TCNOpen TRDP Light V2.0.3

Generated by Doxygen 1.8.18

1 The TRDP Light Library API Specification	1
1.1 General Information	1
1.1.1 Purpose	1
1.1.2 Scope	1
1.1.3 Related documents	1
1.1.4 Abbreviations and Definitions	1
1.2 Terminology	2
1.3 Use Cases	2
1.4 Conventions of the API	5
2 Data Structure Index	7
2.1 Data Structures	7
3 File Index	9
3.1 File List	9
4 Data Structure Documentation	11
4.1 DNS_HEADER Struct Reference	11
4.1.1 Detailed Description	11
4.2 GNU_PACKED Struct Reference	11
4.2.1 Detailed Description	19
4.2.2 Field Documentation	20
4.2.2.1 callBack	20
4.2.2.2 comld	20
4.2.2.3 confVehCnt	20
4.2.2.4 confVehList	20
4.2.2.5 cstCnt	20
4.2.2.6 cstList	21
4.2.2.7 cstUUID	21
4.2.2.8 defQos	21
4.2.2.9 defTtl	21
4.2.2.10 deviceName	22
4.2.2.11 etbld	22
4.2.2.12 etbInhibit	22
4.2.2.13 etbLength	22
4.2.2.14 etbShort	22
4.2.2.15 etbTopoCnt	23
4.2.2.16 inhibit	23
4.2.2.17 isLead	23
4.2.2.18 joinedAddr	23
4.2.2.19 leadDir	23
4.2.2.20 leadVehOfCst	24
4.2.2.21 numCrcErr	24

4.2.2.22 numMissed	24
4.2.2.23 numProtErr	24
4.2.2.24 numRcv	24
4.2.2.25 numRecv	24
4.2.2.26 numSend	25
4.2.2.27 numTopoErr	25
4.2.2.28 opCstList	25
4.2.2.29 opTrnDirState	25
4.2.2.30 opTrnTopoCnt	25
4.2.2.31 opVehList	26
4.2.2.32 ownOpCstNo	26
4.2.2.33 reserved01 [1/2]	26
4.2.2.34 reserved01 [2/2]	26
4.2.2.35 reserved 02 [1/2]	26
4.2.2.36 reserved02 [2/2]	27
4.2.2.37 reserved03	27
4.2.2.38 reserved04	27
4.2.2.39 reserved06	27
4.2.2.40 safetyTrail	27
4.2.2.41 serviceEntry	28
4.2.2.42 timeout	28
4.2.2.43 toBehav	28
4.2.2.44 trnCstNo	28
4.2.2.45 trnDirState	28
4.2.2.46 trnld	29
4.2.2.47 trnNetDir	29
4.2.2.48 trnOperator	29
4.2.2.49 trnTopoCnt	29
4.2.2.50 trnVehNo	29
4.2.2.51 vehld	29
4.2.2.52 vehOrient	30
4.2.2.53 version	30
4.3 service_info Struct Reference	30
4.3.1 Detailed Description	31
4.3.2 Field Documentation	31
4.3.2.1 fctDev	31
4.4 srv_info_req Struct Reference	32
4.4.1 Detailed Description	32
4.5 TAU_MARSHALL_INFO_T Struct Reference	32
4.5.1 Detailed Description	33
4.6 TCN_URI Struct Reference	33
4.6.1 Detailed Description	33

4.7 TRDP_CLTR_CST_INFO_T Struct Reference
4.7.1 Detailed Description
4.8 TRDP_COM_PARAM_T Struct Reference
4.8.1 Detailed Description
4.9 TRDP_COMID_DSID_MAP_T Struct Reference
4.9.1 Detailed Description
4.10 TRDP_CONSIST_INFO_T Struct Reference
4.10.1 Detailed Description
4.10.2 Field Documentation
4.10.2.1 cstld
4.10.2.2 cstOwner
4.11 TRDP_DATASET Struct Reference
4.11.1 Detailed Description
4.12 TRDP_DATASET_ELEMENT_T Struct Reference
4.12.1 Detailed Description
4.13 TRDP_DBG_CONFIG_T Struct Reference
4.13.1 Detailed Description
4.14 TRDP_DNS_REPLY Struct Reference
4.14.1 Detailed Description
4.14.2 Field Documentation
4.14.2.1 tcnUriCnt
4.15 TRDP_DNS_REQUEST Struct Reference
4.15.1 Detailed Description
4.15.2 Field Documentation
4.15.2.1 tcnUriCnt
4.16 TRDP_ETB_INFO_T Struct Reference
4.16.1 Detailed Description
4.16.2 Field Documentation
4.16.2.1 cnCnt
4.17 TRDP_FUNCTION_INFO_T Struct Reference
4.17.1 Detailed Description
4.17.2 Field Documentation
4.17.2.1 cnld
4.17.2.2 cstVehNo
4.17.2.3 etbld
4.17.2.4 fctld
4.18 TRDP_IDX_TABLE_T Struct Reference
4.18.1 Detailed Description
4.18.2 Field Documentation
4.18.2.1 maxNoOfExtPublishers
4.18.2.2 maxNoOfHighCatPublishers
4.18.2.3 maxNoOfHighCatSubscriptions

4.18.2.4 maxNoOfLowCatPublishers	46
4.18.2.5 maxNoOfLowCatSubscriptions	46
4.18.2.6 maxNoOfMidCatPublishers	46
4.18.2.7 maxNoOfMidCatSubscriptions	47
4.19 TRDP_MARSHALL_CONFIG_T Struct Reference	47
4.19.1 Detailed Description	47
4.20 TRDP_MD_CONFIG_T Struct Reference	48
4.20.1 Detailed Description	49
4.21 TRDP_MD_INFO_T Struct Reference	49
4.21.1 Detailed Description	50
4.22 TRDP_MEM_CONFIG_T Struct Reference	50
4.22.1 Detailed Description	51
4.23 TRDP_PD_CONFIG_T Struct Reference	51
4.23.1 Detailed Description	52
4.24 TRDP_PD_INFO_T Struct Reference	52
4.24.1 Detailed Description	53
4.25 TRDP_PROCESS_CONFIG_T Struct Reference	54
4.25.1 Detailed Description	54
4.26 TRDP_PROP_T Struct Reference	54
4.26.1 Detailed Description	55
4.27 TRDP_SDT_PAR_T Struct Reference	55
4.27.1 Detailed Description	55
4.28 TRDP_VEHICLE_INFO_T Struct Reference	56
4.28.1 Detailed Description	56
4.28.2 Field Documentation	56
4.28.2.1 vehld	57
4.29 TRDP_XML_DOC_HANDLE_T Struct Reference	57
4.29.1 Detailed Description	57
4.30 VOS_SOCK_OPT_T Struct Reference	57
4.30.1 Detailed Description	58
4.31 VOS_VERSION_T Struct Reference	58
4.31.1 Detailed Description	59
5 File Documentation	61
5.1 iec61375-2-3.h File Reference	_
5.1.1 Detailed Description	
5.1.2 Macro Definition Documentation	
5.1.2.1 ETB_CTRL_COMID	
5.1.2.2 TRDP_ETBCTRL_DSID	
5.1.2.3 TRDP_MAX_FILE_NAME_LEN	
5.1.2.4 TRDP_MAX_LABEL_LEN	
5.1.2.5 TRDP_MAX_MD_DATA_SIZE	
OTTER THE _ MIN_ ME_DAMA_OTEE TO THE TOTAL TO THE TOTAL TOTA	

5.1.2.6 TRDP_MAX_URI_HOST_LEN	. 68
5.1.2.7 TRDP_MAX_URI_LEN	. 68
5.1.2.8 TRDP_MAX_URI_USER_LEN	. 68
5.1.2.9 TRDP_MD_DEFAULT_REPLY_TIMEOUT	. 68
5.1.2.10 TRDP_MD_INFINITE_TIME	. 68
5.1.2.11 TRDP_MIN_PD_HEADER_SIZE	. 69
5.1.2.12 TRDP_MSG_PD	. 69
5.1.2.13 TRDP_PD_UDP_PORT	. 69
5.1.2.14 TRDP_PROCESS_DEFAULT_CYCLE_TIME	. 69
5.1.2.15 TRDP_PROTOCOL_VERSION_CHECK_MASK	. 69
5.1.2.16 TRDP_USR_URI_SIZE	. 70
5.1.2.17 TTDB_NET_DIR_REQ_COMID	. 70
5.1.2.18 TTDB_OP_DIR_INFO_COMID	. 70
5.1.2.19 TTDB_STAT_CST_REQ_COMID	. 70
5.1.2.20 TTDB_TRN_DIR_REQ_COMID	. 70
5.2 tau_cstinfo.c File Reference	. 71
5.2.1 Detailed Description	. 72
5.2.2 Function Documentation	. 72
5.2.2.1 cstInfoGetPropSize()	. 72
5.3 tau_ctrl.c File Reference	. 73
5.3.1 Detailed Description	. 75
5.3.2 Function Documentation	. 75
5.3.2.1 tau_getEcspStat()	. 75
5.3.2.2 tau_initEcspCtrl()	. 75
5.3.2.3 tau_requestEcspConfirm()	. 76
5.3.2.4 tau_setEcspCtrl()	. 76
5.3.2.5 tau_terminateEcspCtrl()	. 77
5.4 tau_ctrl.h File Reference	. 77
5.4.1 Detailed Description	. 79
5.4.2 Function Documentation	. 80
5.4.2.1 tau_getEcspStat()	. 80
5.4.2.2 tau_initEcspCtrl()	. 80
5.4.2.3 tau_requestEcspConfirm()	. 81
5.4.2.4 tau_setEcspCtrl()	. 81
5.4.2.5 tau_terminateEcspCtrl()	. 82
5.5 tau_ctrl_types.h File Reference	. 82
5.5.1 Detailed Description	. 84
5.6 tau_dnr.c File Reference	. 85
5.6.1 Detailed Description	. 86
5.6.2 Function Documentation	. 87
5.6.2.1 tau_addr2Uri()	. 87
5.6.2.2 tau_delnitDnr()	. 87

5.6.2.3 tau_DNRstatus()	88
5.6.2.4 tau_getOwnAddr()	88
5.6.2.5 tau_initDnr()	88
5.6.2.6 tau_uri2Addr()	89
5.7 tau_dnr.h File Reference	90
5.7.1 Detailed Description	92
5.7.2 Enumeration Type Documentation	92
5.7.2.1 TRDP_DNR_OPTS	92
5.7.3 Function Documentation	92
5.7.3.1 tau_addr2Uri()	92
5.7.3.2 tau_deInitDnr()	93
5.7.3.3 tau_DNRstatus()	94
5.7.3.4 tau_getOwnAddr()	94
5.7.3.5 tau_initDnr()	95
5.7.3.6 tau_uri2Addr()	96
5.8 tau_dnr_types.h File Reference	97
5.8.1 Detailed Description	98
5.9 tau_marshall.c File Reference	99
5.9.1 Detailed Description	100
5.9.2 Function Documentation	100
	100
5.9.2.1 tau_calcDatasetSize()	100
5.9.2.1 tau_calcDatasetSize()	
	101
5.9.2.2 tau_calcDatasetSizeByComId()	101 102
5.9.2.2 tau_calcDatasetSizeByComId()	101 102 102
5.9.2.2 tau_calcDatasetSizeByComId()	101 102 102 104
5.9.2.2 tau_calcDatasetSizeByComId() 5.9.2.3 tau_initMarshall() 5.9.2.4 tau_marshall() 5.9.2.5 tau_marshallDs()	101 102 102 104
5.9.2.2 tau_calcDatasetSizeByComId() 5.9.2.3 tau_initMarshall() 5.9.2.4 tau_marshall() 5.9.2.5 tau_marshallDs() 5.9.2.6 tau_unmarshall()	101 102 102 104 105 105
5.9.2.2 tau_calcDatasetSizeByComId()	101 102 102 104 105 105
5.9.2.2 tau_calcDatasetSizeByComld()	101 102 102 104 105 105 106
5.9.2.2 tau_calcDatasetSizeByComld()	101 102 102 104 105 105 106 108
5.9.2.2 tau_calcDatasetSizeByComld()	101 102 104 105 105 106 108 108
5.9.2.2 tau_calcDatasetSizeByComld() 5.9.2.3 tau_initMarshall() 5.9.2.4 tau_marshall() 5.9.2.5 tau_marshallDs() 5.9.2.6 tau_unmarshall() 5.9.2.7 tau_unmarshallDs() 5.10 tau_marshall.h File Reference 5.10.1 Detailed Description 5.10.2 Function Documentation 5.10.2.1 tau_calcDatasetSize()	101 102 102 104 105 106 108 108 108
5.9.2.2 tau_calcDatasetSizeByComld() 5.9.2.3 tau_initMarshall() 5.9.2.4 tau_marshall() 5.9.2.5 tau_marshallDs() 5.9.2.6 tau_unmarshall() 5.9.2.7 tau_unmarshallDs() 5.10 tau_marshall.h File Reference 5.10.1 Detailed Description 5.10.2 Function Documentation 5.10.2.1 tau_calcDatasetSize() 5.10.2.2 tau_calcDatasetSizeByComld()	101 102 102 104 105 106 108 108 108 109 110
5.9.2.2 tau_calcDatasetSizeByComld() 5.9.2.3 tau_initMarshall() 5.9.2.4 tau_marshall() 5.9.2.5 tau_marshallDs() 5.9.2.6 tau_unmarshall() 5.9.2.7 tau_unmarshallDs() 5.10 tau_marshall.h File Reference 5.10.1 Detailed Description 5.10.2 Function Documentation 5.10.2.1 tau_calcDatasetSize() 5.10.2.2 tau_calcDatasetSizeByComld() 5.10.2.3 tau_initMarshall()	101 102 104 105 106 108 108 108 109 110
5.9.2.2 tau_calcDatasetSizeByComld() 5.9.2.3 tau_initMarshall() 5.9.2.4 tau_marshall() 5.9.2.5 tau_marshallDs() 5.9.2.6 tau_unmarshall() 5.9.2.7 tau_unmarshallDs() 5.10 tau_marshall.h File Reference 5.10.1 Detailed Description 5.10.2 Function Documentation 5.10.2.1 tau_calcDatasetSize() 5.10.2.2 tau_calcDatasetSizeByComld() 5.10.2.3 tau_initMarshall() 5.10.2.4 tau_marshall()	101 102 104 105 106 108 108 109 110 111 112
5.9.2.2 tau_calcDatasetSizeByComld() 5.9.2.3 tau_initMarshall() 5.9.2.4 tau_marshall() 5.9.2.5 tau_marshallDs() 5.9.2.6 tau_unmarshall() 5.9.2.7 tau_unmarshallDs() 5.10 tau_marshall.h File Reference 5.10.1 Detailed Description 5.10.2 Function Documentation 5.10.2.1 tau_calcDatasetSize() 5.10.2.2 tau_calcDatasetSizeByComld() 5.10.2.3 tau_initMarshall() 5.10.2.4 tau_marshall() 5.10.2.5 tau_marshallDs()	101 102 104 105 106 108 108 109 110 111 112 113
5.9.2.2 tau_calcDatasetSizeByComld() 5.9.2.3 tau_initMarshall() 5.9.2.4 tau_marshall() 5.9.2.5 tau_marshallDs() 5.9.2.6 tau_unmarshall() 5.9.2.7 tau_unmarshallDs() 5.10 tau_marshall.h File Reference 5.10.1 Detailed Description 5.10.2 Function Documentation 5.10.2.1 tau_calcDatasetSize() 5.10.2.2 tau_calcDatasetSizeByComld() 5.10.2.3 tau_initMarshall() 5.10.2.4 tau_marshall() 5.10.2.5 tau_marshallDs() 5.10.2.6 tau_unmarshall()	101 102 104 105 106 108 108 109 110 111 112 113 114
5.9.2.2 tau_calcDatasetSizeByComld() 5.9.2.3 tau_initMarshall() 5.9.2.4 tau_marshallDs() 5.9.2.5 tau_marshallDs() 5.9.2.6 tau_unmarshallDs() 5.9.2.7 tau_unmarshallDs() 5.10 tau_marshall.h File Reference 5.10.1 Detailed Description 5.10.2 Function Documentation 5.10.2.1 tau_calcDatasetSize() 5.10.2.2 tau_calcDatasetSizeByComld() 5.10.2.3 tau_initMarshall() 5.10.2.4 tau_marshall() 5.10.2.5 tau_marshallDs() 5.10.2.6 tau_unmarshallDs()	101 102 104 105 106 108 108 109 110 111 112 113 114 116
5.9.2.2 tau_calcDatasetSizeByComld() 5.9.2.3 tau_initMarshall() 5.9.2.4 tau_marshall() 5.9.2.5 tau_marshallDs() 5.9.2.6 tau_unmarshall() 5.9.2.7 tau_unmarshallDs() 5.10 tau_marshall.h File Reference 5.10.1 Detailed Description 5.10.2 Function Documentation 5.10.2.1 tau_calcDatasetSize() 5.10.2.2 tau_calcDatasetSizeByComld() 5.10.2.3 tau_initMarshall() 5.10.2.4 tau_marshall() 5.10.2.5 tau_marshallDs() 5.10.2.6 tau_unmarshall() 5.10.2.7 tau_unmarshall() 5.10.2.7 tau_unmarshallDs()	101 102 104 105 106 108 108 109 110 111 112 113 114 116 117
5.9.2.2 tau_calcDatasetSizeByComld() 5.9.2.3 tau_initMarshall() 5.9.2.4 tau_marshall() 5.9.2.5 tau_marshallDs() 5.9.2.6 tau_unmarshallDs() 5.9.2.7 tau_unmarshallDs() 5.10 tau_marshall.h File Reference 5.10.1 Detailed Description 5.10.2 Function Documentation 5.10.2.1 tau_calcDatasetSize() 5.10.2.2 tau_calcDatasetSizeByComld() 5.10.2.3 tau_initMarshall() 5.10.2.4 tau_marshallDs() 5.10.2.5 tau_marshallDs() 5.10.2.6 tau_unmarshall() 5.10.2.7 tau_unmarshall() 5.10.2.7 tau_unmarshall() 5.10.2.7 tau_unmarshall()	101 102 104 105 106 108 108 109 110 111 112 113 114 116 117

5.11.2.3 tau_freeServicesList()
5.11.2.4 tau_getServicesList()
5.11.2.5 tau_updService()
5.12 tau_so_if.h File Reference
5.12.1 Detailed Description
5.12.2 Function Documentation
5.12.2.1 tau_addService()
5.12.2.2 tau_delService()
5.12.2.3 tau_freeServicesList()
5.12.2.4 tau_getServicesList()
5.12.2.5 tau_updService()
5.13 tau_tti.c File Reference
5.13.1 Detailed Description
5.13.2 Macro Definition Documentation
5.13.2.1 TTI_CACHED_CONSISTS
5.13.3 Function Documentation
5.13.3.1 tau_deInitTTI()
5.13.3.2 tau_getCstFctCnt()
5.13.3.3 tau_getCstFctInfo()
5.13.3.4 tau_getCstInfo()
5.13.3.5 tau_getCstVehCnt()
5.13.3.6 tau_getOpTrDirectory()
5.13.3.7 tau_getOpTrnDirectoryStatusInfo()
5.13.3.8 tau_getOwnlds()
5.13.3.9 tau_getOwnOpCstNo()
5.13.3.10 tau_getOwnTrnCstNo()
5.13.3.11 tau_getStaticCstInfo()
5.13.3.12 tau_getTrDirectory()
5.13.3.13 tau_getTrnCstCnt()
5.13.3.14 tau_getTrnVehCnt()
5.13.3.15 tau_getTTI()
5.13.3.16 tau_getVehInfo()
5.13.3.17 tau_getVehOrient()
5.13.3.18 tau_initTTlaccess()
5.14 tau_tti.h File Reference
5.14.1 Detailed Description
5.14.2 Function Documentation
5.14.2.1 tau_deInitTTI()
5.14.2.2 tau_getCstFctCnt()
5.14.2.3 tau_getCstFctInfo()
5.14.2.4 tau_getCstInfo()
5.14.2.5 tau_getCstVehCnt()

5.14.2.6 tau_getOpTrDirectory()
5.14.2.7 tau_getOpTrnDirectoryStatusInfo()
5.14.2.8 tau_getOwnlds()
5.14.2.9 tau_getOwnOpCstNo()
5.14.2.10 tau_getOwnTrnCstNo()
5.14.2.11 tau_getStaticCstInfo()
5.14.2.12 tau_getTrDirectory()
5.14.2.13 tau_getTrnCstCnt()
5.14.2.14 tau_getTrnVehCnt()
5.14.2.15 tau_getTTI()
5.14.2.16 tau_getVehInfo()
5.14.2.17 tau_getVehOrient()
5.14.2.18 tau_initTTlaccess()
5.15 tau_tti_types.h File Reference
5.15.1 Detailed Description
5.16 tau_xml.c File Reference
5.16.1 Detailed Description
5.16.2 Macro Definition Documentation
5.16.2.1 TRDP_SDT_DEFAULT_CMTHR
5.16.2.2 TRDP_SDT_DEFAULT_LMIMAX
5.16.3 Function Documentation
5.16.3.1 tau_freeTelegrams()
5.16.3.2 tau_freeXmlDatasetConfig()
5.16.3.3 tau_freeXmlDoc()
5.16.3.4 tau_prepareXmlDoc()
5.16.3.5 tau_prepareXmlMem()
5.16.3.6 tau_readXmlDatasetConfig()
5.16.3.7 tau_readXmlDeviceConfig()
5.16.3.8 tau_readXmlInterfaceConfig()
5.16.3.9 tau_readXmlMappedDeviceConfig()
5.16.3.10 tau_readXmlMappedDevices()
5.16.3.11 tau_readXmlMappedInterfaceConfig()
5.16.3.12 tau_readXmlServiceConfig()
5.17 tau_xml.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()	65
5.17.4.4 tau_prepareXmlDoc()	65
5.17.4.5 tau_prepareXmlMem()	66
5.17.4.6 tau_readXmlDatasetConfig()	66
5.17.4.7 tau_readXmlDeviceConfig()	67
5.17.4.8 tau_readXmlInterfaceConfig()	68
5.17.4.9 tau_readXmlMappedDeviceConfig()	68
5.17.4.10 tau_readXmlMappedDevices()	69
5.17.4.11 tau_readXmlMappedInterfaceConfig()	69
5.17.4.12 tau_readXmlServiceConfig()	70
5.18 tlc_if.c File Reference	70
5.18.1 Detailed Description	72
5.18.2 Function Documentation	72
5.18.2.1 tlc_closeSession()	72
5.18.2.2 tlc_configSession()	73
5.18.2.3 tlc_getETBTopoCount()	73
5.18.2.4 tlc_getInterval()	74
5.18.2.5 tlc_getOpTrainTopoCount()	74
5.18.2.6 tlc_getOwnlpAddress()	75
5.18.2.7 tlc_getVersion()	75
5.18.2.8 tlc_getVersionString()	75
5.18.2.9 tlc_init()	76
5.18.2.10 tlc_openSession()	76
5.18.2.11 tlc_presetIndexSession()	77
5.18.2.12 tlc_process()	77
5.18.2.13 tlc_reinitSession()	78
5.18.2.14 tlc_setETBTopoCount()	79
5.18.2.15 tlc_setOpTrainTopoCount()	79
5.18.2.16 tlc_terminate()	79
5.18.2.17 tlc_updateSession()	80
5.18.2.18 trdp_getAccess()	80
5.18.2.19 trdp_isValidSession()	81
5.18.2.20 trdp_releaseAccess()	81
5.18.2.21 trdp_sessionQueue()	82
5.19 tlm_if.c File Reference	82
5.19.1 Detailed Description	83
5.19.2 Function Documentation	84
5.19.2.1 tlm_abortSession()	84
5.19.2.2 tlm_addListener()	84
5.19.2.3 tlm_confirm()	85
5.19.2.4 tlm_delListener()	86
5.19.2.5 tlm_getInterval()	86

5.19.2.6 tlm_notify()	 187
5.19.2.7 tlm_process()	 188
5.19.2.8 tlm_readdListener()	 188
5.19.2.9 tlm_reply()	 189
5.19.2.10 tlm_replyQuery()	 190
5.19.2.11 tlm_request()	 190
5.20 tlp_if.c File Reference	 191
5.20.1 Detailed Description	 193
5.20.2 Function Documentation	 193
5.20.2.1 tlp_get()	 193
5.20.2.2 tlp_getInterval()	 194
5.20.2.3 tlp_getRedundant()	 194
5.20.2.4 tlp_processReceive()	 195
5.20.2.5 tlp_processSend()	 195
5.20.2.6 tlp_publish()	 196
5.20.2.7 tlp_put()	 197
5.20.2.8 tlp_putImmediate()	 197
5.20.2.9 tlp_republish()	 198
5.20.2.10 tlp_request()	 199
5.20.2.11 tlp_resubscribe()	 200
5.20.2.12 tlp_setRedundant()	 200
5.20.2.13 tlp_subscribe()	 201
5.20.2.14 tlp_unpublish()	 202
5.20.2.15 tlp_unsubscribe()	 202
5.21 trdp_if_light.h File Reference	 203
5.21.1 Detailed Description	 207
5.21.2 Function Documentation	 207
5.21.2.1 tlc_closeSession()	 207
5.21.2.2 tlc_configSession()	 208
5.21.2.3 tlc_getETBTopoCount()	 208
5.21.2.4 tlc_getInterval()	 209
5.21.2.5 tlc_getJoinStatistics()	 209
5.21.2.6 tlc_getOpTrainTopoCount()	 210
5.21.2.7 tlc_getOwnlpAddress()	 210
5.21.2.8 tlc_getPubStatistics()	 210
5.21.2.9 tlc_getRedStatistics()	 211
5.21.2.10 tlc_getStatistics()	 211
5.21.2.11 tlc_getSubsStatistics()	 212
5.21.2.12 tlc_getVersion()	 212
5.21.2.13 tlc_getVersionString()	 213
5.21.2.14 tlc_init()	 213
5.21.2.15 tlc_openSession()	 214

5.21.2.16 tlc_presetIndexSession()	4
5.21.2.17 tlc_process()	5
5.21.2.18 tlc_reinitSession()	5
5.21.2.19 tlc_resetStatistics()	6
5.21.2.20 tlc_setETBTopoCount()	6
5.21.2.21 tlc_setOpTrainTopoCount()	7
5.21.2.22 tlc_terminate()	7
5.21.2.23 tlc_updateSession()	7
5.21.2.24 tlm_abortSession()	8
5.21.2.25 tlm_addListener()	8
5.21.2.26 tlm_confirm()	9
5.21.2.27 tlm_delListener()	0
5.21.2.28 tlm_getInterval()	0
5.21.2.29 tlm_notify()	.1
5.21.2.30 tlm_process()	2
5.21.2.31 tlm_readdListener()	2
5.21.2.32 tlm_reply()	3
5.21.2.33 tlm_replyQuery()	4
5.21.2.34 tlm_request()	4
5.21.2.35 tlp_get()	5
5.21.2.36 tlp_getInterval()	6
5.21.2.37 tlp_getRedundant()	7
5.21.2.38 tlp_processReceive()	7
5.21.2.39 tlp_processSend()	8
5.21.2.40 tlp_publish()	8
5.21.2.41 tlp_put()	9
5.21.2.42 tlp_putImmediate()	0
5.21.2.43 tlp_republish()	0
5.21.2.44 tlp_request()	1
5.21.2.45 tlp_resubscribe()	2
5.21.2.46 tlp_setRedundant()	3
5.21.2.47 tlp_subscribe()	3
5.21.2.48 tlp_unpublish()	5
5.21.2.49 tlp_unsubscribe()	6
5.22 trdp_serviceRegistry.h File Reference	6
5.22.1 Detailed Description	0
5.22.2 Macro Definition Documentation	0
5.22.2.1 SOA_SAME_SERVICEID	1
5.22.2.2 SRM_SERVICE_READ_REQ_COMID	1
5.22.2.3 SRM_SRVINFO_NOTIFY_COMID	1
5.23 trdp_stats.c File Reference	1
5.23.1 Detailed Description	2

5.23.2 Function Documentation	243
5.23.2.1 tlc_getJoinStatistics()	243
5.23.2.2 tlc_getPubStatistics()	243
5.23.2.3 tlc_getRedStatistics()	244
5.23.2.4 tlc_getStatistics()	244
5.23.2.5 tlc_getSubsStatistics()	245
5.23.2.6 tlc_resetStatistics()	245
5.23.2.7 trdp_initStats()	247
5.23.2.8 trdp_pdPrepareStats()	247
5.23.2.9 trdp_UpdateStats()	248
5.24 trdp_tsn_def.h File Reference	248
5.24.1 Detailed Description	249
5.24.2 Macro Definition Documentation	250
5.24.2.1 TRDP_MIN_PD2_HEADER_SIZE	250
5.24.2.2 TRDP_MSG_TSN_PD	250
5.24.2.3 TRDP_PD_DEFAULT_QOS	250
5.25 trdp_types.h File Reference	250
5.25.1 Detailed Description	256
5.25.2 Macro Definition Documentation	256
5.25.2.1 TRDP_FLAGS_DEFAULT	256
5.25.3 Typedef Documentation	257
5.25.3.1 TRDP_IP_ADDR_T	257
5.25.3.2 TRDP_MARSHALL_T	257
5.25.3.3 TRDP_MD_CALLBACK_T	257
5.25.3.4 TRDP_PD_CALLBACK_T	258
5.25.3.5 TRDP_PRINT_DBG_T	258
5.25.3.6 TRDP_TIME_T	258
5.25.3.7 TRDP_UNMARSHALL_T	258
5.25.4 Enumeration Type Documentation	259
5.25.4.1 TRDP_DATA_TYPE_T	259
5.25.4.2 TRDP_ERR_T	260
5.25.4.3 TRDP_RED_STATE_T	262
5.25.4.4 TRDP_REPLY_STATUS_T	262
5.25.4.5 TRDP_TO_BEHAVIOR_T	262
5.26 vos_mem.c File Reference	262
5.26.1 Detailed Description	264
5.26.2 Function Documentation	264
5.26.2.1 vos_bsearch()	264
5.26.2.2 vos_memAlloc()	265
5.26.2.3 vos_memCount()	265
5.26.2.4 vos_memDelete()	266
5.26.2.5 vos_memFree()	266

5.26.2.6 vos_memInit()
5.26.2.7 vos_qsort()
5.26.2.8 vos_queueCreate()
5.26.2.9 vos_queueDestroy()
5.26.2.10 vos_queueReceive()
5.26.2.11 vos_queueSend()
5.26.2.12 vos_strncat()
5.26.2.13 vos_strncpy()
5.26.2.14 vos_strnicmp()
5.27 vos_mem.h File Reference
5.27.1 Detailed Description
5.27.2 Macro Definition Documentation
5.27.2.1 VOS_MEM_MAX_PREALLOCATE
5.27.2.2 VOS_MEM_PREALLOCATE
5.27.3 Function Documentation
5.27.3.1 vos_bsearch()
5.27.3.2 vos_memAlloc()
5.27.3.3 vos_memCount()
5.27.3.4 vos_memDelete()
5.27.3.5 vos_memFree()
5.27.3.6 vos_memInit()
5.27.3.7 vos_qsort()
5.27.3.8 vos_queueCreate()
5.27.3.9 vos_queueDestroy()
5.27.3.10 vos_queueReceive()
5.27.3.11 vos_queueSend()
5.27.3.12 vos_strncat()
5.27.3.13 vos_strncpy()
5.27.3.14 vos_strnicmp()
5.28 vos_shared_mem.h File Reference
5.28.1 Detailed Description
5.28.2 Function Documentation
5.28.2.1 vos_sharedClose()
5.28.2.2 vos_sharedOpen()
5.29 vos_sock.h File Reference
5.29.1 Detailed Description
5.29.2 Macro Definition Documentation
5.29.2.1 VOS_MAX_SOCKET_CNT
5.29.2.2 VOS_TTL_MULTICAST
5.29.3 Function Documentation
5.29.3.1 vos_determineBindAddr()
5.29.3.2 vos. dottedIP()

5.29.3.3 vos_getInterfaces()	288
5.29.3.4 vos_htonl()	289
5.29.3.5 vos_htonll()	289
5.29.3.6 vos_htons()	289
5.29.3.7 vos_ipDotted()	290
5.29.3.8 vos_isMulticast()	290
5.29.3.9 vos_netIfUp()	291
5.29.3.10 vos_ntohl()	291
5.29.3.11 vos_ntohll()	291
5.29.3.12 vos_ntohs()	292
5.29.3.13 vos_select()	292
5.29.3.14 vos_sockAccept()	292
5.29.3.15 vos_sockBind()	293
5.29.3.16 vos_sockClose()	293
5.29.3.17 vos_sockConnect()	294
5.29.3.18 vos_sockGetMAC()	294
5.29.3.19 vos_sockInit()	295
5.29.3.20 vos_sockJoinMC()	295
5.29.3.21 vos_sockLeaveMC()	296
5.29.3.22 vos_sockListen()	296
5.29.3.23 vos_sockOpenTCP()	297
5.29.3.24 vos_sockOpenUDP()	297
5.29.3.25 vos_sockReceiveTCP()	297
5.29.3.26 vos_sockReceiveUDP()	298
5.29.3.27 vos_sockSendTCP()	299
5.29.3.28 vos_sockSendUDP()	299
5.29.3.29 vos_sockSetMulticastIf()	300
5.29.3.30 vos_sockSetOptions()	300
5.29.3.31 vos_sockTerm()	301
5.30 vos_thread.h File Reference	301
5.30.1 Detailed Description	304
5.30.2 Function Documentation	304
5.30.2.1 vos_addTime()	304
5.30.2.2 vos_clearTime()	305
5.30.2.3 vos_cmpTime()	305
5.30.2.4 vos_divTime()	305
5.30.2.5 vos_getRealTime()	306
5.30.2.6 vos_getTime()	306
5.30.2.7 vos_getTimeStamp()	306
5.30.2.8 vos_getUuid()	306
5.30.2.9 vos_mulTime()	307
5.30.2.10 vos_mutexCreate()	307

5.30.2.11 vos_mutexDelete()
5.30.2.12 vos_mutexLock()
5.30.2.13 vos_mutexTryLock()
5.30.2.14 vos_mutexUnlock()
5.30.2.15 vos_semaCreate()
5.30.2.16 vos_semaDelete()
5.30.2.17 vos_semaGive()
5.30.2.18 vos_semaTake()
5.30.2.19 vos_subTime()
5.30.2.20 vos_threadCreate()
5.30.2.21 vos_threadCreateSync()
5.30.2.22 vos_threadDelay()
5.30.2.23 vos_threadInit()
5.30.2.24 vos_threadIsActive()
5.30.2.25 vos_threadSelf()
5.30.2.26 vos_threadTerm()
5.30.2.27 vos_threadTerminate()
5.31 vos_types.h File Reference
5.31.1 Detailed Description
5.31.2 Typedef Documentation
5.31.2.1 VOS_PRINT_DBG_T
5.31.2.2 VOS_TIMEVAL_T
5.31.3 Enumeration Type Documentation
5.31.3.1 VOS_ERR_T 31
5.31.3.2 VOS_LOG_T
5.32 vos_utils.c File Reference
5.32.1 Detailed Description
5.32.2 Function Documentation
5.32.2.1 vos_crc32()
5.32.2.2 vos_getErrorString()
5.32.2.3 vos_getVersion()
5.32.2.4 vos_getVersionString()
5.32.2.5 vos_hostIsBigEndian()
5.32.2.6 vos_init()
5.32.2.7 vos_sc32()
5.32.2.8 vos_terminate()
5.33 vos_utils.h File Reference
5.33.1 Detailed Description
5.33.2 Macro Definition Documentation
5.33.2.1 INITFCS
5.33.2.2 VOS_MAX_ERR_STR_SIZE
5.33.2.3 VOS MAX FRMT SIZE

Index		331
	5.33.3.8 vos_terminate()	329
	5.33.3.7 vos_sc32()	329
	5.33.3.6 vos_init()	328
	5.33.3.5 vos_hostIsBigEndian()	328
	5.33.3.4 vos_getVersionString()	327
	5.33.3.3 vos_getVersion()	327
	5.33.3.2 vos_getErrorString()	327
	5.33.3.1 vos_crc32()	326
	5.33.3 Function Documentation	326
	5.33.2.4 VOS_MAX_PRNT_STR_SIZE	326

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:

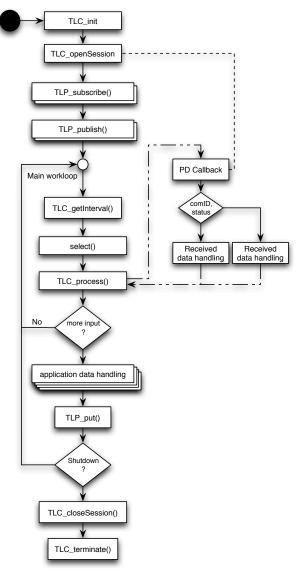


Figure 1.1 Sample client workflow (Single Thread)

1.3 Use Cases 3

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

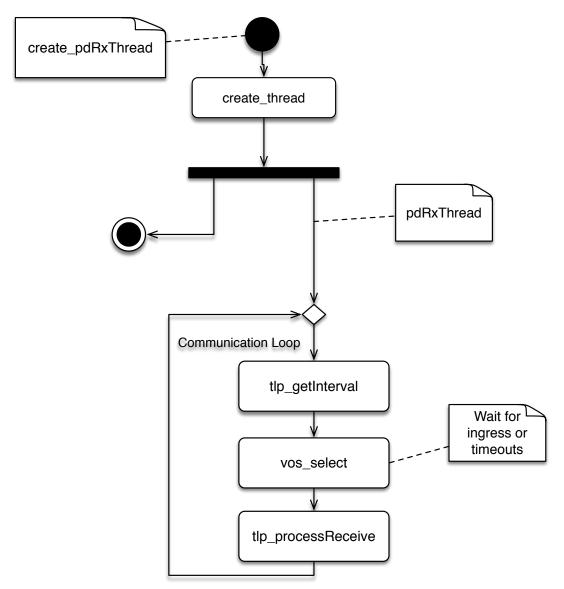


Figure 1.2 Multi-threaded processing of PD Reception

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

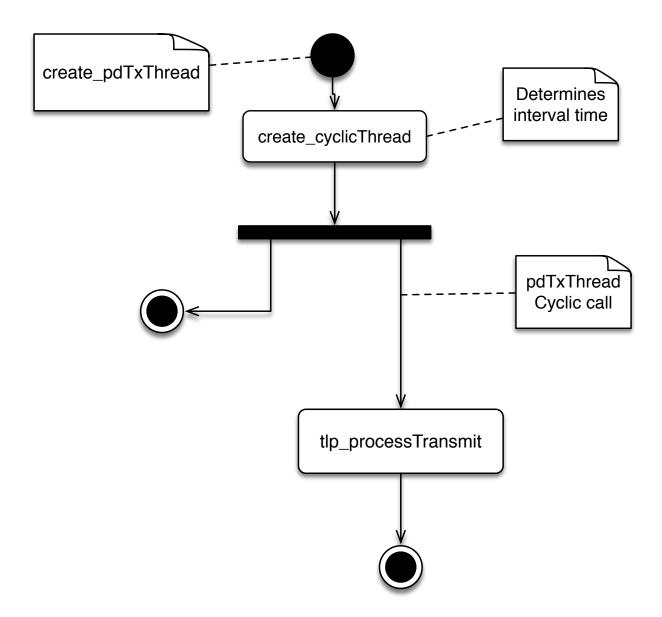


Figure 1.3 Multi-threaded processing of PD Transmit

1.4 Conventions of the API 5

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

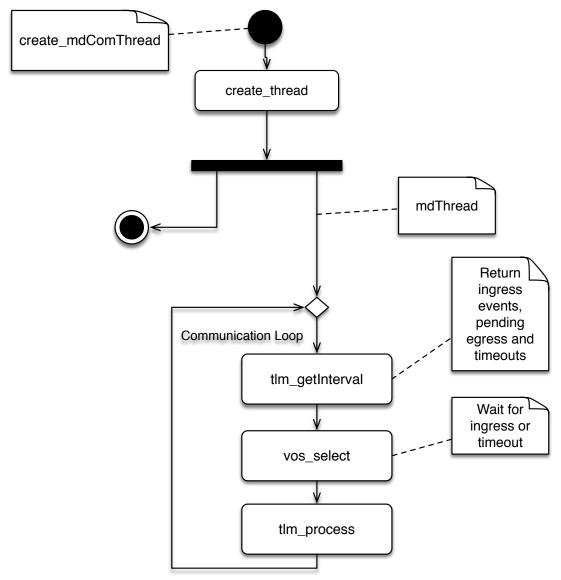


Figure 1.4 Multi-threaded processing of MD

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: $finclude \ "trdp_if_light.h"$

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).

Data Structure Index

2.1 Data Structures

Here are the data structures with brief descriptions:

DNS_HEADER	
DNS header structure	11
GNU_PACKED	
Types for ETB control	11
service_info	
Preliminary definition of a service info entry	30
srv_info_req	
Preliminary definition of a service info request	32
Marshalling info, used to and from wire	32
TCN URI	
TCN-DNS simplified header structures	33
TRDP CLTR CST INFO T	
Closed train consists information	34
TRDP_COM_PARAM_T	
Quality/type of service, time to live, no	34
TRDP_COMID_DSID_MAP_T	
Comld - data set mapping element definition	
35	
TRDP_CONSIST_INFO_T	
Consist information structure	35
TRDP_DATASET	
Dataset definition	
37	
TRDP_DATASET_ELEMENT_T	
Dataset element definition	
38	
TRDP_DBG_CONFIG_T Control for debug output device/file on application level	20
TRDP DNS REPLY	39
TCN-DNS Reply telegram TCN DNS REP DS	40
TRDP DNS REQUEST	40
TCN-DNS Request telegram TCN_DNS_REQ_DS	41
TRDP ETB INFO T	71
Types for train configuration information	45

8 Data Structure Index

TRDP_FUNCTION_INFO_T	
Function/device information structure	43
TRDP_IDX_TABLE_T	
Settings for pre-allocation of index tables for application session initialization	45
TRDP_MARSHALL_CONFIG_T	
Marshaling/unmarshalling configuration	
47	
TRDP_MD_CONFIG_T	
Default MD configuration	48
TRDP_MD_INFO_T	
Message data info from received telegram; allows the application to generate responses	49
TRDP_MEM_CONFIG_T	
Enumeration type for memory pre-fragmentation, reuse of VOS definition	50
TRDP_PD_CONFIG_T	
Default PD configuration	
51	
TRDP_PD_INFO_T	
Process data info from received telegram; allows the application to generate responses	52
TRDP_PROCESS_CONFIG_T	
Various flags/general TRDP options for library initialization	54
TRDP_PROP_T	
Application defined properties	54
TRDP_SDT_PAR_T	
Types to read out the XML configuration	
55	
TRDP_VEHICLE_INFO_T	
Vehicle information structure	56
TRDP_XML_DOC_HANDLE_T	
Parsed XML document handle	57
VOS_SOCK_OPT_T	
Common socket options	
57	
VOS_VERSION_T	
Version information	58

File Index

3.1 File List

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

leco1375-2-3.fl
All definitions from IEC 61375-2-3
tau_cstinfo.c
Functions for consist information access
tau_ctrl.c
Functions for train switch control
tau_ctrl.h
TRDP utility interface definitions
tau_ctrl_types.h
TRDP utility interface definitions
tau_dnr.c
Functions for domain name resolution
tau_dnr.h
TRDP utility interface definitions
tau_dnr_types.h
TRDP utility interface definitions
tau_marshall.c
Marshalling functions for TRDP
tau_marshall.h
TRDP utility interface definitions
tau_so_if.c
Access to service oriented functions of the SRM
tau_so_if.h
Access to the Service Registry
tau_tti.c
Functions for train topology information access
tau_tti.h
TRDP utility interface definitions
tau_tti_types.h
TRDP utility interface definitions
tau_xml.c
Functions for XML file parsing
tau_xml.h
TRDP utility interface definitions
tlc_if.c Functions for FCN communication
Functions for FLAN communication 1/1

10 File Index

tlm if.c		
_	Functions for Message Data Communication	182
tlp_if.c		
	Functions for Process Data Communication	191
trdp_if_li	ght.h	
	TRDP Light interface functions (API)	203
trdp_serv	viceRegistry.h	
	Additional definitions for IEC 61375-2-3 (Service Discovery) The definitions herein are preliminary and will change with the next major release of the IEC 61375-2-3 standard	236
trdp_stat	S.C	
trdp tsn	Statistics functions for TRDP communication	241
trup_tori_	Additional definitions for TSN	248
trdp_type		
1 – 71	Typedefs for TRDP communication	250
vos_men	n.c	
	Memory functions	262
vos_men	n.h	
	Memory and queue functions for OS abstraction	272
vos_shar	red_mem.h	
	Shared Memory functions for OS abstraction	282
vos_sock		
	Typedefs for OS abstraction	284
vos_thre		
	Threading functions for OS abstraction	301
vos_type		
	Typedefs for OS abstraction	314
vos_utils		
	Common functions for VOS	319
vos_utils		
	Typedefs for OS abstraction	323

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:

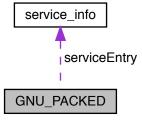
• tau_dnr.c

4.2 GNU_PACKED Struct Reference

Types for ETB control.

#include <trdp_types.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 \leftrightarrow 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 reqSafetyCode

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 etbnlnaugState

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 1 4 = end node TCN direction 2

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 safeSeqCount

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: 'FFFFFFFH'H)

UINT8 opTrnOrient

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

UINT8 opCstCnt

number of consists in train (1..63)

• TRDP_OP_CONSIST_T opCstList [TRDP_MAX_CST_CNT]

operational consist list starting with op.

• UINT8 reserved05

reserved for future use (= 0)

UINT8 opVehCnt

number of vehicles in train (1..63)

TRDP_OP_VEHICLE_T opVehList [TRDP_MAX_VEH_CNT]

operational vehicle list starting with op.

• TRDP_OP_TRAIN_DIR_STATE_T state

operational state of the train

UINT32 cstNetProp

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

UINT16 entryCnt

number of entries in train network directory

TRDP_TRAIN_NET_DIR_ENTRY_T trnNetDir [TRDP_MAX_CST_CNT]

train network directory

• TRDP_OP_TRAIN_DIR_T opTrnDir

operational directory

• TRDP TRAIN DIR T trnDir

train directory

UINT16 noOfEntries

number of entries in array

• SRM_SERVICE_INFO_T serviceEntry [1]

var.

• UINT32 comId

Comld to request: 35...41.

· UINT32 total

total memory size

· UINT32 free

free memory size

UINT32 minFree

minimal free memory size in statistics interval

UINT32 numAllocBlocks

allocated memory blocks

UINT32 numAllocErr

allocation errors

UINT32 numFreeErr

free errors

• UINT32 blockSize [VOS_MEM_NBLOCKSIZES]

preallocated memory blocks

• UINT32 usedBlockSize [VOS_MEM_NBLOCKSIZES]

used memory blocks

UINT32 defQos

default QoS for PD

UINT32 defTtl

default TTL for PD

UINT32 defTimeout

default timeout in us for PD

UINT32 numSubs

number of subscribed Comld's

UINT32 numPub

number of published 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 ThostName

host name

TRDP_NET_LABEL_T leaderName

leader host name

• TRDP_IP_ADDR_T ownlpAddr

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.

4.2.1 Detailed Description

Types for ETB control.

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, redId, redState cycle, ttl, qos, counter
- PD join table: joined MC address table
- · MD listener table: Comld destlpAddr, destUri, cbFct?, counter
- · Memory usage Structure 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

Comld to request: 35...41.

Comld to listen to.

Published Comld

Subscribed ComId

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 cstCnt

UINT8 GNU_PACKED::cstCnt

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

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

4.2.2.6 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.

4.2.2.7 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)

correction)

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 deviceName

TRDP_NET_LABEL_T GNU_PACKED::deviceName

function device of ECSC which sends the telegram

function device of ED which sends the telegram

4.2.2.11 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)

4.2.2.12 etbInhibit

UINT8 GNU_PACKED::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

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

4.2.2.13 etbLength

UINT8 GNU_PACKED::etbLength

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

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

4.2.2.14 etbShort

UINT8 GNU_PACKED::etbShort

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

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

4.2.2.15 etbTopoCnt

UINT32 GNU_PACKED::etbTopoCnt

ETB topography counter

train network directory CRC

4.2.2.16 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

4.2.2.17 isLead

ANTIVALENT8 GNU_PACKED::isLead

vehicle is leading

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

4.2.2.18 joinedAddr

```
TRDP_IP_ADDR_T GNU_PACKED::joinedAddr
```

Joined IP address

Joined IP address.

4.2.2.19 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 = leading direction 2 = le

4.2.2.20 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 $1\ 2 = second\ vehicle\ etc.$

4.2.2.21 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

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

4.2.2.31 opVehList

reserved for future use (= 0)

```
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 reserved01 [1/2]
UINT16 GNU_PACKED::reserved01
reserved (=0)
reserved for future use (= 0)
reserved for future use (= 0)
reserved (=0)
4.2.2.34 reserved01 [2/2]
UINT8 GNU_PACKED::reserved01
reserved (=0)
reserved for future use (= 0)
4.2.2.35 reserved02 [1/2]
UINT16 GNU_PACKED::reserved02
reserved (=0)
reserved (= 0)
reserved (=0)
```

4.2.2.36 reserved02 [2/2]

```
UINT16 GNU_PACKED::reserved02
reserved (=0)
reserved (= 0)
reserved (=0)
4.2.2.37 reserved03
UINT8 GNU_PACKED::reserved03
reserved (=0)
reserved for future use (= 0)
4.2.2.38 reserved04
UINT8 GNU_PACKED::reserved04
reserved (=0)
reserved for future use (= 0)
4.2.2.39 reserved06
UINT8 GNU_PACKED::reserved06
reserved (=0)
reserved for future use (= 0)
4.2.2.40 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.
ETBCTRL-VDP trailer, completely set to 0 == SDTv2 not used
```

4.2.2.41 serviceEntry

```
SRM_SERVICE_INFO_T GNU_PACKED::serviceEntry[1]
```

var.

number of entries

4.2.2.42 timeout

```
UINT32 GNU_PACKED::timeout
```

Time-out value in us.

0 = No time-out supervision

4.2.2.43 toBehav

```
UINT32 GNU_PACKED::toBehav
```

Behavior at time-out.

Set data to zero / keep last value

4.2.2.44 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)

4.2.2.45 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.

4.2.2.46 trnld

```
TRDP_NET_LABEL_T GNU_PACKED::trnId train identifier, application defined (e.g. 'ICE75', 'IC346'), informal
```

4.2.2.47 trnNetDir

```
TRDP_TRAIN_NET_DIR_T GNU_PACKED::trnNetDir
dynamic train info
network directory
```

4.2.2.48 trnOperator

```
TRDP_NET_LABEL_T GNU_PACKED::trnOperator
train operator, e.g.
'trenitalia.it', informal
```

4.2.2.49 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)
```

4.2.2.50 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

4.2.2.51 vehld

```
TRDP_NET_LABEL_T GNU_PACKED::vehId
```

Unique vehicle identifier, application defined (e.g.

UIC Identifier)

4.2.2.52 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

4.2.2.53 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

4.3 service info Struct Reference

Preliminary definition of a service info entry.

#include <trdp_serviceRegistry.h>

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

4.3.1 Detailed Description

Preliminary definition of a service info entry.

4.3.2 Field Documentation

4.3.2.1 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

4.4 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</p>
- UINT32 srvTcList []
 - list of srvTopoCnt values obtained from all consists set to 0 if unknown ordered list starting with trnCstNo = 1

4.4.1 Detailed Description

Preliminary definition of a service info request.

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

• trdp_serviceRegistry.h

4.5 TAU_MARSHALL_INFO_T Struct Reference

Marshalling info, used to and from wire.

INT32 level

track recursive level

• UINT8 * pSrc source pointer

• UINT8 * pSrcEnd last source

UINT8 * pDst
 destination pointer

• UINT8 * pDstEnd last destination

4.5.1 Detailed Description

Marshalling info, used to and from wire.

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

• tau_marshall.c

4.6 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

4.6.1 Detailed Description

TCN-DNS simplified header structures.

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

tau_dnr_types.h

4.7 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)

4.7.1 Detailed Description

Closed train consists information.

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

· tau_tti_types.h

4.8 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

4.8.1 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

4.9 TRDP_COMID_DSID_MAP_T Struct Reference

Comld - data set mapping element definition

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

Data Fields

- UINT32 comld
- UINT32 datasetId
 corresponding dataset Id

4.9.1 Detailed Description

Comld - data set mapping element definition

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

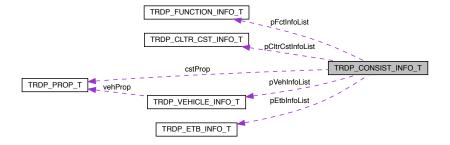
· trdp_types.h

4.10 TRDP_CONSIST_INFO_T Struct Reference

consist information structure

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

Collaboration diagram for TRDP_CONSIST_INFO_T:



• TRDP_SHORT_VERSION_T version

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

UINT8 cstClass

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

UINT8 reserved01

reserved for future use (= 0)

• TRDP_NET_LABEL_T cstld

application defined consist identifier, e.g.

TRDP_NET_LABEL_T cstType

consist type, application defined

TRDP_NET_LABEL_T cstOwner

consist owner, e.g.

TRDP UUID T cstUUID

consist UUID

UINT32 reserved02

reserved for future use (= 0)

TRDP_PROP_T cstProp

static consist properties

• UINT16 reserved03

reserved for future use (= 0)

UINT16 etbCnt

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

TRDP_ETB_INFO_T * pEtbInfoList

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

• UINT16 reserved04

reserved for future use (= 0)

UINT16 vehCnt

number of vehicles in consist 1..32

TRDP_VEHICLE_INFO_T * pVehInfoList

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

• UINT16 reserved05

reserved for future use (= 0)

UINT16 fctCnt

number of consist functions value range 0..1024

• TRDP_FUNCTION_INFO_T * pFctInfoList

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

UINT16 reserved06

reserved for future use (= 0)

UINT16 cltrCstCnt

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

TRDP_CLTR_CST_INFO_T * pCltrCstInfoList

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

UINT32 cstTopoCnt

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

4.10.1 Detailed Description

consist information structure

4.10.2 Field Documentation

4.10.2.1 cstld

```
TRDP_NET_LABEL_T TRDP_CONSIST_INFO_T::cstId
```

application defined consist identifier, e.g.

UIC identifier

4.10.2.2 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:

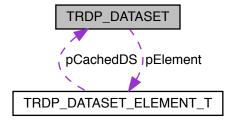
• tau_tti_types.h

4.11 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

4.11.1 Detailed Description

Dataset definition

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

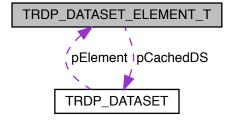
• trdp_types.h

4.12 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

4.12.1 Detailed Description

Dataset element definition

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

• trdp_types.h

4.13 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.

4.13.1 Detailed Description

Control for debug output device/file on application level.

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

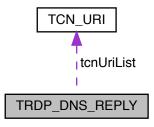
• tau xml.h

4.14 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

4.14.1 Detailed Description

TCN-DNS Reply telegram TCN_DNS_REP_DS.

4.14.2 Field Documentation

4.14.2.1 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:

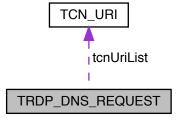
• tau_dnr_types.h

4.15 TRDP_DNS_REQUEST Struct Reference

TCN-DNS Request telegram TCN_DNS_REQ_DS.

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

Collaboration diagram for TRDP_DNS_REQUEST:



• TRDP_SHORT_VERSION_T version

1.0

• TRDP_NET_LABEL_T deviceName

function device of ED which sends the telegram

UINT32 etbTopoCnt

ETB topography counter.

UINT32 opTrnTopoCnt

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

· UINT8 etbld

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

UINT8 tcnUriCnt

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

• TCN_URI_T tcnUriList [255]

defined for max size

 TRDP_ETB_CTRL_VDP_T safetyTrail
 SDT trailer

4.15.1 Detailed Description

TCN-DNS Request telegram TCN_DNS_REQ_DS.

4.15.2 Field Documentation

4.15.2.1 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

4.16 TRDP ETB INFO T Struct Reference

Types for train configuration information.

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

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)

4.16.1 Detailed Description

Types for train configuration information.

ETB information

4.16.2 Field Documentation

4.16.2.1 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

4.17 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)

4.17.1 Detailed Description

function/device information structure

4.17.2 Field Documentation

4.17.2.1 cnld

```
UINT8 TRDP_FUNCTION_INFO_T::cnId
```

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

Value range: 0..31

4.17.2.2 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

4.17.2.3 etbld

UINT8 TRDP_FUNCTION_INFO_T::etbId

number of connected train backbone.

Value range: 0..3

4.17.2.4 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

4.18 TRDP IDX TABLE T Struct Reference

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

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

Data Fields

UINT32 maxNoOfLowCatSubscriptions

Max

• UINT32 maxNoOfMidCatSubscriptions

Max

UINT32 maxNoOfHighCatSubscriptions

Max

UINT32 maxNoOfLowCatPublishers

Мах.

UINT32 maxDepthOfLowCatPublishers

depth / overlapped publishers with intervals <= 100ms

UINT32 maxNoOfMidCatPublishers

Мах.

• UINT32 maxDepthOfMidCatPublishers

depth / overlapped publishers with intervals <= 1000ms

UINT32 maxNoOfHighCatPublishers

Мах.

UINT32 maxDepthOfHighCatPublishers

depth / overlapped publishers with intervals <= 10000ms

UINT32 maxNoOfExtPublishers

Мах.

4.18.1 Detailed Description

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

4.18.2 Field Documentation

4.18.2.1 maxNoOfExtPublishers

```
UINT32 TRDP_IDX_TABLE_T::maxNoOfExtPublishers
```

Max.

number of expected publishers with intervals > 10000ms

4.18.2.2 maxNoOfHighCatPublishers

UINT32 TRDP_IDX_TABLE_T::maxNoOfHighCatPublishers

Max.

number of expected publishers with intervals <= 10000ms

4.18.2.3 maxNoOfHighCatSubscriptions

UINT32 TRDP_IDX_TABLE_T::maxNoOfHighCatSubscriptions

Max.

number of expected subscriptions with intervals > 1000ms

4.18.2.4 maxNoOfLowCatPublishers

UINT32 TRDP_IDX_TABLE_T::maxNoOfLowCatPublishers

Max.

number of expected publishers with intervals <= 100ms

4.18.2.5 maxNoOfLowCatSubscriptions

UINT32 TRDP_IDX_TABLE_T::maxNoOfLowCatSubscriptions

Max.

number of expected subscriptions with intervals <= 100ms

4.18.2.6 maxNoOfMidCatPublishers

UINT32 TRDP_IDX_TABLE_T::maxNoOfMidCatPublishers

Max.

number of expected publishers with intervals <= 1000ms

4.18.2.7 maxNoOfMidCatSubscriptions

UINT32 TRDP_IDX_TABLE_T::maxNoOfMidCatSubscriptions

Max.

number of expected subscriptions with intervals <= 1000ms

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

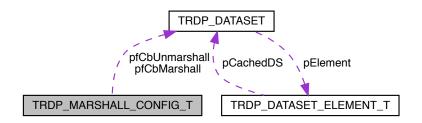
• trdp_types.h

4.19 TRDP_MARSHALL_CONFIG_T Struct Reference

Marshaling/unmarshalling configuration

#include <trdp_types.h>

Collaboration diagram for TRDP_MARSHALL_CONFIG_T:



Data Fields

- TRDP_MARSHALL_T pfCbMarshall
 - Pointer to marshall callback function
- TRDP_UNMARSHALL_T pfCbUnmarshall

Pointer to unmarshall callback function

void * pRefCon

Pointer to user context for call back

4.19.1 Detailed Description

Marshaling/unmarshalling configuration

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

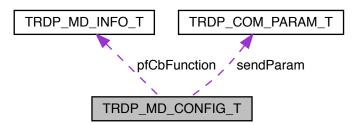
trdp_types.h

4.20 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

4.20.1 Detailed Description

Default MD configuration.

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

· trdp_types.h

4.21 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 comId
 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

4.21.1 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

4.22 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

4.22.1 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:

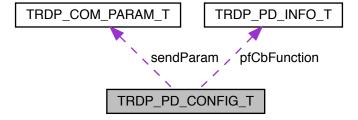
• trdp_types.h

4.23 TRDP_PD_CONFIG_T Struct Reference

Default PD configuration

#include <trdp_types.h>

Collaboration diagram for TRDP_PD_CONFIG_T:



• 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)

4.23.1 Detailed Description

Default PD configuration

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

· trdp_types.h

4.24 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 replyIpAddr

IP address for reply (request only)

const void * pUserRef

User reference given with the local subscribe

• TRDP_ERR_T resultCode

error code

• TRDP_URI_HOST_T srcHostURI

source URI host part (unused)

TRDP_URI_HOST_T destHostURI

destination URI host part (unused)

• TRDP_TO_BEHAVIOR_T toBehavior

callback can decide about handling of data on timeout

UINT32 serviceId

the reserved field of the PD header

4.24.1 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

4.25 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_T options

TRDP options.

4.25.1 Detailed Description

Various flags/general TRDP options for library initialization.

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

• trdp_types.h

4.26 TRDP_PROP_T Struct Reference

Application defined properties.

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

Data Fields

• TRDP_SHORT_VERSION_T ver

properties version information, application defined

UINT16 len

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

• UINT8 prop [1]

properties, application defined

4.26.1 Detailed Description

Application defined properties.

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

• tau_tti_types.h

4.27 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.

4.27.1 Detailed Description

Types to read out the XML configuration

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

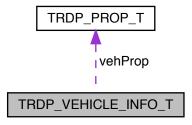
• tau_xml.h

4.28 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

4.28.1 Detailed Description

vehicle information structure

4.28.2 Field Documentation

4.28.2.1 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

4.29 TRDP_XML_DOC_HANDLE_T Struct Reference

Parsed XML document handle.

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

Data Fields

struct XML_HANDLE * pXmlDocument
 XML document context.

4.29.1 Detailed Description

Parsed XML document handle.

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

• tau_xml.h

4.30 VOS_SOCK_OPT_T Struct Reference

Common socket options

```
#include <vos_sock.h>
```

Data Fields

• UINT8 qos

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

4.30.1 Detailed Description

Common socket options

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

• vos_sock.h

4.31 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

4.31.1 Detailed Description

Version information.

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

• vos_types.h

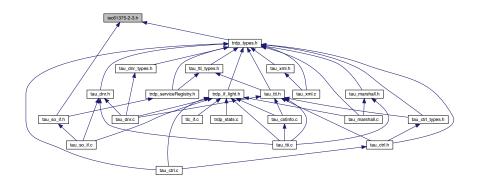
Chapter 5

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_PROTOCOL_VERSION_CHECK_MASK 0xFF00u Protocol version is defined in trdp_private.h.

#define TRDP_SESS_ID_SIZE 16u
 Session ID (UUID) size in MD header

• #define TRDP_USR_URI_SIZE 32u

max.

• #define TRDP_MD_INFINITE_TIME (0)

Definitions for time out behaviour accd.

#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 3000000u
 [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
 [us] 5s

#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
 MD reply

#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

#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"
 ETBx
- #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.lCst.lClTrn.lTrn"
 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
    [us] 5s
```

 #define ECSP_STATUS_DEST_URI "devECSC.anyVeh.lCst.lClTrn.lTrn" 10.0.0.100

• #define ECSP_CONF_REQ_COMID 122u

ECSP Confirmation Request telegram MD:

 #define ECSP_CONF_REQ_TO_US 3000000u [us]

#define ECSP_CONF_REQ_URI "devECSP.anyVeh.lCst.lClTrn.lTrn"
 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"
 ETBx

#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

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

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_PROTOCOL_VERSION_CHECK_MASK

#define TRDP_PROTOCOL_VERSION_CHECK_MASK 0xFF00u

Protocol version is defined in trdp_private.h.

Version check, two digits are relevant

5.1.2.16 TRDP_USR_URI_SIZE

#define TRDP_USR_URI_SIZE 32u

max.

User URI size in MD header

5.1.2.17 TTDB_NET_DIR_REQ_COMID

#define TTDB_NET_DIR_REQ_COMID 106u

TTDB manager telegram MD: Get the NETWORK_TRAIN_DIRECTORY

MD request

5.1.2.18 TTDB_OP_DIR_INFO_COMID

#define TTDB_OP_DIR_INFO_COMID 101u

TTDB manager telegram MD: Push the OP_TRAIN_DIRECTORY

MD notification

5.1.2.19 TTDB_STAT_CST_REQ_COMID

#define TTDB_STAT_CST_REQ_COMID 104u

TTDB manager telegram MD: Get the static consist information

MD request

5.1.2.20 TTDB_TRN_DIR_REQ_COMID

#define TTDB_TRN_DIR_REQ_COMID 102u

TTDB manager telegram MD: Get the TRAIN_DIRECTORY

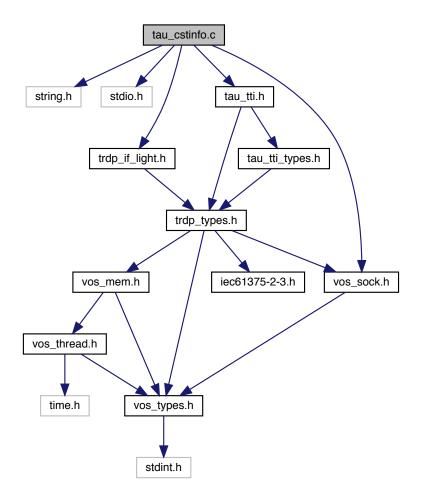
MD request

5.2 tau_cstinfo.c File Reference

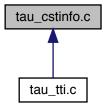
Functions for consist information access.

```
#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:



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



Functions

• UINT16 cstInfoGetPropSize (TRDP_CONSIST_INFO_T *pCstInfo)

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

5.2.1 Detailed Description

Functions for consist information access.

Note

Project: TCNOpen TRDP prototype stack

Author

B. Loehr (initial version)

Remarks

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

5.2.2 Function Documentation

5.2.2.1 cstInfoGetPropSize()

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

Parameters

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

Return values



Here is the call graph for this function:

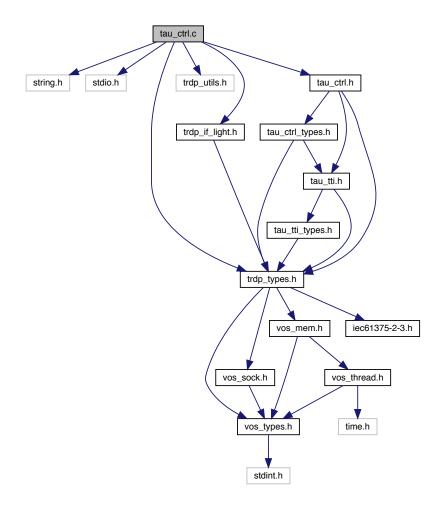


5.3 tau_ctrl.c File Reference

Functions for train switch control.

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

Include dependency graph for tau_ctrl.c:



Functions

• EXT_DECL_TRDP_ERR_T_tau_initEcspCtrl (TRDP_APP_SESSION_T_appHandle, TRDP_IP_ADDR_T ecsplpAddr)

Function to init ECSP control interface.

• EXT_DECL TRDP_ERR_T tau_terminateEcspCtrl (TRDP_APP_SESSION_T appHandle)

Function to close ECSP control interface.

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

Function to set ECSP control information.

• EXT_DECL TRDP_ERR_T tau_getEcspStat (TRDP_APP_SESSION_T appHandle, TRDP_ECSP_STAT_T *pEcspStat, TRDP PD INFO T *pPdInfo)

Function to get ECSP status information.

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

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

5.3.1 Detailed Description

Functions for train switch control.

Note

Project: TCNOpen TRDP prototype stack

Author

Armin-H. Weiss (initial version)

Remarks

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

5.3.2 Function Documentation

5.3.2.1 tau_getEcspStat()

Function to get ECSP status information.

Parameters

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

Return values

TRDP_NO_ERR	no error
TRDP_NOINIT_ERR	module not initialised
TRDP_PARAM_ERR	Parameter error

5.3.2.2 tau_initEcspCtrl()

```
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 reply
Ī	in	pfCbFunction	Pointer to callback function, NULL for default
ſ	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

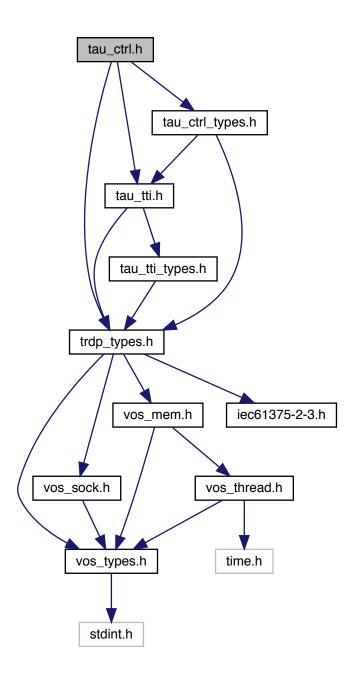
TRDP_NO_ERR	no error
TRDP_NOINIT_ERR	module not initialised
TRDP_UNKNOWN_ERR	undefined error

5.4 tau_ctrl.h File Reference

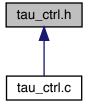
TRDP utility interface definitions.

```
#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_T ecsplpAddr)

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()

```
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
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

Parameters

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

Return values

TRDP_NO_ERR	no error
TRDP_NOINIT_ERR	module not initialised
TRDP_PARAM_ERR	Parameter error

5.4.2.4 tau_setEcspCtrl()

Function to set ECSP control information.

Parameters

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

Return values

TRDP_NO_ERR	no error
TRDP_NOINIT_ERR	module not initialised
TRDP_PARAM_ERR	Parameter error

5.4.2.5 tau_terminateEcspCtrl()

Function to close ECSP control interface.

Parameters

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

Return values

TRDP_NO_ERR	no error
TRDP_UNKNOWN_ERR	undefined error

Parameters

	I	
in	appHandle	Application handle

Return values

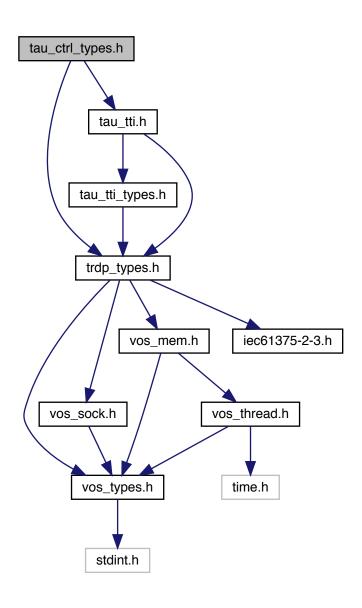
TRDP_NO_ERR	no error
TRDP_NOINIT_ERR	module not initialised
TRDP_UNKNOWN_ERR	undefined error

5.5 tau_ctrl_types.h File Reference

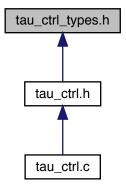
TRDP utility interface definitions.

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

Include dependency graph for tau_ctrl_types.h:



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



Data Structures

struct GNU_PACKED

Types for ETB control.

• struct GNU_PACKED

Types for ETB control.

• struct GNU_PACKED

Types for ETB control.

5.5.1 Detailed Description

TRDP utility interface definitions.

This module provides the interface to the following

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

Note

Project: TCNOpen TRDP prototype stack

Author

Armin-H. Weiss (initial version)

Remarks

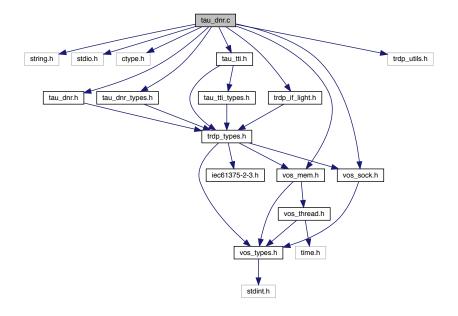
This Source Code Form is subject to the terms of the Mozilla Public License, v. 2.0. If a copy of the MPL was not distributed with this file, You can obtain one at http://mozilla.org/MPL/2.0/. Copyright Bombardier Transportation Inc. or its subsidiaries and others, 2013. All rights reserved.

5.6 tau_dnr.c File Reference

Functions for domain name resolution.

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

Include dependency graph for tau_dnr.c:



Data Structures

struct DNS HEADER

DNS header structure.

Macros

• #define TAU MAX NO IF 4u

Default interface should be in the first 4.

#define TAU_DNS_TIME_OUT_LONG 10u

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

· #define TAU DNS TIME OUT SHORT 1u

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

Typedefs

 typedef struct DNS_HEADER_TAU_DNS_HEADER_T DNS header structure.

Functions

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

Function to init the DNR subsystem Initialize the DNR resolver.

• EXT_DECL void tau_delnitDnr (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. iai.ai	' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' '

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

Return values

TRDP_NO_ERR	no error
TRDP_INIT_ERR	initialisation error

< default DNR/ECSP settings

5.6.2.6 tau_uri2Addr()

Function to convert a URI to an IP address.

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

Parameters

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

Return values

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	Parameter error

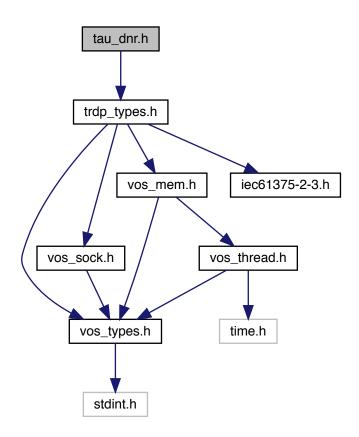
Return values

TRDP_UNRESOLVED_ERR	Could not resolve error
TRDP_TOPO_ERR	Cache/DB entry is invalid

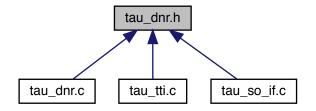
5.7 tau_dnr.h File Reference

TRDP utility interface definitions.

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



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



Typedefs

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

Enumerations

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

Functions

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

Function to init DNR.

• EXT_DECL void tau_deInitDnr (TRDP_APP_SESSION_T appHandle)

Release any resources allocated by DNR.

- EXT_DECL TRDP_DNR_STATE_T tau_DNRstatus (TRDP_APP_SESSION_T appHandle)
 - Function to get the status of DNR.
- EXT_DECL TRDP_IP_ADDR_T tau_getOwnAddr (TRDP_APP_SESSION_T appHandle)
 - Function to get the own IP address.
- EXT_DECL TRDP_ERR_T tau_uri2Addr (TRDP_APP_SESSION_T appHandle, TRDP_IP_ADDR_T *p
 — Addr, const TRDP_URI_T pUri)

Function to convert a URI to an IP address.

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

Function to convert an IP address to a URI.

5.7.1 Detailed Description

TRDP utility interface definitions.

This module provides the interface to the following utilities

• IP - URI address translation

Note

Project: TCNOpen TRDP prototype stack

Author

Armin-H. Weiss (initial version)

Remarks

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

5.7.2 Enumeration Type Documentation

5.7.2.1 TRDP_DNR_OPTS

```
enum TRDP_DNR_OPTS
```

DNR options.

Enumerator

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

5.7.3 Function Documentation

5.7.3.1 tau_addr2Uri()

```
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_deInitDnr()

Release any resources allocated by DNR.

Parameters

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

none	Release any resources allocated by DNR.

Parameters

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

Return values

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	Parameter error

5.7.3.3 tau_DNRstatus()

```
EXT_DECL TRDP_DNR_STATE_T tau_DNRstatus (

TRDP_APP_SESSION_T appHandle )
```

Function to get the status of DNR.

Parameters

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

Return values

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

5.7.3.4 tau_getOwnAddr()

Function to get the own IP address.

Parameters

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

Return values

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

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

Parameters

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

Return values

```
own IP address
```

5.7.3.5 tau_initDnr()

```
EXT_DECL TRDP_ERR_T tau_initDnr (

TRDP_APP_SESSION_T appHandle,

TRDP_IP_ADDR_T dnsIpAddr,

UINT16 dnsPort,

const CHAR8 * pHostsFileName,

TRDP_DNR_OPTS_T dnsOptions,

BOOL8 waitForDnr )
```

Function to init DNR.

Parameters

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

Return values

TRDP_NO_ERR	no error
TRDP_INIT_ERR	initialisation error

Function to init DNR.

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

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

Parameters

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

Parameters

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

Return values

TRDP_NO_ERR	no error
TRDP_INIT_ERR	initialisation error

< default DNR/ECSP settings

5.7.3.6 tau_uri2Addr()

Function to convert a URI to an IP address.

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

Parameters

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

Return values

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	Parameter error

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

Parameters

i	.n	appHandle	Handle returned by tlc_openSession()
0	ut	pAddr	Pointer to return the IP address
i	.n	pUri	Pointer to an URI or an IP Address string, NULL==own URI

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	Parameter error

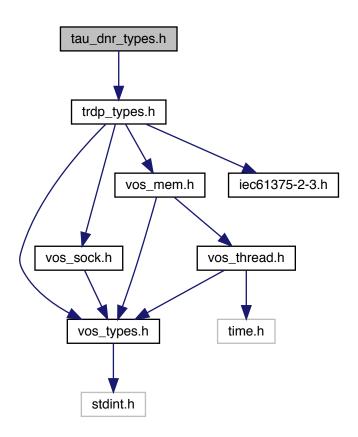
Return values

TRDP_UNRESOLVED_ERR	Could not resolve error
TRDP_TOPO_ERR	Cache/DB entry is invalid

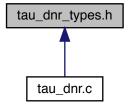
5.8 tau_dnr_types.h File Reference

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

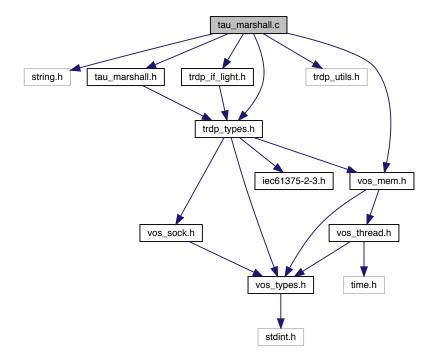
This Source Code Form is subject to the terms of the Mozilla Public License, v. 2.0. If a copy of the MPL was not distributed with this file, You can obtain one at http://mozilla.org/MPL/2.0/. Copyright NewTec GmbH, 2017. All rights reserved.

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.

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_calcDatasetSizeByComld()

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

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

Return values

TRDP_COMID_ERR	comid not existing
TRDP_MARSHALLING_ERR	dataset/source size mismatch

5.9.2.3 tau_initMarshall()

Function to initialise the marshalling/unmarshalling.

Types for marshalling / unmarshalling

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

Parameters

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

Return values

TRDP_NO_ERR	no error
TRDP_MEM_ERR	provided buffer to small
TRDP_PARAM_ERR	Parameter error

5.9.2.4 tau_marshall()

marshall function.

Parameters

in	pRefCon	pointer to user context
in	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

5.9.2.5 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

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

Return values

TRDP_STATE_ERR	Too deep recursion
TRDP_COMID_ERR	comid not existing
TRDP_MARSHALLING_ERR	dataset/source size mismatch

5.9.2.6 tau_unmarshall()

unmarshall 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_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()

```
UINT8 * pSrc,
UINT32 srcSize,
UINT8 * pDest,
UINT32 * pDestSize,
TRDP_DATASET_T ** ppDSPointer )
```

unmarshall data set function.

Parameters

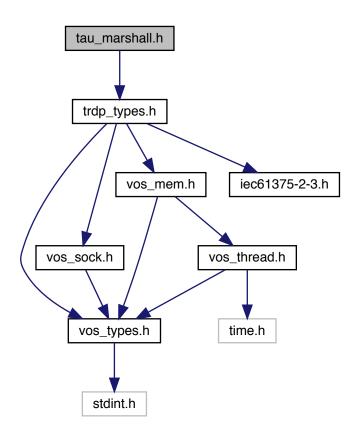
in	pRefCon	pointer to user context
in	dsld	Data set id to identify the structure out of a configuration
in	pSrc	pointer to received original message
in	srcSize	size of the source buffer
in	pDest	pointer to a buffer for the treated message
in,out	pDestSize	size of the provide buffer / size of the treated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

Return values

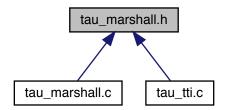
TRDP_INIT_ERR	marshalling not initialised
TRDP_NO_ERR	no error
TRDP_MEM_ERR	provided buffer to small
TRDP_PARAM_ERR	Parameter error
TRDP_STATE_ERR	Too deep recursion
TRDP_COMID_ERR	comid not existing
TRDP_MARSHALLING_ERR	dataset/source size mismatch

5.10 tau_marshall.h File Reference

#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[])

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)
 marshall function.

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)

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.

unmarshall data set function.

• 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.

5.10.1 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_calcDatasetSizeByComld()

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
		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

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

5.10.2.4 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()

```
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.

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()

```
UINT8 * pDest,
UINT32 * pDestSize,
TRDP_DATASET_T ** ppDSPointer )
```

unmarshall 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

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_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()

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

Parameters

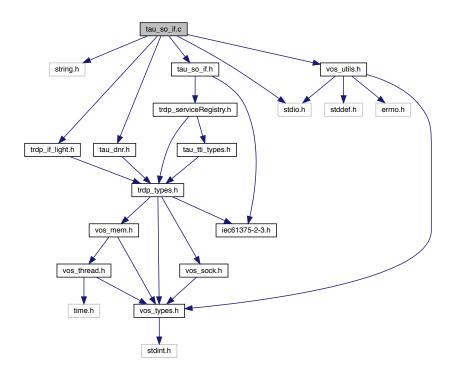
in	pRefCon	pointer to user context
in	dsld	Data set id to identify the structure out of a configuration
in	pSrc	pointer to received original message
in	srcSize	size of the source buffer
in	pDest	pointer to a buffer for the treated message
in,out	pDestSize	size of the provide buffer / size of the treated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

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

5.11 tau so if.c File Reference

Access to service oriented functions of the SRM.

```
#include <string.h>
#include <stdio.h>
#include "trdp_if_light.h"
#include "tau_dnr.h"
#include "tau_so_if.h"
#include dependency graph for tau_so_if.c:
```



Functions

EXT_DECL TRDP_ERR_T tau_addService (TRDP_APP_SESSION_T appHandle, SRM_SERVICE_INFO_T *pServiceToAdd, BOOL8 waitForCompletion)

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

• EXT_DECL TRDP_ERR_T tau_delService (TRDP_APP_SESSION_T appHandle, SRM_SERVICE_INFO_T *pServiceToRemove, BOOL8 waitForCompletion)

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

 EXT_DECL TRDP_ERR_T tau_updService (TRDP_APP_SESSION_T appHandle, SRM_SERVICE_INFO_T *pServiceToUpdate, BOOL8 waitForCompletion)

Register an update a service.

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

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

EXT_DECL void tau_freeServicesList (SRM_SERVICE_ENTRIES_T *pServicesListBuffer)

Release the memory of a list received by tau_getServiceList.

5.11.1 Detailed Description

Access to service oriented functions of the SRM.

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

Note

Project: TCNOpen TRDP prototype stack

Author

B. Loehr (initial version)

Remarks

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

5.11.2 Function Documentation

5.11.2.1 tau addService()

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

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

Parameters

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

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

5.11.2.2 tau_delService()

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

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

Parameters

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

Return values

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

5.11.2.3 tau_freeServicesList()

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

Release the memory of a list received by tau_getServiceList.

Parameters

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

Return values

none

5.11.2.4 tau_getServicesList()

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

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

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

Parameters 4 8 1

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

Return values

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

5.11.2.5 tau_updService()

Register an update a service.

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

Parameters

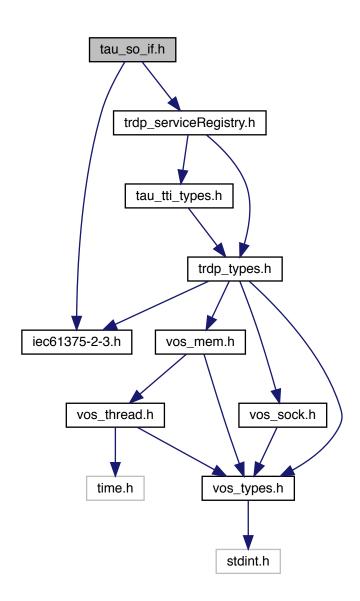
in	appHandle	Handle returned by tlc_openSession().
in,out	pServiceToUpdate	Pointer to a service registry structure to be updated
in	waitForCompletion	if true, block for reply

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

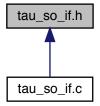
5.12 tau_so_if.h File Reference

Access to the Service Registry.

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



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



Functions

EXT_DECL TRDP_ERR_T tau_addService (TRDP_APP_SESSION_T appHandle, SRM_SERVICE_INFO_T *pServiceToAdd, BOOL8 waitForCompletion)

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

 EXT_DECL TRDP_ERR_T tau_delService (TRDP_APP_SESSION_T appHandle, SRM_SERVICE_INFO_T *pServiceToAdd, BOOL8 waitForCompletion)

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

 EXT_DECL TRDP_ERR_T tau_updService (TRDP_APP_SESSION_T appHandle, SRM_SERVICE_INFO_T *pServiceToAdd, BOOL8 waitForCompletion)

Register an update a service.

- EXT_DECL TRDP_ERR_T tau_getServicesList (TRDP_APP_SESSION_T appHandle, SRM_SERVICE_← ENTRIES_T **ppServicesToAdd, UINT32 *pNoOfServices, SRM_SERVICE_ENTRIES_T *pFilterEntry)
 - Get a list of the services known by the service registry of the local TTDB / SRM.

• EXT_DECL void tau_freeServicesList (SRM_SERVICE_ENTRIES_T *pServicesListBuffer)

Release the memory of a list received by tau_getServiceList.

5.12.1 Detailed Description

Access to the Service Registry.

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

Note

Project: TCNOpen TRDP prototype stack & FDF/DbD

Author

Bernd Loehr, NewTec GmbH, 2019-06-17

Remarks

Copyright 2019, NewTec GmbH

ld

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

5.12.2 Function Documentation

5.12.2.1 tau_addService()

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

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

Parameters

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

Return values

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

5.12.2.2 tau_delService()

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

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

Parameters

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

Return values

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

5.12.2.3 tau_freeServicesList()

```
EXT_DECL void tau_freeServicesList ( {\tt SRM\_SERVICE\_ENTRIES\_T~*~pServicesListBuffer~)}
```

Release the memory of a list received by tau_getServiceList.

Parameters

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

Return values

none

5.12.2.4 tau_getServicesList()

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

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

Parameters

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

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	Parameter error

Return values

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

5.12.2.5 tau_updService()

Register an update a service.

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

Parameters

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

Return values

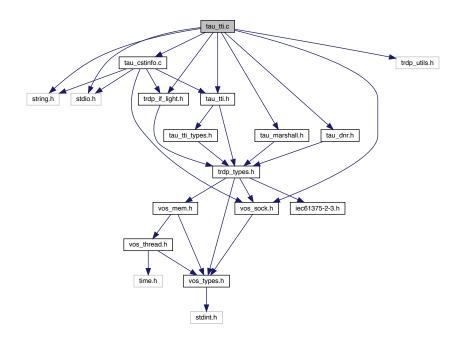
TRDP_NO_ERR	no error
TRDP_PARAM_ERR	Parameter error
TRDP_NODATA_ERR	Data currently not available, try again later
TRDP_TIMEOUT_ERR	Reply timed out
TRDP_SEMA_ERR	Semaphore could not be aquired

5.13 tau_tti.c File Reference

Functions for train topology information access.

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

Include dependency graph for tau_tti.c:



Macros

• #define TTI_CACHED_CONSISTS 8u

We hold this number of consist infos (ca.

Functions

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

Function to init TTI access.

• EXT_DECL void tau_deInitTTI (TRDP_APP_SESSION_T appHandle)

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

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

Function to retrieve the operational train directory state.

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

Function to retrieve the operational train directory state info.

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

Function to retrieve the train directory.

EXT_DECL TRDP_ERR_T tau_getStaticCstInfo (TRDP_APP_SESSION_T appHandle, TRDP_CONSIST_INFO_T *pCstInfo, TRDP_UUID_T const cstUUID)

Function to retrieve the consist info.

• EXT_DECL TRDP_ERR_T tau_getTTI (TRDP_APP_SESSION_T appHandle, TRDP_OP_TRAIN_DIR_S

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

_TRAIN_NET_DIR_T *pTrnNetDir)

Function to retrieve the operational train directory.

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

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

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

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

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

 EXT_DECL TRDP_ERR_T tau_getCstFctInfo (TRDP_APP_SESSION_T appHandle, TRDP_FUNCTION_INFO_T *pFctInfo, const TRDP_LABEL_T pCstLabel, UINT16 maxFctCnt)

Function to retrieve the function information of the consist.

EXT_DECL TRDP_ERR_T tau_getVehInfo (TRDP_APP_SESSION_T appHandle, TRDP_VEHICLE_INFO_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_INFO_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.

EXT_DECL TRDP_ERR_T tau_getOwnlds (TRDP_APP_SESSION_T appHandle, TRDP_LABEL_T *p
 — DevId, TRDP_LABEL_T *pVehId, TRDP_LABEL_T *pCstId)

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, 2016-2020. 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

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

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	Parameter error

5.13.3.5 tau_getCstVehCnt()

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

Parameters

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

Return values

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	Parameter error
TRDP_NODATA_ERR	Try again

5.13.3.6 tau_getOpTrDirectory()

Function to retrieve the operational train directory state.

Parameters

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

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

5.13.3.7 tau_getOpTrnDirectoryStatusInfo()

Function to retrieve the operational train directory state info.

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

Parameters

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

Return values

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	Parameter error

5.13.3.8 tau_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	pDevId	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

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

Return values

ownOpCstNo own opera	tional consist number value 0 on error
----------------------	--

5.13.3.10 tau_getOwnTrnCstNo()

Get own train consist number.

Parameters

in appHandle The handle returne	d by tlc_init
---------------------------------	---------------

Return values

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

5.13.3.11 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 consist info.

Function to retrieve the operational train directory.

Parameters

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

Return values

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	Parameter error

5.13.3.12 tau_getTrDirectory()

Function to retrieve the train directory.

Parameters

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

Return values

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	Parameter error
TRDP_NODATA_ERR	Try later

5.13.3.13 tau_getTrnCstCnt()

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

Parameters

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

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	Parameter error
TRDP_NODATA_ERR	Try again

5.13.3.14 tau_getTrnVehCnt()

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

Parameters

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

Return values

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	Parameter error
TRDP_NODATA_ERR	Try again

5.13.3.15 tau_getTTI()

```
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()

```
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.

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()

```
{\tt 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	plpAddr ECSP IP address. Currently not used.	
in	hostsFileName	Optional host file name as ECSP replacement. Currently not implemented.	

Return values

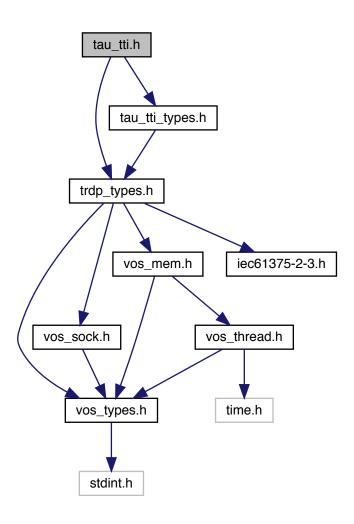
TRDP_NO_ERR	no error
TRDP_INIT_ERR	initialisation error

5.14 tau_tti.h File Reference

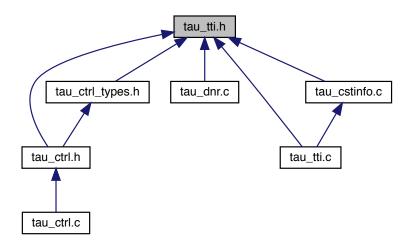
TRDP utility interface definitions.

```
#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.

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 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_INFO_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_INFO_T *pCstInfo, const TRDP_LABEL_T pCstLabel)

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

• EXT_DECL TRDP_ERR_T tau_getVehOrient (TRDP_APP_SESSION_T appHandle, UINT8 *pVehOrient, UINT8 *pCstOrient, TRDP_LABEL_T pVehLabel, TRDP_LABEL_T pCstLabel)

Function to retrieve the orientation of the given vehicle.

Who am I?.

• EXT_DECL UINT8 tau_getOwnOpCstNo (TRDP_APP_SESSION_T appHandle)

Get own operational consist number.

• EXT_DECL UINT8 tau_getOwnTrnCstNo (TRDP_APP_SESSION_T appHandle)

Get own train consist number.

5.14.1 Detailed Description

TRDP utility interface definitions.

This module provides the interface to the following utilities

· train topology information access

Note

Project: TCNOpen TRDP prototype stack

Author

Armin-H. Weiss (initial version)

Remarks

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

5.14.2 Function Documentation

5.14.2.1 tau_deInitTTI()

Function to terminate TTI access.

Parameters

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

Return values

Parameters

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

Return values

```
none
```

5.14.2.2 tau_getCstFctCnt()

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

Parameters

	in	appHandle	Handle returned by tlc_openSession().	
	out	pCstFctCnt Pointer to the number of functions to be returned		
ſ	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()

```
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	xFctCnt 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	andle 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 cons			

Return values

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	Parameter error

5.14.2.5 tau_getCstVehCnt()

```
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.

Parameters

in appHandle Handle returned by tlc_openSession().		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 con-		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 cons		Pointer to a consist label. NULL means own consist.	

Return values

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	Parameter error
TRDP_NODATA_ERR	Try again

5.14.2.6 tau_getOpTrDirectory()

Function to retrieve the operational train directory state.

Parameters

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

Return values

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	Parameter error

Parameters

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

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

5.14.2.7 tau_getOpTrnDirectoryStatusInfo()

Function to retrieve the operational train directory state info.

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

Parameters

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

Return values

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	Parameter error

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

Parameters

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

Return values

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	Parameter error

5.14.2.8 tau_getOwnlds()

```
EXT_DECL TRDP_ERR_T tau_getOwnIds (

TRDP_APP_SESSION_T appHandle,

TRDP_LABEL_T * pDevId,

TRDP_LABEL_T * pVehId,

TRDP_LABEL_T * pCstId )
```

Who am I?.

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

Parameters

in	appHandle	Handle returned by tlc_openSession()
out	pDevId	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

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

Return values

ownOpCstNo own operational consist number value 0 on en	or
---	----

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().		Handle returned by tlc_openSession().	
out pCstInfo Pointer to a consist info structure to be returned		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

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

Return values

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	Parameter error

Parameters

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

Return values

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	Parameter error

5.14.2.16 tau_getVehInfo()

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

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

Parameters

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

Return values

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	Parameter error

5.14.2.17 tau_getVehOrient()

Function to retrieve the orientation of the given vehicle.

Parameters

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

Return values

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	Parameter error

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

Parameters

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

5.14.2.18 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.

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

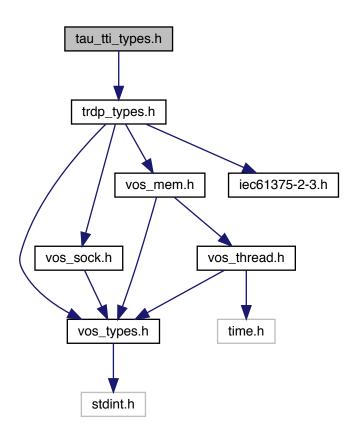
ſ	in	appHandle Handle returned by tlc_openSession().	
Ī	in	userAction Semaphore to fire if inauguration took place.	
Ī	in	ecsplpAddr	ECSP IP address. Currently not used.
Ī	in	n hostsFileName Optional host file name as ECSP replacement. Currently not implemente	

TRDP_NO_ERR	no error
TRDP_INIT_ERR	initialisation error

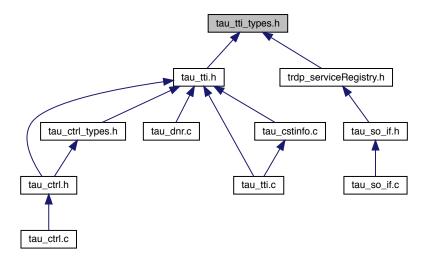
5.15 tau_tti_types.h File Reference

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.

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

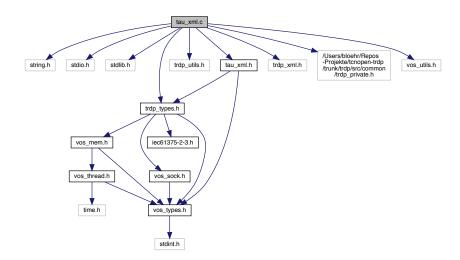
This Source Code Form is subject to the terms of the Mozilla Public License, v. 2.0. If a copy of the MPL was not distributed with this file, You can obtain one at http://mozilla.org/MPL/2.0/. Copyright Bombardier Transportation Inc. or its subsidiaries and others, 2014. All rights reserved.

tau xml.c File Reference 5.16

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_HANDLE_T *pDocHnd)

Open XML stream, prepare XPath context.

EXT_DECL void tau_freeXmlDoc (TRDP_XML_DOC_HANDLE_T *pDocHnd)

Free all the memory allocated by tau_prepareXmlDoc.

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

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

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

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

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

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

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

 EXT_DECL_TRDP_ERR_T tau_readXmlMappedDeviceConfig (const_TRDP_XML_DOC_HANDLE_T *p→ DocHnd, const CHAR8 *pHostname, UINT32 *pNumlfConfig, TRDP_IF_CONFIG_T **pplfConfig)

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

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

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

 EXT_DECL_TRDP_ERR_T tau_readXmlDatasetConfig (const_TRDP_XML_DOC_HANDLE_T *pDoc← Hnd, UINT32 *pNumComld, TRDP_COMID_DSID_MAP_T **ppComldDsIdMap, UINT32 *pNumDataset, apTRDP_DATASET_T *apDataset)

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

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

Function to free the memory for the DataSet configuration.

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

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

5.16.1 Detailed Description

Functions for XML file parsing.

SOX parsing of XML configuration file

Note

Project: TCNOpen TRDP prototype stack

Author

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

Remarks

This Source Code Form is subject to the terms of the Mozilla Public License, v. 2.0. If a copy of the MPL was not distributed with this file, You can obtain one at http://mozilla.org/MPL/2.0/. Copyright NewTec GmbH, 2016-2020. 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	
Ī	in pComldDsldMap Pointer to an array of structures of type TRDP_COMID_DSID_		Pointer to an array of structures of type TRDP_COMID_DSID_MAP_T	
Ī	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

ir	pDocHnd	Handle of the parsed XML file
----	---------	-------------------------------

5.16.3.4 tau_prepareXmIDoc()

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

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	File does not exist

5.16.3.5 tau_prepareXmIMem()

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

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 .

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
Generated b	y polyygaExchgPar	Number of configured telegrams
out	ppExchgPar	Pointer to array of telegram configurations

Return values

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

5.16.3.9 tau_readXmlMappedDeviceConfig()

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

Parameters

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

Return values

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

5.16.3.10 tau_readXmlMappedDevices()

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

Parameters

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

Return values

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

5.16.3.11 tau_readXmlMappedInterfaceConfig()

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

Parameters

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

Return values

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

5.16.3.12 tau_readXmlServiceConfig()

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

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

Parameters

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

Generated by Doxygen

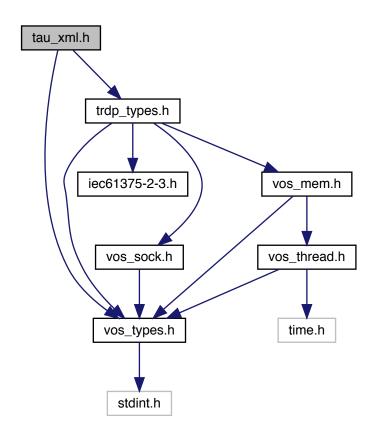
Return values

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

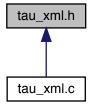
5.17 tau_xml.h File Reference

TRDP utility interface definitions.

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



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



Data Structures

• struct TRDP_SDT_PAR_T

Types to read out the XML configuration

• struct TRDP_DBG_CONFIG_T

Control for debug output device/file on application level.

struct TRDP_XML_DOC_HANDLE_T

Parsed XML document handle.

Macros

• #define TRDP_DBG_DEFAULT 0

Control for debug output format on application level.

• #define TRDP_DBG_OFF 0x01

Printout off

#define TRDP_DBG_ERR 0x02

Printout error.

• #define TRDP_DBG_WARN 0x04

Printout warning and error.

• #define TRDP_DBG_INFO 0x08

Printout info, warning and error.

• #define TRDP_DBG_DBG 0x10

Printout debug, info, warning and error.

• #define TRDP_DBG_TIME 0x20

Printout timestamp.

• #define TRDP_DBG_LOC 0x40

Printout file name and line.

• #define TRDP_DBG_CAT 0x80

Printout category (DBG, INFO, WARN, ERR)

Enumerations

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

Type attribute for telegrams.

Functions

EXT_DECL TRDP_ERR_T tau_prepareXmlDoc (const CHAR8 *pFileName, TRDP_XML_DOC_HANDLE_T *pDocHnd)

Load XML file into DOM tree, prepare XPath context.

• EXT_DECLTRDP_ERR_T tau_prepareXmlMem (char *pBuffer, size_t bufSize, TRDP_XML_DOC_HANDLE_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 *pDoc← Hnd, UINT32 *pNumComld, TRDP_COMID_DSID_MAP_T **ppComldDsIdMap, UINT32 *pNumDataset, papTRDP_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.

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

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

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

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

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

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

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

Return values

none

5.17.4.3 tau_freeXmlDoc()

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

Free all the memory allocated by tau_prepareXmlDoc.

Parameters

	in	pDocHnd	Handle of the parsed XML file
--	----	---------	-------------------------------

5.17.4.4 tau_prepareXmIDoc()

Load XML file into DOM tree, prepare XPath context.

Parameters

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

Return values

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	File does not exist

Load XML file into DOM tree, prepare XPath context.

Parameters

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

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	File does not exist

5.17.4.5 tau_prepareXmIMem()

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.

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	papDataset	Pointer to an array of pointers to a structures of type TRDP_DATASET_T

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)

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	pNumIfConfig	Number of configured interfaces
out	pplfConfig	Pointer to an array of interface parameter sets

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_readXmlMappedDeviceConfig()

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

Parameters

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

Return values

TRDP_MEM_ERR	provided buffer to small
TRDP_PARAM_ERR	File not existing

5.17.4.10 tau_readXmlMappedDevices()

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

Parameters

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

Return values

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

5.17.4.11 tau_readXmlMappedInterfaceConfig()

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

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

Return values

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

5.17.4.12 tau_readXmlServiceConfig()

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

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

Parameters

in pDocHnd Handle of the XML document prepared by tau_prepareXm		Handle of the XML document prepared by tau_prepareXmlDoc	
ĺ	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

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

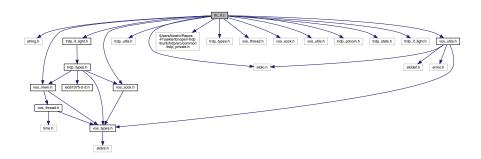
Return values

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

5.18 tlc_if.c File Reference

Functions for ECN communication.

```
#include <string.h>
#include "trdp_if_light.h"
#include "trdp_utils.h"
#include "trdp_pdcom.h"
#include "trdp_stats.h"
#include "vos_sock.h"
#include "vos_mem.h"
#include "vos_utils.h"
Include dependency graph for tlc if.c:
```



Functions

• BOOL8 trdp_isValidSession (TRDP_APP_SESSION_T pSessionHandle)

Check if the session handle is valid.

TRDP_APP_SESSION_T * trdp_sessionQueue (void)

Get the session queue head pointer.

• TRDP_ERR_T trdp_getAccess (TRDP_APP_SESSION_T appHandle, int force)

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

• void trdp_releaseAccess (TRDP_APP_SESSION_T appHandle)

Release access to the session.

- 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 *pPdDefault, const TRDP_PROCESS_CONFIG_T *pProcessConfig)

Open a session with the TRDP stack.

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.

• EXT_DECL TRDP_ERR_T tlc_updateSession (TRDP_APP_SESSION_T appHandle)

Update a session.

• EXT_DECL TRDP_ERR_T tlc_presetIndexSession (TRDP_APP_SESSION_T appHandle, TRDP_IDX_TABLE_T *pIndexTableSizes)

Preset the index table sizes of a session.

• EXT_DECL TRDP_ERR_T tlc_closeSession (TRDP_APP_SESSION_T appHandle)

Close a session.

• EXT_DECL TRDP_ERR_T tlc_terminate (void)

Un-Initialize.

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

Re-Initialize.

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

Get the lowest time interval for PDs.

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

Work loop of the TRDP handler.

EXT_DECL const char * tlc_getVersionString (void)

Return a human readable version representation.

• EXT_DECL const TRDP_VERSION_T * tlc_getVersion (void)

Return version

EXT_DECL TRDP_ERR_T tlc_setETBTopoCount (TRDP_APP_SESSION_T appHandle, UINT32 etbTopo ← Cnt)

Set new topocount for trainwide communication.

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

• EXT DECL UINT32 tlc getETBTopoCount (TRDP APP SESSION TappHandle)

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

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

Return values

TRDP_NO_ERR	no error
TRDP_INIT_ERR	not yet inited
TRDP_PARAM_ERR	parameter error

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

Γ	in	appHandle The handle returned by tlc_openSession	
in appHandle The handle returned by tlc_openSession		The naticie returned by tic_opensession	
	out plnterval pointer to needed interval		pointer to needed interval
Ī	in, out <i>pFileDesc</i> pointer to file descriptor set		pointer to file descriptor set
out pNoDesc pointer to put no of highest used descriptors (for se		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!

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

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

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

Return values

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

5.18.2.11 tlc_presetIndexSession()

Preset the index table sizes of a session.

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

Parameters

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

Return values

TRDP_NO_ERR	no error
TRDP_INIT_ERR	not yet inited
TRDP_PARAM_ERR	parameter error

5.18.2.12 tlc_process()

```
EXT_DECL TRDP_ERR_T tlc_process (

TRDP_APP_SESSION_T appHandle,
```

```
TRDP_FDS_T * pRfds,
INT32 * pCount )
```

Work loop of the TRDP handler.

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

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

Also see User Manual.

Parameters

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

Return values

TRDP_NO_ERR	no error
TRDP_NOINIT_ERR	handle invalid

5.18.2.13 tlc_reinitSession()

Re-Initialize.

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

Parameters

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

TRDP_NO_ERR	no error
TRDP_NOINIT_ERR	handle invalid
TRDP_PARAM_ERR	handle NULL

5.18.2.14 tlc_setETBTopoCount()

Set new topocount for trainwide communication.

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

Parameters

in	appHandle	the handle returned by tlc_openSession
in	etbTopoCnt	New etbTopoCnt value

Return values

TRDP_NO_ERR	no error
TRDP_NOINIT_ERR	handle invalid

5.18.2.15 tlc_setOpTrainTopoCount()

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

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

Parameters

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

Return values

TRDP_NO_ERR	no error
TRDP_NOINIT_ERR	handle invalid

5.18.2.16 tlc_terminate()

Un-Initialize.

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

Return values

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

5.18.2.17 tlc_updateSession()

Update a session.

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

Parameters

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

Return values

TRDP_NO_ERR	no error
TRDP_INIT_ERR	not yet inited
TRDP_PARAM_ERR	parameter error

5.18.2.18 trdp_getAccess()

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

Parameters

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

TRDP_NO_ERR	
TRDP_INIT_ERR	
TRDP_MUTEX_ERR	

5.18.2.19 trdp_isValidSession()

```
BOOL8 trdp_isValidSession ( {\tt TRDP\_APP\_SESSION\_T}\ pSession{\tt Handle}\ )
```

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	

5.18.2.20 trdp_releaseAccess()

Release access to the session.

Parameters

in	appHandle	A handle for further calls to the trdp stack

Return values



Here is the call graph for this function:



5.18.2.21 trdp_sessionQueue()

```
TRDP_APP_SESSION_T * trdp_sessionQueue ( \mbox{void} \quad \mbox{)}
```

Get the session queue head pointer.

Return values

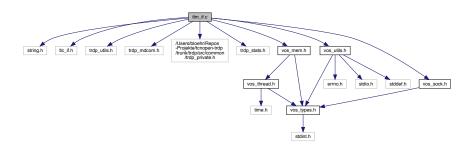
&sSession

5.19 tlm if.c File Reference

Functions for Message Data Communication.

```
#include <string.h>
#include "tlc_if.h"
#include "trdp_utils.h"
#include "trdp_mdcom.h"
#include "trdp_stats.h"
#include "vos_sock.h"
#include "vos_mem.h"
#include "vos_utils.h"
```

Include dependency graph for tlm if.c:



Functions

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

Get the lowest time interval for MDs.

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

Message Data Work loop of the TRDP handler.

• EXT_DECL TRDP_ERR_T tlm_notify (TRDP_APP_SESSION_T appHandle, const void *pUserRef, TRDP_MD_CALLBACK_T pfCbFunction, UINT32 comId, UINT32 etbTopoCnt, UINT32 opTrnTopo← Cnt, 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_USE← R_T srcURI, 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, 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 srcURI, const TRDP_URI_USER_T destURI)

Initiate sending MD request message.

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

Subscribe to MD messages.

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

Remove Listener.

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

Resubscribe to MD messages.

• EXT_DECL TRDP_ERR_T tlm_reply (TRDP_APP_SESSION_T appHandle, const TRDP_UUID_T *p↔ SessionId, UINT32 comId, UINT16 userStatus, const TRDP_SEND_PARAM_T *pSendParam, const UINT8 *pData, UINT32 dataSize)

Send a MD reply message.

EXT_DECL_TRDP_ERR_T tlm_replyQuery (TRDP_APP_SESSION_T appHandle, const TRDP_UUID_T *pSessionId, UINT32 comId, UINT16 userStatus, UINT32 confirmTimeout, const TRDP_SEND_PARAM_T *pSendParam, const UINT8 *pData, UINT32 dataSize)

Send a MD reply query message.

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

Initiate sending MD confirm message.

 EXT_DECL TRDP_ERR_T tlm_abortSession (TRDP_APP_SESSION_T appHandle, const TRDP_UUID_T *pSessionId)

Cancel an open session.

5.19.1 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, 2013-2020. All rights reserved.

5.19.2 Function Documentation

5.19.2.1 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	n p← Session ID returned by request	
	SessionId	

Return values

TRDP_NO_ERR	no error
TRDP_NOSESSION_ERR	no such session
TRDP_NOINIT_ERR	handle invalid

5.19.2.2 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	dandle the handle returned by tlc_openSession	
out	pListenHandle	Handle for this listener returned	
in	pUserRef	ef user supplied value returned with received message	
in	pfCbFunction	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	tbTopoCnt 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	2 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

5.19.2.3 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	ppHandle the handle returned by tlc_openSession	
in	pSessionId Session ID returned by request		
in	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

Return values

TRDP_MEM_ERR	out of memory
TRDP_NOSESSION_ERR	no such session
TRDP_NOINIT_ERR	handle invalid

5.19.2.4 tlm_delListener()

Remove Listener.

Parameters

	in	appHandle	the handle returned by tlc_openSession
ſ	out	listenHandle	Handle for this listener

Return values

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	parameter error
TRDP_NOINIT_ERR	handle invalid

5.19.2.5 tlm_getInterval()

Get the lowest time interval for MDs.

Return the maximum time interval suitable for 'select()' so that we can report time outs to the higher layer.

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

5.19.2.6 tlm_notify()

```
EXT_DECL TRDP_ERR_T tlm_notify (

TRDP_APP_SESSION_T appHandle,
const void * pUserRef,
TRDP_MD_CALLBACK_T pfCbFunction,
UINT32 comId,
UINT32 etbTopoCnt,
UINT32 opTrnTopoCnt,
TRDP_IP_ADDR_T srcIpAddr,
TRDP_IP_ADDR_T destIpAddr,
TRDP_FLAGS_T pktFlags,
const TRDP_SEND_PARAM_T * pSendParam,
const UINT8 * pData,
UINT32 dataSize,
const TRDP_URI_USER_T srcURI,
const TRDP_URI_USER_T destURI)
```

Initiate sending MD notification message.

Send a MD notification message

Parameters

i ai ai ii cu	013	
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	srcURI	only functional group of source URI
in	destURI	only functional group of destination URI

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	parameter error
TRDP_MEM_ERR	out of memory

Return values

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

5.19.2.7 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

5.19.2.8 tlm_readdListener()

```
EXT_DECL TRDP_ERR_T tlm_readdListener (

TRDP_APP_SESSION_T appHandle,

TRDP_LIS_T listenHandle,

UINT32 etbTopoCnt,

UINT32 opTrnTopoCnt,

TRDP_IP_ADDR_T srcIpAddr1,

TRDP_IP_ADDR_T srcIpAddr2,

TRDP_IP_ADDR_T mcDestIpAddr )
```

Resubscribe to MD messages.

Readd a listener after topoCount changes to get notified when messages are received

in	appHandle	the handle returned by tlc_openSession
out	listenHandle	Handle for this listener
in	etbTopoCnt	ETB topocount to use, 0 if consist local communication

Parameters

in	opTrnTopoCnt	operational topocount, != 0 for orientation/direction sensitive communication
in	srclpAddr1	Source IP address, lower address in case of address range, set to 0 if not used
in	srclpAddr2	upper address in case of address range, set to 0 if not used
in	mcDestlpAddr	multicast group to listen on

Return values

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

5.19.2.9 tlm_reply()

```
EXT_DECL TRDP_ERR_T tlm_reply (

TRDP_APP_SESSION_T appHandle,
const TRDP_UUID_T * pSessionId,
UINT32 comId,
UINT16 userStatus,
const TRDP_SEND_PARAM_T * pSendParam,
const UINT8 * pData,
UINT32 dataSize )
```

Send a MD reply message.

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

Parameters

in	appHandle	the handle returned by tlc_openSession
in	pSessionId	Session ID returned by indication
in	comld	comld of packet to be sent
in	userStatus	Info for requester about application errors
in	pSendParam	Pointer to send parameters, NULL to use default send parameters
in	pData	pointer to packet data / dataset
in	dataSize	size of packet data

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	parameter error
TRDP_MEM_ERR	Out of memory
TRDP_NO_SESSION_ERR	no such session
TRDP_NOINIT_ERR	handle invalid

5.19.2.10 tlm_replyQuery()

```
EXT_DECL TRDP_ERR_T tlm_replyQuery (

TRDP_APP_SESSION_T appHandle,

const TRDP_UUID_T * pSessionId,

UINT32 comId,

UINT16 userStatus,

UINT32 confirmTimeout,

const TRDP_SEND_PARAM_T * pSendParam,

const UINT3 * pData,

UINT32 dataSize )
```

Send a MD reply query message.

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

Parameters

in	appHandle	the handle returned by tlc_openSession
in	pSessionId	Session ID returned by indication
in	comld	comld of packet to be sent
in	userStatus	Info for requester about application errors
in	confirmTimeout	timeout for confirmation
in	pSendParam	Pointer to send parameters, NULL to use default send parameters
in	pData	pointer to packet data / dataset
in	dataSize	size of packet data

Return values

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

5.19.2.11 tlm_request()

```
EXT_DECL TRDP_ERR_T tlm_request (

TRDP_APP_SESSION_T appHandle,
const void * pUserRef,
TRDP_MD_CALLBACK_T pfCbFunction,
TRDP_UUID_T * pSessionId,
UINT32 comId,
UINT32 etbTopoCnt,
UINT32 opTrnTopoCnt,
TRDP_IP_ADDR_T srcIpAddr,
```

```
TRDP_IP_ADDR_T destIpAddr,
TRDP_FLAGS_T pktFlags,
UINT32 numReplies,
UINT32 replyTimeout,
const TRDP_SEND_PARAM_T * pSendParam,
const UINT8 * pData,
UINT32 dataSize,
const TRDP_URI_USER_T srcURI,
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	srcURI	only functional group of source URI
in	destURI	only functional group of destination URI

Return values

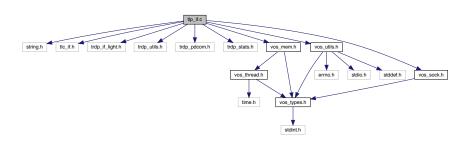
TRDP_NO_ERR	no error
TRDP_PARAM_ERR	parameter error
TRDP_MEM_ERR	out of memory
TRDP_NOINIT_ERR	handle invalid

5.20 tlp_if.c File Reference

Functions for Process Data Communication.

```
#include <string.h>
#include "tlc_if.h"
#include "trdp_utils.h"
#include "trdp_pdcom.h"
```

```
#include "trdp_stats.h"
#include "vos_sock.h"
#include "vos_mem.h"
#include "vos_utils.h"
Include dependency graph for tlp_if.c:
```



Functions

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

Get the lowest time interval for PDs.

• EXT_DECL TRDP_ERR_T tlp_processReceive (TRDP_APP_SESSION_T appHandle, TRDP_FDS_T *p↔ Rfds, INT32 *pCount)

Work loop of the TRDP handler.

- EXT_DECL TRDP_ERR_T tlp_processSend (TRDP_APP_SESSION_T appHandle)
 - Work loop of the TRDP handler.
- EXT_DECL TRDP_ERR_T tlp_setRedundant (TRDP_APP_SESSION_T appHandle, UINT32 redId, BOOL8 leader)

Do not send non-redundant PDs when we are follower.

EXT_DECL TRDP_ERR_T tlp_getRedundant (TRDP_APP_SESSION_T appHandle, UINT32 redId, BOOL8 *pLeader)

Get status of redundant Comlds.

• EXT_DECL TRDP_ERR_T tlp_publish (TRDP_APP_SESSION_T appHandle, TRDP_PUB_T *pPubHandle, const void *pUserRef, TRDP_PD_CALLBACK_T pfCbFunction, UINT32 serviceId, UINT32 comId, UINT32 etbTopoCnt, UINT32 opTrnTopoCnt, TRDP_IP_ADDR_T srclpAddr, TRDP_IP_ADDR_T destIpAddr, UINT32 interval, UINT32 redId, TRDP_FLAGS_T pktFlags, const TRDP_SEND_PARAM_T *pSendParam, const U← INT8 *pData, UINT32 dataSize)

Prepare for sending PD messages.

- EXT_DECL TRDP_ERR_T tlp_republish (TRDP_APP_SESSION_T appHandle, TRDP_PUB_T pubHandle, UINT32 etbTopoCnt, UINT32 opTrnTopoCnt, TRDP_IP_ADDR_T srclpAddr, TRDP_IP_ADDR_T destlpAddr)
 - Prepare for sending PD messages.
- EXT_DECL TRDP_ERR_T tlp_unpublish (TRDP_APP_SESSION_T appHandle, TRDP_PUB_T pubHandle) Stop sending PD messages.
- EXT_DECL TRDP_ERR_T tlp_put (TRDP_APP_SESSION_T appHandle, TRDP_PUB_T pubHandle, const UINT8 *pData, UINT32 dataSize)

Update the process data to send.

• EXT_DECL TRDP_ERR_T tlp_putImmediate (TRDP_APP_SESSION_T appHandle, TRDP_PUB_T pub

Handle, const UINT8 *pData, UINT32 dataSize, VOS TIMEVAL T *pTxTime)

Update and send process data.

EXT_DECL TRDP_ERR_T tlp_request (TRDP_APP_SESSION_T appHandle, TRDP_SUB_T subHandle, UINT32 serviceld, UINT32 comld, UINT32 etbTopoCnt, UINT32 opTrnTopoCnt, TRDP_IP_ADDR_T srcIp← Addr, TRDP_IP_ADDR_T destlpAddr, UINT32 redld, TRDP_FLAGS_T pktFlags, const TRDP_SEND_PARAM_T *pSendParam, const UINT8 *pData, UINT32 dataSize, UINT32 replyComld, TRDP_IP_ADDR_T replyIp← Addr)

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, TRDP_PD_INFO_T *pPdInfo, UINT8 *pData, UINT32 *pDataSize)

Get the last valid PD message.

5.20.1 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, 2013-2020. All rights reserved.

5.20.2 Function Documentation

5.20.2.1 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 <i>subHandle</i>		the handle returned by subscription	
in,out	pPdInfo	pointer to application's info buffer	
in, out pData pointer to application's data buff		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

5.20.2.2 tlp_getInterval()

Get the lowest time interval for PDs.

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

Parameters

in	appHandle The handle returned by tlc_openSession		
out	pInterval	nterval 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

5.20.2.3 tlp_getRedundant()

```
EXT_DECL TRDP_ERR_T tlp_getRedundant (

TRDP_APP_SESSION_T appHandle,
```

```
UINT32 redId,
BOOL8 * pLeader )
```

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

5.20.2.4 tlp_processReceive()

Work loop of the TRDP handler.

Check the sockets for incoming PD telegrams. Search the receive queue for pending PDs (time out) and report them, either by informing the higher layer via the callback mechanism or just by marking the subscriber as timed-out

Parameters

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

Return values

TRDP_NO_ERR	no error
TRDP_NOINIT_ERR	handle invalid

5.20.2.5 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

5.20.2.6 tlp_publish()

```
EXT_DECL TRDP_ERR_T tlp_publish (
             TRDP_APP_SESSION_T appHandle,
             {\tt 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

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	redld	0 - Non-redundant, > 0 valid redundancy group

Parameters

	in	pktFlags	OPTION: TRDP_FLAGS_DEFAULT, TRDP_FLAGS_NONE, TRDP_FLAGS_MARSHALL, TRDP_FLAGS_CALLBACK
Ì	in	pSendParam	optional pointer to send parameter, NULL - default parameters are used
Ī	in	pData	optional pointer to data packet / dataset, NULL if sending starts later with tlp_put()
Ī	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

5.20.2.7 tlp_put()

```
EXT_DECL TRDP_ERR_T tlp_put (
          TRDP_APP_SESSION_T appHandle,
          TRDP_PUB_T pubHandle,
          const UINT8 * pData,
          UINT32 dataSize )
```

Update the process data to send.

 $\label{thm:local_process} \mbox{Update previously published data. The new telegram will be sent earliest when tlc_process is called.}$

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

5.20.2.8 tlp_putImmediate()

```
{\tt EXT\_DECL\ TRDP\_ERR\_T\ tlp\_putImmediate\ (}
```

```
TRDP_APP_SESSION_T appHandle,
TRDP_PUB_T pubHandle,
const UINT8 * pData,
UINT32 dataSize,
VOS_TIMEVAL_T * pTxTime )
```

Update and send process data.

Update previously published data. The new telegram will be sent immediatly or at txTime, if txTime != 0 and TSN == 1 Should be used if application (or higher layer, e.g. ara::com and acyclic events) needs full control over process data schedule.

Note: For TSN this function is not protected by any mutexes and should not be called while adding or removing any publishers, subscribers or even sessions! Also: Marshalling is not supported!

Parameters

in	appHandle	the handle returned by tlc_openSession
in	pubHandle	the handle returned by publish
in,out	pData	pointer to application's data buffer
in,out	dataSize	size of data
in	pTxTime	when to send (absolute time), optional for TSN only

Return values

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	parameter error on uninitialized parameter or changed dataSize compared to published one
TRDP_NOPUB_ERR	not published
TRDP_NOINIT_ERR	handle invalid

5.20.2.9 tlp_republish()

```
EXT_DECL TRDP_ERR_T tlp_republish (

TRDP_APP_SESSION_T appHandle,

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

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 Generate	d by Doxygen

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

5.20.2.10 tlp_request()

```
EXT_DECL TRDP_ERR_T tlp_request (
             TRDP_APP_SESSION_T appHandle,
            TRDP_SUB_T subHandle,
            UINT32 serviceId,
            UINT32 comId,
            UINT32 etbTopoCnt,
            UINT32 opTrnTopoCnt,
             TRDP_IP_ADDR_T srcIpAddr,
             TRDP_IP_ADDR_T destIpAddr,
             UINT32 redId,
             TRDP_FLAGS_T pktFlags,
             const TRDP_SEND_PARAM_T * pSendParam,
             const UINT8 * pData,
             UINT32 dataSize,
             UINT32 replyComId,
             TRDP_IP_ADDR_T replyIpAddr )
```

Initiate sending PD messages (PULL).

Send a PD request message

Parameters

handle from related subscribe	
optional serviceld this telegram belongs to (default = 0)	
comld of packet to be sent	
ETB topocount to use, 0 if consist local communication	
nmunication	
where to send the packet to	
DP_FLAGS_MARSHALL,	
re used	
R	

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

5.20.2.11 tlp_resubscribe()

Reprepare for receiving PD messages.

Resubscribe to a specific PD ComID and source IP

Parameters

in	appHandle	the handle returned by tlc_openSession	
in	subHandle	handle for this subscription	
in	etbTopoCnt	ETB topocount to use, 0 if consist local communication	
in	opTrnTopoCnt	operational topocount, != 0 for orientation/direction sensitive communication	
in	srclpAddr1 Source IP address, lower address in case of address range, set to 0 if not used		
in	srclpAddr2 upper address in case of address range, set to 0 if not used		
in	destlpAddr	IP address to join	

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

5.20.2.12 tlp_setRedundant()

```
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.

Parameters

in	appHandle	the handle returned by tlc_openSession	
in	redld	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

5.20.2.13 tlp_subscribe()

```
EXT_DECL TRDP_ERR_T tlp_subscribe (
            TRDP_APP_SESSION_T appHandle,
            TRDP_SUB_T * pSubHandle,
            const void * pUserRef,
             TRDP_PD_CALLBACK_T pfCbFunction,
             UINT32 serviceId,
             UINT32 comId,
             UINT32 etbTopoCnt,
             UINT32 opTrnTopoCnt,
             TRDP_IP_ADDR_T srcIpAddr1,
             TRDP_IP_ADDR_T srcIpAddr2,
             TRDP_IP_ADDR_T destIpAddr,
             TRDP_FLAGS_T pktFlags,
             const TRDP_COM_PARAM_T * pRecParams,
             UINT32 timeout,
             TRDP_TO_BEHAVIOR_T toBehavior )
```

Prepare for receiving PD messages.

Subscribe to a specific PD ComID and source IP.

Parameters

in	appHandle	the handle returned by tlc_openSession	
out	pSubHandle	return a handle for this subscription	
in	pUserRef	user supplied value returned within the info structure	
in	pfCbFunction	Pointer to subscriber specific callback function, NULL to use default function	
in	serviceld	optional serviceld this telegram belongs to (default = 0)	
in	comld	comld of packet to receive	
in	etbTopoCnt	ETB topocount to use, 0 if consist local communication	
in	opTrnTopoCnt	operational topocount, != 0 for orientation/direction sensitive communication	
in	srclpAddr1	srclpAddr1 Source IP address, lower address in case of address range, set to 0 if not used	

Parameters

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	t 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

5.20.2.14 tlp_unpublish()

```
EXT_DECL TRDP_ERR_T tlp_unpublish (
          TRDP_APP_SESSION_T appHandle,
          TRDP_PUB_T pubHandle )
```

Stop sending PD messages.

Parameters

in	appHandle	the handle returned by tlc_openSession
in	pubHandle	the handle returned by prepare

Return values

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	parameter error
TRDP_NOPUB_ERR	not published
TRDP_NOINIT_ERR	handle invalid

5.20.2.15 tlp_unsubscribe()

```
EXT_DECL TRDP_ERR_T tlp_unsubscribe (
          TRDP_APP_SESSION_T appHandle,
          TRDP_SUB_T subHandle )
```

Stop receiving PD messages.

Unsubscribe to a specific PD ComID

Parameters

in	appHandle	the handle returned by tlc_openSession
in	subHandle	the handle for this subscription

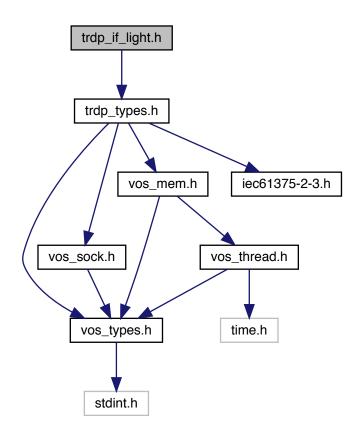
Return values

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

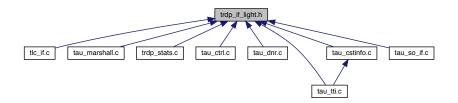
5.21 trdp_if_light.h File Reference

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_PROCESS_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_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.

• EXT_DECL TRDP_ERR_T tlc_updateSession (TRDP_APP_SESSION_T appHandle)

Update a session.

• EXT_DECL TRDP_ERR_T tlc_presetIndexSession (TRDP_APP_SESSION_T appHandle, TRDP_IDX_TABLE_T *pIndexTableSizes)

Preset the index table sizes of a session.

EXT_DECL TRDP_ERR_T tlc_closeSession (TRDP_APP_SESSION_T appHandle)

Close a session.

• EXT DECL TRDP ERR T tlc terminate (void)

Un-Initialize.

• EXT_DECL TRDP_ERR_T tlc_setETBTopoCount (TRDP_APP_SESSION_T appHandle, UINT32 etbTopo ← Cnt)

Set new topocount for trainwide communication.

• EXT_DECL UINT32 tlc_getETBTopoCount (TRDP_APP_SESSION_T appHandle)

Set new topocount for trainwide communication.

EXT_DECL TRDP_ERR_T tlc_setOpTrainTopoCount (TRDP_APP_SESSION_T appHandle, UINT32 op
 — TrnTopoCnt)

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

• EXT DECL UINT32 tlc getOpTrainTopoCount (TRDP APP SESSION TappHandle)

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_DECLTRDP_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 comId, UINT32 etbTopoCnt, UINT32 opTrnTopoCnt, TRDP_IP_ADDR_T srcIp← Addr, 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 replyIp← Addr)

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 destIpAddr, 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, TRDP_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, TRDP_MD_CALLBACK_T pfCbFunction, UINT32 comId, UINT32 etbTopoCnt, UINT32 opTrnTopo← Cnt, 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_USE← R_T srcURI, 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, 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 srcURI, 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 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.

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

Resubscribe to MD messages.

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

Remove Listener.

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

Send a MD reply message.

 EXT_DECL TRDP_ERR_T tlm_replyQuery (TRDP_APP_SESSION_T appHandle, const TRDP_UUID_T *pSessionId, UINT32 comId, UINT16 userStatus, UINT32 confirmTimeout, const TRDP_SEND_PARAM_T *pSendParam, const 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.

5.21.1 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, 2013-2020. All rights reserved.

5.21.2 Function Documentation

5.21.2.1 tlc_closeSession()

Close a session.

Clean up and release all resources of that session

Parameters

	appHandle The handle returned by tlc_openSession
--	--

Return values

TRDP_NO_ERR	no error
TRDP_NOINIT_ERR	handle invalid
TRDP_PARAM_ERR	handle NULL

5.21.2.2 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	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

5.21.2.3 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
```

5.21.2.4 tlc_getInterval()

```
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.

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

Parameters

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

Return values

TRDP_NO_ERR	no error
TRDP_NOINIT_ERR	handle invalid

5.21.2.5 tlc_getJoinStatistics()

Return join statistics.

Memory for statistics information must be provided by the user.

Parameters

in	appHandle	the handle returned by tlc_openSession
,	pNumJoin	Pointer to the number of joined IP Adresses
Generated by Dox 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

5.21.2.6 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
--------------	---------------------------------

5.21.2.7 tlc_getOwnlpAddress()

Get the interface address.

Parameters

С	ut	appHandle	A handle for further calls to the trdp stack
---	----	-----------	--

Return values

```
real⇔
IP
```

5.21.2.8 tlc_getPubStatistics()

```
EXT_DECL TRDP_ERR_T tlc_getPubStatistics (

TRDP_APP_SESSION_T appHandle,
```

```
UINT16 * pNumPub,
TRDP_PUB_STATISTICS_T * pStatistics )
```

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

5.21.2.9 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

5.21.2.10 tlc_getStatistics()

```
{\tt EXT\_DECL\ TRDP\_ERR\_T\ tlc\_getStatistics\ (}
```

```
TRDP_APP_SESSION_T appHandle,
TRDP_STATISTICS_T * pStatistics )
```

Return statistics.

Memory for statistics information must be provided by the user.

Parameters

in	appHandle	the handle returned by tlc_openSession
out <i>pStatistics</i>		Pointer to statistics for this application session

Return values

TRDP_NO_ERR	no error
TRDP_NOINIT_ERR	handle invalid
TRDP_PARAM_ERR	parameter error

5.21.2.11 tlc_getSubsStatistics()

Return PD subscription statistics.

Memory for statistics information must be provided by the user.

Parameters

in	appHandle	the handle returned by tlc_openSession
in,out	pNumSubs	In: The number of subscriptions requested Out: Number of subscriptions returned
in,out	pStatistics	Pointer to an array with the subscription statistics information

Return values

TRDP_NO_ERR	no error
TRDP_NOINIT_ERR	handle invalid
TRDP_PARAM_ERR	parameter error
TRDP_MEM_ERR	there are more subscriptions than requested

5.21.2.12 tlc_getVersion()

Return version.

Return pointer to version structure

Return values

```
TRDP_VERSION↔
_T
```

5.21.2.13 tlc_getVersionString()

Return a human readable version representation.

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

Return values

const	string

5.21.2.14 tlc_init()

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

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

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

Parameters

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

Return values

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

5.21.2.15 tlc_openSession()

```
EXT_DECL TRDP_ERR_T tlc_openSession (

TRDP_APP_SESSION_T * pAppHandle,

TRDP_IP_ADDR_T ownIpAddr,

TRDP_IP_ADDR_T leaderIpAddr,

const TRDP_MARSHALL_CONFIG_T * pMarshall,

const TRDP_PD_CONFIG_T * pPdDefault,

const TRDP_MD_CONFIG_T * pMdDefault,

const TRDP_PROCESS_CONFIG_T * pProcessConfig )
```

Open a session with the TRDP stack.

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

Parameters

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

Return values

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

5.21.2.16 tlc_presetIndexSession()

Preset the index table sizes of a session.

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

Parameters

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

Return values

TRDP_NO_ERR	no error
TRDP_INIT_ERR	not yet inited
TRDP_PARAM_ERR	parameter error

5.21.2.17 tlc_process()

Work loop of the TRDP handler.

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

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

Also see User Manual.

Parameters

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

Return values

TRDP_NO_ERR	no error
TRDP_NOINIT_ERR	handle invalid

5.21.2.18 tlc_reinitSession()

Re-Initialize.

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

Parameters

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

Return values

TRDP_NO_ERR	no error
TRDP_NOINIT_ERR	handle invalid
TRDP_PARAM_ERR	handle NULL

5.21.2.19 tlc_resetStatistics()

Reset statistics.

Parameters

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

Return values

TRDP_NO_ERR	no error
TRDP_NOINIT_ERR	handle invalid
TRDP_PARAM_ERR	parameter error

5.21.2.20 tlc_setETBTopoCount()

Set new topocount for trainwide communication.

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

Parameters

in	appHandle	the handle returned by tlc_openSession
in	etbTopoCnt	New etbTopoCnt value

Return values

TRDP_NO_ERR	no error
TRDP_NOINIT_ERR	handle invalid

5.21.2.21 tlc_setOpTrainTopoCount()

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

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

Parameters

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

5.21.2.22 tlc_terminate()

Un-Initialize.

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

Return values

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

5.21.2.23 tlc_updateSession()

```
{\tt EXT\_DECL\ TRDP\_ERR\_T\ tlc\_updateSession\ (}
```

```
TRDP_APP_SESSION_T appHandle )
```

Update a session.

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

Parameters

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

Return values

TRDP_NO_ERR	no error
TRDP_INIT_ERR	not yet inited
TRDP_PARAM_ERR	parameter error

5.21.2.24 tlm_abortSession()

Cancel an open session.

Abort an open session; any pending messages will be dropped

Parameters

in	appHandle	the handle returned by tlc_openSession
in	p⇔ SessionId	Session ID returned by request

Return values

TRDP_NO_ERR	no error
TRDP_NOSESSION_ERR	no such session
TRDP_NOINIT_ERR	handle invalid

5.21.2.25 tlm_addListener()

```
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

5.21.2.26 tlm_confirm()

Initiate sending MD confirm message.

Send a MD confirmation message User reference, source and destination IP addresses as well as topo counts and packet flags are taken from the session

Parameters

	in	appHandle	the handle returned by tlc_openSession	
in pSessionId Session ID returned by request		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		Info for requester about application errors		
		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

5.21.2.27 tlm_delListener()

Remove Listener.

Parameters

	in	appHandle	the handle returned by tlc_openSession
ſ	out <i>listenHandle</i>		Handle for this listener

Return values

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	parameter error
TRDP_NOINIT_ERR	handle invalid

5.21.2.28 tlm_getInterval()

```
EXT_DECL TRDP_ERR_T tlm_getInterval (
          TRDP_APP_SESSION_T appHandle,
          TRDP_TIME_T * pInterval,
          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	pInterval	val 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

5.21.2.29 tlm_notify()

```
EXT_DECL TRDP_ERR_T tlm_notify (

TRDP_APP_SESSION_T appHandle,
const void * pUserRef,
TRDP_MD_CALLBACK_T pfCbFunction,
UINT32 comId,
UINT32 etbTopoCnt,
UINT32 opTrnTopoCnt,
TRDP_IP_ADDR_T srcIpAddr,
TRDP_IP_ADDR_T destIpAddr,
TRDP_FLAGS_T pktFlags,
const TRDP_SEND_PARAM_T * pSendParam,
const UINT3 * pData,
UINT32 dataSize,
const TRDP_URI_USER_T srcURI,
const TRDP_URI_USER_T destURI)
```

Initiate sending MD notification message.

Send a MD notification message

Parameters

i didiliot	arameters				
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	srcURI	only functional group of source URI			
in	destURI	only functional group of destination URI			

Return values

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

5.21.2.30 tlm_process()

Message Data Work loop of the TRDP handler.

Search the queue for pending MDs to be sent Search the receive queue for pending MDs (replies, time outs) and incoming requests

Parameters

in	appHandle	The handle returned by tlc_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

5.21.2.31 tlm_readdListener()

```
EXT_DECL TRDP_ERR_T tlm_readdListener (

TRDP_APP_SESSION_T appHandle,

TRDP_LIS_T listenHandle,

UINT32 etbTopoCnt,

UINT32 opTrnTopoCnt,

TRDP_IP_ADDR_T srcIpAddr1,

TRDP_IP_ADDR_T srcIpAddr2,

TRDP_IP_ADDR_T mcDestIpAddr )
```

Resubscribe to MD messages.

Readd a listener after topoCount changes to get notified when messages are received

Parameters

in appHandle the handle returned by tlc_openSession		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

5.21.2.32 tlm_reply()

```
EXT_DECL TRDP_ERR_T tlm_reply (

TRDP_APP_SESSION_T appHandle,

const TRDP_UUID_T * pSessionId,

UINT32 comId,

UINT16 userStatus,

const TRDP_SEND_PARAM_T * pSendParam,

const UINT8 * pData,

UINT32 dataSize )
```

Send a MD reply message.

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

Parameters

in	appHandle	the handle returned by tlc_openSession	
in	pSessionId	Session ID returned by indication	
in	comld	comld of packet to be sent	
in	userStatus	Info for requester about application errors	
in pSendParam Pointer to send parameters, NULL to use		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

Return values

TRDP NOINIT ERR	handle invalid

5.21.2.33 tlm_replyQuery()

```
EXT_DECL TRDP_ERR_T tlm_replyQuery (

TRDP_APP_SESSION_T appHandle,

const TRDP_UUID_T * pSessionId,

UINT32 comId,

UINT16 userStatus,

UINT32 confirmTimeout,

const TRDP_SEND_PARAM_T * pSendParam,

const 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

5.21.2.34 tlm_request()

```
EXT_DECL TRDP_ERR_T tlm_request (

TRDP_APP_SESSION_T appHandle,

const void * pUserRef,

TRDP_MD_CALLBACK_T pfCbFunction,
```

```
TRDP_UUID_T * pSessionId,

UINT32 comId,

UINT32 etbTopoCnt,

UINT32 opTrnTopoCnt,

TRDP_IP_ADDR_T srcIpAddr,

TRDP_IP_ADDR_T destIpAddr,

TRDP_FLAGS_T pktFlags,

UINT32 numReplies,

UINT32 replyTimeout,

const TRDP_SEND_PARAM_T * pSendParam,

const UINT8 * pData,

UINT32 dataSize,

const TRDP_URI_USER_T srcURI,

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	srcURI	only functional group of source URI	
in	destURI	only functional group of destination URI	

Return values

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

5.21.2.35 tlp_get()

```
EXT_DECL TRDP_ERR_T tlp_get (
          TRDP_APP_SESSION_T appHandle,
```

```
TRDP_SUB_T subHandle,
TRDP_PD_INFO_T * pPdInfo,
UINT8 * pData,
UINT32 * pDataSize )
```

Get the last valid PD message.

This allows polling of PDs instead of event driven handling by callbacks

Parameters

in	appHandle	the handle returned by tlc_openSession
in	subHandle	the handle returned by subscription
in,out	pPdInfo	pointer to application's info buffer
in,out	pData	pointer to application's data buffer
in,out	pDataSize	in: size of buffer, out: size of data

Return values

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	parameter error
TRDP_SUB_ERR	not subscribed
TRDP_TIMEOUT_ERR	packet timed out
TRDP_NOINIT_ERR	handle invalid
TRDP_COMID_ERR	ComID not found when marshalling

5.21.2.36 tlp_getInterval()

```
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.

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

Parameters

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

Return values

TRDP_NO_ERR	no error
TRDP_NOINIT_ERR	handle invalid

5.21.2.37 tlp_getRedundant()

Get status of redundant Comlds.

Only the status of the first found redundancy group entry will be returned!

Parameters

in	appHandle	the handle returned by tlc_openSession
in	redId	will be returned for all ComID's with the given redId
in,out	pLeader	TRUE if we're sending this redundancy group (leader)

Return values

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	redld invalid or not existing
TRDP_NOINIT_ERR	handle invalid

5.21.2.38 tlp_processReceive()

Work loop of the TRDP handler.

Check the sockets for incoming PD telegrams. Search the receive queue for pending PDs (time out) and report them, either by informing the higher layer via the callback mechanism or just by marking the subscriber as timed-out

Parameters

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

Return values

TRDP_NO_ERR	no error
TRDP_NOINIT_ERR	handle invalid

5.21.2.39 tlp_processSend()

Work loop of the TRDP handler.

Search the queue for pending PDs to be sent

Parameters

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

Return values

TRDP_NO_ERR	no error
TRDP_NOINIT_ERR	handle invalid

5.21.2.40 tlp_publish()

```
EXT_DECL TRDP_ERR_T tlp_publish (
            TRDP_APP_SESSION_T appHandle,
            TRDP_PUB_T * pPubHandle,
             const void * pUserRef,
             TRDP_PD_CALLBACK_T pfCbFunction,
             UINT32 serviceId,
             UINT32 comId,
             UINT32 etbTopoCnt,
             UINT32 opTrnTopoCnt,
             TRDP_IP_ADDR_T srcIpAddr,
             TRDP_IP_ADDR_T destIpAddr,
             UINT32 interval,
             UINT32 redId,
             TRDP_FLAGS_T pktFlags,
             const TRDP_SEND_PARAM_T * pSendParam,
             const UINT8 * pData,
             UINT32 dataSize )
```

Prepare for sending PD messages.

Queue a PD message, it will be send when tlc_publish has been called

Parameters

in	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

Parameters

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

5.21.2.41 tlp_put()

```
EXT_DECL TRDP_ERR_T tlp_put (
          TRDP_APP_SESSION_T appHandle,
          TRDP_PUB_T pubHandle,
          const UINT8 * pData,
          UINT32 dataSize )
```

Update the process data to send.

Update previously published data. The new telegram will be sent earliest when tlc_process is called.

Parameters

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

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

Return values

TRDP_COMID_ERR	ComID not found when marshalling
----------------	----------------------------------

5.21.2.42 tlp_putlmmediate()

```
EXT_DECL TRDP_ERR_T tlp_putImmediate (

TRDP_APP_SESSION_T appHandle,

TRDP_PUB_T pubHandle,

const UINT8 * pData,

UINT32 dataSize,

VOS_TIMEVAL_T * pTxTime )
```

Update and send process data.

Update previously published data. The new telegram will be sent immediatly or at txTime, if txTime != 0 and TSN == 1 Should be used if application (or higher layer, e.g. ara::com and acyclic events) needs full control over process data schedule.

Note: For TSN this function is not protected by any mutexes and should not be called while adding or removing any publishers, subscribers or even sessions! Also: Marshalling is not supported!

Parameters

in	pTxTime	when to send (absolute time), optional for TSN only
in,out	dataSize	size of data
in,out	pData	pointer to application's data buffer
in <i>pubHandle</i>		the handle returned by publish
in	appHandle	the handle returned by tlc_openSession

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

5.21.2.43 tlp_republish()

```
EXT_DECL TRDP_ERR_T tlp_republish (

TRDP_APP_SESSION_T appHandle,

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

5.21.2.44 tlp_request()

```
EXT_DECL TRDP_ERR_T tlp_request (
             TRDP_APP_SESSION_T appHandle,
             TRDP_SUB_T subHandle,
             UINT32 serviceId,
            UINT32 comId,
             UINT32 etbTopoCnt,
             UINT32 opTrnTopoCnt,
             TRDP_IP_ADDR_T srcIpAddr,
             TRDP_IP_ADDR_T destIpAddr,
             UINT32 redId,
             TRDP_FLAGS_T pktFlags,
             const TRDP_SEND_PARAM_T * pSendParam,
             const UINT8 * pData,
             UINT32 dataSize,
             UINT32 replyComId,
             TRDP_IP_ADDR_T replyIpAddr )
```

Initiate sending PD messages (PULL).

Send a PD request message

Parameters

in	appHandle	the handle returned by tlc_openSession
in	subHandle	handle from related subscribe
in	serviceld	optional serviceld this telegram belongs to (default = 0)
in	comld	comld of packet to be sent
in	etbTopoCnt	ETB topocount to use, 0 if consist local communication
in	opTrnTopoCnt	operational topocount, != 0 for orientation/direction sensitive communication

Parameters

in	srclpAddr	own IP address, 0 - srcIP will be set by the stack
in	destlpAddr	where to send the packet to
in	redld	0 - Non-redundant, > 0 valid redundancy group
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

5.21.2.45 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

Return values

TRDP_NO_ERR

Return values

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

5.21.2.46 tlp_setRedundant()

```
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.

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

5.21.2.47 tlp_subscribe()

```
EXT_DECL TRDP_ERR_T tlp_subscribe (
             TRDP_APP_SESSION_T appHandle,
             TRDP_SUB_T * pSubHandle,
             const void * pUserRef,
             TRDP_PD_CALLBACK_T pfCbFunction,
             UINT32 serviceId,
             UINT32 comId,
             UINT32 etbTopoCnt,
             UINT32 opTrnTopoCnt,
             TRDP_IP_ADDR_T srcIpAddr1,
             TRDP_IP_ADDR_T srcIpAddr2,
             TRDP_IP_ADDR_T destIpAddr,
             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 serviceId optional serviceId this telegram belongs to (default = 0) in comId comId of packet to receive in etbTopoCnt ETB topocount to use, 0 if consist local communication
in pUserRef user supplied value returned within the info structure in pfCbFunction Pointer to subscriber specific callback function, NULL to use default function in serviceId optional serviceId this telegram belongs to (default = 0) in comId comId of packet to receive
in pfCbFunction Pointer to subscriber specific callback function, NULL to use default function in serviceId optional serviceId this telegram belongs to (default = 0) in comId comId of packet to receive
in serviceId optional serviceId this telegram belongs to (default = 0) in comId comId comId of packet to receive
in 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 (>= 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

5.21.2.48 tlp_unpublish()

```
EXT_DECL TRDP_ERR_T tlp_unpublish (
                TRDP_APP_SESSION_T appHandle,
                TRDP_PUB_T pubHandle )
```

Stop sending PD messages.

Parameters

in	appHandle	the handle returned by tlc_openSession
in	pubHandle	the handle returned by prepare

Return values

TRDP_NO_ERR	no error
TRDP_PARAM_ERR	parameter error
TRDP_NOPUB_ERR	not published
TRDP_NOINIT_ERR	handle invalid

5.21.2.49 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

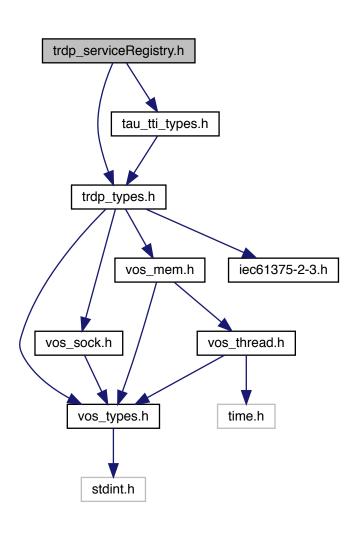
TRDP_NO_ERR	no error
TRDP_PARAM_ERR	parameter error
TRDP_NOSUB_ERR	not subscribed
TRDP_NOINIT_ERR	handle invalid

5.22 trdp_serviceRegistry.h File Reference

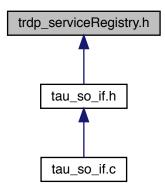
Additional definitions for IEC 61375-2-3 (Service Discovery) The definitions herein are preliminary and will change with the next major release of the IEC 61375-2-3 standard.

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

Include dependency graph for trdp_serviceRegistry.h:



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



Data Structures

· struct service_info

Preliminary definition of a service info entry.

· struct srv_info_req

Preliminary definition of a service info request.

struct GNU_PACKED

Types for ETB control.

Macros

- #define SRM_SRVINFO_NOTIFY_COMID 200u
 Additional defines to be reserved for SR Manager
- #define SRM_SRVINFO_NOTIFY_URI "grpSRM.anyVeh.aCst.aClTrn.lTrn" multicast group
- #define SRM_SRVINFO_NOTIFY_DS "CST_SRV_INFO" SRM_CST_SRV_INFO_T
- #define SRM_SRV_REQ_NOTIFY_COMID 201u SRVINFOREQ request data:
- #define SRM_SRV_REQ_NOTIFY_URI "grpSRM.anyVeh.aCst.aClTrn.lTrn" multicast group
- #define SRM_SRV_REQ_NOTIFY_DS "SRV_INFO_REQ" SRM_SRV_INFO_REQ_T
- #define SRM_SERVICE_READ_REQ_COMID 112u
 Additional COMIDs to be reserved for SR Manager

- #define SRM_SERVICE_READ_REQ_TO 3000000u
 [us] 3s timeout
- #define SRM_SERVICE_READ_REP_COMID 113u
 MD reply
- #define SRM_SERVICE_READ_REP_DS "SRM_SERVICE_ENTRIES_T"
 SRM_SERVICE_ENTRIES_T
- #define SRM_SERVICE_READ_REP_DSID SRM_SERVICE_DSID SRM_SERVICE_ENTRIES_T
- #define SRM_SERVICE_ADD_REQ_COMID 114u
 SRM manager telegram MD: Add service instance(s) to the Service Registry
- #define SRM_SERVICE_ADD_REQ_TO 3000000u
 [us] 3s timeout
- #define SRM_SERVICE_ADD_REQ_DS "SRM_SERVICE_ENTRIES_T"
 SRM_SERVICE_ENTRIES_T
- #define SRM_SERVICE_ADD_REQ_DSID SRM_SERVICE_DSID SRM_SERVICE_ENTRIES_T
- #define SRM_SERVICE_ADD_REP_COMID 115u
 Reply returns instanceId
- #define SRM_SERVICE_ADD_REP_DSID SRM_SERVICE_DSID SRM_SERVICE_ENTRIES_T
- #define SRM_SERVICE_UPD_NOTIFY_COMID 116u
 SRM manager telegram MD: Update service instance(s) to the Service Registry
- #define SRM_SERVICE_UPD_NOTIFY_TTL 3000000u
 [us] default time-to-live
- #define SRM_SERVICE_UPD_NOTIFY_DS "SRM_SERVICE_ENTRIES_T" SRM_SERVICE_ENTRIES_T
- #define SRM_SERVICE_UPD_NOTIFY_DSID SRM_SERVICE_DSID SRM_SERVICE_ENTRIES_T
- #define SRM_SERVICE_DEL_REQ_COMID 117u
 SRM manager telegram MD: Remove Service instance(s) from the Service Registry
- #define SRM_SERVICE_DEL_REQ_TO 3000000u
 [us] 3s timeout
- #define SRM_SERVICE_DEL_REQ_DS "SRM_SERVICE_ENTRIES_T"
 SRM_SERVICE_ENTRIES_T
- #define SRM_SERVICE_DEL_REQ_DSID SRM_SERVICE_DSID SRM_SERVICE_ENTRIES_T

- #define SRM_SERVICE_DEL_REP_COMID 118u
 MD reply OK or not
- #define SOA_SERVICEID(instld, typeId) ((instld) << 24 | (typeId))
 serviceId from instanceId and typeId
- #define SOA_TYPE(serviceId) ((serviceId) & 0xFFFFFF)

return 24 Bit service type part of serviceld

#define SOA_INST(serviceId) (((serviceId) >> 24) & 0xFF)

return 8 Bit instance ID part of serviceId

#define SOA_SAME_SERVICEID_OR0(a, b) (((a) == 0u) || ((a) == (b)))

return TRUE if serviceId(a) is 0 or equals the second serviceId (b)

• #define SOA_SAME_SERVICEID(a, b) ((a) == (b))

return TRUE if servicelds (incl.

#define SOA_SAME_SERVICE_TYPE(a, b) (SOA_TYPE(a) == SOA_TYPE(b))

return TRUE if service types match

Typedefs

typedef struct service_info SRM_SERVICE_INFO_T

Preliminary definition of a service info entry.

typedef struct srv_info_req SRM_SRV_INFO_REQ_T

Preliminary definition of a service info request.

5.22.1 Detailed Description

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

Note

Project: CTA2 WP3 / TCNOpen TRDP

Author

Bernd Loehr, NewTec GmbH, 2019-04-08

Remarks

Copyright 2019 Bombardier Transportation & NewTec GmbH

5.22.2 Macro Definition Documentation

5.22.2.1 SOA_SAME_SERVICEID

return TRUE if serviceIds (incl.

instance) match

5.22.2.2 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

5.22.2.3 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:

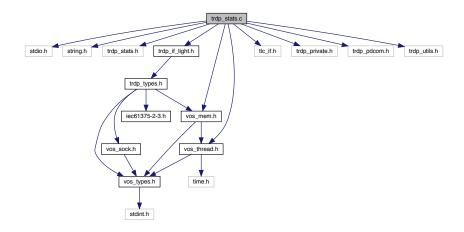
5.23 trdp_stats.c File Reference

Statistics functions for TRDP communication.

```
#include <stdio.h>
#include <string.h>
#include "trdp_stats.h"
#include "trdp_if_light.h"
#include "tlc_if.h"
#include "trdp_private.h"
#include "trdp_pdcom.h"
#include "trdp_utils.h"
#include "vos_mem.h"
```

#include "vos_thread.h"

Include dependency graph for trdp_stats.c:



Functions

• void trdp_UpdateStats (TRDP_APP_SESSION_T appHandle)

Update the statistics.

• void trdp_initStats (TRDP_APP_SESSION_T appHandle)

Init statistics

• EXT_DECL TRDP_ERR_T tlc_resetStatistics (TRDP_APP_SESSION_T appHandle)

Reset statistics.

• EXT_DECL TRDP_ERR_T tlc_getStatistics (TRDP_APP_SESSION_T appHandle, TRDP_STATISTICS_T *pStatistics)

Return statistics.

• EXT_DECL TRDP_ERR_T tlc_getSubsStatistics (TRDP_APP_SESSION_T appHandle, UINT16 *pNum← Subs, TRDP_SUBS_STATISTICS_T *pStatistics)

Return PD subscription statistics.

• EXT_DECL TRDP_ERR_T tlc_getPubStatistics (TRDP_APP_SESSION_T appHandle, UINT16 *pNumPub, TRDP PUB STATISTICS T *pStatistics)

Return PD publish statistics.

• EXT_DECL TRDP_ERR_T tlc_getRedStatistics (TRDP_APP_SESSION_T appHandle, UINT16 *pNumRed, TRDP_RED_STATISTICS_T *pStatistics)

Return redundancy group statistics.

• EXT_DECL TRDP_ERR_T tlc_getJoinStatistics (TRDP_APP_SESSION_T appHandle, UINT16 *pNumJoin, UINT32 *plpAddr)

Return join statistics.

void trdp_pdPrepareStats (TRDP_APP_SESSION_T appHandle, PD_ELE_T *pPacket)

Fill the statistics packet.

5.23.1 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.

5.23.2 Function Documentation

5.23.2.1 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
ľ	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

5.23.2.2 tlc_getPubStatistics()

```
EXT_DECL TRDP_ERR_T tlc_getPubStatistics (

TRDP_APP_SESSION_T appHandle,
```

```
\label{eq:continuity} \mbox{UINT16 * $pNumPub$,} \\ \mbox{TRDP_PUB\_STATISTICS\_T * $pStatistics })
```

Return PD publish statistics.

Memory for statistics information must be provided by the user.

Parameters

in	appHandle	the handle returned by tlc_openSession
in,out	pNumPub	Pointer to the number of publishers
out	pStatistics	Pointer to a list with the publish statistics information

Return values

TRDP_NO_ERR	no error
TRDP_NOINIT_ERR	handle invalid
TRDP_PARAM_ERR	parameter error
TRDP_MEM_ERR	there are more subscriptions than requested

5.23.2.3 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

5.23.2.4 tlc_getStatistics()

```
{\tt EXT\_DECL\ TRDP\_ERR\_T\ tlc\_getStatistics\ (}
```

```
TRDP_APP_SESSION_T appHandle,
TRDP_STATISTICS_T * pStatistics )
```

Return statistics.

Memory for statistics information must be provided by the user.

Parameters

in	appHandle	the handle returned by tlc_openSession
out	pStatistics	Pointer to statistics for this application session

Return values

TRDP_NO_ERR	no error
TRDP_NOINIT_ERR	handle invalid
TRDP_PARAM_ERR	parameter error

5.23.2.5 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

5.23.2.6 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

5.23.2.7 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:



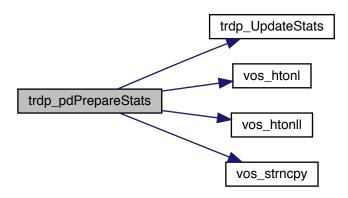
5.23.2.8 trdp_pdPrepareStats()

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:



5.23.2.9 trdp_UpdateStats()

Update the statistics.

Parameters

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

5.24 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

5.24.1 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

5.24.2 Macro Definition Documentation

5.24.2.1 TRDP_MIN_PD2_HEADER_SIZE

```
#define TRDP_MIN_PD2_HEADER_SIZE sizeof(PD2_HEADER_T)
```

PD packet properties

TSN header size with FCS

5.24.2.2 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

5.24.2.3 TRDP_PD_DEFAULT_QOS

```
#define TRDP_PD_DEFAULT_QOS 2u
```

Default PD communication parameters

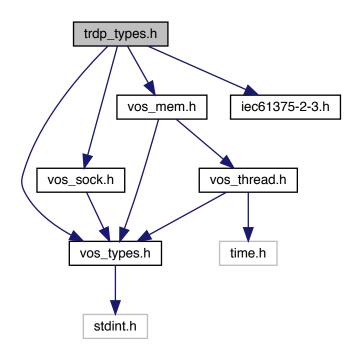
matching new proposed priority classes

5.25 trdp_types.h File Reference

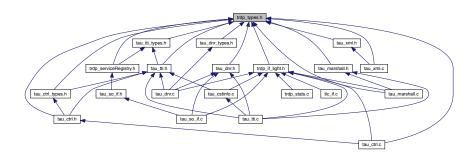
Typedefs for TRDP communication.

```
#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

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

struct TRDP_COM_PARAM_T

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

• struct TRDP_DATASET_ELEMENT_T

Dataset element definition

struct TRDP DATASET

Dataset definition

• struct TRDP COMID DSID MAP T

Comld - data set mapping element definition

struct GNU_PACKED

Types for ETB control.

• struct GNU_PACKED

Types for ETB control.

struct GNU_PACKED

Types for ETB control.

struct GNU PACKED

Types for ETB control.

struct GNU_PACKED

Types for ETB control.

struct GNU_PACKED

Types for ETB control.

• struct GNU_PACKED

Types for ETB control.

struct GNU_PACKED

Types for ETB control.

struct GNU_PACKED

Types for ETB control.

struct TRDP_MARSHALL_CONFIG_T

Marshaling/unmarshalling configuration

• struct TRDP_PD_CONFIG_T

Default PD configuration

struct TRDP_MD_CONFIG_T

Default MD configuration.

• struct TRDP_MEM_CONFIG_T

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

struct TRDP_PROCESS_CONFIG_T

Various flags/general TRDP options for library initialization.

struct TRDP_IDX_TABLE_T

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

Macros

• #define USE HEAP 0

If this is set, we can allocate dynamically memory

• #define TRDP_FLAGS_DEFAULT 0u

Various flags for PD and MD packets

• #define TRDP_FLAGS_NONE 0x01u

No flags set

- #define TRDP_FLAGS_MARSHALL 0x02u
 Optional marshalling/unmarshalling in TRDP stack
- #define TRDP_FLAGS_CALLBACK 0x04u
 Use of callback function
- #define TRDP_FLAGS_TCP 0x08u

 Use TCP for message data
- #define TRDP_FLAGS_FORCE_CB 0x10u
 Force a callback for every received packet
- #define TRDP_FLAGS_TSN 0x20u
 Hard Real Time PD
- #define TRDP_FLAGS_TSN_SDT 0x40u SDT PD
- #define TRDP_FLAGS_TSN_MSDT 0x80u
 Multi SDT PD
- #define TRDP_INFINITE_TIMEOUT 0xfffffffu
 Infinite reply timeout
- #define TRDP_DEFAULT_PD_TIMEOUT 100000u
 Default PD timeout 100ms from 61375-2-3 Table C.7
- #define TRDP_TIMER_GRANULARITY 5000u granularity in us - we allow 5ms now!
- #define TRDP_BOOL8 TRDP_BITSET8

1 bit relevant (equal to zero = false, not equal to zero = true)

#define TRDP_ANTIVALENT8 TRDP_BITSET8

2 bit relevant (0x0 = errror, 0x01 = false, 0x02 = true, 0x03 undefined)

• #define TRDP_OPTION_NONE 0u

Various flags/general TRDP options for library initialization.

#define TRDP_OPTION_BLOCK 0x01u

Default: Use nonblocking I/O calls, polling necessary Set: Read calls will block, use select()

- #define TRDP_OPTION_TRAFFIC_SHAPING 0x02u
 Use traffic shaping distribute packet sending Default: OFF
- #define TRDP_OPTION_NO_REUSE_ADDR 0x04u
 Do not allow re-use of address/port (-> no multihoming) Default: Allow
- #define TRDP_OPTION_NO_MC_LOOP_BACK 0x08u
 Do not allow loop back of multicast traffic Default: Allow
- #define TRDP_OPTION_NO_UDP_CHK 0x10u
 Suppress UDP CRC generation Default: Compute UDP CRC
- #define TRDP_OPTION_WAIT_FOR_DNR 0x20u

Wait for DNR Default: Don't wait

#define TRDP_OPTION_NO_PD_STATS 0x40u

Suppress PD statistics \ Default: Don't suppress

• #define TRDP_OPTION_DEFAULT_CONFIG 0x80u

no XML process config, defaults were used

Typedefs

typedef VOS IP4 ADDR T TRDP IP ADDR T

TRDP general type definitions.

typedef CHAR8 TRDP_NET_LABEL_T[TRDP_MAX_LABEL_LEN]

Definition for usage in network packets, not necessarily \0 terminated!

typedef VOS VERSION T TRDP VERSION T

Version information.

typedef VOS TIMEVAL T TRDP TIME T

Timer value compatible with timeval / select.

• typedef VOS_FDS_T TRDP_FDS_T

File descriptor set compatible with fd_set / select.

typedef VOS_UUID_T TRDP_UUID_T

UUID definition reuses the VOS definition.

typedef struct TRDP_DATASET_T

Dataset definition

typedef TRDP_DATASET_T * pTRDP_DATASET_T

Array of pointers to dataset

typedef VOS_PRINT_DBG_T TRDP_PRINT_DBG_T

TRDP configuration type definitions.

typedef VOS LOG T TRDP LOG T

Categories for logging, reuse of the VOS definition.

typedef TRDP_ERR_T(* TRDP_MARSHALL_T) (void *pRefCon, UINT32 comId, UINT8 *pSrc, UINT32 srcSize, UINT8 *pDst, UINT32 *pDstSize, TRDP_DATASET_T **ppCachedDS)

 $Function \ type \ for \ marshalling \ .$

• typedef TRDP_ERR_T(* TRDP_UNMARSHALL_T) (void *pRefCon, UINT32 comId, UINT8 *pSrc, UINT32 srcSize, UINT8 *pDst, UINT32 *pDstSize, TRDP_DATASET_T **ppCachedDS)

Function type for unmarshalling.

• typedef void(* TRDP_PD_CALLBACK_T) (void *pRefCon, TRDP_APP_SESSION_T appHandle, const TRDP_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 TRDP_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.

5.25.1 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, 2015-2020. All rights reserved.

5.25.2 Macro Definition Documentation

5.25.2.1 TRDP FLAGS DEFAULT

```
#define TRDP_FLAGS_DEFAULT Ou
```

Various flags for PD and MD packets

Default value defined in tlc_openDession will be taken

5.25.3 Typedef Documentation

5.25.3.1 TRDP_IP_ADDR_T

```
typedef VOS_IP4_ADDR_T TRDP_IP_ADDR_T
```

TRDP general type definitions.

5.25.3.2 TRDP_MARSHALL_T

```
typedef TRDP_ERR_T(* TRDP_MARSHALL_T) (void *pRefCon, UINT32 comId, UINT8 *pSrc, UINT32 src← Size, UINT8 *pDst, UINT32 *pDstSize, TRDP_DATASET_T **ppCachedDS)
```

Function type for marshalling .

The function must know about the dataset's alignment etc.

Parameters

in	pRefCon	pointer to user context
in	comId	Comld to identify the structure out of a configuration
in	pSrc	pointer to received original message
in	srcSize	size of the source buffer
in	pDst	pointer to a buffer for the treated message
in,out	pDstSize	size of the provide buffer / size of the treated message
in,out	ppCachedDS	pointer to pointer of cached dataset

Return values

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

5.25.3.3 TRDP_MD_CALLBACK_T

typedef void(* TRDP_MD_CALLBACK_T) (void *pRefCon, TRDP_APP_SESSION_T appHandle, const TRDP_MD_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

5.25.3.4 TRDP_PD_CALLBACK_T

typedef void(* TRDP_PD_CALLBACK_T) (void *pRefCon, TRDP_APP_SESSION_T appHandle, const TRDP_PD_INFO_T
*pMsg, UINT8 *pData, UINT32 dataSize)

Callback for receiving indications, timeouts, releases, responses.

Parameters

in	pRefCon	pointer to user context
in	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

5.25.3.5 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.

5.25.3.6 TRDP_TIME_T

typedef VOS_TIMEVAL_T TRDP_TIME_T

Timer value compatible with timeval / select.

Relative or absolute date, depending on usage

5.25.3.7 TRDP_UNMARSHALL_T

typedef TRDP_ERR_T(* TRDP_UNMARSHALL_T) (void *pRefCon, UINT32 comId, UINT8 *pSrc, UINT32 src← 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 configuration
in	pSrc	pointer to received original message
in	srcSize	data length from TRDP packet header
in	pDst	pointer to a buffer for the treated message
in,out	pDstSize	size of the provide buffer / size of the treated message
in,out	ppCachedDS	pointer to pointer of cached dataset

Return values

TRDP_NO_ERR	no error
TRDP_MEM_ERR	provide buffer to small
TRDP_COMID_ERR	comid not existing

5.25.4 Enumeration Type Documentation

5.25.4.1 TRDP_DATA_TYPE_T

enum TRDP_DATA_TYPE_T

TRDP dataset description definitions.

Dataset element definition

Enumerator

Eliulileialoi	
TRDP_INVALID	Invalid/unknown
TRDP_BITSET8	=UINT8
TRDP_CHAR8	char, can be used also as UTF8
TRDP_UTF16	Unicode UTF-16 character
TRDP_INT8	Signed integer, 8 bit
TRDP_INT16	Signed integer, 16 bit
TRDP_INT32	Signed integer, 32 bit
TRDP_INT64	Signed integer, 64 bit
TRDP_UINT8	Unsigned integer, 8 bit
TRDP_UINT16	Unsigned integer, 16 bit

Enumerator

TRDP_UINT32	Unsigned integer, 32 bit
TRDP_UINT64	Unsigned integer, 64 bit
TRDP_REAL32	Floating point real, 32 bit
TRDP_REAL64	Floating point real, 64 bit
TRDP_TIMEDATE32	32 bit UNIX time
TRDP_TIMEDATE48	48 bit TCN time (32 bit UNIX time and 16 bit ticks)
TRDP_TIMEDATE64	32 bit UNIX time + 32 bit microseconds
TRDP_TYPE_MAX	Values greater are considered nested datasets

5.25.4.2 TRDP_ERR_T

enum TRDP_ERR_T

Return codes for all API functions, -1..-29 taken over from vos.

Enumerator

or out of range
or out of range
nitialization
ndle
e: no data received
on not supported
ata can't be received/sent
available
ailable
9

Enumerator

Enumerator	
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
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

5.25.4.3 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

5.25.4.4 TRDP_REPLY_STATUS_T

```
enum TRDP_REPLY_STATUS_T
```

TRDP data transfer type definitions.

Reply status messages

5.25.4.5 TRDP_TO_BEHAVIOR_T

```
enum TRDP_TO_BEHAVIOR_T
```

How invalid PD shall be handled

Enumerator

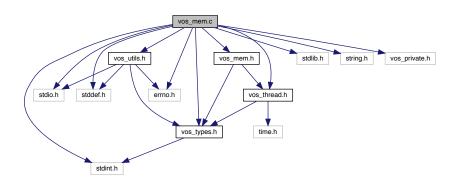
TRDP_TO_DEFAULT	Default value defined in tlc_openDession will be taken
TRDP_TO_SET_TO_ZERO	If set, data will be reset to zero on time out
TRDP_TO_KEEP_LAST_VALUE	If set, last received values will be returned

5.26 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

EXT_DECL VOS_ERR_T vos_memInit (UINT8 *pMemoryArea, UINT32 size, const UINT32 frag
 — Mem[VOS_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.

String concatenation with length limitation.

- 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)
- 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.

5.26.1 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.

5.26.2 Function Documentation

5.26.2.1 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 Generated by Doxygen
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

5.26.2.2 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

5.26.2.3 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

5.26.2.4 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 used	
--	----	-------------	-----------------------------	--

5.26.2.5 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
--	----	-----------	-------------------------------------

5.26.2.6 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

5.26.2.7 vos_qsort()

Sort an array.

This is just a wrapper for the standard qsort function.

Parameters

in,out	pBuf	Pointer to the array to sort
in	num	number of elements
in	size	size of one element
in	compare	Pointer to compare function return -n if $arg1 < arg2$, return 0 if $arg1 == arg2$, return +n if $arg1 > arg2$ where n is an integer != 0
		+11 II dig1 / dig2 where it is an integer := 0

Return values

```
none
```

5.26.2.8 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	out 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

5.26.2.9 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

5.26.2.10 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

5.26.2.11 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

5.26.2.12 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
```

5.26.2.13 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
```

5.26.2.14 vos_strnicmp()

Case insensitive string compare.

in	pStr1	Null terminated string to compare
in	pStr2 Null terminated string to compare	
in	count	Maximum number of characters to compare

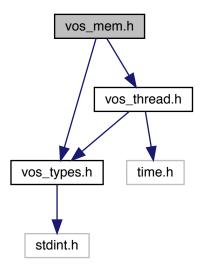
Return values

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

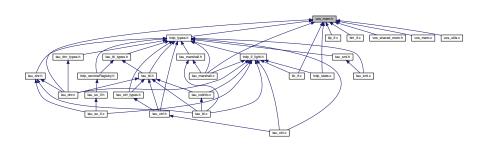
5.27 vos_mem.h File Reference

Memory and queue functions for OS abstraction.

```
#include "vos_types.h"
#include "vos_thread.h"
Include dependency graph for vos_mem.h:
```



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



Macros

#define VOS_MEM_NBLOCKSIZES 15u

No of pre-defined block sizes.

#define VOS MEM MAX PREALLOCATE 10u

We internally allocate memory always by these block sizes.

• #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 frag
 — Mem[VOS_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.

5.27.1 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.

5.27.2 Macro Definition Documentation

5.27.2.1 VOS_MEM_MAX_PREALLOCATE

```
#define VOS_MEM_MAX_PREALLOCATE 10u
```

We internally allocate memory always by these block sizes.

The largest available block is 524288 Bytes, provided the overal size of the used memory allocation area is larger. Max. no. of blocks to pre-allocate

5.27.2.2 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.

5.27.3 Function Documentation

5.27.3.1 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

Daintan	to formed alamanatan NILILI
Pointer	to found element or NULL

5.27.3.2 vos_memAlloc()

Allocate a block of memory (from memory area above).

Parameters

in	size	Size of requested block
	00	0.=0 0.1040.0000

Return values

Pointer	to memory area
NULL if no memory availab	

5.27.3.3 vos_memCount()

```
EXT_DECL VOS_ERR_T vos_memCount (

UINT32 * pAllocatedMemory,

UINT32 * pFreeMemory,

UINT32 * pMinFree,

UINT32 * pNumAllocBlocks,

UINT32 * pNumAllocErr,

UINT32 * pNumFreeErr,

UINT32 blockSize[VOS_MEM_NBLOCKSIZES],

UINT32 usedBlockSize[VOS_MEM_NBLOCKSIZES])
```

Return used and available memory (of memory area above).

Parameters

out	pAllocatedMemory	Pointer to allocated memory size
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

5.27.3.4 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.

in	pMemoryArea	Pointer to memory area used

5.27.3.5 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
- 1		J	,

5.27.3.6 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		Pointer to memory area to use	
	in	in size Size of provided memory area	
in fragMem Pointer to list of preallocate block sizes, used to fragment memory for large b			

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.

in pMemoryArea Pointer to memory area to use		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		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

5.27.3.7 vos_qsort()

Sort an array.

This is just a wrapper for the standard qsort function.

Parameters

in,out	pBuf	Pointer to the array to sort	
in	num	number of elements	
in	size	size of one element	
in	compare	Pointer to compare function return -n if arg1 < arg2, return 0 if arg1 == arg2, return	
		+n if arg1 > arg2 where n is an integer != 0	

Return values

```
none
```

5.27.3.8 vos_queueCreate()

Initialize a message queue.

Returns a handle for further calls

	in queueType in maxNoOfMsg		Define queue type (1 = FIFO, 2 = LIFO, 3 = PRIO)
			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

5.27.3.9 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

5.27.3.10 vos_queueReceive()

```
EXT_DECL VOS_ERR_T vos_queueReceive (

VOS_QUEUE_T queueHandle,

UINT8 ** ppData,

UINT32 * pSize,

UINT32 usTimeout )
```

Get a message.

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

5.27.3.11 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

5.27.3.12 vos_strncat()

String concatenation with length limitation.

in	pStrDst	Destination string
in	count	Size of destination buffer
in	pStrSrc	Null terminated string to append

Return values

none

5.27.3.13 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

5.27.3.14 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

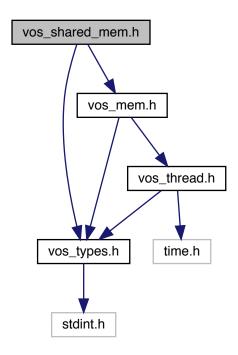
0	- equal	
<0	- string1 less than string 2	
>0	- string 1 greater than string 2	

5.28 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

• EXT_DECL VOS_ERR_T vos_sharedOpen (const CHAR8 *pKey, VOS_SHRD_T *pHandle, UINT8 **pp↔ MemoryArea, UINT32 *pSize)

Create a shared memory area or attach to existing one.

• EXT_DECL VOS_ERR_T vos_sharedClose (VOS_SHRD_T handle, const UINT8 *pMemoryArea)

Close connection to the shared memory area.

5.28.1 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.

5.28.2 Function Documentation

5.28.2.1 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

5.28.2.2 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

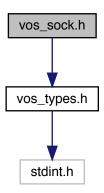
Return values

VOS_NO_ERR	no error
VOS_MEM_ERR	no memory available

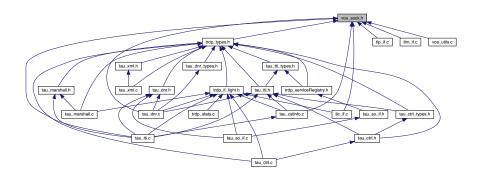
5.29 vos_sock.h File Reference

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 socklnit (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.
- $\bullet \ \ \mathsf{EXT_DECL} \ \ \mathsf{VOS_ERR_T} \ \ \mathsf{vos_sockSetMulticastIf} \ \ (\mathsf{SOCKET} \ \mathsf{sock}, \ \mathsf{UINT32} \ \ \mathsf{mclfAddress})$

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.

5.29.1 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.

5.29.2 Macro Definition Documentation

5.29.2.1 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

5.29.2.2 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

5.29.3 Function Documentation

5.29.3.1 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

Address	to bind to
---------	------------

5.29.3.2 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
---------	-----------------------------

5.29.3.3 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

Return values

VOS_MEM_ERR	memory allocation error
VOS_SOCK_ERR	GetAdaptersInfo() error

5.29.3.4 vos_htonl()

Byte swapping 4 Bytes.

Parameters

in <i>val</i>	Initial value.
---------------	----------------

Return values

swapped valu

5.29.3.5 vos_htonII()

Byte swapping 8 Bytes.

Parameters

in	val	Initial value.

Return values

```
swapped value
```

5.29.3.6 vos_htons()

Byte swapping 2 Bytes.

Parameters

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

Return values

```
swapped value
```

5.29.3.7 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.
```

5.29.3.8 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

5.29.3.9 vos_netIfUp()

```
EXT_DECL BOOL8 vos_netIfUp ( {\tt VOS\_IP4\_ADDR\_T} \ ifAddress \ )
```

Get the state of an interface.

Parameters

in	ifAddress	address of interface to check
----	-----------	-------------------------------

Return values

TRUE interface is up and ready FALSE interface is down / not ready

5.29.3.10 vos_ntohl()

Byte swapping 4 Bytes.

Parameters

in	val	Initial value.

Return values

```
swapped value
```

5.29.3.11 vos_ntohll()

Byte swapping 8 Bytes.

Parameters

in	val	Initial value.

Return values

swapped	value

5.29.3.12 vos_ntohs()

Byte swapping 2 Bytes.

Parameters

|--|

Return values

```
swapped value
```

5.29.3.13 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
```

5.29.3.14 vos_sockAccept()

```
EXT_DECL VOS_ERR_T vos_sockAccept ( SOCKET sock,
```

```
SOCKET * pSock,
UINT32 * pIPAddress,
UINT16 * pPort )
```

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

5.29.3.15 vos_sockBind()

Bind a socket to an address and port.

Parameters

in	sock	socket descriptor
in	ipAddress	source IP to receive from, 0 for any
in	port	port to receive from

Return values

VOS_NO_ERR	no error
VOS_PARAM_ERR	parameter out of range/invalid
VOS_IO_ERR	Input/Output error
VOS_MEM_ERR	resource error

5.29.3.16 vos_sockClose()

```
{\tt EXT\_DECL~VOS\_ERR\_T~vos\_sockClose~(}
```

```
SOCKET sock )
```

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

5.29.3.17 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

5.29.3.18 vos_sockGetMAC()

Return the MAC address of the default adapter.

out	рМАС	return MAC address.
-----	------	---------------------

Return values

VOS_NO_ERR	no error
VOS_PARAM_ERR	pMAC == NULL
VOS_SOCK_ERR	socket not available or option not supported

5.29.3.19 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

5.29.3.20 vos_sockJoinMC()

Join a multicast group.

Note: Some target systems might not support this option.

Parameters

i	.n	sock	socket descriptor
i	.n	mcAddress	multicast group to join
i	.n	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

5.29.3.21 vos_sockLeaveMC()

Leave a multicast group.

Note: Some target systems might not support this option.

Parameters

	in	sock	socket descriptor
ſ	in	mcAddress	multicast group to join
ľ	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

5.29.3.22 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

5.29.3.23 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	

5.29.3.24 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

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	

5.29.3.25 vos_sockReceiveTCP()

```
EXT_DECL VOS_ERR_T vos_sockReceiveTCP (
SOCKET sock,
```

```
UINT8 * pBuffer,
UINT32 * pSize )
```

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

5.29.3.26 vos_sockReceiveUDP()

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.

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	

5.29.3.27 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

5.29.3.28 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

5.29.3.29 vos_sockSetMulticastIf()

Set Using Multicast I/F.

Parameters

in	sock	socket descriptor
in	mclfAddress	using Multicast I/F Address

Return values

VOS_NO_ERR	no error
VOS_PARAM_ERR	sock descriptor unknown, parameter error

5.29.3.30 vos_sockSetOptions()

Set socket options.

Note: Some target systems might not support each option.

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

Return values

VOS_NO_ERR	no error
VOS_PARAM_ERR	parameter out of range/invalid

5.29.3.31 vos_sockTerm()

```
\begin{tabular}{ll} {\tt EXT\_DECL} & {\tt void} & {\tt vos\_sockTerm} & (\\ & & {\tt void} & ) \end{tabular}
```

De-Initialize the socket library.

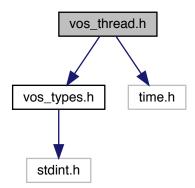
Must be called after last socket call

5.30 vos_thread.h File Reference

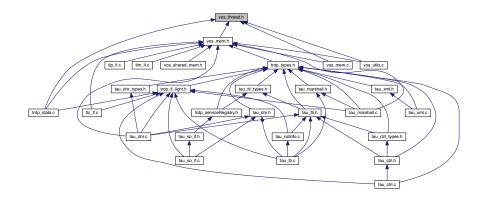
Threading functions for OS abstraction.

```
#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, VOS_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, VOS_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_threadIsActive (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.

EXT_DECL VOS_ERR_T vos_semaCreate (VOS_SEMA_T *pSema, VOS_SEMA_STATE_T initialState)
 Create a semaphore.

EXT_DECL void vos_semaDelete (VOS_SEMA_T sema)

Delete a semaphore.

• EXT_DECL VOS_ERR_T vos_semaTake (VOS_SEMA_T sema, UINT32 timeout)

Take a semaphore.

• EXT_DECL void vos_semaGive (VOS_SEMA_T sema)

Give a semaphore.

5.30.1 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.

5.30.2 Function Documentation

5.30.2.1 vos_addTime()

Add the second to the first time stamp, return sum in first.

in,out	pTime	Pointer to time value
in	pAdd	Pointer to time value

5.30.2.2 vos_clearTime()

Clear the time stamp.

Parameters

out	pTime	Pointer to time value
-----	-------	-----------------------

5.30.2.3 vos_cmpTime()

Compare the second from the first time stamp, return diff in first.

Parameters

in, ou	ıt <i>pTime</i>	Pointer to time value
in	рСтр	Pointer to time value to compare

Return values

0	pTime == pCmp
-1	pTime < pCmp
1	pTime > pCmp

5.30.2.4 vos_divTime()

Divide the first time by the second, return quotient in first.

in,out	pTime	Pointer to time value
in	divisor	Divisor

5.30.2.5 vos_getRealTime()

Return the current real time in sec and us.

Parameters

out <i>pTime</i>	Pointer to time value
------------------	-----------------------

5.30.2.6 vos getTime()

Return the current monotonic time in sec and us.

Parameters

out	pTime	Pointer to time value	
-----	-------	-----------------------	--

5.30.2.7 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"
-----------	------------------------

5.30.2.8 vos_getUuid()

Get a universal unique identifier according to RFC 4122 time based version.

Parameters

out	pUuID	Pointer to a universal unique identifier	
-----	-------	--	--

5.30.2.9 vos_mulTime()

Multiply the first time by the second, return product in first.

Parameters

in,out	pTime	Pointer to time value
in	mul	Factor

5.30.2.10 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

5.30.2.11 vos_mutexDelete()

Delete a mutex.

Release the resources taken by the mutex.

Parameters

in	pMutex	mutex handle
----	--------	--------------

Return values

```
VOS_NO_ERR no error
```

5.30.2.12 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

5.30.2.13 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

5.30.2.14 vos_mutexUnlock()

Release a mutex.

Unlock the mutex.

Parameters

in <i>pMutex</i> mutex handle

5.30.2.15 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

5.30.2.16 vos_semaDelete()

Delete a semaphore.

This will eventually release any processes waiting for the semaphore.

Parameters

in	sema	semaphore handle
----	------	------------------

5.30.2.17 vos_semaGive()

```
EXT_DECL void vos_sema
Give ( \begin{tabular}{ll} VOS\_SEMA\_T & sema \end{tabular} \label{eq:vos_sema}
```

Give a semaphore.

Release (increase) a semaphore.

Parameters

in sema semaphore handle	in	sema	semaphore handle
--------------------------	----	------	------------------

5.30.2.18 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

5.30.2.19 vos_subTime()

```
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.

Parameters

in,out	pTime	Pointer to time value
in	pSub	Pointer to time value

5.30.2.20 vos_threadCreate()

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

VOC NO EDD	DO OFFICE
VOS_NO_ERR	no error
VOS INIT ERR	module not initialised
VOS_NOINIT_ERR	invalid handle
VOS PARAM ERR	parameter out of range/invalid

5.30.2.21 vos_threadCreateSync()

Create a thread.

Create a thread and return a thread handle for further requests. Not each parameter may be supported by all target systems!

Parameters

out	pThread	Pointer to returned thread handle
in	pName	Pointer to name of the thread (optional)
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

5.30.2.22 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

5.30.2.23 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

5.30.2.24 vos_threadIsActive()

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

in	thread	Thread 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

5.30.2.25 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

5.30.2.26 vos_threadTerm()

De-Initialize the thread library.

Must be called after last thread/timer call

5.30.2.27 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

5.31 vos_types.h File Reference

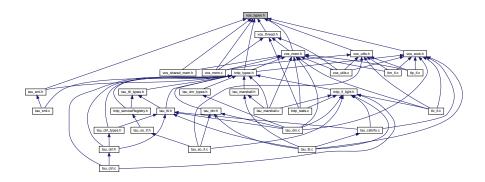
Typedefs for OS abstraction.

#include <stdint.h>
Include dependency graph for vos_types.h:

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
```

5.31.1 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.

5.31.2 Typedef Documentation

5.31.2.1 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

5.31.2.2 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

5.31.3 Enumeration Type Documentation

5.31.3.1 VOS_ERR_T

enum VOS_ERR_T

Return codes for all VOS API functions

Enumerator

VOS_NO_ERR	No error

Enumerator

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

5.31.3.2 VOS_LOG_T

enum VOS_LOG_T

Categories for logging

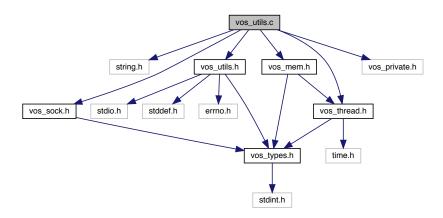
Enumerator

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

5.32 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.

5.32.1 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.

5.32.2 Function Documentation

5.32.2.1 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	

5.32.2.2 vos_getErrorString()

Return a human readable error representation.

Parameters

in error The TRDP o	or VOS error code
---------------------	-------------------

Return values

const	string pointer to error string
-------	--------------------------------

5.32.2.3 vos_getVersion()

Return version.

Return pointer to version structure

Return values

```
VOS_VERSION_T
```

5.32.2.4 vos_getVersionString()

Return a human readable version representation.

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

Return values

```
const string
```

5.32.2.5 vos_hostlsBigEndian()

```
int vos\_hostIsBigEndian (
```

```
void )
```

Return 1 if this is a big endian machine.

Return values

0	if machine is little endian
1	if machine is big endian

5.32.2.6 vos_init()

Initialize the virtual operating system.

Initialize the vos library.

Parameters

	in	pRefCon	context for debug output function
ſ	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		

5.32.2.7 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
------	----------------------------

5.32.2.8 vos_terminate()

```
\begin{tabular}{ll} EXT\_DECL & void & vos\_terminate & ( \\ & void & ) \end{tabular}
```

Delnitialize the vos library.

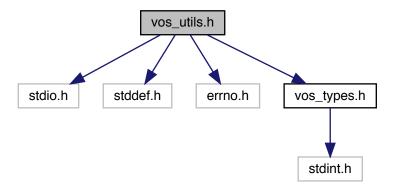
Should be called last after TRDP stack/application does not use any VOS function anymore.

5.33 vos_utils.h File Reference

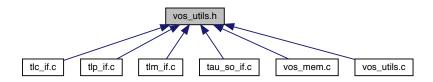
Typedefs for OS abstraction.

```
#include <stdio.h>
#include <stddef.h>
#include <errno.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)

Max

#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_hostIsBigEndian (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)

DeInitialize 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.

5.33.1 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.

5.33.2 Macro Definition Documentation

5.33.2.1 INITFCS

#define INITFCS Oxffffffffu

CRC/FCS constants.

Initial FCS value

5.33.2.2 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

5.33.2.3 VOS_MAX_FRMT_SIZE

#define VOS_MAX_FRMT_SIZE 64u

Мах.

size of the 'format' part

5.33.2.4 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

5.33.3 Function Documentation

5.33.3.1 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

5.33.3.2 vos_getErrorString()

Return a human readable error representation.

Parameters

in error The TRDP or VOS error cod	е
------------------------------------	---

Return values

const string pointer to error string

5.33.3.3 vos_getVersion()

Return version.

Return pointer to version structure

Return values

const VOS_VERSION_T

Return pointer to version structure

Return values

VOS_VERSION_T

5.33.3.4 vos_getVersionString()

Return a human readable version representation.

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

Return values

const	string
-------	--------

5.33.3.5 vos_hostlsBigEndian()

```
\begin{tabular}{lll} 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

5.33.3.6 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

in	pRefCon	context for debug output function
in	pDebugOutput	Pointer to debug output function.

Return values

VOS_NO_ERR	no error VOS_INTEGRATION_ERR if endianess/alignment mismatch VOS_SOCK_ERR	Ī
	sockets not supported VOS_UNKNOWN_ERR initialisation error	

5.33.3.7 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
```

5.33.3.8 vos_terminate()

Delnitialize the vos library.

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

Index

callBack	service info, 31
GNU PACKED, 20	fetId
cnCnt	TRDP FUNCTION INFO T, 44
TRDP ETB INFO T, 43	71121 _1 010011011_111 0_1, 11
cnld	GNU PACKED, 11
TRDP FUNCTION INFO T, 44	callBack, 20
comId	comld, 20
GNU PACKED, 20	confVehCnt, 20
confVehCnt	confVehList, 20
GNU PACKED, 20	cstCnt, 20
confVehList	cstList, 20
GNU PACKED, 20	cstUUID, 21
cstCnt	defQos, 21
GNU_PACKED, 20	defTtl, 21
cstld	deviceName, 21
TRDP_CONSIST_INFO_T, 37	etbld, 22
cstInfoGetPropSize	etbInhibit, 22
tau_cstinfo.c, 72	etbLength, 22
cstList	etbShort, 22
GNU_PACKED, 20	etbTopoCnt, 22
cstOwner	inhibit, 23
TRDP_CONSIST_INFO_T, 37	isLead, 23
cstUUID	joinedAddr, 23
GNU_PACKED, 21	leadDir, 23
cstVehNo	leadVehOfCst, 23
TRDP_FUNCTION_INFO_T, 44	numCrcErr, 24
	numMissed, 24
defQos	numProtErr, 24
GNU_PACKED, 21	numRev, 24
defTtl	numRecv, 24 numSend, 24
GNU_PACKED, 21	numTopoErr, 25
deviceName	opCstList, 25
GNU_PACKED, 21	opTrnDirState, 25
DNS_HEADER, 11	opTrnTopoCnt, 25
ETB CTRL COMID	opVehList, 25
iec61375-2-3.h, 67	ownOpCstNo, 26
etbld	reserved01, 26
GNU PACKED, 22	reserved02, 26
TRDP_FUNCTION_INFO_T, 44	reserved03, 27
etbInhibit	reserved04, 27
GNU_PACKED, 22	reserved06, 27
etbLength	safetyTrail, 27
GNU PACKED, 22	serviceEntry, 27
etbShort	timeout, 28
GNU_PACKED, 22	toBehav, 28
etbTopoCnt	trnCstNo, 28
GNU_PACKED, 22	trnDirState, 28
	trnld, 28
fctDev	trnNetDir, 29

trnOperator, 29	GNU PACKED, 24
trnTopoCnt, 29	numMissed
•	
trnVehNo, 29	GNU_PACKED, 24
vehld, 29	numProtErr
vehOrient, 29	GNU PACKED, 24
version, 30	numRcv
voroion, oo	
iec61375-2-3.h, 61	GNU_PACKED, 24
	numRecv
ETB_CTRL_COMID, 67	GNU_PACKED, 24
TRDP_ETBCTRL_DSID, 67	numSend
TRDP_MAX_FILE_NAME_LEN, 67	GNU PACKED, 24
TRDP MAX LABEL LEN, 67	numTopoErr
TRDP MAX MD DATA SIZE, 67	•
:	GNU_PACKED, 25
TRDP_MAX_URI_HOST_LEN, 68	
TRDP_MAX_URI_LEN, 68	opCstList
TRDP_MAX_URI_USER_LEN, 68	GNU PACKED, 25
TRDP MD DEFAULT REPLY TIMEOUT, 68	opTrnDirState
TRDP MD INFINITE TIME, 68	•
:	GNU_PACKED, 25
TRDP_MIN_PD_HEADER_SIZE, 68	opTrnTopoCnt
TRDP_MSG_PD, 69	GNU_PACKED, 25
TRDP_PD_UDP_PORT, 69	opVehList
TRDP PROCESS DEFAULT CYCLE TIME, 69	GNU PACKED, 25
TRDP PROTOCOL VERSION CHECK MASK,	<u> </u>
	ownOpCstNo
69	GNU_PACKED, 26
TRDP_USR_URI_SIZE, 69	
TTDB_NET_DIR_REQ_COMID, 70	reserved01
TTDB OP DIR INFO COMID, 70	GNU PACKED, 26
TTDB STAT CST REQ COMID, 70	reserved02
:	
TTDB_TRN_DIR_REQ_COMID, 70	GNU_PACKED, 26
inhibit	reserved03
GNU_PACKED, 23	GNU_PACKED, 27
INITFCS	reserved04
vos_utils.h, 325	GNU PACKED, 27
isLead	
	reserved06
GNU_PACKED, 23	GNU_PACKED, 27
joinedAddr	safetyTrail
GNU_PACKED, 23	GNU PACKED, 27
	service_info, 30
leadDir	
GNU_PACKED, 23	fctDev, 31
leadVehOfCst	serviceEntry
GNU PACKED, 23	GNU_PACKED, 27
GNO_FACKED, 23	SOA SAME SERVICEID
maxNoOfExtPublishers	trdp_serviceRegistry.h, 240
	SRM SERVICE READ REQ COMID
TRDP_IDX_TABLE_T, 45	
maxNoOfHighCatPublishers	trdp_serviceRegistry.h, 241
TRDP_IDX_TABLE_T, 45	SRM_SRVINFO_NOTIFY_COMID
maxNoOfHighCatSubscriptions	trdp_serviceRegistry.h, 241
TRDP IDX TABLE T, 46	srv info req, 32
	51 V_11110_10q1, 02
maxNoOfLowCatPublishers	tour addrOllvi
TRDP_IDX_TABLE_T, 46	tau_addr2Uri
maxNoOfLowCatSubscriptions	tau_dnr.c, 87
TRDP IDX TABLE T, 46	tau_dnr.h, 92
maxNoOfMidCatPublishers	tau addService
	tau_so_if.c, 117
TRDP_IDX_TABLE_T, 46	
maxNoOfMidCatSubscriptions	tau_so_if.h, 122
TRDP_IDX_TABLE_T, 46	tau_calcDatasetSize
	tau_marshall.c, 100
numCrcErr	tau_marshall.h, 108
	_ ,

tau_calcDatasetSizeByComId	tau_xml.h, 165
tau_marshall.c, 101	tau_getCstFctCnt
tau_marshall.h, 109	tau_tti.c, 127
tau_cstinfo.c, 71	tau_tti.h, 139
cstInfoGetPropSize, 72	tau_getCstFctInfo
tau_ctrl.c, 73	tau_tti.c, 128
tau_getEcspStat, 75	tau_tti.h, 139
tau_initEcspCtrl, 75	tau_getCstInfo
tau_requestEcspConfirm, 76	tau_tti.c, 128
tau_setEcspCtrl, 76	tau_tti.h, 140
tau_terminateEcspCtrl, 77	tau_getCstVehCnt
tau_ctrl.h, 77	tau_tti.c, 129
tau_getEcspStat, 80	tau_tti.h, 140
tau_initEcspCtrl, 80	tau_getEcspStat
tau_requestEcspConfirm, 81	tau_ctrl.c, 75
tau_setEcspCtrl, 81	tau_ctrl.h, 80
tau_terminateEcspCtrl, 82	tau_getOpTrDirectory
tau_ctrl_types.h, 82	tau_tti.c, 129
tau_deInitDnr	tau_tti.h, 141
tau_dnr.c, 87	tau_getOpTrnDirectoryStatusInfo
tau_dnr.h, 93	tau_tti.c, 129
tau_deInitTTI	tau_tti.h, 142
tau_tti.c, 127	tau_getOwnAddr
tau_tti.h, 138	tau_dnr.c, 88
tau_delService	tau_dnr.h, 94
tau_so_if.c, 118	tau_getOwnIds
tau_so_if.h, 122	tau_tti.c, 130
tau_dnr.c, 85	tau_tti.h, 142
tau_addr2Uri, 87	tau_getOwnOpCstNo
tau_deInitDnr, 87	tau_tti.c, 130
tau_DNRstatus, 87	tau_tti.h, 143
tau_getOwnAddr, 88	tau_getOwnTrnCstNo
tau_initDnr, 88	tau_tti.c, 131
tau_uri2Addr, 89	tau_tti.h, 143
tau_dnr.h, 90	tau_getServicesList
tau_addr2Uri, 92	tau_so_if.c, 118
tau_deInitDnr, 93	tau_so_if.h, 123
tau_DNRstatus, 94	tau_getStaticCstInfo
tau_getOwnAddr, 94	tau_tti.c, 131
tau_initDnr, 95	tau_tti.h, 143
tau_uri2Addr, 96	tau_getTrDirectory
TRDP_DNR_OPTS, 92	tau_tti.c, 132
TRDP_DNR_OWN_THREAD, 92	tau_tti.h, 144
tau_dnr_types.h, 97	tau_getTrnCstCnt
tau_DNRstatus	tau_tti.c, 132
tau_dnr.c, 87	tau_tti.h, 145
tau_dnr.h, 94	tau_getTrnVehCnt
tau_freeServicesList	tau_tti.c, 132
tau_so_if.c, 118	tau_tti.h, 145
tau_so_if.h, 123	tau_getTTI
tau_freeTelegrams	tau_tti.c, 133
tau_xml.c, 154	tau_tti.h, 146
tau_xml.h, 164	tau_getVehInfo
tau_freeXmlDatasetConfig	tau_tti.c, 133
tau_xml.c, 154	tau_tti.h, 146
tau_xml.h, 164	tau_getVehOrient
tau_freeXmlDoc	tau_tti.c, 134
tau_xml.c, 155	tau_tti.h, 147

tau_initDnr	tau_xml.h, 169
tau_dnr.c, 88	tau_readXmlServiceConfig
tau_dnr.h, 95	tau_xml.c, 159
tau_initEcspCtrl	tau_xml.h, 170
tau_ctrl.c, 75	tau_requestEcspConfirm
tau_ctrl.h, 80	tau_ctrl.c, 76
tau_initMarshall	tau_ctrl.h, 81
tau_marshall.c, 102	tau_setEcspCtrl
tau marshall.h, 110	tau_ctrl.c, 76
tau_initTTlaccess	tau_ctrl.h, 81
tau_tti.c, 134	tau_so_if.c, 116
tau_tti.h, 148	tau_addService, 117
tau_marshall	tau_delService, 118
tau_marshall.c, 102	tau freeServicesList, 118
tau_marshall.h, 111	tau_getServicesList, 118
tau_marshall.c, 99	tau updService, 119
tau_calcDatasetSize, 100	tau so if.h, 120
tau_calcDatasetSizeByComld, 101	tau addService, 122
tau initMarshall, 102	tau_delService, 122
tau_marshall, 102	tau freeServicesList, 123
	tau getServicesList, 123
tau_marshallDs, 104	tau_getServicesList, 123
tau_unmarshall, 105	— ·
tau_unmarshallDs, 105	tau_terminateEcspCtrl
tau_marshall.h, 106	tau_ctrl.c, 77
tau_calcDatasetSize, 108	tau_ctrl.h, 82
tau_calcDatasetSizeByComld, 109	tau_tti.c, 124
tau_initMarshall, 110	tau_delnitTTI, 127
tau_marshall, 111	tau_getCstFctCnt, 127
tau_marshallDs, 112	tau_getCstFctInfo, 128
tau_unmarshall, 113	tau_getCstInfo, 128
tau_unmarshallDs, 114	tau_getCstVehCnt, 129
TAU_MARSHALL_INFO_T, 32	tau_getOpTrDirectory, 129
tau_marshallDs	tau_getOpTrnDirectoryStatusInfo, 129
tau_marshall.c, 104	tau_getOwnlds, 130
tau_marshall.h, 112	tau_getOwnOpCstNo, 130
tau_prepareXmlDoc	tau_getOwnTrnCstNo, 131
tau_xml.c, 155	tau_getStaticCstInfo, 131
tau_xml.h, 165	tau_getTrDirectory, 132
tau_prepareXmlMem	tau_getTrnCstCnt, 132
tau_xml.c, 156	tau_getTrnVehCnt, 132
tau_xml.h, 166	tau_getTTI, 133
tau_readXmlDatasetConfig	tau_getVehInfo, 133
tau_xml.c, 156	tau_getVehOrient, 134
tau_xml.h, 166	tau_initTTlaccess, 134
tau_readXmlDeviceConfig	TTI_CACHED_CONSISTS, 127
tau_xml.c, 156	tau_tti.h, 135
tau xml.h, 166	tau_deInitTTI, 138
tau_readXmlInterfaceConfig	tau_getCstFctCnt, 139
tau_xml.c, 157	tau_getCstFctInfo, 139
tau xml.h, 167	tau_getCstInfo, 140
tau_readXmlMappedDeviceConfig	tau_getCstVehCnt, 140
tau_xml.c, 158	tau_getOpTrDirectory, 141
tau_xml.h, 168	tau_getOpTrnDirectoryStatusInfo, 142
tau_readXmlMappedDevices	tau getOwnlds, 142
tau_xml.c, 158	tau_getOwnOpCstNo, 143
tau_xml.h, 169	tau_getOwnTrnCstNo, 143
tau_readXmlMappedInterfaceConfig	tau_getStaticCstInfo, 143
tau xml.c, 159	tau_getTrDirectory, 144
	<u></u>

tau_getTrnCstCnt, 145	GNU_PACKED, 28
tau_getTrnVehCnt, 145	tlc_closeSession
tau_getTTI, 146	tlc_if.c, 172
tau_getVehInfo, 146	trdp_if_light.h, 207
tau_getVehOrient, 147	tlc_configSession
tau_initTTlaccess, 148	tlc_if.c, 173
tau_tti_types.h, 149	trdp_if_light.h, 208
tau_unmarshall	tlc_getETBTopoCount
tau_marshall.c, 105	tlc_if.c, 173
tau_marshall.h, 113	trdp_if_light.h, 208
tau_unmarshallDs	tlc_getInterval
tau_marshall.c, 105	tlc_if.c, 174
tau_marshall.h, 114	trdp_if_light.h, 209
tau_updService	tlc_getJoinStatistics
tau_so_if.c, 119	trdp_if_light.h, 209
tau_so_if.h, 124	trdp_stats.c, 243
tau_uri2Addr	tlc_getOpTrainTopoCount
tau_dnr.c, 89	tlc_if.c, 174
tau_dnr.h, 96	trdp_if_light.h, 210
tau_xml.c, 152	tlc_getOwnlpAddress
tau_freeTelegrams, 154	tlc_if.c, 175
tau_freeXmlDatasetConfig, 154	trdp_if_light.h, 210
tau_freeXmlDoc, 155	tlc_getPubStatistics
tau_prepareXmlDoc, 155	trdp_if_light.h, 210
tau_prepareXmlMem, 156	trdp_stats.c, 243
tau_readXmlDatasetConfig, 156	tlc_getRedStatistics
tau_readXmlDeviceConfig, 156	trdp_if_light.h, 211
tau_readXmlInterfaceConfig, 157	trdp_stats.c, 244
tau_readXmlMappedDeviceConfig, 158	tlc_getStatistics
tau_readXmlMappedDevices, 158	trdp_if_light.h, 211
tau_readXmlMappedInterfaceConfig, 159	trdp_stats.c, 244
tau_readXmlServiceConfig, 159	tlc_getSubsStatistics
TRDP_SDT_DEFAULT_CMTHR, 154	trdp_if_light.h, 212
TRDP_SDT_DEFAULT_LMIMAX, 154	trdp_stats.c, 245
tau_xml.h, 160	tlc_getVersion
tau_freeTelegrams, 164	tlc_if.c, 175
tau_freeXmlDatasetConfig, 164	trdp_if_light.h, 212
tau_freeXmlDoc, 165	tlc_getVersionString
tau_prepareXmlDoc, 165	tlc_if.c, 175
tau_prepareXmlMem, 166	trdp_if_light.h, 213
tau_readXmlDatasetConfig, 166	tlc_if.c, 170
tau_readXmlDeviceConfig, 166	tlc_closeSession, 172
tau_readXmlInterfaceConfig, 167	tlc_configSession, 173
tau_readXmlMappedDeviceConfig, 168	tlc_getETBTopoCount, 173
tau_readXmlMappedDevices, 169	tlc_getInterval, 174
tau_readXmlMappedInterfaceConfig, 169	tlc_getOpTrainTopoCount, 174
tau_readXmlServiceConfig, 170	tlc_getOwnIpAddress, 175
TRDP_DBG_DEFAULT, 163	tlc_getVersion, 175
TRDP_EXCHG_OPTION_T, 163	tlc_getVersionString, 175
TRDP_EXCHG_SINK, 164	tlc_init, 176
TRDP_EXCHG_SOURCE, 164	tlc_openSession, 176
TRDP_EXCHG_SOURCESINK, 164	tlc_presetIndexSession, 177
TRDP_EXCHG_UNSET, 164	tlc_process, 177
TCN_URI, 33	tlc_reinitSession, 178
tcnUriCnt	tlc_setETBTopoCount, 178
TRDP_DNS_REPLY, 41	tlc_setOpTrainTopoCount, 179
TRDP_DNS_REQUEST, 42	tlc_terminate, 179
timeout	tlc_updateSession, 180

trdp_getAccess, 180	tlm_reply, 189
trdp_isValidSession, 181	tlm_replyQuery, 190
trdp_releaseAccess, 181	tlm_request, 190
trdp_sessionQueue, 181	tlm_notify
tlc_init	tlm_if.c, 187
tlc_if.c, 176	trdp_if_light.h, 221
trdp_if_light.h, 213	tlm_process
tlc_openSession	tlm_if.c, 188
tlc_if.c, 176	trdp_if_light.h, 222
trdp_if_light.h, 214	tlm_readdListener
tlc_presetIndexSession	tlm_if.c, 188
tlc_if.c, 177	trdp_if_light.h, 222
trdp_if_light.h, 214	tlm_reply
tlc_process	tlm_if.c, 189
tlc_if.c, 177	trdp_if_light.h, 223
trdp_if_light.h, 215	tlm_replyQuery
tlc_reinitSession	tlm_if.c, 190
tlc_if.c, 178	trdp_if_light.h, 224
trdp_if_light.h, 215	tlm_request
tlc_resetStatistics	tlm_if.c, 190
trdp_if_light.h, 216	trdp_if_light.h, 224
trdp_stats.c, 245	tlp_get
tlc_setETBTopoCount	tlp_if.c, 193
tlc_if.c, 178	trdp_if_light.h, 225
trdp_if_light.h, 216	tlp_getInterval
tlc_setOpTrainTopoCount	tlp_if.c, 194
tlc_if.c, 179	trdp_if_light.h, 226
trdp_if_light.h, 217	tlp_getRedundant
tlc_terminate	tlp_if.c, 194
tlc_if.c, 179	trdp_if_light.h, 227
trdp_if_light.h, 217	tlp_if.c, 191
tlc_updateSession	tlp_get, 193
tlc_if.c, 180	tlp_getInterval, 194
trdp_if_light.h, 217	tlp_getRedundant, 194
tlm_abortSession	tlp_processReceive, 195
tlm_if.c, 184	tlp_processSend, 195
trdp_if_light.h, 218	tlp_publish, 196
tlm_addListener	tlp_put, 197
tlm_if.c, 184	tlp_putImmediate, 197
trdp_if_light.h, 218	tlp_republish, 198
tlm_confirm	tlp_request, 199
tlm_if.c, 185	tlp_resubscribe, 200
trdp_if_light.h, 219	tlp_setRedundant, 200
tlm_delListener	tlp_subscribe, 201
tlm_if.c, 186	tlp_unpublish, 202
trdp_if_light.h, 220	tlp_unsubscribe, 202
tlm_getInterval	tlp_processReceive
tlm_if.c, 186	tlp_if.c, 195
trdp_if_light.h, 220	trdp_if_light.h, 227
tlm_if.c, 182	tlp_processSend
tlm_abortSession, 184	tlp_if.c, 195
tlm_addListener, 184	trdp_if_light.h, 228
tlm_confirm, 185	tlp_publish
tlm_delListener, 186	tlp_if.c, 196
tlm_getInterval, 186	trdp_if_light.h, 228
tlm_notify, 187	tlp_put
tlm_process, 188	tlp_if.c, 197
tlm_readdListener, 188	trdp_if_light.h, 229

tlp_putImmediate	tau_dnr.h, 92
tlp_if.c, 197	TRDP_DNR_OWN_THREAD
trdp_if_light.h, 230	tau_dnr.h, 92
tlp_republish	TRDP_DNS_REPLY, 40
tlp_if.c, 198	tcnUriCnt, 41
trdp_if_light.h, 230	TRDP_DNS_REQUEST, 41
tlp_request	tcnUriCnt, 42
tlp_if.c, 199	TRDP_ERR_T
trdp_if_light.h, 231	trdp_types.h, 260
tlp_resubscribe	TRDP_ETB_INFO_T, 42
tlp_if.c, 200	cnCnt, 43
trdp_if_light.h, 232	TRDP_ETBCTRL_DSID
tlp_setRedundant	iec61375-2-3.h, 67
tlp_if.c, 200	TRDP EXCHG OPTION T
trdp_if_light.h, 233	tau xml.h, 163
tlp_subscribe	TRDP EXCHG SINK
tlp_if.c, 201	tau xml.h, 164
trdp_if_light.h, 233	TRDP EXCHG SOURCE
tlp_unpublish	tau_xml.h, 164
tlp_if.c, 202	TRDP_EXCHG_SOURCESINK
trdp_if_light.h, 235	tau_xml.h, 164
tlp_unsubscribe	TRDP_EXCHG_UNSET
tlp_if.c, 202	tau_xml.h, 164
trdp_if_light.h, 235	TRDP FLAGS DEFAULT
toBehav	trdp_types.h, 256
	TRDP_FUNCTION_INFO_T, 43
GNU_PACKED, 28	
TRDP_APP_CONFIRMTO_ERR	cnld, 44
trdp_types.h, 261	cstVehNo, 44
TRDP_APP_REPLYTO_ERR	etbld, 44
trdp_types.h, 261	fctld, 44
TRDP_APP_TIMEOUT_ERR	trdp_getAccess
trdp_types.h, 261	tlc_if.c, 180
TRDP_BITSET8	TRDP_IDX_TABLE_T, 45
trdp_types.h, 259	maxNoOfExtPublishers, 45
TRDP_BLOCK_ERR	maxNoOfHighCatPublishers, 45
trdp_types.h, 261	maxNoOfHighCatSubscriptions, 46
TRDP_CHAR8	maxNoOfLowCatPublishers, 46
trdp_types.h, 259	maxNoOfLowCatSubscriptions, 46
TRDP_CLTR_CST_INFO_T, 34	maxNoOfMidCatPublishers, 46
TRDP_COM_PARAM_T, 34	maxNoOfMidCatSubscriptions, 46
TRDP_COMID_DSID_MAP_T, 35	trdp_if_light.h, 203
TRDP_COMID_ERR	tlc_closeSession, 207
trdp_types.h, 261	tlc_configSession, 208
TRDP_CONFIRMTO_ERR	tlc_getETBTopoCount, 208
trdp_types.h, 261	tlc_getInterval, 209
TRDP_CONSIST_INFO_T, 35	tlc_getJoinStatistics, 209
cstld, 37	tlc_getOpTrainTopoCount, 210
cstOwner, 37	tlc_getOwnlpAddress, 210
TRDP_CRC_ERR	tlc_getPubStatistics, 210
trdp_types.h, 261	tlc_getRedStatistics, 211
TRDP_DATA_TYPE_T	tlc_getStatistics, 211
trdp_types.h, 259	tlc_getSubsStatistics, 212
TRDP DATASET, 37	tlc_getVersion, 212
TRDP DATASET ELEMENT T, 38	tlc_getVersionString, 213
TRDP_DBG_CONFIG_T, 39	tlc init, 213
TRDP_DBG_DEFAULT	tlc_openSession, 214
tau_xml.h, 163	tlc_presetIndexSession, 214
TRDP DNR OPTS	tlc_process, 215

tlc_reinitSession, 215	trdp_types.h, 257
tlc_resetStatistics, 216	TRDP_MARSHALLING_ERR
tlc_setETBTopoCount, 216	trdp_types.h, 261
tlc_setOpTrainTopoCount, 217	TRDP_MAX_FILE_NAME_LEN
tlc terminate, 217	iec61375-2-3.h, 67
tlc_updateSession, 217	TRDP_MAX_LABEL_LEN
tlm_abortSession, 218	iec61375-2-3.h, 67
tlm_addListener, 218	TRDP_MAX_MD_DATA_SIZE
tlm_confirm, 219	iec61375-2-3.h, 67
tlm delListener, 220	TRDP MAX URI HOST LEN
tlm_getInterval, 220	iec61375-2-3.h, 68
tlm_notify, 221	TRDP_MAX_URI_LEN
tlm_process, 222	iec61375-2-3.h, 68
tlm_readdListener, 222	TRDP_MAX_URI_USER_LEN
tlm_reply, 223	iec61375-2-3.h, 68
tlm_replyQuery, 224	TRDP_MD_CALLBACK_T
tlm_request, 224	trdp_types.h, 257
tlp_get, 225	TRDP_MD_CONFIG_T, 48
tlp_getInterval, 226	TRDP_MD_DEFAULT_REPLY_TIMEOUT
tlp_getRedundant, 227	iec61375-2-3.h, 68
tlp_processReceive, 227	TRDP_MD_INFINITE_TIME
tlp_processSend, 228	iec61375-2-3.h, 68
tlp_publish, 228	TRDP MD INFO T, 49
tlp_put, 229	TRDP_MEM_CONFIG_T, 50
tlp_putImmediate, 230	TRDP_MEM_ERR
tlp_republish, 230	trdp_types.h, 260
tlp_request, 231	TRDP_MIN_PD2_HEADER_SIZE
tlp_resubscribe, 232	trdp_tsn_def.h, 250
tlp_setRedundant, 233	TRDP_MIN_PD_HEADER_SIZE
• —	
tlp_subscribe, 233	iec61375-2-3.h, 68
tlp_unpublish, 235	TRDP_MSG_PD
tlp_unsubscribe, 235	iec61375-2-3.h, 69
TRDP_INIT_ERR	TRDP_MSG_TSN_PD
trdp_types.h, 260	trdp_tsn_def.h, 250
trdp_initStats	TRDP_MUTEX_ERR
trdp_stats.c, 247	trdp_types.h, 260
TRDP_INT16	TRDP_NO_ERR
trdp_types.h, 259	trdp_types.h, 260
TRDP_INT32	TRDP_NOCONN_ERR
trdp_types.h, 259	trdp_types.h, 261
TRDP_INT64	TRDP_NODATA_ERR
trdp types.h, 259	trdp types.h, 260
TRDP INT8	TRDP_NOINIT_ERR
trdp_types.h, 259	trdp types.h, 260
TRDP_INTEGRATION_ERR	TRDP NOLIST ERR
trdp_types.h, 261	trdp_types.h, 261
TRDP INUSE ERR	TRDP NOPUB ERR
	trdp_types.h, 261
trdp_types.h, 261	
TRDP_INVALID	TRDP_NOSESSION_ERR
trdp_types.h, 259	trdp_types.h, 261
TRDP_IO_ERR	TRDP_NOSUB_ERR
trdp_types.h, 260	trdp_types.h, 261
TRDP_IP_ADDR_T	TRDP_PACKET_ERR
trdp_types.h, 257	trdp_types.h, 261
trdp_isValidSession	TRDP_PARAM_ERR
tlc_if.c, 181	trdp_types.h, 260
TRDP_MARSHALL_CONFIG_T, 47	TRDP_PD_CALLBACK_T
TRDP_MARSHALL_T	trdp_types.h, 258
<u>—</u>	

TRDP_PD_CONFIG_T, 51	tlc_getJoinStatistics, 243
TRDP_PD_DEFAULT_QOS	tlc_getPubStatistics, 243
trdp_tsn_def.h, 250	tlc_getRedStatistics, 244
TRDP_PD_INFO_T, 52	tlc_getStatistics, 244
TRDP_PD_UDP_PORT	tlc_getSubsStatistics, 245
iec61375-2-3.h, 69	tlc_resetStatistics, 245
trdp_pdPrepareStats	trdp_initStats, 247
trdp_stats.c, 247	trdp_pdPrepareStats, 247
TRDP_PRINT_DBG_T	trdp UpdateStats, 248
trdp_types.h, 258	TRDP_THREAD_ERR
TRDP_PROCESS_CONFIG_T, 54	trdp types.h, 260
TRDP_PROCESS_DEFAULT_CYCLE_TIME	TRDP_TIME_T
iec61375-2-3.h, 69	trdp_types.h, 258
TRDP_PROP_T, 54	TRDP TIMEDATE32
TRDP_PROTOCOL_VERSION_CHECK_MASK	trdp_types.h, 260
iec61375-2-3.h, 69	TRDP_TIMEDATE48
TRDP_QUEUE_ERR	trdp_types.h, 260
trdp types.h, 260	TRDP_TIMEDATE64
TRDP_QUEUE_FULL_ERR	trdp_types.h, 260
trdp_types.h, 260	TRDP TIMEOUT ERR
TRDP_REAL32	trdp_types.h, 260
trdp_types.h, 260	TRDP_TO_BEHAVIOR_T
TRDP_REAL64	trdp_types.h, 262
trdp_types.h, 260	TRDP_TO_DEFAULT
TRDP_RED_FOLLOWER	trdp_types.h, 262
trdp_types.h, 262	TRDP_TO_KEEP_LAST_VALUE
TRDP_RED_LEADER	trdp_types.h, 262
trdp_types.h, 262	TRDP_TO_SET_TO_ZERO
TRDP_RED_STATE_T	trdp_types.h, 262
trdp_types.h, 261	TRDP_TOPO_ERR
trdp_releaseAccess	trdp_types.h, 261
tlc_if.c, 181	trdp_tsn_def.h, 248
TRDP_REPLY_STATUS_T	TRDP_MIN_PD2_HEADER_SIZE, 250
trdp_types.h, 262	TRDP_MSG_TSN_PD, 250
TRDP_REPLYTO_ERR	TRDP_PD_DEFAULT_QOS, 250
trdp_types.h, 261	TRDP_TYPE_MAX
TRDP_REQCONFIRMTO_ERR	trdp_types.h, 260
trdp_types.h, 261	trdp_types.h, 250
TRDP_SDT_DEFAULT_CMTHR	TRDP_APP_CONFIRMTO_ERR, 261
tau_xml.c, 154	TRDP_APP_REPLYTO_ERR, 261
TRDP_SDT_DEFAULT_LMIMAX	TRDP_APP_TIMEOUT_ERR, 261
tau_xml.c, 154	TRDP_BITSET8, 259
TRDP_SDT_PAR_T, 55	TRDP_BLOCK_ERR, 261
TRDP_SEMA_ERR	TRDP_CHAR8, 259
trdp_types.h, 260	TRDP_COMID_ERR, 261
trdp_serviceRegistry.h, 236	TRDP_CONFIRMTO_ERR, 261
SOA_SAME_SERVICEID, 240	TRDP_CRC_ERR, 261
SRM_SERVICE_READ_REQ_COMID, 241	TRDP_DATA_TYPE_T, 259
SRM_SRVINFO_NOTIFY_COMID, 241	TRDP_ERR_T, 260
TRDP_SESSION_ABORT_ERR	TRDP_FLAGS_DEFAULT, 256
trdp_types.h, 261	TRDP_INIT_ERR, 260
trdp_sessionQueue	TRDP_INT16, 259
tlc_if.c, 181	TRDP_INT32, 259
TRDP_SOCK_ERR	TRDP_INT64, 259
trdp_types.h, 260	TRDP_INT8, 259
TRDP_STATE_ERR	TRDP_INTEGRATION_ERR, 261
trdp_types.h, 261	TRDP_INUSE_ERR, 261
trdp_stats.c, 241	TRDP_INVALID, 259

TRDP_IO_ERR, 260	trdp_types.h, 260
TRDP_IP_ADDR_T, 257	TRDP_UINT64
TRDP MARSHALL T, 257	trdp_types.h, 260
TRDP MARSHALLING ERR, 261	TRDP UINT8
TRDP MD CALLBACK T, 257	trdp_types.h, 259
TRDP MEM ERR, 260	TRDP_UNKNOWN_ERR
TRDP_MUTEX_ERR, 260	trdp_types.h, 261
	TRDP_UNMARSHALL_T
TRDP_NO_ERR, 260	
TRDP_NOCONN_ERR, 261	trdp_types.h, 258
TRDP_NODATA_ERR, 260	TRDP_UNRESOLVED_ERR
TRDP_NOINIT_ERR, 260	trdp_types.h, 261
TRDP_NOLIST_ERR, 261	trdp_UpdateStats
TRDP_NOPUB_ERR, 261	trdp_stats.c, 248
TRDP_NOSESSION_ERR, 261	TRDP_USR_URI_SIZE
TRDP_NOSUB_ERR, 261	iec61375-2-3.h, 69
TRDP_PACKET_ERR, 261	TRDP_UTF16
TRDP_PARAM_ERR, 260	trdp_types.h, 259
TRDP PD CALLBACK T, 258	TRDP_VEHICLE_INFO_T, 56
TRDP PRINT DBG T, 258	vehld, 56
TRDP_QUEUE_ERR, 260	TRDP WIRE ERR
	trdp_types.h, 261
TRDP_QUEUE_FULL_ERR, 260	TRDP_XML_DOC_HANDLE_T, 57
TRDP_REAL32, 260	TRDP_XML_PARSER_ERR
TRDP_REAL64, 260	trdp_types.h, 261
TRDP_RED_FOLLOWER, 262	trnCstNo
TRDP_RED_LEADER, 262	
TRDP_RED_STATE_T, 261	GNU_PACKED, 28
TRDP_REPLY_STATUS_T, 262	trnDirState
TRDP_REPLYTO_ERR, 261	GNU_PACKED, 28
TRDP_REQCONFIRMTO_ERR, 261	trnld
TRDP_SEMA_ERR, 260	GNU_PACKED, 28
TRDP SESSION ABORT ERR, 261	trnNetDir
TRDP SOCK ERR, 260	GNU_PACKED, 29
TRDP STATE ERR, 261	trnOperator
TRDP THREAD ERR, 260	GNU_PACKED, 29
TRDP TIME T, 258	trnTopoCnt
TRDP_TIMEDATE32, 260	GNU_PACKED, 29
TRDP_TIMEDATE48, 260	trnVehNo
	GNU_PACKED, 29
TRDP_TIMEDATE64, 260	TTDB_NET_DIR_REQ_COMID
TRDP_TIMEOUT_ERR, 260	iec61375-2-3.h, 70
TRDP_TO_BEHAVIOR_T, 262	TTDB OP DIR INFO COMID
TRDP_TO_DEFAULT, 262	iec61375-2-3.h, 70
TRDP_TO_KEEP_LAST_VALUE, 262	TTDB_STAT_CST_REQ_COMID
TRDP_TO_SET_TO_ZERO, 262	iec61375-2-3.h, 70
TRDP_TOPO_ERR, 261	TTDB TRN DIR REQ COMID
TRDP_TYPE_MAX, 260	
TRDP_UINT16, 259	iec61375-2-3.h, 70
TRDP UINT32, 260	TTI_CACHED_CONSISTS
TRDP_UINT64, 260	tau_tti.c, 127
TRDP_UINT8, 259	vehld
TRDP UNKNOWN ERR, 261	
TRDP UNMARSHALL T, 258	GNU_PACKED, 29
TRDP_UNRESOLVED_ERR, 261	TRDP_VEHICLE_INFO_T, 56
	vehOrient
TRDP_UTF16, 259	GNU_PACKED, 29
TRDP_WIRE_ERR, 261	version
TRDP_XML_PARSER_ERR, 261	GNU_PACKED, 30
TRDP_UINT16	vos_addTime
trdp_types.h, 259	vos_thread.h, 304
TRDP_UINT32	VOS_BLOCK_ERR

ves tures la 040	inDatted
vos_types.h, 318	vos_ipDotted
vos_bsearch	vos_sock.h, 290
vos_mem.c, 264	vos_isMulticast
vos_mem.h, 275	vos_sock.h, 290
vos_clearTime	VOS_LOG_DBG
vos_thread.h, 304	vos_types.h, 318
vos_cmpTime	VOS_LOG_ERROR
vos_thread.h, 305	vos_types.h, 318
vos_crc32	VOS_LOG_INFO
vos_utils.c, 320	vos_types.h, 318
vos_utils.h, 326	VOS_LOG_T
vos_determineBindAddr	vos_types.h, 318
vos_sock.h, 287	VOS_LOG_USR
vos_divTime	vos_types.h, 318
vos_thread.h, 305	VOS_LOG_WARNING
vos_dottedIP	vos_types.h, 318
vos_sock.h, 288	VOS_MAX_ERR_STR_SIZE
VOS_ERR_T	vos_utils.h, 325
vos_types.h, 317	VOS_MAX_FRMT_SIZE
vos getErrorString	vos_utils.h, 325
vos utils.c, 320	VOS MAX PRNT STR SIZE
vos_utils.h, 327	vos_utils.h, 325
vos getInterfaces	VOS MAX SOCKET CNT
vos_sock.h, 288	vos_sock.h, 287
vos getRealTime	vos_mem.c, 262
vos_thread.h, 305	vos_bsearch, 264
vos getTime	vos memAlloc, 265
vos_thread.h, 306	vos memCount, 265
vos_getTimeStamp	vos_memDelete, 266
vos_thread.h, 306	
	vos_memFree, 266
vos_getUuid	vos_memInit, 266
vos_thread.h, 306	vos_qsort, 267
vos_getVersion	vos_queueCreate, 267
vos_utils.c, 321	vos_queueDestroy, 268
vos_utils.h, 327	vos_queueReceive, 268
vos_getVersionString	vos_queueSend, 269
vos_utils.c, 321	vos_strncat, 269
vos_utils.h, 327	vos_strncpy, 271
vos_hostIsBigEndian	vos_strnicmp, 271
vos_utils.c, 321	vos_mem.h, 272
vos_utils.h, 328	vos_bsearch, 275
vos_htonl	VOS_MEM_MAX_PREALLOCATE, 274
vos_sock.h, 289	VOS_MEM_PREALLOCATE, 274
vos_htonll	vos_memAlloc, 275
vos_sock.h, 289	vos_memCount, 275
vos_htons	vos_memDelete, 276
vos_sock.h, 289	vos_memFree, 277
vos_init	vos_memInit, 277
vos_utils.c, 322	vos_qsort, 278
vos_utils.h, 328	vos_queueCreate, 278
VOS_INIT_ERR	vos_queueDestroy, 279
vos_types.h, 318	vos_queueReceive, 279
VOS_INTEGRATION_ERR	vos_queueSend, 280
vos_types.h, 318	vos_strncat, 280
VOS INUSE ERR	vos_strncpy, 281
vos_types.h, 318	vos_strnicmp, 281
VOS IO ERR	VOS MEM ERR
vos_types.h, 318	vos_types.h, 318

VOS_MEM_MAX_PREALLOCATE	VOS_QUEUE_FULL_ERR
vos_mem.h, 274	vos_types.h, 318
VOS_MEM_PREALLOCATE	vos_queueCreate
vos_mem.h, 274	vos_mem.c, 267
vos memAlloc	vos mem.h, 278
vos_mem.c, 265	vos_queueDestroy
vos_mem.h, 275	vos_mem.c, 268
vos_memCount	vos_mem.h, 279
vos_mem.c, 265	vos_queueReceive
vos_mem.h, 275	vos mem.c, 268
vos_memDelete	vos mem.h, 279
vos_mem.c, 266	vos_queueSend
	— ·
vos_mem.h, 276	vos_mem.c, 269
vos_memFree	vos_mem.h, 280
vos_mem.c, 266	vos_sc32
vos_mem.h, 277	vos_utils.c, 322
vos_memInit	vos_utils.h, 329
vos_mem.c, 266	vos_select
vos_mem.h, 277	vos_sock.h, 292
vos_mulTime	VOS_SEMA_ERR
vos_thread.h, 307	vos_types.h, 318
VOS_MUTEX_ERR	vos_semaCreate
vos_types.h, 318	vos_thread.h, 309
vos_mutexCreate	vos_semaDelete
vos_thread.h, 307	vos_thread.h, 309
vos_mutexDelete	vos_semaGive
vos_thread.h, 307	vos_thread.h, 310
vos_mutexLock	vos_semaTake
vos_thread.h, 308	vos_thread.h, 310
vos_mutexTryLock	vos_shared_mem.h, 282
vos thread.h, 308	vos sharedClose, 283
vos_mutexUnlock	vos sharedOpen, 283
vos_thread.h, 309	vos_sharedClose
vos_netIfUp	vos_shared_mem.h, 283
vos_sock.h, 290	vos_sharedOpen
VOS_NO_ERR	vos_shared_mem.h, 283
vos_types.h, 317	vos_sock.h, 284
VOS_NOCONN_ERR	vos_determineBindAddr, 287
vos_types.h, 318	vos_dottedIP, 288
VOS NODATA ERR	vos_getInterfaces, 288
vos types.h, 318	vos htonl, 289
VOS NOINIT ERR	
	vos_htonal, 289
vos_types.h, 318	vos_htons, 289
vos_ntohl	vos_ipDotted, 290
vos_sock.h, 291	vos_isMulticast, 290
vos_ntohll	VOS_MAX_SOCKET_CNT, 287
vos_sock.h, 291	vos_netIfUp, 290
vos_ntohs	vos_ntohl, 291
vos_sock.h, 292	vos_ntohll, 291
VOS_PARAM_ERR	vos_ntohs, 292
vos_types.h, 318	vos_select, 292
VOS_PRINT_DBG_T	vos_sockAccept, 292
vos_types.h, 317	vos_sockBind, 293
vos_qsort	vos_sockClose, 293
vos_mem.c, 267	vos_sockConnect, 294
vos_mem.h, 278	vos_sockGetMAC, 294
VOS_QUEUE_ERR	vos_sockInit, 295
vos_types.h, 318	vos_sockJoinMC, 295

vos_sockLeaveMC, 295	vos_mem.c, 271
vos_sockListen, 296	vos_mem.h, 281
vos_sockOpenTCP, 296	vos_subTime
vos_sockOpenUDP, 297	vos_thread.h, 310
vos_sockReceiveTCP, 297	vos_terminate
vos_sockReceiveUDP, 298	vos_utils.c, 323
vos_sockSendTCP, 299	vos_utils.h, 329
vos_sockSendUDP, 299	vos_thread.h, 301
vos_sockSetMulticastIf, 300	vos_addTime, 304
vos_sockSetOptions, 300	vos_clearTime, 304
vos_sockTerm, 301	vos_cmpTime, 305
VOS_TTL_MULTICAST, 287	vos_divTime, 305
VOS_SOCK_ERR	vos_getRealTime, 305
vos_types.h, 318	vos_getTime, 306
VOS_SOCK_OPT_T, 57	vos_getTimeStamp, 306
vos_sockAccept	vos_getUuid, 306
vos_sock.h, 292	vos_mulTime, 307
vos_sockBind	vos_mutexCreate, 307
vos_sock.h, 293	vos_mutexDelete, 307
vos_sockClose	vos_mutexLock, 308
vos_sock.h, 293	vos_mutexTryLock, 308
vos_sockConnect	vos_mutexUnlock, 309
vos_sock.h, 294	vos_semaCreate, 309
vos_sockGetMAC	vos_semaDelete, 309
vos_sock.h, 294	vos_semaGive, 310
vos_socklnit	vos_semaTake, 310
vos_sock.h, 295	vos_subTime, 310
vos_sockJoinMC	vos_threadCreate, 311
vos_sock.h, 295	vos_threadCreateSync, 311
vos_sockLeaveMC	vos_threadDelay, 312
vos_sock.h, 295	vos_threadInit, 313
vos_sockListen	vos_threadSolf_313
vos_sock.h, 296 vos sockOpenTCP	vos_threadSelf, 313 vos_threadTerm, 314
	-
vos_sock.h, 296 vos_sockOpenUDP	vos_threadTerminate, 314 VOS_THREAD_ERR
vos sock.h, 297	
vos_sock.n, 297 vos sockReceiveTCP	vos_types.h, 318 vos_threadCreate
vos sock.h, 297	vos_thread.h, 311
vos sockReceiveUDP	vos_threadCreateSync
vos sock.h, 298	vos_thread.h, 311
vos sockSendTCP	vos_threadDelay
vos_sock.h, 299	vos thread.h, 312
vos sockSendUDP	vos threadInit
vos sock.h, 299	vos thread.h, 313
vos sockSetMulticastIf	vos threadIsActive
vos sock.h, 300	vos thread.h, 313
vos sockSetOptions	vos_threadSelf
vos_sock.h, 300	vos thread.h, 313
vos sockTerm	vos threadTerm
vos_sock.h, 301	vos thread.h, 314
vos strncat	vos_threadTerminate
vos_mem.c, 269	vos thread.h, 314
vos mem.h, 280	VOS TIMEOUT ERR
vos_strncpy	vos_types.h, 318
vos_mem.c, 271	VOS TIMEVAL T
vos mem.h, 281	vos_types.h, 317
vos_strnicmp	VOS_TTL_MULTICAST
100_0tt1110111p	VOO_TIL_WIGHTIOAGT

```
vos_sock.h, 287
vos_types.h, 314
    VOS_BLOCK_ERR, 318
    VOS_ERR_T, 317
    VOS_INIT_ERR, 318
    VOS INTEGRATION ERR, 318
    VOS_INUSE_ERR, 318
    VOS_IO_ERR, 318
    VOS LOG DBG, 318
    VOS_LOG_ERROR, 318
    VOS_LOG_INFO, 318
    VOS_LOG_T, 318
    VOS_LOG_USR, 318
    VOS_LOG_WARNING, 318
    VOS_MEM_ERR, 318
    VOS_MUTEX_ERR, 318
    VOS NO ERR, 317
    VOS NOCONN ERR, 318
    VOS_NODATA_ERR, 318
    VOS_NOINIT_ERR, 318
    VOS PARAM ERR, 318
    VOS_PRINT_DBG_T, 317
    VOS_QUEUE_ERR, 318
    VOS_QUEUE_FULL_ERR, 318
    VOS_SEMA_ERR, 318
    VOS_SOCK_ERR, 318
    VOS_THREAD_ERR, 318
    VOS TIMEOUT ERR, 318
    VOS TIMEVAL T, 317
    VOS UNKNOWN ERR, 318
VOS_UNKNOWN_ERR
    vos_types.h, 318
vos_utils.c, 319
    vos_crc32, 320
    vos_getErrorString, 320
    vos_getVersion, 321
    vos_getVersionString, 321
    vos_hostIsBigEndian, 321
    vos init, 322
    vos sc32, 322
    vos terminate, 323
vos_utils.h, 323
    INITFCS, 325
    vos crc32, 326
    vos_getErrorString, 327
    vos_getVersion, 327
    vos_getVersionString, 327
    vos_hostIsBigEndian, 328
    vos init, 328
    VOS_MAX_ERR_STR_SIZE, 325
    VOS_MAX_FRMT_SIZE, 325
    VOS MAX PRNT STR SIZE, 325
    vos_sc32, 329
    vos_terminate, 329
```

VOS_VERSION_T, 58