# TCNOpen TRDP Light Draft V2.0.0

Generated by Doxygen 1.8.13

## **Contents**

1	The	e TRDP Light Library API Specification						
	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	37
	4.6.2	Field Documentation	37
		4.6.2.1 fctDev	37
4.7	service	RegistryEntry Struct Reference	37
	4.7.1	Detailed Description	38
4.8	srv_inf	p_req Struct Reference	38
	4.8.1	Detailed Description	38
4.9	TAU_N	MARSHALL_INFO_T Struct Reference	38
	4.9.1	Detailed Description	39
4.10	TCN_U	JRI Struct Reference	39
	4.10.1	Detailed Description	39
4.11	TRDP_	CLTR_CST_INFO_T Struct Reference	40
	4.11.1	Detailed Description	40
4.12	TRDP_	COM_PARAM_T Struct Reference	40
	4.12.1	Detailed Description	41
4.13	TRDP_	COMID_DSID_MAP_T Struct Reference	41
	4.13.1	Detailed Description	41
4.14	TRDP_	CONSIST_INFO_T Struct Reference	41
	4.14.1	Detailed Description	42
	4.14.2	Field Documentation	43
		4.14.2.1 cstld	43
		4.14.2.2 cstOwner	43
4.15	TRDP_	DATASET Struct Reference	43
	4.15.1	Detailed Description	44
4.16	TRDP_	_DATASET_ELEMENT_T Struct Reference	44
	4.16.1	Detailed Description	45
4.17	TRDP_	_DBG_CONFIG_T Struct Reference	45
	4.17.1	Detailed Description	45
4.18	TRDP_	DNS_REPLY Struct Reference	46
	4.18.1	Detailed Description	46

CONTENTS

	4.18.2	Field Documentation	47
		4.18.2.1 tcnUriCnt	47
4.19	TRDP_	DNS_REQUEST Struct Reference	47
	4.19.1	Detailed Description	48
	4.19.2	Field Documentation	48
		4.19.2.1 tcnUriCnt	48
4.20	TRDP_	ETB_INFO_T Struct Reference	48
	4.20.1	Detailed Description	48
	4.20.2	Field Documentation	48
		4.20.2.1 cnCnt	49
4.21	TRDP_	FUNCTION_INFO_T Struct Reference	49
	4.21.1	Detailed Description	49
	4.21.2	Field Documentation	49
		4.21.2.1 cnld	50
		4.21.2.2 cstVehNo	50
		4.21.2.3 etbld	50
		4.21.2.4 fctld	50
4.22	TRDP_	HANDLE Struct Reference	50
	4.22.1	Detailed Description	51
4.23	TRDP_	MARSHALL_CONFIG_T Struct Reference	51
	4.23.1	Detailed Description	52
4.24	TRDP_	MD_CONFIG_T Struct Reference	52
	4.24.1	Detailed Description	53
4.25	TRDP_	MD_INFO_T Struct Reference	53
	4.25.1	Detailed Description	54
4.26	TRDP_	MEM_CONFIG_T Struct Reference	54
	4.26.1	Detailed Description	55
4.27	TRDP_	PD_CONFIG_T Struct Reference	55
	4.27.1	Detailed Description	56
4.28	TRDP_	PD_INFO_T Struct Reference	56

vi

	4.28.1 Detailed Description	57
4.29	TRDP_PROCESS_CONFIG_T Struct Reference	57
	4.29.1 Detailed Description	57
4.30	TRDP_PROP_T Struct Reference	57
	4.30.1 Detailed Description	58
4.31	TRDP_SDT_PAR_T Struct Reference	58
	4.31.1 Detailed Description	59
4.32	TRDP_SEQ_CNT_ENTRY_T Struct Reference	59
	4.32.1 Detailed Description	59
4.33	TRDP_SESSION Struct Reference	59
	4.33.1 Detailed Description	60
4.34	TRDP_SOCKET_TCP Struct Reference	61
	4.34.1 Detailed Description	61
4.35	TRDP_SOCKETS Struct Reference	61
	4.35.1 Detailed Description	62
	4.35.2 Field Documentation	62
	4.35.2.1 usage	62
4.36	TRDP_VEHICLE_INFO_T Struct Reference	63
	4.36.1 Detailed Description	63
	4.36.2 Field Documentation	63
	4.36.2.1 vehld	64
4.37	TRDP_XML_DOC_HANDLE_T Struct Reference	64
	4.37.1 Detailed Description	64
4.38	VOS_SOCK_OPT_T Struct Reference	64
	4.38.1 Detailed Description	65
4.39	VOS_VERSION_T Struct Reference	65
	4.39.1 Detailed Description	65

CONTENTS vii

5	File	Docum	entation		67
	5.1	iec613	75-2-3.h F	ile Reference	67
		5.1.1	Detailed	Description	71
		5.1.2	Macro De	efinition Documentation	71
			5.1.2.1	ETB_CTRL_COMID	71
			5.1.2.2	TRDP_ETBCTRL_DSID	72
			5.1.2.3	TRDP_MAX_FILE_NAME_LEN	72
			5.1.2.4	TRDP_MAX_LABEL_LEN	72
			5.1.2.5	TRDP_MAX_MD_DATA_SIZE	72
			5.1.2.6	TRDP_MAX_URI_HOST_LEN	72
			5.1.2.7	TRDP_MAX_URI_LEN	72
			5.1.2.8	TRDP_MAX_URI_USER_LEN	73
			5.1.2.9	TRDP_MD_DEFAULT_REPLY_TIMEOUT	73
			5.1.2.10	TRDP_MD_INFINITE_TIME	73
			5.1.2.11	TRDP_MIN_PD_HEADER_SIZE	73
			5.1.2.12	TRDP_MSG_PD	73
			5.1.2.13	TRDP_PD_UDP_PORT	73
			5.1.2.14	TRDP_PROCESS_DEFAULT_CYCLE_TIME	74
			5.1.2.15	TRDP_USR_URI_SIZE	74
			5.1.2.16	TTDB_NET_DIR_REQ_COMID	74
			5.1.2.17	TTDB_OP_DIR_INFO_COMID	74
			5.1.2.18	TTDB_STAT_CST_REQ_COMID	74
			5.1.2.19	TTDB_TRN_DIR_REQ_COMID	74
	5.2	tau_cs	tinfo.c File	Reference	75
		5.2.1	Detailed	Description	76
		5.2.2	Function	Documentation	76
			5.2.2.1	cstInfoGetPropSize()	76
	5.3	tau_ctr	l.c File Re	ference	77
		5.3.1	Detailed	Description	78
		5.3.2	Function	Documentation	79

viii CONTENTS

		5.3.2.1	tau_getEcspStat()	79
		5.3.2.2	tau_initEcspCtrl()	79
		5.3.2.3	tau_requestEcspConfirm()	80
		5.3.2.4	tau_setEcspCtrl()	80
		5.3.2.5	tau_terminateEcspCtrl()	8
5.4	tau_ctr	rl.h File Re	eference	8
	5.4.1	Detailed	Description	83
	5.4.2	Function	n Documentation	84
		5.4.2.1	tau_getEcspStat()	84
		5.4.2.2	tau_initEcspCtrl()	84
		5.4.2.3	tau_requestEcspConfirm()	85
		5.4.2.4	tau_setEcspCtrl()	85
		5.4.2.5	tau_terminateEcspCtrl()	86
5.5	tau_ctr	rl_types.h	File Reference	86
	5.5.1	Detailed	Description	88
5.6	tau_dn	nr.c File Re	eference	
	5.6.1	Detailed	Description	90
	5.6.2	Function	n Documentation	9 <sup>-</sup>
		5.6.2.1	tau_addr2Uri()	9 <sup>-</sup>
		5.6.2.2	tau_deInitDnr()	9 <sup>-</sup>
		5.6.2.3	tau_DNRstatus()	92
		5.6.2.4	tau_getOwnAddr()	92
		5.6.2.5	tau_initDnr()	92
		5.6.2.6	tau_uri2Addr()	
5.7	tau_dn	nr.h File Re	reference	94
	5.7.1	Detailed	Description	96
	5.7.2	Enumera	ration Type Documentation	96
		5.7.2.1	TRDP_DNR_OPTS	96
	5.7.3	Function	n Documentation	96
		5.7.3.1	tau_addr2Uri()	96

CONTENTS

		5.7.3.2	tau_deInitDnr()	97
		5.7.3.3	tau_DNRstatus()	98
		5.7.3.4	tau_getOwnAddr()	98
		5.7.3.5	tau_initDnr()	99
		5.7.3.6	tau_uri2Addr()	100
5.8	tau_dn	r_types.h	File Reference	101
	5.8.1	Detailed	Description	102
5.9	tau_ma	arshall.c Fi	ile Reference	103
	5.9.1	Detailed	Description	104
	5.9.2	Function	Documentation	104
		5.9.2.1	tau_calcDatasetSize()	104
		5.9.2.2	tau_calcDatasetSizeByComId()	105
		5.9.2.3	tau_initMarshall()	106
		5.9.2.4	tau_marshall()	106
		5.9.2.5	tau_marshallDs()	107
		5.9.2.6	tau_unmarshall()	108
		5.9.2.7	tau_unmarshallDs()	108
5.10	tau_ma	arshall.h F	ile Reference	109
	5.10.1	Detailed	Description	111
	5.10.2	Function	Documentation	111
		5.10.2.1	tau_calcDatasetSize()	111
		5.10.2.2	tau_calcDatasetSizeByComId()	112
		5.10.2.3	tau_initMarshall()	113
		5.10.2.4	tau_marshall()	114
		5.10.2.5	tau_marshallDs()	115
		5.10.2.6	tau_unmarshall()	116
		5.10.2.7	tau_unmarshallDs()	118
5.11	tau_so	_if.c File F	Reference	119
	5.11.1	Detailed	Description	121
	5.11.2	Function	Documentation	121

CONTENTS

		5.11.2.1	tau_addServices()	. 121
		5.11.2.2	tau_delServices()	. 122
		5.11.2.3	tau_freeServiceList()	. 122
		5.11.2.4	tau_getServiceList()	. 123
		5.11.2.5	tau_updServices()	. 123
5.12	tau_so	_if.h File R	Reference	. 124
	5.12.1	Detailed I	Description	. 125
	5.12.2	Function	Documentation	. 126
		5.12.2.1	tau_addServices()	. 126
		5.12.2.2	tau_delServices()	. 126
		5.12.2.3	tau_freeServiceList()	. 127
		5.12.2.4	tau_getServiceList()	. 127
		5.12.2.5	tau_updServices()	. 128
5.13	tau_tti.	c File Refe	erence	. 128
	5.13.1	Detailed I	Description	. 130
	5.13.2	Macro De	efinition Documentation	. 131
		5.13.2.1	TTI_CACHED_CONSISTS	. 131
	5.13.3	Function	Documentation	. 131
		5.13.3.1	tau_deInitTTI()	. 131
		5.13.3.2	tau_getCstFctCnt()	. 131
		5.13.3.3	tau_getCstFctInfo()	. 132
		5.13.3.4	tau_getCstInfo()	. 132
		5.13.3.5	tau_getCstVehCnt()	. 133
		5.13.3.6	tau_getOpTrDirectory()	. 133
		5.13.3.7	tau_getOpTrnDirectoryStatusInfo()	. 134
		5.13.3.8	tau_getOwnlds()	. 134
		5.13.3.9	tau_getOwnOpCstNo()	. 135
		5.13.3.10	tau_getOwnTrnCstNo()	. 135
		5.13.3.11	tau_getStaticCstInfo()	. 135
		5.13.3.12	2 tau_getTrDirectory()	. 136

CONTENTS xi

5.13.3.13 tau_getTrnCstCnt()	13	36
5.13.3.14 tau_getTrnVehCnt()	13	37
5.13.3.15 tau_getTTI()	13	37
5.13.3.16 tau_getVehInfo()	13	37
5.13.3.17 tau_getVehOrient()	13	38
5.13.3.18 tau_initTTlaccess()	13	38
5.14 tau_tti.h File Reference	13	39
5.14.1 Detailed Description	14	12
5.14.2 Function Documentation	14	12
5.14.2.1 tau_deInitTTI()	14	12
5.14.2.2 tau_getCstFctCnt()	14	<del>1</del> 3
5.14.2.3 tau_getCstFctInfo()	14	<del>1</del> 3
5.14.2.4 tau_getCstInfo()	14	14
5.14.2.5 tau_getCstVehCnt()	14	14
5.14.2.6 tau_getOpTrDirectory()	14	<del>1</del> 5
5.14.2.7 tau_getOpTrnDirectoryStatusInfo()	14	<del>1</del> 6
5.14.2.8 tau_getOwnlds()	14	<del>1</del> 6
5.14.2.9 tau_getOwnOpCstNo()	14	17
5.14.2.10 tau_getOwnTrnCstNo()	14	17
5.14.2.11 tau_getStaticCstInfo()	14	<del>1</del> 8
5.14.2.12 tau_getTrDirectory()	14	18
5.14.2.13 tau_getTrnCstCnt()	14	19
5.14.2.14 tau_getTrnVehCnt()	14	19
5.14.2.15 tau_getTTI()	15	50
5.14.2.16 tau_getVehInfo()	15	50
5.14.2.17 tau_getVehOrient()	15	51
5.14.2.18 tau_initTTlaccess()	15	52
5.15 tau_tti_types.h File Reference	15	53
5.15.1 Detailed Description	15	55
5.16 tau_xml.c File Reference	15	56

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_readXmlServiceConfig()
5.17	tau_xm	nl.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()
		5.17.4.7 tau_readXmlDeviceConfig()
		5.17.4.8 tau_readXmlInterfaceConfig()
		5.17.4.9 tau_readXmlServiceConfig()

CONTENTS xiii

5.18	tlc_if.c	File Reference
	5.18.1	Detailed Description
	5.18.2	Function Documentation
		5.18.2.1 tlc_closeSession()
		5.18.2.2 tlc_configSession()
		5.18.2.3 tlc_getETBTopoCount()
		5.18.2.4 tlc_getInterval()
		5.18.2.5 tlc_getOpTrainTopoCount()
		5.18.2.6 tlc_getOwnlpAddress()
		5.18.2.7 tlc_getVersion()
		5.18.2.8 tlc_getVersionString()
		5.18.2.9 tlc_init()
		5.18.2.10 tlc_openSession()
		5.18.2.11 tlc_process()
		5.18.2.12 tlc_reinitSession()
		5.18.2.13 tlc_setETBTopoCount()
		5.18.2.14 tlc_setOpTrainTopoCount()
		5.18.2.15 tlc_terminate()
		5.18.2.16 tlc_updateSession()
		5.18.2.17 trdp_getAccess()
		5.18.2.18 trdp_isValidSession()
		5.18.2.19 trdp_releaseAccess()
		5.18.2.20 trdp_sessionQueue()
5.19	tlc_if.h	File Reference
	5.19.1	Detailed Description
	5.19.2	Function Documentation
		5.19.2.1 trdp_isValidSession()
		5.19.2.2 trdp_sessionQueue()
5.20	tlm_if.c	File Reference
	5.20.1	Detailed Description

xiv CONTENTS

	5.20.2	Function Documentation	8
		5.20.2.1 tlm_abortSession()	8
		5.20.2.2 tlm_addListener()	8
		5.20.2.3 tlm_confirm()	9
		5.20.2.4 tlm_delListener()	0
		5.20.2.5 tlm_getInterval()	0
		5.20.2.6 tlm_notify()	1
		5.20.2.7 tlm_process()	2
		5.20.2.8 tlm_readdListener()	2
		5.20.2.9 tlm_reply()	3
		5.20.2.10 tlm_replyQuery()	4
		5.20.2.11 tlm_request()	5
5.21	tlp_if.c	File Reference	6
	5.21.1	Detailed Description	8
	5.21.2	Function Documentation	8
		5.21.2.1 tlp_get()	9
		5.21.2.2 tlp_getInterval()	9
		5.21.2.3 tlp_getRedundant()	0
		5.21.2.4 tlp_processReceive()	0
		5.21.2.5 tlp_processSend()	1
		5.21.2.6 tlp_publish()	1
		5.21.2.7 tlp_put()	2
		5.21.2.8 tlp_putImmediate()	3
		5.21.2.9 tlp_republish()	3
		5.21.2.10 tlp_request()	4
		5.21.2.11 tlp_resubscribe()	5
		5.21.2.12 tlp_setRedundant()	6
		5.21.2.13 tlp_subscribe()	6
		5.21.2.14 tlp_unpublish()	7
		5.21.2.15 tlp_unsubscribe()	8

CONTENTS xv

5.22	trdp_dl	lmain.c File	e Refe	erence				 	 	 	 	 	 	208
	5.22.1	Detailed I	Descri	iption				 	 	 	 	 	 	208
5.23	trdp_if_	_light.h File	Refe	rence				 	 	 	 	 	 	209
	5.23.1	Detailed I	Descri	iption				 	 	 	 	 	 	213
	5.23.2	Function	Docur	nentatio	on			 	 	 	 	 	 	213
		5.23.2.1	tlc_cl	loseSes	sion()			 	 	 	 	 	 	213
		5.23.2.2	tlc_c	onfigSe	ssion()			 	 	 	 	 	 	213
		5.23.2.3	tlc_g	etETBT	ороСог	ınt()		 	 	 	 	 	 	214
		5.23.2.4	tlc_g	etInterv	al() .			 	 	 	 	 	 	214
		5.23.2.5	tlc_g	etJoinS	tatistics	s() .		 	 	 	 	 	 	215
		5.23.2.6	tlc_g	etOpTra	ıinTopo	Count	() .	 	 	 	 	 	 	216
		5.23.2.7	tlc_g	etOwnlp	oAddres	ss() .		 	 	 	 	 	 	216
		5.23.2.8	tlc_g	etPubSt	tatistics	· ()		 	 	 	 	 	 	216
		5.23.2.9	tlc_g	etRedS	tatistics	s() .		 	 	 	 	 	 	217
		5.23.2.10	tlc_g	etStatis	tics() .			 	 	 	 	 	 	217
		5.23.2.11	tlc_g	etSubsS	Statistic	s() .		 	 	 	 	 	 	218
		5.23.2.12	tlc_g	etVersic	on() .			 	 	 	 	 	 	218
		5.23.2.13	tlc_g	etVersic	onString	<b>)</b> () .		 	 	 	 	 	 	219
		5.23.2.14	· tlc_in	nit()				 	 	 	 	 	 	219
		5.23.2.15	tlc_o	penSes	sion()			 	 	 	 	 	 	219
		5.23.2.16	tlc_p	rocess()	)			 	 	 	 	 	 	220
		5.23.2.17	' tlc_re	einitSes	sion()			 	 	 	 	 	 	221
		5.23.2.18	tlc_re	esetStat	istics()			 	 	 	 	 	 	221
		5.23.2.19	tlc_se	etETBT	opoCou	ınt()		 	 	 	 	 	 	222
		5.23.2.20	tlc_se	etOpTra	inTopo	Count	() .	 	 	 	 	 	 	222
		5.23.2.21	tlc_te	erminate	∍()			 	 	 	 	 	 	222
		5.23.2.22	tlc_u	pdateSe	ession()	)		 	 	 	 	 	 	223
		5.23.2.23	tlm_a	abortSe	ssion()			 	 	 	 	 	 	223
		5.23.2.24	tlm_a	addListe	ener()			 	 	 	 	 	 	224
		5.23.2.25	tlm_c	confirm(	)			 	 	 	 	 	 	225

xvi CONTENTS

	5.23.2.26 tlm_delListener()
	5.23.2.27 tlm_getInterval()
	5.23.2.28 tlm_notify()
	5.23.2.29 tlm_process()
	5.23.2.30 tlm_readdListener()
	5.23.2.31 tlm_reply()
	5.23.2.32 tlm_replyQuery()
	5.23.2.33 tlm_request()
	5.23.2.34 tlp_get()
	5.23.2.35 tlp_getInterval()
	5.23.2.36 tlp_getRedundant()
	5.23.2.37 tlp_processReceive()
	5.23.2.38 tlp_processSend()
	5.23.2.39 tlp_publish()
	5.23.2.40 tlp_put()
	5.23.2.41 tlp_putImmediate()
	5.23.2.42 tlp_republish()
	5.23.2.43 tlp_request()
	5.23.2.44 tlp_resubscribe()
	5.23.2.45 tlp_setRedundant()
	5.23.2.46 tlp_subscribe()
	5.23.2.47 tlp_unpublish()
	5.23.2.48 tlp_unsubscribe()
5.24 trdp_m	ndcom.c File Reference
5.24.1	Detailed Description
5.24.2	Function Documentation
	5.24.2.1 trdp_mdCall()
	5.24.2.2 trdp_mdCheckListenSocks()
	5.24.2.3 trdp_mdCheckPending()
	5.24.2.4 trdp_mdCheckTimeouts()

CONTENTS xvii

	5.24.2.5	trdp_mdConfirm()	246
	5.24.2.6	trdp_mdFreeSession()	247
	5.24.2.7	trdp_mdGetTCPSocket()	247
	5.24.2.8	trdp_mdReply()	248
	5.24.2.9	trdp_mdSend()	248
5.25 trdp_	_mdcom.h F	ile Reference	249
5.25	.1 Detailed	Description	250
5.25	.2 Function	Documentation	251
	5.25.2.1	trdp_mdCall()	251
	5.25.2.2	trdp_mdCheckListenSocks()	252
	5.25.2.3	trdp_mdCheckPending()	252
	5.25.2.4	trdp_mdCheckTimeouts()	253
	5.25.2.5	trdp_mdConfirm()	253
	5.25.2.6	trdp_mdFreeSession()	254
	5.25.2.7	trdp_mdGetTCPSocket()	254
	5.25.2.8	trdp_mdReply()	254
	5.25.2.9	trdp_mdSend()	255
5.26 trdp_	_pdcom.c Fil	le Reference	255
5.26	.1 Detailed	Description	257
5.26	.2 Function	Documentation	257
	5.26.2.1	trdp_pdCheck()	257
	5.26.2.2	trdp_pdCheckListenSocks()	258
	5.26.2.3	trdp_pdCheckPending()	258
	5.26.2.4	trdp_pdDistribute()	259
	5.26.2.5	trdp_pdHandleTimeOuts()	259
	5.26.2.6	trdp_pdInit()	260
	5.26.2.7	trdp_pdPut()	261
	5.26.2.8	trdp_pdReceive()	261
	5.26.2.9	trdp_pdSend()	262
	5.26.2.10	0 trdp_pdSendElement()	262

xviii CONTENTS

	5.26.2.11 trdp_pdSendImmediate()	263
	5.26.2.12 trdp_pdSendQueued()	264
	5.26.2.13 trdp_pdUpdate()	265
5.27 trdp_pc	dcom.h File Reference	265
5.27.1	Detailed Description	267
5.27.2	Function Documentation	267
	5.27.2.1 trdp_pdCheck()	268
	5.27.2.2 trdp_pdCheckListenSocks()	268
	5.27.2.3 trdp_pdCheckPending()	268
	5.27.2.4 trdp_pdDistribute()	269
	5.27.2.5 trdp_pdHandleTimeOuts()	270
	5.27.2.6 trdp_pdInit()	270
	5.27.2.7 trdp_pdPut()	271
	5.27.2.8 trdp_pdReceive()	271
	5.27.2.9 trdp_pdSend()	272
	5.27.2.10 trdp_pdSendElement()	272
	5.27.2.11 trdp_pdSendImmediate()	273
	5.27.2.12 trdp_pdSendQueued()	274
	5.27.2.13 trdp_pdUpdate()	275
5.28 trdp_pc	dindex.c File Reference	275
5.28.1	Detailed Description	276
5.29 trdp_pc	dindex.h File Reference	277
5.29.1	Detailed Description	278
5.30 trdp_pr	ivate.h File Reference	279
5.30.1	Detailed Description	281
5.30.2	Macro Definition Documentation	282
	5.30.2.1 TRDP_MAX_PD_SOCKET_CNT	282
5.30.3	Enumeration Type Documentation	282
	5.30.3.1 TRDP_MD_ELE_ST_T	282
	5.30.3.2 TRDP_SOCK_TYPE_T	282

CONTENTS xix

5.31	trdp_se	erviceRegi	stry.h File Refe	rence .			 	 	 	 	. :	283
	5.31.1	Detailed	Description .				 	 	 	 	. :	286
	5.31.2	Macro De	efinition Docum	entation			 	 	 	 	. :	286
		5.31.2.1	SOA_SAME_	SERVICE	ID		 	 	 	 	. :	286
		5.31.2.2	SRM_SERVI	CE_READ	REQ_0	COMID	 	 	 	 	. :	286
		5.31.2.3	SRM_SRVIN	FO_NOTII	FY_CON	MID .	 	 	 	 	. :	287
5.32	trdp_st	ats.c File F	Reference				 	 	 	 		287
	5.32.1	Detailed	Description .				 	 	 	 		288
	5.32.2	Function	Documentation	ı			 	 	 	 		288
		5.32.2.1	tlc_getJoinSta	atistics()			 	 	 	 	. :	288
		5.32.2.2	tlc_getPubSta	atistics()			 	 	 	 	. :	289
		5.32.2.3	tlc_getRedSta	atistics()			 	 	 	 	. :	289
		5.32.2.4	tlc_getStatisti	cs()			 	 	 	 		290
		5.32.2.5	tlc_getSubsS	tatistics()			 	 	 	 		290
		5.32.2.6	tlc_resetStatis	stics()			 	 	 	 		291
		5.32.2.7	trdp_initStats	()			 	 	 	 		291
		5.32.2.8	trdp_pdPrepa	reStats()			 	 	 	 		292
		5.32.2.9	trdp_UpdateS	Stats()			 	 	 	 	. :	293
5.33	trdp_st	ats.h File I	Reference				 	 	 	 	. :	293
	5.33.1	Detailed	Description .				 	 	 	 	. :	294
	5.33.2	Function	Documentation	ı			 	 	 	 	. :	294
		5.33.2.1	trdp_initStats	()			 	 	 	 	. :	294
		5.33.2.2	trdp_pdPrepa	reStats()			 	 	 	 	. :	295
5.34	trdp_ts	n_def.h Fil	e Reference .				 	 	 	 	. :	296
	5.34.1	Detailed	Description .				 	 	 	 	. :	296
	5.34.2	Macro De	efinition Docum	entation			 	 	 	 	. :	297
		5.34.2.1	TRDP_MIN_F	PD2_HEA	DER_SI	ZE	 	 	 	 	. :	297
		5.34.2.2	TRDP_MSG_	_TSN_PD			 	 	 	 	. :	297
		5.34.2.3	TRDP_PD_D	EFAULT_	QOS		 	 	 	 	. :	297
5.35	trdp_ty	pes.h File	Reference				 	 	 	 	. :	297

CONTENTS

	5.35.1	Detailed Description
	5.35.2	Macro Definition Documentation
		5.35.2.1 TRDP_FLAGS_DEFAULT
	5.35.3	Typedef Documentation
		5.35.3.1 TRDP_IP_ADDR_T
		5.35.3.2 TRDP_MARSHALL_T
		5.35.3.3 TRDP_MD_CALLBACK_T
		5.35.3.4 TRDP_PD_CALLBACK_T
		5.35.3.5 TRDP_PRINT_DBG_T
		5.35.3.6 TRDP_TIME_T
		5.35.3.7 TRDP_UNMARSHALL_T
	5.35.4	Enumeration Type Documentation
		5.35.4.1 TRDP_DATA_TYPE_T 309
		5.35.4.2 TRDP_ERR_T
		5.35.4.3 TRDP_RED_STATE_T 30°
		5.35.4.4 TRDP_REPLY_STATUS_T
		5.35.4.5 TRDP_TO_BEHAVIOR_T
5.36	trdp_ut	ils.c File Reference
	5.36.1	Detailed Description
	5.36.2	Function Documentation
		5.36.2.1 printSocketUsage()
		5.36.2.2 trdp_checkSequenceCounter()
		5.36.2.3 trdp_findMCjoins()
		5.36.2.4 trdp_findSubAddr()
		5.36.2.5 trdp_getSeqCnt()
		5.36.2.6 trdp_initSockets()
		5.36.2.7 trdp_isAddressed()
		5.36.2.8 trdp_isInIPrange()
		5.36.2.9 trdp_packetSizeMD()
		5.36.2.10 trdp_packetSizePD()

CONTENTS xxi

		5.36.2.11 trdp_queueAppLast()
		5.36.2.12 trdp_queueDelElement()
		5.36.2.13 trdp_queueFindComId()
		5.36.2.14 trdp_queueFindExistingSub()
		5.36.2.15 trdp_queueFindPubAddr()
		5.36.2.16 trdp_queueFindSubAddr()
		5.36.2.17 trdp_queueInsFirst()
		5.36.2.18 trdp_releaseSocket()
		5.36.2.19 trdp_requestSocket()
		5.36.2.20 trdp_resetSequenceCounter()
		5.36.2.21 trdp_SockAddJoin()
		5.36.2.22 trdp_SockDelJoin()
		5.36.2.23 trdp_SockIsJoined()
		5.36.2.24 trdp_validTopoCounters()
5.37	trdp_ut	ils.h File Reference
	5.37.1	Detailed Description
	5.37.2	Function Documentation
		5.37.2.1 printSocketUsage()
		5.37.2.2 trdp_checkSequenceCounter()
		5.37.2.3 trdp_findMCjoins()
		5.37.2.4 trdp_findSubAddr()
		5.37.2.5 trdp_getSeqCnt()
		5.37.2.6 trdp_initSockets()
		5.37.2.7 trdp_isAddressed()
		5.37.2.8 trdp_isInIPrange()
		5.37.2.9 trdp_packetSizeMD()
		5.37.2.10 trdp_packetSizePD()
		5.37.2.11 trdp_queueAppLast()
		5.37.2.12 trdp_queueDelElement()
		5.37.2.13 trdp_queueFindComId()

xxii CONTENTS

		5.37.2.14 trdp_queueFindExistingSub()	28
		5.37.2.15 trdp_queueFindPubAddr()	29
		5.37.2.16 trdp_queueFindSubAddr()	29
		5.37.2.17 trdp_queueInsFirst()	30
		5.37.2.18 trdp_releaseSocket()	30
		5.37.2.19 trdp_requestSocket()	30
		5.37.2.20 trdp_resetSequenceCounter()	31
		5.37.2.21 trdp_SockAddJoin()	32
		5.37.2.22 trdp_SockDelJoin()	32
		5.37.2.23 trdp_SockIsJoined()	32
		5.37.2.24 trdp_validTopoCounters()	33
5.38	trdp_xr	ml.c File Reference	33
	5.38.1	Detailed Description	35
	5.38.2	Function Documentation	35
		5.38.2.1 trdp_XMLClose()	35
		5.38.2.2 trdp_XMLCountStartTag()	35
		5.38.2.3 trdp_XMLEnter()	36
		5.38.2.4 trdp_XMLGetAttribute()	36
		5.38.2.5 trdp_XMLLeave()	37
		5.38.2.6 trdp_XMLMemOpen()	37
		5.38.2.7 trdp_XMLOpen()	38
		5.38.2.8 trdp_XMLRewind()	38
		5.38.2.9 trdp_XMLSeekStartTag()	38
		5.38.2.10 trdp_XMLSeekStartTagAny()	39
5.39	trdp_xr	ml.h File Reference	39
	5.39.1	Detailed Description	11
	5.39.2	Function Documentation	11
		5.39.2.1 trdp_XMLClose()	<b>!</b> 1
		5.39.2.2 trdp_XMLCountStartTag()	12
		5.39.2.3 trdp_XMLEnter()	12

CONTENTS xxiii

		5.39.2.4	trdp_XMI	_GetAttr	ibute()		 	 	 	 	 		342
		5.39.2.5	trdp_XMI	_Leave()	)		 	 	 	 	 		343
		5.39.2.6	trdp_XMI	_MemO <sub>l</sub>	pen()		 	 	 	 	 		343
		5.39.2.7	trdp_XMI	_Open()			 	 	 	 	 		344
		5.39.2.8	trdp_XMI	_Rewind	l()		 	 	 	 	 		344
		5.39.2.9	trdp_XMI	_SeekSt	artTag	. ()	 	 	 	 	 		345
		5.39.2.10	trdp_XMI	_SeekSt	:artTag	Any()	 	 	 	 	 		345
5.40	vos_me	em.c File Ro	eference				 	 	 	 	 		345
	5.40.1	Detailed D	Description	n			 	 	 	 	 		347
	5.40.2	Function D	Document	ation .			 	 	 	 	 		347
		5.40.2.1	vos_bsea	arch() .			 	 	 	 	 		347
		5.40.2.2	vos_men	nAlloc()			 	 	 	 	 		348
		5.40.2.3	vos_men	nCount()	)		 	 	 	 	 		348
		5.40.2.4	vos_men	nDelete(	)		 	 	 	 	 		349
		5.40.2.5	vos_men	nFree()			 	 	 	 	 		349
		5.40.2.6	vos_men	nInit() .			 	 	 	 	 		349
		5.40.2.7	vos_qsor	t()			 	 	 	 	 		350
		5.40.2.8	vos_que	JeCreate	∍() .		 	 	 	 	 		350
		5.40.2.9	vos_que	ueDestro	oy() .		 	 	 	 	 		351
		5.40.2.10	vos_que	JeRecei <sup>®</sup>	ve() .		 	 	 	 	 		351
		5.40.2.11	vos_que	ueSend(	)		 	 	 	 	 		352
		5.40.2.12	vos_strno	cat()			 	 	 	 	 		352
		5.40.2.13	vos_strno	cpy()			 	 	 	 	 		354
		5.40.2.14	vos_strni	cmp() .			 	 	 	 	 		354
5.41	vos_me	em.h File R	eference				 	 	 	 	 		355
	5.41.1	Detailed D	Description	n			 	 	 	 	 		357
	5.41.2	Macro Def	finition Do	cument	ation		 	 	 	 	 		357
		5.41.2.1	VOS_ME	:M_BLO	CKSIZ	ZES.	 	 	 	 	 		357
		5.41.2.2	VOS_ME	:M_PRE	ALLO	CATE	 	 	 	 	 		357
	5.41.3	Function D	Document	ation .			 	 	 	 	 		357

xxiv CONTENTS

	5.41.3.1	vos_bsearch()		 	 	. 357
	5.41.3.2	vos_memAlloc()		 	 	. 358
	5.41.3.3	vos_memCount()		 	 	. 358
	5.41.3.4	vos_memDelete()		 	 	. 359
	5.41.3.5	vos_memFree()		 	 	. 359
	5.41.3.6	vos_memInit()		 	 	. 360
	5.41.3.7	vos_qsort()		 	 	. 360
	5.41.3.8	vos_queueCreate()		 	 	. 361
	5.41.3.9	vos_queueDestroy()		 	 	. 362
	5.41.3.10	vos_queueReceive()		 	 	. 362
	5.41.3.1	vos_queueSend()		 	 	. 363
	5.41.3.12	vos_strncat()		 	 	. 363
	5.41.3.10	vos_strncpy()		 	 	. 363
	5.41.3.14	vos_strnicmp()		 	 	. 364
5.42 vos_	_shared_mer	n.h File Reference		 	 	. 364
5.42	2.1 Detailed	Description		 	 	. 365
5.42	2.2 Function	Documentation		 	 	. 366
	5.42.2.1	vos_sharedClose()		 	 	. 366
	5.42.2.2	vos_sharedOpen()		 	 	. 366
5.43 vos_	_sock.h File f	deference		 	 	. 367
5.43	3.1 Detailed	Description		 	 	. 369
5.43	3.2 Macro D	efinition Documentation		 	 	. 370
	5.43.2.1	VOS_MAX_SOCKET_	_CNT	 	 	. 370
	5.43.2.2	VOS_TTL_MULTICAS	ST	 	 	. 370
5.43	3.3 Function	Documentation		 	 	. 370
	5.43.3.1	vos_determineBindAd	dr()	 	 	. 370
	5.43.3.2	vos_dottedIP()		 	 	. 371
	5.43.3.3	vos_getInterfaces().		 	 	. 371
	5.43.3.4	vos_htonl()		 	 	. 372
	5.43.3.5	vos_htonll()		 	 	. 372

CONTENTS xxv

		5.43.3.6 vos_htons()
		5.43.3.7 vos_ipDotted()
		5.43.3.8 vos_isMulticast()
		5.43.3.9 vos_netIfUp()
		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()
		5.43.3.24 vos_sockOpenUDP()
		5.43.3.25 vos_sockReceiveTCP()
		5.43.3.26 vos_sockReceiveUDP()
		5.43.3.27 vos_sockSendTCP()
		5.43.3.28 vos_sockSendUDP()
		5.43.3.29 vos_sockSetMulticastIf()
		5.43.3.30 vos_sockSetOptions()
		5.43.3.31 vos_sockTerm()
5.44	vos_thr	ead.h File Reference
	5.44.1	Detailed Description
	5.44.2	Function Documentation
		5.44.2.1 vos_addTime()

xxvi CONTENTS

		5.44.2.2	vos_clear	Time() .		 	 	 	 	 		387
		5.44.2.3	vos_cmp	Γime() .		 	 	 	 	 		387
		5.44.2.4	vos_divTi	me()		 	 	 	 	 		388
		5.44.2.5	vos_getR	ealTime()	)	 	 	 	 	 		388
		5.44.2.6	vos_getT	me()		 	 	 	 	 		388
		5.44.2.7	vos_getT	meStamp	o()	 	 	 	 	 		389
		5.44.2.8	vos_getU	uid()		 	 	 	 	 		389
		5.44.2.9	vos_mulT	ime()		 	 	 	 	 		389
		5.44.2.10	vos_mute	xCreate()	)	 	 	 	 	 		390
		5.44.2.11	vos_mute	xDelete()		 	 	 	 	 		390
		5.44.2.12	vos_mute	xLock() .		 	 	 	 	 		390
		5.44.2.13	vos_mute	xTryLock	()	 	 	 	 	 		391
		5.44.2.14	vos_mute	xUnlock()	)	 	 	 	 	 		391
		5.44.2.15	vos_sema	aCreate()		 	 	 	 	 		392
		5.44.2.16	vos_sema	aDelete()		 	 	 	 	 		392
		5.44.2.17	vos_sema	aGive() .		 	 	 	 	 		392
		5.44.2.18	vos_sema	aTake() .		 	 	 	 	 		393
		5.44.2.19	vos_subT	ime()		 	 	 	 	 		393
		5.44.2.20	vos_threa	ıdCreate(	)	 	 	 	 	 		393
		5.44.2.21	vos_threa	ıdCreateS	Sync()	 	 	 	 	 		394
		5.44.2.22	vos_threa	ıdDelay()		 	 	 	 	 		395
		5.44.2.23	vos_threa	ıdlnit() .		 	 	 	 	 		395
		5.44.2.24	vos_threa	ıdlsActive	<del>)</del> ()	 	 	 	 	 		396
		5.44.2.25	vos_threa	ıdSelf() .		 	 	 	 	 		396
		5.44.2.26	vos_threa	ıdTerm()		 	 	 	 	 		396
		5.44.2.27	vos_threa	ıdTermina	ate() .	 	 	 	 	 		397
5.45	vos_typ	oes.h File R	leference			 	 	 	 	 		397
	5.45.1	Detailed D	escription	١		 	 	 	 	 		399
	5.45.2	Typedef D	ocumenta	ıtion		 	 	 	 	 		399
		5.45.2.1	VOS_PR	INT_DBG	<u>_</u> T .	 	 	 	 	 		399

CONTENTS xxvii

	5.45.2.2	VOS_TIMEVAL_T	 399
5.45.3	Enumera	tion Type Documentation	 400
	5.45.3.1	VOS_ERR_T	 400
	5.45.3.2	VOS_LOG_T	 400
5.46 vos_u	tils.c File R	eference	 401
5.46.1	Detailed	Description	 402
5.46.2	Function	Documentation	 402
	5.46.2.1	vos_crc32()	 402
	5.46.2.2	vos_getErrorString()	 403
	5.46.2.3	vos_getVersion()	 403
	5.46.2.4	vos_getVersionString()	 403
	5.46.2.5	vos_hostIsBigEndian()	 404
	5.46.2.6	vos_init()	 404
	5.46.2.7	vos_sc32()	 404
	5.46.2.8	vos_terminate()	 405
5.47 vos_u	tils.h File R	deference	 405
5.47.1	Detailed	Description	 407
5.47.2	Macro De	efinition Documentation	 407
	5.47.2.1	INITFCS	 407
	5.47.2.2	VOS_MAX_ERR_STR_SIZE	 407
	5.47.2.3	VOS_MAX_FRMT_SIZE	 407
	5.47.2.4	VOS_MAX_PRNT_STR_SIZE	 408
5.47.3	Function	Documentation	 408
	5.47.3.1	vos_crc32()	 408
	5.47.3.2	vos_getErrorString()	 409
	5.47.3.3	vos_getVersion()	 409
	5.47.3.4	vos_getVersionString()	 409
	5.47.3.5	vos_hostIsBigEndian()	 410
	5.47.3.6	vos_init()	 410
	5.47.3.7	vos_sc32()	 411
	5.47.3.8	vos_terminate()	 411
Index			413

### **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:

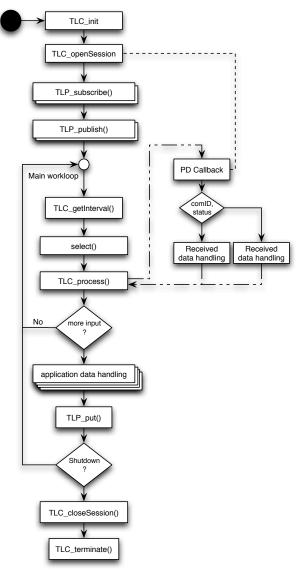


Figure 1.1 Sample client workflow (Single Thread)

1.3 Use Cases 3

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

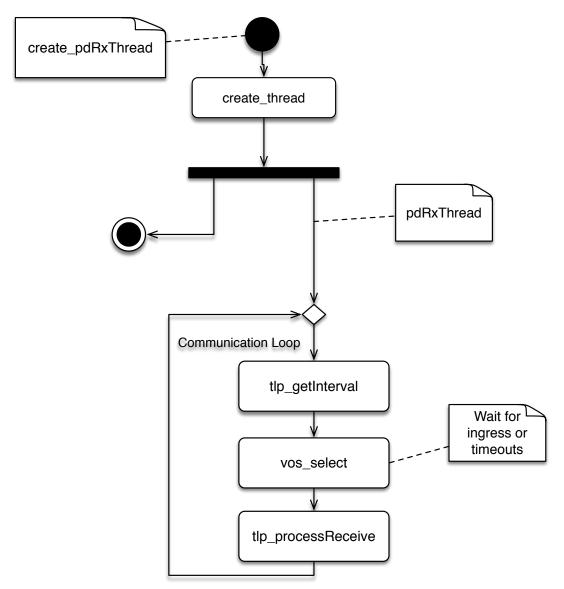


Figure 1.2 Multi-threaded processing of PD Reception

The transmit thread should be a cyclic thread – cycle times down to 1ms are supported:

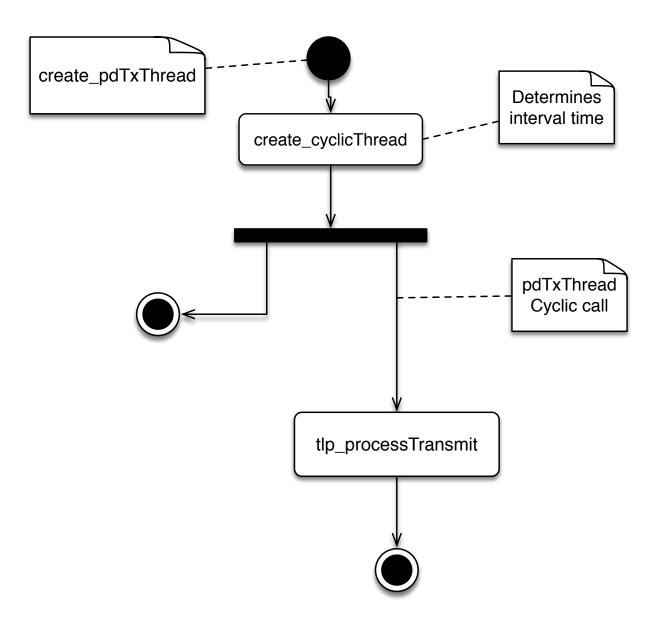


Figure 1.3 Multi-threaded processing of PD Transmit

1.4 Conventions of the API 5

If Message Data support is needed (MD\_SUPPORT=1):

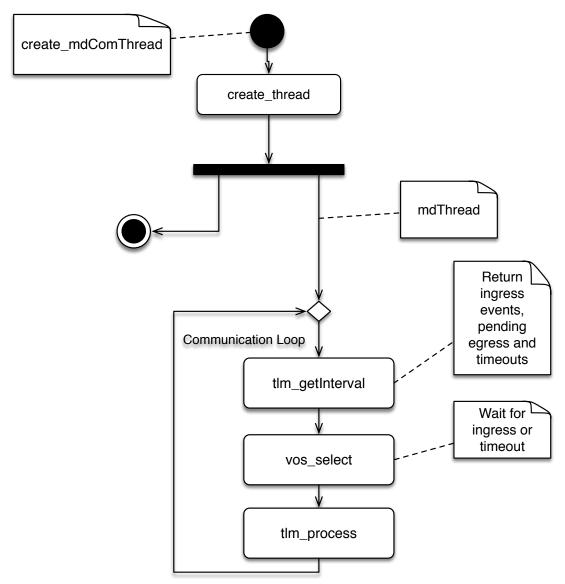


Figure 1.4 Multi-threaded processing of MD

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,
	36
serviceRegistryEntry	0-
Preliminary definition of a service registry entry	37
srv_info_req  Preliminary definition of a service info request	38
TAU MARSHALL INFO T	SC
Marshalling info, used to and from wire	38
TCN URI	JC
TCN-DNS simplified header structures	39
TRDP CLTR CST INFO T	00
Closed train consists information	40
TRDP COM PARAM T	
Quality/type of service, time to live , no	40
TRDP_COMID_DSID_MAP_T	
Comld - data set mapping element definition	41
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	46

Data Structure Index

TRDP_DNS_REQUEST	
TCN-DNS Request telegram TCN_DNS_REQ_DS	47
TRDP_ETB_INFO_T	
Types for train configuration information	48
TRDP_FUNCTION_INFO_T	
Function/device information structure	49
TRDP_HANDLE	
Hidden handle definition, used as unique addressing item	50
Marshaling/unmarshalling configuration	51
TRDP MD CONFIG T	٠.
Default MD configuration	52
TRDP_MD_INFO_T	02
Message data info from received telegram; allows the application to generate responses	53
TRDP_MEM_CONFIG_T	
Enumeration type for memory pre-fragmentation, reuse of VOS definition	54
TRDP_PD_CONFIG_T	
Default PD configuration	55
TRDP_PD_INFO_T	
Process data info from received telegram; allows the application to generate responses	56
TRDP_PROCESS_CONFIG_T	
Various flags/general TRDP options for library initialization	57
TRDP_PROP_T	
Application defined properties	57
TRDP_SDT_PAR_T	
Types to read out the XML configuration	58
TRDP_SEQ_CNT_ENTRY_T	
Tuples of last received sequence counter per comld	59
TRDP_SESSION	
Session/application variables store	59
TRDP_SOCKET_TCP	
TCP parameters	61
TRDP_SOCKETS	
Socket item	61
TRDP_VEHICLE_INFO_T	
Vehicle information structure	63
TRDP_XML_DOC_HANDLE_T	
Parsed XML document handle	64
VOS_SOCK_OPT_T	
Common socket options	64
VOS_VERSION_T	
Version information	65

# **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	67
tau_cstinfo.c	
Functions for consist information access	75
tau_ctrl.c	
	77
tau_ctrl.h	
	31
tau_ctrl_types.h  TRDP utility interface definitions	36
tau_dnr.c	JC
	39
tau_dnr.h	
	94
tau_dnr_types.h	
TRDP utility interface definitions	)1
tau_marshall.c	
Marshalling functions for TRDP	)3
tau_marshall.h	
TRDP utility interface definitions	ງເ
tau_so_if.c  Functions for service oriented functions of the TTDB	1 (
tau_so_if.h	1 2
Access to the Service Registry	22
tau tti.c	-
Functions for train topology information access	28
tau_tti.h	
TRDP utility interface definitions	39
tau_tti_types.h	
TRDP utility interface definitions	53
tau_xml.c	
Functions for XML file parsing	56
tau_xml.h	٠,
TRDP utility interface definitions	)_
tic_ii.c	_,

10 File Index

tlc_if.h	
Typedefs for TRDP communication	183
Functions for Message Data Communication	186
tlp_if.c	400
Functions for Process Data Communication	196
Windows DLL main function	208
trdp_if_light.h  TRDP Light interface functions (API)	209
trdp_mdcom.c  Functions for MD communication	242
trdp_mdcom.h  Functions for MD communication	249
trdp_pdcom.c	
Functions for PD communication	255
Functions for PD communication	265
trdp_pdindex.c Functions for indexed PD communication	275
trdp_pdindex.h  Functions for indexed PD communication	277
trdp_private.h	270
Typedefs for TRDP communication	279
Additional definitions for IEC 61375-2-3 (Service Discovery) The definitions herein are preliminary and may change with the next major release of the IEC 61375-2-3 standard	283
trdp_stats.c  Statistics functions for TRDP communication	287
trdp_stats.h	
Statistics for TRDP communication	293
Additional definitions for TSN	296
trdp_types.h         Typedefs for TRDP communication	297
trdp_utils.c  Helper functions for TRDP communication	308
trdp_utils.h  Common utilities for TRDP communication	320
trdp_xml.c	
Simple XML parser	333
Simple XML parser	339
Memory functions	345
Vos_mem.h  Memory and queue functions for OS abstraction	355
vos_shared_mem.h  Shared Memory functions for OS abstraction	364
vos_sock.h	
Typedefs for OS abstraction	367
Threading functions for OS abstraction	384
Typedefs for OS abstraction	397
vos_utils.c  Common functions for VOS	401

3.1 File List

vos_utils.h															
Typedefs for OS abstraction	 												 		405

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:

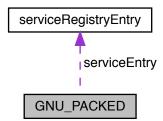
• tau\_dnr.c

# 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\_REGISTRY\_ENTRY 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 Comld'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_REGISTRY_ENTRY 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:

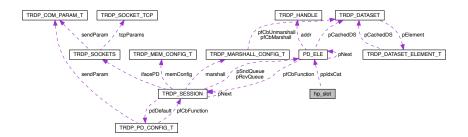
- · tau\_ctrl\_types.h
- tau\_tti\_types.h
- trdp\_serviceRegistry.h
- trdp\_types.h
- · trdp\_private.h

# 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])

#### 4.3.1 Detailed Description

Low cycle-time slots.

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

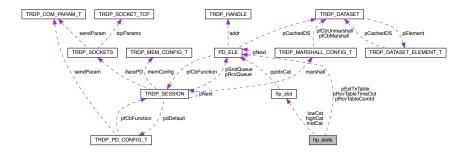
· trdp\_pdindex.h

## 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.

UINT8 noOfExtTxEntries

number of 'special' PDs to be handled

• PD\_ELE\_T \*\* pExtTxTable

Pointer to array of PDs to be handled.

#### 4.4.1 Detailed Description

entry for the application session

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

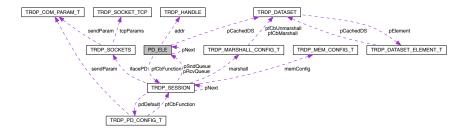
· trdp\_pdindex.h

## 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
- UINT24 srvld
  - service identifier as defined in X
- UINT8 srvInst
  - service instance as defined in X
- 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 reserved02
  - reserved for future use (= 0)
- 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 serviceRegistryEntry Struct Reference

Preliminary definition of a service registry entry.

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

#### **Data Fields**

TRDP\_SHORT\_VERSION\_T version

1.0 service version

· BITSET8 flags

0x01 | 0x02 == Safe Service

· UINT8 instanceId

8 Bit relevant

UINT32 serviceTypeId

lower 24 Bit relevant

• CHAR8 serviceName [32]

name of the service

• CHAR8 serviceURI [80]

destination URI for services

• TRDP\_IP\_ADDR\_T destMCIP

destination multicast for services

CHAR8 hostname [80]

device name - FQN (optional)

TRDP\_IP\_ADDR\_T machineIP

current IP address of service host

• TIMEDATE64 timeToLive

when to check for life sign

TIMEDATE64 lastUpdated

last time seen (optional)

TIMEDATE64 timeSlot

timeslot for TSN (optional)

#### 4.7.1 Detailed Description

Preliminary definition of a service registry entry.

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

· trdp serviceRegistry.h

# 4.8 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

#### 4.8.1 Detailed Description

Preliminary definition of a service info request.

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

• trdp\_serviceRegistry.h

## 4.9 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

#### 4.9.1 Detailed Description

Marshalling info, used to and from wire.

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

• tau\_marshall.c

# 4.10 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

#### 4.10.1 Detailed Description

TCN-DNS simplified header structures.

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

tau\_dnr\_types.h

# 4.11 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)

#### 4.11.1 Detailed Description

Closed train consists information.

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

• tau\_tti\_types.h

# 4.12 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.

#### 4.12.1 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

# 4.13 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

#### 4.13.1 Detailed Description

Comld - data set mapping element definition.

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

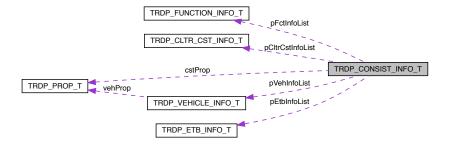
• trdp\_types.h

# 4.14 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: 'FFFFFFFH'

#### 4.14.1 Detailed Description

consist information structure

## 4.14.2 Field Documentation

```
4.14.2.1 cstld
```

```
TRDP_NET_LABEL_T TRDP_CONSIST_INFO_T::cstId
```

application defined consist identifier, e.g.

**UIC** identifier

#### 4.14.2.2 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:

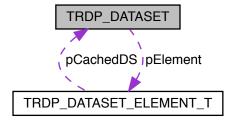
• tau\_tti\_types.h

# 4.15 TRDP\_DATASET Struct Reference

Dataset definition.

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

Collaboration diagram for TRDP\_DATASET:



• 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.

## 4.15.1 Detailed Description

Dataset definition.

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

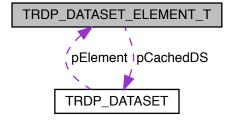
• trdp\_types.h

# 4.16 TRDP\_DATASET\_ELEMENT\_T Struct Reference

Dataset element definition.

#include <trdp\_types.h>

Collaboration diagram for TRDP\_DATASET\_ELEMENT\_T:



· 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.

## 4.16.1 Detailed Description

Dataset element definition.

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

• trdp\_types.h

# 4.17 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.

## 4.17.1 Detailed Description

Control for debug output device/file on application level.

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

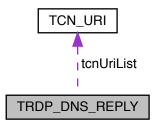
• tau\_xml.h

## 4.18 TRDP\_DNS\_REPLY Struct Reference

TCN-DNS Reply telegram TCN DNS REP DS.

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

Collaboration diagram for TRDP\_DNS\_REPLY:



## **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)

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.

## 4.18.1 Detailed Description

TCN-DNS Reply telegram TCN\_DNS\_REP\_DS.

## 4.18.2 Field Documentation

#### 4.18.2.1 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:

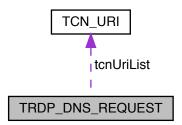
· tau\_dnr\_types.h

## 4.19 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 etblo

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.

## 4.19.1 Detailed Description

TCN-DNS Request telegram TCN\_DNS\_REQ\_DS.

#### 4.19.2 Field Documentation

#### 4.19.2.1 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

# 4.20 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)

## 4.20.1 Detailed Description

Types for train configuration information.

ETB information

## 4.20.2 Field Documentation

#### 4.20.2.1 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

# 4.21 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)

## 4.21.1 Detailed Description

function/device information structure

## 4.21.2 Field Documentation

#### 4.21.2.1 cnld

```
UINT8 TRDP_FUNCTION_INFO_T::cnId
```

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

Value range: 0..31

#### 4.21.2.2 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

#### 4.21.2.3 etbld

```
UINT8 TRDP_FUNCTION_INFO_T::etbId
```

number of connected train backbone.

Value range: 0..3

## 4.21.2.4 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

# 4.22 TRDP\_HANDLE Struct Reference

Hidden handle definition, used as unique addressing item.

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

UINT32 comId

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 destlpAddr

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

## 4.22.1 Detailed Description

Hidden handle definition, used as unique addressing item.

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

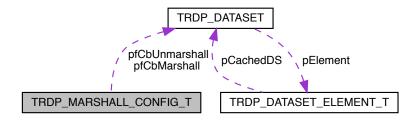
· trdp\_private.h

# 4.23 TRDP\_MARSHALL\_CONFIG\_T Struct Reference

Marshaling/unmarshalling configuration.

#include <trdp\_types.h>

Collaboration diagram for TRDP\_MARSHALL\_CONFIG\_T:



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:

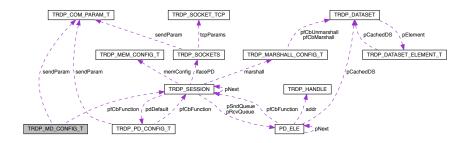
• trdp\_types.h

# 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 destIpAddr

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>

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:

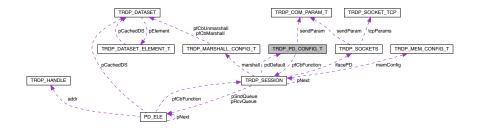
• trdp\_types.h

# 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>
```

#### **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 comId

ComID.

UINT32 etbTopoCnt

received ETB topocount

UINT32 opTrnTopoCnt

received operational train directory topocount

UINT32 replyComId

ComID for reply (request only)

· TRDP IP ADDR T replylpAddr

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>
```

#### **Data Fields**

• 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 Toptions

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>
```

• 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>
```

## **Data Fields**

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:

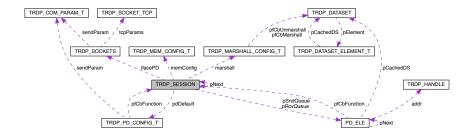
• trdp\_private.h

# 4.33 TRDP\_SESSION Struct Reference

Session/application variables store.

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

Collaboration diagram for TRDP\_SESSION:



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>

## **Data Fields**

· 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:

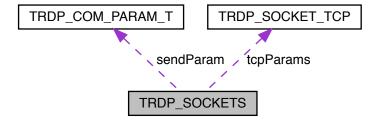
• trdp\_private.h

## 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

Nο

• 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:

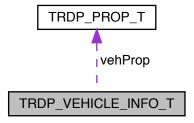
trdp\_private.h

# 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

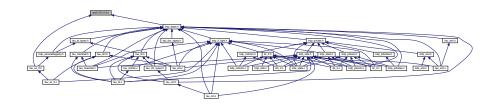
# **Chapter 5**

# **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.

68 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"

     TRAIN_DIRECTORY.
```

• #define TTDB\_STAT\_CST\_REQ\_COMID 104u

TTDB manager telegram MD: Get the static consist information.

70 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

[us]

#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

72 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

74 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.

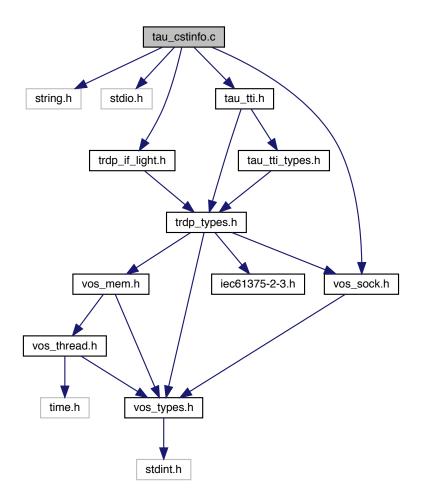
MD request

# 5.2 tau\_cstinfo.c File Reference

Functions for consist information access.

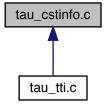
```
#include <string.h>
#include <stdio.h>
#include "trdp_if_light.h"
#include "tau_tti.h"
#include "vos_sock.h"
```

Include dependency graph for tau\_cstinfo.c:



76 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:



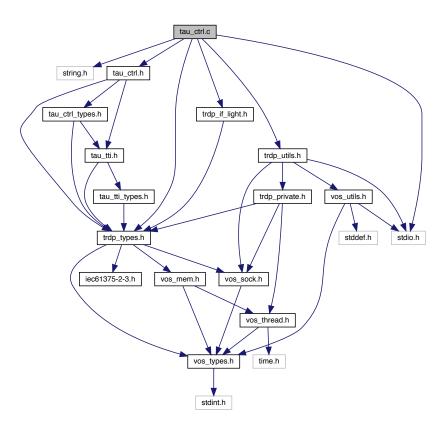
# 5.3 tau\_ctrl.c File Reference

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"
```

78 File Documentation

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()

```
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.

## **Parameters**

	in	appHandle	Application handle
	in,out	pEcspStat	Pointer to the ECSP status structure
Ī	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()

```
EXT_DECL TRDP_ERR_T tau_initEcspCtrl (

TRDP_APP_SESSION_T appHandle,

TRDP_IP_ADDR_T ecspIpAddr )
```

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		user reference returned with reply
in pfCbFunction Pointer to callback function, NULL		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

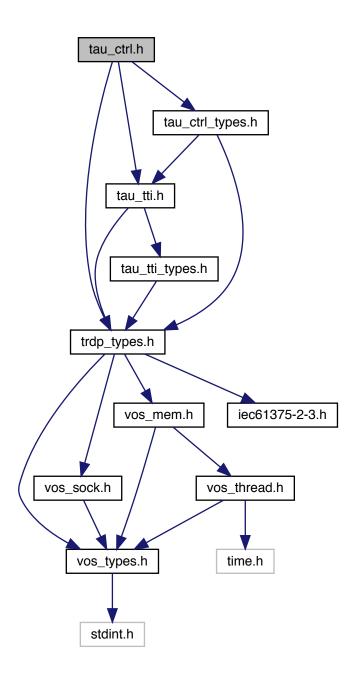
TRDP_NO_ERR	no error
TRDP_NOINIT_ERR	module not initialised
TRDP_UNKNOWN_ERR	undefined error

# 5.4 tau\_ctrl.h File Reference

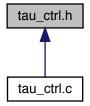
TRDP utility interface definitions.

```
#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()

```
EXT_DECL TRDP_ERR_T tau_setEcspCtrl (

TRDP_APP_SESSION_T appHandle,

TRDP_ECSP_CTRL_T * pEcspCtrl )
```

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**

### **Return values**

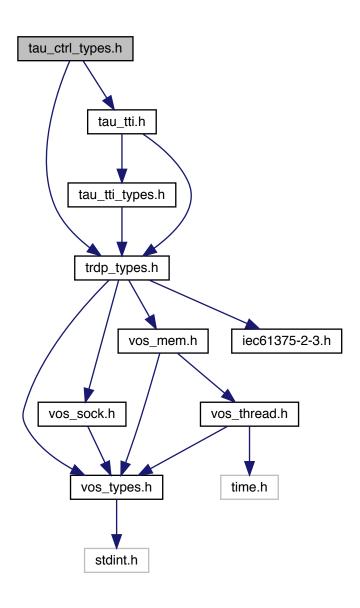
TRDP_NO_ERR	no error
TRDP_NOINIT_ERR	module not initialised
TRDP_UNKNOWN_ERR	undefined error

# 5.5 tau\_ctrl\_types.h File Reference

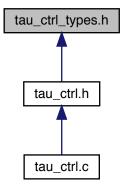
TRDP utility interface definitions.

```
#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

• struct GNU\_PACKED

Types for ETB control.

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

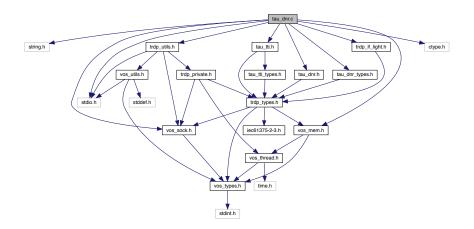
This Source Code Form is subject to the terms of the Mozilla Public License, v. 2.0. If a copy of the MPL was not distributed with this file, You can obtain one at http://mozilla.org/MPL/2.0/. Copyright Bombardier Transportation Inc. or its subsidiaries and others, 2013. All rights reserved.

## 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	appHandle	Handle returned by tlc_openSession()
	app. rairais	

### 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	ddr DNS/ECSP IP address.	
in	dnsPort	DNS port number.	
in	pHostsFileName	FileName 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

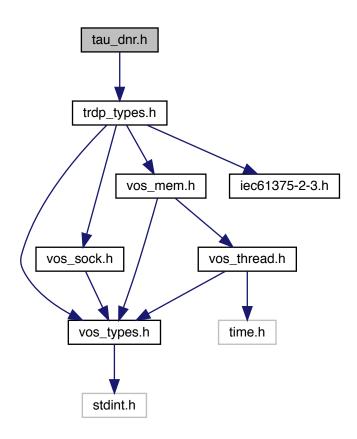
TRDP_NO_ERR	no error
TRDP_PARAM_ERR	Parameter error
TRDP_UNRESOLVED_ERR	Could not resolve error

## Return values

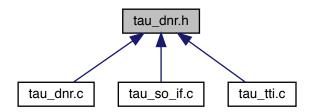
# 5.7 tau\_dnr.h File Reference

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()

```
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()	I
--	----	-----------	--------------------------------------	---

### 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**

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

### **Return values**

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	Parameter error
TRDP_UNRESOLVED_ERR	Could not resolve error

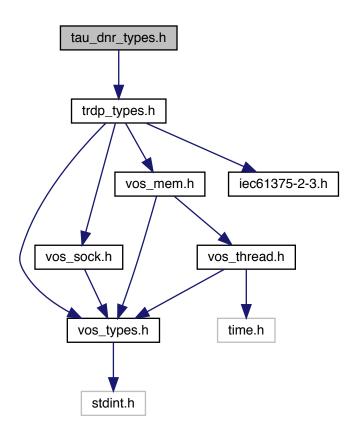
## Return values

TRDP TOPO ERR	Cache/DB entry is invalid

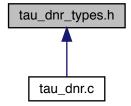
# 5.8 tau\_dnr\_types.h File Reference

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)

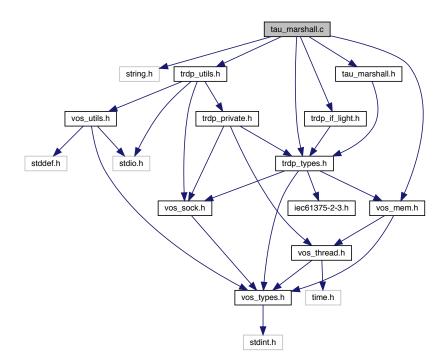
## Remarks

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	pComIdDsIdMap	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

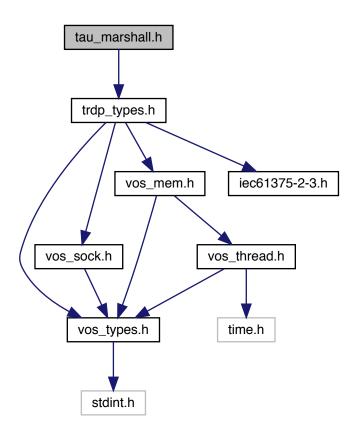
### Return values

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

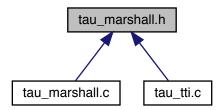
# 5.10 tau\_marshall.h File Reference

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**

• EXT\_DECL TRDP\_ERR\_T tau\_initMarshall (void \*\*ppRefCon, UINT32 numComId, TRDP\_COMID\_DSID → MAP\_T \*pComIdDsIdMap, UINT32 numDataSet, TRDP\_DATASET\_T \*pDataset[])

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 dsId, 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	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
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	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_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	efCon 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

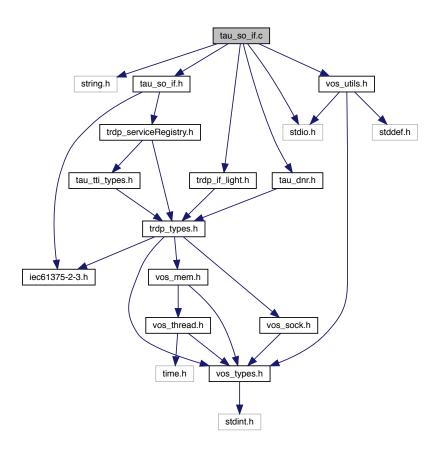
TRDP_INIT_ERR	marshalling not initialised
TRDP_NO_ERR	no error
TRDP_MEM_ERR	provided buffer to small
TRDP_PARAM_ERR	Parameter error
TRDP_STATE_ERR	Too deep recursion
TRDP_COMID_ERR	comid not existing
TRDP_MARSHALLING_ERR	dataset/source size mismatch

# 5.11 tau\_so\_if.c File Reference

Functions for service oriented functions of the TTDB.

```
#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\_addServices (TRDP\_APP\_SESSION\_T appHandle, UINT16 noOfServices, SRM\_SERVICE\_ARRAY\_T \*pServicesToAdd, BOOL8 waitForCompletion)

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

• EXT\_DECL TRDP\_ERR\_T tau\_delServices (TRDP\_APP\_SESSION\_T appHandle, UINT16 noOfServices, SRM\_SERVICE\_ARRAY\_T \*pServicesToRemove, BOOL8 waitForCompletion)

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

• EXT\_DECL TRDP\_ERR\_T tau\_updServices (TRDP\_APP\_SESSION\_T appHandle, UINT16 noOfServices, SRM\_SERVICE\_ARRAY\_T \*pServicesToUpdate, BOOL8 waitForCompletion)

Register an update a service.

• EXT\_DECL TRDP\_ERR\_T tau\_getServiceList (TRDP\_APP\_SESSION\_T appHandle, SRM\_SERVICE\_A⇔ RRAY T \*\*ppServicesListBuffer)

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

• EXT DECL void tau freeServiceList (SRM SERVICE ARRAY T \*pServicesListBuffer)

Release the memory of a list received by tau\_getServiceList.

# 5.11.1 Detailed Description

Functions for service oriented functions of the TTDB.

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\_addServices()

```
EXT_DECL TRDP_ERR_T tau_addServices (

TRDP_APP_SESSION_T appHandle,

UINT16 noOfServices,

SRM_SERVICE_ARRAY_T * pServicesToAdd,

BOOL8 waitForCompletion )
```

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	noOfServices	No of services in the array
in,out	pServicesToAdd	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

## Return values

TRDP_SEMA_ERR	Semaphore could not be aquired
---------------	--------------------------------

# 5.11.2.2 tau\_delServices()

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	noOfServices	No of services in the array
in, out	pServicesToRemove	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\_freeServiceList()

Release the memory of a list received by tau\_getServiceList.

### **Parameters**

ſ	in	nServicesListBuffer	Pointer to list aquired by getServiceList.
	T11	poervicescistodilei	I diliter to list aquired by getoerviceList.

none	

### 5.11.2.4 tau\_getServiceList()

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().
in, out ppServicesListBuffer		Pointer to pointer containing the list. Has to be vos_memfree'd by user

### 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\_updServices()

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	noOfServices	No of services in the array
in,out	pServicesToUpdate	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

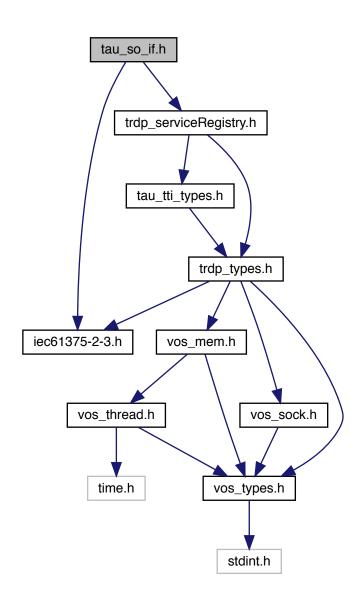
# Return values

TRDP_TIMEOUT_ERR	Reply timed out
TRDP_SEMA_ERR	Semaphore could not be aquired

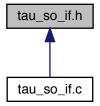
# 5.12 tau\_so\_if.h File Reference

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\_addServices (TRDP\_APP\_SESSION\_T appHandle, UINT16 noOfServices, SRM\_SERVICE\_ARRAY\_T \*pServicesToAdd, BOOL8 waitForCompletion)

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

 EXT\_DECL TRDP\_ERR\_T tau\_delServices (TRDP\_APP\_SESSION\_T appHandle, UINT16 noOfServices, SRM\_SERVICE\_ARRAY\_T \*pServicesToAdd, BOOL8 waitForCompletion)

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

 EXT\_DECL TRDP\_ERR\_T tau\_updServices (TRDP\_APP\_SESSION\_T appHandle, UINT16 noOfServices, SRM\_SERVICE\_ARRAY\_T \*pServicesToAdd, BOOL8 waitForCompletion)

Register an update a service.

• EXT\_DECL TRDP\_ERR\_T tau\_getServiceList (TRDP\_APP\_SESSION\_T appHandle, SRM\_SERVICE\_A ← RRAY\_T \*\*ppServicesToAdd)

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

• EXT\_DECL void tau\_freeServiceList (SRM\_SERVICE\_ARRAY\_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 2052 2019-08-28 10:54:24Z bloehr

# 5.12.2 Function Documentation

## 5.12.2.1 tau\_addServices()

```
EXT_DECL TRDP_ERR_T tau_addServices (

TRDP_APP_SESSION_T appHandle,

UINT16 noOfServices,

SRM_SERVICE_ARRAY_T * pServicesToAdd,

BOOL8 waitForCompletion )
```

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	noOfServices	No of services in the array	
in,out	pServicesToAdd	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\_delServices()

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	noOfServices	No of services in the array
in,out	pServicesToRemove	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\_freeServiceList()

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\_getServiceList()

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().
in,out	ppServicesListBuffer	Pointer to pointer containing the list. Has to be vos_memfree'd by user

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

### 5.12.2.5 tau\_updServices()

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	noOfServices	No of services in the array
in,out	pServicesToUpdate	Pointer to a service registry structure to be updated
in	waitForCompletion	if true, block for reply

### **Return values**

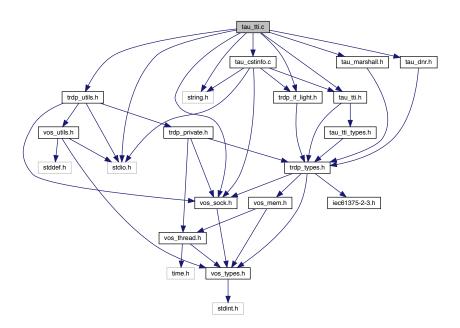
TRDP_NO_ERR	no error
TRDP_PARAM_ERR	Parameter error
TRDP_NODATA_ERR	Data currently not available, try again later
TRDP_TIMEOUT_ERR	Reply timed out
TRDP_SEMA_ERR	Semaphore could not be aquired

# 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.

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 consist.

## Return values

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	Parameter error

# 5.13.3.3 tau\_getCstFctInfo()

```
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.

### **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**

		appHandle	Handle returned by tlc_openSession().
		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

OWN Trn CotNo	own train consist number value 0 on error
OWITHIUSING	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 <i>pCstInfo</i> Pointer to a co		Pointer to a consist info structure to be returned.
in <i>cstUUID</i> UUID of th		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 return		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()

```
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

# 5.13.3.16 tau\_getVehInfo()

```
EXT_DECL TRDP_ERR_T tau_getVehInfo (

TRDP_APP_SESSION_T appHandle,
```

```
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

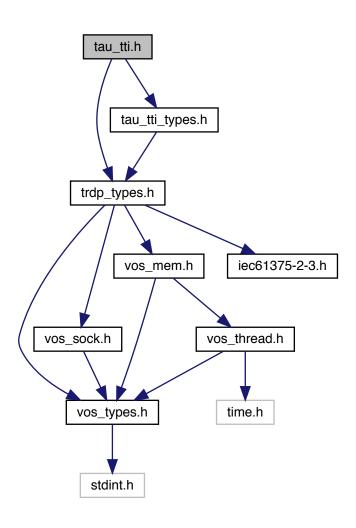
TRDP_NO_ERR	no error
TRDP_INIT_ERR	initialisation error

# 5.14 tau\_tti.h File Reference

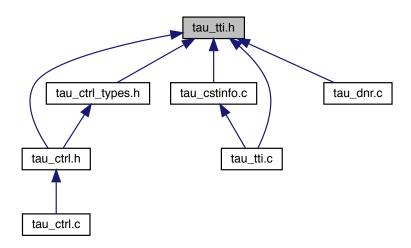
TRDP utility interface definitions.

```
#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\_deInitTTI (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

none	Function to terminate TTI access.
------	-----------------------------------

### **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 consi		

# 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**

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

# 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 consis		

### 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

ownOpCstNo	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
----	-----------	---------------------------------

n consist number value 0 on error	ownTrnCstNo
-----------------------------------	-------------

## 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.

# **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

## **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

# 5.14.2.15 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	pHandle 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	oCstLabel cstLabel = NULL means own consist	

## Return values

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	Parameter error

		to all the all the action and bridge are a consistent.	
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	

## **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_openS		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.

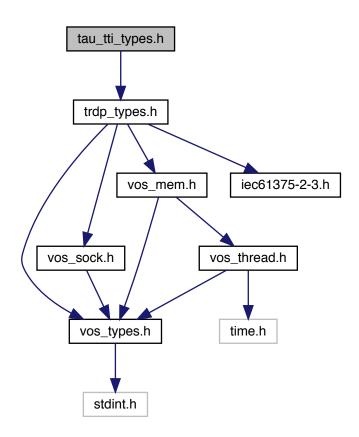
## Return values

TRDP_NO_ERR	no error
TRDP_INIT_ERR	initialisation error

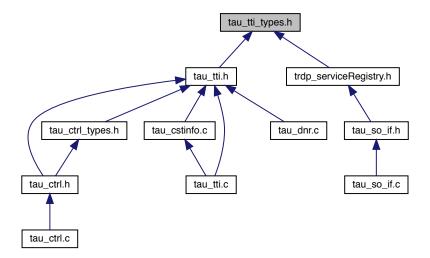
# 5.15 tau\_tti\_types.h File Reference

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)

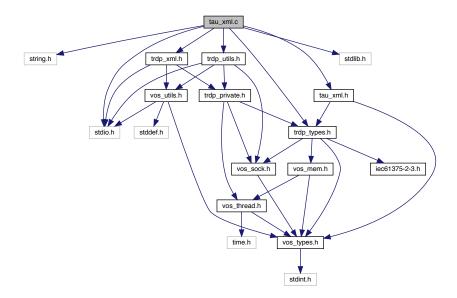
## Remarks

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\_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.

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. 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	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
```

## 5.16.3.3 tau\_freeXmlDoc()

```
EXT_DECL void tau_freeXmlDoc (  \label{eq:tau_dec} {\tt TRDP\_XML\_DOC\_HANDLE\_T} \ * \ pDocHnd \ )
```

Free all the memory allocated by tau\_prepareXmlDoc.

## **Parameters**

i	า	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	ырЫыудЕпхchgPar	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\_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

## Return values

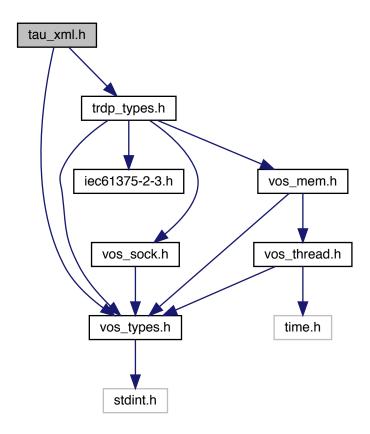
TRDP_NO_ERR	no error
TRDP_MEM_ERR	provided buffer to small
TRDP_PARAM_ERR	File not existing

# 5.17 tau\_xml.h File Reference

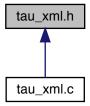
TRDP utility interface definitions.

```
#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**

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.

• 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 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.

## 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	numComld The number of entries in the Comld DatasetId mapping list	
in	n pComldDsIdMap Pointer to an array of structures of type TRDP_COMID_DSID_MAP_T		
in	n numDataset The number of datasets found in the configuration		
in ppDataset Pointer to an array of pointers to a structures of type TRDP_		Pointer to an array of pointers to a structures of type TRDP_DATASET_T	

## Return values

```
none
```

## 5.17.4.3 tau\_freeXmlDoc()

```
EXT_DECL void tau_freeXmlDoc (  \label{eq:tau_dec} {\tt TRDP\_XML\_DOC\_HANDLE\_T} \ * \ pDocHnd \ )
```

Free all the memory allocated by tau\_prepareXmlDoc.

## **Parameters**

i	า	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.

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.

#### **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

## 5.17.4.6 tau\_readXmlDatasetConfig()

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

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	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.

#### **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

The user must release the memory for ppComPar and pplfConfig (using vos\_memFree)

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 .

#### **Parameters**

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
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\_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
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
Ī	out	pNumServiceDefs	Pointer to number of defined Services
Ī	out	ppServiceDefs	Pointer to pointer of the defined Services

## Return values

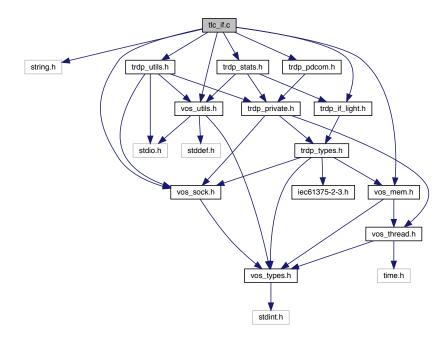
TRDP_NO_ERR	no error
TRDP_MEM_ERR	provided buffer to small
TRDP_PARAM_ERR	File not existing

# 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 is ValidSession (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.

- EXT\_DECL TRDP\_IP\_ADDR\_T tlc\_getOwnlpAddress (TRDP\_APP\_SESSION\_T 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.

Update a session.

- EXT\_DECL TRDP\_ERR\_T tlc\_updateSession (TRDP\_APP\_SESSION\_T appHandle)
- 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\_DECLTRDP\_ERR\_T tlc\_process (TRDP\_APP\_SESSION\_T appHandle, TRDP\_FDS\_T \*pRfds, INT32 \*pCount)

Work loop of the TRDP handler.

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**

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

## 5.18.2.2 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	pMarshall	Marshall 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

#### 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 The handle returned by tlc_openSession	
out	pInterval pointer to needed interval	
in,out	put 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!

ſ	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**

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

#### 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.

out	pAppHandle	A handle for further calls to the trdp stack

#### **Parameters**

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	n pMdDefault Pointer to default MD configuration		
in	7		
		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\_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

#### 5.18.2.12 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
--	----	-----------	--

#### Return values

TRDP_NO_ERR	no error
TRDP_NOINIT_ERR	handle invalid
TRDP_PARAM_ERR	handle NULL

## 5.18.2.13 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

#### Return values

TRDP_NO_ERR	no error
TRDP_NOINIT_ERR	handle invalid

## 5.18.2.14 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

## 5.18.2.15 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

## 5.18.2.16 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	A handle for further calls to the trdp stack
----	-----------	--

### Return values

TRDP_NO_ERR	no error
TRDP_INIT_ERR	not yet inited
TRDP_PARAM_ERR	parameter error

## 5.18.2.17 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	
--	----	-----------	--	--

## Return values

TRDP_NO_ERR	
TRDP_INIT_ERR	
TRDP_MUTEX_ERR	

## 5.18.2.18 trdp\_isValidSession()

Check if the session handle is valid.

#### **Parameters**

in <i>pSessionHandle</i>	pointer to packet data (dataset)
--------------------------	----------------------------------

## Return values

TRUE	is valid
FALSE	is invalid

## 5.18.2.19 trdp\_releaseAccess()

Release access to the session.

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

## Return values



Here is the call graph for this function:



## 5.18.2.20 trdp\_sessionQueue()

Get the session queue head pointer.

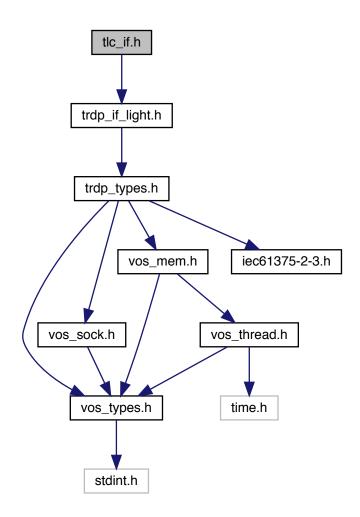
## Return values

&sSession

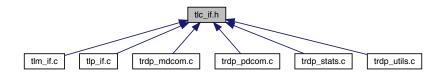
# 5.19 tlc\_if.h File Reference

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.

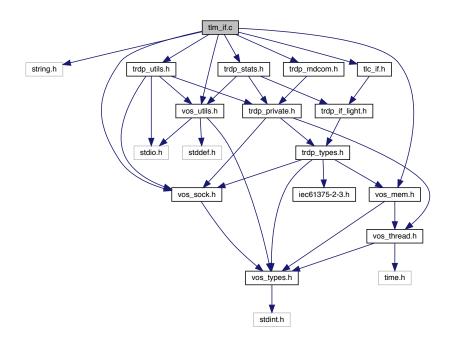
#### Return values

&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.

TRDP\_ERR\_T tlm\_notify (TRDP\_APP\_SESSION\_T appHandle, const void \*pUserRef, TRDP\_MD\_CAL
 LBACK\_T pfCbFunction, UINT32 comId, UINT32 etbTopoCnt, UINT32 opTrnTopoCnt, TRDP\_IP\_ADDR\_T
 srclpAddr, TRDP\_IP\_ADDR\_T destlpAddr, TRDP\_FLAGS\_T pktFlags, const TRDP\_SEND\_PARAM\_T \*p
 SendParam, const UINT8 \*pData, UINT32 dataSize, const TRDP\_URI\_USER\_T sourceURI, const TRDP
 URI\_USER\_T destURI)

Initiate sending MD notification message.

Initiate sending MD request message.

- TRDP\_ERR\_T tlm\_request (TRDP\_APP\_SESSION\_T appHandle, const void \*pUserRef, TRDP\_MD\_CA← LLBACK\_T pfCbFunction, TRDP\_UUID\_T \*pSessionId, UINT32 comId, UINT32 etbTopoCnt, UINT32 op← TrnTopoCnt, TRDP\_IP\_ADDR\_T srclpAddr, TRDP\_IP\_ADDR\_T destIpAddr, TRDP\_FLAGS\_T pktFlags, U← INT32 numReplies, UINT32 replyTimeout, const TRDP\_SEND\_PARAM\_T \*pSendParam, const UINT8 \*p← Data, UINT32 dataSize, const TRDP\_URI\_USER\_T sourceURI, const TRDP\_URI\_USER\_T destURI)
- EXT\_DECL TRDP\_ERR\_T tIm\_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.

- TRDP\_ERR\_T tlm\_delListener (TRDP\_APP\_SESSION\_T appHandle, TRDP\_LIS\_T listenHandle)
   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.

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.

 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 UINT8 \*pData, UINT32 dataSize)

Send a MD reply query message.

TRDP\_ERR\_T tlm\_confirm (TRDP\_APP\_SESSION\_T appHandle, const TRDP\_UUID\_T \*pSessionId, UI
 — NT16 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**

in	appHandle	the handle returned by tlc_openSession	
in	<i>p</i> ⇔	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

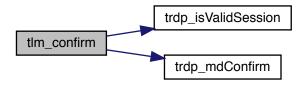
### **Return values**

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	parameter error

# Return values

TRDP_MEM_ERR	out of memory
TRDP_NOSESSION_ERR	no such session
TRDP_NOINIT_ERR	handle invalid

Here is the call graph for this function:



# 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()

```
TRDP_FDS_T * pFileDesc,
INT32 * pNoDesc )
```

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	plnterval 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()

```
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

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		

## **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	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

#### 5.20.2.7 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
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()

```
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
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()

Send a MD reply message.

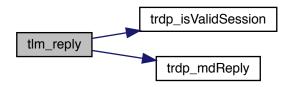
Send a MD reply message after receiving an request User reference, source and destination IP addresses as well as topo counts and packet flags are taken from the session

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	pSendParam	Pointer to send parameters, NULL to use default send parameters
in	pData	pointer to packet data / dataset
in Generate	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

Here is the call graph for this function:



# 5.20.2.10 tlm\_replyQuery()

```
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 UINT8 * pData,

UINT32 dataSize )
```

Send a MD reply query message.

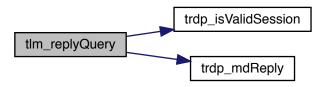
Send a MD reply query message after receiving a request and ask for confirmation. User reference, source and destination IP addresses as well as topo counts and packet flags are taken from the session

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

Here is the call graph for this function:



# 5.20.2.11 tlm\_request()

```
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

in	appHandle	the handle returned by tlc_openSession
in	pUserRef	user supplied value returned with reply

## **Parameters**

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

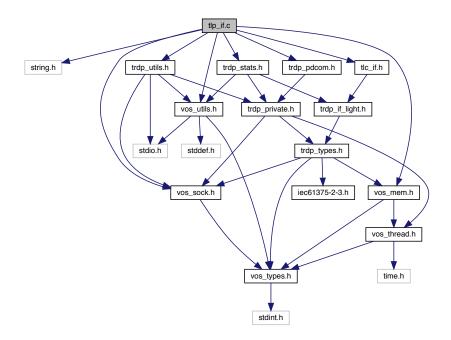
TRDP_NO_ERR	no error
TRDP_PARAM_ERR	parameter error
TRDP_MEM_ERR	out of memory
TRDP_NOINIT_ERR	handle invalid

# 5.21 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.
- 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 comId, 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.\*
- TRDP\_ERR\_T tlp\_unpublish (TRDP\_APP\_SESSION\_T appHandle, TRDP\_PUB\_T pubHandle) Stop sending PD messages.
- TRDP\_ERR\_T tlp\_put (TRDP\_APP\_SESSION\_T appHandle, TRDP\_PUB\_T pubHandle, const UINT8 \*p
   — Data, UINT32 dataSize)

Update the process data to send.

 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.

EXT\_DECL TRDP\_ERR\_T tlp\_request (TRDP\_APP\_SESSION\_T appHandle, TRDP\_SUB\_T subHandle, UINT32 serviceld, UINT32 comld, 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 replyComld, TRDP\_IP\_ADDR\_T replylpAddr)

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.

• EXT\_DECL TRDP\_ERR\_T tlp\_get (TRDP\_APP\_SESSION\_T appHandle, TRDP\_SUB\_T subHandle, TR

DP\_PD\_INFO\_T \*pPdInfo, UINT8 \*pData, UINT32 \*pDataSize)

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

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

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**

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()

```
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.

in	appHandle	the handle returned by tlc_openSession
in	pubHandle	the handle returned by publish
in,out <i>pData</i>		pointer to application's data buffer
in,out	dataSize	size of data

#### 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()

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

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()

```
EXT_DECL TRDP_ERR_T tlp_resubscribe (

TRDP_APP_SESSION_T appHandle,

TRDP_SUB_T subHandle,

UINT32 etbTopoCnt,

UINT32 opTrnTopoCnt,

TRDP_IP_ADDR_T srcIpAddr1,

TRDP_IP_ADDR_T srcIpAddr2,

TRDP_IP_ADDR_T destIpAddr )
```

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
in	redId	will be set for all ComID's with the given redld, 0 to change for all redld
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	comId	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

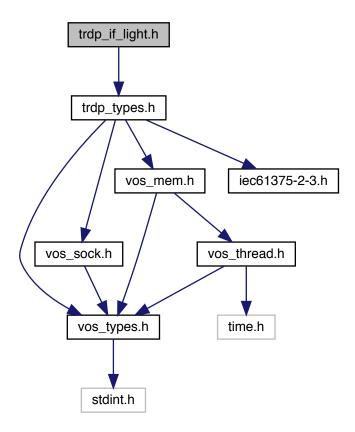
# Remarks

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.

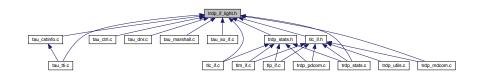
# 5.23 trdp\_if\_light.h File Reference

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 Ttlc closeSession (TRDP APP SESSION TappHandle)

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 comId, 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.

EXT\_DECL TRDP\_ERR\_T tlp\_request (TRDP\_APP\_SESSION\_T appHandle, TRDP\_SUB\_T subHandle, UINT32 serviceld, UINT32 comld, 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 replyComld, 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\_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.

• EXT\_DECL TRDP\_ERR\_T tlp\_get (TRDP\_APP\_SESSION\_T appHandle, TRDP\_SUB\_T subHandle, TR

DP PD INFO T \*pPdInfo, UINT8 \*pData, UINT32 \*pDataSize)

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 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\_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.

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.

• TRDP\_ERR\_T tlm\_replyQuery (TRDP\_APP\_SESSION\_T appHandle, const TRDP\_UUID\_T \*pSessionId, UINT32 comId, UINT32 comId, UINT32 confirmTimeout, const TRDP\_SEND\_PARAM\_T \*pSendParam, const UINT8 \*pData, UINT32 dataSize)

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 join 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**

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

## 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.

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.

in appHandle the handle returned by tlc_ope		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	Handle the handle returned by tlc_openSession	
in, out   pNumSubs   In: The number of subscriptions requested Out: Number of subscriptions return			
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

# Return values

```
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**

string

# 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 <i>pMemConfig</i>		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	shall Pointer to marshalling configuration	
in	pPdDefault	Pointer to default PD configuration	
in	pMdDefault	ult Pointer to default MD configuration	
in pProcessConfig Pointer to process configuration only option parameter is used here to de behavior all other parameters are only used to feed statistics		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\_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.

	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

# 5.23.2.17 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
----	-----------	--

# Return values

TRDP_NO_ERR	no error
TRDP_NOINIT_ERR	handle invalid
TRDP_PARAM_ERR	handle NULL

# 5.23.2.18 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

#### 5.23.2.19 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

## Return values

TRDP_NO_ERR	no error
TRDP_NOINIT_ERR	handle invalid

#### 5.23.2.20 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

#### 5.23.2.21 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

# 5.23.2.22 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	A handle for further calls to the trdp stack	Ì
----	-----------	--	---

#### Return values

TRDP_NO_ERR	no error
TRDP_INIT_ERR	not yet inited
TRDP_PARAM_ERR	parameter error

### 5.23.2.23 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	

TRDP_NO_ERR	no error

### Return values

TRDP_NOSESSION_ERR	no such session
TRDP_NOINIT_ERR	handle invalid

# 5.23.2.24 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	dr 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	

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	parameter error
TRDP_MEM_ERR	out of memory
TRDP NOINIT ERR	handle invalid

### 5.23.2.25 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

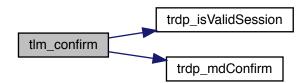
### **Parameters**

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	

### 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

Here is the call graph for this function:



# 5.23.2.26 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

# 5.23.2.27 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

### 5.23.2.28 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
TRDP_NOINIT_ERR	handle invalid

# 5.23.2.29 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
in,out	pCount	pointer to number of ready descriptors

# Return values

TRDP_NO_ERR	no error
TRDP_NOINIT_ERR	handle invalid

# 5.23.2.30 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
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

# 5.23.2.31 tlm\_reply()

```
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

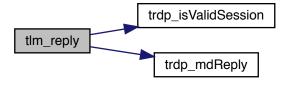
### **Parameters**

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	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

Here is the call graph for this function:



# 5.23.2.32 tlm\_replyQuery()

```
const TRDP_UUID_T * pSessionId,
UINT32 comId,
UINT16 userStatus,
UINT32 confirmTimeout,
const TRDP_SEND_PARAM_T * pSendParam,
const UINT8 * 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

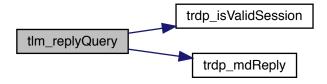
### **Parameters**

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

Here is the call graph for this function:



# 5.23.2.33 tlm\_request()

```
EXT_DECL TRDP_ERR_T tlm_request (

TRDP_APP_SESSION_T appHandle,
```

```
const void * pUserRef,
TRDP_MD_CALLBACK_T pfCbFunction,
{\tt 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	

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	parameter error
TRDP_MEM_ERR	out of memory
TRDP_NOINIT_ERR	handle invalid

```
5.23.2.34 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

# 5.23.2.35 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

# Return values

TRDP_NOINIT_ERR	handle invalid
-----------------	----------------

# 5.23.2.36 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	redId invalid or not existing
TRDP_NOINIT_ERR	handle invalid

# 5.23.2.37 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

# 5.23.2.38 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

### 5.23.2.39 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	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

# 5.23.2.40 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

#### 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

### 5.23.2.41 tlp\_putImmediate()

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

# 5.23.2.42 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

### 5.23.2.43 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

# 5.23.2.44 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

# 5.23.2.45 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

### 5.23.2.46 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

# 5.23.2.47 tlp\_unpublish()

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

# Return values

TRDP_NOINIT_ERR	handle invalid
-----------------	----------------

# 5.23.2.48 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

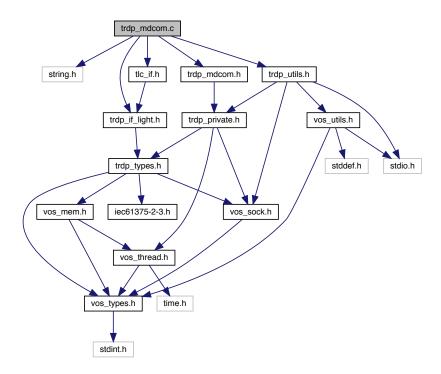
TRDP_NO_ERR	no error
TRDP_PARAM_ERR	parameter error
TRDP_NOSUB_ERR	not subscribed
TRDP_NOINIT_ERR	handle invalid

# 5.24 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**

msgType	TRDP_MSG_MN or TRDP_MSG_MR	
appHandle	the handle returned by tlc_init	
pUserRef	user supplied value returned with reply	
pfCbFunction	Pointer to listener specific callback function, NULL to use default function	
pSessionId	return session ID	
comId	comld of packet to be sent	
etbTopoCnt	ETB topocount to use, 0 if consist local communication	
opTrnTopoCnt	operational topocount, != 0 for orientation/direction sensitive communication	
srclpAddr	own IP address, 0 - srcIP will be set by the stack	
destlpAddr	where to send the packet to	
pktFlags	OPTION: TRDP_FLAGS_DEFAULT, TRDP_FLAGS_NONE,	
	TRDP_FLAGS_MARSHALL	
numExpReplies	number of expected replies, 0 if unknown	
replyTimeout	timeout for reply	
replyStatus	status to be returned	
pSendParam	Pointer to send parameters, NULL to use default send parameters	
pData	pointer to packet data / dataset	
dataSize	size of packet data	
srcURI	only functional group of source URI	
destURI	only functional group of destination URI	
	appHandle pUserRef pfCbFunction pSessionId comId etbTopoCnt opTrnTopoCnt srclpAddr destlpAddr pktFlags numExpReplies replyTimeout replyStatus pSendParam pData dataSize srcURI	

### 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()

```
void trdp_mdCheckTimeouts ( \label{trdp_mdCheckTimeouts} \texttt{TRDP\_SESSION\_PT} \ app\textit{Handle} \ )
```

Checking message data timeouts Call user's callback if needed.

#### **Parameters**

in appHandle session pointe
-----------------------------

### 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()

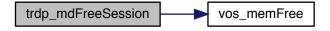
```
void trdp_mdFreeSession ( \label{eq:mdFreeSession} \texttt{MD\_ELE\_T} \ * \ pMDSession \ )
```

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()

```
TRDP_ERR_T trdp_mdReply (

const TRDP_MSG_T msgType,

TRDP_APP_SESSION_T appHandle,

TRDP_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.

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.

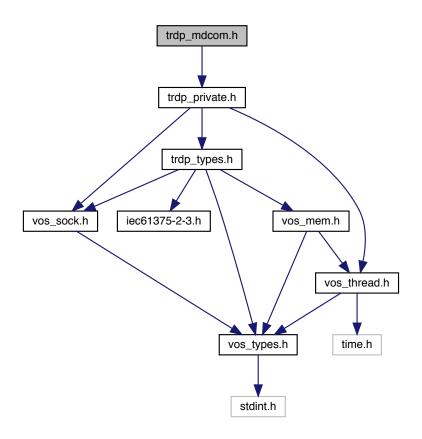
# **Parameters**

in	appHandle	session pointer
----	-----------	-----------------

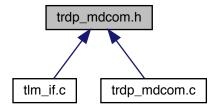
# 5.25 trdp\_mdcom.h File Reference

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**

msgType	TRDP_MSG_MN or TRDP_MSG_MR	
appHandle	the handle returned by tlc_init	
pUserRef	user supplied value returned with reply	
pfCbFunction	Pointer to listener specific callback function, NULL to use default function	
pSessionId	return session ID	
comld comld of packet to be sent		
etbTopoCnt	etbTopoCnt ETB topocount to use, 0 if consist local communication	
opTrnTopoCnt	operational topocount, != 0 for orientation/direction sensitive communication	
	appHandle pUserRef pfCbFunction pSessionId comId etbTopoCnt	

# **Parameters**

srclpAddr	own IP address, 0 - srcIP will be set by the stack	
destlpAddr	where to send the packet to	
pktFlags	OPTION: TRDP_FLAGS_DEFAULT, TRDP_FLAGS_NONE,	
	TRDP_FLAGS_MARSHALL	
numExpReplies	number of expected replies, 0 if unknown	
replyTimeout	timeout for reply	
replyStatus	status to be returned	
pSendParam	Pointer to send parameters, NULL to use default send parameters	
pData	pointer to packet data / dataset	
dataSize	size of packet data	
srcURI	only functional group of source URI	
destURI	only functional group of destination URI	
	destIpAddr pktFlags  numExpReplies replyTimeout replyStatus pSendParam pData dataSize srcURI	

# 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{\ )}
```

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	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**

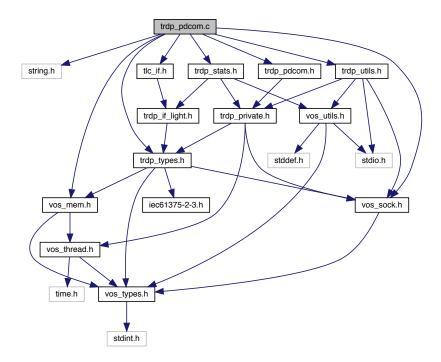
in	appHandle	session pointer
----	-----------	-----------------

# 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.

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\_pdCheck()

Check if the PD header values and the CRCs are sane.

### **Parameters**

ir	า	pPacket	pointer to the packet to check
ir	า	packetSize	max size to check
Οl	ıt	plsTSN	set to TRUE on return if PD2 frame

### **Return values**

TRDP_NO_ERR	
TRDP_CRC_ERR	

# 5.26.2.2 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

# 5.26.2.3 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

#### 5.26.2.4 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.

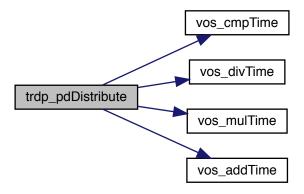
#### **Parameters**

in	pSndQueue	pointer to send queue
----	-----------	-----------------------

#### Return values

```
TRDP_NO_ERR
```

Here is the call graph for this function:



#### 5.26.2.5 trdp\_pdHandleTimeOuts()

Check for time outs.

## **Parameters**

in appHandle	application handle
--------------	--------------------

Here is the call graph for this function:



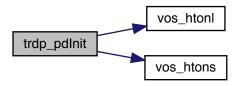
# 5.26.2.6 trdp\_pdInit()

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:



# 5.26.2.7 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
-------------	-----------------------

## 5.26.2.8 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

# 5.26.2.9 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	

# 5.26.2.10 trdp\_pdSendElement()

# Send a due PD message.

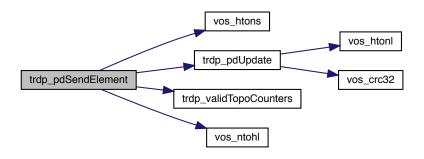
# **Parameters**

iı	า	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:



# 5.26.2.11 trdp\_pdSendImmediate()

# Send PD message immediately.

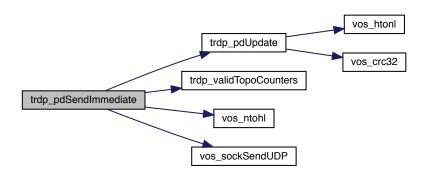
## **Parameters**

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:



# 5.26.2.12 trdp\_pdSendQueued()

Send all due PD messages.

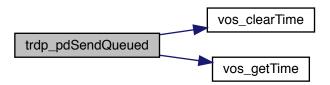
### **Parameters**

in ap	pHandle	session pointer
-------	---------	-----------------

## Return values

TRDP_NO_ERR	no error
TRDP_IO_ERR	socket I/O error

Here is the call graph for this function:



# 5.26.2.13 trdp\_pdUpdate()

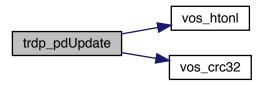
```
void trdp_pdUpdate ( {\tt PD\_ELE\_T~*~pPacket~)}
```

Update the header values.

## **Parameters**

in	pPacket	pointer to the packet to update
----	---------	---------------------------------

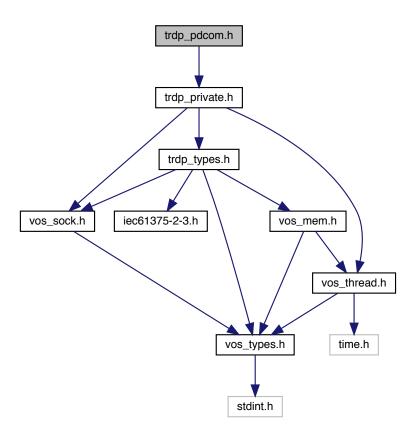
Here is the call graph for this function:



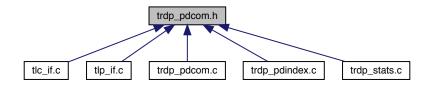
# 5.27 trdp\_pdcom.h File Reference

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 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\_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	

## 5.27.2.2 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

# 5.27.2.3 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

## 5.27.2.4 trdp\_pdDistribute()

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.

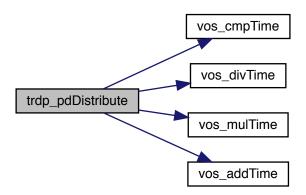
#### **Parameters**

in <i>pSndQueue</i>	pointer to send queue
---------------------	-----------------------

#### **Return values**

```
TRDP_NO_ERR
```

Here is the call graph for this function:



# 5.27.2.5 trdp\_pdHandleTimeOuts()

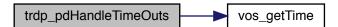
```
void trdp_pdHandleTimeOuts ( \label{trdp_pdHandleTimeOuts} \mbox{TRDP\_SESSION\_PT } \mbox{\it appHandle })
```

Check for time outs.

#### **Parameters**

in	appHandle	application handle
----	-----------	--------------------

Here is the call graph for this function:



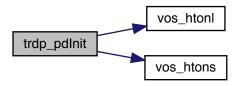
## 5.27.2.6 trdp\_pdlnit()

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:



# 5.27.2.7 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
-------------	-----------------------

## 5.27.2.8 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

# 5.27.2.9 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	

# 5.27.2.10 trdp\_pdSendElement()

# Send a due PD message.

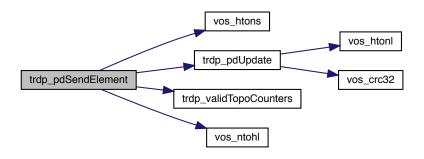
# **Parameters**

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:



# 5.27.2.11 trdp\_pdSendImmediate()

Send PD message immediately.

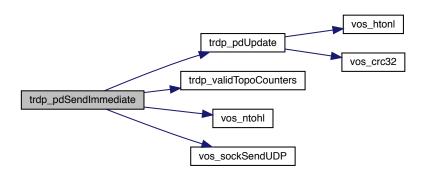
#### **Parameters**

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:



# 5.27.2.12 trdp\_pdSendQueued()

Send all due PD messages.

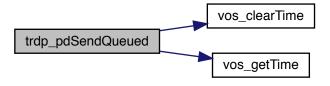
### **Parameters**

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:



## 5.27.2.13 trdp\_pdUpdate()

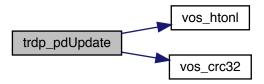
```
void trdp_pdUpdate ( {\tt PD\_ELE\_T~*~pPacket~)}
```

Update the header values.

## **Parameters**

in	pPacket	pointer to the packet to update
----	---------	---------------------------------

Here is the call graph for this function:

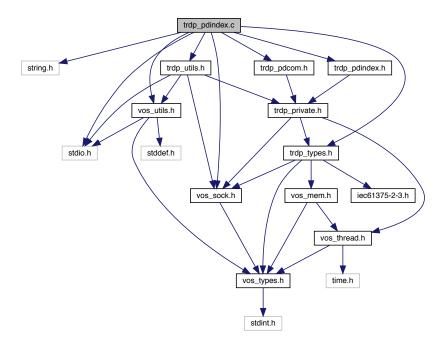


# 5.28 trdp\_pdindex.c File Reference

Functions for indexed PD communication.

```
#include <string.h>
#include <stdio.h>
#include "trdp_types.h"
#include "trdp_utils.h"
#include "trdp_pdcom.h"
#include "vos_utils.h"
#include "vos_sock.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

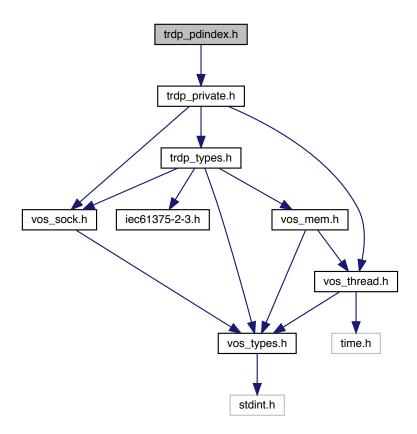
## Remarks

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.

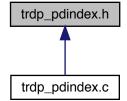
# 5.29 trdp\_pdindex.h File Reference

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.

# **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

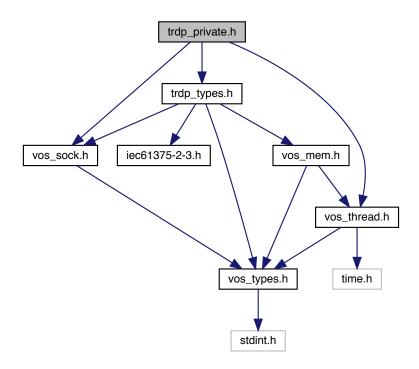
This Source Code Form is subject to the terms of the Mozilla Public License, v. 2.0. If a copy of the MPL was not distributed with this file, You can obtain one at http://mozilla.org/MPL/2.0/. Copyright Bombardier Transportation Inc. or its subsidiaries and others, 2019. All rights reserved.

# 5.30 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

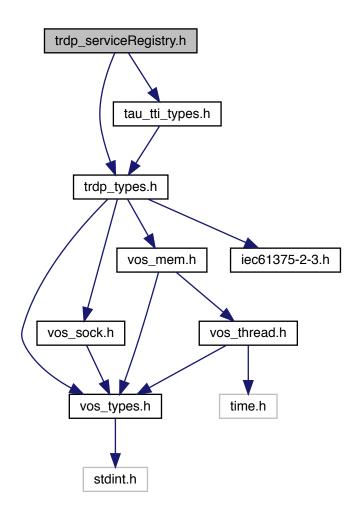
TRDP_SOCK_INVAL	Socket is undefined.
TRDP_SOCK_PD	Socket is used for UDP process data.
TRDP_SOCK_MD_UDP	Socket is used for UDP message data.
TRDP_SOCK_MD_TCP	Socket is used for TCP message data.
TRDP_SOCK_PD_TSN	Socket is used for TSN process data.

# 5.31 trdp\_serviceRegistry.h File Reference

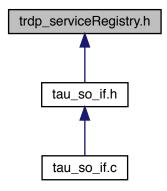
Additional definitions for IEC 61375-2-3 (Service Discovery) The definitions herein are preliminary and may change with the next major release of the IEC 61375-2-3 standard.

```
#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 serviceRegistryEntry

Preliminary definition of a service registry entry.

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\_ARRAY\_T"
 SRM\_SERVICE\_ARRAY\_T.

 #define SRM\_SERVICE\_READ\_REP\_DSID SRM\_SERVICE\_DSID SRM\_SERVICE\_ARRAY\_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\_ARRAY\_T"
 SRM\_SERVICE\_ARRAY\_T.

 #define SRM\_SERVICE\_ADD\_REQ\_DSID SRM\_SERVICE\_DSID SRM\_SERVICE\_ARRAY\_T.

#define SRM\_SERVICE\_ADD\_REP\_COMID 115u

Reply returns instanceld.

#define SRM\_SERVICE\_ADD\_REP\_DSID SRM\_SERVICE\_DSID
 SRM\_SERVICE\_ARRAY\_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\_ARRAY\_T"
 SRM\_SERVICE\_ARRAY\_T.

 #define SRM\_SERVICE\_UPD\_NOTIFY\_DSID SRM\_SERVICE\_DSID SRM\_SERVICE\_ARRAY\_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\_ARRAY\_T"
 SRM\_SERVICE\_ARRAY\_T.

#define SRM\_SERVICE\_DEL\_REQ\_DSID SRM\_SERVICE\_DSID

SRM\_SERVICE\_ARRAY\_T.

#define SRM\_SERVICE\_DEL\_REP\_COMID 118u

MD reply OK or not.

• #define SOA\_TYPE(serviceId) ((serviceId) & 0xFFFFFF)

return 24 Bit service type part of serviceID

#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 serviceIds (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.

typedef struct serviceRegistryEntry SRM\_SERVICE\_REGISTRY\_ENTRY

Preliminary definition of a service registry entry.

# 5.31.1 Detailed Description

Additional definitions for IEC 61375-2-3 (Service Discovery) The definitions herein are preliminary and may 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.

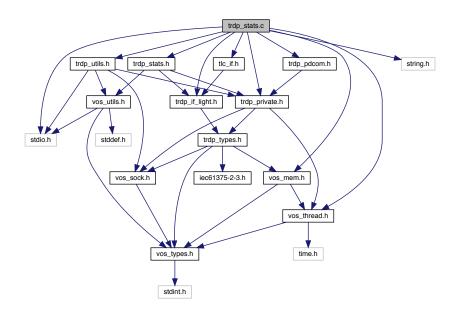
Transport: Trainwide MD over UDP / Multicast SRVINFO notification data:

# 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)
- void trdp\_initStats (TRDP\_APP\_SESSION\_T appHandle)
   Init statistics.

Update the 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		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	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:



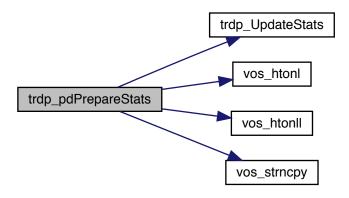
# 5.32.2.8 trdp\_pdPrepareStats()

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**

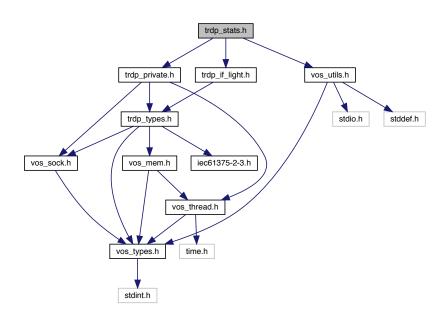
	in	appHandle	the handle returned by tlc_openSession	
--	----	-----------	--	--

# 5.33 trdp\_stats.h File Reference

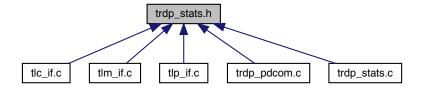
Statistics for TRDP communication.

```
#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**

	n <b>appHandle</b>	the handle returned by tlc_openSession	]
--	--------------------	--	---

- < host name
- < leader host name Here is the call graph for this function:



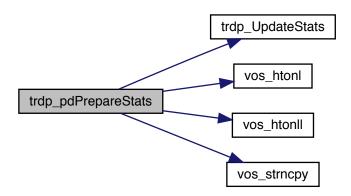
### 5.33.2.2 trdp\_pdPrepareStats()

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

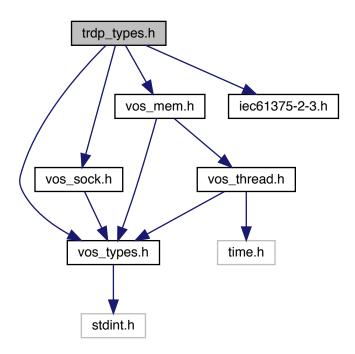
# 5.35 trdp\_types.h File Reference

Typedefs for TRDP communication.

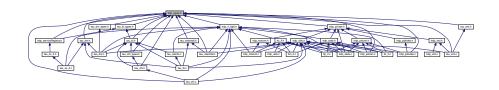
```
#include "vos_types.h"
#include "vos_mem.h"
#include "vos_sock.h"
```

#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.

#### **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 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.
```

### 5.35.1 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, 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

```
 \label{typedef}  \  \, \text{TRDP\_ERR\_T} \  \, (* \  \, \text{TRDP\_MARSHALL\_T}) \  \, (\text{void *pRefCon, UINT32 comId, UINT8 *pSrc, UINT32 src} \leftarrow \\ \text{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 size of the provide buffer / size of the treated message	
in,out	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  $\leftarrow$  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	comld Comld to identify the structure out of a configurati	
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

Invalid/unknown.
=UINT8
char, can be used also as UTF8
Unicode UTF-16 character.
Signed integer, 8 bit.
Signed integer, 16 bit.
Signed integer, 32 bit.
Signed integer, 64 bit.
Unsigned integer, 8 bit.
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.

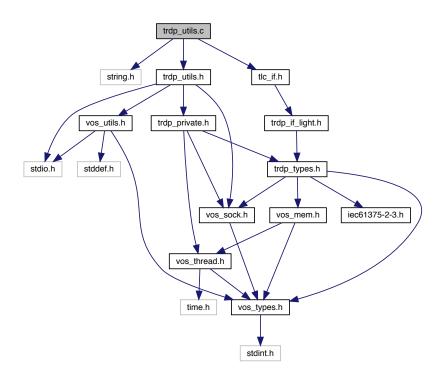
#### Enumerator

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.

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.

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 iface 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

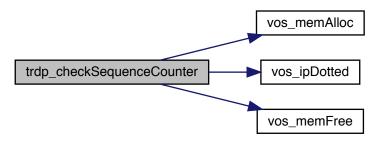
#### **Parameters**

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	Uri Null terminated listener URI string to compare	
in	n destUri Null terminated destination URI string to co		

### 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	listenedSourcelPlow	Lower bound IP	
Ī	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
----

#### 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

packet	size the size of the complete packet to be sent or received
--------	---

### 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}\_\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

### 5.36.2.15 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 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

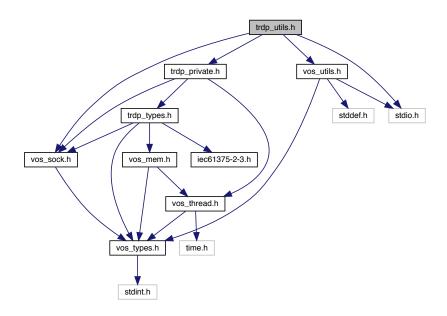
```
TRUE | Filter criteria matched FALSE Filter criteria not matched
```

# 5.37 trdp\_utils.h File Reference

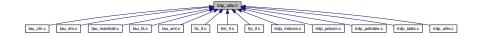
Common utilities for TRDP communication.

```
#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).
- int trdp\_checkSequenceCounter (PD\_ELE\_T \*pElement, UINT32 sequenceCounter, TRDP\_IP\_ADDR\_← T srcIP, TRDP\_MSG\_T msgType)

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

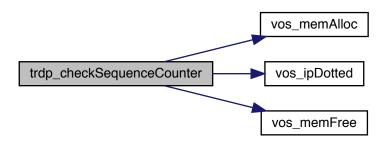
### **Parameters**

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

### 5.37.2.4 trdp\_findSubAddr()

Return the element with same comld and IP addresses.

# Parameters

ir	pHead	pointer to head of queue
ir	addr	Pub/Sub handle (Address, ComID, srcIP & dest IP, serviceId) to search for
ir	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 <i>received</i>		receivedSrcIP	Received IP address
i	n	listenedSourceIPlow	Lower bound IP
i	in listenedSourceIPhigh Upp		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	pDelete	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/TC		
in	options blocking/nonblocking		
in	rcvMostly primarily used for receiving (tbd: bind on sender, too?)		
out	useSocket	ocket 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**

nono	
HOHE	

# 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**

### 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

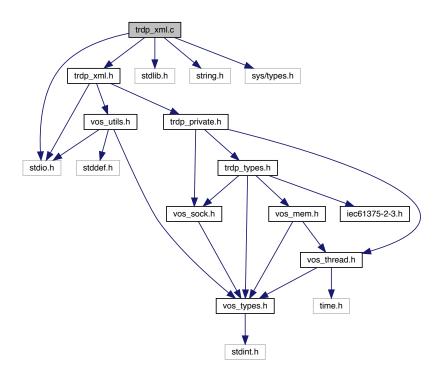
```
TRUE | Filter criteria matched FALSE Filter criteria not matched
```

# 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.
- 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\_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()

```
void trdp_XMLClose ( {\tt XML\_HANDLE\_T~*~p\textit{XML}~)}
```

Closes the XML parsng.

#### **Parameters**

in	pXML	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	pXML	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

## 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).

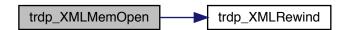
# Parameters

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()

# 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**

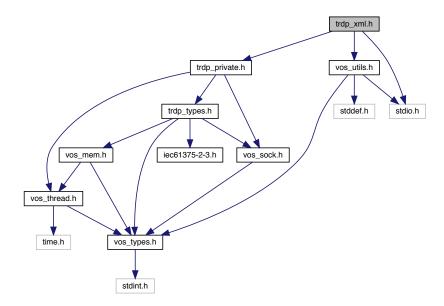
```
0 if found !=0 if not found
```

# 5.39 trdp\_xml.h File Reference

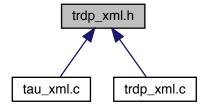
Simple XML parser.

```
#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
ı		p/(///_	1 ciritor to local data

## **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>pXML</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

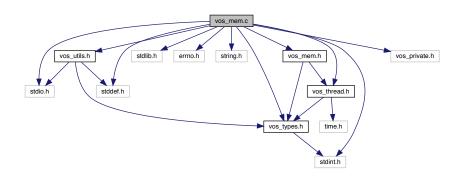
```
0 if found !=0 if not found
```

# 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

Pointer to found element or NULL
----------------------------------

## 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	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

# 5.40.2.7 vos\_qsort()

Sort an array.

This is just a wrapper for the standard qsort function.

## **Parameters**

i didiliotoro			
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

VOC NO EDD	no orror
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

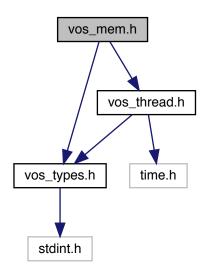
## Return values

0	- equal
<0	- string1 less than string 2
>0	- string 1 greater than string 2

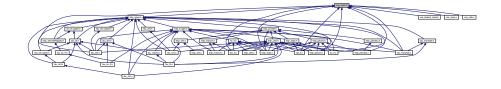
# 5.41 vos\_mem.h File Reference

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\_MAX\_PREALLOCATE 10u

Max blocks to pre-allocate.

#define VOS MEM NBLOCKSIZES 15u

No of pre-defined block sizes.

• #define VOS\_MEM\_BLOCKSIZES

We internally allocate memory always by these block sizes.

• #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.

### **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\_BLOCKSIZES

```
#define VOS_MEM_BLOCKSIZES
```

#### Value:

```
\{34\mathrm{u},\ 48\mathrm{u},\ 128\mathrm{u},\ 180\mathrm{u},\ 256\mathrm{u},\ 512\mathrm{u},\ 1024\mathrm{u},\ 1480\mathrm{u},\ 2048\mathrm{u},\ \setminus \ 4096\mathrm{u},\ 11520\mathrm{u},\ 16384\mathrm{u},\ 32768\mathrm{u},\ 65536\mathrm{u},\ 131072\mathrm{u}\}
```

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.

## 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 bl	lock
------------------------------	------

## Return values

Pointer	to memory area
NULL	if no memory available

# 5.41.3.3 vos\_memCount()

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 <i>pMemoryArea</i>	Pointer to memory area used
-----------------------	-----------------------------

#### 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		pz.oc.	

#### 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

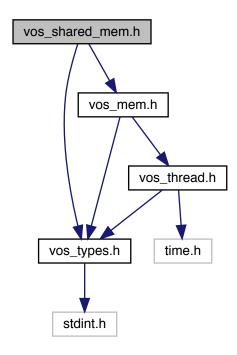
0	- equal
<0	- string1 less than string 2
>0	- string 1 greater than string 2

# 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.

 $\bullet \ \ \mathsf{EXT\_DECL} \ \ \mathsf{VOS\_ERR\_T} \ \ \mathsf{vos\_sharedClose} \ \ (\mathsf{VOS\_SHRD\_T} \ \ \mathsf{handle}, \ \mathsf{const} \ \ \mathsf{UINT8} \ \ *\mathsf{pMemoryArea})$ 

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

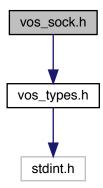
# Return values

VOS_MEM_ERR	no memory available
-------------	---------------------

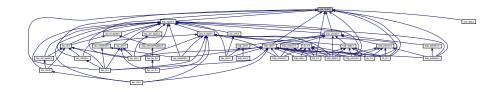
# 5.43 vos\_sock.h File Reference

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	val	Initial value.
----	-----	----------------

## 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()

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()

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()

# Byte swapping 4 Bytes.

#### **Parameters**

## Return values

swapped value

# 5.43.3.11 vos\_ntohll()

# Byte swapping 8 Bytes.

# **Parameters**

in	val	Initial 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()

```
EXT_DECL VOS_ERR_T vos_sockBind (
SOCKET sock,
UINT32 ipAddress,
UINT16 port )
```

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 <i>pMAC</i>	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 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	pointer to data to send	
in,out	pSize	In: size of the data to send, Out: no of bytes sent	
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**

ir	sock	socket descriptor
ir	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
in	pOptions	pointer to socket options (optional)

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.

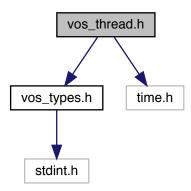
Must be called after last socket call

# 5.44 vos\_thread.h File Reference

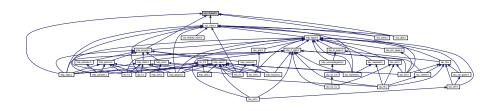
Threading functions for OS abstraction.

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

Include dependency graph for vos\_thread.h:



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.

 $\bullet \ \ \mathsf{EXT\_DECL} \ \ \mathsf{VOS\_ERR\_T} \ \ \mathsf{vos\_semaCreate} \ \ (\mathsf{VOS\_SEMA\_T} \ *\mathsf{pSema}, \ \mathsf{VOS\_SEMA\_STATE\_T} \ \mathsf{initialState})$ 

• EXT\_DECL void vos\_semaDelete (VOS\_SEMA\_T sema)

Delete a semaphore.

Create 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 p7	<i>īme</i> Poir	nter to time value
--------	-----------------	--------------------

# 5.44.2.6 vos\_getTime()

Return the current monotonic time in sec and us.

#### **Parameters**

out <i>pTime</i>	Pointer 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	pMutex	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	pMutex	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
j	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)	
--	----	--------	---	--

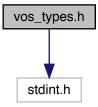
#### Return values

VOS_NO_ERR	no error
VOS_INIT_ERR	module not initialised
VOS_NOINIT_ERR	invalid handle
VOS_PARAM_ERR	parameter out of range/invalid

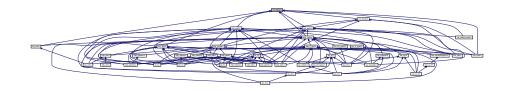
# 5.45 vos\_types.h File Reference

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.

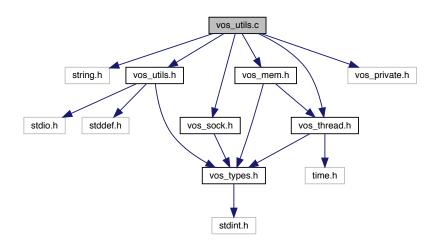
## Enumerator

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()

```
\begin{tabular}{ll} \begin{tabular}{ll} const $char*$ $vos\_getVersionString ( \\ void ) \end{tabular}
```

Return a human readable version representation.

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

# Return values

const	string

## 5.46.2.5 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	in <i>pRefCon</i>	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()

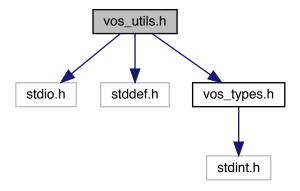
DeInitialize the vos library.

Should be called last after TRDP stack/application does not use any VOS function anymore.

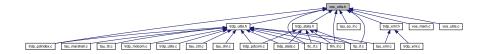
# 5.47 vos\_utils.h File Reference

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, 43	deviceName, 24
cstInfoGetPropSize	etbld, 24
tau cstinfo.c, 76	etbTopoCnt, 24
cstList	filterAddr, 24
GNU PACKED, 23	inhibit, 25
cstOwner	isLead, 25
TRDP_CONSIST_INFO_T, 43	leadDir, 25
cstUUID	leadVehOfCst, 25
GNU PACKED, 23	lifesign, 25
cstVehNo	msgType, 25
TRDP_FUNCTION_INFO_T, 50	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
	reserved01, 28
ETB_CTRL_COMID	reserved02, 28
iec61375-2-3.h, 71	reserved03, 29
etbld	reserved04, 29
GNU PACKED, 24	reserved06, 29
TRDP_FUNCTION_INFO_T, 50	safetyTrail, 29
etbTopoCnt	serviceEntry, 29
GNU PACKED, 24	timeout, 29
	toBehav, 30
fctDev	trnCstNo, 30
service info, 37	trnDirState, 30
fctld	trnld, 30
TRDP_FUNCTION_INFO_T, 50	trnNetDir, 30
filterAddr	trnOperator, 30

trnTopoCnt, 31 trnVehNo, 31	GNU_PACKED, 27
vehld, 31	opCstList
•	GNU_PACKED, 27
vehOrient, 31	opTrnDirState
version, 31	•
h., alat 00	GNU_PACKED, 27
hp_slot, 32	opTrnTopoCnt
hp_slots, 33	GNU_PACKED, 27
INITEOC	opVehList
INITFCS	GNU_PACKED, 27
vos_utils.h, 407	ownOpCstNo
iec61375-2-3.h, 67	GNU_PACKED, 28
ETB_CTRL_COMID, 71	
TRDP_ETBCTRL_DSID, 71	PD_ELE, 34
TRDP_MAX_FILE_NAME_LEN, 72	pFrame, 35
TRDP_MAX_LABEL_LEN, 72	pFrame
TRDP_MAX_MD_DATA_SIZE, 72	PD_ELE, 35
TRDP_MAX_URI_HOST_LEN, 72	printSocketUsage
TRDP_MAX_URI_LEN, 72	trdp_utils.c, 310
TRDP_MAX_URI_USER_LEN, 72	trdp_utils.h, 323
TRDP_MD_DEFAULT_REPLY_TIMEOUT, 73	protocolVersion
TRDP MD INFINITE TIME, 73	GNU_PACKED, 28
TRDP_MIN_PD_HEADER_SIZE, 73	_ ,
TRDP_MSG_PD, 73	reserved01
TRDP_PD_UDP_PORT, 73	GNU PACKED, 28
TRDP_PROCESS_DEFAULT_CYCLE_TIME, 73	<del>-</del>
TRDP_USR_URI_SIZE, 74	GNU_PACKED, 28
TTDB_NET_DIR_REQ_COMID, 74	reserved03
TTDB_NET_DIR_INFO_COMID, 74	GNU_PACKED, 29
TTDB_OF_DIR_INFO_COMID, 74  TTDB_STAT_CST_REQ_COMID, 74	reserved04
TTDB_STAT_CST_REQ_COMID, 74  TTDB TRN DIR REQ COMID, 74	GNU_PACKED, 29
	reserved06
inhibit ONLL PACKED OF	
GNU_PACKED, 25	GNU_PACKED, 29
isLead	SOA SAME SERVICEID
GNU_PACKED, 25	trdp_serviceRegistry.h, 286
Is a dDis	
leadDir	SRM_SERVICE_READ_REQ_COMID
GNU_PACKED, 25	trdp_serviceRegistry.h, 286
leadVehOfCst	SRM_SRVINFO_NOTIFY_COMID
GNU_PACKED, 25	trdp_serviceRegistry.h, 286
lifesign	safetyTrail
GNU_PACKED, 25	GNU_PACKED, 29
_	service_info, 36
msgType	fctDev, 37
GNU_PACKED, 25	serviceEntry
	GNU_PACKED, 29
numCrcErr	serviceRegistryEntry, 37
GNU_PACKED, 26	srv_info_req, 38
numMissed	
GNU_PACKED, 26	TAU_MARSHALL_INFO_T, 38
numProtErr	TCN_URI, 39
GNU_PACKED, 26	TRDP_CLTR_CST_INFO_T, 40
numRcv	TRDP_COM_PARAM_T, 40
GNU_PACKED, 26	TRDP_COMID_DSID_MAP_T, 41
numRecv	TRDP_CONSIST_INFO_T, 41
GNU PACKED, 26	cstld, 43
numSend	cstOwner, 43
GNU PACKED, 26	TRDP_DATA_TYPE_T
numTopoErr	trdp_types.h, 305
······································	

TODO DATAGET ELEMENT T	
TRDP_DATASET_ELEMENT_T, 44	trdp_tsn_def.h, 297
TRDP_DATASET, 43	TRDP_MIN_PD_HEADER_SIZE
TRDP_DBG_CONFIG_T, 45	iec61375-2-3.h, 73
TRDP_DBG_DEFAULT	TRDP_MSG_PD
tau_xml.h, 165	iec61375-2-3.h, 73
TRDP_DNR_OPTS	TRDP_MSG_TSN_PD
tau_dnr.h, 96	trdp_tsn_def.h, 297
TRDP_DNS_REPLY, 46	TRDP_PD_CALLBACK_T
tcnUriCnt, 47	trdp_types.h, 304
TRDP_DNS_REQUEST, 47	TRDP_PD_CONFIG_T, 55
tcnUriCnt, 48	TRDP_PD_DEFAULT_QOS
TRDP_ERR_T	trdp_tsn_def.h, 297
trdp_types.h, 306	TRDP_PD_INFO_T, 56
TRDP_ETB_INFO_T, 48	TRDP_PD_UDP_PORT
cnCnt, 48	iec61375-2-3.h, 73
TRDP_ETBCTRL_DSID	TRDP_PRINT_DBG_T
iec61375-2-3.h, 71	trdp_types.h, 304
TRDP_EXCHG_OPTION_T	TRDP_PROCESS_CONFIG_T, 57
tau_xml.h, 166	TRDP_PROCESS_DEFAULT_CYCLE_TIME
TRDP_FLAGS_DEFAULT	iec61375-2-3.h, 73
trdp_types.h, 302	TRDP_PROP_T, 57
TRDP_FUNCTION_INFO_T, 49	TRDP_RED_STATE_T
cnld, 49	trdp_types.h, 307
cstVehNo, 50	TRDP_REPLY_STATUS_T
etbld, 50	trdp_types.h, 307
fctld, 50	TRDP_SDT_DEFAULT_CMTHR
TRDP_HANDLE, 50	tau_xml.c, 157
TRDP_IP_ADDR_T	TRDP_SDT_DEFAULT_LMIMAX
trdp_types.h, 303	tau_xml.c, 158
TRDP_MARSHALL_CONFIG_T, 51	TRDP_SDT_PAR_T, 58
TRDP_MARSHALL_T	TRDP_SEQ_CNT_ENTRY_T, 59
trdp_types.h, 303	TRDP_SESSION, 59
TRDP_MAX_FILE_NAME_LEN	TRDP_SOCK_TYPE_T
iec61375-2-3.h, 72	trdp_private.h, 282
TRDP_MAX_LABEL_LEN	TRDP_SOCKET_TCP, 61
iec61375-2-3.h, 72	TRDP_SOCKETS, 61
TRDP_MAX_MD_DATA_SIZE	usage, 62
iec61375-2-3.h, 72	TRDP_TIME_T
TRDP_MAX_PD_SOCKET_CNT	trdp_types.h, 304
trdp_private.h, 282	TRDP_TO_BEHAVIOR_T
TRDP_MAX_URI_HOST_LEN	trdp_types.h, 307
iec61375-2-3.h, 72	TRDP_UNMARSHALL_T
TRDP_MAX_URI_LEN	trdp_types.h, 304
iec61375-2-3.h, 72	TRDP_USR_URI_SIZE
TRDP_MAX_URI_USER_LEN	iec61375-2-3.h, 74 TRDP_VEHICLE_INFO_T, 63
iec61375-2-3.h, 72	
TRDP_MD_CALLBACK_T trdp_types.h, 303	vehld, 63 TRDP_XML_DOC_HANDLE_T, 64
TRDP_MD_CONFIG_T, 52	TTDB_NET_DIR_REQ_COMID
TRDP_MD_DEFAULT_REPLY_TIMEOUT	
iec61375-2-3.h, 73	iec61375-2-3.h, 74 TTDB_OP_DIR_INFO_COMID
TRDP_MD_ELE_ST_T	iec61375-2-3.h, 74
trdp_private.h, 282	TTDB_STAT_CST_REQ_COMID
TRDP MD INFINITE TIME	iec61375-2-3.h, 74
iec61375-2-3.h, 73	TTDB_TRN_DIR_REQ_COMID
TRDP_MD_INFO_T, 53	iec61375-2-3.h, 74
TRDP MEM CONFIG T, 54	TTI_CACHED_CONSISTS
TRDP MIN PD2 HEADER SIZE	tau_tti.c, 131

tau_DNRstatus	tau_freeTelegrams
tau_dnr.c, 91	tau_xml.c, 158
tau_dnr.h, 98	tau_xml.h, 166
tau_addServices	tau_freeXmlDatasetConfig
tau_so_if.c, 121	tau_xml.c, 158
tau_so_if.h, 126	tau_xml.h, 166
tau_addr2Uri	tau_freeXmlDoc
tau dnr.c, 91	tau_xml.c, 159
tau_dnr.h, 96	tau_xml.h, 167
tau_calcDatasetSize	tau_getCstFctCnt
tau_marshall.c, 104	tau tti.c, 131
tau_marshall.h, 111	tau_tti.h, 143
tau_calcDatasetSizeByComId	tau_getCstFctInfo
tau_marshall.c, 105	tau_tti.c, 132
tau_marshall.h, 112	tau_tti.h, 143
tau_cstinfo.c, 75	tau_getCstInfo
cstInfoGetPropSize, 76	tau_tti.c, 132
tau_ctrl.c, 77	tau_tti.h, 144
tau_getEcspStat, 79	tau_getCstVehCnt
tau_initEcspCtrl, 79	tau_tti.c, 133
tau_requestEcspConfirm, 80	tau_tti.h, 144
tau_setEcspCtrl, 80	tau_getEcspStat
tau_terminateEcspCtrl, 81	tau_ctrl.c, 79
tau_ctrl.h, 81	tau_ctrl.h, 84
tau_getEcspStat, 84	tau_getOpTrDirectory
tau_initEcspCtrl, 84	tau_tti.c, 133
tau requestEcspConfirm, 85	tau tti.h, 145
tau_setEcspCtrl, 85	tau_getOpTrnDirectoryStatusInfo
tau_terminateEcspCtrl, 86	tau_tti.c, 133
tau_ctrl_types.h, 86	tau_tti.h, 146
tau_deInitDnr	tau_getOwnAddr
tau dnr.c, 91	tau_dnr.c, 92
tau_dnr.h, 97	tau_dnr.h, 98
tau_delnitTTI	tau getOwnIds
tau_tti.c, 131	tau_gerowinds tau_tti.c, 134
_ <i>,</i>	tau_tti.h, 146
tau_tti.h, 142	<del>_</del>
tau_delServices	tau_getOwnOpCstNo
tau_so_if.b. 100	tau_tti.c, 134
tau_so_if.h, 126	tau_tti.h, 147
tau_dnr.c, 89	tau_getOwnTrnCstNo
tau_DNRstatus, 91	tau_tti.c, 135
tau_addr2Uri, 91	tau_tti.h, 147
tau_deInitDnr, 91	tau_getServiceList
tau_getOwnAddr, 92	tau_so_if.c, 123
tau_initDnr, 92	tau_so_if.h, 127
tau_uri2Addr, 93	tau_getStaticCstInfo
tau_dnr.h, 94	tau_tti.c, 135
TRDP_DNR_OPTS, 96	tau_tti.h, 147
tau_DNRstatus, 98	tau_getTTI
tau addr2Uri, 96	tau tti.c, 137
tau delnitDnr, 97	tau_tti.h, 150
tau_getOwnAddr, 98	tau_getTrDirectory
tau_initDnr, 99	tau_tti.c, 136
tau uri2Addr, 100	tau tti.h, 148
tau_dnr_types.h, 101	tau getTrnCstCnt
tau_treeServiceList	tau_tti.c, 136
tau_so_if.c, 122	tau_tti.h, 149
	<del>_</del>
tau_so_if.h, 127	tau_getTrnVehCnt

tau_tti.c, 136	tau_xml.c, 162
tau_tti.h, 149	tau_xml.h, 170
tau_getVehInfo	tau_requestEcspConfirm
tau_tti.c, 137	tau_ctrl.c, 80
tau_tti.h, 150	tau_ctrl.h, 85
tau_getVehOrient	tau_setEcspCtrl
tau_tti.c, 138	tau_ctrl.c, 80
tau_tti.h, 151	tau_ctrl.h, 85
tau_initDnr	tau_so_if.c, 119
tau_dnr.c, 92	tau_addServices, 121
tau_dnr.h, 99	tau_delServices, 122
tau_initEcspCtrl	tau_freeServiceList, 122
tau_ctrl.c, 79	tau_getServiceList, 123
tau_ctrl.h, 84	tau_updServices, 123
tau_initMarshall	tau_so_if.h, 124
tau_marshall.c, 106	tau_addServices, 126
tau_marshall.h, 113	tau_delServices, 126
tau_initTTlaccess	tau_freeServiceList, 127
tau_tti.c, 138	tau_getServiceList, 127
tau_tti.h, 152	tau_updServices, 128
tau_marshall	tau_terminateEcspCtrl
tau_marshall.c, 106	tau_ctrl.c, 81
tau_marshall.h, 114	tau_ctrl.h, 86
tau_marshall.c, 103	tau_tti.c, 128
tau_calcDatasetSize, 104	TTI_CACHED_CONSISTS, 131
tau_calcDatasetSizeByComld, 105	tau_delnitTTI, 131
tau_initMarshall, 106	tau_getCstFctCnt, 131
tau_marshall, 106	tau_getCstFctInfo, 132
tau_marshallDs, 107	tau_getCstInfo, 132
tau_unmarshall, 108	tau_getCstVehCnt, 133
tau_unmarshallDs, 108	tau_getOpTrDirectory, 133
tau_marshall.h, 109	tau_getOpTrnDirectoryStatusInfo, 133
tau_calcDatasetSizePuCamid_113	tau_getOwnOnCotNo_134
tau_calcDatasetSizeByComId, 112 tau initMarshall, 113	tau_getOwnOpCstNo, 134 tau_getOwnTrnCstNo, 135
tau_marshall, 114	tau_getStaticCstInfo, 135
tau_marshallDs, 115	tau_getTTI, 137
tau unmarshall, 116	tau_getTT, 137 tau_getTrDirectory, 136
tau unmarshallDs, 118	tau_getTrnCstCnt, 136
tau_marshallDs	tau_getTrnVehCnt, 136
tau_marshall.c, 107	tau getVehInfo, 137
tau_marshall.h, 115	tau getVehOrient, 138
tau_prepareXmIDoc	tau initTTlaccess, 138
tau xml.c, 159	tau_tti.h, 139
tau xml.h, 167	tau_deInitTTI, 142
tau_prepareXmlMem	tau_getCstFctCnt, 143
tau xml.c, 159	tau getCstFctInfo, 143
tau_xml.h, 168	tau_getCstInfo, 144
tau_readXmlDatasetConfig	tau_getCstVehCnt, 144
tau xml.c, 160	tau getOpTrDirectory, 145
tau xml.h, 168	tau_getOpTrnDirectoryStatusInfo, 146
tau_readXmlDeviceConfig	tau_getOwnlds, 146
tau_xml.c, 160	tau_getOwnOpCstNo, 147
tau xml.h, 169	tau getOwnTrnCstNo, 147
tau_readXmlInterfaceConfig	tau_getStaticCstInfo, 147
tau_xml.c, 161	tau_getTTI, 150
tau xml.h, 170	tau_getTrDirectory, 148
tau_readXmlServiceConfig	tau_getTrnCstCnt, 149
3	

tau_getTrnVehCnt, 149	tlc_getJoinStatistics
tau_getVehInfo, 150	trdp_if_light.h, 215
tau_getVehOrient, 151	trdp_stats.c, 288
tau_initTTlaccess, 152	tlc_getOpTrainTopoCount
tau_tti_types.h, 153	tlc_if.c, 175
tau_unmarshall	trdp_if_light.h, 215
tau_marshall.c, 108	tlc_getOwnlpAddress
tau_marshall.h, 116	tlc_if.c, 176
tau_unmarshallDs	trdp_if_light.h, 216
tau_marshall.c, 108	tlc_getPubStatistics
tau_marshall.h, 118	trdp_if_light.h, 216
tau_updServices	trdp_stats.c, 289
tau_so_if.c, 123	tlc_getRedStatistics
tau_so_if.h, 128	trdp_if_light.h, 217
tau_uri2Addr	trdp_stats.c, 289
tau_dnr.c, 93	tlc_getStatistics
tau_dnr.h, 100	trdp_if_light.h, 217
tau_xml.c, 156	trdp_stats.c, 290
TRDP_SDT_DEFAULT_CMTHR, 157	tlc_getSubsStatistics
TRDP_SDT_DEFAULT_LMIMAX, 158	trdp_if_light.h, 218
tau_freeTelegrams, 158	trdp_stats.c, 290
tau_freeXmlDatasetConfig, 158	tlc_getVersion
tau_freeXmlDoc, 159	tlc_if.c, 176
tau_prepareXmlDoc, 159	trdp_if_light.h, 218
tau_prepareXmlMem, 159	tlc_getVersionString
tau_readXmlDatasetConfig, 160	tlc_if.c, 176
tau_readXmlDeviceConfig, 160	trdp_if_light.h, 218
tau_readXmlInterfaceConfig, 161	tlc_if.c, 171
tau_readXmlServiceConfig, 162	tlc_closeSession, 173
tau_xml.h, 162	tlc_configSession, 174
TRDP_DBG_DEFAULT, 165	tlc_getETBTopoCount, 174
TRDP_EXCHG_OPTION_T, 166	tlc_getInterval, 175
tau_freeTelegrams, 166	tlc_getOpTrainTopoCount, 175
tau_freeXmlDatasetConfig, 166	tlc_getOwnIpAddress, 176
tau_freeXmlDoc, 167	tlc_getVersion, 176
tau_prepareXmlDoc, 167	tlc_getVersionString, 176
tau_prepareXmlMem, 168	tlc_init, 177
tau_readXmlDatasetConfig, 168	tlc_openSession, 177
tau_readXmlDeviceConfig, 169	tlc_process, 178
tau_readXmlInterfaceConfig, 170	tlc_reinitSession, 178
tau_readXmlServiceConfig, 170	tlc_setETBTopoCount, 179
tcnUriCnt	tlc_setOpTrainTopoCount, 179
TRDP_DNS_REPLY, 47	tlc_terminate, 181
TRDP_DNS_REQUEST, 48	tlc_updateSession, 181
timeout	trdp_getAccess, 182
GNU_PACKED, 29	trdp_isValidSession, 182
tlc_closeSession	trdp_releaseAccess, 182
tlc_if.c, 173	trdp_sessionQueue, 183
trdp_if_light.h, 213	tlc_if.h, 183
tlc_configSession	trdp_isValidSession, 185
tlc_if.c, 174	trdp_sessionQueue, 185
trdp_if_light.h, 213	tlc_init
tlc_getETBTopoCount	tlc_if.c, 177
tlc_if.c, 174	trdp_if_light.h, 219
trdp_if_light.h, 214	tlc_openSession
tlc_getInterval	tlc_if.c, 177
tlc_if.c, 175	trdp_if_light.h, 219
trdp_if_light.h, 214	tlc_process

11 17 470	
tlc_if.c, 178	trdp_if_light.h, 229
trdp_if_light.h, 220	tlm_replyQuery
tlc_reinitSession	tlm_if.c, 194
tlc_if.c, 178	trdp_if_light.h, 230
trdp_if_light.h, 221	tlm_request
tlc_resetStatistics	tlm_if.c, 195
trdp_if_light.h, 221	trdp_if_light.h, 231
trdp_stats.c, 291	tlp_get
tlc_setETBTopoCount	tlp_if.c, 198
tlc_if.c, 179	trdp_if_light.h, 232
trdp_if_light.h, 221	tlp_getInterval
tlc_setOpTrainTopoCount	tlp_if.c, 199
tlc_if.c, 179	trdp_if_light.h, 233
trdp_if_light.h, 222	tlp_getRedundant
tlc_terminate	tlp_if.c, 200
tlc_if.c, 181	trdp_if_light.h, 234
trdp_if_light.h, 222	tlp_if.c, 196
tlc_updateSession	tlp_get, 198
tlc_if.c, 181	tlp_getInterval, 199
trdp_if_light.h, 223	tlp_getRedundant, 200
tlm_abortSession	tlp_processReceive, 200
tlm_if.c, 188	tlp processSend, 201
trdp_if_light.h, 223	tlp_publish, 201
tlm addListener	tlp_put, 202
_ tlm_if.c, 188	tlp_putImmediate, 203
trdp_if_light.h, 224	tlp_republish, 203
tlm confirm	tlp_request, 204
tlm if.c, 189	tlp_resubscribe, 205
trdp_if_light.h, 225	tlp_setRedundant, 206
tlm_delListener	tlp_subscribe, 206
tlm if.c, 190	tlp_unpublish, 207
trdp_if_light.h, 225	tlp_unsubscribe, 208
tlm_getInterval	tlp processReceive
tlm if.c, 190	tlp_if.c, 200
trdp_if_light.h, 227	trdp_if_light.h, 234
tlm if.c, 186	tlp processSend
tlm_abortSession, 188	tlp_if.c, 201
tlm_addListener, 188	trdp_if_light.h, 235
tlm_confirm, 189	tlp_publish
tlm_delListener, 190	tlp if.c, 201
tlm_getInterval, 190	trdp_if_light.h, 235
tlm_notify, 191	tlp_put
tlm process, 192	tlp if.c, 202
tlm_readdListener, 192	trdp_if_light.h, 236
	· — — •
tlm_reply, 193	tlp_putImmediate tlp_if.c, 203
tlm_replyQuery, 194	• —
tlm_request, 195	trdp_if_light.h, 237
tlm_notify	tlp_republish
tlm_if.c, 191	tlp_if.c, 203
trdp_if_light.h, 227	trdp_if_light.h, 237
tlm_process	tlp_request
tlm_if.c, 192	tlp_if.c, 204
trdp_if_light.h, 228	trdp_if_light.h, 238
tlm_readdListener	tlp_resubscribe
tlm_if.c, 192	tlp_if.c, 205
trdp_if_light.h, 229	trdp_if_light.h, 239
tlm_reply	tlp_setRedundant
tlm_if.c, 193	tlp_if.c, 206

trdp_if_light.h, 240	trdp_utils.c, 311
tlp_subscribe	trdp_utils.h, 324
tlp_if.c, 206	trdp_findSubAddr
trdp_if_light.h, 240	trdp_utils.c, 311
tlp_unpublish	trdp_utils.h, 324
tlp_if.c, 207	trdp_getAccess
trdp_if_light.h, 241	tlc_if.c, 182
tlp_unsubscribe	trdp_getSeqCnt
tlp_if.c, 208	trdp_utils.c, 312
trdp_if_light.h, 242	trdp_utils.h, 325
toBehav	trdp_if_light.h, 209
GNU_PACKED, 30	tlc_closeSession, 213
trdp_SockAddJoin	tlc_configSession, 213
trdp_utils.c, 319	tlc_getETBTopoCount, 214
trdp_utils.h, 332	tlc_getInterval, 214
trdp_SockDelJoin	tlc_getJoinStatistics, 215
trdp_utils.c, 319	tlc_getOpTrainTopoCount, 215
trdp_utils.h, 332	tlc_getOwnIpAddress, 216
trdp_SockIsJoined	tlc_getPubStatistics, 216
trdp_utils.c, 319	tlc_getRedStatistics, 217
trdp_utils.h, 332	tlc_getStatistics, 217
trdp_UpdateStats	tlc_getSubsStatistics, 218
trdp_stats.c, 292	tlc_getVersion, 218
trdp_XMLClose	tlc_getVersionString, 218
trdp_xml.c, 335	tlc_init, 219
trdp_xml.h, 341	tlc_openSession, 219
trdp_XMLCountStartTag	tlc_process, 220
trdp_xml.c, 335	tlc_reinitSession, 221
trdp_xml.h, 342	tlc_resetStatistics, 221
trdp_XMLEnter	tlc_setETBTopoCount, 221
trdp_xml.c, 336	tlc_setOpTrainTopoCount, 222
trdp_xml.h, 342	tlc_terminate, 222
trdp_XMLGetAttribute	tlc_updateSession, 223
trdp_xml.c, 336	tlm_abortSession, 223
trdp_xml.h, 342	tlm_addListener, 224
trdp_XMLLeave	tlm_confirm, 225
trdp_xml.c, 337	tlm_delListener, 225
trdp_xml.h, 343	tlm_getInterval, 227
trdp_XMLMemOpen	tlm_notify, 227
trdp_xml.c, 337	tlm_process, 228
trdp_xml.h, 343	tlm_readdListener, 229
trdp_XMLOpen	tlm_reply, 229
trdp_xml.c, 337	tlm_replyQuery, 230
trdp_xml.h, 344	tlm_request, 231
trdp_XMLRewind	tlp_get, 232
trdp_xml.c, 338	tlp_getInterval, 233
trdp_xml.h, 344	tlp_getRedundant, 234
trdp_XMLSeekStartTag	tlp_processReceive, 234
trdp_xml.c, 338	tlp_processSend, 235
trdp_xml.h, 344	tlp_publish, 235
trdp_XMLSeekStartTagAny	tlp_put, 236
trdp_xml.c, 339	tlp_putImmediate, 237
trdp_xml.h, 345	tlp_republish, 237
trdp_checkSequenceCounter	tlp_request, 238
trdp_utils.c, 310	tlp_resubscribe, 239
trdp_utils.h, 323	tlp_setRedundant, 240
trdp_dllmain.c, 208	tlp_subscribe, 240
trdp_findMCjoins	tlp_unpublish, 241

tlp_unsubscribe, 242	trdp_mdConfirm, 253
trdp_initSockets	trdp_mdFreeSession, 253
trdp_utils.c, 312	trdp_mdGetTCPSocket, 254
trdp_utils.h, 325	trdp_mdReply, 254
trdp_initStats	trdp_mdSend, 255
trdp_stats.c, 291	trdp_packetSizeMD
trdp_stats.h, 294	trdp_utils.c, 313
trdp_isAddressed	trdp_utils.h, 326
trdp_utils.c, 313	trdp_packetSizePD
trdp_utils.h, 325	trdp_utils.c, 314
trdp_isInIPrange	trdp_utils.h, 327
trdp_utils.c, 313	trdp_pdCheck
trdp_utils.h, 326	trdp_pdcom.c, 257
trdp_isValidSession	trdp_pdcom.h, 267
tlc_if.c, 182	trdp_pdCheckListenSocks
tlc_if.h, 185	trdp_pdcom.c, 258
trdp_mdCall	trdp_pdcom.h, 268
trdp_mdcom.c, 244	trdp_pdCheckPending
trdp_mdcom.h, 251	trdp_pdcom.c, 258
trdp_mdCheckListenSocks	trdp_pdcom.h, 268
trdp_mdcom.c, 245	trdp_pdDistribute
trdp_mdcom.h, 252	trdp_pdcom.c, 258
trdp_mdCheckPending	trdp_pdcom.h, 269
trdp_mdcom.c, 245	trdp_pdHandleTimeOuts
trdp_mdcom.h, 252	trdp_pdcom.c, 259
trdp_mdCheckTimeouts	trdp_pdcom.h, 269
trdp_mdcom.c, 246	trdp_pdInit
trdp_mdcom.h, 253	trdp_pdcom.c, 260
trdp_mdConfirm	trdp_pdcom.h, 270
trdp_mdcom.c, 246	trdp_pdPrepareStats
trdp_mdcom.h, 253	trdp_stats.c, 292
trdp_mdFreeSession	trdp_stats.h, 295
trdp_mdcom.c, 247	trdp_pdPut
trdp_mdcom.h, 253	trdp_pdcom.c, 261
trdp_mdGetTCPSocket	trdp_pdcom.h, 271
trdp_mdcom.c, 247	trdp_pdReceive
trdp_mdcom.h, 254	trdp_pdcom.c, 261
trdp_mdReply	trdp_pdcom.h, 271
trdp_mdcom.c, 248	trdp_pdSend
trdp_mdcom.h, 254	trdp_pdcom.c, 262
trdp_mdSend	trdp_pdcom.h, 272
trdp_mdcom.c, 248	trdp_pdSendElement
trdp_mdcom.h, 255	trdp_pdcom.c, 262
trdp_mdcom.c, 242	trdp_pdcom.h, 272
trdp_mdChadd istanSada 245	trdp_pdSendImmediate
trdp_mdCheckListenSocks, 245	trdp_pdcom.c, 263
trdp_mdCheckPending, 245	trdp_pdcom.h, 273
trdp_mdCheckTimeouts, 246	trdp_pdSendQueued
trdp_mdConfirm, 246	trdp_pdcom.c, 264
trdp_mdFreeSession, 247	trdp_pdcom.h, 274
trdp_mdGetTCPSocket, 247	trdp_pdUpdate
trdp_mdReply, 248	trdp_pdcom.c, 264
trdp_mdSend, 248	trdp_pdcom.h, 274
trdp_mdColl_351	trdp_pdcom.c, 255
trdp_mdCall, 251 trdp_mdCheckListenSocks, 252	trdp_pdCheck, 257 trdp_pdCheckListenSocks, 258
•	• —•
trdp_mdCheckPending, 252 trdp_mdCheckTimeouts, 253	trdp_pdCheckPending, 258 trdp_pdDistribute, 258
trap_maoneok mineouts, 200	trup_publistribute, 200

	trdp_pdHandleTimeOuts, 259	trdp	resetSequenceCounter
	trdp_pdInit, 260		trdp_utils.c, 318
	trdp_pdPut, 261		trdp_utils.h, 331
	trdp_pdReceive, 261	trdp	_serviceRegistry.h, 283
	trdp_pdSend, 262		SOA_SAME_SERVICEID, 286
	trdp_pdSendElement, 262		SRM_SERVICE_READ_REQ_COMID, 286
	trdp_pdSendImmediate, 263		SRM_SRVINFO_NOTIFY_COMID, 286
	trdp_pdSendQueued, 264	trdp	_sessionQueue
	trdp_pdUpdate, 264		tlc_if.c, 183
trdp_	_pdcom.h, 265		tlc_if.h, 185
	trdp_pdCheck, 267	trdp	_stats.c, 287
	trdp_pdCheckListenSocks, 268		tlc_getJoinStatistics, 288
	trdp_pdCheckPending, 268		tlc_getPubStatistics, 289
	trdp_pdDistribute, 269		tlc_getRedStatistics, 289
	trdp_pdHandleTimeOuts, 269		tlc_getStatistics, 290
	trdp_pdInit, 270		tlc_getSubsStatistics, 290
	trdp_pdPut, 271		tlc_resetStatistics, 291
	trdp_pdReceive, 271		trdp_UpdateStats, 292
	trdp_pdSend, 272		trdp_initStats, 291
	trdp_pdSendElement, 272		trdp_pdPrepareStats, 292
	trdp_pdSendImmediate, 273	trdp	_stats.h, 293
	trdp_pdSendQueued, 274		trdp_initStats, 294
	trdp_pdUpdate, 274		trdp_pdPrepareStats, 295
trdp_	_pdindex.c, 275	trdp	_tsndef.h, 296
trdp_	_pdindex.h, 277		TRDP_MIN_PD2_HEADER_SIZE, 297
trdp_	_private.h, 279		TRDP_MSG_TSN_PD, 297
	TRDP_MAX_PD_SOCKET_CNT, 282		TRDP_PD_DEFAULT_QOS, 297
	TRDP_MD_ELE_ST_T, 282	trdp	_types.h, 297
	TRDP_SOCK_TYPE_T, 282		TRDP_DATA_TYPE_T, 305
trdp_	_queueAppLast		TRDP_ERR_T, 306
	trdp_utils.c, 314		TRDP_FLAGS_DEFAULT, 302
	trdp_utils.h, 327		TRDP_IP_ADDR_T, 303
trdp_	_queueDelElement		TRDP_MARSHALL_T, 303
	trdp_utils.c, 314		TRDP_MD_CALLBACK_T, 303
	trdp_utils.h, 327		TRDP_PD_CALLBACK_T, 304
trdp_	_queueFindComId		TRDP_PRINT_DBG_T, 304
	trdp_utils.c, 315		TRDP_RED_STATE_T, 307
	trdp_utils.h, 328		TRDP_REPLY_STATUS_T, 307
trdp_	_queueFindExistingSub		TRDP_TIME_T, 304
	trdp_utils.c, 315		TRDP_TO_BEHAVIOR_T, 307
	trdp_utils.h, 328		TRDP_UNMARSHALL_T, 304
trdp_	_queueFindPubAddr	trdp	_utils.c, 308
	trdp_utils.c, 315		printSocketUsage, 310
	trdp_utils.h, 328		trdp_SockAddJoin, 319
trdp_	_queueFindSubAddr		trdp_SockDelJoin, 319
	trdp_utils.c, 316		trdp_SockIsJoined, 319
	trdp_utils.h, 329		trdp_checkSequenceCounter, 310
trdp_	_queueInsFirst		trdp_findMCjoins, 311
	trdp_utils.c, 316		trdp_findSubAddr, 311
	trdp_utils.h, 329		trdp_getSeqCnt, 312
trdp_	_releaseAccess		trdp_initSockets, 312
	tlc_if.c, 182		trdp_isAddressed, 313
trdp_	_releaseSocket		trdp_isInIPrange, 313
	trdp_utils.c, 317		trdp_packetSizeMD, 313
	trdp_utils.h, 330		trdp_packetSizePD, 314
trdp_	_requestSocket		trdp_queueAppLast, 314
	trdp_utils.c, 317		trdp_queueDelElement, 314
	trdp_utils.h, 330		trdp_queueFindComId, 315

	trdp_queueFindExistingSub, 315	trnCstNo
	trdp_queueFindPubAddr, 315	GNU_PACKED, 30
	trdp_queueFindSubAddr, 316	trnDirState
	trdp_queueInsFirst, 316	GNU_PACKED, 30
	trdp_releaseSocket, 317	trnld
	trdp requestSocket, 317	GNU_PACKED, 30
	trdp_resetSequenceCounter, 318	trnNetDir
	trdp_validTopoCounters, 320	GNU_PACKED, 30
	utils.h, 320	trnOperator
trup_	printSocketUsage, 323	GNU_PACKED, 30
	•	trnTopoCnt
	trdp_SockAddJoin, 332	GNU PACKED, 31
	trdp_SockDelJoin, 332	trnVehNo
	trdp_SockIsJoined, 332	
	trdp_checkSequenceCounter, 323	GNU_PACKED, 31
	trdp_findMCjoins, 324	Heado
	trdp_findSubAddr, 324	usage
	trdp_getSeqCnt, 325	TRDP_SOCKETS, 62
	trdp_initSockets, 325	VOS ERR T
	trdp_isAddressed, 325	
	trdp isInIPrange, 326	vos_types.h, 400
	trdp packetSizeMD, 326	VOS_LOG_T
	trdp_packetSizePD, 327	vos_types.h, 400
	trdp_queueAppLast, 327	VOS_MAX_ERR_STR_SIZE
	trdp_queueDelElement, 327	vos_utils.h, 407
	trdp_queueFindComId, 328	VOS_MAX_FRMT_SIZE
		vos_utils.h, 407
	trdp_queueFindExistingSub, 328	VOS_MAX_PRNT_STR_SIZE
	trdp_queueFindPubAddr, 328	vos_utils.h, 407
	trdp_queueFindSubAddr, 329	VOS_MAX_SOCKET_CNT
	trdp_queueInsFirst, 329	vos_sock.h, 370
	trdp_releaseSocket, 330	VOS_MEM_BLOCKSIZES
	trdp_requestSocket, 330	vos_mem.h, 357
	trdp_resetSequenceCounter, 331	VOS_MEM_PREALLOCATE
	trdp_validTopoCounters, 333	vos_mem.h, 357
trdp_	validTopoCounters	VOS PRINT DBG T
	trdp_utils.c, 320	vos_types.h, 399
	trdp_utils.h, 333	VOS_SOCK_OPT_T, 64
	xml.c, 333	VOS_TIMEVAL_T
	trdp_XMLClose, 335	
	trdp_XMLCountStartTag, 335	vos_types.h, 399
	trdp_XMLEnter, 336	VOS_TTL_MULTICAST
	trdp_XMLGetAttribute, 336	vos_sock.h, 370
	• —	VOS_VERSION_T, 65
	trdp_XMLLeave, 337	vehld
	trdp_XMLMemOpen, 337	GNU_PACKED, 31
	trdp_XMLOpen, 337	TRDP_VEHICLE_INFO_T, 63
	trdp_XMLRewind, 338	vehOrient
	trdp_XMLSeekStartTag, 338	GNU_PACKED, 31
	trdp_XMLSeekStartTagAny, 339	version
trdp_	_xml.h, 339	GNU_PACKED, 31
	trdp_XMLClose, 341	vos_addTime
	trdp_XMLCountStartTag, 342	vos_thread.h, 387
	trdp_XMLEnter, 342	vos_bsearch
	trdp_XMLGetAttribute, 342	vos_mem.c, 347
	trdp_XMLLeave, 343	vos_mem.h, 357
	trdp_XMLMemOpen, 343	vos_clearTime
	trdp_XMLOpen, 344	vos_thread.h, 387
	trdp_XMLRewind, 344	vos_cmpTime
	• —	
	trdp_XMLSeekStartTag, 344	vos_thread.h, 387
	trdp_XMLSeekStartTagAny, 345	vos_crc32

vos_utils.c, 402	vos_mem.h, 355
vos_utils.h, 408	VOS_MEM_BLOCKSIZES, 357
vos_determineBindAddr	VOS_MEM_PREALLOCATE, 357
vos_sock.h, 370	vos_bsearch, 357
vos_divTime	vos_memAlloc, 358
vos_thread.h, 388	vos_memCount, 358
vos_dottedIP	vos_memDelete, 359
vos_sock.h, 371	vos_memFree, 359
vos_getErrorString	vos_memInit, 360
vos_utils.c, 402	vos_qsort, 360
vos_utils.h, 409	vos_queueCreate, 361
vos_getInterfaces	vos_queueDestroy, 361
vos_sock.h, 371	vos_queueReceive, 362
vos_getRealTime	vos_queueSend, 362
vos_thread.h, 388	vos_strncat, 363
vos_getTime	vos_strncpy, 363
vos_thread.h, 388	vos_strnicmp, 364
vos_getTimeStamp	vos_memAlloc
vos_thread.h, 389	vos_mem.c, 348
vos_getUuid	vos_mem.h, 358
vos_thread.h, 389	vos_memCount
vos_getVersion	vos_mem.c, 348
vos_utils.c, 403	vos_mem.h, 358
vos_utils.h, 409	vos_memDelete
vos_getVersionString	vos_mem.c, 349
vos_utils.c, 403	vos_mem.h, 359
vos_utils.h, 409	vos_memFree
vos_hostlsBigEndian vos utils.c, 403	vos_mem.c, 349 vos_mem.h, 359
vos_utils.t., 403 vos_utils.h, 410	vos memInit
vos htonl	vos_mem.c, 349
vos_sock.h, 371	vos_mem.h, 360
vos htonii	vos mulTime
vos_sock.h, 372	vos_thread.h, 389
vos htons	vos_mutexCreate
vos_sock.h, 372	vos_thread.h, 389
vos_init	vos mutexDelete
vos utils.c, 404	vos thread.h, 390
vos utils.h, 410	vos mutexLock
vos ipDotted	vos thread.h, 390
vos sock.h, 373	vos mutexTryLock
vos isMulticast	vos thread.h, 391
vos sock.h, 373	vos mutexUnlock
vos mem.c. 345	vos thread.h, 391
vos bsearch, 347	vos netIfUp
vos memAlloc, 348	vos_sock.h, 373
vos_memCount, 348	vos_ntohl
vos memDelete, 349	vos_sock.h, 374
vos_memFree, 349	vos_ntohll
vos memInit, 349	vos sock.h, 374
vos gsort, 350	vos ntohs
vos queueCreate, 350	vos sock.h, 374
vos_queueDestroy, 351	vos gsort
vos queueReceive, 351	vos mem.c, 350
vos_queueSend, 352	vos_mem.h, 360
vos_strncat, 352	vos_queueCreate
vos strncpy, 354	vos mem.c, 350
vos_strnicmp, 354	vos_mem.h, 361
_ ' '	_ ′

vos_queueDestroy	vos_sockSendTCP, 382
vos_mem.c, 351	vos_sockSendUDP, 382
vos_mem.h, 361	vos_sockSetMulticastIf, 383
vos_queueReceive	vos_sockSetOptions, 383
vos_mem.c, 351	vos_sockTerm, 384
vos_mem.h, 362	vos_sockAccept
vos_queueSend	vos_sock.h, 375
vos_mem.c, 352	vos_sockBind
vos_mem.h, 362	vos_sock.h, 376
vos_sc32	vos_sockClose
vos_utils.c, 404	vos_sock.h, 376
vos_utils.h, 411	vos_sockConnect
vos_select	vos_sock.h, 377
vos_sock.h, 375	vos_sockGetMAC
vos_semaCreate	vos_sock.h, 377
vos_thread.h, 391	vos_sockInit
vos_semaDelete	vos_sock.h, 378
vos_thread.h, 392	vos_sockJoinMC
vos_semaGive	vos_sock.h, 378
vos_thread.h, 392	vos_sockLeaveMC
vos_semaTake	vos_sock.h, 378
vos_thread.h, 393	vos_sockListen
vos_shared_mem.h, 364	vos_sock.h, 379
vos_sharedClose, 366	vos_sockOpenTCP
vos_sharedOpen, 366	vos_sock.h, 379
vos_sharedClose	vos_sockOpenUDP
vos_shared_mem.h, 366	vos_sock.h, 380
vos_sharedOpen	vos_sockReceiveTCP
vos_shared_mem.h, 366	vos_sock.h, 380
vos_sock.h, 367	vos_sockReceiveUDP
VOS_MAX_SOCKET_CNT, 370	vos_sock.h, 381
VOS_TTL_MULTICAST, 370	vos_sockSendTCP
vos_determineBindAddr, 370	vos_sock.h, 382
vos_dottedIP, 371	vos_sockSendUDP
vos_getInterfaces, 371	vos_sock.h, 382 vos_sockSetMulticastIf
vos_htonl, 371	
vos_htonll, 372 vos htons, 372	vos_sock.h, 383 vos_sockSetOptions
vos_fitoris, 372 vos_ipDotted, 373	vos_sock.h, 383
vos_ipDotted, 373 vos_isMulticast, 373	vos_sockTerm
vos netlfUp, 373	vos_sock.h, 384
vos ntohl, 374	vos_sock.ii, 364 vos strncat
vos_ntohll, 374	vos mem.c, 352
vos ntohs, 374	vos mem.h, 363
vos select, 375	vos_niem.n, 303
vos sockAccept, 375	vos_mem.c, 354
vos sockBind, 376	vos_mem.h, 363
vos sockClose, 376	vos_strnicmp
vos sockConnect, 377	vos_mem.c, 354
vos sockGetMAC, 377	vos mem.h, 364
vos socklnit, 378	vos subTime
vos sockJoinMC, 378	vos_thread.h, 393
100 000100111110, 010	
<del>-</del>	
vos_sockLeaveMC, 378	vos_terminate
vos_sockLeaveMC, 378 vos_sockListen, 379	vos_terminate vos_utils.c, 405
vos_sockLeaveMC, 378 vos_sockListen, 379 vos_sockOpenTCP, 379	vos_terminate vos_utils.c, 405 vos_utils.h, 411
vos_sockLeaveMC, 378 vos_sockListen, 379 vos_sockOpenTCP, 379 vos_sockOpenUDP, 380	vos_terminate vos_utils.c, 405 vos_utils.h, 411 vos_thread.h, 384
vos_sockLeaveMC, 378 vos_sockListen, 379 vos_sockOpenTCP, 379	vos_terminate vos_utils.c, 405 vos_utils.h, 411

	vos_cmpTime, 387	VOS_MAX_FRMT_SIZE, 407
	vos_divTime, 388	VOS_MAX_PRNT_STR_SIZE, 407
	vos_getRealTime, 388	vos_crc32, 408
	vos_getTime, 388	vos_getErrorString, 409
	vos_getTimeStamp, 389	vos_getVersion, 409
	vos_getUuid, 389	vos_getVersionString, 409
	vos_mulTime, 389	vos_hostIsBigEndian, 410
	vos_mutexCreate, 389	vos_init, 410
	vos_mutexDelete, 390	vos_sc32, 411
	vos_mutexLock, 390	vos_terminate, 411
	vos_mutexTryLock, 391	
	vos_mutexUnlock, 391	
	vos_semaCreate, 391	
	vos semaDelete, 392	
	vos_semaGive, 392	
	vos_semaTake, 393	
	vos_subTime, 393	
	vos threadCreate, 393	
	vos_threadCreateSync, 394	
	vos_threadDelay, 395	
	vos threadInit, 395	
	vos_threadIsActive, 396	
	vos threadSelf, 396	
	vos threadTerm, 396	
	vos_threadTerminate, 396	
vos	threadCreate	
	vos thread.h, 393	
vos	threadCreateSync	
	vos_thread.h, 394	
vos	threadDelay	
_	vos thread.h, 395	
vos	threadInit	
_	vos thread.h, 395	
vos	threadIsActive	
	vos_thread.h, 396	
vos	threadSelf	
	vos_thread.h, 396	
vos	threadTerm	
	vos_thread.h, 396	
vos	threadTerminate	
	vos_thread.h, 396	
vos	types.h, 397	
	VOS_ERR_T, 400	
	VOS LOG T, 400	
	VOS_PRINT_DBG_T, 399	
	VOS_TIMEVAL_T, 399	
vos	utils.c, 401	
	vos_crc32, 402	
	vos_getErrorString, 402	
	vos_getVersion, 403	
	vos_getVersionString, 403	
	vos_hostIsBigEndian, 403	
	vos_init, 404	
	vos sc32, 404	
	vos_terminate, 405	
VOS	utils.h, 405	
.00_	INITFCS, 407	
	VOS_MAX_ERR_STR_SIZE, 407	
	. 55 5(_E (_5)115(EE, 10)	