ISO 22900-2:2022-06 (E)

Road vehicles - Modular vehicle communication interface (MVCI) - Part 2: Diagnostic protocol data unit (D-PDU API)

Contents

Page

Foreword vi

Introduction vii

1 Scope 1

2 Normative references 1

3 Terms, definitions and abbreviated terms 1
3.1 Terms and definitions 2
3.2 Abbreviated terms 3

4 Specification release version information 6
4.1 Specification release version location 6
4.2 Specification release version 6

5 Modular VCI use cases 6
5.1 OEM merger 6
5.2 OEM cross vehicle platform ECU(s) 6
5.3 Central source diagnostic data and exchange during ECU development 7
5.4 OEM franchised dealer and aftermarket service outlet diagnostic tool support 7

6 Modular VCI software architecture 7
6.1 Overview 7
6.2 Modular VCI D-Server software 8
6.3 Runtime format based on ODX 9
6.4 MVCI protocol module software 9
6.5 MVCI protocol module configurations 9

7 D-PDU API use cases 10
7.1 Overview 10
7.2 Use case 1: Single MVCI protocol module 11
7.3 Use case 2: Multiple MVCI protocol modules supported by same D-PDU API implementation 12
7.4 Use case 3: Multiple MVCI protocol modules supported by different D-PDU API implementations 13

8 Diagnostic protocol data unit (D-PDU) API 14
8.1 Software requirements 14
8.1.1 General requirements 14
8.1.2 Vehicle protocol requirements 15
8.1.3 Timing requirements for protocol handler messages 16
8.1.4 Serialization requirements for protocol handler messages 17
8.1.5 Compatibility requirements 19
8.1.6 Timestamp requirements 19
8.2 API function overview and communication principles 20
8.2.1 Terms used within the D-PDU API 20
8.2.2 Function overview 20
8.2.3 General usage 21
8.2.4 Asynchronous and synchronous communication 24
8.2.5 Usage of resource locking and resource unlocking 25

8.2.6 Usage of ComPrimitives 25
8.3 Tool integration 42
8.3.1 Requirement for generic configuration 42
8.3.2 Tool integrator -- Use case 42
8.4 API functions -- Interface description 44
8.4.1 Overview 44
8.4.2 PDUConstruct 44
8.4.3 PDUDestruct 45
8.4.4 PDUIoCtl 46
8.4.5 PDUGetVersion 48
8.4.6 PDUGetStatus 49
8.4.7 PDUGetLastError 50
8.4.8 PDUGetResourceStatus 51
8.4.9 PDUCreateComLogicalLink 52
8.4.10 PDUDestroyComLogicalLink 55
8.4.11 PDUConnect 57
8.4.12 PDUDisconnect 59
8.4.13 PDULockResource 60
8.4.14 PDUUnlockResource 61
8.4.15 PDUGetComParam 62
8.4.16 PDUSetComParam 69
8.4.17 PDUStartComPrimitive 72
8.4.18 PDUCancelComPrimitive 76
8.4.19 PDUGetEventItem 77
8.4.20 PDUDestroyItem 78
8.4.21 PDURegisterEventCallback 79
8.4.22 EventCallback prototype 81
8.4.23 PDUGetObjectId 83
8.4.24 PDUGetModuleIds 84
8.4.25 PDUGetResourceIds 86
8.4.26 PDUGetConflictingResources 87
8.4.27 PDUGetUniqueRespIdTable 88
8.4.28 PDUSetUniqueRespIdTable 90
8.4.29 PDUModuleConnect 95
8.4.30 PDUModuleDisconnect 98
8.4.31 PDUGetTimestamp 99
8.5 I/O control section 99
8.5.1 IOCTL API command overview 99
8.5.2 PDU_IOCTL_RESET 102
8.5.3 PDU_IOCTL_CLEAR_TX_QUEUE 102
8.5.4 PDU_IOCTL_SUSPEND_TX_QUEUE 103
8.5.5 PDU_IOCTL_RESUME_TX_QUEUE 103
8.5.6 PDU_IOCTL_CLEAR_RX_QUEUE 104
8.5.7 PDU_IOCTL_CLEAR_TX_QUEUE_PENDING 104
8.5.8 PDU_IOCTL_READ_VBATT 105
8.5.9 PDU_IOCTL_SET_PROG_VOLTAGE 105
8.5.10 PDU_IOCTL_READ_PROG_VOLTAGE 106
8.5.11 PDU_IOCTL_GENERIC 107
8.5.12 PDU_IOCTL_SET_BUFFER_SIZE 108
8.5.13 PDU_IOCTL_GET_CABLE_ID 108
8.5.14 PDU_IOCTL_START_MSG_FILTER 109
8.5.15 PDU_IOCTL_STOP_MSG_FILTER 111
8.5.16 PDU_IOCTL_CLEAR_MSG_FILTER 111
8.5.17 PDU_IOCTL_SET_EVENT_QUEUE_PROPERTIES 112
8.5.18 PDU_IOCTL_SEND_BREAK 113
8.5.19 PDU_IOCTL_READ_IGNITION_SENSE_STATE 113
8.5.20 PDU_IOCTL_VEHICLE_ID_REQUEST 114
8.5.21 PDU_IOCTL_SET_ETH_SWITCH_STATE 117
8.5.22 PDU_IOCTL_GET_ENTITY_STATUS 118
8.5.23 PDU_IOCTL_GET_DIAGNOSTIC_POWER_MODE 119

8.5.24 PDU_IOCTL_GET_ETH_PIN_OPTION 120
8.5.25 PDU_IOCTL_TLS_SET_CERTIFICATE 121
8.5.26 PDU_IOCTL_DOIP_GET_CURRENT_SESSION_MODE 122
8.5.27 PDU_IOCTL_ISOBUS_GET_DETECTED_CFS 123
8.6 API functions -- Error handling 123
8.6.1 Synchronous error handling 123
8.6.2 Asynchronous error handling 123
8.7 Installation 124
8.7.1 Generic description 124
8.7.2 Windows installation process 124
8.7.3 Linux installation process 125
8.7.4 Selecting MVCI protocol modules 126
8.8 Application notes 126
8.8.1 Interaction with the MDF 126
8.8.2 Accessing additional hardware features for MVCI protocol modules 126
8.8.3 Documentation and information provided by MVCI protocol module vendors 126

9 Using the D-PDU API with existing applications 127
9.1 SAE J2534-1 and RP1210a existing standards 127

10 Data structures 128
10.1 API functions -- Data structure definitions 128
10.1.1 Abstract basic data types 128
10.1.2 Definitions 129
10.1.3 Bit encoding for UNUM32 129
10.1.4 API data structures 129

Annex A (normative) D-PDU API compatibility mappings 145

Annex B (normative) D-PDU API standard ComParams and protocols 146

Annex C (informative) D-PDU API manufacturer-specific ComParams and protocols 235

Annex D (informative) D-PDU API constants 237

Annex E (informative) Application defined tags 253

Annex F (informative) RDF and MDF description files 254

Annex G (informative) Resource handling scenarios 325

Annex H (informative) D-PDU API partitioning 331

Annex I (informative) Use case scenarios 335

Annex J (informative) OBD protocol initialization 376

Annex K (normative) DoIP implementation 392

Annex L (normative) ISOBUS 425

Bibliography 433