_PRINT_DBG_T pPrintDebugString, void *pRefCon, const TRDP_MEM_CONFIG_T *pMemConfig)

Support for message data can only be excluded during compile time!

• EXT_DECL TRDP_ERR_T tlc_openSession (TRDP_APP_SESSION_T *pAppHandle, TRDP_IP_ADDR ← __T ownIpAddr, TRDP_IP_ADDR_T leaderIpAddr, const TRDP_MARSHALL_CONFIG_T *pMarshall, const TRDP_PD_CONFIG_T *pPdDefault, const TRDP_MD_CONFIG_T *pMdDefault, const TRDP_PROCES ← S CONFIG_T *pProcessConfig)

Open a session with the TRDP stack.

• EXT_DECL TRDP_ERR_T tlc_reinitSession (TRDP_APP_SESSION_T appHandle)

Re-Initialize.

 EXT_DECL TRDP_ERR_T tlc_configSession (TRDP_APP_SESSION_T appHandle, const TRDP_MAR← SHALL_CONFIG_T *pMarshall, const TRDP_PD_CONFIG_T *pPdDefault, const TRDP_MD_CONFIG_T *pMdDefault, const TRDP_PROCESS_CONFIG_T *pProcessConfig)

(Re-)configure a session.

• EXT DECL TRDP ERR T tlc updateSession (TRDP APP SESSION T appHandle)

Update a session.

EXT_DECL TRDP_ERR_T tlc_presetIndexSession (TRDP_APP_SESSION_T appHandle, TRDP_IDX_T

 ABLE T *pIndexTableSizes)

Preset the index table sizes of a session.

• EXT_DECL TRDP_ERR_T tlc_closeSession (TRDP_APP_SESSION_T appHandle)

Close a session.

EXT DECL TRDP ERR T tlc terminate (void)

Un-Initialize.

EXT_DECL TRDP_ERR_T tlc_setETBTopoCount (TRDP_APP_SESSION_T appHandle, UINT32 etbTopo

 Cnt)

Set new topocount for trainwide communication.

• EXT DECL UINT32 tlc getETBTopoCount (TRDP APP SESSION T appHandle)

Set new topocount for trainwide communication.

Set new operational train topocount for direction/orientation sensitive communication.

EXT_DECL UINT32 tlc_getOpTrainTopoCount (TRDP_APP_SESSION_T appHandle)

Set new operational train topocount for direction/orientation sensitive communication.

• EXT_DECL TRDP_ERR_T tlc_getInterval (TRDP_APP_SESSION_T appHandle, TRDP_TIME_T *pInterval, TRDP_FDS_T *pFileDesc, INT32 *pNoDesc)

Get the lowest time interval for PDs.

EXT_DECL TRDP_ERR_T tlc_process (TRDP_APP_SESSION_T appHandle, TRDP_FDS_T *pRfds, INT32 *pCount)

Work loop of the TRDP handler.

• EXT DECL TRDP IP ADDR T tlc getOwnlpAddress (TRDP APP SESSION T appHandle)

Get the interface address.

• EXT_DECL TRDP_ERR_T tlp_getInterval (TRDP_APP_SESSION_T appHandle, TRDP_TIME_T *pInterval, TRDP_FDS_T *pFileDesc, INT32 *pNoDesc)

Get the lowest time interval for PDs.

• EXT_DECL TRDP_ERR_T tlp_processSend (TRDP_APP_SESSION_T appHandle)

Work loop of the TRDP handler.

• EXT_DECL TRDP_ERR_T tlp_processReceive (TRDP_APP_SESSION_T appHandle, TRDP_FDS_T *p↔ Rfds, INT32 *pCount)

Work loop of the TRDP handler.

• EXT_DECL TRDP_ERR_T tlp_publish (TRDP_APP_SESSION_T appHandle, TRDP_PUB_T *pPubHandle, const void *pUserRef, TRDP_PD_CALLBACK_T pfCbFunction, UINT32 serviceld, UINT32 comld, UINT32 etbTopoCnt, UINT32 opTrnTopoCnt, TRDP_IP_ADDR_T srclpAddr, TRDP_IP_ADDR_T destlpAddr, UINT32 interval, UINT32 redId, TRDP_FLAGS_T pktFlags, const TRDP_SEND_PARAM_T *pSendParam, const U← INT8 *pData, UINT32 dataSize)

Prepare for sending PD messages.

- EXT_DECL TRDP_ERR_T tlp_republish (TRDP_APP_SESSION_T appHandle, TRDP_PUB_T pubHandle, UINT32 etbTopoCnt, UINT32 opTrnTopoCnt, TRDP_IP_ADDR_T srclpAddr, TRDP_IP_ADDR_T destlpAddr)

 Prepare for sending PD messages.
- EXT_DECL TRDP_ERR_T tlp_unpublish (TRDP_APP_SESSION_T appHandle, TRDP_PUB_T pubHandle) Stop sending PD messages.
- EXT_DECL TRDP_ERR_T tlp_put (TRDP_APP_SESSION_T appHandle, TRDP_PUB_T pubHandle, const UINT8 *pData, UINT32 dataSize)

Update the process data to send.

EXT_DECL TRDP_ERR_T tlp_putImmediate (TRDP_APP_SESSION_T appHandle, TRDP_PUB_T pub
 Handle, const UINT8 *pData, UINT32 dataSize, VOS TIMEVAL T *pTxTime)

Update and send process data.

EXT_DECL TRDP_ERR_T tlp_setRedundant (TRDP_APP_SESSION_T appHandle, UINT32 redId, BOOL8 leader)

Do not send non-redundant PDs when we are follower.

EXT_DECL TRDP_ERR_T tlp_getRedundant (TRDP_APP_SESSION_T appHandle, UINT32 redId, BOOL8 *pLeader)

Get status of redundant Comlds.

Initiate sending PD messages (PULL).

• EXT_DECL TRDP_ERR_T tlp_subscribe (TRDP_APP_SESSION_T appHandle, TRDP_SUB_T *pSub ← Handle, const void *pUserRef, TRDP_PD_CALLBACK_T pfCbFunction, UINT32 serviceld, UINT32 comld, UINT32 etbTopoCnt, UINT32 opTrnTopoCnt, TRDP_IP_ADDR_T srclpAddr1, TRDP_IP_ADDR_T srclp ← Addr2, TRDP_IP_ADDR_T destlpAddr, TRDP_FLAGS_T pktFlags, const TRDP_COM_PARAM_T *pRec ← Params, UINT32 timeout, TRDP_TO_BEHAVIOR_T toBehavior)

Prepare for receiving PD messages.

• EXT_DECL TRDP_ERR_T tlp_resubscribe (TRDP_APP_SESSION_T appHandle, TRDP_SUB_T sub ← Handle, UINT32 etbTopoCnt, UINT32 opTrnTopoCnt, TRDP_IP_ADDR_T srclpAddr1, TRDP_IP_ADDR_T srclpAddr2, TRDP_IP_ADDR_T destlpAddr)

Reprepare for receiving PD messages.

• EXT_DECL TRDP_ERR_T tlp_unsubscribe (TRDP_APP_SESSION_T appHandle, TRDP_SUB_T sub↔ Handle)

Stop receiving PD messages.

Get the last valid PD message.

• EXT_DECL TRDP_ERR_T tlm_process (TRDP_APP_SESSION_T appHandle, TRDP_FDS_T *pRfds, IN ← T32 *pCount)

Message Data Work loop of the TRDP handler.

 EXT_DECL_TRDP_ERR_T_tlm_getInterval (TRDP_APP_SESSION_T appHandle, TRDP_TIME_T *p→ Interval, TRDP_FDS_T *pFileDesc, INT32 *pNoDesc)

Get the lowest time interval for MDs.

EXT_DECL TRDP_ERR_T tlm_notify (TRDP_APP_SESSION_T appHandle, const void *pUserRef, TRD← P_MD_CALLBACK_T pfCbFunction, UINT32 comld, UINT32 etbTopoCnt, UINT32 opTrnTopoCnt, TRDP_← IP_ADDR_T srclpAddr, TRDP_IP_ADDR_T destlpAddr, TRDP_FLAGS_T pktFlags, const TRDP_SEND_← PARAM_T *pSendParam, const UINT8 *pData, UINT32 dataSize, const TRDP_URI_USER_T sourceURI, const TRDP_URI_USER_T destURI)

Initiate sending MD notification message.

EXT_DECL TRDP_ERR_T tlm_request (TRDP_APP_SESSION_T appHandle, const void *pUserRef, T← RDP_MD_CALLBACK_T pfCbFunction, TRDP_UUID_T *pSessionId, UINT32 comId, UINT32 etbTopo← Cnt, UINT32 opTrnTopoCnt, TRDP_IP_ADDR_T srcIpAddr, TRDP_IP_ADDR_T destIpAddr, TRDP_FLA← GS_T pktFlags, UINT32 numReplies, UINT32 replyTimeout, const TRDP_SEND_PARAM_T *pSendParam, const UINT8 *pData, UINT32 dataSize, const TRDP_URI_USER_T sourceURI, const TRDP_URI_USER_T destURI)

Initiate sending MD request message.

• EXT_DECL TRDP_ERR_T tlm_confirm (TRDP_APP_SESSION_T appHandle, const TRDP_UUID_T *p↔ SessionId, UINT16 userStatus, const TRDP_SEND_PARAM_T *pSendParam)

Initiate sending MD confirm message.

EXT_DECL TRDP_ERR_T tlm_abortSession (TRDP_APP_SESSION_T appHandle, const TRDP_UUID_T *pSessionId)

Cancel an open session.

EXT_DECL TRDP_ERR_T tlm_addListener (TRDP_APP_SESSION_T appHandle, TRDP_LIS_T *pListen← Handle, const void *pUserRef, TRDP_MD_CALLBACK_T pfCbFunction, BOOL8 comIdListener, UINT32 comId, UINT32 etbTopoCnt, UINT32 opTrnTopoCnt, TRDP_IP_ADDR_T srclpAddr1, TRDP_IP_ADDR_← T srclpAddr2, TRDP_IP_ADDR_T mcDestlpAddr, TRDP_FLAGS_T pktFlags, const TRDP_URI_USER_T srcURI, const TRDP_URI_USER_T destURI)

Subscribe to MD messages.

• EXT_DECL TRDP_ERR_T tlm_readdListener (TRDP_APP_SESSION_T appHandle, TRDP_LIS_T listen ← Handle, UINT32 etbTopoCnt, UINT32 opTrnTopoCnt, TRDP_IP_ADDR_T srclpAddr, TRDP_IP_ADDR_← T srclpAddr2, TRDP_IP_ADDR_T mcDestlpAddr)

Resubscribe to MD messages.

• EXT_DECL TRDP_ERR_T tlm_delListener (TRDP_APP_SESSION_T appHandle, TRDP_LIS_T listen ← Handle)

Remove Listener.

• EXT_DECL_TRDP_ERR_T tlm_reply (TRDP_APP_SESSION_T appHandle, const_TRDP_UUID_T *p↔ SessionId, UINT32 comId, UINT16 userStatus, const_TRDP_SEND_PARAM_T *pSendParam, const_UINT8 *pData, UINT32 dataSize)

Send a MD reply message.

Send a MD reply query message.

EXT_DECL const CHAR8 * tlc_getVersionString (void)

Return a human readable version representation.

EXT_DECL const TRDP_VERSION_T * tlc_getVersion (void)

Return version.

EXT_DECL TRDP_ERR_T tlc_getStatistics (TRDP_APP_SESSION_T appHandle, TRDP_STATISTICS_T *pStatistics)

Return statistics.

 EXT_DECL TRDP_ERR_T tlc_getSubsStatistics (TRDP_APP_SESSION_T appHandle, UINT16 *pNum← Subs, TRDP_SUBS_STATISTICS_T *pStatistics)

Return PD subscription statistics.

• EXT_DECL TRDP_ERR_T tlc_getPubStatistics (TRDP_APP_SESSION_T appHandle, UINT16 *pNumPub, TRDP PUB STATISTICS T *pStatistics)

Return PD publish statistics.

• EXT_DECL TRDP_ERR_T tlc_getRedStatistics (TRDP_APP_SESSION_T appHandle, UINT16 *pNumRed, TRDP RED STATISTICS T *pStatistics)

Return redundancy group statistics.

 EXT_DECL TRDP_ERR_T tlc_getJoinStatistics (TRDP_APP_SESSION_T appHandle, UINT16 *pNumJoin, UINT32 *plpAddr)

Return ioin statistics.

EXT DECL TRDP ERR T tlc resetStatistics (TRDP APP SESSION T appHandle)

Reset statistics.

5.23.1 Detailed Description

TRDP Light interface functions (API)

Low level functions for communicating using the TRDP protocol

Note

Project: TCNOpen TRDP prototype stack

Author

Bernd Loehr, NewTec GmbH

Remarks

This Source Code Form is subject to the terms of the Mozilla Public License, v. 2.0. If a copy of the MPL was not distributed with this file, You can obtain one at http://mozilla.org/MPL/2.0/. Copyright Bombardier Transportation Inc. or its subsidiaries and others, 2013-2019. All rights reserved.

5.23.2 Function Documentation

5.23.2.1 tlc_closeSession()

Close a session.

Clean up and release all resources of that session

Parameters

	in	appHandle	The handle returned by tlc_openSession
--	----	-----------	--

Return values

TRDP_NO_ERR	no error
TRDP_NOINIT_ERR	handle invalid
TRDP_PARAM_ERR	handle NULL

5.23.2.2 tlc_configSession()

```
const TRDP_MARSHALL_CONFIG_T * pMarshall,
const TRDP_PD_CONFIG_T * pPdDefault,
const TRDP_MD_CONFIG_T * pMdDefault,
const TRDP_PROCESS_CONFIG_T * pProcessConfig )
```

(Re-)configure a session.

tlc_configSession is called by openSession, but may also be called later on to change the defaults. Only the supplied settings (pointer != NULL) will be evaluated.

Parameters

appHandle	A handle for further calls to the trdp stack
pMarshall	Pointer to marshalling configuration
pPdDefault	Pointer to default PD configuration
pMdDefault	Pointer to default MD configuration
pProcessConfig	Pointer to process configuration only option parameter is used here to define session behavior all other parameters are only used to feed statistics
	pMarshall pPdDefault pMdDefault

Return values

TRDP_NO_ERR	no error
TRDP_INIT_ERR	not yet inited
TRDP_PARAM_ERR	parameter error

5.23.2.3 tlc_getETBTopoCount()

Set new topocount for trainwide communication.

This value is used for validating outgoing and incoming packets only!

Parameters

in	appHandle	the handle returned by tlc_openSession
----	-----------	--

Return values

```
etbTopoCnt
```

5.23.2.4 tlc_getInterval()

```
TRDP_TIME_T * pInterval,
TRDP_FDS_T * pFileDesc,
INT32 * pNoDesc )
```

Get the lowest time interval for PDs.

Return the maximum time interval suitable for 'select()' so that we can send due PD packets in time. If the PD send queue is empty, return zero time

Parameters

in	appHandle	The handle returned by tlc_openSession
out	pInterval	pointer to needed interval
in,out	pFileDesc	pointer to file descriptor set
out	pNoDesc	pointer to put no of highest used descriptors (for select())

Return values

TRDP_NO_ERR	no error
TRDP_NOINIT_ERR	handle invalid

5.23.2.5 tlc_getJoinStatistics()

Return join statistics.

Memory for statistics information must be provided by the user.

Parameters

in	appHandle	the handle returned by tlc_openSession
in,out	pNumJoin	Pointer to the number of joined IP Adresses
out	plpAddr	Pointer to a list with the joined IP adresses

Return values

TRDP_NO_ERR	no error
TRDP_NOINIT_ERR	handle invalid
TRDP_PARAM_ERR	parameter error
TRDP_MEM_ERR	there are more items than requested

5.23.2.6 tlc_getOpTrainTopoCount()

Set new operational train topocount for direction/orientation sensitive communication.

This value is used for validating outgoing and incoming packets only!

Parameters

in	appHandle	The handle returned by tlc_openSession]
----	-----------	--	---

Return values

New operational topocount value	opTrnTopoCnt
---------------------------------	--------------

5.23.2.7 tlc_getOwnlpAddress()

Get the interface address.

Parameters

out	appHandle	A handle for further calls to the trdp stack
-----	-----------	--

Return values

```
real←
IP
```

5.23.2.8 tlc_getPubStatistics()

Return PD publish statistics.

Memory for statistics information must be provided by the user.

Parameters

in	appHandle	the handle returned by tlc_openSession
in,out	pNumPub	Pointer to the number of publishers
out	pStatistics	Pointer to a list with the publish statistics information

Return values

TRDP_NO_ERR	no error	
TRDP_NOINIT_ERR	handle invalid	
TRDP_PARAM_ERR	parameter error	
TRDP_MEM_ERR	there are more subscriptions than requested	

5.23.2.9 tlc_getRedStatistics()

Return redundancy group statistics.

Memory for statistics information must be provided by the user.

Parameters

in	appHandle	the handle returned by tlc_openSession	
in,out	pNumRed	Pointer to the number of redundancy groups	
out	pStatistics	Pointer to a list with the redundancy group information	

Return values

TRDP NO ERR	no error	
TRDP_NOINIT_ERR	handle invalid	
TRDP_PARAM_ERR	parameter error	
TRDP_MEM_ERR	there are more subscriptions than requested	

5.23.2.10 tlc_getStatistics()

Return statistics.

Memory for statistics information must be provided by the user.

Parameters

in	appHandle	the handle returned by tlc_openSession
out	pStatistics	Pointer to statistics for this application session

Return values

TRDP_NO_ERR	no error
TRDP_NOINIT_ERR	handle invalid
TRDP_PARAM_ERR	parameter error

5.23.2.11 tlc_getSubsStatistics()

Return PD subscription statistics.

Memory for statistics information must be provided by the user.

Parameters

in	appHandle	the handle returned by tlc_openSession	
in, out	pNumSubs	In: The number of subscriptions requested Out: Number of subscriptions returned	
in, out	pStatistics	Pointer to an array with the subscription statistics information	

Return values

TRDP_NO_ERR	no error	
TRDP_NOINIT_ERR	handle invalid	
TRDP_PARAM_ERR	parameter error	
TRDP_MEM_ERR	there are more subscriptions than requested	

5.23.2.12 tlc_getVersion()

Return version.

Return pointer to version structure

```
TRDP_VERSION↔
_T
```

5.23.2.13 tlc_getVersionString()

Return a human readable version representation.

Return string in the form 'v.r.u.b'

Return values

5.23.2.14 tlc_init()

Support for message data can only be excluded during compile time!

Support for message data can only be excluded during compile time!

tlc_init initializes the memory subsystem and takes a function pointer to an output function for logging.

Parameters

in	pPrintDebugString	Pointer to debug print function
in	pRefCon	user context
in	pMemConfig	Pointer to memory configuration

Return values

TRDP_NO_ERR	no error
TRDP_MEM_ERR	memory allocation failed
TRDP_PARAM_ERR	initialization error

5.23.2.15 tlc_openSession()

```
const TRDP_MD_CONFIG_T * pMdDefault,
const TRDP_PROCESS_CONFIG_T * pProcessConfig )
```

Open a session with the TRDP stack.

tlc_openSession returns in pAppHandle a unique handle to be used in further calls to the stack.

Parameters

out	pAppHandle	A handle for further calls to the trdp stack	
in	ownlpAddr	Own IP address, can be different for each process in multihoming systems, if zero,	
		the default interface / IP will be used.	
in	leaderlpAddr	IP address of redundancy leader	
in	pMarshall	Pointer to marshalling configuration	
in	pPdDefault	Pointer to default PD configuration	
in	pMdDefault	Pointer to default MD configuration	
in	pProcessConfig	Pointer to process configuration only option parameter is used here to define session behavior all other parameters are only used to feed statistics	

Return values

TRDP_NO_ERR	no error
TRDP_INIT_ERR	not yet inited
TRDP_PARAM_ERR	parameter error
TRDP_SOCK_ERR	socket error

5.23.2.16 tlc_presetIndexSession()

Preset the index table sizes of a session.

tlc_presetIndexSession allows to preallocate the table sizes in HIGH_PERF_INDEXED mode. If no table sizes are provided, the default sizes are used. In normal mode, this is a no-op. This function should be called during initialisation stage, e.g. right after a session has been opened.

Parameters

in	appHandle	The handle returned by tlc_openSession
in	pIndexTableSizes	Pointer to a table of sizes to reserve the memory

TRDP_NO_ERR	no error
TRDP_INIT_ERR	not yet inited
TRDP_PARAM_ERR	parameter error

5.23.2.17 tlc_process()

Work loop of the TRDP handler.

Search the queue for pending PDs and MDs to be sent Search the receive queue for pending PDs and MDs (time out)

Note: If using tlc_process(), do not use tlp_process*() and tlm_process() calls at the same time! Single thread usage -> use tlc_getInterval(), vos_select(), tlc_process() Multiple threads -> thread 1: use tlp_getInterval(), vos_select(), tlp_processReceive() -> thread 2: cyclically call tlp_processSend() -> thread 3: use tlm_getInterval(), vos_select(), tlm_process() for message data

Also see User Manual.

Parameters

in	appHandle	The handle returned by tlc_openSession
in	pRfds	pointer to set of ready descriptors
in,out	pCount	pointer to number of ready descriptors

Return values

TRDP_NO_ERR	no error
TRDP_NOINIT_ERR	handle invalid

5.23.2.18 tlc_reinitSession()

Re-Initialize.

Should be called by the application when a link-down/link-up event has occured during normal operation. We need to re-join the multicast groups...

Parameters

in	appHandle	The handle returned by tlc_openSession
----	-----------	--

TRDP_NO_ERR	no error
TRDP_NOINIT_ERR	handle invalid

Return values

TRDP_PARAM_ERR	handle NULL
----------------	-------------

5.23.2.19 tlc_resetStatistics()

Reset statistics.

Parameters

in	appHandle	the handle returned by tlc_openSession
----	-----------	--

Return values

TRDP_NO_ERR	no error
TRDP_NOINIT_ERR	handle invalid
TRDP_PARAM_ERR	parameter error

5.23.2.20 tlc_setETBTopoCount()

Set new topocount for trainwide communication.

This value is used for validating outgoing and incoming packets only!

Parameters

in	appHandle	the handle returned by tlc_openSession
in	etbTopoCnt	New etbTopoCnt value

TRDP_NO_ERR	no error
TRDP_NOINIT_ERR	handle invalid

5.23.2.21 tlc_setOpTrainTopoCount()

Set new operational train topocount for direction/orientation sensitive communication.

This value is used for validating outgoing and incoming packets only!

Parameters

i	in	appHandle	The handle returned by tlc_openSession
i	in	opTrnTopoCnt	New operational topocount value

Return values

TRDP_NO_ERR	no error
TRDP_NOINIT_ERR	handle invalid

5.23.2.22 tlc_terminate()

Un-Initialize.

Clean up and close all sessions. Mainly used for debugging/test runs. No further calls to library allowed

Return values

TRDP_NO_ERR	no error
TRDP_INIT_ERR	no error
TRDP_MEM_ERR	TrafficStore nothing
TRDP_MUTEX_ERR	TrafficStore mutex err

5.23.2.23 tlc_updateSession()

Update a session.

tlc_updateSession signals the end of the set-up phase to the stack. It shall be called after the last publisher and subscriber was added and will create and compute the index tables to be used by the high-performance targets. This function is currently a no-op on standard targets.

Parameters

	in	appHandle	The handle returned by tlc_openSession	
--	----	-----------	--	--

Return values

TRDP_NO_ERR	no error
TRDP_INIT_ERR	not yet inited
TRDP_PARAM_ERR	parameter error

5.23.2.24 tlm_abortSession()

Cancel an open session.

Abort an open session; any pending messages will be dropped

Parameters

in	appHandle	the handle returned by tlc_openSession
in	p⇔	Session ID returned by request
	SessionId	

Return values

TRDP_NO_ERR	no error
TRDP_NOSESSION_ERR	no such session
TRDP_NOINIT_ERR	handle invalid

5.23.2.25 tlm_addListener()

```
EXT_DECL TRDP_ERR_T tlm_addListener (

TRDP_APP_SESSION_T appHandle,

TRDP_LIS_T * pListenHandle,

const void * pUserRef,

TRDP_MD_CALLBACK_T pfCbFunction,

BOOL8 comIdListener,

UINT32 comId,

UINT32 etbTopoCnt,

UINT32 opTrnTopoCnt,

TRDP_IP_ADDR_T srcIpAddr1,

TRDP_IP_ADDR_T srcIpAddr2,

TRDP_IP_ADDR_T mcDestIpAddr,
```

```
TRDP_FLAGS_T pktFlags,
const TRDP_URI_USER_T srcURI,
const TRDP_URI_USER_T destURI )
```

Subscribe to MD messages.

Add a listener to TRDP to get notified when messages are received

Parameters

in	appHandle	the handle returned by tlc_openSession	
out	pListenHandle	Handle for this listener returned	
in	pUserRef	ef user supplied value returned with received message	
in	pfCbFunction	n Pointer to listener specific callback function, NULL to use default function	
in	comldListener	set TRUE if comld shall be observed	
in	comld	comld to be observed	
in	etbTopoCnt	ETB topocount to use, 0 if consist local communication	
in	opTrnTopoCnt	nt operational topocount, != 0 for orientation/direction sensitive communication	
in	srclpAddr1	Source IP address, lower address in case of address range, set to 0 if not used	
in	srclpAddr2	upper address in case of address range, set to 0 if not used	
in	mcDestlpAddr	multicast group to listen on	
in	pktFlags	tFlags OPTION: TRDP_FLAGS_DEFAULT, TRDP_FLAGS_MARSHALL	
in	srcURI	only functional group of source URI, set to NULL if not used	
in	destURI	only functional group of destination URI, set to NULL if not used	

Return values

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	parameter error
TRDP_MEM_ERR	out of memory
TRDP_NOINIT_ERR	handle invalid

5.23.2.26 tlm_confirm()

Initiate sending MD confirm message.

Send a MD confirmation message User reference, source and destination IP addresses as well as topo counts and packet flags are taken from the session

Parameters

in	appHandle	the handle returned by tlc_openSession	
in	pSessionId	Session ID returned by request	
in	userStatus	Info for requester about application errors	
in	pSendParam	Pointer to send parameters, NULL to use default send parameters	

Generated by Doxygen

Return values

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	parameter error
TRDP_MEM_ERR	out of memory
TRDP_NOSESSION_ERR	no such session
TRDP_NOINIT_ERR	handle invalid

5.23.2.27 tlm_delListener()

Remove Listener.

Parameters

ſ	in	appHandle	the handle returned by tlc_openSession
ſ	out	listenHandle	Handle for this listener

Return values

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	parameter error
TRDP_NOINIT_ERR	handle invalid

5.23.2.28 tlm_getInterval()

Get the lowest time interval for MDs.

Return the maximum time interval suitable for 'select()' so that we can report time outs to the higher layer.

Parameters

in	appHandle	The handle returned by tlc_openSession	
out	pInterval	pointer to needed interval	
in, out	pFileDesc	pointer to file descriptor set	
out	pNoDesc	pointer to put no of highest used descriptors (for select())	

Return values

TRDP_NO_ERR	no error
TRDP_NOINIT_ERR	handle invalid

5.23.2.29 tlm_notify()

```
EXT_DECL TRDP_ERR_T tlm_notify (

TRDP_APP_SESSION_T appHandle,
const void * pUserRef,
TRDP_MD_CALLBACK_T pfCbFunction,
UINT32 comId,
UINT32 etbTopoCnt,
UINT32 opTrnTopoCnt,
TRDP_IP_ADDR_T srcIpAddr,
TRDP_IP_ADDR_T destIpAddr,
TRDP_FLAGS_T pktFlags,
const TRDP_SEND_PARAM_T * pSendParam,
const UINT8 * pData,
UINT32 dataSize,
const TRDP_URI_USER_T sourceURI,
const TRDP_URI_USER_T destURI)
```

Initiate sending MD notification message.

Send a MD notification message

Parameters

· araiii	71010	
in	appHandle	the handle returned by tlc_openSession
in	pUserRef	user supplied value returned with reply
in	pfCbFunction	Pointer to listener specific callback function, NULL to use default function
in	comld	comld of packet to be sent
in	etbTopoCnt	ETB topocount to use, 0 if consist local communication
in	opTrnTopoCnt	operational topocount, != 0 for orientation/direction sensitive communication
in	srclpAddr	own IP address, 0 - srcIP will be set by the stack
in	destlpAddr	where to send the packet to
in	pktFlags	OPTION: TRDP_FLAGS_DEFAULT, TRDP_FLAGS_NONE, TRDP_FLAGS_MARSHALL, TRDP_FLAGS_CALLBACK
in	pSendParam	optional pointer to send parameter, NULL - default parameters are used
in	pData	pointer to packet data / dataset
in	dataSize	size of packet data
in	sourceURI	only functional group of source URI
in	destURI	only functional group of destination URI

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	parameter error
TRDP_MEM_ERR	out of memory

Return values

TRDP_NOINIT_ERR	handle invalid
-----------------	----------------

5.23.2.30 tlm_process()

Message Data Work loop of the TRDP handler.

Search the queue for pending MDs to be sent Search the receive queue for pending MDs (replies, time outs) and incoming requests

Parameters

	in	appHandle The handle returned by tlc_openS		
	in	pRfds	pointer to set of ready descriptors	
Ī	in,out	pCount	pointer to number of ready descriptors	

Return values

TRDP_NO_ERR	no error
TRDP_NOINIT_ERR	handle invalid

5.23.2.31 tlm_readdListener()

```
EXT_DECL TRDP_ERR_T tlm_readdListener (

TRDP_APP_SESSION_T appHandle,

TRDP_LIS_T listenHandle,

UINT32 etbTopoCnt,

UINT32 opTrnTopoCnt,

TRDP_IP_ADDR_T srcIpAddr1,

TRDP_IP_ADDR_T srcIpAddr2,

TRDP_IP_ADDR_T mcDestIpAddr )
```

Resubscribe to MD messages.

Readd a listener after topoCount changes to get notified when messages are received

Parameters

in	appHandle	the handle returned by tlc_openSession	
out	listenHandle	Handle for this listener	
in	etbTopoCnt	ETB topocount to use, 0 if consist local communication	

Parameters

in	opTrnTopoCnt	operational topocount, != 0 for orientation/direction sensitive communication	
in	srclpAddr1	Source IP address, lower address in case of address range, set to 0 if not used	
in	srclpAddr2	upper address in case of address range, set to 0 if not used	
in	mcDestlpAddr	multicast group to listen on	

Return values

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	parameter error
TRDP_MEM_ERR	out of memory
TRDP_NOINIT_ERR	handle invalid

5.23.2.32 tlm_reply()

```
EXT_DECL TRDP_ERR_T tlm_reply (

TRDP_APP_SESSION_T appHandle,

const TRDP_UUID_T * pSessionId,

UINT32 comId,

UINT16 userStatus,

const TRDP_SEND_PARAM_T * pSendParam,

const UINT8 * pData,

UINT32 dataSize )
```

Send a MD reply message.

Send a MD reply message after receiving an request User reference, source and destination IP addresses as well as topo counts and packet flags are taken from the session

Parameters

in	appHandle	the handle returned by tlc_openSession
in	pSessionId	Session ID returned by indication
in	comId	comld of packet to be sent
in	userStatus	Info for requester about application errors
in	pSendParam	Pointer to send parameters, NULL to use default send parameters
in	pData	pointer to packet data / dataset
in	dataSize	size of packet data

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	parameter error
TRDP_MEM_ERR	Out of memory
TRDP_NO_SESSION_ERR	no such session
TRDP_NOINIT_ERR	handle invalid

5.23.2.33 tlm_replyQuery()

```
EXT_DECL TRDP_ERR_T tlm_replyQuery (

TRDP_APP_SESSION_T appHandle,

const TRDP_UUID_T * pSessionId,

UINT32 comId,

UINT16 userStatus,

UINT32 confirmTimeout,

const TRDP_SEND_PARAM_T * pSendParam,

const UINT3 * pData,

UINT32 dataSize )
```

Send a MD reply query message.

Send a MD reply query message after receiving a request and ask for confirmation. User reference, source and destination IP addresses as well as topo counts and packet flags are taken from the session

Parameters

in	appHandle	the handle returned by tlc_openSession	
in	pSessionId	Session ID returned by indication	
in	comld	comld of packet to be sent	
in	userStatus	Info for requester about application errors	
in	confirmTimeout	timeout for confirmation	
in	pSendParam	Pointer to send parameters, NULL to use default send parameters	
in	pData	pointer to packet data / dataset	
in	dataSize	size of packet data	

Return values

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	parameter error
TRDP_MEM_ERR	out of memory
TRDP_NO_SESSION_ERR	no such session
TRDP_NOINIT_ERR	handle invalid

5.23.2.34 tlm_request()

```
EXT_DECL TRDP_ERR_T tlm_request (

TRDP_APP_SESSION_T appHandle,
const void * pUserRef,
TRDP_MD_CALLBACK_T pfCbFunction,
TRDP_UUID_T * pSessionId,
UINT32 comId,
UINT32 etbTopoCnt,
UINT32 opTrnTopoCnt,
TRDP_IP_ADDR_T srcIpAddr,
```

```
TRDP_IP_ADDR_T destIpAddr,

TRDP_FLAGS_T pktFlags,

UINT32 numReplies,

UINT32 replyTimeout,

const TRDP_SEND_PARAM_T * pSendParam,

const UINT8 * pData,

UINT32 dataSize,

const TRDP_URI_USER_T sourceURI,

const TRDP_URI_USER_T destURI)
```

Initiate sending MD request message.

Send a MD request message

Parameters

in	appHandle	the handle returned by tlc_openSession	
in	pUserRef	user supplied value returned with reply	
in	pfCbFunction	Pointer to listener specific callback function, NULL to use default function	
out	pSessionId	return session ID	
in	comld	comld of packet to be sent	
in	etbTopoCnt	ETB topocount to use, 0 if consist local communication	
in	opTrnTopoCnt	operational topocount, != 0 for orientation/direction sensitive communication	
in	srclpAddr	own IP address, 0 - srcIP will be set by the stack	
in	destlpAddr	where to send the packet to	
in	pktFlags	OPTION: TRDP_FLAGS_DEFAULT, TRDP_FLAGS_NONE, TRDP_FLAGS_MARSHALL	
in	numReplies	number of expected replies, 0 if unknown	
in	replyTimeout	timeout for reply	
in	pSendParam	Pointer to send parameters, NULL to use default send parameters	
in	pData	pointer to packet data / dataset	
in	dataSize	size of packet data	
in	sourceURI	only functional group of source URI	
in	destURI	only functional group of destination URI	

Return values

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	parameter error
TRDP_MEM_ERR	out of memory
TRDP_NOINIT_ERR	handle invalid

5.23.2.35 tlp_get()

```
EXT_DECL TRDP_ERR_T tlp_get (
          TRDP_APP_SESSION_T appHandle,
          TRDP_SUB_T subHandle,
          TRDP_PD_INFO_T * pPdInfo,
          UINT8 * pData,
          UINT32 * pDataSize )
```

Get the last valid PD message.

This allows polling of PDs instead of event driven handling by callbacks

Parameters

in	appHandle	the handle returned by tlc_openSession	
in	subHandle the handle returned by subscription		
in,out	pPdInfo	pointer to application's info buffer	
in,out	pData	pointer to application's data buffer	
in,out	pDataSize	in: size of buffer, out: size of data	

Return values

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	parameter error
TRDP_SUB_ERR	not subscribed
TRDP_TIMEOUT_ERR	packet timed out
TRDP_NOINIT_ERR	handle invalid
TRDP_COMID_ERR	ComID not found when marshalling

5.23.2.36 tlp_getInterval()

Get the lowest time interval for PDs.

Return the maximum time interval suitable for 'select()' so that we can send due PD packets in time. If the PD send queue is empty, return zero time

Parameters

in	appHandle	The handle returned by tlc_openSession	
out	pInterval	pointer to needed interval	
in,out	pFileDesc	pointer to file descriptor set	
out	pNoDesc	pointer to put no of highest used descriptors (for select())	

TRDP_NO_ERR	no error
TRDP_NOINIT_ERR	handle invalid

5.23.2.37 tlp_getRedundant()

Get status of redundant Comlds.

Only the status of the first found redundancy group entry will be returned!

Parameters

in	appHandle	the handle returned by tlc_openSession	
in	redId	will be returned for all ComID's with the given redId	
in,out	pLeader	TRUE if we're sending this redundancy group (leader)	

Return values

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	redld invalid or not existing
TRDP_NOINIT_ERR	handle invalid

5.23.2.38 tlp_processReceive()

Work loop of the TRDP handler.

Check the sockets for incoming PD telegrams. Search the receive queue for pending PDs (time out) and report them, either by informing the higher layer via the callback mechanism or just by marking the subscriber as timed-out

Parameters

in	appHandle	The handle returned by tlc_openSession	
in	pRfds	pointer to set of ready descriptors	
in, out	pCount	pointer to number of ready descriptors	

	TRDP_NO_ERR	no error
TR	DP NOINIT ERR	handle invalid

5.23.2.39 tlp_processSend()

Work loop of the TRDP handler.

Search the queue for pending PDs to be sent

Parameters

	in	appHandle	The handle returned by tlc_openSession
--	----	-----------	--

Return values

TRDP_NO_ERR	no error
TRDP_NOINIT_ERR	handle invalid

5.23.2.40 tlp_publish()

```
EXT_DECL TRDP_ERR_T tlp_publish (
             TRDP_APP_SESSION_T appHandle,
             TRDP_PUB_T * pPubHandle,
             const void * pUserRef,
             TRDP_PD_CALLBACK_T pfCbFunction,
             UINT32 serviceId,
             UINT32 comId,
             UINT32 etbTopoCnt,
             UINT32 opTrnTopoCnt,
             TRDP_IP_ADDR_T srcIpAddr,
             TRDP_IP_ADDR_T destIpAddr,
             UINT32 interval,
             UINT32 redId,
             TRDP_FLAGS_T pktFlags,
             const TRDP_SEND_PARAM_T * pSendParam,
             const UINT8 * pData,
             UINT32 dataSize )
```

Prepare for sending PD messages.

Queue a PD message, it will be send when tlc_publish has been called

Parameters

in	in appHandle the handle returned by tlc_openSession	
out pPubHandle returned handle for related re/unpublish		returned handle for related re/unpublish
in	pUserRef	user supplied value returned within the info structure of callback function
in	pfCbFunction	Pointer to pre-send callback function, NULL if not used
in serviceId optional serviceId this telegram belongs to (default = 0)		optional serviceld this telegram belongs to (default = 0)
in	comld	comld of packet to send
in	etbTopoCnt	ETB topocount to use, 0 if consist local communication

Parameters

in	opTrnTopoCnt	operational topocount, != 0 for orientation/direction sensitive communication
in srclpAddr own IP address, 0 - srcIP will be set by the stack		own IP address, 0 - srcIP will be set by the stack
in destlpAddr where to send the packet to		where to send the packet to
in	interval	frequency of PD packet (>= 10ms) in usec
in	redId	0 - Non-redundant, > 0 valid redundancy group
in	pktFlags	OPTION: TRDP_FLAGS_DEFAULT, TRDP_FLAGS_NONE, TRDP_FLAGS_MARSHALL, TRDP_FLAGS_CALLBACK
in	pSendParam	optional pointer to send parameter, NULL - default parameters are used
in	pData	optional pointer to data packet / dataset, NULL if sending starts later with tlp_put()
in	dataSize	size of data packet >= 0 and <= TRDP_MAX_PD_DATA_SIZE

Return values

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	parameter error
TRDP_MEM_ERR	could not insert (out of memory)
TRDP_NOINIT_ERR	handle invalid

5.23.2.41 tlp_put()

```
EXT_DECL TRDP_ERR_T tlp_put (
          TRDP_APP_SESSION_T appHandle,
          TRDP_PUB_T pubHandle,
          const UINT8 * pData,
          UINT32 dataSize )
```

Update the process data to send.

Update previously published data. The new telegram will be sent earliest when tlc_process is called.

Parameters

in	appHandle	the handle returned by tlc_openSession
in	pubHandle	the handle returned by publish
in,out	pData	pointer to application's data buffer
in,out	dataSize	size of data

TRDP_NO_ERR	no error	
TRDP_PARAM_ERR	parameter error on uninitialized parameter or changed dataSize compared to published one	
TRDP_NOPUB_ERR	not published	
TRDP_NOINIT_ERR	handle invalid	
TRDP_COMID_ERR	ComID not found when marshalling	

5.23.2.42 tlp_putImmediate()

```
EXT_DECL TRDP_ERR_T tlp_putImmediate (

TRDP_APP_SESSION_T appHandle,

TRDP_PUB_T pubHandle,

const UINT8 * pData,

UINT32 dataSize,

VOS_TIMEVAL_T * pTxTime )
```

Update and send process data.

Update previously published data. The new telegram will be sent immediatly or at txTime, if txTime != 0 and TSN == 1 Should be used if application (or higher layer, e.g. ara::com and acyclic events) needs full control over process data schedule.

Note: For TSN this function is not protected by any mutexes and should not be called while adding or removing any publishers, subscribers or even sessions! Also: Marshalling is not supported!

Parameters

in appHandle the handle returned		the handle returned by tlc_openSession
in	pubHandle	the handle returned by publish
in,out	pData	pointer to application's data buffer
in,out	dataSize	size of data
in	pTxTime	when to send (absolute time), optional for TSN only

Return values

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	parameter error on uninitialized parameter or changed dataSize compared to published one
TRDP_NOPUB_ERR	not published
TRDP_NOINIT_ERR	handle invalid

5.23.2.43 tlp_republish()

Prepare for sending PD messages.

Reinitialize and queue a PD message, it will be send when tlc_publish has been called

Parameters

in	appHandle	the handle returned by tlc_openSession
in	in pubHandle handle for related unpublish	
in	etbTopoCnt	ETB topocount to use, 0 if consist local communication
in	opTrnTopoCnt	operational topocount, != 0 for orientation/direction sensitive communication
in	srclpAddr	own IP address, 0 - srcIP will be set by the stack
in	destlpAddr	where to send the packet to

Return values

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	parameter error
TRDP_MEM_ERR	could not insert (out of memory)
TRDP_NOINIT_ERR	handle invalid

5.23.2.44 tlp_request()

```
EXT_DECL TRDP_ERR_T tlp_request (
            TRDP_APP_SESSION_T appHandle,
            TRDP_SUB_T subHandle,
            UINT32 serviceId,
            UINT32 comId,
            UINT32 etbTopoCnt,
             UINT32 opTrnTopoCnt,
             TRDP_IP_ADDR_T srcIpAddr,
             TRDP_IP_ADDR_T destIpAddr,
             UINT32 redId,
             TRDP_FLAGS_T pktFlags,
            const TRDP_SEND_PARAM_T * pSendParam,
             const UINT8 * pData,
             UINT32 dataSize,
             UINT32 replyComId,
             TRDP_IP_ADDR_T replyIpAddr )
```

Initiate sending PD messages (PULL).

Send a PD request message

Parameters

in	appHandle	the handle returned by tlc_openSession	
in	subHandle	handle from related subscribe	
in	serviceld	optional serviceld this telegram belongs to (default = 0)	
in	comld	comld of packet to be sent	
in	etbTopoCnt	ETB topocount to use, 0 if consist local communication	
in	opTrnTopoCnt	operational topocount, != 0 for orientation/direction sensitive communication	
in	srclpAddr	own IP address, 0 - srcIP will be set by the stack	
in	destlpAddr	where to send the packet to	
in	redld	0 - Non-redundant, > 0 valid redundancy group	

Parameters

in	pktFlags	OPTION: TRDP_FLAGS_DEFAULT, TRDP_FLAGS_NONE, TRDP_FLAGS_MARSHALL,	
		TRDP_FLAGS_CALLBACK	
in	pSendParam	optional pointer to send parameter, NULL - default parameters are used	
in	pData	pointer to packet data / dataset	
in	dataSize	size of packet data	
in	replyComId	comld of reply (default comID of subscription)	
in	replylpAddr	IP for reply	

Return values

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	parameter error
TRDP_MEM_ERR	could not insert (out of memory)
TRDP_NOINIT_ERR	handle invalid
TRDP_NOSUB_ERR	no matching subscription found

5.23.2.45 tlp_resubscribe()

Reprepare for receiving PD messages.

Resubscribe to a specific PD ComID and source IP

Parameters

in	appHandle	the handle returned by tlc_openSession	
in	subHandle	handle for this subscription	
in	etbTopoCnt	ETB topocount to use, 0 if consist local communication	
in	opTrnTopoCnt	operational topocount, != 0 for orientation/direction sensitive communication	
in	srclpAddr1	Source IP address, lower address in case of address range, set to 0 if not used	
in	srclpAddr2	upper address in case of address range, set to 0 if not used	
in	destlpAddr	IP address to join	

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	parameter error
TRDP_MEM_ERR	could not reserve memory (out of memory)
TRDP_NOINIT_ERR	handle invalid

Return values

TRDP_SOCK_ERR	Resource (socket) not available, subscription canceled
---------------	--

5.23.2.46 tlp_setRedundant()

Do not send non-redundant PDs when we are follower.

Parameters

in	appHandle	the handle returned by tlc_openSession
in	redId	will be set for all ComID's with the given redId, 0 to change for all redId
in	leader	TRUE if we send

Return values

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	parameter error / redld not existing
TRDP_NOINIT_ERR	handle invalid

5.23.2.47 tlp_subscribe()

```
EXT_DECL TRDP_ERR_T tlp_subscribe (
             TRDP_APP_SESSION_T appHandle,
             TRDP\_SUB\_T * pSubHandle,
             const void * pUserRef,
             TRDP_PD_CALLBACK_T pfCbFunction,
             UINT32 serviceId,
             UINT32 comId,
             UINT32 etbTopoCnt,
             UINT32 opTrnTopoCnt,
             TRDP_IP_ADDR_T srcIpAddr1,
             TRDP_IP_ADDR_T srcIpAddr2,
             TRDP_IP_ADDR_T destIpAddr,
             {\tt TRDP\_FLAGS\_T} \ \ pktFlags,
             const TRDP_COM_PARAM_T * pRecParams,
             UINT32 timeout,
             TRDP_TO_BEHAVIOR_T toBehavior )
```

Prepare for receiving PD messages.

Subscribe to a specific PD ComID and source IP.

Parameters

appHandle	the handle returned by tlc_openSession	
pSubHandle	return a handle for this subscription	
pUserRef user supplied value returned within the info structure		
pfCbFunction	Pointer to subscriber specific callback function, NULL to use default function	
serviceld	optional serviceld this telegram belongs to (default = 0)	
comld	comld of packet to receive	
etbTopoCnt	ETB topocount to use, 0 if consist local communication	
opTrnTopoCnt	operational topocount, != 0 for orientation/direction sensitive communication	
srclpAddr1	Source IP address, lower address in case of address range, set to 0 if not used	
srclpAddr2	upper address in case of address range, set to 0 if not used	
destlpAddr	IP address to join	
pktFlags	OPTION: TRDP_FLAGS_DEFAULT, TRDP_FLAGS_NONE, TRDP_FLAGS_MARSHALL, TRDP_FLAGS_CALLBACK	
pRecParams	optional pointer to send parameter, NULL - default parameters are used	
timeout	timeout (>= 10ms) in usec	
toBehavior	timeout behavior	
	pSubHandle pUserRef pfCbFunction serviceId comId etbTopoCnt opTrnTopoCnt srcIpAddr1 srcIpAddr2 destIpAddr pktFlags pRecParams timeout	

Return values

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	parameter error
TRDP_MEM_ERR	could not reserve memory (out of memory)
TRDP_NOINIT_ERR	handle invalid

5.23.2.48 tlp_unpublish()

```
EXT_DECL TRDP_ERR_T tlp_unpublish (
          TRDP_APP_SESSION_T appHandle,
          TRDP_PUB_T pubHandle )
```

Stop sending PD messages.

Parameters

in	appHandle	the handle returned by tlc_openSession
in	pubHandle	the handle returned by prepare

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	parameter error
TRDP_NOPUB_ERR	not published
TRDP_NOINIT_ERR	handle invalid

5.23.2.49 tlp_unsubscribe()

```
EXT_DECL TRDP_ERR_T tlp_unsubscribe (
          TRDP_APP_SESSION_T appHandle,
          TRDP_SUB_T subHandle )
```

Stop receiving PD messages.

Unsubscribe to a specific PD ComID

Parameters

in	appHandle	the handle returned by tlc_openSession
in	subHandle	the handle for this subscription

Return values

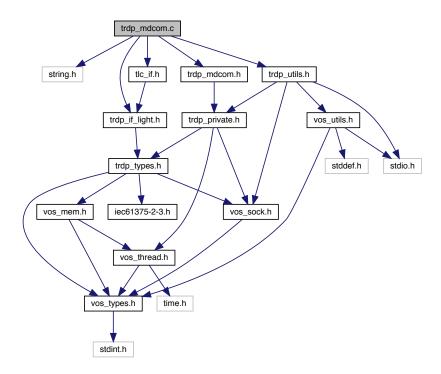
TRDP_NO_ERR	no error
TRDP_PARAM_ERR	parameter error
TRDP_NOSUB_ERR	not subscribed
TRDP_NOINIT_ERR	handle invalid

5.24 trdp_mdcom.c File Reference

Functions for MD communication.

```
#include <string.h>
#include "trdp_if_light.h"
#include "tlc_if.h"
#include "trdp_utils.h"
#include "trdp_mdcom.h"
```

Include dependency graph for trdp_mdcom.c:



Functions

• TRDP ERR T trdp mdGetTCPSocket (TRDP SESSION PT pSession)

Initialize the specific parameters for message data Open a listening socket.

void trdp_mdFreeSession (MD_ELE_T *pMDSession)

Free memory of session.

• TRDP_ERR_T trdp_mdSend (TRDP_SESSION_PT appHandle)

Sending MD messages Send the messages stored in the sendQueue Call user's callback if needed.

void trdp_mdCheckPending (TRDP_APP_SESSION_T appHandle, TRDP_FDS_T *pFileDesc, INT32 *p
 — NoDesc)

Check for pending packets, set FD if non blocking.

void trdp_mdCheckListenSocks (const TRDP_SESSION_PT appHandle, TRDP_FDS_T *pRfds, INT32 *p
 — Count)

Checking receive connection requests and data Call user's callback if needed.

- void trdp mdCheckTimeouts (TRDP SESSION PT appHandle)
 - Checking message data timeouts Call user's callback if needed.
- TRDP_ERR_T trdp_mdReply (const TRDP_MSG_T msgType, TRDP_APP_SESSION_T appHandle, TRD← P_UUID_T pSessionId, UINT32 comId, UINT32 timeout, INT32 replyStatus, const TRDP_SEND_PARAM_T *pSendParam, const UINT8 *pData, UINT32 dataSize)

Send a MD reply/reply query message.

• TRDP_ERR_T trdp_mdCall (const TRDP_MSG_T msgType, TRDP_APP_SESSION_T appHandle, const void *pUserRef, TRDP_MD_CALLBACK_T pfCbFunction, TRDP_UUID_T *pSessionId, UINT32 comId, U ← INT32 etbTopoCnt, UINT32 opTrnTopoCnt, TRDP_IP_ADDR_T srclpAddr, TRDP_IP_ADDR_T destIpAddr, TRDP_FLAGS_T pktFlags, UINT32 numExpReplies, UINT32 replyTimeout, INT32 replyStatus, const TR← DP_SEND_PARAM_T *pSendParam, const UINT8 *pData, UINT32 dataSize, const TRDP_URI_USER_T srcURI, const TRDP_URI_USER_T destURI)

Initiate sending MD request message - private SW level Send a MD request message.

 TRDP_ERR_T trdp_mdConfirm (TRDP_APP_SESSION_T appHandle, const TRDP_UUID_T *pSessionId, UINT16 userStatus, const TRDP_SEND_PARAM_T *pSendParam)

Initiate sending MD confirm message - private SW level Send a MD confirmation message User reference, source and destination IP addresses as well as topo counts and packet flags are taken from the session.

5.24.1 Detailed Description

Functions for MD communication.

Note

Project: TCNOpen TRDP prototype stack

Author

Simone Pachera, FARsystems Gari Oiarbide, CAF Michael Koch, Bombardier Transportations Bernd Loehr, NewTec

Remarks

This Source Code Form is subject to the terms of the Mozilla Public License, v. 2.0. If a copy of the MPL was not distributed with this file, You can obtain one at http://mozilla.org/MPL/2.0/. Copyright Bombardier Transportation Inc. or its subsidiaries and others, 2013-2019. All rights reserved.

5.24.2 Function Documentation

5.24.2.1 trdp_mdCall()

```
TRDP_ERR_T trdp_mdCall (
             const TRDP_MSG_T msgType,
             TRDP_APP_SESSION_T appHandle,
             const void * pUserRef,
             TRDP_MD_CALLBACK_T pfCbFunction,
             TRDP_UUID_T * pSessionId,
             UINT32 comId,
             UINT32 etbTopoCnt,
             UINT32 opTrnTopoCnt,
             TRDP_IP_ADDR_T srcIpAddr,
             TRDP_IP_ADDR_T destIpAddr,
             TRDP_FLAGS_T pktFlags,
             UINT32 numExpReplies,
             UINT32 replyTimeout,
             INT32 replyStatus,
             const TRDP_SEND_PARAM_T * pSendParam,
             const UINT8 * pData,
             UINT32 dataSize,
             const TRDP_URI_USER_T srcURI,
             const TRDP_URI_USER_T destURI )
```

Initiate sending MD request message - private SW level Send a MD request message.

Parameters

	T	TDDD MOO AND TDDD MOO MD
in	msgType	TRDP_MSG_MN or TRDP_MSG_MR
in	appHandle	the handle returned by tlc_init
in	pUserRef	user supplied value returned with reply
in	pfCbFunction	Pointer to listener specific callback function, NULL to use default function
out	pSessionId	return session ID
in	comld	comld of packet to be sent
in	etbTopoCnt	ETB topocount to use, 0 if consist local communication
in	opTrnTopoCnt	operational topocount, != 0 for orientation/direction sensitive communication
in	srclpAddr	own IP address, 0 - srcIP will be set by the stack
in	destlpAddr	where to send the packet to
in	pktFlags	OPTION: TRDP_FLAGS_DEFAULT, TRDP_FLAGS_NONE,
		TRDP_FLAGS_MARSHALL
in	numExpReplies	number of expected replies, 0 if unknown
in	replyTimeout	timeout for reply
in	replyStatus	status to be returned
in	pSendParam	Pointer to send parameters, NULL to use default send parameters
in	pData	pointer to packet data / dataset
in	dataSize	size of packet data
in	srcURI	only functional group of source URI
in	destURI	only functional group of destination URI

Return values

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	parameter error
TRDP_MEM_ERR	out of memory

5.24.2.2 trdp_mdCheckListenSocks()

Checking receive connection requests and data Call user's callback if needed.

Parameters

in	appHandle	session pointer
in	pRfds	pointer to set of ready descriptors
in,out	pCount	pointer to number of ready descriptors

5.24.2.3 trdp_mdCheckPending()

Check for pending packets, set FD if non blocking.

Parameters

in	appHandle	session pointer
in,out	pFileDesc	pointer to set of ready descriptors
in,out	pNoDesc	pointer to number of ready descriptors

5.24.2.4 trdp_mdCheckTimeouts()

Checking message data timeouts Call user's callback if needed.

Parameters

in	appHandle	session pointer
T11	appi iaiiuie	Session pointer

5.24.2.5 trdp_mdConfirm()

Initiate sending MD confirm message - private SW level Send a MD confirmation message User reference, source and destination IP addresses as well as topo counts and packet flags are taken from the session.

Parameters

in	appHandle	the handle returned by tlc_init
in	pSessionId	Session ID returned by request
in	userStatus	Info for requester about application errors
in	pSendParam	Pointer to send parameters, NULL to use default send parameters

TRDP_NO_ERR	no error

Return values

TRDP_PARAM_ERR	parameter error
TRDP_MEM_ERR	out of memory
TRDP_NOSESSION_ERR	no such session

5.24.2.6 trdp_mdFreeSession()

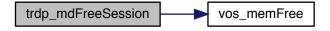
```
void trdp_mdFreeSession ( \label{eq:mdFreeSession} \texttt{MD\_ELE\_T} \ * \ pMDSession \ )
```

Free memory of session.

Parameters

in	pMDSession	session pointer
----	------------	-----------------

Here is the call graph for this function:



5.24.2.7 trdp_mdGetTCPSocket()

Initialize the specific parameters for message data Open a listening socket.

Parameters

in	pSession	session parameters

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	initialization error

5.24.2.8 trdp_mdReply()

Send a MD reply/reply query message.

Send either a MD reply message or a MD reply query message after receiving a request and ask for confirmation. User reference, source and destination IP addresses as well as topo counts and packet flags are taken from the session

Parameters

in	msgType	TRDP_MSG_MP or TRDP_MSG_MQ
in	appHandle	the handle returned by tlc_init
in	pSessionId	Session ID returned by indication
in	comld	comld of packet to be sent
in	timeout	time out for confirmations (zero for TRDP_MSG_MP)
in	replyStatus	Info for requester about application errors
in	pSendParam	Pointer to send parameters, NULL to use default send parameters
in	pData	pointer to packet data / dataset
in	dataSize	size of packet data

Return values

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	parameter error
TRDP_MEM_ERR	out of memory
TRDP_NO_SESSION_ERR	no such session

5.24.2.9 trdp_mdSend()

Sending MD messages Send the messages stored in the sendQueue Call user's callback if needed.

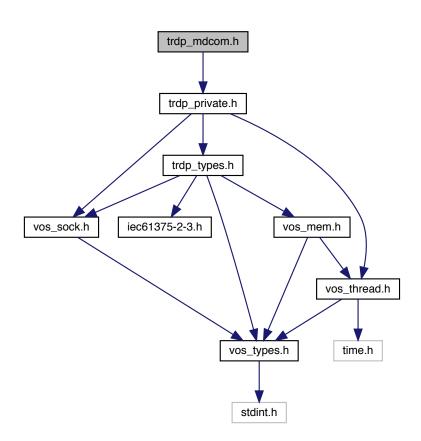
Parameters

in	appHandle	session pointer
----	-----------	-----------------

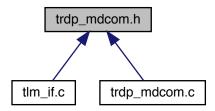
5.25 trdp_mdcom.h File Reference

Functions for MD communication.

#include "trdp_private.h"
Include dependency graph for trdp_mdcom.h:



This graph shows which files directly or indirectly include this file:



Functions

- TRDP ERR T trdp mdGetTCPSocket (TRDP SESSION PT pSession)
 - Initialize the specific parameters for message data Open a listening socket.
- void trdp_mdFreeSession (MD_ELE_T *pMDSession)

Free memory of session.

• TRDP ERR T trdp mdSend (TRDP SESSION PT appHandle)

Sending MD messages Send the messages stored in the sendQueue Call user's callback if needed.

 void trdp_mdCheckPending (TRDP_APP_SESSION_T appHandle, TRDP_FDS_T *pFileDesc, INT32 *p→ NoDesc)

Check for pending packets, set FD if non blocking.

void trdp_mdCheckListenSocks (const TRDP_SESSION_PT appHandle, TRDP_FDS_T *pRfds, INT32 *p← Count)

Checking receive connection requests and data Call user's callback if needed.

- void trdp_mdCheckTimeouts (TRDP_SESSION_PT appHandle)
 - Checking message data timeouts Call user's callback if needed.
- TRDP_ERR_T trdp_mdConfirm (TRDP_APP_SESSION_T appHandle, const TRDP_UUID_T *pSessionId, UINT16 userStatus, const TRDP_SEND_PARAM_T *pSendParam)

Initiate sending MD confirm message - private SW level Send a MD confirmation message User reference, source and destination IP addresses as well as topo counts and packet flags are taken from the session.

 TRDP_ERR_T trdp_mdReply (const TRDP_MSG_T msgType, TRDP_APP_SESSION_T appHandle, TRD← P_UUID_T pSessionId, UINT32 comId, UINT32 timeout, INT32 replyStatus, const TRDP_SEND_PARAM_T *pSendParam, const UINT8 *pData, UINT32 dataSize)

Send a MD reply/reply query message.

TRDP_ERR_T trdp_mdCall (const TRDP_MSG_T msgType, TRDP_APP_SESSION_T appHandle, const void *pUserRef, TRDP_MD_CALLBACK_T pfCbFunction, TRDP_UUID_T *pSessionId, UINT32 comId, U← INT32 etbTopoCnt, UINT32 opTrnTopoCnt, TRDP_IP_ADDR_T srclpAddr, TRDP_IP_ADDR_T destlpAddr, TRDP_FLAGS_T pktFlags, UINT32 numExpReplies, UINT32 replyTimeout, INT32 replyStatus, const TR← DP_SEND_PARAM_T *pSendParam, const UINT8 *pData, UINT32 dataSize, const TRDP_URI_USER_T srcURI, const TRDP_URI_USER_T destURI)

Initiate sending MD request message - private SW level Send a MD request message.

5.25.1 Detailed Description

Functions for MD communication.

Note

Project: TCNOpen TRDP prototype stack

Author

Bernd Loehr, NewTec GmbH

Remarks

This Source Code Form is subject to the terms of the Mozilla Public License, v. 2.0. If a copy of the MPL was not distributed with this file, You can obtain one at http://mozilla.org/MPL/2.0/. Copyright Bombardier Transportation Inc. or its subsidiaries and others, 2013. All rights reserved.

5.25.2 Function Documentation

5.25.2.1 trdp_mdCall()

```
TRDP_ERR_T trdp_mdCall (
            const TRDP_MSG_T msgType,
             TRDP_APP_SESSION_T appHandle,
             const void * pUserRef,
             TRDP_MD_CALLBACK_T pfCbFunction,
             TRDP_UUID_T * pSessionId,
             UINT32 comId,
             UINT32 etbTopoCnt,
             UINT32 opTrnTopoCnt,
             TRDP_IP_ADDR_T srcIpAddr,
             TRDP_IP_ADDR_T destIpAddr,
             TRDP_FLAGS_T pktFlags,
             UINT32 numExpReplies,
             UINT32 replyTimeout,
             INT32 replyStatus,
             const TRDP_SEND_PARAM_T * pSendParam,
             const UINT8 * pData,
             UINT32 dataSize,
             const TRDP_URI_USER_T srcURI,
             const TRDP_URI_USER_T destURI )
```

Initiate sending MD request message - private SW level Send a MD request message.

Parameters

in	msgType	TRDP_MSG_MN or TRDP_MSG_MR	
in	appHandle	the handle returned by tlc_init	
in	pUserRef	user supplied value returned with reply	
in	pfCbFunction	Pointer to listener specific callback function, NULL to use default function	
out	pSessionId	return session ID	
in	comld	comld of packet to be sent	
in	etbTopoCnt	ETB topocount to use, 0 if consist local communication	
in	opTrnTopoCnt	operational topocount, != 0 for orientation/direction sensitive communication	

Parameters

in	srclpAddr	own IP address, 0 - srcIP will be set by the stack	
in	destlpAddr	where to send the packet to	
in	pktFlags	OPTION: TRDP_FLAGS_DEFAULT, TRDP_FLAGS_NONE,	
		TRDP_FLAGS_MARSHALL	
in	numExpReplies	number of expected replies, 0 if unknown	
in	replyTimeout	timeout for reply	
in	replyStatus	status to be returned	
in	pSendParam	Pointer to send parameters, NULL to use default send parameters	
in	pData	pointer to packet data / dataset	
in	dataSize	size of packet data	
in	srcURI	only functional group of source URI	
in	destURI	only functional group of destination URI	

Return values

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	parameter error
TRDP_MEM_ERR	out of memory

5.25.2.2 trdp_mdCheckListenSocks()

Checking receive connection requests and data Call user's callback if needed.

Parameters

in	appHandle	session pointer
in	pRfds	pointer to set of ready descriptors
in,out	pCount	pointer to number of ready descriptors

5.25.2.3 trdp_mdCheckPending()

Check for pending packets, set FD if non blocking.

Parameters

in	appHandle	session pointer
in,out	pFileDesc	pointer to set of ready descriptors
in,out	pNoDesc	pointer to number of ready descriptors

5.25.2.4 trdp_mdCheckTimeouts()

```
void trdp_mdCheckTimeouts ( \label{trdp_mdCheckTimeouts} \mbox{TRDP\_SESSION\_PT } \mbox{\it appHandle} \mbox{\it )}
```

Checking message data timeouts Call user's callback if needed.

Parameters

in	appHandle	session pointer
----	-----------	-----------------

5.25.2.5 trdp_mdConfirm()

Initiate sending MD confirm message - private SW level Send a MD confirmation message User reference, source and destination IP addresses as well as topo counts and packet flags are taken from the session.

Parameters

in	appHandle	the handle returned by tlc_init
in	pSessionId	Session ID returned by request
in	userStatus	Info for requester about application errors
in	pSendParam	Pointer to send parameters, NULL to use default send parameters

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	parameter error
TRDP_MEM_ERR	out of memory
TRDP_NOSESSION_ERR	no such session

5.25.2.6 trdp_mdFreeSession()

```
void trdp_mdFreeSession ( \label{eq:mdFreeSession} \texttt{MD\_ELE\_T} \ * \ pMDSession \ )
```

Free memory of session.

Parameters

in	pMDSession	session pointer
----	------------	-----------------

Here is the call graph for this function:



5.25.2.7 trdp_mdGetTCPSocket()

Initialize the specific parameters for message data Open a listening socket.

Parameters

in	pSession	session parameters
----	----------	--------------------

Return values

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	initialization error

5.25.2.8 trdp_mdReply()

```
UINT32 comId,
UINT32 timeout,
INT32 replyStatus,
const TRDP_SEND_PARAM_T * pSendParam,
const UINT8 * pData,
UINT32 dataSize )
```

Send a MD reply/reply query message.

Send either a MD reply message or a MD reply query message after receiving a request and ask for confirmation. User reference, source and destination IP addresses as well as topo counts and packet flags are taken from the session

Parameters

in	msgType	TRDP_MSG_MP or TRDP_MSG_MQ	
in	appHandle	appHandle the handle returned by tlc_init	
in	pSessionId	Session ID returned by indication	
in	comld	comld of packet to be sent	
in	timeout	time out for confirmations (zero for TRDP_MSG_MP)	
in	replyStatus	Info for requester about application errors	
in	pSendParam	Pointer to send parameters, NULL to use default send parameters	
in	pData	pointer to packet data / dataset	
in	dataSize	size of packet data	

Return values

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	parameter error
TRDP_MEM_ERR	out of memory
TRDP_NO_SESSION_ERR	no such session

5.25.2.9 trdp_mdSend()

Sending MD messages Send the messages stored in the sendQueue Call user's callback if needed.

Parameters

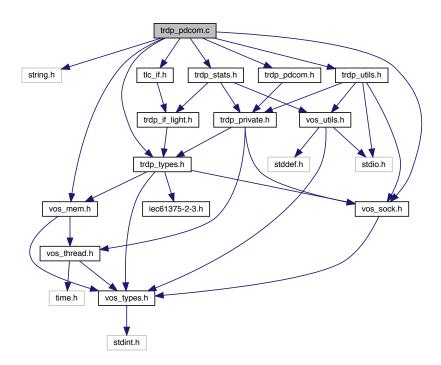
in	appHandle	session pointer
----	-----------	-----------------

5.26 trdp_pdcom.c File Reference

Functions for PD communication.

```
#include <string.h>
#include "trdp_types.h"
#include "trdp_utils.h"
#include "trdp_pdcom.h"
#include "tlc_if.h"
#include "trdp_stats.h"
#include "vos_sock.h"
#include "vos_mem.h"
```

Include dependency graph for trdp pdcom.c:



Functions

• void trdp_pdInit (PD_ELE_T *pPacket, TRDP_MSG_T type, UINT32 etbTopoCnt, UINT32 opTrnTopoCnt, UINT32 replyComId, UINT32 replyIpAddress, UINT32 serviceId)

Initialize/construct the packet Set the header infos.

TRDP_ERR_T trdp_pdPut (PD_ELE_T *pPacket, TRDP_MARSHALL_T marshall, void *refCon, const UI

NT8 *pData, UINT32 dataSize)

Copy data Update the data to be sent.

- TRDP_ERR_T trdp_pdSendImmediate (TRDP_SESSION_PT appHandle, PD_ELE_T *pSendPD) Send PD message immediately.
- TRDP_ERR_T trdp_pdGet (PD_ELE_T *pPacket, TRDP_UNMARSHALL_T unmarshall, void *refCon, const UINT8 *pData, UINT32 *pDataSize)

Copy data Set the header infos.

- TRDP_ERR_T trdp_pdSendElement (TRDP_SESSION_PT appHandle, PD_ELE_T **ppElement)

 Send a due PD message.
- TRDP_ERR_T trdp_pdSendQueued (TRDP_SESSION_PT appHandle)

Send all due PD messages.

• TRDP_ERR_T trdp_pdReceive (TRDP_SESSION_PT appHandle, SOCKET sock)

Receiving PD messages Read the receive socket for arriving PDs, copy the packet to a new PD_ELE_T Check for protocol errors and compare the received data to the data in our receive queue.

 void trdp_pdCheckPending (TRDP_APP_SESSION_T appHandle, TRDP_FDS_T *pFileDesc, INT32 *p↔ NoDesc, int checkSend)

Check for pending packets, set FD if non blocking.

void trdp_pdHandleTimeOuts (TRDP_SESSION_PT appHandle)

Check for time outs.

void trdp_handleTimeout (TRDP_SESSION_PT appHandle, PD_ELE_T *pPacket)

Check a packet for timeout packet Call user's callback, if necessary.

TRDP_ERR_T trdp_pdCheckListenSocks (TRDP_SESSION_PT appHandle, TRDP_FDS_T *pRfds, INT32 *pCount)

Checking receive connection requests and data Call user's callback if needed.

void trdp_pdUpdate (PD_ELE_T *pPacket)

Update the header values.

• TRDP_ERR_T trdp_pdCheck (PD_HEADER_T *pPacket, UINT32 packetSize, int *plsTSN)

Check if the PD header values and the CRCs are sane.

• TRDP_ERR_T trdp_pdSend (SOCKET pdSock, PD_ELE_T *pPacket, UINT16 port)

Send one PD packet.

TRDP_ERR_T trdp_pdDistribute (PD_ELE_T *pSndQueue)

Distribute send time of PD packets over time.

5.26.1 Detailed Description

Functions for PD communication.

Note

Project: TCNOpen TRDP prototype stack

Author

Bernd Loehr, NewTec GmbH

Remarks

This Source Code Form is subject to the terms of the Mozilla Public License, v. 2.0. If a copy of the MPL was not distributed with this file, You can obtain one at http://mozilla.org/MPL/2.0/. Copyright Bombardier Transportation Inc. or its subsidiaries and others, 2013-2019. All rights reserved.

5.26.2 Function Documentation

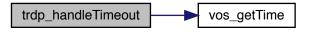
5.26.2.1 trdp_handleTimeout()

Check a packet for timeout packet Call user's callback, if necessary.

Parameters

in	appHandle	Session handle
in	pPacket	pointer to the packet element to check

Here is the call graph for this function:



5.26.2.2 trdp_pdCheck()

Check if the PD header values and the CRCs are sane.

Parameters

in	pPacket	pointer to the packet to check
in	packetSize	max size to check
out	plsTSN	set to TRUE on return if PD2 frame

Return values

TRDP_NO_ERR	
TRDP_CRC_ERR	

5.26.2.3 trdp_pdCheckListenSocks()

Checking receive connection requests and data Call user's callback if needed.

Parameters

in	appHandle	session pointer
in	pRfds	pointer to set of ready descriptors
in,out	pCount	pointer to number of ready descriptors

5.26.2.4 trdp_pdCheckPending()

Check for pending packets, set FD if non blocking.

Parameters

in	appHandle	session pointer
in,out	pFileDesc	pointer to set of ready descriptors
in,out	pNoDesc	pointer to number of ready descriptors
in	checkSend	check send queue, too

Here is the call graph for this function:



5.26.2.5 trdp_pdDistribute()

```
TRDP_ERR_T trdp_pdDistribute ( {\tt PD\_ELE\_T} \ * \ pSndQueue \ )
```

Distribute send time of PD packets over time.

The duration of PD packets on a 100MBit/s network ranges from 3us to 150us max. Because a cyclic thread scheduling below 5ms would put a too heavy load on the system, and PD packets cannot get larger than 1432 (+ UDP header), we will not account for differences in packet size. Another factor is the differences in intervals for different packets: We should only change the starting times of the packets within 1/2 the interval time. Otherwise a late addition of packets could lead to timeouts of already queued packets. Scheduling will be computed based on the smallest interval time.

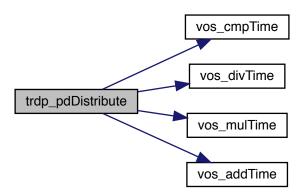
Parameters

in	pSndQueue	pointer to send queue
----	-----------	-----------------------

Return values

```
TRDP_NO_ERR
```

Here is the call graph for this function:



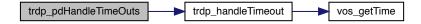
5.26.2.6 trdp_pdHandleTimeOuts()

Check for time outs.

Parameters

in	appHandle	application handle

Here is the call graph for this function:



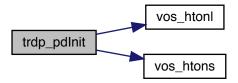
5.26.2.7 trdp_pdlnit()

Initialize/construct the packet Set the header infos.

Parameters

in	pPacket	pointer to the packet element to init
in	type	type the packet
in	etbTopoCnt	topocount to use for PD frame
in	opTrnTopoCnt	topocount to use for PD frame
in	replyComId	Pull request comId
in	replylpAddress	Pull request lp
in	serviceld	Service Id

Here is the call graph for this function:



5.26.2.8 trdp_pdPut()

Copy data Update the data to be sent.

Parameters

in	pPacket	pointer to the packet element to send
in	marshall	pointer to marshalling function
in	refCon	reference for marshalling function
in	pData	pointer to data
in	dataSize	size of data

Return values

TRDP_NO_ERR	no error other errors
-------------	-----------------------

5.26.2.9 trdp_pdReceive()

Receiving PD messages Read the receive socket for arriving PDs, copy the packet to a new PD_ELE_T Check for protocol errors and compare the received data to the data in our receive queue.

If it is a new packet, check if it is a PD Request (PULL). If it is an update, exchange the existing entry with the new one Call user's callback if needed

Parameters

in	appHandle	session pointer
in	sock	the socket to read from

Return values

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	parameter error
TRDP_WIRE_ERR	protocol error (late packet, version mismatch)
TRDP_QUEUE_ERR	not in queue
TRDP_CRC_ERR	header checksum
TRDP_TOPOCOUNT_ERR	invalid topocount

5.26.2.10 trdp_pdSend()

Send one PD packet.

Parameters

in	pdSock	socket descriptor
in	pPacket	pointer to packet to be sent
in	port	port on which to send

Return values

TRDP_NO_ERR	
TRDP_IO_ERR	

5.26.2.11 trdp_pdSendElement()

Send a due PD message.

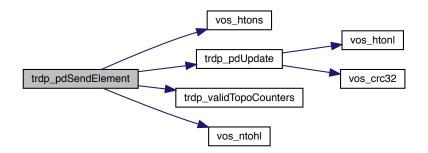
Parameters

in	appHandle	session pointer
in	ppElement	pointer to pointer of the element to send

Return values

TRDP_NO_ERR	no error
TRDP_IO_ERR	socket I/O error

Here is the call graph for this function:



5.26.2.12 trdp_pdSendImmediate()

Send PD message immediately.

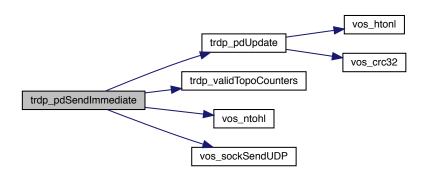
Parameters

in	appHandle	session pointer
in	pSendPD	pointer to element to be sent

Return values

TRDP_NO_ERR	no error
TRDP_IO_ERR	socket I/O error

Here is the call graph for this function:



5.26.2.13 trdp_pdSendQueued()

Send all due PD messages.

Parameters

in appHandle	session pointer
---------------------	-----------------

Return values

TRDP NO ERR	no error
	110 01101

Return values

```
TRDP_IO_ERR | socket I/O error
```

Here is the call graph for this function:



5.26.2.14 trdp_pdUpdate()

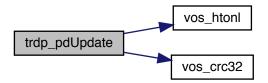
```
void trdp_pdUpdate ( {\tt PD\_ELE\_T\ *\ pPacket\ )}
```

Update the header values.

Parameters

in	pPacket	pointer to the packet to update
----	---------	---------------------------------

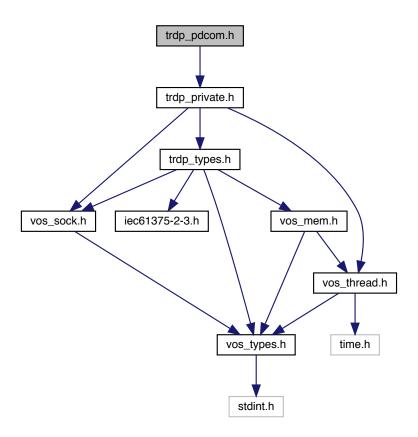
Here is the call graph for this function:



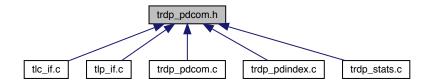
5.27 trdp_pdcom.h File Reference

Functions for PD communication.

#include "trdp_private.h"
Include dependency graph for trdp_pdcom.h:



This graph shows which files directly or indirectly include this file:



Functions

void trdp_pdInit (PD_ELE_T *, TRDP_MSG_T, UINT32 topoCount, UINT32 optopoCount, UINT32 reply
 — ComId, UINT32 replyIpAddress, UINT32 serviceId)

Initialize/construct the packet Set the header infos.

void trdp_pdUpdate (PD_ELE_T *)

Update the header values.

 TRDP_ERR_T trdp_pdPut (PD_ELE_T *, TRDP_MARSHALL_T func, void *refCon, const UINT8 *pData, UINT32 dataSize)

Copy data Update the data to be sent.

TRDP ERR T trdp pdCheck (PD HEADER T *pPacket, UINT32 packetSize, int *pIsTSN)

Check if the PD header values and the CRCs are sane.

• TRDP_ERR_T trdp_pdSend (SOCKET pdSock, PD_ELE_T *pPacket, UINT16 port)

Send one PD packet.

 TRDP_ERR_T trdp_pdGet (PD_ELE_T *pPacket, TRDP_UNMARSHALL_T unmarshall, void *refCon, const UINT8 *pData, UINT32 *pDataSize)

Copy data Set the header infos.

TRDP_ERR_T trdp_pdSendElement (TRDP_SESSION_PT appHandle, PD_ELE_T **ppElement)

Send a due PD message.

• TRDP ERR T trdp pdSendQueued (TRDP SESSION PT appHandle)

Send all due PD messages.

• TRDP_ERR_T trdp_pdSendImmediate (TRDP_SESSION_PT appHandle, PD_ELE_T *pSendPD)

Send PD message immediately.

TRDP_ERR_T trdp_pdReceive (TRDP_SESSION_PT pSessionHandle, SOCKET sock)

Receiving PD messages Read the receive socket for arriving PDs, copy the packet to a new PD_ELE_T Check for protocol errors and compare the received data to the data in our receive queue.

void trdp_pdCheckPending (TRDP_APP_SESSION_T appHandle, TRDP_FDS_T *pFileDesc, INT32 *p
 — NoDesc, int checkSending)

Check for pending packets, set FD if non blocking.

• void trdp_handleTimeout (TRDP_SESSION_PT appHandle, PD_ELE_T *plterPD)

Check a packet for timeout packet Call user's callback, if necessary.

• void trdp_pdHandleTimeOuts (TRDP_SESSION_PT appHandle)

Check for time outs.

TRDP_ERR_T trdp_pdCheckListenSocks (TRDP_SESSION_PT appHandle, TRDP_FDS_T *pRfds, INT32 *pCount)

Checking receive connection requests and data Call user's callback if needed.

• TRDP_ERR_T trdp_pdDistribute (PD_ELE_T *pSndQueue)

Distribute send time of PD packets over time.

5.27.1 Detailed Description

Functions for PD communication.

Note

Project: TCNOpen TRDP prototype stack

Author

Bernd Loehr, NewTec GmbH

Remarks

This Source Code Form is subject to the terms of the Mozilla Public License, v. 2.0. If a copy of the MPL was not distributed with this file, You can obtain one at http://mozilla.org/MPL/2.0/. Copyright Bombardier Transportation Inc. or its subsidiaries and others, 2013-2019. All rights reserved.

5.27.2 Function Documentation

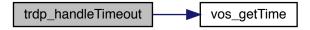
5.27.2.1 trdp_handleTimeout()

Check a packet for timeout packet Call user's callback, if necessary.

Parameters

j	in	appHandle	Session handle
j	in	pPacket	pointer to the packet element to check

Here is the call graph for this function:



5.27.2.2 trdp_pdCheck()

Check if the PD header values and the CRCs are sane.

Parameters

in	pPacket	pointer to the packet to check
in	packetSize	max size to check
out	plsTSN	set to TRUE on return if PD2 frame

Return values

TRDP_NO_ERR	
TRDP_CRC_ERR	

5.27.2.3 trdp_pdCheckListenSocks()

Checking receive connection requests and data Call user's callback if needed.

Parameters

in	appHandle	session pointer
in	pRfds	pointer to set of ready descriptors
in,out	pCount	pointer to number of ready descriptors

5.27.2.4 trdp_pdCheckPending()

Check for pending packets, set FD if non blocking.

Parameters

in	appHandle	session pointer
in,out	pFileDesc	pointer to set of ready descriptors
in,out	pNoDesc	pointer to number of ready descriptors
in	checkSend	check send queue, too

Here is the call graph for this function:



5.27.2.5 trdp_pdDistribute()

```
TRDP_ERR_T trdp_pdDistribute ( {\tt PD\_ELE\_T\ *\ pSndQueue\ )}
```

Distribute send time of PD packets over time.

The duration of PD packets on a 100MBit/s network ranges from 3us to 150us max. Because a cyclic thread scheduling below 5ms would put a too heavy load on the system, and PD packets cannot get larger than 1432 (+ UDP header), we will not account for differences in packet size. Another factor is the differences in intervals for different packets: We should only change the starting times of the packets within 1/2 the interval time. Otherwise a late addition of packets could lead to timeouts of already queued packets. Scheduling will be computed based on the smallest interval time.

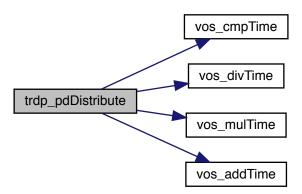
Parameters

in	pSndQueue	pointer to send queue
----	-----------	-----------------------

Return values

```
TRDP_NO_ERR
```

Here is the call graph for this function:



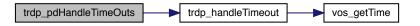
5.27.2.6 trdp_pdHandleTimeOuts()

Check for time outs.

Parameters

in appHandle	application handle
--------------	--------------------

Here is the call graph for this function:



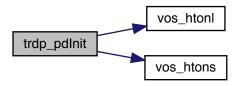
5.27.2.7 trdp_pdlnit()

Initialize/construct the packet Set the header infos.

Parameters

in	pPacket	pointer to the packet element to init
in	type	type the packet
in	etbTopoCnt	topocount to use for PD frame
in	opTrnTopoCnt	topocount to use for PD frame
in	replyComld	Pull request comId
in	replylpAddress	Pull request lp
in	serviceld	Service Id

Here is the call graph for this function:



5.27.2.8 trdp_pdPut()

Copy data Update the data to be sent.

Parameters

in	pPacket	pointer to the packet element to send
in	marshall	pointer to marshalling function
in	refCon	reference for marshalling function
in	pData	pointer to data
in	dataSize	size of data

Return values

TRDP_NO_ERR	no error other errors
-------------	-----------------------

5.27.2.9 trdp_pdReceive()

Receiving PD messages Read the receive socket for arriving PDs, copy the packet to a new PD_ELE_T Check for protocol errors and compare the received data to the data in our receive queue.

If it is a new packet, check if it is a PD Request (PULL). If it is an update, exchange the existing entry with the new one Call user's callback if needed

Parameters

in	appHandle	session pointer
in	sock	the socket to read from

Return values

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	parameter error
TRDP_WIRE_ERR	protocol error (late packet, version mismatch)
TRDP_QUEUE_ERR	not in queue
TRDP_CRC_ERR	header checksum
TRDP_TOPOCOUNT_ERR	invalid topocount

5.27.2.10 trdp_pdSend()

Send one PD packet.

Parameters

in	pdSock	socket descriptor
in	pPacket	pointer to packet to be sent
in	port	port on which to send

Return values

TRDP_NO_ERR	
TRDP_IO_ERR	

5.27.2.11 trdp_pdSendElement()

Send a due PD message.

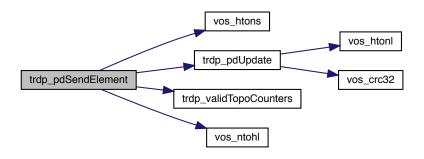
Parameters

in	appHandle	session pointer
in	ppElement	pointer to pointer of the element to send

Return values

TRDP_NO_ERR	no error
TRDP_IO_ERR	socket I/O error

Here is the call graph for this function:



5.27.2.12 trdp_pdSendImmediate()

Send PD message immediately.

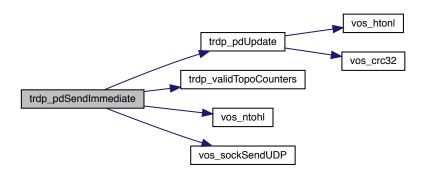
Parameters

in	appHandle	session pointer
in	pSendPD	pointer to element to be sent

Return values

TRDP_NO_ERR	no error	
TRDP_IO_ERR	socket I/O error	

Here is the call graph for this function:



5.27.2.13 trdp_pdSendQueued()

Send all due PD messages.

Parameters

in	appHandle	session pointer
----	-----------	-----------------

Return values

TRDP_NO_ERR	no error	
TRDP_IO_ERR	socket I/O error	

Here is the call graph for this function:



5.27.2.14 trdp_pdUpdate()

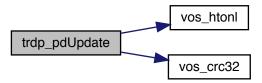
```
void trdp_pdUpdate ( {\tt PD\_ELE\_T~*~pPacket~)}
```

Update the header values.

Parameters

in	pPacket	pointer to the packet to update
----	---------	---------------------------------

Here is the call graph for this function:

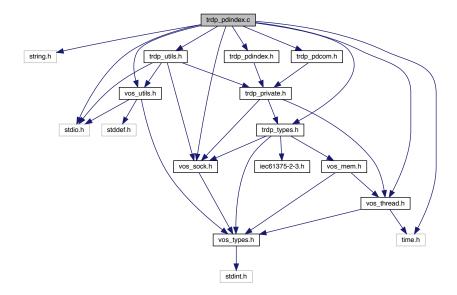


5.28 trdp_pdindex.c File Reference

Functions for indexed PD communication.

```
#include <string.h>
#include <stdio.h>
#include <time.h>
#include "trdp_types.h"
#include "trdp_utils.h"
#include "trdp_pdcom.h"
#include "vos_utils.h"
#include "vos_sock.h"
#include "vos_thread.h"
#include "trdp_pdindex.h"
```

Include dependency graph for trdp_pdindex.c:



5.28.1 Detailed Description

Functions for indexed PD communication.

Faster access to the internal process data telegram send and receive functions

Note

Project: TCNOpen TRDP prototype stack

Author

Bernd Loehr, NewTec GmbH

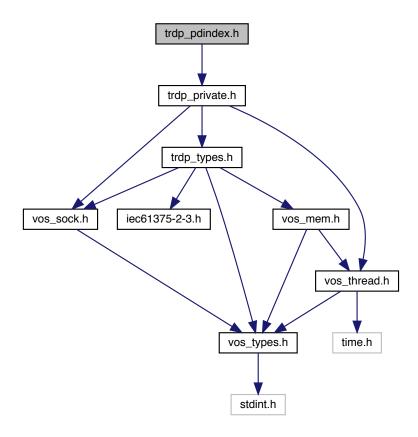
Remarks

This Source Code Form is subject to the terms of the Mozilla Public License, v. 2.0. If a copy of the MPL was not distributed with this file, You can obtain one at http://mozilla.org/MPL/2.0/. Copyright Bombardier Transportation Inc. or its subsidiaries and others, 2019. All rights reserved.

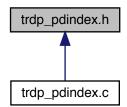
5.29 trdp_pdindex.h File Reference

Functions for indexed PD communication.

#include "trdp_private.h"
Include dependency graph for trdp_pdindex.h:



This graph shows which files directly or indirectly include this file:



Data Structures

- struct hp_slot

 Low cycle-time slots.
- struct hp_slots

 entry for the application session

Macros

#define TRDP_DEFAULT_CYCLE 1000u

Supported and recomended cycle times for the tlp_processTransmit loop.

• #define TRDP_DEFAULT_INDEX_SIZES

Default table size settings in HIGH_PERF_INDEXED Mode.

Typedefs

- typedef struct hp_slot TRDP_HP_CAT_SLOT_T Low cycle-time slots.
- typedef struct hp_slots TRDP_HP_CAT_SLOTS_T entry for the application session

5.29.1 Detailed Description

Functions for indexed PD communication.

Faster access to the internal process data telegram send and receive functions

Note

Project: TCNOpen TRDP prototype stack

Author

Bernd Loehr, NewTec GmbH

Remarks

This Source Code Form is subject to the terms of the Mozilla Public License, v. 2.0. If a copy of the MPL was not distributed with this file, You can obtain one at http://mozilla.org/MPL/2.0/. Copyright Bombardier Transportation Inc. or its subsidiaries and others, 2019. All rights reserved.

5.29.2 Macro Definition Documentation

5.29.2.1 TRDP_DEFAULT_INDEX_SIZES

```
#define TRDP_DEFAULT_INDEX_SIZES
```

Value:

```
{100, \ 200, \ 10, \ 100, \ 15, \ 200, \ 15, \ 10, \ 5, \ 10 }
```

Default table size settings in HIGH_PERF_INDEXED Mode.

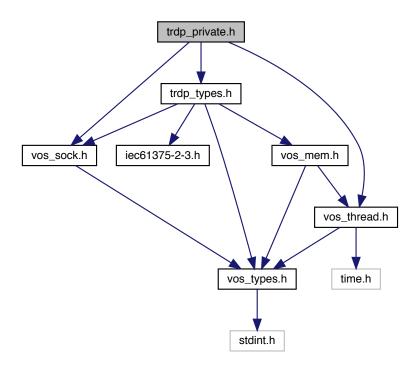
Max. number of expected publishers with intervals > 10000 ms

5.30 trdp_private.h File Reference

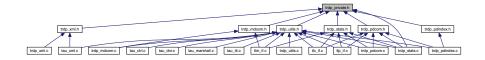
Typedefs for TRDP communication.

```
#include "trdp_types.h"
#include "vos_thread.h"
#include "vos sock.h"
```

Include dependency graph for trdp_private.h:



This graph shows which files directly or indirectly include this file:



Data Structures

• struct TRDP_HANDLE

Hidden handle definition, used as unique addressing item.

• struct TRDP_SEQ_CNT_ENTRY_T

Tuples of last received sequence counter per comld.

struct TRDP_SOCKET_TCP

TCP parameters.

struct TRDP_SOCKETS

Socket item.

struct GNU PACKED

Types for ETB control.

struct GNU PACKED

Types for ETB control.

struct GNU_PACKED

Types for ETB control.

• struct PD ELE

Queue element for PD packets to send or receive.

• struct TRDP SESSION

Session/application variables store.

Macros

• #define TRDP TIMER GRANULARITY 5000u

granularity in us - we allow 5ms now!

#define TRDP_MAX_PD_SOCKET_CNT VOS_MAX_SOCKET_CNT

Separate PD and MD socket lists.

• #define TRDP_MD_MAN_CYCLE_TIME 5000u

cycle time [us] = delay for outgoing MD

#define TRDP DEBUG DEFAULT FILE SIZE 65536u

Default maximum size of log file.

• #define TRDP_SEQ_CNT_START_ARRAY_SIZE 64u

This should be enough for the start.

• #define TRDP IF WAIT FOR READY 120u

120 seconds (120 tries each second to bind to an IP address)

• #define TRDP_PROTO_VER 0x0101u

compatible protocol version with service Id

#define TRDP_PRIV_NONE 0u

Internal flags for packets.

• #define TRDP_TIMED_OUT 0x2u

if set, inform the user

#define TRDP_INVALID_DATA 0x4u

if set, inform the user

#define TRDP_REQ_2B_SENT 0x8u

if set, the request needs to be sent

• #define TRDP REDUNDANT 0x20u

if set, packet should not be sent (redundant)

• #define TRDP_CHECK_COMID 0x40u

if set, do filter comld (addListener)

• #define TRDP_IS_TSN 0x80u

if set, PD will be sent on trdp_put() only

Typedefs

```
    typedef struct TRDP_HANDLE TRDP_ADDRESSES_T
        Hidden handle definition, used as unique addressing item.
    typedef struct TRDP_SOCKET_TCP TRDP_SOCKET_TCP_T
        TCP parameters.
    typedef struct TRDP_SOCKETS TRDP_SOCKETS_T
        Socket item.
    typedef struct PD_ELE PD_ELE_T
        Queue element for PD packets to send or receive.
    typedef struct TRDP_SESSION TRDP_SESSION_T
        Session/application variables store.
```

Enumerations

```
enum TRDP_MD_ELE_ST_T {
 TRDP_ST_NONE = 0u,
 TRDP_ST_TX_NOTIFY_ARM = 1u,
 TRDP_ST_TX_REQUEST_ARM = 2u,
 TRDP_ST_TX_REPLY_ARM = 3u,
 TRDP ST TX REPLYQUERY ARM = 4u,
 TRDP ST TX CONFIRM ARM = 5u,
 TRDP_ST_RX_READY = 6,
 TRDP_ST_TX_REQUEST_W4REPLY = 7u,
 TRDP ST RX REPLYQUERY W4C = 8u,
 TRDP ST RX REQ W4AP REPLY = 9u,
 TRDP_ST_TX_REQ_W4AP_CONFIRM = 10u,
 TRDP_ST_RX_REPLY_SENT = 11u,
 TRDP_ST_RX_NOTIFY_RECEIVED = 12u,
 TRDP_ST_TX_REPLY_RECEIVED = 13u,
 TRDP_ST_RX_CONF_RECEIVED = 14u }
    Internal MD state.
enum TRDP_SOCK_TYPE_T {
 TRDP_SOCK_INVAL = 0u,
 TRDP\_SOCK\_PD = 1u,
 TRDP\_SOCK\_MD\_UDP = 2u,
 TRDP\_SOCK\_MD\_TCP = 3u,
 TRDP_SOCK_PD_TSN = 4u }
    Socket usage.
```

5.30.1 Detailed Description

Typedefs for TRDP communication.

TRDP internal type definitions

Note

Project: TCNOpen TRDP prototype stack

Author

Bernd Loehr, NewTec GmbH

Remarks

This Source Code Form is subject to the terms of the Mozilla Public License, v. 2.0. If a copy of the MPL was not distributed with this file, You can obtain one at http://mozilla.org/MPL/2.0/. Copyright Bombardier Transportation Inc. or its subsidiaries and others, 2013-2019. All rights reserved.

5.30.2 Macro Definition Documentation

5.30.2.1 TRDP_MAX_PD_SOCKET_CNT

```
#define TRDP_MAX_PD_SOCKET_CNT VOS_MAX_SOCKET_CNT
```

Separate PD and MD socket lists.

Reserve 1/4 of sockets for MD, if supported all available sockets for PD

5.30.3 Enumeration Type Documentation

5.30.3.1 TRDP_MD_ELE_ST_T

enum TRDP_MD_ELE_ST_T

Internal MD state.

Enumerator

TRDP_ST_NONE	neutral value
TRDP_ST_TX_NOTIFY_ARM	ready to send notify MD
TRDP_ST_TX_REQUEST_ARM	ready to send request MD
TRDP_ST_TX_REPLY_ARM	ready to send reply MD
TRDP_ST_TX_REPLYQUERY_ARM	ready to send reply with confirm request MD
TRDP_ST_TX_CONFIRM_ARM	ready to send confirm MD
TRDP_ST_RX_READY	armed listener
TRDP_ST_TX_REQUEST_W4REPLY	request sent, wait for reply
TRDP_ST_RX_REPLYQUERY_W4C	reply send, with confirm request MD
TRDP_ST_RX_REQ_W4AP_REPLY	request received, wait for application reply send
TRDP_ST_TX_REQ_W4AP_CONFIRM	reply conf. rq. tx, wait for application conf send
TRDP_ST_RX_REPLY_SENT	reply sent
TRDP_ST_RX_NOTIFY_RECEIVED	notification received, wait for application to accept
TRDP_ST_TX_REPLY_RECEIVED	reply received
TRDP_ST_RX_CONF_RECEIVED	confirmation received

5.30.3.2 TRDP_SOCK_TYPE_T

enum TRDP_SOCK_TYPE_T

Socket usage.

Enumerator

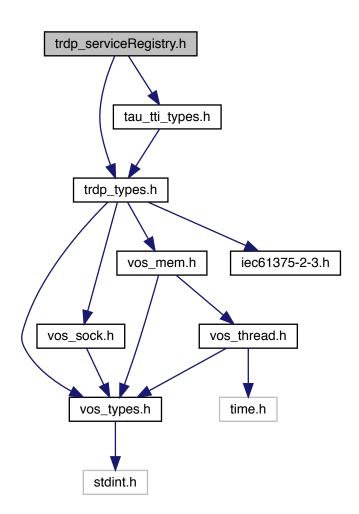
TRDP_SOCK_INVAL	Socket is undefined.
TRDP_SOCK_PD	Socket is used for UDP process data.
TRDP_SOCK_MD_UDP	Socket is used for UDP message data.
TRDP_SOCK_MD_TCP	Socket is used for TCP message data.
TRDP_SOCK_PD_TSN	Socket is used for TSN process data.

5.31 trdp_serviceRegistry.h File Reference

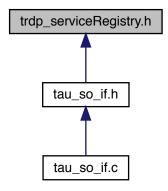
Additional definitions for IEC 61375-2-3 (Service Discovery) The definitions herein are preliminary and will change with the next major release of the IEC 61375-2-3 standard.

```
#include "trdp_types.h"
#include "tau_tti_types.h"
```

Include dependency graph for trdp_serviceRegistry.h:



This graph shows which files directly or indirectly include this file:



Data Structures

· struct service_info

Preliminary definition of a service info entry.

struct srv_info_req

Preliminary definition of a service info request.

struct GNU_PACKED

Types for ETB control.

Macros

#define SRM_SRVINFO_NOTIFY_COMID 200u

Additional defines to be reserved for SR Manager.

 #define SRM_SRVINFO_NOTIFY_URI "grpSRM.anyVeh.aCst.aClTrn.lTrn" multicast group

#define SRM_SRVINFO_NOTIFY_DS "CST_SRV_INFO"

SRM_CST_SRV_INFO_T.

• #define SRM_SRV_REQ_NOTIFY_COMID 201u

SRVINFOREQ request data:

• #define SRM_SRV_REQ_NOTIFY_URI "grpSRM.anyVeh.aCst.aClTrn.lTrn"

multicast group

• #define SRM_SRV_REQ_NOTIFY_DS "SRV_INFO_REQ"

SRM_SRV_INFO_REQ_T.

• #define SRM SERVICE READ REQ COMID 112u

Additional COMIDs to be reserved for SR Manager.

• #define SRM_SERVICE_READ_REQ_TO 3000000u

[us] 3s timeout

#define SRM_SERVICE_READ_REP_COMID 113u

MD reply

• #define SRM_SERVICE_READ_REP_DS "SRM_SERVICE_ENTRIES_T"

```
SRM_SERVICE_ENTRIES_T.

    #define SRM_SERVICE_READ_REP_DSID SRM_SERVICE_DSID

     SRM_SERVICE_ENTRIES_T.

    #define SRM SERVICE ADD REQ COMID 114u

     SRM manager telegram MD: Add service instance(s) to the Service Registry.

    #define SRM_SERVICE_ADD_REQ_TO 3000000u

     [us] 3s timeout

    #define SRM SERVICE ADD REQ DS "SRM SERVICE ENTRIES T"

     SRM SERVICE ENTRIES T.

    #define SRM_SERVICE_ADD_REQ_DSID SRM_SERVICE_DSID

     SRM_SERVICE_ENTRIES_T.

    #define SRM_SERVICE_ADD_REP_COMID 115u

     Reply returns instanceld.

    #define SRM SERVICE ADD REP DSID SRM SERVICE DSID

     SRM_SERVICE_ENTRIES_T.

    #define SRM_SERVICE_UPD_NOTIFY_COMID 116u

     SRM manager telegram MD: Update service instance(s) to the Service Registry.

    #define SRM_SERVICE_UPD_NOTIFY_TTL 3000000u

     [us] default time-to-live

    #define SRM_SERVICE_UPD_NOTIFY_DS "SRM_SERVICE_ENTRIES_T"

     SRM_SERVICE_ENTRIES_T.

    #define SRM SERVICE UPD NOTIFY DSID SRM SERVICE DSID

     SRM SERVICE ENTRIES T.

    #define SRM_SERVICE_DEL_REQ_COMID 117u

     SRM manager telegram MD: Remove Service instance(s) from the Service Registry.
• #define SRM SERVICE DEL REQ TO 3000000u
     [us] 3s timeout

    #define SRM SERVICE DEL REQ DS "SRM SERVICE ENTRIES T"

     SRM_SERVICE_ENTRIES_T.

    #define SRM SERVICE DEL REQ DSID SRM SERVICE DSID

     SRM_SERVICE_ENTRIES_T.

    #define SRM_SERVICE_DEL_REP_COMID 118u

     MD reply OK or not.

    #define SOA_SERVICEID(instld, typeId) ((instld) << 24 | (typeId))</li>

     serviceld from instanceld and typeld

    #define SOA TYPE(serviceId) ((serviceId) & 0xFFFFFF)

     return 24 Bit service type part of serviceld

    #define SOA_INST(serviceId) (((serviceId) >> 24) & 0xFF)

     return 8 Bit instance ID part of serviceId

    #define SOA_SAME_SERVICEID_OR0(a, b) (((a) == 0u) || ((a) == (b)))

     return TRUE if serviceId(a) is 0 or equals the second serviceId (b)
• #define SOA_SAME_SERVICEID(a, b) ((a) == (b))
     return TRUE if servicelds (incl.

    #define SOA_SAME_SERVICE_TYPE(a, b) (SOA_TYPE(a) == SOA_TYPE(b))

     return TRUE if service types match
```

Typedefs

typedef struct service info SRM SERVICE INFO T

Preliminary definition of a service info entry.

typedef struct srv_info_req SRM_SRV_INFO_REQ_T

Preliminary definition of a service info request.

5.31.1 Detailed Description

Additional definitions for IEC 61375-2-3 (Service Discovery) The definitions herein are preliminary and will change with the next major release of the IEC 61375-2-3 standard.

Note

Project: CTA2 WP3 / TCNOpen TRDP

Author

Bernd Loehr, NewTec GmbH, 2019-04-08

Remarks

Copyright 2019 Bombardier Transportation & NewTec GmbH

5.31.2 Macro Definition Documentation

5.31.2.1 SOA_SAME_SERVICEID

return TRUE if servicelds (incl.

instance) match

5.31.2.2 SRM_SERVICE_READ_REQ_COMID

```
#define SRM_SERVICE_READ_REQ_COMID 112u
```

Additional COMIDs to be reserved for SR Manager.

Transport: MD over TCP preferred for reliability SRM manager telegram MD: Read Services from the Consist-local Service Registry

5.31.2.3 SRM_SRVINFO_NOTIFY_COMID

```
#define SRM_SRVINFO_NOTIFY_COMID 200u
```

Additional defines to be reserved for SR Manager.

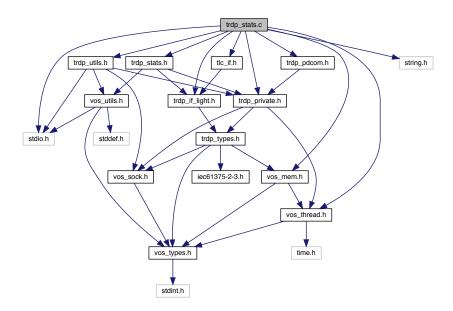
Transport: Trainwide MD over UDP / Multicast SRVINFO notification data:

5.32 trdp_stats.c File Reference

Statistics functions for TRDP communication.

```
#include <stdio.h>
#include <string.h>
#include "trdp_stats.h"
#include "trdp_if_light.h"
#include "tlc_if.h"
#include "trdp_private.h"
#include "trdp_pdcom.h"
#include "trdp_utils.h"
#include "vos_mem.h"
#include "vos_thread.h"
```

Include dependency graph for trdp_stats.c:



Functions

void trdp_UpdateStats (TRDP_APP_SESSION_T appHandle)

Update the statistics.

void trdp_initStats (TRDP_APP_SESSION_T appHandle)

Init statistics.

EXT_DECL TRDP_ERR_T tlc_resetStatistics (TRDP_APP_SESSION_T appHandle)
 Reset statistics.

• EXT_DECL TRDP_ERR_T tlc_getStatistics (TRDP_APP_SESSION_T appHandle, TRDP_STATISTICS_T *pStatistics)

Return statistics.

• EXT_DECL TRDP_ERR_T tlc_getSubsStatistics (TRDP_APP_SESSION_T appHandle, UINT16 *pNum← Subs, TRDP_SUBS_STATISTICS_T *pStatistics)

Return PD subscription statistics.

• EXT_DECL TRDP_ERR_T tlc_getPubStatistics (TRDP_APP_SESSION_T appHandle, UINT16 *pNumPub, TRDP_PUB_STATISTICS_T *pStatistics)

Return PD publish statistics.

• EXT_DECL TRDP_ERR_T tlc_getRedStatistics (TRDP_APP_SESSION_T appHandle, UINT16 *pNumRed, TRDP_RED_STATISTICS_T *pStatistics)

Return redundancy group statistics.

 EXT_DECL TRDP_ERR_T tlc_getJoinStatistics (TRDP_APP_SESSION_T appHandle, UINT16 *pNumJoin, UINT32 *plpAddr)

Return join statistics.

• void trdp_pdPrepareStats (TRDP_APP_SESSION_T appHandle, PD_ELE_T *pPacket) Fill the statistics packet.

5.32.1 Detailed Description

Statistics functions for TRDP communication.

Note

Project: TCNOpen TRDP prototype stack

Author

Bernd Loehr, NewTec GmbH

Remarks

This Source Code Form is subject to the terms of the Mozilla Public License, v. 2.0. If a copy of the MPL was not distributed with this file, You can obtain one at http://mozilla.org/MPL/2.0/. Copyright Bombardier Transportation Inc. or its subsidiaries and others, 2013. All rights reserved.

5.32.2 Function Documentation

5.32.2.1 tlc_getJoinStatistics()

Return join statistics.

Memory for statistics information must be provided by the user.

Parameters

in	appHandle	the handle returned by tlc_openSession
in,out	pNumJoin	Pointer to the number of joined IP Adresses
out	plpAddr	Pointer to a list with the joined IP adresses

Return values

TRDP_NO_ERR	no error
TRDP_NOINIT_ERR	handle invalid
TRDP_PARAM_ERR	parameter error
TRDP_MEM_ERR	there are more items than requested

5.32.2.2 tlc_getPubStatistics()

Return PD publish statistics.

Memory for statistics information must be provided by the user.

Parameters

in	appHandle	the handle returned by tlc_openSession	
in,out	pNumPub	Pointer to the number of publishers	
out	pStatistics	Pointer to a list with the publish statistics information	

Return values

TRDP_NO_ERR	no error
TRDP_NOINIT_ERR	handle invalid
TRDP_PARAM_ERR	parameter error
TRDP_MEM_ERR	there are more subscriptions than requested

5.32.2.3 tlc_getRedStatistics()

Return redundancy group statistics.

Memory for statistics information must be provided by the user.

Parameters

in	appHandle the handle returned by tlc_openSession	
in,out	pNumRed	Pointer to the number of redundancy groups
out	pStatistics	Pointer to a list with the redundancy group information

Return values

TRDP_NO_ERR	no error
TRDP_NOINIT_ERR	handle invalid
TRDP_PARAM_ERR	parameter error
TRDP_MEM_ERR	there are more subscriptions than requested

5.32.2.4 tlc_getStatistics()

Return statistics.

Memory for statistics information must be provided by the user.

Parameters

in	appHandle	the handle returned by tlc_openSession
out	pStatistics	Pointer to statistics for this application session

Return values

TRDP_NO_ERR	no error
TRDP_NOINIT_ERR	handle invalid
TRDP_PARAM_ERR	parameter error

5.32.2.5 tlc_getSubsStatistics()

Return PD subscription statistics.

Memory for statistics information must be provided by the user.

Parameters

in	appHandle	the handle returned by tlc_openSession	
in,out	pNumSubs	In: The number of subscriptions requested Out: Number of subscriptions returned	
in, out	pStatistics	Statistics Pointer to an array with the subscription statistics information	

Return values

TRDP_NO_ERR	no error
TRDP_NOINIT_ERR	handle invalid
TRDP_PARAM_ERR	parameter error
TRDP_MEM_ERR	there are more subscriptions than requested

5.32.2.6 tlc_resetStatistics()

Reset statistics.

Parameters

in	appHandle	the handle returned by tlc_openSession
----	-----------	--

Return values

TRDP_NO_ERR	no error
TRDP_NOINIT_ERR	handle invalid
TRDP_PARAM_ERR	parameter error

5.32.2.7 trdp_initStats()

Init statistics.

Clear the stats structure for a session.

Parameters

in	appHandle	the handle returned by tlc_openSession
----	-----------	--

< host name

< leader host name Here is the call graph for this function:



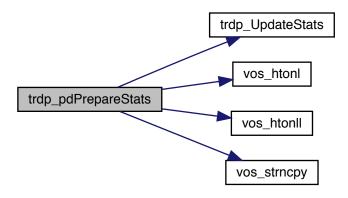
5.32.2.8 trdp_pdPrepareStats()

Fill the statistics packet.

Parameters

in	appHandle	the handle returned by tlc_openSession
in,out	pPacket	pointer to the packet to fill

Here is the call graph for this function:



5.32.2.9 trdp_UpdateStats()

Update the statistics.

Parameters

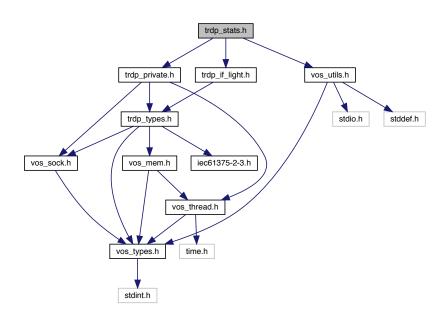
	in	appHandle	the handle returned by tlc_openSession
--	----	-----------	--

5.33 trdp_stats.h File Reference

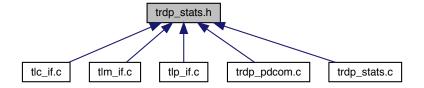
Statistics for TRDP communication.

```
#include "trdp_if_light.h"
#include "trdp_private.h"
#include "vos_utils.h"
```

Include dependency graph for trdp_stats.h:



This graph shows which files directly or indirectly include this file:



Functions

- void trdp_initStats (TRDP_APP_SESSION_T appHandle)
 Init statistics.
- void trdp_pdPrepareStats (TRDP_APP_SESSION_T appHandle, PD_ELE_T *pPacket) Fill the statistics packet.

5.33.1 Detailed Description

Statistics for TRDP communication.

Note

Project: TCNOpen TRDP prototype stack

Author

Bernd Loehr, NewTec GmbH

Remarks

This Source Code Form is subject to the terms of the Mozilla Public License, v. 2.0. If a copy of the MPL was not distributed with this file, You can obtain one at http://mozilla.org/MPL/2.0/. Copyright Bombardier Transportation Inc. or its subsidiaries and others, 2013. All rights reserved.

5.33.2 Function Documentation

5.33.2.1 trdp_initStats()

Init statistics.

Clear the stats structure for a session.

Parameters

in	appHandle	the handle returned by tlc_openSession]
----	-----------	--	---

- < host name
- < leader host name Here is the call graph for this function:



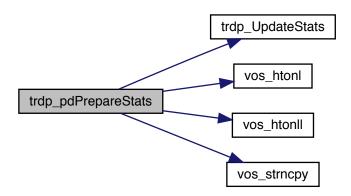
5.33.2.2 trdp_pdPrepareStats()

Fill the statistics packet.

Parameters

in	appHandle	the handle returned by tlc_openSession
in,out	pPacket	pointer to the packet to fill

Here is the call graph for this function:



5.34 trdp_tsn_def.h File Reference

Additional definitions for TSN.

Macros

• #define TRDP_MD_DEFAULT_QOS 2u

matching new proposed priority classes

• #define TRDP_PD_DEFAULT_QOS 2u

Default PD communication parameters.

• #define TRDP_PD_DEFAULT_TSN_PRIORITY 3u

matching new proposed priority classes

#define TRDP PD DEFAULT TSN FALSE

matching new proposed priority classes

• #define TRDP_MIN_PD2_HEADER_SIZE sizeof(PD2_HEADER_T)

PD packet properties.

#define TRDP_MAX_PD2_DATA_SIZE 1458u

PD2 data.

• #define TRDP MSG TSN PD 0x01u

New Message Types for TRDP Version 2 TSN-PDU (preliminary)

#define TRDP_MSG_TSN_PD_SDT 0x02u

TSN safe PD Data.

#define TRDP_MSG_TSN_PD_MSDT 0x03u

TSN multiple SDT PD Data.

#define TRDP_MSG_TSN_PD_RES 0x04u

TSN reserved.

• #define TRDP_VER_TSN_PROTO 0x02u

Protocol version for TSN.

5.34.1 Detailed Description

Additional definitions for TSN.

This header file defines proposed extensions and additions to IEC61375-2-3:2017 The definitions herein are preliminary and may change with the next major release of the IEC 61375-2-3 standard.

Note

Project: TCNOpen TRDP prototype stack & FDF/DbD

Author

Bernd Loehr, NewTec GmbH, 2019-02-19

Remarks

Copyright 2019, NewTec GmbH

ld

trdp_tsn_def.h 1932 2019-07-03 15:31:16Z bloehr

5.34.2 Macro Definition Documentation

```
5.34.2.1 TRDP_MIN_PD2_HEADER_SIZE
```

```
#define TRDP_MIN_PD2_HEADER_SIZE sizeof(PD2_HEADER_T)
```

PD packet properties.

TSN header size with FCS

5.34.2.2 TRDP_MSG_TSN_PD

```
#define TRDP_MSG_TSN_PD 0x01u
```

New Message Types for TRDP Version 2 TSN-PDU (preliminary)

TSN non safe PD Data

5.34.2.3 TRDP_PD_DEFAULT_QOS

```
#define TRDP_PD_DEFAULT_QOS 2u
```

Default PD communication parameters.

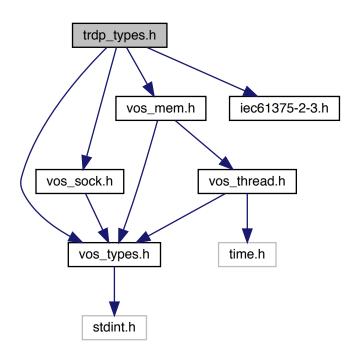
matching new proposed priority classes

5.35 trdp_types.h File Reference

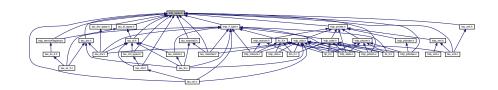
Typedefs for TRDP communication.

```
#include "vos_types.h"
#include "vos_mem.h"
#include "vos_sock.h"
```

#include "iec61375-2-3.h"
Include dependency graph for trdp_types.h:



This graph shows which files directly or indirectly include this file:



Data Structures

• struct TRDP_PD_INFO_T

Process data info from received telegram; allows the application to generate responses.

struct TRDP_MD_INFO_T

 ${\it Message \ data \ info \ from \ received \ telegram; allows \ the \ application \ to \ generate \ responses.}$

• struct TRDP_COM_PARAM_T

Quality/type of service, time to live , no.

struct TRDP_DATASET_ELEMENT_T

Dataset element definition.

struct TRDP_DATASET

Dataset definition.

• struct TRDP_COMID_DSID_MAP_T

Comld - data set mapping element definition.

struct GNU_PACKED

Types for ETB control.

struct GNU PACKED

Types for ETB control.

struct GNU_PACKED

Types for ETB control.

struct GNU PACKED

Types for ETB control.

struct GNU_PACKED

Types for ETB control.

struct GNU_PACKED

Types for ETB control.

struct GNU_PACKED

Types for ETB control.

• struct GNU_PACKED

Types for ETB control.

• struct GNU_PACKED

Types for ETB control.

struct TRDP_MARSHALL_CONFIG_T

Marshaling/unmarshalling configuration.

• struct TRDP PD CONFIG T

Default PD configuration.

struct TRDP_MD_CONFIG_T

Default MD configuration.

struct TRDP_MEM_CONFIG_T

Enumeration type for memory pre-fragmentation, reuse of VOS definition.

struct TRDP_PROCESS_CONFIG_T

Various flags/general TRDP options for library initialization.

struct TRDP_IDX_TABLE_T

Settings for pre-allocation of index tables for application session initialization.

Macros

• #define USE_HEAP 0

If this is set, we can allocate dynamically memory.

• #define TRDP FLAGS DEFAULT 0u

Various flags for PD and MD packets.

#define TRDP_FLAGS_NONE 0x01u

No flags set.

#define TRDP FLAGS MARSHALL 0x02u

Optional marshalling/unmarshalling in TRDP stack.

#define TRDP_FLAGS_CALLBACK 0x04u

Use of callback function.

#define TRDP_FLAGS_TCP 0x08u

Use TCP for message data.

• #define TRDP_FLAGS_FORCE_CB 0x10u

Force a callback for every received packet.

#define TRDP_FLAGS_TSN 0x20u

Hard Real Time PD.

 #define TRDP_FLAGS_TSN_SDT 0x40u SDT PD.

• #define TRDP FLAGS TSN MSDT 0x80u

Multi SDT PD.

#define TRDP_INFINITE_TIMEOUT 0xfffffffu

Infinite reply timeout.

• #define TRDP DEFAULT PD TIMEOUT 100000u

Default PD timeout 100ms from 61375-2-3 Table C.7.

#define TRDP BOOL8 TRDP BITSET8

1 bit relevant (equal to zero = false, not equal to zero = true)

#define TRDP ANTIVALENT8 TRDP BITSET8

2 bit relevant (0x0 = errror, 0x01 = false, 0x02 = true, 0x03 undefined)

#define TRDP_OPTION_NONE 0u

Various flags/general TRDP options for library initialization.

#define TRDP OPTION BLOCK 0x01u

Default: Use nonblocking I/O calls, polling necessary Set: Read calls will block, use select()

#define TRDP OPTION TRAFFIC SHAPING 0x02u

Use traffic shaping - distribute packet sending Default: OFF.

• #define TRDP OPTION NO REUSE ADDR 0x04u

Do not allow re-use of address/port (-> no multihoming) Default: Allow.

#define TRDP_OPTION_NO_MC_LOOP_BACK 0x08u

Do not allow loop back of multicast traffic Default: Allow.

#define TRDP_OPTION_NO_UDP_CHK 0x10u

Suppress UDP CRC generation Default: Compute UDP CRC.

• #define TRDP_OPTION_WAIT_FOR_DNR 0x20u

Wait for DNR Default: Don't wait.

#define TRDP OPTION NO PD STATS 0x40u

Suppress PD statistics \ Default: Don't suppress.

#define TRDP_OPTION_DEFAULT_CONFIG 0x80u

no XML process config, defaults were used

Typedefs

typedef VOS IP4 ADDR T TRDP IP ADDR T

TRDP general type definitions.

typedef CHAR8 TRDP_NET_LABEL_T[TRDP_MAX_LABEL_LEN]

Definition for usage in network packets, not necessarily \0 terminated!

typedef VOS_VERSION_T TRDP_VERSION_T

Version information.

typedef VOS TIMEVAL T TRDP TIME T

Timer value compatible with timeval / select.

typedef VOS_FDS_T TRDP_FDS_T

File descriptor set compatible with fd_set / select.

typedef VOS_UUID_T TRDP_UUID_T

UUID definition reuses the VOS definition.

typedef struct TRDP_DATASET_T

Dataset definition.

typedef TRDP DATASET T * pTRDP DATASET T

Array of pointers to dataset.

typedef VOS_PRINT_DBG_T TRDP_PRINT_DBG_T

TRDP configuration type definitions.

typedef VOS_LOG_T TRDP_LOG_T

Categories for logging, reuse of the VOS definition.

typedef TRDP_ERR_T(* TRDP_MARSHALL_T) (void *pRefCon, UINT32 comId, UINT8 *pSrc, UINT32 srcSize, UINT8 *pDst, UINT32 *pDstSize, TRDP_DATASET_T **ppCachedDS)

Function type for marshalling .

typedef TRDP_ERR_T(* TRDP_UNMARSHALL_T) (void *pRefCon, UINT32 comId, UINT8 *pSrc, UINT32 srcSize, UINT8 *pDst, UINT32 *pDstSize, TRDP_DATASET_T **ppCachedDS)

Function type for unmarshalling.

 typedef void(* TRDP_PD_CALLBACK_T) (void *pRefCon, TRDP_APP_SESSION_T appHandle, const T← RDP_PD_INFO_T *pMsg, UINT8 *pData, UINT32 dataSize)

Callback for receiving indications, timeouts, releases, responses.

 typedef void(* TRDP_MD_CALLBACK_T) (void *pRefCon, TRDP_APP_SESSION_T appHandle, const T← RDP_MD_INFO_T *pMsg, UINT8 *pData, UINT32 dataSize)

Callback for receiving indications, timeouts, releases, responses.

Enumerations

```
• enum TRDP ERR T {
 TRDP NO ERR = 0,
 TRDP PARAM ERR = -1,
 TRDP INIT ERR = -2,
 TRDP_NOINIT_ERR = -3,
 TRDP_TIMEOUT_ERR = -4,
 TRDP_NODATA_ERR = -5,
 TRDP\_SOCK\_ERR = -6,
 TRDP_IO_ERR = -7,
 TRDP MEM ERR = -8,
 TRDP SEMA ERR = -9,
 TRDP QUEUE ERR = -10,
 TRDP_QUEUE_FULL_ERR = -11,
 TRDP MUTEX ERR = -12,
 TRDP THREAD ERR = -13,
 TRDP\_BLOCK\_ERR = -14,
 TRDP_INTEGRATION_ERR = -15,
 TRDP NOCONN ERR = -16,
 TRDP NOSESSION ERR = -30,
 TRDP SESSION ABORT ERR = -31,
 TRDP NOSUB ERR = -32,
 TRDP NOPUB ERR = -33,
 TRDP NOLIST ERR = -34,
 TRDP\_CRC\_ERR = -35,
 TRDP_WIRE_ERR = -36,
 TRDP\_TOPO\_ERR = -37,
 TRDP COMID ERR = -38,
 TRDP_STATE_ERR = -39,
 TRDP_APP_TIMEOUT_ERR = -40,
 TRDP APP REPLYTO ERR = -41,
 TRDP APP CONFIRMTO ERR = -42,
 TRDP REPLYTO ERR = -43,
 TRDP CONFIRMTO ERR = -44,
 TRDP REQCONFIRMTO ERR = -45,
 TRDP PACKET ERR = -46,
 TRDP_UNRESOLVED_ERR = -47,
 TRDP_XML_PARSER_ERR = -48,
```

```
TRDP_INUSE_ERR = -49,
 TRDP MARSHALLING ERR = -50,
 TRDP_UNKNOWN_ERR = -99 }
    Return codes for all API functions, -1..-29 taken over from vos.
• enum TRDP_REPLY_STATUS_T
    TRDP data transfer type definitions.
enum TRDP_RED_STATE_T {
 TRDP_RED_FOLLOWER = 0u,
 TRDP RED LEADER = 1u }
    Redundancy states.
• enum TRDP_TO_BEHAVIOR_T {
 TRDP_TO_DEFAULT = 0u,
 TRDP TO SET TO ZERO = 1u,
 TRDP_TO_KEEP_LAST_VALUE = 2u }
    How invalid PD shall be handled.
• enum TRDP DATA TYPE T {
 TRDP_INVALID = 0u,
 TRDP_BITSET8 = 1u,
 TRDP_CHAR8 = 2u,
 TRDP\_UTF16 = 3u,
 TRDP_INT8 = 4u,
 TRDP_INT16 = 5u,
 TRDP INT32 = 6u,
 TRDP INT64 = 7u,
 TRDP UINT8 = 8u,
 TRDP UINT16 = 9u,
 TRDP_UINT32 = 10u,
 TRDP_UINT64 = 11u,
 TRDP_REAL32 = 12u,
 TRDP_REAL64 = 13u,
 TRDP TIMEDATE32 = 14u,
 TRDP_TIMEDATE48 = 15u,
 TRDP_TIMEDATE64 = 16u,
 TRDP_TYPE_MAX = 30u }
    TRDP dataset description definitions.
```

The databat decomposition deministrations

5.35.1 Detailed Description

Note

Project: TCNOpen TRDP prototype stack

Author

Bernd Loehr, NewTec GmbH

Remarks

This Source Code Form is subject to the terms of the Mozilla Public License, v. 2.0. If a copy of the MPL was not distributed with this file, You can obtain one at http://mozilla.org/MPL/2.0/. Copyright Bombardier Transportation Inc. or its subsidiaries and others, 2015-2019. All rights reserved.

5.35.2 Macro Definition Documentation

5.35.2.1 TRDP_FLAGS_DEFAULT

```
#define TRDP_FLAGS_DEFAULT Ou
```

Various flags for PD and MD packets.

Default value defined in tlc_openDession will be taken

5.35.3 Typedef Documentation

5.35.3.1 TRDP_IP_ADDR_T

```
typedef VOS_IP4_ADDR_T TRDP_IP_ADDR_T
```

TRDP general type definitions.

5.35.3.2 TRDP_MARSHALL_T

```
typedef TRDP_ERR_T(* TRDP_MARSHALL_T) (void *pRefCon, UINT32 comId, UINT8 *pSrc, UINT32 src← Size, UINT8 *pDst, UINT32 *pDstSize, TRDP_DATASET_T **ppCachedDS)
```

Function type for marshalling .

The function must know about the dataset's alignment etc.

Parameters

in	pRefCon	pointer to user context
in	comId	ComId to identify the structure out of a configuration
in	pSrc	pointer to received original message
in	srcSize	size of the source buffer
in	pDst	pointer to a buffer for the treated message
in,out	pDstSize	size of the provide buffer / size of the treated message
in,out	ppCachedDS	pointer to pointer of cached dataset

Return values

TRDP_NO_ERR	no error
TRDP_MEM_ERR	provided buffer to small
TRDP_COMID_ERR	comid not existing

5.35.3.3 TRDP_MD_CALLBACK_T

typedef void(* TRDP_MD_CALLBACK_T) (void *pRefCon, TRDP_APP_SESSION_T appHandle, const TRDP_M← D_INFO_T *pMsg, UINT8 *pData, UINT32 dataSize)

Callback for receiving indications, timeouts, releases, responses.

Parameters

in	appHandle	handle returned also by tlc_init
in	pRefCon	pointer to user context
in	pMsg	pointer to received message information
in	pData	pointer to received data
in	dataSize	size of received data pointer to received data

5.35.3.4 TRDP_PD_CALLBACK_T

typedef void(* TRDP_PD_CALLBACK_T) (void *pRefCon, TRDP_APP_SESSION_T appHandle, const TRDP_P \leftarrow D_INFO_T *pMsg, UINT8 *pData, UINT32 dataSize)

Callback for receiving indications, timeouts, releases, responses.

Parameters

in	pRefCon	pointer to user context
in	appHandle	application handle returned by tlc_openSession
in	pMsg	pointer to received message information
in	pData	pointer to received data
in	dataSize	size of received data pointer to received data

5.35.3.5 TRDP_PRINT_DBG_T

typedef VOS_PRINT_DBG_T TRDP_PRINT_DBG_T

TRDP configuration type definitions.

Callback function definition for error/debug output, reuse of the VOS defined function.

5.35.3.6 TRDP_TIME_T

typedef VOS_TIMEVAL_T TRDP_TIME_T

Timer value compatible with timeval / select.

Relative or absolute date, depending on usage

5.35.3.7 TRDP_UNMARSHALL_T

typedef TRDP_ERR_T(* TRDP_UNMARSHALL_T) (void *pRefCon, UINT32 comId, UINT8 *pSrc, UINT32 src \leftarrow Size, UINT8 *pDst, UINT32 *pDstSize, TRDP_DATASET_T **ppCachedDS)

Function type for unmarshalling.

The function must know about the dataset's alignment etc.

Parameters

in	pRefCon	pointer to user context
in	comId	Comld to identify the structure out of a configuration
in	pSrc	pointer to received original message
in	srcSize	data length from TRDP packet header
in	pDst	pointer to a buffer for the treated message
in,out	pDstSize	size of the provide buffer / size of the treated message
in,out	ppCachedDS	pointer to pointer of cached dataset

Return values

TRDP_NO_ERR	no error
TRDP_MEM_ERR	provide buffer to small
TRDP_COMID_ERR	comid not existing

5.35.4 Enumeration Type Documentation

5.35.4.1 TRDP_DATA_TYPE_T

enum TRDP_DATA_TYPE_T

TRDP dataset description definitions.

Dataset element definition

Enumerator

TRDP_INVALID	Invalid/unknown.
TRDP_BITSET8	=UINT8
TRDP_CHAR8	char, can be used also as UTF8
TRDP_UTF16	Unicode UTF-16 character.
TRDP_INT8	Signed integer, 8 bit.
TRDP_INT16	Signed integer, 16 bit.
TRDP_INT32	Signed integer, 32 bit.
TRDP_INT64	Signed integer, 64 bit.
TRDP_UINT8	Unsigned integer, 8 bit.
TRDP_UINT16	Unsigned integer, 16 bit.

Enumerator

TRDP_UINT32	Unsigned integer, 32 bit.
TRDP_UINT64	Unsigned integer, 64 bit.
TRDP_REAL32	Floating point real, 32 bit.
TRDP_REAL64	Floating point real, 64 bit.
TRDP_TIMEDATE32	32 bit UNIX time
TRDP_TIMEDATE48	48 bit TCN time (32 bit UNIX time and 16 bit ticks)
TRDP_TIMEDATE64	32 bit UNIX time + 32 bit microseconds
TRDP_TYPE_MAX	Values greater are considered nested datasets.

5.35.4.2 TRDP_ERR_T

enum TRDP_ERR_T

Return codes for all API functions, -1..-29 taken over from vos.

Enumerator

Enumerator	
TRDP_NO_ERR	No error.
TRDP_PARAM_ERR	Parameter missing or out of range.
TRDP_INIT_ERR	Call without valid initialization.
TRDP_NOINIT_ERR	Call with invalid handle.
TRDP_TIMEOUT_ERR	Timout.
TRDP_NODATA_ERR	Non blocking mode: no data received.
TRDP_SOCK_ERR	Socket error / option not supported.
TRDP_IO_ERR	Socket IO error, data can't be received/sent.
TRDP_MEM_ERR	No more memory available.
TRDP_SEMA_ERR	Semaphore not available.
TRDP_QUEUE_ERR	Queue empty.
TRDP_QUEUE_FULL_ERR	Queue full.
TRDP_MUTEX_ERR	Mutex not available.
TRDP_THREAD_ERR	Thread error.
TRDP_BLOCK_ERR	System call would have blocked in blocking mode.
TRDP_INTEGRATION_ERR	Alignment or endianess for selected target wrong.
TRDP_NOCONN_ERR	No TCP connection.
TRDP_NOSESSION_ERR	No such session.
TRDP_SESSION_ABORT_ERR	Session aborted.
TRDP_NOSUB_ERR	No subscriber.
TRDP_NOPUB_ERR	No publisher.
TRDP_NOLIST_ERR	No listener.
TRDP_CRC_ERR	Wrong CRC.
TRDP_WIRE_ERR	Wire.
TRDP_TOPO_ERR	Invalid topo count.
TRDP_COMID_ERR	Unknown Comld.
TRDP_STATE_ERR	Call in wrong state.
TRDP_APP_TIMEOUT_ERR	Application Timeout.
TRDP_APP_REPLYTO_ERR	Application Reply Sent Timeout.

Enumerator

TRDP_APP_CONFIRMTO_ERR	Application Confirm Sent Timeout.
TRDP_REPLYTO_ERR	Protocol Reply Timeout.
TRDP_CONFIRMTO_ERR	Protocol Confirm Timeout.
TRDP_REQCONFIRMTO_ERR	Protocol Confirm Timeout (Request sender)
TRDP_PACKET_ERR	Incomplete message data packet.
TRDP_UNRESOLVED_ERR	DNR: address could not be resolved.
TRDP_XML_PARSER_ERR	Returned by the tau_xml subsystem.
TRDP_INUSE_ERR	Resource is still in use.
TRDP_MARSHALLING_ERR	Source size exceeded, dataset mismatch.
TRDP_UNKNOWN_ERR	Unspecified error.

5.35.4.3 TRDP_RED_STATE_T

enum TRDP_RED_STATE_T

Redundancy states.

Enumerator

TRDP_RED_FOLLOWER	Redundancy follower - redundant PD will be not sent out.
TRDP_RED_LEADER	Redundancy leader - redundant PD will be sent out.

5.35.4.4 TRDP_REPLY_STATUS_T

enum TRDP_REPLY_STATUS_T

TRDP data transfer type definitions.

Reply status messages

5.35.4.5 TRDP_TO_BEHAVIOR_T

enum TRDP_TO_BEHAVIOR_T

How invalid PD shall be handled.

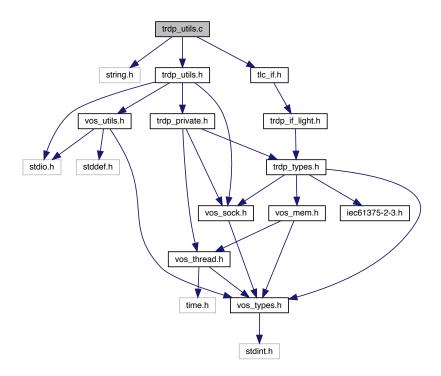
Enumerator

TRDP TO DEFAULT	Default value defined in tlc openDession will be taken.
TRDP_TO_SET_TO_ZERO	If set, data will be reset to zero on time out.
TRDP_TO_KEEP_LAST_VALUE	If set, last received values will be returned.

5.36 trdp_utils.c File Reference

Helper functions for TRDP communication.

```
#include <string.h>
#include "tlc_if.h"
#include "trdp_utils.h"
Include dependency graph for trdp_utils.c:
```



Functions

void printSocketUsage (TRDP_SOCKETS_T iface[])

Debug socket usage output.

BOOL8 trdp_SockIsJoined (const TRDP_IP_ADDR_T mcList[VOS_MAX_MULTICAST_CNT], TRDP_IP_

ADDR_T mcGroup)

Check if a mc group is in the list.

Add mc group to the list.

BOOL8 trdp_SockDelJoin (TRDP_IP_ADDR_T mcList[VOS_MAX_MULTICAST_CNT], TRDP_IP_ADDR_T mcGroup)

remove mc group from the list

- TRDP_IP_ADDR_T trdp_findMCjoins (TRDP_APP_SESSION_T appHandle, TRDP_IP_ADDR_T mcGroup)

 Check an MC group not used by other sockets / subscribers/ listeners.
- UINT32 trdp packetSizePD (UINT32 dataSize)

Get the packet size from the raw data size.

UINT32 trdp_packetSizeMD (UINT32 dataSize)

Get the packet size from the raw data size.

PD_ELE_T * trdp_queueFindComId (PD_ELE_T *pHead, UINT32 comId)

Return the element with same comld.

PD_ELE_T * trdp_queueFindPubAddr (PD_ELE_T *pHead, TRDP_ADDRESSES_T *addr)

Return the element with same comld, serviceld and IP addresses.

PD_ELE_T * trdp_queueFindSubAddr (PD_ELE_T *pHead, TRDP_ADDRESSES_T *addr)

Return the element with same comld and IP addresses.

PD_ELE_T * trdp_findSubAddr (PD_ELE_T *pHead, TRDP_ADDRESSES_T *addr, UINT32 comld)

Return the element with same comld and IP addresses.

PD_ELE_T * trdp_queueFindExistingSub (PD_ELE_T *pHead, TRDP_ADDRESSES_T *addr)

Return the element with same comld and IP addresses.

void trdp_queueDelElement (PD_ELE_T **ppHead, PD_ELE_T *pDelete)

Delete an element.

Check topography counters The applied conformance pattern follows Table A.5/A.21 (positive match): Telegram to be sent Locally stored value (appSession) Case etbTopoCnt opTrnTopoCnt etbTopoCntFilter opTrnTopoCntFilter 1 any any 0 0 2 any equal 0 equal 3 equal any equal 0 4 equal equal equal equal.

void trdp_queueAppLast (PD_ELE_T **ppHead, PD_ELE_T *pNew)

Append an element at end of queue.

void trdp_queueInsFirst (PD_ELE_T **ppHead, PD_ELE_T *pNew)

Insert an element at front of queue.

• void trdp initSockets (TRDP SOCKETS T iface[], UINT8 noOfEntries)

Handle the socket pool: Initialize it.

• TRDP_ERR_T trdp_requestSocket (TRDP_SOCKETS_T iface[], UINT16 port, const TRDP_SEND_PA← RAM_T *params, TRDP_IP_ADDR_T srcIP, TRDP_IP_ADDR_T mcGroup, TRDP_SOCK_TYPE_T type, TRDP_OPTION_T options, BOOL8 rcvMostly, SOCKET useSocket, INT32 *pIndex, TRDP_IP_ADDR_← T cornerlp)

Handle the socket pool: Request a socket from our socket pool First we loop through the socket pool and check if there is already a socket which would suit us.

void trdp_releaseSocket (TRDP_SOCKETS_T iface[], INT32 IIndex, UINT32 connectTimeout, BOOL8 checkAll, TRDP_IP_ADDR_T mcGroupUsed)

Handle the socket pool: if a received TCP socket is unused, the socket connection timeout is started.

- UINT32 trdp_getSeqCnt (UINT32 comId, TRDP_MSG_T msgType, TRDP_IP_ADDR_T srclpAddr)
 Get the initial sequence counter for the comID/message type and subnet (source IP).
- void trdp_resetSequenceCounter (PD_ELE_T *pElement, TRDP_IP_ADDR_T srcIP, TRDP_MSG_T msg
 — Type)

remove the sequence counter for the comID/source IP.

check and update the sequence counter for the comID/source IP.

- BOOL8 trdp_isAddressed (const TRDP_URI_USER_T listUri, const TRDP_URI_USER_T destUri)
 Check if listener URI is in addressing range of destination URI.
- BOOL8 trdp_isInIPrange (TRDP_IP_ADDR_T receivedSrcIP, TRDP_IP_ADDR_T listenedSourceIPlow, T
 —
 RDP IP ADDR T listenedSourceIPhigh)

Check if received IP is in addressing range of listener's IPs.

5.36.1 Detailed Description

Helper functions for TRDP communication.

Note

Project: TCNOpen TRDP prototype stack

Author

Bernd Loehr, NewTec GmbH

Remarks

This Source Code Form is subject to the terms of the Mozilla Public License, v. 2.0. If a copy of the MPL was not distributed with this file, You can obtain one at http://mozilla.org/MPL/2.0/. Copyright Bombardier Transportation Inc. or its subsidiaries and others, 2013-2019. All rights reserved.

5.36.2 Function Documentation

5.36.2.1 printSocketUsage()

Debug socket usage output.

Parameters

in <i>iface</i>	List of sockets
-----------------	-----------------

5.36.2.2 trdp_checkSequenceCounter()

check and update the sequence counter for the comID/source IP.

If the comID/srcIP is not found, update it and return 0 - else if already received, return 1 On memory error, return -1

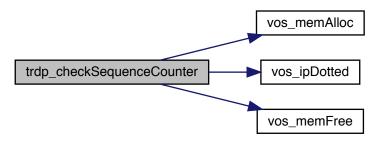
Parameters

in	pElement	subscription element
in	sequenceCounter	sequence counter to check
in	srcIP	Source IP address
in	msgType	type of the message

Return values

```
0 - no duplicate 1 - duplicate or old sequence counter -1 - memory error
```

Here is the call graph for this function:



5.36.2.3 trdp_findMCjoins()

Check an MC group not used by other sockets / subscribers/ listeners.

Parameters

in	appHandle	the handle returned by tlc_openSession
in	mcGroup	multicast group to look for

Return values

```
multi cast group if unused VOS_INADDR_ANY if used
```

5.36.2.4 trdp_findSubAddr()

Return the element with same comld and IP addresses.

Parameters

in	pHead	pointer to head of queue
in	addr	Pub/Sub handle (Address, ComID, srcIP & dest IP, serviceId) to search for
in	comld	Comld to stay on on a sorted search, 0 when searching on unsorted queues

Return values

!=	NULL pointer to PD element	
NULL No PD element found		

5.36.2.5 trdp_getSeqCnt()

Get the initial sequence counter for the comID/message type and subnet (source IP).

If the comID/srcIP is not found elsewhere, return 0 - else return its current sequence number (the redundant packet needs the same seqNo)

Note: The standard demands that sequenceCounter is managed per comID/msgType at each publisher, but shall be the same for redundant telegrams (subnet/srcIP).

Parameters

in	comld	comID to look for
in	msgType	PD/MD type
in	srclpAddr	Source IP address

Return values

return	the sequence number

5.36.2.6 trdp_initSockets()

Handle the socket pool: Initialize it.

Parameters

in	iface	pointer to the socket pool
	0/5	
ın	noOtEntries	entries in the socket pool

5.36.2.7 trdp_isAddressed()

```
BOOL8 trdp_isAddressed (

const TRDP_URI_USER_T listUri,

const TRDP_URI_USER_T destUri )
```

Check if listener URI is in addressing range of destination URI.

Parameters

in	listUri	Null terminated listener URI string to compare
in	destUri	Null terminated destination URI string to compare

Return values

FALSE	- not in addressing range	
TRUE	- listener URI is in addressing range of destination URI	

5.36.2.8 trdp_isInIPrange()

```
BOOL8 trdp_isInIPrange (

TRDP_IP_ADDR_T receivedSrcIP,

TRDP_IP_ADDR_T listenedSourceIPlow,

TRDP_IP_ADDR_T listenedSourceIPhigh )
```

Check if received IP is in addressing range of listener's IPs.

Parameters

in	receivedSrcIP	Received IP address
in	listenedSourceIPlow	Lower bound IP
in	listenedSourceIPhigh	Upper bound IP

Return values

FALSE	- not in addressing range	
TRUE	- received IP is in addressing range of listener	

5.36.2.9 trdp_packetSizeMD()

Get the packet size from the raw data size.

Parameters

	in	dataSize	net data size (without padding)	
--	----	----------	---------------------------------	--

Return values

packet	size the size of the complete packet to be sent or received
--------	---

5.36.2.10 trdp_packetSizePD()

Get the packet size from the raw data size.

Parameters

	in	dataSize	net data size (without padding)	
--	----	----------	---------------------------------	--

Return values

5.36.2.11 trdp_queueAppLast()

Append an element at end of queue.

Parameters

in	ppHead	pointer to pointer to head of queue
in	pNew	pointer to element to append

5.36.2.12 trdp_queueDelElement()

Delete an element.

Parameters

in	ppHead	pointer to pointer to head of queue
in	pDelete	pointer to element to delete

5.36.2.13 trdp_queueFindComId()

Return the element with same comld.

Parameters

in	pHead	pointer to head of queue
in	comld	ComID to search for

Return values

!=	NULL pointer to PD element
NULL	No PD element found

5.36.2.14 trdp_queueFindExistingSub()

```
PD_ELE_T* trdp_queueFindExistingSub ( \label{eq:pd_ele} \texttt{PD_ELE_T} * \textit{pHead}, \\ \texttt{TRDP\_ADDRESSES\_T} * \textit{addr} \; )
```

Return the element with same comld and IP addresses.

Parameters

in	pHead	pointer to head of queue	
in	addr	Pub/Sub handle (Address, ComID, srcIP & dest IP) to search for	

Return values

	NULL pointer to PD element
NULL	No PD element found

5.36.2.15 trdp_queueFindPubAddr()

Return the element with same comId, serviceId and IP addresses.

Parameters

in	pHead	pointer to head of queue	
in	addr	Pub/Sub handle (Address, ComID, srcIP & dest IP, serviceId) to search for	

Return values

!=	NULL pointer to PD element
NULL	No PD element found

5.36.2.16 trdp_queueFindSubAddr()

Return the element with same comld and IP addresses.

Parameters

in	pHead	pointer to head of queue	
in	addr	Pub/Sub handle (Address, ComID, srcIP & dest IP, serviceId) to search for	

Return values

!=	NULL pointer to PD element
NULL	No PD element found

Here is the call graph for this function:



5.36.2.17 trdp_queueInsFirst()

Insert an element at front of queue.

Parameters

in	ppHead	pointer to pointer to head of queue
in	pNew	pointer to element to insert

5.36.2.18 trdp_releaseSocket()

Handle the socket pool: if a received TCP socket is unused, the socket connection timeout is started.

In Udp, Release a socket from our socket pool

Parameters

in,out	iface	socket pool
in	IIndex	index of socket to release
in	connectTimeout	time out
in	checkAll	release all TCP pending sockets
in	mcGroupUsed	release MC group subscription

5.36.2.19 trdp_requestSocket()

```
TRDP_ERR_T trdp_requestSocket (
          TRDP_SOCKETS_T iface[],
          UINT16 port,
          const TRDP_SEND_PARAM_T * params,
          TRDP_IP_ADDR_T srcIP,
          TRDP_IP_ADDR_T mcGroup,
          TRDP_SOCK_TYPE_T type,
          TRDP_OPTION_T options,
          BOOL8 rcvMostly,
          SOCKET useSocket,
```

```
INT32 * pIndex,
TRDP_IP_ADDR_T cornerIp )
```

Handle the socket pool: Request a socket from our socket pool First we loop through the socket pool and check if there is already a socket which would suit us.

If a multicast group should be joined, we do that on an otherwise suitable socket - up to 20 multicast goups can be joined per socket. If a socket for multicast publishing is requested, we also use the source IP to determine the interface for outgoing multicast traffic.

Parameters

in,out	iface	socket pool
in	port	port to use
in	params	parameters to use
in	srcIP	IP to bind to (0 = any address)
in	mcGroup	MC group to join (0 = do not join)
in	type	type determines port to bind to (PD, MD/UDP, MD/TCP)
in	options	blocking/nonblocking
in	rcvMostly	primarily used for receiving (tbd: bind on sender, too?)
out	useSocket	socket to use, do not open a new one
out	pIndex	returned index of socket pool
in	cornerlp	only used for receiving

Return values

TRDP_NO_ERR	
TRDP_PARAM_ERR	

5.36.2.20 trdp_resetSequenceCounter()

remove the sequence counter for the comID/source IP.

The sequence counter should be reset if there was a packet time out.

Parameters

in	pElement	subscription element
in	srcIP	Source IP address
in	msgType	message type

Return values

5.36.2.21 trdp_SockAddJoin()

```
BOOL8 trdp_SockAddJoin (

TRDP_IP_ADDR_T mcList[VOS_MAX_MULTICAST_CNT],

TRDP_IP_ADDR_T mcGroup )
```

Add mc group to the list.

Parameters

in	mcList	List of multicast groups
in	mcGroup	multicast group

Return values

```
1 if added 0 if list is full
```

5.36.2.22 trdp_SockDelJoin()

```
BOOL8 trdp_SockDelJoin (

TRDP_IP_ADDR_T mcList[VOS_MAX_MULTICAST_CNT],

TRDP_IP_ADDR_T mcGroup )
```

remove mc group from the list

Parameters

in	mcList	List of multicast groups
in	mcGroup	multicast group

Return values

```
1 if deleted 0 was not in list
```

5.36.2.23 trdp_SockIsJoined()

Check if a mc group is in the list.

Parameters

in	mcList	List of multicast groups
in	mcGroup	multicast group

Return values

5.36.2.24 trdp_validTopoCounters()

Check topography counters The applied conformance pattern follows Table A.5/A.21 (positive match): Telegram to be sent Locally stored value (appSession) Case etbTopoCnt opTrnTopoCnt etbTopoCntFilter opTrnTopoCntFilter 1 any any 0 0 2 any equal 0 equal 3 equal any equal 0 4 equal equal equal equal.

Parameters

in	etbTopoCnt ETB topography counter to be check	
in	opTrnTopoCnt Operational topography counter to be checked	
in	etbTopoCntFilter	ETB topography counter filter value
in	opTrnTopoCntFilter	Operational topography counter filter value

Return values

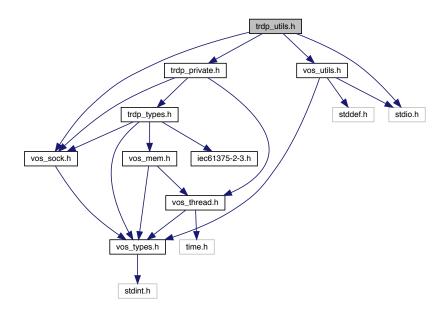
```
TRUE | Filter criteria matched FALSE Filter criteria not matched
```

5.37 trdp_utils.h File Reference

Common utilities for TRDP communication.

```
#include <stdio.h>
#include "trdp_private.h"
#include "vos_utils.h"
#include "vos_sock.h"
```

Include dependency graph for trdp_utils.h:



This graph shows which files directly or indirectly include this file:



Functions

- void printSocketUsage (TRDP_SOCKETS_T iface[])
 - Debug socket usage output.
- BOOL8 trdp_SockIsJoined (const TRDP_IP_ADDR_T mcList[VOS_MAX_MULTICAST_CNT], TRDP_IP_

 ADDR_T mcGroup)

Check if a mc group is in the list.

BOOL8 trdp_SockAddJoin (TRDP_IP_ADDR_T mcList[VOS_MAX_MULTICAST_CNT], TRDP_IP_ADDR
 — T mcGroup)

Add mc group to the list.

BOOL8 trdp_SockDelJoin (TRDP_IP_ADDR_T mcList[VOS_MAX_MULTICAST_CNT], TRDP_IP_ADDR_T mcGroup)

remove mc group from the list

- PD ELE T * trdp queueFindComId (PD ELE T *pHead, UINT32 comId)
 - Return the element with same comld.
- PD_ELE_T * trdp_findSubAddr (PD_ELE_T *pHead, TRDP_ADDRESSES_T *pAddr, UINT32 comId)

 Return the element with same comId and IP addresses.
- PD_ELE_T * trdp_queueFindSubAddr (PD_ELE_T *pHead, TRDP_ADDRESSES_T *pAddr)

 Return the element with same comld and IP addresses.
- PD_ELE_T * trdp_queueFindExistingSub (PD_ELE_T *pHead, TRDP_ADDRESSES_T *pAddr)

Return the element with same comld and IP addresses.

PD ELE T * trdp queueFindPubAddr (PD ELE T *pHead, TRDP ADDRESSES T *addr)

Return the element with same comld, serviceld and IP addresses.

void trdp_queueDelElement (PD_ELE_T **pHead, PD_ELE_T *pDelete)

Delete an element.

void trdp_queueAppLast (PD_ELE_T **pHead, PD_ELE_T *pNew)

Append an element at end of queue.

void trdp_queueInsFirst (PD_ELE_T **pHead, PD_ELE_T *pNew)

Insert an element at front of queue.

void trdp initSockets (TRDP SOCKETS T iface[], UINT8 noOfEntries)

Handle the socket pool: Initialize it.

void trdp_resetSequenceCounter (PD_ELE_T *pElement, TRDP_IP_ADDR_T srcIP, TRDP_MSG_T msg
 — Type)

remove the sequence counter for the comID/source IP.

- TRDP_IP_ADDR_T trdp_findMCjoins (TRDP_APP_SESSION_T appHandle, TRDP_IP_ADDR_T mcGroup)

 Check an MC group not used by other sockets / subscribers/ listeners.
- TRDP_ERR_T trdp_requestSocket (TRDP_SOCKETS_T iface[], UINT16 port, const TRDP_SEND_PA

 RAM_T *params, TRDP_IP_ADDR_T srcIP, TRDP_IP_ADDR_T mcGroup, TRDP_SOCK_TYPE_T type,
 TRDP_OPTION_T options, BOOL8 rcvMostly, SOCKET useSocket, INT32 *pIndex, TRDP_IP_ADDR_←
 T cornerlp)

Handle the socket pool: Request a socket from our socket pool First we loop through the socket pool and check if there is already a socket which would suit us.

void trdp_releaseSocket (TRDP_SOCKETS_T iface[], INT32 IIndex, UINT32 connectTimeout, BOOL8 checkAll, TRDP IP ADDR T mcGroupUsed)

Handle the socket pool: if a received TCP socket is unused, the socket connection timeout is started.

UINT32 trdp packetSizePD (UINT32 dataSize)

Get the packet size from the raw data size.

UINT32 trdp packetSizeMD (UINT32 dataSize)

Get the packet size from the raw data size.

- UINT32 trdp_getSeqCnt (UINT32 comId, TRDP_MSG_T msgType, TRDP_IP_ADDR_T srclpAddr)
 Get the initial sequence counter for the comID/message type and subnet (source IP).

check and update the sequence counter for the comID/source IP.

- BOOL8 trdp_isAddressed (const TRDP_URI_USER_T listUri, const TRDP_URI_USER_T destUri)

 Check if listener URI is in addressing range of destination URI.
- BOOL8 trdp_validTopoCounters (UINT32 etbTopoCnt, UINT32 opTrnTopoCnt, UINT32 etbTopoCntFilter, U
 INT32 opTrnTopoCntFilter)

Check topography counters The applied conformance pattern follows Table A.5/A.21 (positive match): Telegram to be sent Locally stored value (appSession) Case etbTopoCnt opTrnTopoCnt etbTopoCntFilter opTrnTopoCntFilter 1 any any 0 0 2 any equal 0 equal 3 equal any equal 0 4 equal equal equal equal.

• BOOL8 trdp_isInIPrange (TRDP_IP_ADDR_T receivedSrcIP, TRDP_IP_ADDR_T listenedSourceIPlow, T← RDP_IP_ADDR_T listenedSourceIPhigh)

Check if received IP is in addressing range of listener's IPs.

5.37.1 Detailed Description

Common utilities for TRDP communication.

Note

Project: TCNOpen TRDP prototype stack

Author

Bernd Loehr, NewTec GmbH

Remarks

This Source Code Form is subject to the terms of the Mozilla Public License, v. 2.0. If a copy of the MPL was not distributed with this file, You can obtain one at http://mozilla.org/MPL/2.0/. Copyright Bombardier Transportation Inc. or its subsidiaries and others, 2013-2019. All rights reserved.

5.37.2 Function Documentation

5.37.2.1 printSocketUsage()

Debug socket usage output.

Parameters

in	iface	List of sockets
----	-------	-----------------

5.37.2.2 trdp_checkSequenceCounter()

check and update the sequence counter for the comID/source IP.

If the comID/srcIP is not found, update it and return 0 - else if already received, return 1 On memory error, return -1

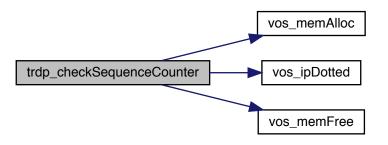
Parameters

in	pElement	subscription element
in	sequenceCounter	sequence counter to check
in	srcIP	Source IP address
in	msgType	type of the message

Return values

```
0 - no duplicate 1 - duplicate or old sequence counter -1 - memory error
```

Here is the call graph for this function:



5.37.2.3 trdp_findMCjoins()

Check an MC group not used by other sockets / subscribers/ listeners.

Parameters

in	appHandle	the handle returned by tlc_openSession
in	mcGroup	multicast group to look for

Return values

m	ulti	cast group if unused VOS_INADDR_ANY if used
---	------	---

5.37.2.4 trdp_findSubAddr()

Return the element with same comld and IP addresses.

Parameters

in	pHead	pointer to head of queue
in	addr	Pub/Sub handle (Address, ComID, srcIP & dest IP, serviceId) to search for
in	comld	Comld to stay on on a sorted search, 0 when searching on unsorted queues

Return values

!=	NULL pointer to PD element
NULL	No PD element found

5.37.2.5 trdp_getSeqCnt()

Get the initial sequence counter for the comID/message type and subnet (source IP).

If the comID/srcIP is not found elsewhere, return 0 - else return its current sequence number (the redundant packet needs the same seqNo)

Note: The standard demands that sequenceCounter is managed per comID/msgType at each publisher, but shall be the same for redundant telegrams (subnet/srcIP).

Parameters

in	comId	comID to look for
in	msgType	PD/MD type
in	srclpAddr	Source IP address

Return values

return	the sequence number
--------	---------------------

5.37.2.6 trdp_initSockets()

Handle the socket pool: Initialize it.

Parameters

in	iface	pointer to the socket pool
in	noOfEntries	entries in the socket pool

5.37.2.7 trdp_isAddressed()

```
BOOL8 trdp_isAddressed (

const TRDP_URI_USER_T listUri,

const TRDP_URI_USER_T destUri )
```

Check if listener URI is in addressing range of destination URI.

Parameters

in	listUri	Null terminated listener URI string to compare
in	destUri	Null terminated destination URI string to compare

Return values

FALSE	- not in addressing range
TRUE	- listener URI is in addressing range of destination URI

5.37.2.8 trdp_isInIPrange()

```
BOOL8 trdp_isInIPrange (

TRDP_IP_ADDR_T receivedSrcIP,

TRDP_IP_ADDR_T listenedSourceIPlow,

TRDP_IP_ADDR_T listenedSourceIPhigh )
```

Check if received IP is in addressing range of listener's IPs.

Parameters

in	receivedSrcIP	Received IP address
in	listenedSourceIPlow	Lower bound IP
in	listenedSourceIPhigh	Upper bound IP

Return values

FALSE	- not in addressing range
TRUE	- received IP is in addressing range of listener

5.37.2.9 trdp_packetSizeMD()

Get the packet size from the raw data size.

Parameters

-	in	dataSize	net data size (without padding)
---	----	----------	---------------------------------

Return values

packet	size the size of the complete packet to be sent or received
--------	---

5.37.2.10 trdp_packetSizePD()

Get the packet size from the raw data size.

Parameters

	in	dataSize	net data size (without padding)
--	----	----------	---------------------------------

Return values

packet	size the size of the complete packet to be sent or received
--------	---

5.37.2.11 trdp_queueAppLast()

Append an element at end of queue.

Parameters

in	ppHead	pointer to pointer to head of queue
in	pNew	pointer to element to append

5.37.2.12 trdp_queueDelElement()

Delete an element.

Parameters

in	ppHead	pointer to pointer to head of queue
in <i>pDelete</i>		pointer to element to delete

5.37.2.13 trdp_queueFindComId()

Return the element with same comld.

Parameters

in	pHead	pointer to head of queue
in	comld	ComID to search for

Return values

!=	NULL pointer to PD element
NULL	No PD element found

5.37.2.14 trdp_queueFindExistingSub()

```
PD_ELE_T* trdp_queueFindExistingSub ( \label{eq:pd_ele} \texttt{PD}\_\texttt{ELE}\_\texttt{T} \ * \ p\textit{Head}, \\ \texttt{TRDP}\_\texttt{ADDRESSES}\_\texttt{T} \ * \ addr \ )
```

Return the element with same comld and IP addresses.

Parameters

in	pHead	pointer to head of queue	
in	addr	Pub/Sub handle (Address, ComID, srcIP & dest IP) to search for	

Return values

!=	NULL pointer to PD element
NULL	No PD element found

5.37.2.15 trdp_queueFindPubAddr()

Return the element with same comId, serviceId and IP addresses.

Parameters

in	pHead	pointer to head of queue
in	addr	Pub/Sub handle (Address, ComID, srcIP & dest IP, serviceId) to search for

Return values

!=	NULL pointer to PD element
NULL	No PD element found

5.37.2.16 trdp_queueFindSubAddr()

Return the element with same comld and IP addresses.

Parameters

in	pHead	pointer to head of queue	
in	addr	Pub/Sub handle (Address, ComID, srcIP & dest IP, serviceId) to search for	

Return values

!=	NULL pointer to PD element
NULL	No PD element found

Here is the call graph for this function:



5.37.2.17 trdp_queueInsFirst()

Insert an element at front of queue.

Parameters

in	ppHead	pointer to pointer to head of queue
in	pNew	pointer to element to insert

5.37.2.18 trdp_releaseSocket()

Handle the socket pool: if a received TCP socket is unused, the socket connection timeout is started.

In Udp, Release a socket from our socket pool

Parameters

in,out	iface	socket pool
in	IIndex	index of socket to release
in	connectTimeout	time out
in	checkAll	release all TCP pending sockets
in	mcGroupUsed	release MC group subscription

5.37.2.19 trdp_requestSocket()

```
INT32 * pIndex,
TRDP_IP_ADDR_T cornerIp )
```

Handle the socket pool: Request a socket from our socket pool First we loop through the socket pool and check if there is already a socket which would suit us.

If a multicast group should be joined, we do that on an otherwise suitable socket - up to 20 multicast goups can be joined per socket. If a socket for multicast publishing is requested, we also use the source IP to determine the interface for outgoing multicast traffic.

Parameters

in,out	iface	socket pool
in	port	port to use
in	params	parameters to use
in	srcIP	IP to bind to (0 = any address)
in	mcGroup	MC group to join (0 = do not join)
in	type	type determines port to bind to (PD, MD/UDP, MD/TCP)
in	options	blocking/nonblocking
in	rcvMostly	primarily used for receiving (tbd: bind on sender, too?)
out	useSocket	socket to use, do not open a new one
out	pIndex	returned index of socket pool
in	cornerlp	only used for receiving

Return values

TRDP_NO_ERR	
TRDP_PARAM_ERR	

5.37.2.20 trdp_resetSequenceCounter()

remove the sequence counter for the comID/source IP.

The sequence counter should be reset if there was a packet time out.

Parameters

in	pElement	subscription element
in	srcIP	Source IP address
in	msgType	message type

Return values

5.37.2.21 trdp_SockAddJoin()

```
BOOL8 trdp_SockAddJoin (

TRDP_IP_ADDR_T mcList[VOS_MAX_MULTICAST_CNT],

TRDP_IP_ADDR_T mcGroup )
```

Add mc group to the list.

Parameters

in	mcList	List of multicast groups
in	mcGroup	multicast group

Return values

```
1 if added 0 if list is full
```

5.37.2.22 trdp_SockDelJoin()

```
BOOL8 trdp_SockDelJoin (

TRDP_IP_ADDR_T mcList[VOS_MAX_MULTICAST_CNT],

TRDP_IP_ADDR_T mcGroup )
```

remove mc group from the list

Parameters

in	mcList	List of multicast groups
in	mcGroup	multicast group

Return values

```
1 if deleted 0 was not in list
```

5.37.2.23 trdp_SockIsJoined()

Check if a mc group is in the list.

Parameters

in	mcList	List of multicast groups
in	mcGroup	multicast group

Return values

```
1 if found 0 if not found
```

5.37.2.24 trdp_validTopoCounters()

Check topography counters The applied conformance pattern follows Table A.5/A.21 (positive match): Telegram to be sent Locally stored value (appSession) Case etbTopoCnt opTrnTopoCnt etbTopoCntFilter opTrnTopoCntFilter 1 any any 0 0 2 any equal 0 equal 3 equal any equal 0 4 equal equal equal equal.

Parameters

in	etbTopoCnt	ETB topography counter to be checked
in	opTrnTopoCnt	Operational topography counter to be checked
in	etbTopoCntFilter	ETB topography counter filter value
in	opTrnTopoCntFilter	Operational topography counter filter value

Return values

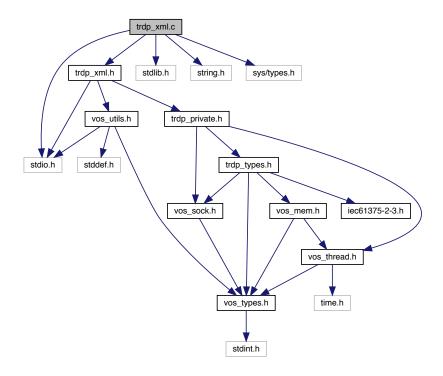
```
TRUE | Filter criteria matched FALSE Filter criteria not matched
```

5.38 trdp_xml.c File Reference

Simple XML parser.

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <sys/types.h>
#include "trdp_xml.h"
```

Include dependency graph for trdp_xml.c:



Functions

- TRDP_ERR_T trdp_XMLOpen (XML_HANDLE_T *pXML, const char *file)
 Opens the XML parsing.
- $\bullet \ \ \mathsf{TRDP_ERR_T} \ \mathsf{trdp_XMLMemOpen} \ (\mathsf{XML_HANDLE_T} \ *\mathsf{pXML}, \ \mathsf{char} \ *\mathsf{pBuffer}, \ \mathsf{size_t} \ \mathsf{bufSize})$
- Opens the XML parsing from a buffer (string stream).
 void trdp_XMLRewind (XML_HANDLE_T *pXML)

Rewind to start.

void trdp XMLClose (XML HANDLE T *pXML)

Closes the XML parsng.

• int trdp_XMLSeekStartTagAny (XML_HANDLE_T *pXML, char *tag, int maxlen)

Seek next tag on starting depth and return it in provided buffer.

int trdp_XMLSeekStartTag (XML_HANDLE_T *pXML, const char *tag)

Seek a specific tag.

int trdp_XMLCountStartTag (XML_HANDLE_T *pXML, const char *tag)

Count a specific tag.

void trdp_XMLEnter (XML_HANDLE_T *pXML)

Enter level in XML file.

void trdp_XMLLeave (XML_HANDLE_T *pXML)

Leave level in XML file.

XML_TOKEN_T trdp_XMLGetAttribute (XML_HANDLE_T *pXML, CHAR8 *attribute, UINT32 *pValueInt, CHAR8 *value)

Get value of next attribute, as string and if possible as integer.

5.38.1 Detailed Description

Simple XML parser.

Hint: Missing optional elements must be handled using the count-function, otherwise following elements will be following ignored!

Note

Project: TCNOpen TRDP prototype stack

Author

Bernd Loehr, NewTec GmbH; based on code by Peter Brander, Bombardier

Remarks

This Source Code Form is subject to the terms of the Mozilla Public License, v. 2.0. If a copy of the MPL was not distributed with this file, You can obtain one at http://mozilla.org/MPL/2.0/.

5.38.2 Function Documentation

5.38.2.1 trdp_XMLClose()

Closes the XML parsng.

Parameters

in <i>pXML</i>	Pointer to local data
----------------	-----------------------

Return values

none

5.38.2.2 trdp_XMLCountStartTag()

Count a specific tag.

Parameters

in	pXML	Pointer to local data
in	tag	Tag to count

Return values

```
0 if found !=0 if not found
```

5.38.2.3 trdp_XMLEnter()

Enter level in XML file.

Parameters

in <i>pXI</i>	ΛL	Pointer to local data
---------------	----	-----------------------

Return values

```
none
```

5.38.2.4 trdp_XMLGetAttribute()

Get value of next attribute, as string and if possible as integer.

Parameters

in	pXML	Pointer to local data
in	attribute	Pointer to attribute
out	pValueInt	Pointer to resulting integer value
out	value	Pointer to resulting string value

Return values

TOK ATTRIBUTE	if found token if not found

5.38.2.5 trdp_XMLLeave()

Leave level in XML file.

Parameters

	in	pXML	Pointer to local data
--	----	------	-----------------------

Return values

```
none
```

5.38.2.6 trdp_XMLMemOpen()

Opens the XML parsing from a buffer (string stream).

Parameters

in	pXML	Pointer to local data
in	pBuffer	Pointer to XML stream buffer
in	bufSize	Size of XML stream buffer

Return values

```
TRDP_IO_ERR
```

Here is the call graph for this function:



5.38.2.7 trdp_XMLOpen()

Opens the XML parsing.

Parameters

in	pXML	Pointer to local data
in	file	Pathname of XML file

Return values

none	

5.38.2.8 trdp_XMLRewind()

```
void trdp_XMLRewind ( {\tt XML\_HANDLE\_T~*~pXML~)}
```

Rewind to start.

Parameters

in	pXML	Pointer to local data
	 	

Return values

```
none
```

5.38.2.9 trdp_XMLSeekStartTag()

Seek a specific tag.

Parameters

in	pXML	Pointer to local data
in	tag	Tag to be found

Return values

```
0 if found !=0 if not found
```

5.38.2.10 trdp_XMLSeekStartTagAny()

Seek next tag on starting depth and return it in provided buffer.

Start tags on deeper depths are ignored.

Parameters

in	pXML	Pointer to local data
in,out	tag	Buffer for found tag
in	maxlen	Length of buffer

Return values

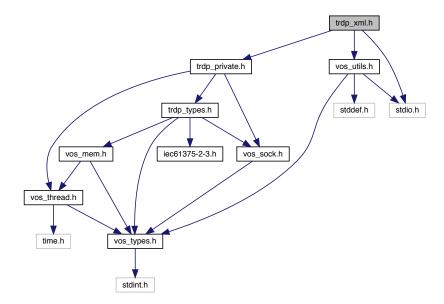
```
0 if found !=0 if not found
```

5.39 trdp_xml.h File Reference

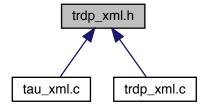
Simple XML parser.

```
#include <stdio.h>
#include "trdp_private.h"
#include "vos_utils.h"
```

Include dependency graph for trdp_xml.h:



This graph shows which files directly or indirectly include this file:



Functions

- TRDP_ERR_T trdp_XMLOpen (XML_HANDLE_T *pXML, const char *file)

 Opens the XML parsing.
- TRDP_ERR_T trdp_XMLMemOpen (XML_HANDLE_T *pXML, char *pBuffer, size_t bufSize)

 Opens the XML parsing from a buffer (string stream).
- void trdp_XMLClose (XML_HANDLE_T *pXML)

Closes the XML parsng.

- int trdp_XMLCountStartTag (XML_HANDLE_T *pXML, const char *tag)

 Count a specific tag.
- int trdp_XMLSeekStartTagAny (XML_HANDLE_T *pXML, char *tag, int maxlen)

 Seek next tag on starting depth and return it in provided buffer.

• int trdp_XMLSeekStartTag (XML_HANDLE_T *pXML, const char *tag)

Seek a specific tag.

XML_TOKEN_T trdp_XMLGetAttribute (XML_HANDLE_T *pXML, CHAR8 *attribute, UINT32 *pValueInt, CHAR8 *value)

Get value of next attribute, as string and if possible as integer.

void trdp_XMLRewind (XML_HANDLE_T *pXML)

Rewind to start.

void trdp XMLEnter (XML HANDLE T *pXML)

Enter level in XML file.

void trdp_XMLLeave (XML_HANDLE_T *pXML)

Leave level in XML file.

5.39.1 Detailed Description

Simple XML parser.

Note

Project: TCNOpen TRDP prototype stack

Author

Bernd Loehr, NewTec GmbH

Remarks

This Source Code Form is subject to the terms of the Mozilla Public License, v. 2.0. If a copy of the MPL was not distributed with this file, You can obtain one at http://mozilla.org/MPL/2.0/. Copyright NewTec GmbH or its subsidiaries and others, 2016. All rights reserved.

5.39.2 Function Documentation

5.39.2.1 trdp_XMLClose()

Closes the XML parsng.

Parameters

in	pXML	Pointer to local data

Return values

none

5.39.2.2 trdp_XMLCountStartTag()

Count a specific tag.

Parameters

in	pXML	Pointer to local data
in	tag	Tag to count

Return values

```
0 if found !=0 if not found
```

5.39.2.3 trdp_XMLEnter()

Enter level in XML file.

Parameters

in pXML Pointer to local data	 a
-------------------------------	-------

Return values

```
none
```

5.39.2.4 trdp_XMLGetAttribute()

Get value of next attribute, as string and if possible as integer.

Parameters

in	pXML	Pointer to local data
in	attribute	Pointer to attribute
out	pValueInt	Pointer to resulting integer value
out	value	Pointer to resulting string value

Return values

5.39.2.5 trdp_XMLLeave()

```
void trdp_XMLLeave ( {\tt XML\_HANDLE\_T~*~pXML~})
```

Leave level in XML file.

Parameters

in <i>pXM</i>	Pointer to local data
---------------	-----------------------

Return values

none

5.39.2.6 trdp_XMLMemOpen()

Opens the XML parsing from a buffer (string stream).

Parameters

in	pXML	Pointer to local data
in	pBuffer	Pointer to XML stream buffer
in	bufSize	Size of XML stream buffer

Return values

TRDP_IO_ERR

Here is the call graph for this function:



5.39.2.7 trdp_XMLOpen()

Opens the XML parsing.

Parameters

in	pXML	Pointer to local data
in	file	Pathname of XML file

Return values

none

5.39.2.8 trdp_XMLRewind()

Rewind to start.

Parameters

in pXML Pointer to lo	cal data
-----------------------	----------

Return values

none

5.39.2.9 trdp_XMLSeekStartTag()

Seek a specific tag.

Parameters

in	pXML	Pointer to local data
in	tag	Tag to be found

Return values

```
0 if found !=0 if not found
```

5.39.2.10 trdp_XMLSeekStartTagAny()

Seek next tag on starting depth and return it in provided buffer.

Start tags on deeper depths are ignored.

Parameters

in	pXML	Pointer to local data
in,out	tag	Buffer for found tag
in	maxlen	Length of buffer

Return values

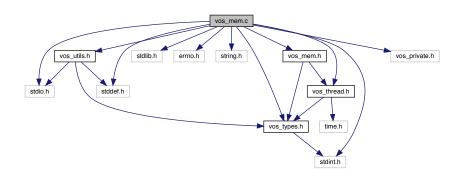
```
0 if found !=0 if not found
```

5.40 vos_mem.c File Reference

Memory functions.

```
#include <stdio.h>
#include <stddef.h>
#include <stdint.h>
#include <stdlib.h>
#include <errno.h>
```

```
#include <string.h>
#include "vos_types.h"
#include "vos_utils.h"
#include "vos_mem.h"
#include "vos_thread.h"
#include "vos_private.h"
Include dependency graph for vos mem.c:
```



Functions

Initialize the memory unit.

• EXT_DECL void vos_memDelete (UINT8 *pMemoryArea)

Delete the memory area.

EXT_DECL UINT8 * vos_memAlloc (UINT32 size)

Allocate a block of memory (from memory area above).

EXT_DECL void vos_memFree (void *pMemBlock)

Deallocate a block of memory (from memory area above).

• EXT_DECL VOS_ERR_T vos_memCount (UINT32 *pAllocatedMemory, UINT32 *pFreeMemory, UINT32 *pMinFree, UINT32 *pNumAllocBlocks, UINT32 *pNumAllocErr, UINT32 *pNumFreeErr, UINT32 block← Size[VOS MEM NBLOCKSIZES], UINT32 usedBlockSize[VOS MEM NBLOCKSIZES])

Return used and available memory (of memory area above).

EXT_DECL void vos_qsort (void *pBuf, UINT32 num, UINT32 size, int(*compare)(const void *, const void *))

Sort an array.

EXT_DECL void * vos_bsearch (const void *pKey, const void *pBuf, UINT32 num, UINT32 size, int(*compare)(const void *, const void *))

Binary search in a sorted array.

EXT_DECL_INT32 vos_strnicmp (const CHAR8 *pStr1, const CHAR8 *pStr2, UINT32 count)

Case insensitive string compare.

• EXT_DECL void vos_strncpy (CHAR8 *pStrDst, const CHAR8 *pStrSrc, UINT32 count)

String copy with length limitation.

• EXT_DECL void vos_strncat (CHAR8 *pStrDst, UINT32 count, const CHAR8 *pStrSrc)

String concatenation with length limitation.

• EXT_DECL VOS_ERR_T vos_queueCreate (VOS_QUEUE_POLICY_T queueType, UINT32 maxNoOfMsg, VOS QUEUE T *pQueueHandle)

Initialize a message queue.

• EXT_DECL VOS_ERR_T vos_queueSend (VOS_QUEUE_T queueHandle, UINT8 *pData, UINT32 size)

Send a message.

• EXT_DECL VOS_ERR_T vos_queueReceive (VOS_QUEUE_T queueHandle, UINT8 **ppData, UINT32 *pSize, UINT32 usTimeout)

Get a message.

EXT_DECL VOS_ERR_T vos_queueDestroy (VOS_QUEUE_T queueHandle)

Destroy a message queue.

5.40.1 Detailed Description

Memory functions.

OS abstraction of memory access and control

Note

Project: TCNOpen TRDP prototype stack

Author

Bernd Loehr, NewTec GmbH

Remarks

This Source Code Form is subject to the terms of the Mozilla Public License, v. 2.0. If a copy of the MPL was not distributed with this file, You can obtain one at http://mozilla.org/MPL/2.0/. Copyright Bombardier Transportation Inc. or its subsidiaries and others, 2013. All rights reserved.

5.40.2 Function Documentation

5.40.2.1 vos_bsearch()

Binary search in a sorted array.

This is just a wrapper for the standard bsearch function.

Parameters

in	pKey	Key to search for
in	pBuf	Pointer to the array to search
in	num	number of elements
in	size	size of one element
in Generate	compare ed by Doxygen	Pointer to compare function return -n if arg1 < arg2, return 0 if arg1 == arg2, return +n if arg1 > arg2 where n is an integer != 0

Return values

5.40.2.2 vos_memAlloc()

Allocate a block of memory (from memory area above).

Parameters

in	size	Size of requested block
----	------	-------------------------

Return values

Pointer	to memory area
NULL	if no memory available

5.40.2.3 vos_memCount()

```
EXT_DECL VOS_ERR_T vos_memCount (

UINT32 * pAllocatedMemory,

UINT32 * pFreeMemory,

UINT32 * pMinFree,

UINT32 * pNumAllocBlocks,

UINT32 * pNumAllocErr,

UINT32 * pNumFreeErr,

UINT32 blockSize[VOS_MEM_NBLOCKSIZES],

UINT32 usedBlockSize[VOS_MEM_NBLOCKSIZES])
```

Return used and available memory (of memory area above).

Parameters

out	pAllocatedMemory	Pointer to allocated memory size
out	pFreeMemory	Pointer to free memory size
out	pMinFree	Pointer to minimal free memory size in statistics interval
out	pNumAllocBlocks	Pointer to number of allocated memory blocks
out	pNumAllocErr	Pointer to number of allocation errors
out	pNumFreeErr	Pointer to number of free errors
out	blockSize	Pointer to list of memory block sizes
out	usedBlockSize	Pointer to list of used memoryblocks

Return values

VOS_NO_ERR	no error
VOS_INIT_ERR	module not initialised

5.40.2.4 vos_memDelete()

Delete the memory area.

This will eventually invalidate any previously allocated memory blocks! It should be called last before the application quits. No further access to the memory blocks is allowed after this call.

Parameters

in	pMemoryArea	Pointer to memory area used
----	-------------	-----------------------------

5.40.2.5 vos_memFree()

```
EXT_DECL void vos_memFree ( \mbox{void} \ * \ p\mbox{\it MemBlock} \ )
```

Deallocate a block of memory (from memory area above).

Parameters

	in	pMemBlock	Pointer to memory block to be freed
--	----	-----------	-------------------------------------

5.40.2.6 vos_memInit()

Initialize the memory unit.

Init a supplied block of memory and prepare it for use with vos_memAlloc and vos_memFree. The used block sizes can be supplied and will be preallocated. If half of the overall size of the requested memory area would be pre-allocated, either by the default pre-allocation table or a provided one, no pre-allocation takes place.

Parameters

	in	in pMemoryArea Pointer to memory area to use	
Ī	in	size	Size of provided memory area
Ī	in	fragMem	Pointer to list of preallocated block sizes, used to fragment memory for large blocks

Return values

VOS_NO_ERR	no error
VOS_PARAM_ERR	parameter out of range/invalid
VOS_MEM_ERR	no memory available
VOS_MUTEX_ERR	no mutex available

5.40.2.7 vos_qsort()

Sort an array.

This is just a wrapper for the standard qsort function.

Parameters

in, out	pBuf	Pointer to the array to sort
in	num	number of elements
in	size	size of one element
in	compare	Pointer to compare function return -n if $arg1 < arg2$, return 0 if $arg1 == arg2$, return +n if $arg1 > arg2$ where n is an integer != 0

Return values

```
none
```

5.40.2.8 vos_queueCreate()

Initialize a message queue.

Returns a handle for further calls

Parameters

iı	n	queueType	Define queue type (1 = FIFO, 2 = LIFO, 3 = PRIO)
iı	n	maxNoOfMsg	Maximum number of messages
01	ut	pQueueHandle	Handle of created queue

Return values

VOS_NO_ERR	no error
VOS_INIT_ERR	module not initialised
VOS_NOINIT_ERR	invalid handle
VOS_PARAM_ERR	parameter out of range/invalid
VOS_INIT_ERR	not supported
VOS_QUEUE_ERR	error creating queue

5.40.2.9 vos_queueDestroy()

Destroy a message queue.

Free all resources used by this queue

Parameters

in	queueHandle	Queue handle
----	-------------	--------------

Return values

VOS_NO_ERR	no error
VOS_INIT_ERR	module not initialised
VOS_NOINIT_ERR	invalid handle
VOS_PARAM_ERR	parameter out of range/invalid

5.40.2.10 vos_queueReceive()

Get a message.

Parameters

in	queueHandle	Queue handle
out	ppData	Pointer to data pointer to be received
out	pSize	Size of receive data
in	usTimeout	Maximum time to wait for a message (in usec)

Return values

VOSNO_ERR	no error
VOS_INIT_ERR	module not initialised
VOS_NOINIT_ERR	invalid handle
VOS_PARAM_ERR	parameter out of range/invalid
VOS_QUEUE_ERR	queue is empty

5.40.2.11 vos_queueSend()

Send a message.

Parameters

in	queueHandle	Queue handle
in	pData	Pointer to data to be sent
in	size	Size of data to be sent

Return values

VOS_NO_ERR	no error
VOS_INIT_ERR	module not initialised
VOS_NOINIT_ERR	invalid handle
VOS_PARAM_ERR	parameter out of range/invalid
VOS_INIT_ERR	not supported
VOS_QUEUE_ERR	error creating queue

5.40.2.12 vos_strncat()

String concatenation with length limitation.

Parameters

in	pStrDst	Destination string
in	count	Size of destination buffer
in	pStrSrc	Null terminated string to append

Return values

```
none
```

5.40.2.13 vos_strncpy()

String copy with length limitation.

Parameters

in	pStrDst	Destination string
in	pStrSrc	Null terminated string to copy
in	count	Maximum number of characters to copy

Return values

```
none
```

5.40.2.14 vos_strnicmp()

Case insensitive string compare.

Parameters

in	pStr1	Null terminated string to compare
in	pStr2	Null terminated string to compare
in	count	Maximum number of characters to compare

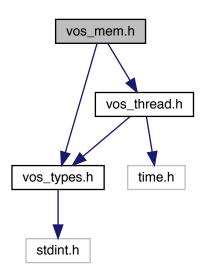
Return values

0	- equal
<0	- string1 less than string 2
>0	- string 1 greater than string 2

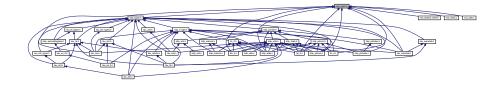
5.41 vos_mem.h File Reference

Memory and queue functions for OS abstraction.

```
#include "vos_types.h"
#include "vos_thread.h"
Include dependency graph for vos_mem.h:
```



This graph shows which files directly or indirectly include this file:



Macros

- #define VOS_MEM_NBLOCKSIZES 15u
 - No of pre-defined block sizes.
- #define VOS_MEM_MAX_PREALLOCATE 10u

We internally allocate memory always by these block sizes.

Typedefs

typedef struct VOS_QUEUE * VOS_QUEUE_T
 Opaque queue define.

Enumerations

· enum VOS QUEUE POLICY T

Queue policy matching pthread/Posix defines.

Functions

 EXT_DECL VOS_ERR_T vos_memInit (UINT8 *pMemoryArea, UINT32 size, const UINT32 fragMem[VO← S_MEM_NBLOCKSIZES])

Initialize the memory unit.

• EXT_DECL void vos_memDelete (UINT8 *pMemoryArea)

Delete the memory area.

• EXT_DECL UINT8 * vos_memAlloc (UINT32 size)

Allocate a block of memory (from memory area above).

EXT_DECL void vos_memFree (void *pMemBlock)

Deallocate a block of memory (from memory area above).

• EXT_DECL VOS_ERR_T vos_memCount (UINT32 *pAllocatedMemory, UINT32 *pFreeMemory, UINT32 *pMinFree, UINT32 *pNumAllocBlocks, UINT32 *pNumAllocErr, UINT32 *pNumFreeErr, UINT32 block← Size[VOS MEM NBLOCKSIZES], UINT32 usedBlockSize[VOS MEM NBLOCKSIZES])

Return used and available memory (of memory area above).

EXT_DECL void vos_qsort (void *pBuf, UINT32 num, UINT32 size, int(*compare)(const void *, const void *))

Sort an array.

• EXT_DECL void * vos_bsearch (const void *pKey, const void *pBuf, UINT32 num, UINT32 size, int(*compare)(const void *, const void *))

Binary search in a sorted array.

• EXT_DECL INT32 vos_strnicmp (const CHAR8 *pStr1, const CHAR8 *pStr2, UINT32 count)

Case insensitive string compare.

EXT_DECL void vos_strncpy (CHAR8 *pStrDst, const CHAR8 *pStrSrc, UINT32 count)

String copy with length limitation.

• EXT_DECL void vos_strncat (CHAR8 *pStrDst, UINT32 count, const CHAR8 *pStrSrc)

String concatenation with length limitation.

• EXT_DECL VOS_ERR_T vos_queueCreate (VOS_QUEUE_POLICY_T queueType, UINT32 maxNoOfMsg, VOS_QUEUE_T *pQueueHandle)

Initialize a message queue.

- EXT_DECL VOS_ERR_T vos_queueSend (VOS_QUEUE_T queueHandle, UINT8 *pData, UINT32 size) Send a message.
- EXT_DECL VOS_ERR_T vos_queueReceive (VOS_QUEUE_T queueHandle, UINT8 **ppData, UINT32 *pSize, UINT32 usTimeout)

Get a message.

EXT_DECL VOS_ERR_T vos_queueDestroy (VOS_QUEUE_T queueHandle)

Destroy a message queue.

5.41.1 Detailed Description

Memory and queue functions for OS abstraction.

This module provides memory control supervison

Note

Project: TCNOpen TRDP prototype stack

Author

Bernd Loehr, NewTec GmbH Peter Brander (Memory scheme)

Remarks

This Source Code Form is subject to the terms of the Mozilla Public License, v. 2.0. If a copy of the MPL was not distributed with this file, You can obtain one at http://mozilla.org/MPL/2.0/. Copyright Bombardier Transportation Inc. or its subsidiaries and others, 2013. All rights reserved.

5.41.2 Macro Definition Documentation

5.41.2.1 VOS_MEM_MAX_PREALLOCATE

```
#define VOS_MEM_MAX_PREALLOCATE 10u
```

We internally allocate memory always by these block sizes.

The largest available block is 524288 Bytes, provided the overal size of the used memory allocation area is larger. Max. no. of blocks to pre-allocate

5.41.2.2 VOS_MEM_PREALLOCATE

```
#define VOS_MEM_PREALLOCATE {0u, 0u, 0u, 0u, 0u, 0u, 0u, 4u, 0u, 0u, 0u, 0u, 0u, 0u, 0u, 0u}
```

Default pre-allocation of free memory blocks.

To avoid problems with too many small blocks and no large one. Specify how many of each block size that should be pre-allocated (and freed!) to pre-segment the memory area.

5.41.3 Function Documentation

5.41.3.1 vos_bsearch()

Binary search in a sorted array.

This is just a wrapper for the standard bsearch function.

Parameters

in	pKey	Key to search for
in	pBuf	Pointer to the array to search
in	num	number of elements
in	size	size of one element
in	compare	Pointer to compare function return -n if arg1 < arg2, return 0 if arg1 == arg2, return +n if arg1
		> arg2 where n is an integer != 0

Return values

Pointer to found element	or NULL
--------------------------	---------

5.41.3.2 vos_memAlloc()

Allocate a block of memory (from memory area above).

Parameters

	in	size	Size of requested block	
--	----	------	-------------------------	--

Return values

Pointer	to memory area
NULL	if no memory available

5.41.3.3 vos_memCount()

```
EXT_DECL VOS_ERR_T vos_memCount (

UINT32 * pAllocatedMemory,

UINT32 * pFreeMemory,

UINT32 * pMinFree,

UINT32 * pNumAllocBlocks,

UINT32 * pNumAllocErr,

UINT32 * pNumFreeErr,

UINT32 blockSize[VOS_MEM_NBLOCKSIZES],

UINT32 usedBlockSize[VOS_MEM_NBLOCKSIZES])
```

Return used and available memory (of memory area above).

Parameters

out	pAllocatedMemory	Pointer to allocated memory size
-----	------------------	----------------------------------

Parameters

out	pFreeMemory	Pointer to free memory size
out	pMinFree	Pointer to minimal free memory size in statistics interval
out	pNumAllocBlocks	Pointer to number of allocated memory blocks
out	pNumAllocErr	Pointer to number of allocation errors
out	pNumFreeErr	Pointer to number of free errors
out	blockSize	Pointer to list of memory block sizes
out	usedBlockSize	Pointer to list of used memoryblocks

Return values

VOS_NO_ERR	no error
VOS_INIT_ERR	module not initialised

5.41.3.4 vos_memDelete()

```
EXT_DECL void vos_memDelete ( {\tt UINT8*pMemoryArea} \ )
```

Delete the memory area.

This will eventually invalidate any previously allocated memory blocks! It should be called last before the application quits. No further access to the memory blocks is allowed after this call.

Parameters

in	pMemoryArea	Pointer to memory area to use

This will eventually invalidate any previously allocated memory blocks! It should be called last before the application quits. No further access to the memory blocks is allowed after this call.

Parameters

in pMemoryArea Pointer to memory area use	d
---	---

5.41.3.5 vos_memFree()

Deallocate a block of memory (from memory area above).

Parameters

	in	pMemBlock	Pointer to memory block to be freed
۱		pittern = ioon	

5.41.3.6 vos_memInit()

Initialize the memory unit.

Init a supplied block of memory and prepare it for use with vos_alloc and vos_dealloc. The used block sizes can be supplied and will be preallocated.

Parameters

in	pMemoryArea	Pointer to memory area to use
in	size	Size of provided memory area
in	fragMem	Pointer to list of preallocate block sizes, used to fragment memory for large blocks

Return values

VOS_NO_ERR	no error
VOS_PARAM_ERR	parameter out of range/invalid
VOS_MEM_ERR	no memory available

Init a supplied block of memory and prepare it for use with vos_memAlloc and vos_memFree. The used block sizes can be supplied and will be preallocated. If half of the overall size of the requested memory area would be pre-allocated, either by the default pre-allocation table or a provided one, no pre-allocation takes place.

Parameters

in	pMemoryArea	Pointer to memory area to use
in	size	Size of provided memory area
in	fragMem	Pointer to list of preallocated block sizes, used to fragment memory for large blocks

Return values

VOS_NO_ERR	no error
VOS_PARAM_ERR	parameter out of range/invalid
VOS_MEM_ERR	no memory available
VOS_MUTEX_ERR	no mutex available

5.41.3.7 vos_qsort()

```
UINT32 num,
UINT32 size,
int(*)(const void *, const void *) compare )
```

Sort an array.

This is just a wrapper for the standard qsort function.

Parameters

in,out	pBuf	Pointer to the array to sort
in	num	number of elements
in	size	size of one element
in	compare	Pointer to compare function return -n if arg1 < arg2, return 0 if arg1 == arg2, return +n
		if arg1 > arg2 where n is an integer != 0

Return values

```
none
```

5.41.3.8 vos_queueCreate()

Initialize a message queue.

Returns a handle for further calls

Parameters

in	queueType	Define queue type (1 = FIFO, 2 = LIFO, 3 = PRIO)
in	maxNoOfMsg	Maximum number of messages
out	pQueueHandle	Handle of created queue

Return values

VOS_NO_ERR	no error
VOS_INIT_ERR	module not initialised
VOS_NOINIT_ERR	invalid handle
VOS_PARAM_ERR	parameter out of range/invalid
VOS_INIT_ERR	not supported
VOS_QUEUE_ERR	error creating queue

5.41.3.9 vos_queueDestroy()

Destroy a message queue.

Free all resources used by this queue

Parameters

in	queueHandle	Queue handle
----	-------------	--------------

Return values

VOS_NO_ERR	no error
VOS_INIT_ERR	module not initialised
VOS_NOINIT_ERR	invalid handle
VOS_PARAM_ERR	parameter out of range/invalid

5.41.3.10 vos_queueReceive()

```
EXT_DECL VOS_ERR_T vos_queueReceive (

VOS_QUEUE_T queueHandle,

UINT8 ** ppData,

UINT32 * pSize,

UINT32 usTimeout )
```

Get a message.

Parameters

in	queueHandle	Queue handle
out	ppData	Pointer to data pointer to be received
out	pSize	Size of receive data
in	usTimeout	Maximum time to wait for a message (in usec)

Return values

VOSNO_ERR	no error
VOS_INIT_ERR	module not initialised
VOS_NOINIT_ERR	invalid handle
VOS_PARAM_ERR	parameter out of range/invalid
VOS_QUEUE_ERR	queue is empty

5.41.3.11 vos_queueSend()

Send a message.

Parameters

in	queueHandle	Queue handle
in	pData	Pointer to data to be sent
in	size	Size of data to be sent

Return values

VOS_NO_ERR	no error
VOS_INIT_ERR	module not initialised
VOS_NOINIT_ERR	invalid handle
VOS_PARAM_ERR	parameter out of range/invalid
VOS_INIT_ERR	not supported
VOS_QUEUE_ERR	error creating queue

5.41.3.12 vos_strncat()

String concatenation with length limitation.

Parameters

in	pStrDst	Destination string
in	count	Size of destination buffer
in	pStrSrc	Null terminated string to append

Return values

```
none
```

5.41.3.13 vos_strncpy()

```
EXT_DECL void vos_strncpy ( {\tt CHAR8} \ * \ pStrDst,
```

```
const CHAR8 * pStrSrc,
UINT32 count )
```

String copy with length limitation.

Parameters

in	pStrDst	Destination string
in	pStrSrc	Null terminated string to copy
in	count	Maximum number of characters to copy

Return values

```
none
```

5.41.3.14 vos_strnicmp()

Case insensitive string compare.

Parameters

in	pStr1	Null terminated string to compare
in	pStr2	Null terminated string to compare
in	count	Maximum number of characters to compare

Return values

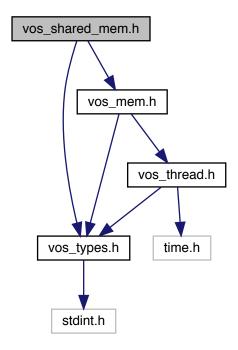
0	- equal
<0	- string1 less than string 2
>0	- string 1 greater than string 2

5.42 vos_shared_mem.h File Reference

Shared Memory functions for OS abstraction.

```
#include "vos_types.h"
#include "vos_mem.h"
```

Include dependency graph for vos_shared_mem.h:



Functions

Create a shared memory area or attach to existing one.

• EXT_DECL VOS_ERR_T vos_sharedClose (VOS_SHRD_T handle, const UINT8 *pMemoryArea)

Close connection to the shared memory area.

5.42.1 Detailed Description

Shared Memory functions for OS abstraction.

This module provides shared memory control supervison

Note

Project: TCNOpen TRDP prototype stack

Author

Kazumasa Aiba, TOSHIBA

Remarks

This Source Code Form is subject to the terms of the Mozilla Public License, v. 2.0. If a copy of the MPL was not distributed with this file, You can obtain one at http://mozilla.org/MPL/2.0/. Copyright TOSHIBA, Japan, 2013.

5.42.2 Function Documentation

5.42.2.1 vos_sharedClose()

Close connection to the shared memory area.

If the area was created by the calling process, the area will be closed (freed). If the area was attached, it will be detached. This function is not available in each target implementation.

Parameters

	in	handle	Returned handle
ſ	in	pMemoryArea	Pointer to memory area

Return values

VOS_NO_ERR	no error
VOS_MEM_ERR	no memory available

5.42.2.2 vos_sharedOpen()

Create a shared memory area or attach to existing one.

The first call with the a specified key will create a shared memory area with the supplied size and will return a handle and a pointer to that area. If the area already exists, the area will be opened. This function is not available in each target implementation.

Parameters

in	pKey	Unique identifier (file name)
out	pHandle	Pointer to returned handle
out	ppMemoryArea	Pointer to pointer to memory area
in,out	pSize	Pointer to size of area to allocate, on return actual size after attach

Return values

VOS_NO_ERR	no error
------------	----------

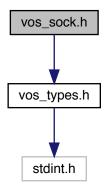
Return values

VOS_MEM_ERR	no memory available
-------------	---------------------

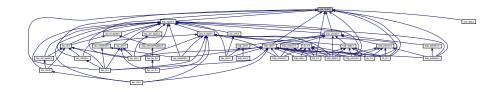
5.43 vos_sock.h File Reference

Typedefs for OS abstraction.

#include "vos_types.h"
Include dependency graph for vos_sock.h:



This graph shows which files directly or indirectly include this file:



Data Structures

struct VOS_SOCK_OPT_T

Common socket options.

Macros

#define VOS_MAX_SOCKET_CNT 4

The maximum number of sockets influences memory usage; for small systems we should define a smaller set.

#define VOS_MAX_MULTICAST_CNT 5

The maximum number of multicast groups one socket can join.

#define VOS_TTL_MULTICAST 64

The maximum number of hops a multicast packet can take.

#define VOS MAX IF NAME SIZE 16

The maximum number of IP interface adapters that can be handled by VOS.

#define VOS_MAX_NUM_IF 8

The maximum number of unicast addresses that can be handled by VOS.

#define VOS MAX NUM UNICAST 10

The MAC size supported by VOS.

#define VOS MAC SIZE 6

Size of socket send and receive buffer.

#define VOS_INVALID_SOCKET -1

Invalid socket number.

Functions

• EXT_DECL UINT16 vos_htons (UINT16 val)

Byte swapping 2 Bytes.

EXT_DECL UINT16 vos_ntohs (UINT16 val)

Byte swapping 2 Bytes.

• EXT_DECL UINT32 vos_htonl (UINT32 val)

Byte swapping 4 Bytes.

• EXT_DECL UINT32 vos_ntohl (UINT32 val)

Byte swapping 4 Bytes.

• EXT_DECL UINT64 vos_htonll (UINT64 val)

Byte swapping 8 Bytes.

EXT_DECL UINT64 vos_ntohll (UINT64 val)

Byte swapping 8 Bytes.

EXT_DECL UINT32 vos_dottedIP (const CHAR8 *pDottedIP)

Convert IP address from dotted dec.

EXT_DECL const CHAR8 * vos_ipDotted (UINT32 ipAddress)

Convert IP address to dotted dec.

EXT_DECL BOOL8 vos_isMulticast (UINT32 ipAddress)

Check if the supplied address is a multicast group address.

EXT_DECL VOS_ERR_T vos_getInterfaces (UINT32 *pAddrCnt, VOS_IF_REC_T ifAddrs[])

Get a list of interface addresses The caller has to provide an array of interface records to be filled.

• EXT_DECL BOOL8 vos_netIfUp (VOS_IP4_ADDR_T ifAddress)

Get the state of an interface.

• EXT_DECL_INT32_vos_select (SOCKET_highDesc, VOS_FDS_T_*pReadableFD, VOS_FDS_T_*p← WriteableFD, VOS_FDS_T *pErrorFD, VOS_TIMEVAL_T *pTimeOut)

select function.

EXT_DECL VOS_ERR_T vos_sockInit (void)

Initialize the socket library.

• EXT DECL void vos sockTerm (void)

De-Initialize the socket library.

EXT_DECL VOS_ERR_T vos_sockGetMAC (UINT8 pMAC[VOS_MAC_SIZE])

Return the MAC address of the default adapter.

- EXT_DECL VOS_ERR_T vos_sockOpenUDP (SOCKET *pSock, const VOS_SOCK_OPT_T *pOptions)

 Create an UDP socket.
- EXT_DECL VOS_ERR_T vos_sockOpenTCP (SOCKET *pSock, const VOS_SOCK_OPT_T *pOptions)

 Create a TCP socket.
- EXT_DECL VOS_ERR_T vos_sockClose (SOCKET sock)

Close a socket.

- EXT_DECL VOS_ERR_T vos_sockSetOptions (SOCKET sock, const VOS_SOCK_OPT_T *pOptions)
 Set socket options.
- EXT_DECL VOS_ERR_T vos_sockJoinMC (SOCKET sock, UINT32 mcAddress, UINT32 ipAddress)
 Join a multicast group.
- EXT_DECL VOS_ERR_T vos_sockLeaveMC (SOCKET sock, UINT32 mcAddress, UINT32 ipAddress)

 Leave a multicast group.
- EXT_DECL VOS_ERR_T vos_sockSendUDP (SOCKET sock, const UINT8 *pBuffer, UINT32 *pSize, UIN← T32 ipAddress, UINT16 port)

Send UDP data.

EXT_DECL VOS_ERR_T vos_sockReceiveUDP (SOCKET sock, UINT8 *pBuffer, UINT32 *pSize, UINT32 *pSrcIPAddr, UINT16 *pSrcIPPort, UINT32 *pDstIPAddr, BOOL8 peek)

Receive UDP data.

EXT_DECL VOS_ERR_T vos_sockBind (SOCKET sock, UINT32 ipAddress, UINT16 port)

Bind a socket to an address and port.

EXT_DECL VOS_ERR_T vos_sockListen (SOCKET sock, UINT32 backlog)

Listen for incoming TCP connections.

EXT_DECL VOS_ERR_T vos_sockAccept (SOCKET sock, SOCKET *pSock, UINT32 *pIPAddress, UINT16 *pPort)

Accept an incoming TCP connection.

- EXT_DECL VOS_ERR_T vos_sockConnect (SOCKET sock, UINT32 ipAddress, UINT16 port)

 Open a TCP connection.
- EXT_DECL VOS_ERR_T vos_sockSendTCP (SOCKET sock, const UINT8 *pBuffer, UINT32 *pSize)
 Send TCP data.
- EXT_DECL VOS_ERR_T vos_sockReceiveTCP (SOCKET sock, UINT8 *pBuffer, UINT32 *pSize)

 Receive TCP data.
- EXT_DECL VOS_ERR_T vos_sockSetMulticastIf (SOCKET sock, UINT32 mclfAddress)
- Set Using Multicast I/F.

 EXT_DECL_VOS_IP4_ADDR_T_vos_determineBindAddr_(VOS_IP4_ADDR_T_srcIP, VOS_IP4_ADDR_←
 T mcGroup, VOS_IP4_ADDR_T rcvMostly)

Determines the address to bind to since the behaviour in the different OS is different.

5.43.1 Detailed Description

Typedefs for OS abstraction.

This is the declaration for the OS independend socket interface

Note

Project: TCNOpen TRDP prototype stack

Author

Bernd Loehr, NewTec GmbH

Remarks

This Source Code Form is subject to the terms of the Mozilla Public License, v. 2.0. If a copy of the MPL was not distributed with this file, You can obtain one at http://mozilla.org/MPL/2.0/. Copyright Bombardier Transportation Inc. or its subsidiaries and others, 2013. All rights reserved.

5.43.2 Macro Definition Documentation

5.43.2.1 VOS_MAX_SOCKET_CNT

```
#define VOS_MAX_SOCKET_CNT 4
```

The maximum number of sockets influences memory usage; for small systems we should define a smaller set.

The maximum number of concurrent usable sockets per application session

5.43.2.2 VOS_TTL_MULTICAST

```
#define VOS_TTL_MULTICAST 64
```

The maximum number of hops a multicast packet can take.

The maximum size for the interface name

5.43.3 Function Documentation

5.43.3.1 vos_determineBindAddr()

Determines the address to bind to since the behaviour in the different OS is different.

Parameters

in	srcIP	IP to bind to (0 = any address)
in	mcGroup	MC group to join (0 = do not join)
in	rcvMostly	primarily used for receiving (tbd: bind on sender, too?)

Return values

5.43.3.2 vos_dottedIP()

Convert IP address from dotted dec.

to !host! endianess

Parameters

in	p⊷	IP address as dotted decimal.
	DottedIP	

Return values

address	in UINT32 in host endianess
---------	-----------------------------

5.43.3.3 vos_getInterfaces()

Get a list of interface addresses The caller has to provide an array of interface records to be filled.

Parameters

in,out	pAddrCnt	in: pointer to array size of interface record out: pointer to number of interface records read
in,out	ifAddrs	array of interface records

Return values

VOS_NO_ERR	no error
VOS_PARAM_ERR	pAddrCnt and/or ifAddrs == NULL
VOS_MEM_ERR	memory allocation error
VOS_SOCK_ERR	GetAdaptersInfo() error

5.43.3.4 vos_htonl()

```
EXT_DECL UINT32 vos_htonl ( UINT32 val )
```

Byte swapping 4 Bytes.

Parameters

in	val	Initial value.
----	-----	----------------

Return values

swapped	value
---------	-------

5.43.3.5 vos_htonII()

Byte swapping 8 Bytes.

Parameters

in	val	Initial value.

Return values

```
swapped value
```

5.43.3.6 vos_htons()

Byte swapping 2 Bytes.

Parameters

in <i>val</i> Initial val	lue.
---------------------------	------

Return values

swapped	value
---------	-------

5.43.3.7 vos_ipDotted()

Convert IP address to dotted dec.

from !host! endianess

Parameters

in	ipAddress	address in UINT32 in host endianess
----	-----------	-------------------------------------

Return values

```
IP address as dotted decimal.
```

5.43.3.8 vos_isMulticast()

```
EXT_DECL BOOL8 vos_isMulticast ( {\tt UINT32\ \it ipAddress\ )}
```

Check if the supplied address is a multicast group address.

Parameters

in	ipAddress	IP address to check.

Return values

TRUE	address is a multicast address
FALSE	address is not a multicast address

5.43.3.9 vos_netIfUp()

```
EXT_DECL BOOL8 vos_netIfUp ( {\tt VOS\_IP4\_ADDR\_T~\it ifAddress~)}
```

Get the state of an interface.

Parameters

in	ifAddress	address of interface to check

Return values

TRUE interface is up and ready FALSE interface is down / not ready

5.43.3.10 vos_ntohl()

```
EXT_DECL UINT32 vos_ntohl ( UINT32 val )
```

Byte swapping 4 Bytes.

Parameters

in <i>val</i>	Initial value.
---------------	----------------

Return values

swapped value

5.43.3.11 vos_ntohll()

Byte swapping 8 Bytes.

Parameters

in <i>val</i> I	nitial value.
-----------------	---------------

Return values

```
swapped value
```

5.43.3.12 vos_ntohs()

Byte swapping 2 Bytes.

Parameters

Return values

```
swapped value
```

5.43.3.13 vos_select()

```
EXT_DECL INT32 vos_select (

SOCKET highDesc,

VOS_FDS_T * pReadableFD,

VOS_FDS_T * pWriteableFD,

VOS_FDS_T * pErrorFD,

VOS_TIMEVAL_T * pTimeOut )
```

select function.

Set the ready sockets in the supplied sets. Note: Some target systems might define this function as NOP.

Parameters

in	highDesc	max. socket descriptor + 1
in,out	pReadableFD	pointer to readable socket set
in,out	pWriteableFD	pointer to writeable socket set
in,out	pErrorFD	pointer to error socket set
in	pTimeOut	pointer to time out value

Return values

```
number of ready file descriptors
```

5.43.3.14 vos_sockAccept()

Accept an incoming TCP connection.

Accept incoming connections on the provided socket. May block and will return a new socket descriptor when accepting a connection. The original socket *pSock, remains open.

Parameters

in	sock	Socket descriptor
out	pSock	Pointer to socket descriptor, on exit new socket
out	pIPAddress	source IP to receive on, 0 for any
out	pPort	port to receive on, 17224 for PD

Return values

VOS_NO_ERR	no error
VOS_PARAM_ERR	NULL parameter, parameter error
VOS_UNKNOWN_ERR	sock descriptor unknown error

5.43.3.15 vos_sockBind()

Bind a socket to an address and port.

Parameters

in	sock	socket descriptor
in	ipAddress	source IP to receive from, 0 for any
in	port	port to receive from

Return values

VOS_NO_ERR	no error
VOS_PARAM_ERR	parameter out of range/invalid
VOS_IO_ERR	Input/Output error
VOS_MEM_ERR	resource error

5.43.3.16 vos_sockClose()

Close a socket.

Release any resources aquired by this socket

Parameters

in	sock	socket descriptor
----	------	-------------------

Return values

VOS_NO_ERR	no error
VOS_PARAM_ERR	pSock == NULL

5.43.3.17 vos_sockConnect()

```
EXT_DECL VOS_ERR_T vos_sockConnect (

SOCKET sock,

UINT32 ipAddress,

UINT16 port )
```

Open a TCP connection.

Parameters

in	sock	socket descriptor
in	ipAddress	destination IP
in	port	destination port

Return values

VOS_NO_ERR	no error
VOS_PARAM_ERR	parameter out of range/invalid
VOS_IO_ERR	Input/Output error

5.43.3.18 vos_sockGetMAC()

Return the MAC address of the default adapter.

Parameters

out pMAC return MAC address.

Return values

VOS_NO_ERR	no error
VOS_PARAM_ERR	pMAC == NULL

Return values

VOS_SOCK_ERR	socket not available or option not supported	
--------------	--	--

5.43.3.19 vos_socklnit()

Initialize the socket library.

Must be called once before any other call

Return values

VOS_NO_ERR	no error
VOS_SOCK_ERR	sockets not supported

5.43.3.20 vos_sockJoinMC()

Join a multicast group.

Note: Some target systems might not support this option.

Parameters

in	sock	socket descriptor
in	mcAddress	multicast group to join
in	ipAddress	depicts interface on which to join, default 0 for any

Return values

VOS_NO_ERR	no error
VOS_PARAM_ERR	parameter out of range/invalid
VOS_SOCK_ERR	option not supported

5.43.3.21 vos_sockLeaveMC()

```
EXT_DECL VOS_ERR_T vos_sockLeaveMC (
```

```
SOCKET sock,
UINT32 mcAddress,
UINT32 ipAddress )
```

Leave a multicast group.

Note: Some target systems might not support this option.

Parameters

in	sock	socket descriptor	
in	mcAddress	multicast group to join	
in ipAddress depicts interface on which to leave, default 0 for		depicts interface on which to leave, default 0 for any	

Return values

VOS_NO_ERR	no error
VOS_INIT_ERR	module not initialised
VOS_NOINIT_ERR	invalid handle
VOS_PARAM_ERR	parameter out of range/invalid
VOS_SOCK_ERR	option not supported

5.43.3.22 vos_sockListen()

Listen for incoming TCP connections.

Parameters

in	sock	socket descriptor
in	backlog	maximum connection attempts if system is busy

Return values

VOS_NO_ERR	no error
VOS_PARAM_ERR	parameter out of range/invalid
VOS_IO_ERR	Input/Output error
VOS_MEM_ERR	resource error

5.43.3.23 vos_sockOpenTCP()

Create a TCP socket.

Return a socket descriptor for further calls. The socket options are optional and can be applied later.

Parameters

out	pSock	pointer to socket descriptor returned
in	pOptions	pointer to socket options (optional)

Return values

VOS_NO_ERR	no error
VOS_PARAM_ERR	pSock == NULL
VOS_SOCK_ERR	socket not available or option not supported

5.43.3.24 vos_sockOpenUDP()

Create an UDP socket.

Return a socket descriptor for further calls. The socket options are optional and can be applied later. Note: Some target systems might not support every option.

Parameters

С	out	pSock	pointer to socket descriptor returned
i	n	pOptions	pointer to socket options (optional)

Return values

VOS_NO_ERR	no error
VOS_PARAM_ERR	pSock == NULL
VOS_SOCK_ERR	socket not available or option not supported

5.43.3.25 vos_sockReceiveTCP()

Receive TCP data.

The caller must provide a sufficient sized buffer. If the supplied buffer is smaller than the bytes received, *pSize will reflect the number of copied bytes and the call should be repeated until *pSize is 0 (zero). If the socket was created in blocking-mode (default), then this call will block and will only return if data has been received or the socket was closed or an error occured. If called in non-blocking mode, and no data is available, VOS_NODATA_ERR will be returned.

Parameters

in	sock	socket descriptor
out	pBuffer	pointer to applications data buffer
in,out	pSize	pointer to the received data size

Return values

VOS_NO_ERR	no error	
VOS_PARAM_ERR	sock descriptor unknown, parameter error	
VOS_IO_ERR	data could not be read	
VOS_NODATA_ERR	no data in non-blocking	
VOS_BLOCK_ERR	call would have blocked in blocking mode	

5.43.3.26 vos_sockReceiveUDP()

```
EXT_DECL VOS_ERR_T vos_sockReceiveUDP (

SOCKET sock,

UINT8 * pBuffer,

UINT32 * pSize,

UINT32 * pSrcIPAddr,

UINT16 * pSrcIPPort,

UINT32 * pDstIPAddr,

BOOL8 peek )
```

Receive UDP data.

The caller must provide a sufficient sized buffer. If the supplied buffer is smaller than the bytes received, *pSize will reflect the number of copied bytes and the call should be repeated until *pSize is 0 (zero). If the socket was created in blocking-mode (default), then this call will block and will only return if data has been received or the socket was closed or an error occured. If called in non-blocking mode, and no data is available, VOS_NODATA_ERR will be returned. If pointers are provided, source IP, source port and destination IP will be reported on return.

Parameters

in	sock	socket descriptor
out	pBuffer	pointer to applications data buffer
in,out	pSize	pointer to the received data size
out	pSrcIPAddr	pointer to source IP
out	pSrcIPPort	pointer to source port
out	pDstIPAddr	pointer to dest IP
in	peek	if true, leave data in queue

Return values

VOS_NO_ERR	no error	
VOS_PARAM_ERR	sock descriptor unknown, parameter error	
VOS_IO_ERR	data could not be read	
VOS_NODATA_ERR	no data	
VOS_BLOCK_ERR	Call would have blocked in blocking mode	

5.43.3.27 vos_sockSendTCP()

Send TCP data.

Send data to the supplied address and port.

Parameters

in	sock	socket descriptor	
in	pBuffer	pointer to data to send	
in,out	pSize	In: size of the data to send, Out: no of bytes sent	

Return values

VOS_NO_ERR	no error
VOS_PARAM_ERR	sock descriptor unknown, parameter error
VOS_IO_ERR	data could not be sent
VOS_NOCONN_ERR	no TCP connection
VOS_BLOCK_ERR	call would have blocked in blocking mode, data partially sent

5.43.3.28 vos_sockSendUDP()

Send UDP data.

Send data to the given address and port.

Parameters

in	sock	socket descriptor	
in	pBuffer	uffer pointer to data to send	
in,out	pSize In: size of the data to send, Out: no of bytes se		
in	ipAddress	destination IP	
in	port	destination port	

Return values

VOS_NO_ERR	no error	
VOS_PARAM_ERR	parameter out of range/invalid	
VOS_IO_ERR	data could not be sent	
VOS_BLOCK_ERR	Call would have blocked in blocking mode	

5.43.3.29 vos_sockSetMulticastIf()

Set Using Multicast I/F.

Parameters

Ī	in	sock	socket descriptor
Ī	in	mclfAddress	using Multicast I/F Address

Return values

VOS_NO_ERR	no error
VOS_PARAM_ERR	sock descriptor unknown, parameter error

5.43.3.30 vos_sockSetOptions()

```
EXT_DECL VOS_ERR_T vos_sockSetOptions ( {\tt SOCKET}\ sock, {\tt const}\ {\tt VOS\_SOCK\_OPT\_T}\ *\ pOptions\ )
```

Set socket options.

Note: Some target systems might not support each option.

Parameters

in	sock	socket descriptor	l
in	pOptions	pointer to socket options (optional)	L
	, ,		

Generated by Doxygen

Return values

VOS_NO_ERR	no error
VOS_PARAM_ERR	parameter out of range/invalid

5.43.3.31 vos_sockTerm()

De-Initialize the socket library.

Must be called after last socket call

5.44 vos_thread.h File Reference

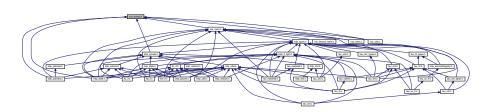
Threading functions for OS abstraction.

```
#include "vos_types.h"
#include <time.h>
Include dependency graph for vos_thread.h:
```

vos_thread.h
vos_types.h time.h

stdint.h

This graph shows which files directly or indirectly include this file:



Macros

#define VOS_MAX_THREAD_CNT 100

The maximum number of concurrent usable threads.

#define VOS_SEMA_WAIT_FOREVER 0xFFFFFFFU

Timeout value to wait forever for a semaphore.

Typedefs

typedef UINT8 VOS_THREAD_PRIORITY_T

Thread priority range from 1 (lowest) to 255 (highest), 0 default of the target system.

typedef void(__cdecl * VOS_THREAD_FUNC_T) (void *pArg)

Thread function definition.

typedef struct VOS_MUTEX * VOS_MUTEX_T

Hidden mutex handle definition.

typedef struct VOS_SEMA * VOS_SEMA_T

Hidden semaphore handle definition.

typedef void * VOS_THREAD_T

Hidden thread handle definition.

Enumerations

enum VOS_THREAD_POLICY_T

Thread policy matching pthread/Posix defines.

• enum VOS_SEMA_STATE_T

State of the semaphore.

Functions

• EXT_DECL VOS_ERR_T vos_threadInit (void)

Initialize the thread library.

EXT_DECL void vos_threadTerm (void)

De-Initialize the thread library.

EXT_DECL VOS_ERR_T vos_threadCreateSync (VOS_THREAD_T *pThread, const CHAR8 *pName, V←
 OS_THREAD_POLICY_T policy, VOS_THREAD_PRIORITY_T priority, UINT32 interval, VOS_TIMEVAL_T
 *pStartTime, UINT32 stackSize, VOS_THREAD_FUNC_T pFunction, void *pArguments)

Create a thread.

• EXT_DECL VOS_ERR_T vos_threadCreate (VOS_THREAD_T *pThread, const CHAR8 *pName, VOS_← THREAD_POLICY_T policy, VOS_THREAD_PRIORITY_T priority, UINT32 interval, UINT32 stackSize, V← OS_THREAD_FUNC_T pFunction, void *pArguments)

Create a thread.

EXT_DECL VOS_ERR_T vos_threadTerminate (VOS_THREAD_T thread)

Terminate a thread.

• EXT DECL VOS ERR T vos threadlsActive (VOS THREAD T thread)

Is the thread still active? This call will return VOS_NO_ERR if the thread is still active, VOS_PARAM_ERR in case it ran out.

EXT DECL VOS ERR T vos threadDelay (UINT32 delay)

Delay the execution of the current thread by the given delay in us.

EXT_DECL VOS_ERR_T vos_threadSelf (VOS_THREAD_T *pThread)

Return thread handle of calling task.

EXT_DECL void vos_getTime (VOS_TIMEVAL_T *pTime)

Return the current monotonic time in sec and us.

EXT_DECL void vos_getRealTime (VOS_TIMEVAL_T *pTime)

Return the current real time in sec and us.

EXT_DECL const CHAR8 * vos_getTimeStamp (void)

Get a time-stamp string.

EXT DECL void vos clearTime (VOS TIMEVAL T*pTime)

Clear the time stamp.

• EXT_DECL void vos_addTime (VOS_TIMEVAL_T *pTime, const VOS_TIMEVAL_T *pAdd)

Add the second to the first time stamp, return sum in first.

EXT_DECL void vos_subTime (VOS_TIMEVAL_T *pTime, const VOS_TIMEVAL_T *pSub)

Subtract the second from the first time stamp, return diff in first.

EXT_DECL INT32 vos_cmpTime (const VOS_TIMEVAL_T *pTime, const VOS_TIMEVAL_T *pCmp)

Compare the second from the first time stamp, return diff in first.

EXT_DECL void vos_divTime (VOS_TIMEVAL_T *pTime, UINT32 divisor)

Divide the first time by the second, return quotient in first.

EXT_DECL void vos_mulTime (VOS_TIMEVAL_T *pTime, UINT32 mul)

Multiply the first time by the second, return product in first.

• EXT_DECL void vos_getUuid (VOS_UUID_T pUuID)

Get a universal unique identifier according to RFC 4122 time based version.

EXT_DECL VOS_ERR_T vos_mutexCreate (VOS_MUTEX_T *pMutex)

Create a mutex.

EXT_DECL void vos_mutexDelete (VOS_MUTEX_T pMutex)

Delete a mutex.

• EXT_DECL VOS_ERR_T vos_mutexLock (VOS_MUTEX_T pMutex)

Take a mutex.

EXT_DECL VOS_ERR_T vos_mutexTryLock (VOS_MUTEX_T pMutex)

Try to take a mutex.

EXT_DECL VOS_ERR_T vos_mutexUnlock (VOS_MUTEX_T pMutex)

Release a mutex.

 $\bullet \ \ \mathsf{EXT_DECL} \ \ \mathsf{VOS_ERR_T} \ \ \mathsf{vos_semaCreate} \ \ (\mathsf{VOS_SEMA_T} \ *\mathsf{pSema}, \ \mathsf{VOS_SEMA_STATE_T} \ \mathsf{initialState})$

Create a semaphore.

EXT_DECL void vos_semaDelete (VOS_SEMA_T sema)

Delete a semaphore.

EXT_DECL VOS_ERR_T vos_semaTake (VOS_SEMA_T sema, UINT32 timeout)

Take a semaphore.

EXT_DECL void vos_semaGive (VOS_SEMA_T sema)

Give a semaphore.

5.44.1 Detailed Description

Threading functions for OS abstraction.

Thread-, semaphore- and time-handling functions

Note

Project: TCNOpen TRDP prototype stack

Author

Bernd Loehr, NewTec GmbH

Remarks

This Source Code Form is subject to the terms of the Mozilla Public License, v. 2.0. If a copy of the MPL was not distributed with this file, You can obtain one at http://mozilla.org/MPL/2.0/. Copyright Bombardier Transportation Inc. or its subsidiaries and others, 2014. All rights reserved.

5.44.2 Function Documentation

5.44.2.1 vos_addTime()

Add the second to the first time stamp, return sum in first.

Parameters

in,out	pTime	Pointer to time value
in	pAdd	Pointer to time value

5.44.2.2 vos_clearTime()

Clear the time stamp.

Parameters

out <i>pTime</i>	Pointer to time value
------------------	-----------------------

5.44.2.3 vos_cmpTime()

Compare the second from the first time stamp, return diff in first.

Parameters

in,out	pTime	Pointer to time value
in	рСтр	Pointer to time value to compare

Return values

0	pTime == pCmp
-1	pTime < pCmp
1	pTime > pCmp

5.44.2.4 vos_divTime()

Divide the first time by the second, return quotient in first.

Parameters

in,out	pTime	Pointer to time value
in	divisor	Divisor

5.44.2.5 vos_getRealTime()

Return the current real time in sec and us.

Parameters

out	pTime	Pointer to time value

5.44.2.6 vos_getTime()

Return the current monotonic time in sec and us.

Parameters

out <i>pTime</i> Poi	nter to time value
----------------------	--------------------

5.44.2.7 vos_getTimeStamp()

Get a time-stamp string.

Get a time-stamp string for debugging in the form "yyyymmdd-hh:mm:ss.ms" Depending on the used OS / hardware the time might not be a real-time stamp but relative from start of system.

Return values

timestamp	"yyyymmdd-hh:mm:ss.ms"
-----------	------------------------

5.44.2.8 vos_getUuid()

Get a universal unique identifier according to RFC 4122 time based version.

Parameters

	out	pUuID	Pointer to a universal unique identifier
--	-----	-------	--

5.44.2.9 vos_mulTime()

Multiply the first time by the second, return product in first.

Parameters

in,out	pTime	Pointer to time value
in	mul	Factor

5.44.2.10 vos_mutexCreate()

Create a mutex.

Return a mutex handle. The mutex will be available at creation.

Parameters

out	pMutex	Pointer to mutex handle
-----	--------	-------------------------

Return values

VOS_NO_ERR	no error
VOS_INIT_ERR	module not initialised
VOS_PARAM_ERR	pMutex == NULL
VOS_MUTEX_ERR	no mutex available

5.44.2.11 vos_mutexDelete()

Delete a mutex.

Release the resources taken by the mutex.

Parameters

in <i>pMutex</i>	mutex handle
------------------	--------------

Return values

```
VOS_NO_ERR no error
```

5.44.2.12 vos_mutexLock()

Take a mutex.

Wait for the mutex to become available (lock).

Parameters

in p	Mutex	mutex	handle
------	-------	-------	--------

Return values

VOS_NO_ERR	no error
VOS_INIT_ERR	module not initialised
VOS_NOINIT_ERR	invalid handle

5.44.2.13 vos_mutexTryLock()

Try to take a mutex.

If mutex is can't be taken VOS_MUTEX_ERR is returned.

Parameters

in	pMutex	mutex handle
----	--------	--------------

Return values

VOS_NO_ERR	no error
VOS_INIT_ERR	module not initialised
VOS_NOINIT_ERR	invalid handle
VOS_MUTEX_ERR	no mutex available

5.44.2.14 vos_mutexUnlock()

Release a mutex.

Unlock the mutex.

Parameters

in	pMutex	mutex handle

5.44.2.15 vos_semaCreate()

Create a semaphore.

Return a semaphore handle. Depending on the initial state the semaphore will be available on creation or not.

Parameters

out	pSema	Pointer to semaphore handle
in	initialState	The initial state of the sempahore

Return values

VOS_NO_ERR	no error
VOS_INIT_ERR	module not initialised
VOS_PARAM_ERR	parameter out of range/invalid
VOS_SEMA_ERR	no semaphore available

5.44.2.16 vos_semaDelete()

Delete a semaphore.

This will eventually release any processes waiting for the semaphore.

Parameters

in	sema	semaphore handle
----	------	------------------

5.44.2.17 vos_semaGive()

Give a semaphore.

Release (increase) a semaphore.

Parameters

in	sema	semaphore handle
----	------	------------------

5.44.2.18 vos_semaTake()

Take a semaphore.

Try to get (decrease) a semaphore.

Parameters

in	sema	semaphore handle	
in	timeout	Max. time in us to wait, 0 means no wait	

Return values

VOS_NO_ERR	no error
VOS_INIT_ERR	module not initialised
VOS_NOINIT_ERR	invalid handle
VOS_PARAM_ERR	parameter out of range/invalid
VOS_SEMA_ERR	could not get semaphore in time

5.44.2.19 vos_subTime()

Subtract the second from the first time stamp, return diff in first.

Parameters

in,out	pTime	Pointer to time value
in	pSub	Pointer to time value

5.44.2.20 vos_threadCreate()

```
VOS_THREAD_PRIORITY_T priority,
UINT32 interval,
UINT32 stackSize,
VOS_THREAD_FUNC_T pFunction,
void * pArguments )
```

Create a thread.

Create a thread and return a thread handle for further requests. Not each parameter may be supported by all target systems!

Parameters

out	pThread	Pointer to returned thread handle
in	pName	Pointer to name of the thread (optional)
in	policy	Scheduling policy (FIFO, Round Robin or other)
in	priority	Scheduling priority (1255 (highest), default 0)
in	interval	Interval for cyclic threads in us (optional)
in	stackSize	Minimum stacksize, default 0: 16kB
in	pFunction	Pointer to the thread function
in	pArguments	Pointer to the thread function parameters

Return values

VOS_NO_ERR	no error
VOS_INIT_ERR	module not initialised
VOS_NOINIT_ERR	invalid handle
VOS_PARAM_ERR	parameter out of range/invalid

5.44.2.21 vos_threadCreateSync()

Create a thread.

Create a thread and return a thread handle for further requests. Not each parameter may be supported by all target systems!

Parameters

out	pThread	Pointer to returned thread handle
in	pName	Pointer to name of the thread (optional)

Parameters

in	policy	Scheduling policy (FIFO, Round Robin or other)
in	priority	Scheduling priority (1255 (highest), default 0)
in	interval	Interval for cyclic threads in us (optional)
in	pStartTime	Starting time for cyclic threads
in	stackSize	Minimum stacksize, default 0: 16kB
in	pFunction	Pointer to the thread function
in	pArguments	Pointer to the thread function parameters

Return values

VOS_NO_ERR	no error
VOS_INIT_ERR	module not initialised
VOS_NOINIT_ERR	invalid handle
VOS_PARAM_ERR	parameter out of range/invalid

5.44.2.22 vos_threadDelay()

Delay the execution of the current thread by the given delay in us.

Parameters

in	delay	Delay in us

Return values

VOS_NO_ERR	no error
VOS_INIT_ERR	module not initialised

5.44.2.23 vos_threadInit()

Initialize the thread library.

Must be called once before any other call

Return values

VOS_NO_ERR	no error
VOS_INIT_ERR	threading not supported

5.44.2.24 vos_threadlsActive()

Is the thread still active? This call will return VOS_NO_ERR if the thread is still active, VOS_PARAM_ERR in case it ran out.

Parameters

Return values

VOS_NO_ERR	no error
VOS_INIT_ERR	module not initialised
VOS_NOINIT_ERR	invalid handle
VOS_PARAM_ERR	parameter out of range/invalid

5.44.2.25 vos_threadSelf()

Return thread handle of calling task.

Parameters

out	pThread	pointer to thread handle

Return values

VOS_NO_ERR	no error
VOS_PARAM_ERR	parameter out of range/invalid

5.44.2.26 vos_threadTerm()

```
\begin{tabular}{ll} {\tt EXT\_DECL} & {\tt void} & {\tt vos\_threadTerm} & (\\ & & {\tt void} & ) \end{tabular}
```

De-Initialize the thread library.

Must be called after last thread/timer call

5.44.2.27 vos_threadTerminate()

Terminate a thread.

This call will terminate the thread with the given threadld and release all resources. Depending on the underlying architectures, it may just block until the thread ran out.

Parameters

	in	thread	Thread handle (or NULL if current thread)	1
--	----	--------	---	---

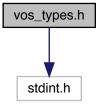
Return values

VOS_NO_ERR	no error
VOS_INIT_ERR	module not initialised
VOS_NOINIT_ERR	invalid handle
VOS_PARAM_ERR	parameter out of range/invalid

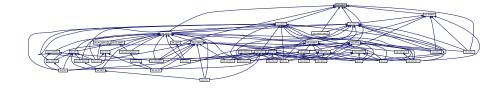
5.45 vos_types.h File Reference

Typedefs for OS abstraction.

```
#include <stdint.h>
Include dependency graph for vos_types.h:
```



This graph shows which files directly or indirectly include this file:



Data Structures

struct VOS_VERSION_T
 Version information.

Macros

• #define INLINE inline

inline macros

#define AV ERROR 0x00

ANTIVALENT8 values.

#define TR_DIR1 0x01

Directions/Orientations.

Typedefs

typedef UINT8 VOS_UUID_T[16]

universal unique identifier according to RFC 4122, time based version

typedef struct timeval VOS_TIMEVAL_T

Timer value compatible with timeval / select.

 typedef void(* VOS_PRINT_DBG_T) (void *pRefCon, VOS_LOG_T category, const CHAR8 *pTime, const CHAR8 *pFile, UINT16 LineNumber, const CHAR8 *pMsgStr)

Function definition for error/debug output.

Enumerations

```
enum VOS_ERR_T {
 VOS NO ERR = 0,
 VOS_PARAM_ERR = -1,
 VOS_INIT_ERR = -2,
 VOS_NOINIT_ERR = -3,
 VOS TIMEOUT ERR = -4,
 VOS NODATA ERR = -5,
 VOS_SOCK_ERR = -6,
 VOS_IO_ERR = -7,
 VOS MEM ERR = -8,
 VOS SEMA_ERR = -9,
 VOS_QUEUE_ERR = -10,
 VOS_QUEUE_FULL_ERR = -11,
 VOS MUTEX ERR = -12,
 VOS THREAD_ERR = -13,
 VOS BLOCK ERR = -14,
 VOS INTEGRATION ERR = -15,
 VOS NOCONN ERR = -16,
 VOS INUSE ERR = -49,
 VOS_UNKNOWN_ERR = -99 }
    Return codes for all VOS API functions.
enum VOS_LOG_T {
 VOS LOG ERROR = 0,
 VOS LOG WARNING = 1,
 VOS LOG INFO = 2,
 VOS LOG DBG = 3,
 VOS_LOG_USR = 4 }
```

Categories for logging.

5.45.1 Detailed Description

Typedefs for OS abstraction.

Note

Project: TCNOpen TRDP prototype stack

Author

Bernd Loehr, NewTec GmbH

Remarks

This Source Code Form is subject to the terms of the Mozilla Public License, v. 2.0. If a copy of the MPL was not distributed with this file, You can obtain one at http://mozilla.org/MPL/2.0/. Copyright Bombardier Transportation Inc. or its subsidiaries and others, 2013. All rights reserved.

5.45.2 Typedef Documentation

5.45.2.1 VOS_PRINT_DBG_T

```
typedef void(* VOS_PRINT_DBG_T) (void *pRefCon, VOS_LOG_T category, const CHAR8 *pTime, const
CHAR8 *pFile, UINT16 LineNumber, const CHAR8 *pMsgStr)
```

Function definition for error/debug output.

The function will be called for logging and error message output. The user can decide, what kind of info will be logged by filtering the category.

Parameters

in	pRefCon	pointer to user context
in	category	Log category (Error, Warning, Info etc.)
in	pTime	pointer to NULL-terminated string of time stamp
in	pFile	pointer to NULL-terminated string of source module
in	LineNumber	Line number
in	pMsgStr	pointer to NULL-terminated string

5.45.2.2 VOS_TIMEVAL_T

typedef struct timeval VOS_TIMEVAL_T

Timer value compatible with timeval / select.

Relative or absolute date, depending on usage Assume 32 Bit system, if not defined

5.45.3 Enumeration Type Documentation

5.45.3.1 VOS_ERR_T

enum VOS_ERR_T

Return codes for all VOS API functions.

Enumerator

VOS_NO_ERR	No error.
VOS_PARAM_ERR	Necessary parameter missing or out of range.
VOS_INIT_ERR	Call without valid initialization.
VOS_NOINIT_ERR	The supplied handle/reference is not valid.
VOS_TIMEOUT_ERR	Timout.
VOS_NODATA_ERR	Non blocking mode: no data received.
VOS_SOCK_ERR	Socket option not supported.
VOS_IO_ERR	Socket IO error, data can't be received/sent.
VOS_MEM_ERR	No more memory available.
VOS_SEMA_ERR	Semaphore not available.
VOS_QUEUE_ERR	Queue empty.
VOS_QUEUE_FULL_ERR	Queue full.
VOS_MUTEX_ERR	Mutex not available.
VOS_THREAD_ERR	Thread creation error.
VOS_BLOCK_ERR	System call would have blocked in blocking mode.
VOS_INTEGRATION_ERR	Alignment or endianess for selected target wrong.
VOS_NOCONN_ERR	No TCP connection.
VOS_INUSE_ERR	Resource is still in use.
VOS_UNKNOWN_ERR	Unknown error.

5.45.3.2 VOS_LOG_T

enum VOS_LOG_T

Categories for logging.

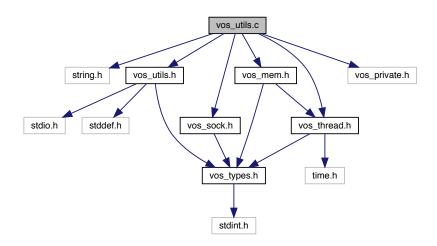
Enumerator

VOS_LOG_ERROR	This is a critical error.
VOS_LOG_WARNING	This is a warning.
VOS_LOG_INFO	This is an info.
VOS_LOG_DBG	This is a debug info.
VOS_LOG_USR	This is a user info.

5.46 vos_utils.c File Reference

Common functions for VOS.

```
#include <string.h>
#include "vos_utils.h"
#include "vos_sock.h"
#include "vos_thread.h"
#include "vos_mem.h"
#include "vos_private.h"
Include dependency graph for vos_utils.c:
```



Functions

• int vos_hostIsBigEndian ()

Return 1 if this is a big endian machine.

VOS_ERR_T vos_init (void *pRefCon, VOS_PRINT_DBG_T pDebugOutput)

Initialize the virtual operating system.

EXT_DECL void vos_terminate (void)

Delnitialize the vos library.

• UINT32 vos_crc32 (UINT32 crc, const UINT8 *pData, UINT32 dataLen)

Compute crc32 according to IEEE802.3.

• UINT32 vos_sc32 (UINT32 crc, const UINT8 *pData, UINT32 dataLen)

Compute crc32 according to IEC 61375-2-3 B.7.

const char * vos_getVersionString (void)

Return a human readable version representation.

• EXT_DECL const VOS_VERSION_T * vos_getVersion (void)

Return version.

EXT_DECL const CHAR8 * vos_getErrorString (VOS_ERR_T error)

Return a human readable error representation.

5.46.1 Detailed Description

Common functions for VOS.

Common functions of the abstraction layer. Mainly debugging support.

Note

Project: TCNOpen TRDP prototype stack

Author

Bernd Loehr, NewTec GmbH

Remarks

This Source Code Form is subject to the terms of the Mozilla Public License, v. 2.0. If a copy of the MPL was not distributed with this file, You can obtain one at http://mozilla.org/MPL/2.0/. Copyright Bombardier Transportation Inc. or its subsidiaries and others, 2013. All rights reserved.

5.46.2 Function Documentation

5.46.2.1 vos_crc32()

Compute crc32 according to IEEE802.3.

Calculate CRC for the given buffer and length.

/ to IEC 61375-2-3 A.3 Note: Returned CRC is inverted

Parameters

in	crc	Initial value.
in,out	pData	Pointer to data.
in	dataLen	length in bytes of data.

Return values

crc32	according to	
	IEEE802.3	

5.46.2.2 vos_getErrorString()

Return a human readable error representation.

Parameters

in	error	The TRDP or VOS error code
----	-------	----------------------------

Return values

const	string pointer to error string
-------	--------------------------------

5.46.2.3 vos_getVersion()

Return version.

Return pointer to version structure

Return values

```
VOS_VERSION↔
T
```

5.46.2.4 vos_getVersionString()

Return a human readable version representation.

Return string in the form 'v.r.u.b'

Return values

const string

5.46.2.5 vos_hostIsBigEndian()

Return 1 if this is a big endian machine.

Return values

0	if machine is little endian
1	if machine is big endian

5.46.2.6 vos_init()

Initialize the virtual operating system.

Initialize the vos library.

Parameters

in	pRefCon	context for debug output function
in	pDebugOutput	Pointer to debug output function.

Return values

VOS_NO_ERR	no error VOS_INTEGRATION_ERR if endianess/alignment mismatch VOS_SOCK_ERR	
	sockets not supported VOS_UNKNOWN_ERR initialisation error	

5.46.2.7 vos_sc32()

Compute crc32 according to IEC 61375-2-3 B.7.

Compute crc32 according to IEC 61375-2-3 B.7 Note: Returned CRC is inverted.

Parameters

in	crc	Initial value.	
in,out	pData	Pointer to data.	
in	dataLen	length in bytes of data.	

Return values

sc32	according to IEC 61375-2-3
------	----------------------------

5.46.2.8 vos_terminate()

```
\begin{tabular}{ll} {\tt EXT\_DECL} & {\tt void} & {\tt vos\_terminate} & (\\ & & {\tt void} & ) \end{tabular}
```

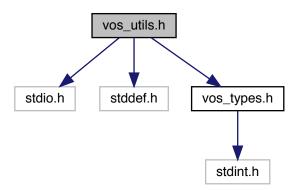
DeInitialize the vos library.

Should be called last after TRDP stack/application does not use any VOS function anymore.

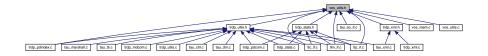
5.47 vos_utils.h File Reference

Typedefs for OS abstraction.

```
#include <stdio.h>
#include <stddef.h>
#include "vos_types.h"
Include dependency graph for vos_utils.h:
```



This graph shows which files directly or indirectly include this file:



Macros

#define VOS MAX PRNT STR SIZE 256u

String size definitions for the debug output functions.

• #define VOS MAX FRMT SIZE 64u

Max

• #define VOS MAX ERR STR SIZE (VOS MAX PRNT STR SIZE - VOS MAX FRMT SIZE)

Мах.

#define VOS_DIR_SEP '/'

This is a helper define for separating a path in debug output.

#define vos_snprintf(str, size, format, args ...) snprintf(str, size, format, ## args) /*lint le586 logging output needed */

Safe printf function.

#define vos_printLogStr(level, string)

Debug output macro without formatting options.

#define vos_printLog(level, format, args ...)

Debug output macro with formatting options.

#define ALIGNOF(type) ((UINT32)offsetof(struct { char c; type member; }, member))

Alignment macros.

• #define INITFCS 0xfffffffu

CRC/FCS constants.

· #define SIZE OF FCS 4u

for better understanding of address calculations

• #define L_ENDIAN

Define endianess if not already done by compiler.

Functions

• EXT DECL int vos hostlsBigEndian (void)

Return 1 if this is a big endian machine.

• EXT_DECL UINT32 vos_crc32 (UINT32 crc, const UINT8 *pData, UINT32 dataLen)

Calculate CRC for the given buffer and length.

• EXT_DECL UINT32 vos_sc32 (UINT32 crc, const UINT8 *pData, UINT32 dataLen)

Compute crc32 according to IEC 61375-2-3 B.7 Note: Returned CRC is inverted.

• EXT_DECL VOS_ERR_T vos_init (void *pRefCon, VOS_PRINT_DBG_T pDebugOutput)

Initialize the vos library.

EXT_DECL void vos_terminate (void)

Delnitialize the vos library.

EXT_DECL const CHAR8 * vos_getVersionString (void)

Return a human readable version representation.

EXT_DECL const VOS_VERSION_T * vos_getVersion (void)

Return version.

• EXT_DECL const CHAR8 * vos_getErrorString (VOS_ERR_T error)

Return a human readable error representation.

5.47.1 Detailed Description

Typedefs for OS abstraction.

Note

Project: TCNOpen TRDP prototype stack

Author

Bernd Loehr, NewTec GmbH

Remarks

This Source Code Form is subject to the terms of the Mozilla Public License, v. 2.0. If a copy of the MPL was not distributed with this file, You can obtain one at http://mozilla.org/MPL/2.0/. Copyright Bombardier Transportation Inc. or its subsidiaries and others, 2018. All rights reserved.

5.47.2 Macro Definition Documentation

5.47.2.1 INITFCS

#define INITFCS Oxffffffffu

CRC/FCS constants.

Initial FCS value

5.47.2.2 VOS_MAX_ERR_STR_SIZE

#define VOS_MAX_ERR_STR_SIZE (VOS_MAX_PRNT_STR_SIZE - VOS_MAX_FRMT_SIZE)

Max.

size of the error part

5.47.2.3 VOS_MAX_FRMT_SIZE

#define VOS_MAX_FRMT_SIZE 64u

Мах.

size of the 'format' part

5.47.2.4 VOS_MAX_PRNT_STR_SIZE

```
#define VOS_MAX_PRNT_STR_SIZE 256u
```

String size definitions for the debug output functions.

Max. size of the debug/error string of debug function

5.47.3 Function Documentation

5.47.3.1 vos_crc32()

Calculate CRC for the given buffer and length.

For TRDP FCS CRC calculation the CRC32 according to IEEE802.3 with start value 0xffffffff is used. Note : Returned CRC is inverted

Parameters

in	crc	Initial value.
in,out	pData	Pointer to data.
in	dataLen	length in bytes of data.

Return values

crc32	according to	
	IEEE802.3	

Calculate CRC for the given buffer and length.

/ to IEC 61375-2-3 A.3 Note: Returned CRC is inverted

Parameters

in	crc	Initial value.
in,out	pData	Pointer to data.
in	dataLen	length in bytes of data.

Return values

crc32	according to
	IEEE802.3

5.47.3.2 vos_getErrorString()

Return a human readable error representation.

Parameters

in	error	The TRDP or VOS error code
----	-------	----------------------------

Return values

```
const string pointer to error string
```

5.47.3.3 vos_getVersion()

Return version.

Return pointer to version structure

Return values



Return pointer to version structure

Return values

```
VOS_VERSION↔
T
```

5.47.3.4 vos_getVersionString()

```
\begin{tabular}{lll} EXT\_DECL & const & CHAR8* & vos\_getVersionString & ( & void & ) \end{tabular}
```

Return a human readable version representation.

Return string in the form 'v.r.u.b'

Return values

const	string
-------	--------

5.47.3.5 vos_hostIsBigEndian()

```
\begin{tabular}{ll} {\tt EXT\_DECL} & int & vos\_hostIsBigEndian & \\ & & void & ) \end{tabular}
```

Return 1 if this is a big endian machine.

Return values

0	if machine is little endian	
1	if machine is big endian	

5.47.3.6 vos_init()

Initialize the vos library.

This is used to set the output function for all VOS error and debug output.

Parameters

in	pRefCon	user context
in	pDebugOutput	pointer to debug output function

Return values

VOS_NO_ERR	no error
VOS_INIT_ERR	unsupported

Initialize the vos library.

Parameters

ı			context for debug output function
	in	pDebugOutput	Pointer to debug output function.

Return values

VOS_NO_ERR	no error VOS_INTEGRATION_ERR if endianess/alignment mismatch VOS_SOCK_ERR	
	sockets not supported VOS_UNKNOWN_ERR initialisation error	

5.47.3.7 vos_sc32()

Compute crc32 according to IEC 61375-2-3 B.7 Note: Returned CRC is inverted.

Parameters

in	crc	Initial value.
in,out	pData	Pointer to data.
in	dataLen	length in bytes of data.

Return values

crc32	according to IEC 61375-2-3
-------	----------------------------

Compute crc32 according to IEC 61375-2-3 B.7 Note: Returned CRC is inverted.

Parameters

in	crc	Initial value.
in,out	pData	Pointer to data.
in	dataLen	length in bytes of data.

Return values

```
sc32 according to IEC 61375-2-3
```

5.47.3.8 vos_terminate()

```
\begin{tabular}{ll} EXT\_DECL & void & vos\_terminate & ( \\ & void & ) \end{tabular}
```

Delnitialize the vos library.

Should be called last after TRDP stack/application does not use any VOS function anymore.

Index

callBack	GNU_PACKED, 24
GNU_PACKED, 22	
cnCnt	GNU_PACKED, 13
TRDP_ETB_INFO_T, 48	callBack, 22
cnld	comld, 22
TRDP FUNCTION INFO T, 49	confVehCnt, 22
comld	confVehList, 23
GNU PACKED, 22	cstList, 23
confVehCnt	cstUUID, 23
GNU PACKED, 22	datasetLength, 23
confVehList	defQos, 23
GNU PACKED, 23	defTtl, 24
cstld	destAddr, 24
TRDP_CONSIST_INFO_T, 42	deviceName, 24
cstInfoGetPropSize	etbld, 24
tau estinfo.c, 78	etbTopoCnt, 24
cstList	filterAddr, 24
GNU PACKED, 23	inhibit, 25
cstOwner	isLead, 25
TRDP_CONSIST_INFO_T, 42	leadDir, 25
cstUUID	leadVehOfCst, 25
GNU PACKED, 23	lifesign, 25
cstVehNo	msgType, 25
TRDP FUNCTION INFO T, 49	numCrcErr, 26
	numMissed, 26
DNS HEADER, 13	numProtErr, 26
datasetLength	numRcv, 26
GNU_PACKED, 23	numRecv, 26
defQos	numSend, 26
GNU PACKED, 23	numTopoErr, 27
defTtl	opCstList, 27
GNU PACKED, 24	opTrnDirState, 27
destAddr	opTrnTopoCnt, 27
GNU_PACKED, 24	opVehList, 27
deviceName	ownOpCstNo, 28
GNU PACKED, 24	protocolVersion, 28
and_i Adred, 24	reserved01, 28
ETB_CTRL_COMID	reserved02, 28
iec61375-2-3.h, 73	reserved03, 29
etbld	reserved04, 29
GNU PACKED, 24	reserved06, 29
TRDP_FUNCTION_INFO_T, 49	safetyTrail, 29
etbTopoCnt	serviceEntry, 29
GNU PACKED, 24	timeout, 29
CINO_I ACINED, 24	toBehav, 30
fctDev	trnCstNo, 30
service info, 37	trnDirState, 30
fctld	trnld, 30
TRDP_FUNCTION_INFO_T, 50	trnNetDir, 30
filterAddr	
IIIGIAGG	trnOperator, 30

trnTopoCnt, 31	numCrcErr
trnVehNo, 31	GNU PACKED, 26
vehld, 31	numMissed
vehOrient, 31	GNU PACKED, 26
version, 31	numProtErr
V0131011, 01	GNU_PACKED, 26
hp_slot, 32	numRcv
hp slots, 33	
11/2/51015, 33	GNU_PACKED, 26
INITFCS	numRecv
	GNU_PACKED, 26
vos_utils.h, 413	numSend
iec61375-2-3.h, 69	GNU_PACKED, 26
ETB_CTRL_COMID, 73	numTopoErr
TRDP_ETBCTRL_DSID, 73	GNU_PACKED, 27
TRDP_MAX_FILE_NAME_LEN, 74	
TRDP_MAX_LABEL_LEN, 74	opCstList
TRDP_MAX_MD_DATA_SIZE, 74	GNU_PACKED, 27
TRDP_MAX_URI_HOST_LEN, 74	opTrnDirState
TRDP_MAX_URI_LEN, 74	GNU PACKED, 27
TRDP MAX URI USER LEN, 74	opTrnTopoCnt
TRDP_MD_DEFAULT_REPLY_TIMEOUT, 75	GNU PACKED, 27
TRDP MD INFINITE TIME, 75	opVehList
TRDP MIN PD HEADER SIZE, 75	GNU PACKED, 27
TRDP MSG PD, 75	ownOpCstNo
TRDP PD UDP PORT, 75	•
TRDP_PROCESS_DEFAULT_CYCLE_TIME, 75	GNU_PACKED, 28
TRDP USR URI SIZE, 76	PD ELE, 34
TTDB NET DIR REQ COMID, 76	pFrame, 35
	•
TTDB_OP_DIR_INFO_COMID, 76	pFrame
TTDB_STAT_CST_REQ_COMID, 76	PD_ELE, 35
TTDB_TRN_DIR_REQ_COMID, 76	printSocketUsage
inhibit	trdp_utils.c, 316
GNU_PACKED, 25	trdp_utils.h, 329
isLead	protocolVersion
GNU_PACKED, 25	GNU_PACKED, 28
1 ID:	104
leadDir	reserved01
GNU_PACKED, 25	GNU_PACKED, 28
leadVehOfCst	reserved02
GNU_PACKED, 25	GNU_PACKED, 28
lifesign	reserved03
GNU_PACKED, 25	GNU_PACKED, 29
	reserved04
maxNoOfExtPublishers	GNU_PACKED, 29
TRDP_IDX_TABLE_T, 51	reserved06
maxNoOfHighCatPublishers	GNU_PACKED, 29
TRDP_IDX_TABLE_T, 52	
maxNoOfHighCatSubscriptions	SOA_SAME_SERVICEID
TRDP_IDX_TABLE_T, 52	trdp_serviceRegistry.h, 292
maxNoOfLowCatPublishers	SRM_SERVICE_READ_REQ_COMID
TRDP_IDX_TABLE_T, 52	trdp_serviceRegistry.h, 292
maxNoOfLowCatSubscriptions	SRM SRVINFO NOTIFY COMID
TRDP IDX TABLE T, 52	trdp_serviceRegistry.h, 292
maxNoOfMidCatPublishers	safetyTrail
TRDP IDX TABLE T, 52	GNU PACKED, 29
maxNoOfMidCatSubscriptions	service_info, 36
TRDP_IDX_TABLE_T, 52	fctDev, 37
msgType	serviceEntry
GNU PACKED, 25	GNU PACKED, 29
3.10_1.101(25, 20	S. 10_1 / 1011_D,

srv_info_req, 37	iec61375-2-3.h, 74 TRDP MAX PD SOCKET CNT
TAU_MARSHALL_INFO_T, 38	trdp private.h, 288
TCN URI, 38	TRDP MAX URI HOST LEN
TRDP CLTR CST INFO T, 39	iec61375-2-3.h, 74
TRDP_COM_PARAM_T, 39	TRDP_MAX_URI_LEN
TRDP_COMID_DSID_MAP_T, 40	iec61375-2-3.h, 74
TRDP_CONSIST_INFO_T, 41	TRDP_MAX_URI_USER_LEN
cstld, 42	
cstOwner, 42	iec61375-2-3.h, 74
TRDP_DATA_TYPE_T	TRDP_MD_CALLBACK_T
trdp_types.h, 311	trdp_types.h, 310
TRDP_DATASET_ELEMENT_T, 44	TRDP_MD_CONFIG_T, 54
TRDP DATASET, 43	TRDP_MD_DEFAULT_REPLY_TIMEOUT
TRDP_DBG_CONFIG_T, 45	iec61375-2-3.h, 75
TRDP_DBG_DEFAULT	TRDP_MD_ELE_ST_T
tau_xml.h, 169	trdp_private.h, 288
TRDP_DEFAULT_INDEX_SIZES	TRDP_MD_INFINITE_TIME
trdp_pdindex.h, 284	iec61375-2-3.h, 75
TRDP_DNR_OPTS	TRDP_MD_INFO_T, 55
	TRDP_MEM_CONFIG_T, 56
tau_dnr.h, 98	TRDP_MIN_PD2_HEADER_SIZE
TRDP_DNS_REPLY, 45	trdp tsn def.h, 303
tcnUriCnt, 46	TRDP MIN PD HEADER SIZE
TRDP_DNS_REQUEST, 47	iec61375-2-3.h, 75
tcnUriCnt, 47	TRDP_MSG_PD
TRDP_ERR_T	iec61375-2-3.h, 75
trdp_types.h, 312	TRDP MSG TSN PD
TRDP_ETB_INFO_T, 48	trdp_tsn_def.h, 303
cnCnt, 48	TRDP_PD_CALLBACK_T
TRDP_ETBCTRL_DSID	trdp_types.h, 310
iec61375-2-3.h, 73	. —
TRDP_EXCHG_OPTION_T	TRDP_PD_CONFIG_T, 57
tau_xml.h, 169	TRDP_PD_DEFAULT_QOS
TRDP_FLAGS_DEFAULT	trdp_tsn_def.h, 303
trdp_types.h, 309	TRDP_PD_INFO_T, 57
TRDP_FUNCTION_INFO_T, 49	TRDP_PD_UDP_PORT
cnld, 49	iec61375-2-3.h, 75
cstVehNo, 49	TRDP_PRINT_DBG_T
etbld, 49	trdp_types.h, 310
fctld, 50	TRDP_PROCESS_CONFIG_T, 58
TRDP_HANDLE, 50	TRDP_PROCESS_DEFAULT_CYCLE_TIME
TRDP_IDX_TABLE_T, 51	iec61375-2-3.h, 75
maxNoOfExtPublishers, 51	TRDP_PROP_T, 59
maxNoOfHighCatPublishers, 52	TRDP_RED_STATE_T
maxNoOfHighCatSubscriptions, 52	trdp_types.h, 313
maxNoOfLowCatPublishers, 52	TRDP_REPLY_STATUS_T
maxNoOfLowCatSubscriptions, 52	trdp_types.h, 313
maxNoOfMidCatPublishers, 52	TRDP_SDT_DEFAULT_CMTHR
maxNoOfMidCatSubscriptions, 52	tau_xml.c, 160
TRDP_IP_ADDR_T	TRDP_SDT_DEFAULT_LMIMAX
trdp_types.h, 309	tau_xml.c, 160
TRDP_MARSHALL_CONFIG_T, 53	TRDP_SDT_PAR_T, 60
TRDP_MARSHALL_T	TRDP_SEQ_CNT_ENTRY_T, 60
trdp_types.h, 309	TRDP_SESSION, 61
TRDP_MAX_FILE_NAME_LEN	TRDP_SOCK_TYPE_T
iec61375-2-3.h, 74	trdp_private.h, 288
TRDP_MAX_LABEL_LEN	TRDP_SOCKET_TCP, 62
iec61375-2-3.h, 74	TRDP SOCKETS, 63
	-
TRDP_MAX_MD_DATA_SIZE	usage, 64

TRDP_TIME_T	tau_so_if.c, 124
trdp_types.h, 310	tau_so_if.h, 128
TRDP_TO_BEHAVIOR_T	tau_dnr.c, 91
trdp_types.h, 313	tau_DNRstatus, 93
TRDP_UNMARSHALL_T	tau_addr2Uri, <mark>93</mark>
trdp_types.h, 310	tau_deInitDnr, 93
TRDP_USR_URI_SIZE	tau_getOwnAddr, 94
iec61375-2-3.h, 76	tau_initDnr, 94
TRDP_VEHICLE_INFO_T, 65	tau_uri2Addr, 95
vehld, 65	tau_dnr.h, 96
TRDP_XML_DOC_HANDLE_T, 66	TRDP_DNR_OPTS, 98
TTDB_NET_DIR_REQ_COMID	tau_DNRstatus, 100
iec61375-2-3.h, 76	tau_addr2Uri, 98
TTDB_OP_DIR_INFO_COMID	tau_deInitDnr, 99
iec61375-2-3.h, 76	tau_getOwnAddr, 100
TTDB_STAT_CST_REQ_COMID	tau_initDnr, 101
iec61375-2-3.h, 76	tau_uri2Addr, 102
TTDB_TRN_DIR_REQ_COMID	tau_dnr_types.h, 103
iec61375-2-3.h, 76	tau_freeServicesList
TTI_CACHED_CONSISTS	tau_so_if.c, 124
tau_tti.c, 133	tau_so_if.h, 129
tau_DNRstatus	tau_freeTelegrams
tau_dnr.c, 93	tau_xml.c, 160
tau_dnr.h, 100	tau_xml.h, 169
tau_addService	tau_freeXmlDatasetConfig
tau_so_if.c, 123	tau_xml.c, 160
tau_so_if.h, 128	tau_xml.h, 170
tau_addr2Uri	tau_freeXmlDoc
tau_dnr.c, 93	tau_xml.c, 161
tau_dnr.h, 98	tau_xml.h, 170
tau_calcDatasetSize	tau_getCstFctCnt
tau_marshall.c, 106	tau_tti.c, 133
tau_marshall.h, 113	tau_tti.h, 145
tau_calcDatasetSizeByComId tau_marshall.c, 107	tau_getCstFctInfo
	tau_tti.c, 134
tau_marshall.h, 114	tau_tti.h, 145 tau_getCstInfo
tau_cstinfo.c, 77 cstInfoGetPropSize, 78	tau_tti.c, 134
tau_ctrl.c, 79	tau_tti.h, 146
tau_getEcspStat, 81	tau_tti.ii, 140 tau_getCstVehCnt
tau_initEcspCtrl, 81	tau tti.c, 135
tau_requestEcspConfirm, 82	tau_tti.h, 146
tau setEcspCtrl, 82	tau_getEcspStat
tau_terminateEcspCtrl, 83	tau ctrl.c, 81
tau ctrl.h, 83	tau_ctrl.h, 86
tau_getEcspStat, 86	tau_getOpTrDirectory
tau initEcspCtrl, 86	tau_tti.c, 135
tau_requestEcspConfirm, 87	tau_tti.h, 147
tau_setEcspCtrl, 87	tau_getOpTrnDirectoryStatusInfo
tau_terminateEcspCtrl, 88	tau tti.c, 135
tau_ctrl_types.h, 88	tau_tti.h, 148
tau_deInitDnr	tau_getOwnAddr
tau_dnr.c, 93	tau_dnr.c, 94
tau_dnr.h, 99	tau_dnr.h, 100
tau_deInitTTI	tau_getOwnIds
tau_tti.c, 133	tau_tti.c, 136
tau_tti.h, 144	tau_tti.h, 148
tau_delService	tau_getOwnOpCstNo
	_9

tau_tti.c, 136	tau_unmarshall, 118
tau_tti.h, 149	tau_unmarshallDs, 120
tau_getOwnTrnCstNo	tau_marshallDs
tau_tti.c, 137	tau_marshall.c, 109
tau_tti.h, 149	tau_marshall.h, 117
tau_getServicesList	tau_prepareXmlDoc
tau_so_if.c, 124	tau_xml.c, 161
tau_so_if.h, 129	tau_xml.h, 171
tau_getStaticCstInfo	tau_prepareXmlMem
tau_tti.c, 137	tau_xml.c, 162
tau_tti.h, 149	tau_xml.h, 171
tau_getTTI	tau_readXmlDatasetConfig
tau_tti.c, 139	tau_xml.c, 162
tau_tti.h, 152	tau xml.h, 172
tau_getTrDirectory	tau_readXmlDeviceConfig
tau_tti.c, 138	tau_xml.c, 162
tau tti.h, 150	tau xml.h, 172
tau_getTrnCstCnt	tau_readXmlInterfaceConfig
tau_tti.c, 138	tau xml.c, 163
tau_tti.h, 151	tau xml.h, 173
tau_getTrnVehCnt	tau_readXmIMappedDeviceConfig
tau_tti.c, 138	tau_xml.c, 164
tau_tti.h, 151	tau xml.h, 174
tau_getVehInfo	tau_readXmlMappedDevices
tau_tti.c, 139	tau_readXminiappedDevices tau_xml.c, 164
tau_tti.h, 152	tau_xml.h, 174
tau_getVehOrient	tau_readXmlMappedInterfaceConfig
tau_getvenonent tau tti.c, 140	tau_xml.c, 165
tau_tti.h, 153	tau_xml.h, 176
tau_initDnr	tau_readXmlServiceConfig
tau_iiiibiii tau dnr.c, 94	
-	tau_xml.c, 165
tau_dnr.h, 101 tau initEcspCtrl	tau_xml.h, 176
- ·	tau_requestEcspConfirm
tau_ctrl.c, 81	tau_ctrl.c, 82
tau_ctrl.h, 86	tau_ctrl.h, 87
tau_initMarshall	tau_setEcspCtrl
tau_marshall.c, 108	tau_ctrl.c, 82
tau_marshall.h, 115	tau_ctrl.h, 87
tau_initTTlaccess	tau_so_if.c, 121
tau_tti.c, 140	tau_addService, 123
tau_tti.h, 154	tau_delService, 124
tau_marshall	tau_freeServicesList, 124
tau_marshall.c, 108	tau_getServicesList, 124
tau_marshall.h, 116	tau_updService, 125
tau_marshall.c, 105	tau_so_if.h, 126
tau_calcDatasetSize, 106	tau_addService, 128
tau_calcDatasetSizeByComld, 107	tau_delService, 128
tau_initMarshall, 108	tau_freeServicesList, 129
tau_marshall, 108	tau_getServicesList, 129
tau_marshallDs, 109	tau_updService, 130
tau_unmarshall, 110	tau_terminateEcspCtrl
tau_unmarshallDs, 110	tau_ctrl.c, 83
tau_marshall.h, 111	tau_ctrl.h, 88
tau_calcDatasetSize, 113	tau_tti.c, 130
tau_calcDatasetSizeByComld, 114	TTI_CACHED_CONSISTS, 133
tau_initMarshall, 115	tau_delnitTTI, 133
tau_marshall, 116	tau_getCstFctCnt, 133
tau_marshallDs, 117	tau_getCstFctInfo, 134

	tau_getCstInfo, 134	tau_readXmlMappedDeviceConfig, 164
	tau_getCstVehCnt, 135	tau_readXmlMappedDevices, 164
	tau_getOpTrDirectory, 135	tau_readXmlMappedInterfaceConfig, 165
	tau_getOpTrnDirectoryStatusInfo, 135	tau_readXmlServiceConfig, 165
	tau_getOwnlds, 136	tau_xml.h, 166
	tau_getOwnOpCstNo, 136	TRDP_DBG_DEFAULT, 169
	tau_getOwnTrnCstNo, 137	TRDP_EXCHG_OPTION_T, 169
	tau_getStaticCstInfo, 137	tau_freeTelegrams, 169
	tau_getTTI, 139	tau_freeXmlDatasetConfig, 170
	tau_getTrDirectory, 138	tau_freeXmlDoc, 170
	tau_getTrnCstCnt, 138	tau_prepareXmlDoc, 171
	tau_getTrnVehCnt, 138	tau_prepareXmlMem, 171
	tau_getVehInfo, 139	tau_readXmlDatasetConfig, 172
	tau_getVehOrient, 140	tau_readXmlDeviceConfig, 172
+ • • •	tau_initTTlaccess, 140	tau_readXmlInterfaceConfig, 173
ıau_	tti.h, 141	tau_readXmlMappedDeviceConfig, 174
	tau_deInitTTI, 144	tau_readXmlMappedDevices, 174
	tau_getCstFctCnt, 145 tau_getCstFctInfo, 145	tau_readXmlMappedInterfaceConfig, 176 tau_readXmlServiceConfig, 176
	tau getOstInfo, 146	tenUriCnt
	tau_getCstVehCnt, 146	TRDP DNS REPLY, 46
	tau_getOpTrDirectory, 147	TRDP DNS REQUEST, 47
	tau_getOpTrnDirectoryStatusInfo, 148	timeout
	tau getOwnlds, 148	GNU PACKED, 29
	tau_getOwnOpCstNo, 149	tlc closeSession
	tau_getOwnTrnCstNo, 149	tlc if.c, 181
	tau_getStaticCstInfo, 149	trdp_if_light.h, 219
	tau_getTTI, 152	tlc_configSession
	tau_getTrDirectory, 150	tlc_if.c, 181
	tau_getTrnCstCnt, 151	trdp_if_light.h, 219
	tau getTrnVehCnt, 151	tlc_getETBTopoCount
	tau_getVehInfo, 152	tlc if.c, 182
	tau_getVehOrient, 153	trdp_if_light.h, 220
	tau_initTTlaccess, 154	tlc_getInterval
tau_	tti_types.h, 155	tlc_if.c, 182
tau_	unmarshall	trdp_if_light.h, 220
	tau_marshall.c, 110	tlc_getJoinStatistics
	tau_marshall.h, 118	trdp_if_light.h, 221
tau_	unmarshallDs	trdp_stats.c, 294
	tau_marshall.c, 110	tlc_getOpTrainTopoCount
	tau_marshall.h, 120	tlc_if.c, 183
tau_	updService	trdp_if_light.h, 221
	tau_so_if.c, 125	tlc_getOwnIpAddress
	tau_so_if.h, 130	tlc_if.c, 183
tau_	uri2Addr	trdp_if_light.h, 222
	tau_dnr.c, 95	tlc_getPubStatistics
	tau_dnr.h, 102	trdp_if_light.h, 222
tau_	xml.c, 158	trdp_stats.c, 295
	TRDP_SDT_DEFAULT_CMTHR, 160	tlc_getRedStatistics
	TRDP_SDT_DEFAULT_LMIMAX, 160	trdp_if_light.h, 223
	tau_freeTelegrams, 160	trdp_stats.c, 295
	tau_freeXmlDatasetConfig, 160	tlc_getStatistics
	tau_freeXmlDoc, 161	trdp_if_light.h, 223
	tau_prepareXmlDoc, 161	trdp_stats.c, 296
	tau_prepareXmlMem, 162	tlc_getSubsStatistics
	tau_readXmlDatasetConfig, 162	trdp_if_light.h, 224
	tau_readXmlDeviceConfig, 162	trdp_stats.c, 296
	tau_readXmlInterfaceConfig, 163	tlc_getVersion

tlc_if.c, 183	tlc_if.c, 188
trdp_if_light.h, 224	trdp_if_light.h, 229
tlc_getVersionString	tlm_abortSession
tlc_if.c, 184	tlm_if.c, 195
trdp_if_light.h, 224	trdp_if_light.h, 230
tlc_if.c, 178	tlm_addListener
tlc_closeSession, 181	tlm_if.c, 195
tlc_configSession, 181	trdp_if_light.h, 230
tlc_getETBTopoCount, 182	tlm_confirm
tlc_getInterval, 182	tlm_if.c, 196
tlc_getOpTrainTopoCount, 183	trdp_if_light.h, 231
tlc_getOwnlpAddress, 183	tlm_delListener
tlc_getVersion, 183	tlm_if.c, 197
tlc_getVersionString, 184	trdp_if_light.h, 232
tlc_init, 184	tlm_getInterval
tlc_openSession, 184	tlm_if.c, 197
tlc_presetIndexSession, 185	trdp_if_light.h, 232
tlc_process, 186	tlm_if.c, 193
tlc_reinitSession, 186	tlm_abortSession, 195
tlc_setETBTopoCount, 187	tlm_addListener, 195
tlc_setOpTrainTopoCount, 187	tlm_confirm, 196
tlc_terminate, 188	tlm_delListener, 197
tlc_updateSession, 188	tlm_getInterval, 197
trdp_getAccess, 188	tlm_notify, 198
trdp_isValidSession, 189	tlm_process, 199
trdp_releaseAccess, 189	tlm_readdListener, 199
trdp_sessionQueue, 190	tlm_reply, 200
tlc_if.h, 190	tlm_replyQuery, 201
	tlm request 001
trdp_isValidSession, 192	tlm_request, 201
trdp_isValidSession, 192 trdp_sessionQueue, 192	tlm_notify
• —	_ ·
trdp_sessionQueue, 192	tlm_notify
trdp_sessionQueue, 192 tlc_init	tlm_notify tlm_if.c, 198
trdp_sessionQueue, 192 tlc_init tlc_if.c, 184	tlm_notify tlm_if.c, 198 trdp_if_light.h, 233
trdp_sessionQueue, 192 tlc_init tlc_if.c, 184 trdp_if_light.h, 225	tlm_notify tlm_if.c, 198 trdp_if_light.h, 233 tlm_process
trdp_sessionQueue, 192 tlc_init tlc_if.c, 184 trdp_if_light.h, 225 tlc_openSession	tlm_notify tlm_if.c, 198 trdp_if_light.h, 233 tlm_process tlm_if.c, 199
trdp_sessionQueue, 192 tlc_init tlc_if.c, 184 trdp_if_light.h, 225 tlc_openSession tlc_if.c, 184	tlm_notify tlm_if.c, 198 trdp_if_light.h, 233 tlm_process tlm_if.c, 199 trdp_if_light.h, 234
trdp_sessionQueue, 192 tlc_init tlc_if.c, 184 trdp_if_light.h, 225 tlc_openSession tlc_if.c, 184 trdp_if_light.h, 225	tlm_notify tlm_if.c, 198 trdp_if_light.h, 233 tlm_process tlm_if.c, 199 trdp_if_light.h, 234 tlm_readdListener tlm_if.c, 199
trdp_sessionQueue, 192 tlc_init tlc_if.c, 184 trdp_if_light.h, 225 tlc_openSession tlc_if.c, 184 trdp_if_light.h, 225 tlc_presetIndexSession	tlm_notify tlm_if.c, 198 trdp_if_light.h, 233 tlm_process tlm_if.c, 199 trdp_if_light.h, 234 tlm_readdListener
trdp_sessionQueue, 192 tlc_init tlc_if.c, 184 trdp_if_light.h, 225 tlc_openSession tlc_if.c, 184 trdp_if_light.h, 225 tlc_presetIndexSession tlc_if.c, 185	tlm_notify tlm_if.c, 198 trdp_if_light.h, 233 tlm_process tlm_if.c, 199 trdp_if_light.h, 234 tlm_readdListener tlm_if.c, 199 trdp_if_light.h, 234
trdp_sessionQueue, 192 tlc_init tlc_if.c, 184 trdp_if_light.h, 225 tlc_openSession tlc_if.c, 184 trdp_if_light.h, 225 tlc_presetIndexSession tlc_if.c, 185 trdp_if_light.h, 226	tlm_notify tlm_if.c, 198 trdp_if_light.h, 233 tlm_process tlm_if.c, 199 trdp_if_light.h, 234 tlm_readdListener tlm_if.c, 199 trdp_if_light.h, 234 tlm_reply
trdp_sessionQueue, 192 tlc_init tlc_if.c, 184 trdp_if_light.h, 225 tlc_openSession tlc_if.c, 184 trdp_if_light.h, 225 tlc_presetIndexSession tlc_if.c, 185 trdp_if_light.h, 226 tlc_process	tlm_notify tlm_if.c, 198 trdp_if_light.h, 233 tlm_process tlm_if.c, 199 trdp_if_light.h, 234 tlm_readdListener tlm_if.c, 199 trdp_if_light.h, 234 tlm_reply tlm_if.c, 200
trdp_sessionQueue, 192 tlc_init tlc_if.c, 184 trdp_if_light.h, 225 tlc_openSession tlc_if.c, 184 trdp_if_light.h, 225 tlc_presetIndexSession tlc_if.c, 185 trdp_if_light.h, 226 tlc_process tlc_if.c, 186	tlm_notify tlm_if.c, 198 trdp_if_light.h, 233 tlm_process tlm_if.c, 199 trdp_if_light.h, 234 tlm_readdListener tlm_if.c, 199 trdp_if_light.h, 234 tlm_reply tlm_if.c, 200 trdp_if_light.h, 235
trdp_sessionQueue, 192 tlc_init tlc_if.c, 184 trdp_if_light.h, 225 tlc_openSession tlc_if.c, 184 trdp_if_light.h, 225 tlc_presetIndexSession tlc_if.c, 185 trdp_if_light.h, 226 tlc_process tlc_if.c, 186 trdp_if_light.h, 227	tlm_notify tlm_if.c, 198 trdp_if_light.h, 233 tlm_process tlm_if.c, 199 trdp_if_light.h, 234 tlm_readdListener tlm_if.c, 199 trdp_if_light.h, 234 tlm_reply tlm_if.c, 200 trdp_if_light.h, 235 tlm_replyQuery
trdp_sessionQueue, 192 tlc_init tlc_if.c, 184 trdp_if_light.h, 225 tlc_openSession tlc_if.c, 184 trdp_if_light.h, 225 tlc_presetIndexSession tlc_if.c, 185 trdp_if_light.h, 226 tlc_process tlc_if.c, 186 trdp_if_light.h, 227 tlc_reinitSession	tlm_notify tlm_if.c, 198 trdp_if_light.h, 233 tlm_process tlm_if.c, 199 trdp_if_light.h, 234 tlm_readdListener tlm_if.c, 199 trdp_if_light.h, 234 tlm_reply tlm_if.c, 200 trdp_if_light.h, 235 tlm_replyQuery tlm_if.c, 201
trdp_sessionQueue, 192 tlc_init tlc_if.c, 184 trdp_if_light.h, 225 tlc_openSession tlc_if.c, 184 trdp_if_light.h, 225 tlc_presetIndexSession tlc_if.c, 185 trdp_if_light.h, 226 tlc_process tlc_if.c, 186 trdp_if_light.h, 227 tlc_reinitSession tlc_if.c, 186	tlm_notify tlm_if.c, 198 trdp_if_light.h, 233 tlm_process tlm_if.c, 199 trdp_if_light.h, 234 tlm_readdListener tlm_if.c, 199 trdp_if_light.h, 234 tlm_reply tlm_if.c, 200 trdp_if_light.h, 235 tlm_replyQuery tlm_if.c, 201 trdp_if_light.h, 236
trdp_sessionQueue, 192 tlc_init tlc_if.c, 184 trdp_if_light.h, 225 tlc_openSession tlc_if.c, 184 trdp_if_light.h, 225 tlc_presetIndexSession tlc_if.c, 185 trdp_if_light.h, 226 tlc_process tlc_if.c, 186 trdp_if_light.h, 227 tlc_reinitSession tlc_if.c, 186 trdp_if_light.h, 227 tlc_resetStatistics	tlm_notify tlm_if.c, 198 trdp_if_light.h, 233 tlm_process tlm_if.c, 199 trdp_if_light.h, 234 tlm_readdListener tlm_if.c, 199 trdp_if_light.h, 234 tlm_reply tlm_if.c, 200 trdp_if_light.h, 235 tlm_replyQuery tlm_if.c, 201 trdp_if_light.h, 236 tlm_request tlm_if.c, 201
trdp_sessionQueue, 192 tlc_init tlc_if.c, 184 trdp_if_light.h, 225 tlc_openSession tlc_if.c, 184 trdp_if_light.h, 225 tlc_presetIndexSession tlc_if.c, 185 trdp_if_light.h, 226 tlc_process tlc_if.c, 186 trdp_if_light.h, 227 tlc_reinitSession tlc_if.c, 186 trdp_if_light.h, 227 tlc_resetStatistics trdp_if_light.h, 228	tlm_notify tlm_if.c, 198 trdp_if_light.h, 233 tlm_process tlm_if.c, 199 trdp_if_light.h, 234 tlm_readdListener tlm_if.c, 199 trdp_if_light.h, 234 tlm_reply tlm_if.c, 200 trdp_if_light.h, 235 tlm_replyQuery tlm_if.c, 201 trdp_if_light.h, 236 tlm_request tlm_if.c, 201 trdp_if_light.h, 236
trdp_sessionQueue, 192 tlc_init tlc_if.c, 184 trdp_if_light.h, 225 tlc_openSession tlc_if.c, 184 trdp_if_light.h, 225 tlc_presetIndexSession tlc_if.c, 185 trdp_if_light.h, 226 tlc_process tlc_if.c, 186 trdp_if_light.h, 227 tlc_reinitSession tlc_if.c, 186 trdp_if_light.h, 227 tlc_resetStatistics trdp_if_light.h, 227	tlm_notify tlm_if.c, 198 trdp_if_light.h, 233 tlm_process tlm_if.c, 199 trdp_if_light.h, 234 tlm_readdListener tlm_if.c, 199 trdp_if_light.h, 234 tlm_reply tlm_if.c, 200 trdp_if_light.h, 235 tlm_replyQuery tlm_if.c, 201 trdp_if_light.h, 236 tlm_request tlm_if.c, 201 trdp_if_light.h, 236 tlp_get
trdp_sessionQueue, 192 tlc_init tlc_if.c, 184 trdp_if_light.h, 225 tlc_openSession tlc_if.c, 184 trdp_if_light.h, 225 tlc_presetIndexSession tlc_if.c, 185 trdp_if_light.h, 226 tlc_process tlc_if.c, 186 trdp_if_light.h, 227 tlc_reinitSession tlc_if.c, 186 trdp_if_light.h, 227 tlc_resetStatistics trdp_if_light.h, 227 tlc_resetStatistics trdp_stats.c, 297 tlc_setETBTopoCount	tlm_notify tlm_if.c, 198 trdp_if_light.h, 233 tlm_process tlm_if.c, 199 trdp_if_light.h, 234 tlm_readdListener tlm_if.c, 199 trdp_if_light.h, 234 tlm_reply tlm_if.c, 200 trdp_if_light.h, 235 tlm_replyQuery tlm_if.c, 201 trdp_if_light.h, 236 tlm_request tlm_if.c, 201 trdp_if_light.h, 236 tlm_request tlm_if.c, 201 trdp_if_light.h, 236 tlp_get tlp_if.c, 205
trdp_sessionQueue, 192 tlc_init tlc_if.c, 184 trdp_if_light.h, 225 tlc_openSession tlc_if.c, 184 trdp_if_light.h, 225 tlc_presetIndexSession tlc_if.c, 185 trdp_if_light.h, 226 tlc_process tlc_if.c, 186 trdp_if_light.h, 227 tlc_reinitSession tlc_if.c, 186 trdp_if_light.h, 227 tlc_resetStatistics trdp_if_light.h, 227 tlc_resetStatistics trdp_if_light.h, 228 trdp_stats.c, 297 tlc_setETBTopoCount tlc_if.c, 187	tlm_notify tlm_if.c, 198 trdp_if_light.h, 233 tlm_process tlm_if.c, 199 trdp_if_light.h, 234 tlm_readdListener tlm_if.c, 199 trdp_if_light.h, 234 tlm_reply tlm_if.c, 200 trdp_if_light.h, 235 tlm_replyQuery tlm_if.c, 201 trdp_if_light.h, 236 tlm_request tlm_if.c, 201 trdp_if_light.h, 236 tlp_get tlp_if.c, 205 trdp_if_light.h, 237
trdp_sessionQueue, 192 tlc_init tlc_if.c, 184 trdp_if_light.h, 225 tlc_openSession tlc_if.c, 184 trdp_if_light.h, 225 tlc_presetIndexSession tlc_if.c, 185 trdp_if_light.h, 226 tlc_process tlc_if.c, 186 trdp_if_light.h, 227 tlc_reinitSession tlc_if.c, 186 trdp_if_light.h, 227 tlc_resetStatistics trdp_if_light.h, 227 tlc_resetStatistics trdp_if_light.h, 228 trdp_stats.c, 297 tlc_setETBTopoCount tlc_if.c, 187 trdp_if_light.h, 228	tlm_notify tlm_if.c, 198 trdp_if_light.h, 233 tlm_process tlm_if.c, 199 trdp_if_light.h, 234 tlm_readdListener tlm_if.c, 199 trdp_if_light.h, 234 tlm_reply tlm_if.c, 200 trdp_if_light.h, 235 tlm_replyQuery tlm_if.c, 201 trdp_if_light.h, 236 tlm_request tlm_if.c, 201 trdp_if_light.h, 236 tlp_get tlp_if.c, 205 trdp_if_light.h, 237 tlp_getInterval
trdp_sessionQueue, 192 tlc_init tlc_if.c, 184 trdp_if_light.h, 225 tlc_openSession tlc_if.c, 184 trdp_if_light.h, 225 tlc_presetIndexSession tlc_if.c, 185 trdp_if_light.h, 226 tlc_process tlc_if.c, 186 trdp_if_light.h, 227 tlc_reinitSession tlc_if.c, 186 trdp_if_light.h, 227 tlc_resetStatistics trdp_if_light.h, 227 tlc_resetStatistics trdp_if_light.h, 228 trdp_stats.c, 297 tlc_setETBTopoCount tlc_if.c, 187 trdp_if_light.h, 228 tlc_setOpTrainTopoCount	tlm_notify tlm_if.c, 198 trdp_if_light.h, 233 tlm_process tlm_if.c, 199 trdp_if_light.h, 234 tlm_readdListener tlm_if.c, 199 trdp_if_light.h, 234 tlm_reply tlm_if.c, 200 trdp_if_light.h, 235 tlm_replyQuery tlm_if.c, 201 trdp_if_light.h, 236 tlm_request tlm_if.c, 201 trdp_if_light.h, 236 tlp_get tlp_if.c, 205 trdp_if_light.h, 237 tlp_getInterval tlp_if.c, 205
trdp_sessionQueue, 192 tlc_init tlc_if.c, 184 trdp_if_light.h, 225 tlc_openSession tlc_if.c, 184 trdp_if_light.h, 225 tlc_presetIndexSession tlc_if.c, 185 trdp_if_light.h, 226 tlc_process tlc_if.c, 186 trdp_if_light.h, 227 tlc_reinitSession tlc_if.c, 186 trdp_if_light.h, 227 tlc_resetStatistics trdp_if_light.h, 228 trdp_stats.c, 297 tlc_setETBTopoCount tlc_if.c, 187 trdp_if_light.h, 228 tlc_setOpTrainTopoCount tlc_if.c, 187	tlm_notify tlm_if.c, 198 trdp_if_light.h, 233 tlm_process tlm_if.c, 199 trdp_if_light.h, 234 tlm_readdListener tlm_if.c, 199 trdp_if_light.h, 234 tlm_reply tlm_if.c, 200 trdp_if_light.h, 235 tlm_replyQuery tlm_if.c, 201 trdp_if_light.h, 236 tlm_request tlm_if.c, 201 trdp_if_light.h, 236 tlp_get tlp_if.c, 205 trdp_if_light.h, 237 tlp_getInterval tlp_if.c, 205 trdp_if_light.h, 238
trdp_sessionQueue, 192 tlc_init tlc_if.c, 184 trdp_if_light.h, 225 tlc_openSession tlc_if.c, 184 trdp_if_light.h, 225 tlc_presetIndexSession tlc_if.c, 185 trdp_if_light.h, 226 tlc_process tlc_if.c, 186 trdp_if_light.h, 227 tlc_reinitSession tlc_if.c, 186 trdp_if_light.h, 227 tlc_resetStatistics trdp_if_light.h, 228 trdp_stats.c, 297 tlc_setETBTopoCount tlc_if.c, 187 trdp_if_light.h, 228 tlc_setOpTrainTopoCount tlc_if.c, 187 trdp_if_light.h, 228	tlm_notify tlm_if.c, 198 trdp_if_light.h, 233 tlm_process tlm_if.c, 199 trdp_if_light.h, 234 tlm_readdListener tlm_if.c, 199 trdp_if_light.h, 234 tlm_reply tlm_if.c, 200 trdp_if_light.h, 235 tlm_replyQuery tlm_if.c, 201 trdp_if_light.h, 236 tlm_request tlm_if.c, 201 trdp_if_light.h, 236 tlm_request tlm_if.c, 201 trdp_if_light.h, 236 tlp_get tlp_if.c, 205 trdp_if_light.h, 237 tlp_getInterval tlp_if.c, 205 trdp_if_light.h, 238 tlp_getRedundant
trdp_sessionQueue, 192 tlc_init tlc_if.c, 184 trdp_if_light.h, 225 tlc_openSession tlc_if.c, 184 trdp_if_light.h, 225 tlc_presetIndexSession tlc_if.c, 185 trdp_if_light.h, 226 tlc_process tlc_if.c, 186 trdp_if_light.h, 227 tlc_reinitSession tlc_if.c, 186 trdp_if_light.h, 227 tlc_resetStatistics trdp_if_light.h, 228 trdp_stats.c, 297 tlc_setETBTopoCount tlc_if.c, 187 trdp_if_light.h, 228 tlc_setOpTrainTopoCount tlc_if.c, 187 trdp_if_light.h, 228 tlc_setOpTrainTopoCount tlc_if.c, 187 trdp_if_light.h, 228 tlc_terminate	tlm_notify tlm_if.c, 198 trdp_if_light.h, 233 tlm_process tlm_if.c, 199 trdp_if_light.h, 234 tlm_readdListener tlm_if.c, 199 trdp_if_light.h, 234 tlm_reply tlm_if.c, 200 trdp_if_light.h, 235 tlm_replyQuery tlm_if.c, 201 trdp_if_light.h, 236 tlm_request tlm_if.c, 201 trdp_if_light.h, 236 tlp_get tlp_if.c, 205 trdp_if_light.h, 237 tlp_getInterval tlp_if.c, 205 trdp_if_light.h, 238 tlp_getRedundant tlp_if.c, 206
trdp_sessionQueue, 192 tlc_init tlc_if.c, 184 trdp_if_light.h, 225 tlc_openSession tlc_if.c, 184 trdp_if_light.h, 225 tlc_presetIndexSession tlc_if.c, 185 trdp_if_light.h, 226 tlc_process tlc_if.c, 186 trdp_if_light.h, 227 tlc_reinitSession tlc_if.c, 186 trdp_if_light.h, 227 tlc_resetStatistics trdp_if_light.h, 228 trdp_stats.c, 297 tlc_setETBTopoCount tlc_if.c, 187 trdp_if_light.h, 228 tlc_setOpTrainTopoCount tlc_if.c, 187 trdp_if_light.h, 228 tlc_terminate tlc_if.c, 188	tlm_notify tlm_if.c, 198 trdp_if_light.h, 233 tlm_process tlm_if.c, 199 trdp_if_light.h, 234 tlm_readdListener tlm_if.c, 199 trdp_if_light.h, 234 tlm_reply tlm_if.c, 200 trdp_if_light.h, 235 tlm_replyQuery tlm_if.c, 201 trdp_if_light.h, 236 tlm_request tlm_if.c, 201 trdp_if_light.h, 236 tlp_get tlp_if.c, 205 trdp_if_light.h, 237 tlp_getInterval tlp_if.c, 205 trdp_if_light.h, 238 tlp_getRedundant tlp_if.c, 206 trdp_if_light.h, 238
trdp_sessionQueue, 192 tlc_init tlc_if.c, 184 trdp_if_light.h, 225 tlc_openSession tlc_if.c, 184 trdp_if_light.h, 225 tlc_presetIndexSession tlc_if.c, 185 trdp_if_light.h, 226 tlc_process tlc_if.c, 186 trdp_if_light.h, 227 tlc_reinitSession tlc_if.c, 186 trdp_if_light.h, 227 tlc_resetStatistics trdp_if_light.h, 228 trdp_stats.c, 297 tlc_setETBTopoCount tlc_if.c, 187 trdp_if_light.h, 228 tlc_setOpTrainTopoCount tlc_if.c, 187 trdp_if_light.h, 228 tlc_setOpTrainTopoCount tlc_if.c, 187 trdp_if_light.h, 228 tlc_terminate	tlm_notify tlm_if.c, 198 trdp_if_light.h, 233 tlm_process tlm_if.c, 199 trdp_if_light.h, 234 tlm_readdListener tlm_if.c, 199 trdp_if_light.h, 234 tlm_reply tlm_if.c, 200 trdp_if_light.h, 235 tlm_replyQuery tlm_if.c, 201 trdp_if_light.h, 236 tlm_request tlm_if.c, 201 trdp_if_light.h, 236 tlp_get tlp_if.c, 205 trdp_if_light.h, 237 tlp_getInterval tlp_if.c, 205 trdp_if_light.h, 238 tlp_getRedundant tlp_if.c, 206

tlp_getInterval, 205	trdp_SockIsJoined
tlp_getRedundant, 206	trdp_utils.c, 325
tlp_processReceive, 206	trdp_utils.h, 338
tlp_processSend, 207	trdp_UpdateStats
tlp_publish, 207	trdp_stats.c, 298
tlp_put, 208	trdp_XMLClose
tlp_putImmediate, 209	trdp_xml.c, 341
tlp_republish, 209	trdp xml.h, 347
tlp_request, 210	trdp_XMLCountStartTag
tlp_resubscribe, 211	trdp_xml.c, 341
tlp_setRedundant, 212	trdp xml.h, 348
tlp_subscribe, 212	trdp_XMLEnter
tlp_unpublish, 213	trdp_xml.c, 342
tlp unsubscribe, 214	trdp xml.h, 348
tlp_processReceive	trdp XMLGetAttribute
tlp_if.c, 206	trdp_xml.c, 342
trdp_if_light.h, 239	trdp_xml.h, 348
tlp_processSend	trdp XMLLeave
tlp_if.c, 207	trdp xml.c, 343
trdp_if_light.h, 239	trdp_xml.h, 349
tlp_publish	trdp_XMLMemOpen
tlp_if.c, 207	trdp_xml.c, 343
•	trdp_xml.h, 349
trdp_if_light.h, 240	•
tlp_put	trdp_XMLOpen
tlp_if.c, 208	trdp_xml.c, 343
trdp_if_light.h, 241	trdp_xml.h, 350
tlp_putImmediate	trdp_XMLRewind
tlp_if.c, 209	trdp_xml.c, 344
trdp_if_light.h, 242	trdp_xml.h, 350
tlp_republish	trdp_XMLSeekStartTag
tlp_if.c, 209	trdp_xml.c, 344
trdp_if_light.h, 242	trdp_xml.h, 350
tlp_request	trdp_XMLSeekStartTagAny
tlp_if.c, 210	trdp_xml.c, 345
trdp_if_light.h, 243	trdp_xml.h, 351
tlp_resubscribe	trdp_checkSequenceCounter
tlp_if.c, 211	trdp_utils.c, 316
trdp_if_light.h, 244	trdp_utils.h, 329
tlp_setRedundant	trdp_dllmain.c, 214
tlp_if.c, 212	trdp_findMCjoins
trdp_if_light.h, 245	trdp_utils.c, 317
tlp_subscribe	trdp_utils.h, 330
tlp_if.c, 212	trdp_findSubAddr
trdp_if_light.h, 245	trdp_utils.c, 317
tlp_unpublish	trdp_utils.h, 330
tlp_if.c, 213	trdp_getAccess
trdp_if_light.h, 246	tlc_if.c, 188
tlp_unsubscribe	trdp_getSeqCnt
tlp_if.c, 214	trdp_utils.c, 318
trdp_if_light.h, 246	trdp_utils.h, 331
toBehav	trdp_handleTimeout
GNU_PACKED, 30	trdp_pdcom.c, 262
trdp_SockAddJoin	trdp_pdcom.h, 273
trdp_utils.c, 325	trdp_if_light.h, 215
trdp_utils.h, 338	tlc_closeSession, 219
trdp_SockDelJoin	tlc_configSession, 219
trdp_utils.c, 325	tlc_getETBTopoCount, 220
trdp_utils.h, 338	tlc_getInterval, 220
11 dp_u1115.111, 000	tio_gotimer val, 220

tic_gelCo/TrainTopoCount, 221 tic_gelCo/TrainTopoCount, 221 tic_gelCo/TrainTopoCount, 221 tic_gelCo/TrainTopoCount, 221 tic_gelCo/TrainTopoCount, 222 tic_gelStatistics, 223 tic_gelStatistics, 223 tic_gelStatistics, 223 tic_gelStatistics, 223 tic_gelStatistics, 223 tic_gelStatistics, 224 tic_gelVersion, 224 tic_gelVersion, 224 tic_init, 225 tic_openSession, 225 tic_openSession, 225 tic_preselIndexSession, 226 tic_preselIndexSession, 227 tic_reinitSession, 227 tic_reinitSession, 227 tic_reinitSession, 227 tic_reinitSession, 227 tic_reinitSession, 227 tic_preselStatistics, 228 tic_selCTFaTopoCount, 228 tic_pmdComCount, 228 tic_pmdCount,		W . Y . 400
tic_getPubStatistics, 222 tic_getRedStatistics, 223 tic_getSubStatistics, 223 tic_getSubStatistics, 223 tic_getSubStatistics, 223 tic_getVersion, 224 tic_getVersion, 224 tic_getVersion, 224 tic_getVersion, 224 tic_getVersion, 225 tic_penSession, 225 tic_penSession, 225 tic_presetIndexSession, 226 tic_presetIndexSession, 227 tic_reintiSession, 227 tic_reintiSession, 227 tic_reintiSession, 227 tic_reintiSession, 227 tic_reintiSession, 228 tic_setCpTrainTopoCount, 228 tic_setOpTrainTopoCount, 228 tic_setOpTrainTopoCount, 228 tic_toupdateSession, 229 tim_abotIsession, 230 tim_addListener, 230 tim_dedListener, 230 tim_dedListener, 231 tim_dedListener, 232 tim_molify, 233 tim_process, 234 tim_request, 236 tim_reptQuery, 236 tim_reptQuery, 236 tim_reptQuery, 236 tim_reptQuery, 238 tip_processReceive, 239 tip_publish, 240 tip_put, 241 tip_putImerdiate, 242 tip_request, 243 tip_resubscribe, 244 tip_setRedundant, 245 tip_publscribe, 244 tip_setRedundant, 245 tip_publscribe, 244 tip_setRedundant, 245 tip_pubscribe, 244 tip_setRedundant, 245 tip_publscribe, 246 trdp_initSockets trdp_utils.n, 331 trdp_istAts.c, 319 trdp_tutils.n, 331 trdp_istAlp, 332 trdp_putils.n, 333 trdp_putils.n, 331 trdp_istAlp, 332 trdp_putils.n, 333 trdp	tlc_getJoinStatistics, 221	tlc_if.c, 189
tic_getPubStatistics, 223 tic_getStatistics, 223 tic_getStatistics, 224 tic_getVersion, 224 tic_getVersionString, 224 tic_getVersionString, 224 tic_getVersionString, 224 tic_getVersionString, 224 tic_getVersionString, 224 tic_presetIndexSession, 225 tic_presetIndexSession, 226 tic_presetIndexSession, 226 tic_presetIndexSession, 227 tic_reintiSession, 228 tic_setCpTrainTopoCount, 228 tic_topmdCesSession tid_modoritic, 233 tim_detListener, 230 tim_detListener, 231 tim_detListener, 231 tim_detListener, 232 tim_detTropSocket trdp_mdcom.c, 253	,	
tic_getRedStatistics, 223 tic_getSubsStatistics, 223 tic_getSubsStatistics, 224 tic_getVersion, 224 tic_getVersion, 224 tic_getVersion, 224 tic_init, 225 tic_openSession, 225 tic_openSession, 225 tic_process, 227 tic_reinitSession, 227 tic_reinitSession, 227 tic_reinitSession, 227 tic_reinitSession, 227 tic_reinitSession, 228 tic_setCTBTopoCount, 228 tic_setCTBTopoCount, 228 tic_setCpTrainTopoCount, 228 tic_toupdateSession, 229 tim_abortSession, 230 tim_addListener, 230 tim_delListener, 230 tim_getInterval, 232 tim_getInterval, 232 tim_readdListener, 234 tim_readdListener, 234 tim_readdListener, 236 tim_request, 236 tim_request, 236 tim_request, 238 tip_potcossReceive, 239 tip_putlish, 240 tip_putl, 241 tip_putlmmediate, 242 tip_request, 243 tip_resubscribe, 244 tip_setRedundant, 245 tip_unpublish, 246 tip_unsubscribe, 246 trdp_initScats trdp_utils.c, 319 trdp_tidLis.d, 331 trdp_isiAldressed trdp_utils.c, 319 trdp_utilis.d, 331 trdp_lisiNprange trdp_utils.d, 331 trdp_bilish, 332 trdp_dCom.c, 263 trdp_mdCenekTimeouts, 258 trdp_mdCheckRimeouts, 259 trdp_mdCheckRimeouts,	— -	• —
tic_getStatistics, 223 tic_getSubsStatistics, 224 tic_getVersion, 224 tic_getVersionString, 224 tic_getVersionString, 224 tic_getVersionString, 224 tic_getVersionString, 224 tic_init, 225 tic_penSession, 225 tic_penSession, 225 tic_penSession, 226 tic_process, 227 tic_reinitSession, 227 tic_reinitSession, 227 tic_reinitSession, 228 tic_setCpTrainTopoCount, 228 tic_setCpTrainTopoCount, 228 tic_setCpTrainTopoCount, 228 tic_terminate, 229 tic_updateSession, 229 tim_abortSession, 230 tim_adolListener, 230 tim_adolListener, 230 tim_delListener, 230 tim_delListener, 231 tim_delListener, 232 tim_getInterval, 232 tim_getInterval, 232 tim_getInterval, 233 tim_repeySt, 236 tim_request, 236 tim_request, 236 tim_request, 238 tip_getRedundant, 238 tip_petRedundant, 238 tip_petRedundant, 238 tip_petRedundant, 238 tip_petRedundant, 240 tip_put_t241 tip_putImmediate, 242 tip_request, 243 tip_resubscribe, 244 tip_setFledundant, 245 tip_unpublish, 240 tip_request, 243 tip_resubscribe, 244 tip_setFledundant, 245 tip_unpublish, 240 tip_republish, 240 tip_request, 243 tip_request, 243 tip_request, 243 tip_request, 243 tip_republish, 240 tip_put_set, 243 tip_republish, 240 tip_put_set, 243 tip_request, 245 tip_unpublish, 246 tip_mdCheckListenSocks, 250 trdp_mdCheckTimeouts, 251 trdp_mdCencir, 258 trdp_mdCencir, 258 trdp_mdCencir, 258 trdp_mdCencir, 259 trdp_mdCencir, 259 trdp_mdCencir, 259 trdp_mdCheckListenSocks, 250 trdp_mdCheckList		•—
tic_gelVersion, 224 tic_getVersionString, 224 tic_getVersionString, 224 tic_init, 225 tic_openSession, 225 tic_openSession, 225 tic_openSession, 226 tic_process, 227 tic_reintISession, 228 tic_setCTBTopoCount, 228 tic_setCDTainTopoCount, 228 tic_pedcom.c, 251 trdp_mdComic_sE5 trdp_mdCom.c, 252 trdp_mdCom.c, 253 trdp_mdCom.c, 2		•—
tic_getVersion, 224 tic_getVersionString, 224 tic_getVersionString, 224 tic_getVersionString, 225 tic_openSession, 225 tic_openSession, 226 tic_presetIndexSession, 226 tic_presetIndexSession, 227 tic_resetStatistics, 228 tic_setCpTrainTopoCount, 228 tic_setCpTrainTopoCount, 228 tic_setCpTrainTopoCount, 228 tic_terminate, 229 tim_abortSession, 230 tim_addListener, 230 tim_addListener, 230 tim_ooffirm, 231 tim_ooffirm, 231 tim_process, 234 tim_reply()uery, 236 tim_request, 238 tim_reply()uery, 236 tim_request, 238 tim_process, 234 tim_reply()uery, 236 tim_request, 238 tip_getInterval, 238 tip_processReceive, 239 tip_putiblish, 240 tip_put, 241 tip_putImmediate, 242 tip_request, 243 tip_resubscribe, 244 tip_setBedundant, 245 tip_unpublish, 246 tip_unpublish, 240 tip_subscribe, 245 tip_unpublish, 246 tip_unbubscribe, 246 trdp_initScokets trdp_tilisLs, 319 trdp_jishInPrange trdp_utilis., 331 trdp_jishInPrange trdp_utilis., 332 trdp_pdcom.c, 265 trdp_mdcom.c, 255 trdp_mdCestPending, 250 trdp_mdcom.c, 255 trdp_mdCestPending, 250 trdp_mdCom.c, 255 trdp_mdCestListenSocks, 250 trdp_mdCom.c, 253 trdp_mdCenck, 255 trdp_mdCenck, 256 trdp_mitiScokets trdp_titilish, 331 trdp_statsh, 300 trdp_jsAddressed trdp_utilish, 331 trdp_statsh, 300 trdp_jsAddressed trdp_utilish, 331 trdp_pitilish, 332 trdp_packetSizePD trdp_utilish, 333 trdp_packetSizePD trdp_utilish, 332 trdp_packetSizePD trdp_utilish, 333 trdp_pdCenck, 263		• —
tic_getVersionString, 224 tic_init, 225 tic_openSession, 225 tic_presetIndex,Session, 226 tic_presetIndex,Session, 226 tic_presetIndex,Session, 227 tic_reinitSession, 227 tic_resetIstatistics, 228 tic_setETBTopoCount, 228 tic_setCpTrainTopoCount, 228 tic_setCpTrainTopoCount, 228 tic_updateSession, 229 tic_updateSession, 230 tim_addListener, 230 tim_addListener, 230 tim_delListener, 231 tim_delListener, 232 tim_delListener, 232 tim_getInterval, 232 tim_meldListener, 234 tim_request, 233 tim_request, 236 tim_request, 236 tim_request, 237 tip_getInterval, 238 tip_processSend, 239 tip_putilsh, 240 tip_put, 241 tip_putlimmediate, 242 tip_request, 243 tip_resubscribe, 245 tip_subscribe, 245 tip_subscribe, 245 tip_unpublish, 246 tip_unsubscribe, 246 trdp_initScats trdp_tils.h, 331 trdp_initStats trdp_tils.h, 331 trdp_isInlPrange trdp_tilis.h, 331 trdp_isInlPrange trdp_tilis.h, 332 trdp_docm.c, 263 trdp_packetSizePD trdp_tilis.h, 332 trdp_pitlis.h, 333 trdp_pdleck trdp_pitlis.h, 332 trdp_pitlis.h, 332 trdp_pitlis.h, 333 trdp_pitlis.h, 334 trdp_pitlis.h, 335 trdp_pitlis.h, 335 trdp_pitlis.h, 336 trdp_pitlis.h, 336 trdp_pitlis.h, 33	tlc_getSubsStatistics, 224	trdp_mdcom.c, 250
tlc_init, 225 tlc_openSession, 225 tlc_openSession, 225 tlc_presetIndexSession, 226 tlc_process, 227 tlc_reinitSession, 227 tlc_reinitSession, 227 tlc_resetStatistics, 228 tlc_setETBTopoCount, 228 tlc_setETBTopoCount, 228 tlc_terminate, 229 tlc_updateSession, 229 tlc_updateSession, 229 tlc_updateSession, 230 tlm_addListener, 230 tlm_addListener, 230 tlm_ooffirm, 231 tlm_oellListener, 232 tlm_oellListener, 232 tlm_notify, 233 tlm_process, 234 tlm_reply, 235 tlm_replyQuery, 236 tlm_request, 236 tlm_petInterval, 238 tlp_petRedundant, 238 tlp_petRedundant, 238 tlp_porcessReceive, 239 tlp_publish, 240 tlp_put_241 tlp_put_mediate, 242 tlp_request, 243 tlp_request, 243 tlp_request, 244 tlp_setRedundant, 245 tlp_subscribe, 244 tlp_setRedundant, 245 tlp_subscribe, 245 tlp_unpublish, 246 tlp_unsubscribe, 246 trdp_tinitSockets trdp_tils_c, 318 trdp_tils_c, 319 trdp_tils_h, 332 trdp_pitlls_h, 332 trdp_pitlls_h, 332 trdp_pitlls_h, 333	tlc_getVersion, 224	trdp_mdcom.h, 257
tlc_openSession, 225 tlc_presetIndexSession, 226 tlc_process, 227 tlc_reinitSession, 227 tlc_reinitSession, 227 tlc_reinitSession, 227 tlc_reinitSession, 227 tlc_resetStatistics, 228 tlc_setETBTopoCount, 228 tlc_setCpTrainTopoCount, 228 tlc_terminate, 229 tlc_updateSession, 229 tlm_abortSession, 230 tlm_addListener, 230 tlm_addListener, 230 tlm_confirm, 231 tlm_delListener, 232 tlm_delListener, 232 tlm_getInterval, 232 tlm_readdListener, 234 tlm_readdListener, 234 tlm_readdListener, 234 tlm_replyQuery, 236 tlm_replyQuery, 236 tlm_getRedundant, 238 tlp_petRedundant, 238 tlp_petRedundant, 238 tlp_processSend, 239 tlp_publish, 240 tlp_publish, 242 tlp_republish, 242 tlp_republish, 242 tlp_republish, 242 tlp_republish, 242 tlp_republish, 244 tlp_setRedundant, 245 tlp_unubscribe, 244 tlp_unubscribe, 244 tlp_unubscribe, 246 trdp_initSockets trdp_utils.n, 331 trdp_initSlats trdp_utils.n, 331 trdp_pitsln.n, 333 trdp_pit	tlc_getVersionString, 224	trdp_mdCheckPending
tic_presetIndexSession, 226 tic_process, 227 tic_resitSession, 227 tic_resitSession, 227 tic_resetStatistics, 228 tic_setCDFTainTopoCount, 258 tid_p_mdCom., 250 tidp_mdCom., 252 tidp_mdCom., 252 tidp_mdCom., 252 tidp_mdCom., 252 tidp_mdCom., 253 tidp_mdCheckListenSocks, 250 tidp_mdCheckListenSocks, 250 tidp_mdCheckListenSocks, 250 tidp_mdCheckTimeouts, 251 tidp_mdCheckTimeouts, 258 tidp_mdCom., 253 tidp_mdCom., 253 tidp_mdCom., 254 tidp_mdCom., 254 tidp_mdCom., 253 tidp_mdCom., 250 tidp_mdCheckTimeouts, 258 tidp_mdCom., 253 tidp_mdCom., 253 tidp_mdCom., 253 tidp_mdCom., 253 tidp_mdCom., 253 tidp_mdCom., 250 tidp_mdCheckTimeouts, 258 tidp_mdCom., 253 tidp_mdCom., 250 tidp_mdCheckTimeouts, 258 tidp_mdCom., 253 tidp_mdCom., 250 tidp_md	tlc_init, 225	trdp_mdcom.c, 250
tlc_process, 227 tlc_reinitSession, 227 tlc_reinitSession, 227 tlc_resetStatistics, 228 tlc_setETBTopoCount, 228 tlc_setOpTrainTopoCount, 228 tlc_terminate, 229 tlc_updateSession, 229 tlc_updateSession, 230 tlm_abortSession, 230 tlm_abortSession, 230 tlm_delListener, 231 tlm_delListener, 232 tlm_delListener, 232 tlm_motify, 233 tlm_process, 234 tlm_process, 234 tlm_readListener, 234 tlm_reply, 235 tlm_replyQuery, 236 tlm_replyQuery, 236 tlm_getInterval, 238 tlp_getRedundant, 238 tlp_getRedundant, 238 tlp_potcessReceive, 239 tlp_put, 241 tlp_put, 241 tlp_put, 241 tlp_put, 241 tlp_put, 241 tlp_petBedundant, 245 tlp_put, 243 tlp_request, 243 tlp_request, 243 tlp_resubscribe, 244 tlp_setRedundant, 245 tlp_unsubscribe, 245 tlp_unsubscribe, 246 tlp_unsubscribe, 246 trdp_initSockets trdp_utils.h, 331 trdp_initSlats trdp_utils.h, 331 trdp_initSlats trdp_utils.c, 319 trdp_utils.h, 332 trdp_pdcom.c, 263 trdp_padcomic, 253 trdp_utils.h, 331 trdp_litsls, 336 trdp_putils.h, 332 trdp_putilis.h, 333 trdp_putilis.h, 332	tlc_openSession, 225	trdp_mdcom.h, 257
tlc_reinitSession, 227 tlc_resetStatistics, 228 tlc_setETBTopoCount, 228 tlc_setOpTrainTopoCount, 228 tlc_terminate, 229 tlc_updateSession, 230 tlm_abortSession, 230 tlm_adolListener, 230 tlm_delListener, 231 tlm_delListener, 232 tlm_getInterval, 232 tlm_getInterval, 232 tlm_readdListener, 234 tlm_reply, 235 tlm_replyQuery, 236 tlm_pet, 237 tlp_getInterval, 238 tlp_processReceive, 239 tlp_publish, 240 tlp_put_241 tlp_putImmediate, 242 tlp_request, 243 tlp_resubscribe, 244 tlp_setRedundant, 245 tlp_setSession, 258 trdp_mdcom.c, 252 trdp_mdcom.c, 252 trdp_mdcom.h, 259 trdp_mdcom.h, 259 trdp_mdcom.h, 259 trdp_mdcom.h, 259 trdp_mdcom.c, 253 trdp_mdcom.c, 247 trdp_mdCom.dell, 249 trdp_mdCheckPending, 250 trdp_mdCheckPending, 250 trdp_mdCheckPending, 250 trdp_mdCheckPending, 250 trdp_mdCheckPending, 250 trdp_mdCheckListenSocks, 250 trdp_mdCheckPending, 250 trdp_mdCheckPending, 250 trdp_mdCheckPending, 250 trdp_mdCheckPending, 257 trdp_mdCheckPending, 257 trdp_mdCheckListenSocks, 257 trdp_mdCheckListenSocks, 257 trdp_mdCheckNimeouts, 258 trdp_mdCheckPending, 257 trdp_mdCheckNimeouts, 258 trdp_mdCheckPending, 257 trdp_mdCheckNimeouts, 258	tlc_presetIndexSession, 226	trdp_mdCheckTimeouts
tlc_resetStatistics, 228 tlc_setCPTrainTopoCount, 228 tlc_tc_terminate, 229 tlc_tc_terminate, 229 tlc_updateSession, 229 tlm_abortSession, 230 tlm_adotListener, 230 tlm_adotListener, 230 tlm_confirm, 231 tlm_confirm, 231 tlm_confirm, 231 tlm_getInterval, 232 tlm_getInterval, 232 tlm_process, 234 tlm_reply, 235 tlm_replyQuery, 236 tlm_replyQuery, 236 tlm_replyQuery, 236 tlm_pettInterval, 238 tlp_pettRedundant, 238 tlp_processSend, 239 tlp_publish, 240 tlp_put, 241 tlp_setRedundant, 245 tlp_subscribe, 244 tlp_setRedundant, 245 tlp_unsubscribe, 246 trdp_intiStats trdp_utils.c, 318 trdp_utils.c, 319 trdp_istInPrange trdp_utils.c, 319 trdp_polcom.c, 251 trdp_mdCom.c, 253 trdp_mdcheckListensocks, 250 trdp_mdCheckListensocks, 250 trdp_mdCheckListensocks, 251 trdp_mdCheckListensocks, 252 trdp_mdCheckListensocks, 257 trdp_mdCheckPending, 257 trdp_mdChe	tlc_process, 227	trdp_mdcom.c, 251
tic_setETBTopoCount, 228 tic_setOpTrainTopoCount, 228 tic_setOpTrainTopoCount, 228 tic_terminate, 229 tic_updateSession, 229 tit_updateSession, 230 tim_abortSession, 230 tim_abortSession, 230 tim_addListener, 230 tim_delListener, 231 tim_delListener, 232 tim_getInterval, 232 tim_potity, 233 tim_process, 234 tim_reply, 235 tim_replyQuery, 236 tim_replyQuery, 236 tip_getRedundant, 238 tip_getRedundant, 238 tip_processSeceive, 239 tip_publish, 240 tip_put, 241 tip_publish, 242 tip_request, 243 tip_resubscribe, 244 tip_resubscribe, 244 tip_setRedundant, 245 tip_subscribe, 245 tip_unpublish, 246 tip_unsubscribe, 246 trdp_initSockets trdp_tils.c, 318 trdp_tils.c, 318 trdp_tils.c, 319 trdp_putcom.c, 251 trdp_mdCencl, 253 trdp_mdcom.c, 252 trdp_mdcom.c, 253 trdp_mdcom.c, 253 trdp_mdcom.c, 253 trdp_mdc	tlc_reinitSession, 227	trdp_mdcom.h, 258
tlc_setOpTrainTopoCount, 228 tlc_terminate, 229 tlc_updateSession, 229 tlm_abortSession, 230 tlm_addListener, 230 tlm_addListener, 230 tlm_delListener, 232 tlm_getInterval, 232 tlm_getInterval, 233 tlm_readdListener, 234 tlm_reply, 235 tlm_replyQuery, 236 tlm_replyQuery, 236 tlm_getRedundant, 238 tlp_petRedundant, 238 tlp_processRed, 239 tlp_putlish, 240 tlp_put, 241 tlp_petQuest, 243 tlp_request, 243 tlp_request, 243 tlp_request, 244 tlp_setRedundant, 245 tlp_setRedundant, 256 trdp_mdCenckPending, 257 trdp_mdCheckPending, 257 trdp_mdCheckPending, 257 trdp_mdCheckPending, 257 trdp_mdCheckPending, 257 trdp_mdCheckPending, 257 trdp_mdCheckPending, 259 trdp_mdCheckPending, 257 trdp_mdCheckPending, 259 trdp_mdCheckPending, 259 trdp	tlc_resetStatistics, 228	trdp_mdConfirm
tlc_setOpTrainTopoCount, 228 tlc_terminate, 229 tlc_updateSession, 229 tlm_abortSession, 230 tlm_addListener, 230 tlm_addListener, 230 tlm_delListener, 232 tlm_getInterval, 232 tlm_getInterval, 233 tlm_readdListener, 234 tlm_reply, 235 tlm_replyQuery, 236 tlm_replyQuery, 236 tlm_getRedundant, 238 tlp_petRedundant, 238 tlp_processRed, 239 tlp_putlish, 240 tlp_put, 241 tlp_petQuest, 243 tlp_request, 243 tlp_request, 243 tlp_request, 244 tlp_setRedundant, 245 tlp_setRedundant, 256 trdp_mdCenckPending, 257 trdp_mdCheckPending, 257 trdp_mdCheckPending, 257 trdp_mdCheckPending, 257 trdp_mdCheckPending, 257 trdp_mdCheckPending, 257 trdp_mdCheckPending, 259 trdp_mdCheckPending, 257 trdp_mdCheckPending, 259 trdp_mdCheckPending, 259 trdp	tlc_setETBTopoCount, 228	trdp_mdcom.c, 251
tlc_terminate, 229 tlc_updateSession, 229 tlc_updateSession, 230 tlm_adotListener, 230 tlm_adotListener, 230 tlm_confirm, 231 tlm_cellistener, 232 tlm_getInterval, 232 tlm_process, 234 tlm_readdListener, 234 tlm_reply, 235 tlm_replyQuery, 236 tlm_replyQuery, 236 tlm_request, 236 tlp_getInterval, 238 tlp_getRedundant, 238 tlp_processReceive, 239 tlp_publish, 240 tlp_put, 241 tlp_putlmediate, 242 tlp_request, 243 tlp_request, 243 tlp_request, 244 tlp_resubscribe, 244 tlp_resubscribe, 245 tlp_unpublish, 246 trdp_initSockets trdp_utils.c, 318 trdp_midSend, 259 trdp_mdCendCheckPending, 256 trdp_mdCom.c, 253 trdp_mdCheckPending, 256 trdp_mdComfirm, 251 trdp_mdCenfTCPSocket, 252 trdp_mdCheckPending, 250 trdp_mdCheckPending, 250 trdp_mdCheckPending, 250 trdp_mdCendCheckPending, 251 trdp_mdCend, 253 trdp_mdCendCheckPending, 250 trdp_mdCend, 253 trdp_mdCendCheckPending, 257 trdp_mdCend, 253 trdp_mdCend, 253 trdp_mdCend, 253 trdp_mdCend, 253 trdp_mdCend, 253 trdp_mdCend, 253 trdp_mdCendCheckListenSocks, 257 trdp_mdCendCheckListenSocks, 257 trdp_mdCheckListenSocks, 257 trdp_mdCheckTimeouts, 258 trdp_mdCheckListenSocks, 257 trdp_mdCheckListen	_ ,	• —
tlc_updateSession, 229 tlm_abortSession, 230 tlm_adoListener, 230 tlm_adoListener, 230 tlm_confirm, 231 tlm_confirm, 231 tlm_delListener, 232 tlm_getInterval, 232 tlm_petInterval, 232 tlm_process, 234 tlm_readdListener, 234 tlm_request, 236 tlm_request, 236 tlm_request, 236 tlm_getInterval, 238 tlp_getRedundant, 238 tlp_processReceive, 239 tlp_processReceive, 239 tlp_publish, 240 tlp_put, 241 tlp_request, 243 tlp_request, 243 tlp_resubscribe, 244 tlp_setRedundant, 245 tlp_unsubscribe, 246 trdp_initSockets trdp_unitSats trdp_initStats trdp_stats.c, 297 trdp_stats.h, 300 trdp_indCenck trdp_mdCenck trdp_mdCenck, 259 trdp_mdCheckListenSocks, 259 trdp_mdCheckListenSocks, 259 trdp_mdConfirm, 251 trdp_mdConfirm, 251 trdp_mdCentCreSocket, 252 trdp_mdGetTCPSocket, 252 trdp_mdCheckPending, 257 trdp_mdCheckPending, 257 trdp_mdCentCreSocket, 258 trdp_mdCentCreSocket, 258 trdp_mdCentCreSocket, 259 trdp_mdSend, 260 trdp_mdCentCreSocket, 259 trdp_mdCentCreSocket, 259 trdp_mdCentCreSocket, 259 trdp_mdCentCreSocket, 259 trdp_mdSend, 260 trdp_packetSizeMD trdp_utils.c, 319 trdp_initSoc, 319 trdp_utils.h, 331 trdp_initSoc, 319 trdp_utils.h, 332 trdp_pdCenck trdp_pdcom.c, 263		. —
ttm_abortSession, 230 ttm_addListener, 230 ttm_addListener, 230 ttm_delListener, 231 ttm_delListener, 232 ttm_delListener, 232 ttm_getInterval, 232 ttm_petInterval, 232 ttm_petInterval, 233 ttm_readdListener, 234 ttm_readdListener, 234 ttm_reply, 235 ttm_replyQuery, 236 ttm_replyQuery, 236 ttm_replyQuery, 236 ttm_replyQuery, 236 ttm_petInterval, 238 ttp_getRedundant, 238 ttp_petRedundant, 238 ttp_processReceive, 239 ttp_publish, 240 ttp_publish, 240 ttp_put, 241 ttp_putImmediate, 242 ttp_republish, 242 ttp_request, 243 ttp_resubscribe, 244 ttp_setRedundant, 245 ttp_subscribe, 245 ttp_unpublish, 246 ttp_unsubscribe, 246 ttp_unsubscribe, 246 ttdp_initSockets trdp_initSockets trdp_stats.c, 297 trdp_stats.h, 331 trdp_initStats trdp_stats.c, 297 trdp_stats.h, 331 trdp_initStats trdp_utils.h, 331 trdp_initStats trdp_utils.h, 331 trdp_initStats trdp_utils.c, 319 trdp_utils.h, 332 trdp_pdCom.c, 263		• =
tlm_addListener, 230 tlm_confirm, 231 tlm_delListener, 232 tlm_getInterval, 232 tlm_getInterval, 233 tlm_process, 234 tlm_reply, 233 tlm_reply, 235 tlm_replyQuery, 236 tlm_request, 236 tlp_getInterval, 238 tlp_getRedundant, 238 tlp_processReceive, 239 tlp_publish, 240 tlp_put_getInterval, 241 tlp_request, 243 tlp_resubscribe, 244 tlp_setRedundant, 245 tlp_subscribe, 245 tlp_unpublish, 246 trdp_intiStats trdp_unblisch, 331 trdp_intiStats trdp_intiStats trdp_itils.c, 319 trdp_intiSt, 331 trdp_intiStats trdp_itils.c, 319 trdp_intiSt, 331 trdp_intiSt, 332 trdp_intiSt, 333 trdp_packetSizePD trdp_utils, 333	_ ·	• —
tlm_confirm, 231 tlm_delListener, 232 tlm_getInterval, 232 tlm_potify, 233 tlm_process, 234 tlm_readdListener, 234 tlm_readdListener, 234 tlm_readdListener, 234 tlm_request, 235 tlm_reply, 235 tlm_request, 236 tlm_request, 236 tlp_get, 237 tlp_getInterval, 238 tlp_getRedundant, 238 tlp_processRedive, 239 tlp_processRedive, 239 tlp_publish, 240 tlp_put, 241 tlp_putlmmediate, 242 tlp_request, 243 tlp_request, 243 tlp_resubscribe, 244 tlp_setRedundant, 245 tlp_subscribe, 245 tlp_unpublish, 246 tlp_unsubscribe, 246 trdp_intlScokets trdp_utils.c, 318 trdp_mdCent_Sessed trdp_utils.c, 319 trdp_intlScokets trdp_utils.c, 319 trdp_utils.h, 331 trdp_mdCent_Ses trdp_utils.c, 319 trdp_utils.h, 331 trdp_mdCekek trdp_utils.c, 319 trdp_utils.h, 331 trdp_getCheck trdp_utils.c, 319 trdp_utils.h, 331 trdp_pdCheck trdp_utils.c, 319 trdp_utils.h, 331 trdp_pdCheck trdp_pdCheck trdp_utils.h, 331 trdp_pdCheck trdp_utils.c, 319 trdp_utils.h, 331 trdp_pdCheck trdp_pdcom.c, 253 trdp_mdCheck trdp_utils.c, 319 trdp_utils.c, 319 trdp_utils.h, 331 trdp_pdCheck trdp_pdcom.c, 263		• —
ttm_delListener, 232 ttm_getInterval, 232 ttm_getInterval, 232 ttm_process, 234 ttm_readdListener, 234 ttm_readdListener, 234 ttm_reply, 235 ttm_replyQuery, 236 ttm_replyQuery, 236 ttm_replyQuery, 236 ttm_replyQuery, 236 ttm_get_astate, 238 ttp_get_astate, 238 ttp_getRedundant, 238 ttp_processReceive, 239 ttp_publish, 240 ttp_put_leat, 241 ttp_putImmediate, 242 ttp_republish, 242 ttp_republish, 242 ttp_resubscribe, 244 ttp_setRedundant, 245 ttp_unpublish, 246 ttp_unsubscribe, 246 ttp_unsubscribe, 246 ttp_unsubscribe, 246 ttdp_initSockets trdp_tils.c, 318 trdp_mdCetSisenSocks, 250 trdp_mdCheckPending, 250 trdp_mdCheckPimeouts, 251 trdp_mdCetTimeouts, 251 trdp_mdGetTCPSocket, 252 trdp_mdReply, 253 trdp_mdCend, 253 trdp_mdCend, 253 trdp_mdCend, 253 trdp_mdCend, 253 trdp_mdCend, 253 trdp_mdCend, 256 trdp_mdCheckListenSocks, 257 trdp_mdCend, 256 trd	-	• —
tlm_getInterval, 232 tlm_notify, 233 tlm_process, 234 tlm_readdListener, 234 tlm_readdListener, 234 tlm_reply, 235 tlm_replyQuery, 236 tlm_replyQuery, 236 tlm_request, 236 tlm_getInterval, 238 tlp_getRedundant, 238 tlp_porcessSend, 239 tlp_processSend, 239 tlp_putIlsh, 240 tlp_putIlsh, 242 tlp_request, 243 tlp_request, 243 tlp_request, 243 tlp_request, 245 tlp_setRedundant, 245 tlp_setRedundant, 245 tlp_toputDish, 246 tlp_unblish, 246 trdp_midCheckListenSocks, 257 trdp_mdCheckTimeouts, 251 trdp_mdCheckTimeouts, 251 trdp_mdGetTCPSocket, 252 trdp_mdGetTCPSocket, 252 trdp_mdGetTCPSocket, 252 trdp_mdCall, 256 trdp_mdCall, 256 trdp_mdCheckListenSocks, 257 trdp_mdCheckListenSocks, 258 trdp_utils.c, 318 trdp_utils.c, 318 trdp_utils.c, 318 trdp_utils.c, 318 trdp_initStats trdp_stats.h, 331 trdp_initStats trdp_stats.h, 300 trdp_isAddressed trdp_utils.c, 319 trdp_utils.c, 319 trdp_utils.c, 319 trdp_utils.c, 319 trdp_utils.h, 332 trdp_pdCheck trdp_pdConc, 263	-	•—
tlm_notify, 233 tlm_process, 234 tlm_process, 234 tlm_readdListener, 234 tlm_reply, 235 tlm_replyQuery, 236 tlm_request, 236 tlm_request, 236 tlp_get, 237 tlp_getInterval, 238 tlp_processReceive, 239 tlp_processReceive, 239 tlp_publish, 240 tlp_putlmmediate, 242 tlp_republish, 242 tlp_request, 243 tlp_request, 243 tlp_resubscribe, 244 tlp_setRedundant, 245 tlp_unsubscribe, 245 tlp_unsubscribe, 246 trdp_initSockets trdp_mtls, 331 trdp_mtls, 331 trdp_mtls, 331 trdp_stals, 331 trdp_stals, 331 trdp_stals, 331 trdp_stals, 331 trdp_stals, 332 trdp_utils, 331 trdp_indle, 245 trdp_utils, 331 trdp_stals, 331 trdp_stals, 331 trdp_stals, 331 trdp_intls, 331 trdp_stals, 331 trdp_stals, 331 trdp_intls, 331 trdp_stals, 331 trdp_stals, 331 trdp_putils, 331 trdp_putils, 331 trdp_stals, 331 trdp_putils, 331 trdp_packetSizePD trdp_utils, 331 trdp_utils, 333 trdp_pdCheck trdp_putils, 333 trdp_pdCheck trdp_utils, 333 trdp_pdCheck trdp_utils, 333 trdp_pdCheck trdp_utils, 333 trdp_pdCheck trdp_putils, 333 trdp_pdCheck trdp_putils, 333 trdp_pdCheck trdp_pdcm., 263		• —
tlm_process, 234 tlm_readdListener, 234 tlm_reply, 235 tlm_replyQuery, 236 tlm_replyQuery, 236 tlm_request, 236 tlm_get, 237 tlp_get, 237 tlp_get, 237 tlp_get, 238 tlp_process, 239 tlp_process, 239 tlp_process, 239 tlp_publish, 240 tlp_publish, 241 tlp_publish, 242 tlp_put, 241 tlp_request, 243 tlp_request, 243 tlp_reauest, 243 tlp_resubscribe, 244 tlp_resubscribe, 244 tlp_resubscribe, 245 tlp_unsubscribe, 246 tlp_unsubscribe, 246 trdp_initStats trdp_mtGet, 239 trdp_mdCheckPending, 250 trdp_mdCheckPimeouts, 251 trdp_mdCheckTimeouts, 251 trdp_mdGetTCPSocket, 252 trdp_mdReply, 253 trdp_mdSend, 253 trdp_mdReply, 253 trdp_mdCheckPending, 250 trdp_mdCend, 254 tlp_resubscribe, 245 tlp_reduest, 243 tlp_reduest, 245 tlp_mdCheckPending, 257 trdp_mdCheckPending, 257 trdp_mdCheckPending, 257 trdp_mdCheckPending, 257 trdp_mdCheckPending, 257 trdp_mdCheckPending, 257 trdp_mdReply, 259 trdp_mdReply, 259 trdp_mdReply, 259 trdp_mdReply, 259 trdp_mdReply, 259 trdp_mdSend, 260 trdp_stats.h, 300 trdp_stats.h, 300 trdp_stats.h, 300 trdp_stats.h, 331 trdp_isls.h, 331 trdp_utils.c, 319 trdp_utils.c, 319 trdp_utils.c, 319 trdp_utils.h, 331 trdp_utils.c, 319 trdp_utils.h, 332 trdp_utils.h, 331 trdp_utils.c, 319 trdp_utils.h, 332 trdp_pdCheck	·	• - • •
tlm_readdListener, 234 ttm_reply, 235 ttm_replyQuery, 236 ttm_replyQuery, 236 ttm_replyQuery, 236 ttm_request, 236 ttm_get, 237 ttp_get, 237 ttp_getInterval, 238 ttp_getRedundant, 238 ttp_processReceive, 239 ttp_processReceive, 239 ttp_publish, 240 ttp_put, 241 ttp_put, 241 ttp_putImmediate, 242 ttp_putImmediate, 242 ttp_request, 243 ttp_resubscribe, 244 ttp_setRedundant, 245 ttp_subscribe, 245 ttp_unpublish, 246 ttp_unpublish, 246 ttp_subscribe, 246 ttp_unsubscribe, 246 ttdp_initSockets trdp_initSockets trdp_initSats trdp_trdp_initSats trdp_trdp_initSats trdp_stats.c, 297 trdp_stats.h, 300 trdp_isAddressed trdp_utils.c, 319 trdp_utils.h, 331 trdp_initSnare trdp_utils.h, 331 trdp_initSnare trdp_utils.c, 319 trdp_utils.h, 331 trdp_jothene trdp_utils.h, 333 trdp_jothene trdp_utils.h, 332 trdp_potom.c, 263	_ ·	• —
tlm_reply, 235 ttm_replyQuery, 236 ttm_replyQuery, 236 ttm_request, 236 ttm_request, 237 ttp_get, 237 ttp_getInterval, 238 ttp_getRedundant, 238 ttp_processReceive, 239 ttp_processReceive, 239 ttp_publish, 240 ttp_putImmediate, 242 ttp_putImmediate, 242 ttp_republish, 242 ttp_request, 243 ttp_resubscribe, 244 ttp_setRedundant, 245 ttp_unpublish, 246 ttp_unpublish, 246 ttp_unpublish, 246 ttp_unpublish, 246 ttp_setRedundant, 245 ttp_resubscribe, 246 ttp_unpublish, 331 trdp_mdGetTCPSocket, 258 trdp_utils.c, 318 trdp_mdCheckPending, 257 trdp_mdCheckDimeouts, 258 trdp_utils.c, 318 trdp_utils.c, 318 trdp_mdGetTCPSocket, 259 trdp_stats.c, 297 trdp_stats.c, 297 trdp_stats.h, 300 trdp_isAddressed trdp_utils.c, 319 trdp_utils.c, 319 trdp_utils.h, 331 trdp_packetSizePD trdp_utils.h, 333 trdp_utils.h, 333 trdp_pdCheck trdp_pdChe		• —
tlm_replyQuery, 236 tlm_request, 236 tlm_request, 236 tlp_get, 237 tlp_get, 237 tlp_getlnterval, 238 tlp_getRedundant, 238 tlp_processReceive, 239 tlp_processSend, 239 tlp_publish, 240 tlp_put, 241 tlp_request, 243 tlp_request, 243 tlp_request, 243 tlp_resubscribe, 244 tlp_setRedundant, 245 tlp_setRedundant, 245 tlp_subscribe, 246 tlp_unsubscribe, 246 trdp_mdCheckIistenSocks, 250 trdp_mdCheckTimeouts, 251 trdp_mdConfirm, 251 trdp_mdGetTCPSocket, 252 trdp_mdReply, 253 trdp_mdReply, 253 tlp_reduest, 243 tlp_resubscribe, 244 tlp_setRedundant, 245 tlp_setRedundant, 245 tlp_unpublish, 246 tlp_subscribe, 246 trdp_initSockets trdp_utils.c, 318 trdp_mdCheckListenSocks, 257 trdp_mdCheckListenSocks, 258 trdp_utils.c, 318 trdp_mdReply, 259 trdp_mdReply, 250 trdp_mdCall	-	• —
tlm_request, 236 tlp_get, 237 tlp_get, 237 tlp_getInterval, 238 tlp_getRedundant, 238 tlp_processReceive, 239 tlp_processSend, 239 tlp_publish, 240 tlp_put, 241 tlp_publish, 242 tlp_putlmmediate, 242 tlp_request, 243 tlp_request, 243 tlp_resubscribe, 244 tlp_setRedundant, 245 tlp_unsubscribe, 245 tlp_unsubscribe, 246 tlp_unsubscribe, 246 trdp_mdCheckListenSocks, 250 trdp_mdCheckPending, 251 trdp_mdCheckPerding, 251 trdp_mdGetTCPSocket, 252 trdp_mdReply, 253 trdp_mdReply, 253 trdp_mdCom.h, 254 tlp_request, 243 tlp_resubscribe, 244 tlp_setRedundant, 245 tlp_setRedundant, 245 tlp_unsubscribe, 246 tlp_unsubscribe, 246 trdp_mdCheckListenSocks, 257 tlp_unpublish, 246 trdp_mdCheckListenSocks, 257 trdp_mdCheckListenSocks, 259 trdp_mdCheckListenSocks, 259 trdp_mdCheckListenSocks, 257 trdp_mdCheckListenSocks, 259 trdp_mdCheckListenSocks, 250 trdp_mdCheckListenSocks	— · ·	• —
tlp_get, 237 tlp_getInterval, 238 tlp_getRedundant, 238 tlp_processReceive, 239 tlp_processSend, 239 tlp_publish, 240 tlp_put, 241 tlp_putImmediate, 242 tlp_republish, 242 tlp_republish, 242 tlp_republish, 243 tlp_resubscribe, 244 tlp_setRedundant, 245 tlp_subscribe, 245 tlp_unsubscribe, 246 trdp_intiSockets trdp_intiSockets trdp_intiSockets trdp_intiStats trdp_intiStats trdp_intiStats trdp_intiSockets trdp_intiSockets trdp_intiStats trdp_intiSockets trdp_intiSockets trdp_intiSockets trdp_intiStats trdp_intiSockets trdp_intiSockets trdp_intiStats trdp_intiSockets trdp_intiStats trdp_intiSockets trdp_intiStats trdp_intiSockets trdp_intiStats trdp_intiSockets trdp_intiStats trdp_intiStats trdp_intiSockets trdp_intiStats trdp_intiStat	_ , , _ ,	• —
tlp_getInterval, 238 tlp_getRedundant, 238 tlp_getRedundant, 238 tlp_processReceive, 239 tlp_processSend, 239 tlp_processSend, 239 tlp_publish, 240 tlp_put, 241 tlp_putlmmediate, 242 tlp_putlmmediate, 242 tlp_request, 243 tlp_resubscribe, 244 tlp_setRedundant, 245 tlp_subscribe, 245 tlp_unpublish, 246 tlp_unpublish, 246 tlp_unpublish, 246 tlp_unsubscribe, 246 trdp_initSockets trdp_utils.c, 318 trdp_utils.h, 331 trdp_mtGetTCPSocket, 259 trdp_mdCheckListenSocks, 257 trdp_mdCheckListenSocks, 257 trdp_mdCheckPending, 257 trdp_mdCheckPending, 257 trdp_mdCheckPending, 257 trdp_mdCheckFimeouts, 258 trdp_utils.c, 318 trdp_mdConfirm, 258 trdp_utils.h, 331 trdp_mdReply, 259 trdp_mtReply, 259 trdp_stats.h, 300 trdp_isAddressed trdp_utils.c, 319 trdp_utils.c, 319 trdp_utils.c, 319 trdp_utils.c, 319 trdp_utils.c, 319 trdp_utils.h, 331 trdp_pdCheck trdp_pdConc., 263		• —
tlp_getRedundant, 238 tlp_processReceive, 239 tlp_processSend, 239 tlp_processSend, 239 tlp_publish, 240 tlp_publish, 241 tlp_put, 241 tlp_putlimmediate, 242 tlp_putlish, 242 tlp_request, 243 tlp_resubscribe, 244 tlp_setRedundant, 245 tlp_subscribe, 245 tlp_unpublish, 246 tlp_unpublish, 246 tlp_unpublish, 246 tlp_unsubscribe, 246 trdp_initSockets trdp_utils.c, 318 trdp_utils.c, 318 trdp_utils.c, 318 trdp_utils.c, 318 trdp_utils.c, 318 trdp_utils.c, 318 trdp_utils.c, 319 trdp_isiAddressed trdp_utils.c, 319 trdp_utils.c, 320 trdp_pdCheck trdp_pdCom.c, 263	. —•	• —
tlp_processReceive, 239 tlp_processSend, 239 tlp_publish, 240 tlp_publish, 241 tlp_publish, 242 tlp_put, 241 tlp_republish, 242 tlp_republish, 242 tlp_republish, 242 tlp_republish, 243 tlp_republish, 244 tlp_resubscribe, 244 tlp_resubscribe, 244 tlp_setRedundant, 245 tlp_unpublish, 246 tlp_unpublish, 246 tlp_unsubscribe, 246 trdp_initSockets trdp_utils.c, 318 trdp_utils.c, 318 trdp_utils.c, 297 trdp_stats.h, 300 trdp_stats.h, 300 trdp_isAddressed trdp_utils.c, 319 trdp_utils.c, 319 trdp_utils.c, 319 trdp_utils.c, 319 trdp_utils.c, 319 trdp_utils.c, 319 trdp_utils.h, 331 trdp_pdCheck trdp_pdCheck trdp_pdCheck trdp_pdCheck trdp_pdCheck trdp_utils.h, 331 trdp_packetSizePD trdp_utils.c, 319 trdp_utils.c, 319 trdp_utils.h, 333 trdp_pdCheck	•	• —
ttp_processSend, 239 ttp_publish, 240 ttp_publish, 241 ttp_put, 241 ttp_put, 241 ttp_putlmmediate, 242 ttp_putlmmediate, 242 ttp_republish, 242 ttp_republish, 242 ttp_republish, 243 ttp_request, 243 ttp_resubscribe, 244 ttp_setRedundant, 245 ttp_unpublish, 246 ttp_unpublish, 246 ttp_unsubscribe, 246 ttp_unsubscribe, 246 ttp_unsubscribe, 246 ttrdp_mdConfirm, 251 trdp_mdCept, 253 ttrdp_mdSend, 253 ttrdp_mdConl, 254 ttrdp_mdCall, 256 trdp_mdCheckListenSocks, 257 ttp_unpublish, 246 ttrdp_mdCheckPending, 257 ttp_unpublish, 246 ttp_mdConfirm, 258 trdp_mdCell, 256 trdp_mdConl, 256 trdp_mdCheckTimeouts, 258 trdp_mdConfirm, 251 trdp_mdCell, 253 trdp_mdCell, 256 trdp_mdCheckTimeouts, 258 trdp_mdCeptilis.c, 358 trdp_mdRetTCPSocket, 259 trdp_mdRetTCPSocket, 259 trdp_mdRetTCPSocket, 259 trdp_mdSend, 260 trdp_stats.c, 297 trdp_stats.h, 300 trdp_stats.h, 300 trdp_stats.h, 300 trdp_packetSizeMD trdp_utils.c, 319 trdp_utils.c, 319 trdp_utils.h, 331 trdp_packetSizePD trdp_utils.h, 331 trdp_utils.c, 320 trdp_islnIPrange trdp_utils.h, 333 trdp_pdCheck trdp_pdCom.c, 263		. —
ttp_publish, 240 ttrdp_mdFreeSession, 252 ttp_put, 241 ttrdp_mdGetTCPSocket, 252 ttp_putImmediate, 242 ttrdp_mdReply, 253 ttp_republish, 242 ttrdp_mdSend, 253 ttp_request, 243 ttp_resubscribe, 244 ttp_mdCall, 256 ttp_setRedundant, 245 ttp_setRedundant, 245 ttp_unpublish, 246 ttp_unpublish, 246 ttp_unsubscribe, 246 trdp_mdCheckTimeouts, 258 ttp_unsubscribe, 246 trdp_mdConfirm, 258 trdp_initSockets trdp_mdGetTCPSocket, 259 trdp_utils.h, 331 trdp_mdReply, 259 trdp_initStats trdp_mdReply, 259 trdp_stats.c, 297 trdp_stats.h, 300 trdp_stats.h, 300 trdp_stats.h, 300 trdp_stats.h, 331 trdp_packetSizeMD trdp_stats.h, 331 trdp_packetSizePD trdp_utils.h, 331 trdp_packetSizePD trdp_utils.h, 333 trdp_pdCheck trdp_pdCom.c, 263	• —	'
tlp_put, 241 trdp_mdGetTCPSocket, 252 tlp_putImmediate, 242 tlp_republish, 242 tlp_republish, 242 tlp_request, 243 tlp_request, 243 tlp_resubscribe, 244 tlp_setRedundant, 245 tlp_subscribe, 245 tlp_unpublish, 246 tlp_unpublish, 246 tlp_unsubscribe, 246 tlp_unsubscribe, 246 trdp_mdConfirm, 258 trdp_initSockets trdp_mdGetTCPSocket, 259 trdp_utils.n, 331 trdp_mdGetTCPSocket, 259 trdp_initStats trdp_mdGetTCPSocket, 259 trdp_stats.c, 297 trdp_stats.h, 300 trdp_stats.h, 300 trdp_stats.h, 331 trdp_packetSizeMD trdp_isAddressed trdp_utils.h, 331 trdp_packetSizePD trdp_utils.h, 331 trdp_packetSizePD trdp_isInlPrange trdp_utils.h, 333 trdp_utils.h, 333 trdp_utils.h, 333 trdp_utils.h, 333 trdp_utils.h, 333 trdp_utils.h, 333 trdp_pdCheck trdp_pdCom.c, 263	. —	• —
ttp_putImmediate, 242 ttrdp_mdReply, 253 ttp_republish, 242 ttrdp_mdSend, 253 ttp_request, 243 ttp_resubscribe, 244 ttp_resubscribe, 244 ttp_setRedundant, 245 ttp_subscribe, 245 ttp_unpublish, 246 ttp_unpublish, 246 ttp_unsubscribe, 246 trdp_initSockets trdp_utils.c, 318 trdp_utils.h, 331 trdp_mdCetTCPSocket, 259 trdp_initStats trdp_stats.c, 297 trdp_stats.h, 300 trdp_stats.h, 300 trdp_isAddressed trdp_utils.h, 331 trdp_utils.h, 331 trdp_utils.h, 331 trdp_utils.c, 319 trdp_utils.h, 331 trdp_packetSizePD trdp_utils.h, 331 trdp_utils.h, 331 trdp_pdCheck trdp_pdCheck trdp_utils.h, 332 trdp_utils.h, 331 trdp_utils.h, 333 trdp_utils.h, 333 trdp_utils.h, 333	. —	
ttp_republish, 242 ttp_request, 243 ttp_resubscribe, 244 ttp_setRedundant, 245 ttp_subscribe, 245 ttp_unpublish, 246 ttp_unpublish, 246 ttp_unsubscribe, 246 trdp_mdCheckPending, 257 ttp_unpublish, 246 trdp_mdCheckTimeouts, 258 trdp_mdConfirm, 258 trdp_utils.c, 318 trdp_utils.h, 331 trdp_mdGetTCPSocket, 259 trdp_initStats trdp_stats.c, 297 trdp_stats.h, 300 trdp_stats.h, 300 trdp_stats.h, 331 trdp_utils.c, 319 trdp_utils.h, 331 trdp_utils.c, 319 trdp_utils.h, 331 trdp_packetSizePD trdp_utils.h, 331 trdp_packetSizePD trdp_isInlPrange trdp_utils.h, 333 trdp_utils.c, 319 trdp_pdCheck trdp_pdCom.c, 263	. —	trdp_mdGetTCPSocket, 252
tlp_request, 243 tlp_resubscribe, 244 tlp_setRedundant, 245 tlp_subscribe, 245 tlp_subscribe, 245 tlp_unpublish, 246 tlp_unsubscribe, 246 tlp_unsubscribe, 246 trdp_mdCheckPending, 257 tlp_unsubscribe, 246 trdp_mdCheckTimeouts, 258 tlp_unsubscribe, 246 trdp_mdConfirm, 258 trdp_initSockets trdp_utils.c, 318 trdp_utils.h, 331 trdp_mdGetTCPSocket, 259 trdp_initStats trdp_mdReply, 259 trdp_stats.c, 297 trdp_stats.c, 297 trdp_stats.h, 300 trdp_stats.h, 300 trdp_utils.c, 319 trdp_utils.c, 319 trdp_utils.c, 319 trdp_utils.h, 331 trdp_utils.c, 320 trdp_isInIPrange trdp_utils.c, 319 trdp_pdCheck trdp_pdCom.c, 263	• —	. —
tlp_resubscribe, 244 tlp_setRedundant, 245 tlp_subscribe, 245 tlp_subscribe, 245 tlp_unpublish, 246 tlp_unsubscribe, 246 trdp_mdCheckPending, 257 tlp_unsubscribe, 246 trdp_mdCheckTimeouts, 258 tlp_unsubscribe, 246 trdp_initSockets trdp_utils.c, 318 trdp_utils.h, 331 trdp_utils.h, 331 trdp_mdReply, 259 trdp_stats.c, 297 trdp_stats.h, 300 trdp_stats.h, 300 trdp_utils.c, 319 trdp_utils.c, 319 trdp_utils.h, 331 trdp_utils.h, 332 trdp_utils.h, 333 trdp_utils.h, 333 trdp_utils.h, 331 trdp_packetSizePD trdp_utils.h, 333 trdp_utils.c, 319 trdp_odCheck trdp_pdCom.c, 263	tlp_republish, 242	trdp_mdSend, 253
tlp_setRedundant, 245 tlp_subscribe, 245 tlp_subscribe, 245 tlp_unpublish, 246 tlp_unsubscribe, 246 trdp_initSockets trdp_utils.c, 318 trdp_utils.h, 331 trdp_stats.c, 297 trdp_stats.h, 300 trdp_stats.h, 300 trdp_utils.c, 319 trdp_utils.c, 319 trdp_utils.h, 331 trdp_utils.c, 319 trdp_utils.h, 331 trdp_utils.c, 319 trdp_utils.h, 331 trdp_utils.c, 319 trdp_utils.c, 319 trdp_utils.h, 331 trdp_utils.h, 332 trdp_utils.h, 331 trdp_utils.h, 331 trdp_utils.h, 332 trdp_utils.h, 331 trdp_utils.h, 333 trdp_utils.c, 319 trdp_utils.c, 319 trdp_utils.h, 333 trdp_utils.h, 332 trdp_utils.h, 332 trdp_utils.h, 333 trdp_utils.h, 332 trdp_utils.h, 332 trdp_utils.h, 332 trdp_utils.h, 332 trdp_utils.h, 332 trdp_utils.h, 332	tlp_request, 243	trdp_mdcom.h, 254
tlp_subscribe, 245 tlp_unpublish, 246 tlp_unsubscribe, 246 tlp_unsubscribe, 246 trdp_initSockets trdp_utils.c, 318 trdp_utils.h, 331 trdp_initStats trdp_stats.c, 297 trdp_stats.h, 300 trdp_stats.h, 300 trdp_utils.c, 319 trdp_utils.c, 319 trdp_utils.h, 331 trdp_utils.c, 319 trdp_utils.h, 331 trdp_utils.c, 319 trdp_utils.c, 319 trdp_utils.c, 319 trdp_utils.h, 331 trdp_utils.c, 320 trdp_utils.c, 319 trdp_utils.h, 333 trdp_utils.h, 332 trdp_utils.h, 332 trdp_utils.h, 333 trdp_utils.h, 332 trdp_pdCheck trdp_pdcom.c, 263	tlp_resubscribe, 244	trdp_mdCall, 256
tlp_unpublish, 246 tlp_unsubscribe, 246 tlp_unsubscribe, 246 trdp_initSockets trdp_utils.c, 318 trdp_utils.h, 331 trdp_initStats trdp_mdSend, 260 trdp_stats.c, 297 trdp_isAddressed trdp_utils.h, 331 trdp_utils.c, 319 trdp_utils.h, 331 trdp_utils.c, 319 trdp_utils.c, 319 trdp_utils.h, 331 trdp_utils.c, 319 trdp_utils.c, 319 trdp_utils.h, 331 trdp_utils.c, 320 trdp_utils.c, 319 trdp_utils.c, 319 trdp_utils.c, 319 trdp_utils.c, 319 trdp_utils.c, 320 trdp_utils.h, 332 trdp_utils.h, 333 trdp_utils.h, 332 trdp_utils.h, 332 trdp_utils.h, 332 trdp_utils.h, 332 trdp_utils.h, 332 trdp_pdCheck trdp_pdcom.c, 263	tlp_setRedundant, 245	trdp_mdCheckListenSocks, 257
tlp_unsubscribe, 246 trdp_initSockets trdp_utils.c, 318 trdp_utils.h, 331 trdp_mdReply, 259 trdp_initStats trdp_mdSend, 260 trdp_stats.c, 297 trdp_isAddressed trdp_utils.h, 332 trdp_utils.h, 331 trdp_utils.c, 319 trdp_utils.c, 319 trdp_utils.h, 331 trdp_utils.c, 320 trdp_isInIPrange trdp_utils.c, 319 trdp_utils.c, 319 trdp_utils.c, 319 trdp_utils.c, 319 trdp_utils.c, 320 trdp_utils.h, 332 trdp_utils.c, 319 trdp_utils.c, 320 trdp_utils.h, 333 trdp_utils.h, 332 trdp_utils.h, 333 trdp_utils.h, 332 trdp_pdCheck trdp_pdcom.c, 263	tlp_subscribe, 245	trdp_mdCheckPending, 257
trdp_initSockets trdp_utils.c, 318 trdp_utils.h, 331 trdp_initStats trdp_stats.c, 297 trdp_isAddressed trdp_utils.h, 331 trdp_utils.c, 319 trdp_utils.c, 319 trdp_utils.h, 331 trdp_utils.c, 319 trdp_utils.c, 319 trdp_utils.h, 331 trdp_utils.h, 331 trdp_utils.c, 320 trdp_isInIPrange trdp_utils.c, 319 trdp_utils.h, 332 trdp_utils.h, 332 trdp_utils.c, 319 trdp_utils.c, 319 trdp_utils.c, 319 trdp_utils.c, 319 trdp_utils.c, 319 trdp_pdCheck trdp_utils.h, 332	tlp_unpublish, 246	trdp_mdCheckTimeouts, 258
trdp_utils.c, 318 trdp_mdGetTCPSocket, 259 trdp_utils.h, 331 trdp_mdReply, 259 trdp_initStats trdp_mdSend, 260 trdp_stats.c, 297 trdp_stats.h, 300 trdp_stats.h, 300 trdp_utils.c, 319 trdp_utils.c, 319 trdp_utils.c, 319 trdp_utils.h, 331 trdp_utils.h, 331 trdp_utils.c, 319 trdp_utils.c, 319 trdp_utils.c, 319 trdp_utils.c, 320 trdp_utils.c, 319 trdp_utils.c, 319 trdp_utils.c, 319 trdp_utils.h, 333 trdp_utils.h, 332 trdp_pdCheck trdp_pdcom.c, 263	tlp_unsubscribe, 246	trdp_mdConfirm, 258
trdp_utils.h, 331 trdp_mdReply, 259 trdp_initStats trdp_mdSend, 260 trdp_stats.c, 297 trdp_stats.h, 300 trdp_isAddressed trdp_utils.c, 319 trdp_utils.c, 319 trdp_utils.h, 332 trdp_utils.h, 331 trdp_utils.h, 331 trdp_utils.c, 319 trdp_utils.c, 319 trdp_utils.c, 320 trdp_utils.c, 319 trdp_utils.c, 319 trdp_utils.c, 319 trdp_pdCheck trdp_utils.h, 332	trdp_initSockets	trdp_mdFreeSession, 258
trdp_utils.h, 331 trdp_mdReply, 259 trdp_initStats trdp_mdSend, 260 trdp_stats.c, 297 trdp_stats.h, 300 trdp_isAddressed trdp_utils.c, 319 trdp_utils.c, 319 trdp_utils.h, 332 trdp_utils.h, 331 trdp_utils.h, 331 trdp_utils.c, 319 trdp_utils.c, 319 trdp_utils.c, 320 trdp_utils.c, 319 trdp_utils.c, 319 trdp_utils.c, 319 trdp_pdCheck trdp_utils.h, 332	trdp_utils.c, 318	trdp_mdGetTCPSocket, 259
trdp_initStats trdp_mdSend, 260 trdp_stats.c, 297 trdp_stats.h, 300 trdp_isAddressed trdp_utils.c, 319 trdp_utils.c, 319 trdp_utils.h, 331 trdp_utils.h, 331 trdp_utils.c, 319 trdp_isInIPrange trdp_utils.h, 333 trdp_utils.c, 319 trdp_pdCheck trdp_utils.h, 332 trdp_pdcom.c, 263	• —	• —
trdp_stats.c, 297 trdp_stats.h, 300 trdp_isAddressed trdp_utils.c, 319 trdp_utils.c, 319 trdp_utils.c, 319 trdp_utils.h, 331 trdp_utils.h, 331 trdp_utils.c, 319 trdp_utils.c, 319 trdp_utils.c, 319 trdp_utils.c, 319 trdp_utils.c, 319 trdp_pdCheck trdp_utils.h, 332	• —	
trdp_stats.h, 300 trdp_isAddressed trdp_utils.c, 319 trdp_utils.c, 319 trdp_utils.c, 319 trdp_utils.h, 331 trdp_utils.h, 331 trdp_utils.c, 320 trdp_isInIPrange trdp_utils.c, 319 trdp_utils.c, 319 trdp_pdCheck trdp_utils.h, 332 trdp_pdcom.c, 263	• —	• —
trdp_isAddressed trdp_utils.h, 332 trdp_utils.c, 319 trdp_utils.h, 331 trdp_utils.h, 331 trdp_isInIPrange trdp_utils.c, 320 trdp_utils.c, 319 trdp_utils.c, 319 trdp_utils.h, 332 trdp_utils.h, 332 trdp_pdCheck trdp_pdcom.c, 263	• —	• —•
trdp_utils.c, 319 trdp_utils.h, 331 trdp_utils.h, 331 trdp_islnIPrange trdp_utils.c, 319 trdp_utils.c, 319 trdp_utils.h, 332 trdp_utils.h, 332 trdp_pdCheck trdp_pdcom.c, 263	• —	• —
trdp_utils.h, 331 trdp_utils.c, 320 trdp_islnlPrange trdp_utils.h, 333 trdp_utils.c, 319 trdp_pdCheck trdp_utils.h, 332 trdp_pdcom.c, 263	• —	• —
trdp_isInIPrange trdp_utils.h, 333 trdp_utils.c, 319 trdp_utils.h, 332 trdp_pdCheck trdp_utils.h, 332 trdp_pdcom.c, 263	• —	. —
trdp_utils.c, 319 trdp_pdCheck trdp_utils.h, 332 trdp_pdcom.c, 263	. —	• —
trdp_utils.h, 332 trdp_pdcom.c, 263	• — •	• —
• –	• —	. —
trup_is variuo ession trup_puconi.n, 273	• —	. —
	114P_13 ValidOe331011	114p_p466111.11, 275

trdp_pdCheckListenSocks	trdp_pdCheckPending, 274
trdp_pdcom.c, 263	trdp_pdDistribute, 274
trdp_pdcom.h, 274	trdp_pdHandleTimeOuts, 275
trdp_pdCheckPending	trdp_pdInit, 276
trdp_pdcom.c, 264	trdp_pdPut, 277
trdp_pdcom.h, 274	trdp_pdReceive, 277
trdp_pdDistribute	trdp_pdSend, 278
trdp_pdcom.c, 264	trdp_pdSendElement, 278
trdp_pdcom.h, 274	trdp_pdSendImmediate, 279
trdp_pdHandleTimeOuts	trdp_pdSendQueued, 280
trdp pdcom.c, 265	trdp_pdUpdate, 280
trdp_pdcom.h, 275	trdp pdindex.c, 281
trdp_pdInit	trdp_pdindex.h, 282
trdp_pdcom.c, 266	TRDP_DEFAULT_INDEX_SIZES, 284
trdp_pdcom.h, 276	trdp_private.h, 285
trdp_pdeornin, 270 trdp_pdPrepareStats	TRDP_MAX_PD_SOCKET_CNT, 288
trdp_stats.c, 298	TRDP_MD_ELE_ST_T, 288
trdp_stats.b, 301	TRDP_SOCK_TYPE_T, 288
•	
trdp_pdPut	trdp_queueAppLast
trdp_pdcom.c, 266	trdp_utils.c, 320
trdp_pdcom.h, 277	trdp_utils.h, 333
trdp_pdReceive	trdp_queueDelElement
trdp_pdcom.c, 267	trdp_utils.c, 320
trdp_pdcom.h, 277	trdp_utils.h, 333
trdp_pdSend	trdp_queueFindComId
trdp_pdcom.c, 267	trdp_utils.c, 321
trdp_pdcom.h, 278	trdp_utils.h, 334
trdp_pdSendElement	trdp_queueFindExistingSub
trdp_pdcom.c, 268	trdp_utils.c, 321
trdp_pdcom.h, 278	trdp_utils.h, 334
trdp_pdSendImmediate	trdp_queueFindPubAddr
trdp_pdcom.c, 268	trdp_utils.c, 321
trdp_pdcom.h, 279	trdp_utils.h, 334
trdp_pdSendQueued	trdp_queueFindSubAddr
trdp_pdcom.c, 269	trdp_utils.c, 322
trdp_pdcom.h, 280	trdp utils.h, 335
trdp_pdUpdate	trdp_queueInsFirst
trdp_pdcom.c, 270	trdp_utils.c, 322
trdp_pdcom.h, 280	trdp_utils.h, 335
trdp_pdcom.c, 260	trdp_utils.rr, 555
• —	• —
trdp_handleTimeout, 262	tlc_if.c, 189
trdp_pdCheck, 263	trdp_releaseSocket
trdp_pdCheckListenSocks, 263	trdp_utils.c, 323
trdp_pdCheckPending, 264	trdp_utils.h, 336
trdp_pdDistribute, 264	trdp_requestSocket
trdp_pdHandleTimeOuts, 265	trdp_utils.c, 323
trdp_pdInit, 266	trdp_utils.h, 336
trdp_pdPut, 266	trdp_resetSequenceCounter
trdp_pdReceive, 267	trdp_utils.c, 324
trdp_pdSend, 267	trdp_utils.h, 337
trdp_pdSendElement, 268	trdp_serviceRegistry.h, 289
trdp_pdSendImmediate, 268	SOA_SAME_SERVICEID, 292
trdp_pdSendQueued, 269	SRM_SERVICE_READ_REQ_COMID, 292
trdp_pdUpdate, 270	SRM SRVINFO NOTIFY COMID, 292
trdp_pdopdate, 270	trdp_sessionQueue
trdp_handleTimeout, 273	tlc_if.c, 190
• —	
trdp_pdCheck, 273	tlc_if.h, 192
trdp_pdCheckListenSocks, 274	trdp_stats.c, 293

	tlc_getJoinStatistics, 294	trdp_SockDelJoin, 338
	tlc_getPubStatistics, 295	trdp_SockIsJoined, 338
	tlc_getRedStatistics, 295	trdp_checkSequenceCounter, 329
	tlc_getStatistics, 296	trdp_findMCjoins, 330
	tlc_getSubsStatistics, 296	trdp_findSubAddr, 330
	tlc_resetStatistics, 297	trdp_getSeqCnt, 331
	trdp_UpdateStats, 298	trdp_initSockets, 331
	trdp_initStats, 297	trdp_isAddressed, 331
	trdp_pdPrepareStats, 298	trdp_isInIPrange, 332
trdp	stats.h, 299	trdp_packetSizeMD, 332
	trdp_initStats, 300	trdp_packetSizePD, 333
	trdp pdPrepareStats, 301	trdp_queueAppLast, 333
trdn	tsn_def.h, 302	trdp_queueDelElement, 333
ιιαρ_	TRDP_MIN_PD2_HEADER_SIZE, 303	trdp_queueFindComId, 334
	TRDP_MSG_TSN_PD, 303	trdp_queueFindExistingSub, 334
	TRDP_PD_DEFAULT_QOS, 303	trdp_queueFindPubAddr, 334
trap_	_types.h, 303	trdp_queueFindSubAddr, 335
	TRDP_DATA_TYPE_T, 311	trdp_queueInsFirst, 335
	TRDP_ERR_T, 312	trdp_releaseSocket, 336
	TRDP_FLAGS_DEFAULT, 309	trdp_requestSocket, 336
	TRDP_IP_ADDR_T, 309	trdp_resetSequenceCounter, 337
	TRDP_MARSHALL_T, 309	trdp_validTopoCounters, 339
	TRDP_MD_CALLBACK_T, 310	trdp_validTopoCounters
	TRDP_PD_CALLBACK_T, 310	trdp_utils.c, 326
	TRDP_PRINT_DBG_T, 310	trdp_utils.h, 339
	TRDP_RED_STATE_T, 313	trdp_xml.c, 339
	TRDP REPLY STATUS T, 313	trdp_XMLClose, 341
	TRDP TIME T, 310	trdp_XMLCountStartTag, 341
	TRDP_TO_BEHAVIOR_T, 313	trdp_XMLEnter, 342
	TRDP_UNMARSHALL_T, 310	trdp_XMLGetAttribute, 342
trdn	utils.c, 314	trdp_XMLLeave, 343
iiup_		trdp_XMLMemOpen, 343
	printSocketUsage, 316	• —
	trdp_SockAddJoin, 325	trdp_XMLOpen, 343
	trdp_SockDelJoin, 325	trdp_XMLRewind, 344
	trdp_SockIsJoined, 325	trdp_XMLSeekStartTag, 344
	trdp_checkSequenceCounter, 316	trdp_XMLSeekStartTagAny, 345
	trdp_findMCjoins, 317	trdp_xml.h, 345
	trdp_findSubAddr, 317	trdp_XMLClose, 347
	trdp_getSeqCnt, 318	trdp_XMLCountStartTag, 348
	trdp_initSockets, 318	trdp_XMLEnter, 348
	trdp_isAddressed, 319	trdp_XMLGetAttribute, 348
	trdp_isInIPrange, 319	trdp_XMLLeave, 349
	trdp_packetSizeMD, 319	trdp_XMLMemOpen, 349
	trdp_packetSizePD, 320	trdp_XMLOpen, 350
	trdp_queueAppLast, 320	trdp_XMLRewind, 350
	trdp_queueDelElement, 320	trdp_XMLSeekStartTag, 350
	trdp queueFindComld, 321	trdp XMLSeekStartTagAny, 351
	trdp_queueFindExistingSub, 321	trnCstNo
	trdp_queueFindPubAddr, 321	GNU PACKED, 30
	. — .	-
	trdp_queueFindSubAddr, 322	trnDirState
	trdp_queueInsFirst, 322	GNU_PACKED, 30
	trdp_releaseSocket, 323	trnld
	trdp_requestSocket, 323	GNU_PACKED, 30
	trdp_resetSequenceCounter, 324	trnNetDir
	trdp_validTopoCounters, 326	GNU_PACKED, 30
trdp_	_utils.h, 326	trnOperator
	printSocketUsage, 329	GNU_PACKED, 30
	trdp_SockAddJoin, 338	trnTopoCnt

GNU_PACKED, 31	vos_getInterfaces
trnVehNo	vos_sock.h, 377
GNU_PACKED, 31	vos_getRealTime
	vos_thread.h, 394
usage	vos_getTime
TRDP_SOCKETS, 64	vos_thread.h, 394
V00 555 5	vos_getTimeStamp
VOS_ERR_T	vos_thread.h, 395
vos_types.h, 406	vos getUuid
VOS_LOG_T	vos_thread.h, 395
vos_types.h, 406	vos_getVersion
VOS_MAX_ERR_STR_SIZE	vos_utils.c, 409
vos_utils.h, 413	vos_utils.h, 415
VOS_MAX_FRMT_SIZE	vos_getVersionString
vos_utils.h, 413	vos_utils.c, 409
VOS_MAX_PRNT_STR_SIZE	
vos_utils.h, 413	vos_utils.h, 415
VOS_MAX_SOCKET_CNT	vos_hostIsBigEndian
vos_sock.h, 376	vos_utils.c, 409
VOS_MEM_MAX_PREALLOCATE	vos_utils.h, 416
vos_mem.h, 363	vos_htonl
VOS_MEM_PREALLOCATE	vos_sock.h, 377
vos_mem.h, 363	vos_htonll
VOS_PRINT_DBG_T	vos_sock.h, 378
vos_types.h, 405	vos_htons
VOS_SOCK_OPT_T, 66	vos_sock.h, 378
	vos_init
VOS_TIMEVAL_T	vos_utils.c, 410
vos_types.h, 405	vos_utils.h, 416
VOS_TTL_MULTICAST	vos_ipDotted
vos_sock.h, 376	vos_sock.h, 379
VOS_VERSION_T, 67	vos isMulticast
vehld	vos sock.h, 379
GNU_PACKED, 31	vos mem.c, 351
TRDP_VEHICLE_INFO_T, 65	
vehOrient	vos_bsearch, 353
GNU_PACKED, 31	vos_memAlloc, 354
version	vos_memCount, 354
GNU_PACKED, 31	vos_memDelete, 355
vos_addTime	vos_memFree, 355
vos_thread.h, 393	vos_memInit, 355
vos_bsearch	vos_qsort, 356
vos_mem.c, 353	vos_queueCreate, 356
vos mem.h, 363	vos_queueDestroy, 357
vos clearTime	vos_queueReceive, 357
vos_thread.h, 393	vos_queueSend, 358
vos cmpTime	vos_strncat, 358
vos_thread.h, 393	vos_strncpy, 360
vos crc32	vos strnicmp, 360
vos utils.c, 408	vos mem.h, 361
vos utils.h, 414	VOS MEM MAX PREALLOCATE, 363
_ :	VOS MEM PREALLOCATE, 363
vos_determineBindAddr	vos_bsearch, 363
vos_sock.h, 376	vos_memAlloc, 364
vos_divTime	vos_memCount, 364
vos_thread.h, 394	
vos_dottedIP	vos_memDelete, 365
vos_sock.h, 377	vos_memFree, 365
vos_getErrorString	vos_memInit, 366
vos_utils.c, 408	vos_qsort, 366
vos_utils.h, 415	vos_queueCreate, 367

Destruction 2027	one of the Ad 7
vos_queueDestroy, 367	vos_utils.h, 417
vos_queueReceive, 368	vos_select
vos_queueSend, 368	vos_sock.h, 381
vos_strncat, 369 vos_strncpy, 369	vos_semaCreate vos_thread.h, 397
vos strnicmp, 370	vos_tillead.fl, 397
vos_striicinp, 370 vos memAlloc	_
-	vos_thread.h, 398 vos_semaGive
vos_mem.c, 354 vos_mem.h, 364	vos thread.h, 398
vos_memCount	vos semaTake
vos_mem.c, 354	vos thread.h, 399
vos mem.h, 364	vos_shared_mem.h, 370
vos memDelete	vos sharedClose, 372
vos_mem.c, 355	vos_sharedOpen, 372
vos mem.h, 365	vos sharedClose
vos memFree	vos_shared_mem.h, 372
vos_mem.c, 355	vos sharedOpen
vos mem.h, 365	vos shared mem.h, 372
vos memInit	vos sock.h, 373
vos mem.c, 355	VOS MAX SOCKET CNT, 376
vos_mem.h, 366	VOS TTL MULTICAST, 376
vos mulTime	vos determineBindAddr, 376
vos thread.h, 395	vos_dottedIP, 377
vos mutexCreate	vos getInterfaces, 377
vos_thread.h, 395	vos_htonl, 377
vos_mutexDelete	vos_htonll, 378
vos thread.h, 396	vos htons, 378
vos mutexLock	vos_ipDotted, 379
vos thread.h, 396	vos isMulticast, 379
vos_mutexTryLock	vos_netlfUp, 379
vos thread.h, 397	vos ntohl, 380
vos mutexUnlock	vos_ntohll, 380
vos_thread.h, 397	vos_ntohs, 380
vos netlfUp	vos select, 381
vos_sock.h, 379	vos_sockAccept, 381
vos_ntohl	vos_sockBind, 382
vos_sock.h, 380	vos_sockClose, 382
vos ntohll	vos sockConnect, 383
vos sock.h, 380	vos sockGetMAC, 383
vos ntohs	vos_sockInit, 384
vos sock.h, 380	vos sockJoinMC, 384
vos qsort	vos sockLeaveMC, 384
vos_mem.c, 356	vos_sockListen, 385
vos_mem.h, 366	vos_sockOpenTCP, 385
vos_queueCreate	vos_sockOpenUDP, 386
vos_mem.c, 356	vos_sockReceiveTCP, 386
vos_mem.h, 367	vos_sockReceiveUDP, 387
vos_queueDestroy	vos_sockSendTCP, 388
vos_mem.c, 357	vos_sockSendUDP, 388
vos_mem.h, 367	vos_sockSetMulticastIf, 389
vos_queueReceive	vos_sockSetOptions, 389
vos_mem.c, 357	vos_sockTerm, 390
vos_mem.h, 368	vos_sockAccept
vos_queueSend	vos_sock.h, 381
vos_mem.c, 358	vos_sockBind
vos_mem.h, 368	vos_sock.h, 382
vos_sc32	vos_sockClose
vos_utils.c, 410	vos_sock.h, 382

vos_sockConnect	vos_mutexUnlock, 397
vos_sock.h, 383	vos_semaCreate, 397
vos_sockGetMAC	vos_semaDelete, 398
vos_sock.h, 383	vos_semaGive, 398
vos_sockInit	vos_semaTake, 399
vos_sock.h, 384	vos_subTime, 399
vos_sockJoinMC	vos_threadCreate, 399
vos_sock.h, 384	vos_threadCreateSync, 400
vos_sockLeaveMC	vos_threadDelay, 401
vos_sock.h, 384	vos_threadInit, 401
vos_sockListen	vos_threadIsActive, 402
vos_sock.h, 385	vos_threadSelf, 402
vos_sockOpenTCP	vos_threadTerm, 402
vos_sock.h, 385	vos_threadTerminate, 402
vos_sockOpenUDP	vos_threadCreate
vos_sock.h, 386	vos_thread.h, 399
vos_sockReceiveTCP	vos_threadCreateSync
vos_sock.h, 386	vos_thread.h, 400
vos_sockReceiveUDP	vos_threadDelay
vos_sock.h, 387	vos_thread.h, 401
vos_sockSendTCP	vos_threadInit
vos_sock.h, 388	vos_thread.h, 401
vos_sockSendUDP	vos_threadIsActive
vos sock.h, 388	vos_thread.h, 402
vos sockSetMulticastIf	vos_threadSelf
vos_sock.h, 389	vos_thread.h, 402
vos_sockSetOptions	vos_threadTerm
vos_sock.h, 389	vos_thread.h, 402
vos sockTerm	vos_threadTerminate
vos_sock.h, 390	vos_thread.h, 402
vos_strncat	vos_types.h, 403
vos_mem.c, 358	VOS_ERR_T, 406
vos_mem.h, 369	VOS_LOG_T, 406
vos strncpy	VOS_PRINT_DBG_T, 405
vos_mem.c, 360	VOS_TIMEVAL_T, 405
vos mem.h, 369	vos_utils.c, 407
vos_strnicmp	vos_crc32, 408
vos_mem.c, 360	vos_getErrorString, 408
vos_mem.h, 370	vos_getVersion, 409
vos subTime	vos_getVersionString, 409
vos_thread.h, 399	vos_hostIsBigEndian, 409
vos terminate	vos_init, 410
vos_utils.c, 411	vos_sc32, 410
vos utils.h, 417	vos_terminate, 411
vos_thread.h, 390	vos_utils.h, 411
vos_addTime, 393	INITFCS, 413
vos_clearTime, 393	VOS_MAX_ERR_STR_SIZE, 413
vos_cmpTime, 393	VOS_MAX_FRMT_SIZE, 413
vos_divTime, 394	VOS_MAX_PRNT_STR_SIZE, 413
vos_getRealTime, 394	vos_crc32, 414
vos_getTime, 394	vos_getErrorString, 415
vos_getTime, 394 vos_getTimeStamp, 395	vos_getVersion, 415
vos_getUuid, 395	vos_getVersionString, 415
vos gerouid, 395 vos mulTime, 395	vos_hostIsBigEndian, 416
-	vos_init, 416
vos_mutexCreate, 395	vos_sc32, 417
vos_mutexLede, 396	vos_terminate, 417
vos_mutexLock, 396 vos_mutexTryLock, 397	
VOS_ITIULEX IT YEOCK, SS/	