ERRATA

Section or Other	Date	Description of Problem	Problem Resolution
A.2	08/01/2003	Current requirements do not allow compatibility with future direction of ISO 15765-2. Alternative method of network layer error handling would allow this forward compatibility.	Add new section 2.3.2.2 titled "Alternative Network Layer Error Handling" with the following text: As an alternative to the response handling described within this specification, the ECU diagnostic application may choose to not generate any response message when one of the following network layer errors occur: wrong sequence number in consecutive frame, invalid flow status in flow control, or N_Cr timeout exceeded. Furthermore, if the network layer receives either a single frame request with a single frame data length greater than 7 or a first frame request with a first frame data length less than 8, the ECU network layer may choose to ignore the invalidly formatted frame with no indication to the ECU diagnostic application.

Ford Motor Company

CAN

(Controller Area Network)

Generic Diagnostic Specification

Worldwide Requirements

Version 2003.0
Date Issued: 25 April 2003

Research & Vehicle Technology Office
Electrical/Electronic Systems Engineering
Core Network Communications

REVISION HISTORY

Previous Version	Current Version	Version Description	Author	Date
N/A	v2000.0	Initial Release	S. DiLodovico	21- July - 2000
v2000.0	v2000.1	Clarification and minor alterations.	S. DiLodovico	27-July – 2000
v2000.1	v2000.2	Changed and clarified Response code $78 handling	S. DiLodovico	28-July – 2000
v2000.2	v2001.0	Clarification and additions to diagnostic services and references.	S. DiLodovico / J . Miller	09-February-2001
v2001.0	v2001.1	Inclusion of Errata sheet details into the specification and clarification of issues.	J. Miller	30-November-2001
v2001.1	v2003.0	Inclusion of Errata sheet details into the specification and clarification of issues.	J. Miller	25-April-2003

Change Log

Release	Previous Section	Current Section	Change Description
v2003.0	All	All	Inclusion of all Errata sheet details from the CAN Generic Diagnostic Specification Errata Sheet dated 07/01/2002.
v2003.0	All	All	Replaced all uses of "mode" with "service" when referring to a specific diagnostic service such as $22 (readDataByCommonID).
v2003.0	All	All	Updated all references to v2001.1 of the core diagnostic documents to v2003.0 of the core diagnostic documents.
v2003.0	1.2	1.2	Added clarification in regards to the deviation process
v2003.0	1.3, Table 1	1.3, Table 1	Added reference to Communication "Linked Based" Service Diagnostic Requirements
v2003.0	1.3, Table 1	1.3, Table 1	Added reference to CAN Generic Diagnostic Specification Assembly Plant Powertrain Configuration and Testing Supplement
v2003.0	1.5	1.5	Added acronyms and descriptions for DSP, FCSD, FNOS, SSDS, and VO.
v2003.0	2	2	Clarified requirement regarding use of test procedure specification.
v2003.0	Table 3	Table 3	Removed reference to subsections 5.8.1 and 5.8.2 in Item [7].
v2003.0	Table 3	Table 3	Removed reference to subsection 5.3.1.3 in Item [17].
v2003.0	2.2.2.6	2.2.2.6	Added reference to required timing for initial NRC $78 response for clarification.
v2003.0	2.2.2.7.1	2.2.2.7.1	Clarified the suspension of active diagnostic functionality when a transition is made to a different diagnostic session that does not support this diagnostic functionality.
v2003.0	2.2.2.7.6.1	2.2.2.7.6.1	Clarified requirements for when a security seed shall remain valid or active.
v2003.0	2.2.2.7.8	2.2.2.7.8	Clarified that DTCs are not required to be reported in the order they are detected
v2003.0	2.2.2.7.8	2.2.2.7.8	Added requirement that a complete description of the logic used by the module to set each DTC shall be documented in the ECU's SSDS.
v2003.0	2.2.2.7.9	2.2.2.7.9	Allowed Freeze Frame number data to be generically defined by embedded PID references.
v2003.0	Table 13	Table 13	Clarified meaning of routineStatusByte values
v2003.0	Table 29	Table 29	Added state encoded value to PID $E6F3 to represent v2003.0 of this spec
v2003.0	2.2.3.1.1	2.2.3.1.1	Clarified that rapid data responses are not disabled via CommunicationControl
v2003.0	2.2.3.1.2	2.2.3.1.2	Clarified that ControlDTCSetting is used to disable the logging of continuous DTCs.
v2003.0	2.2.3.2.1	2.2.3.2.1	Modified missing message requirements to utilize SAE defined "Lost Communication" DTCs ($C100-C2FF) in place of DTC $D900.
v2003.0	2.2.3.7	2.2.3.7	Added additional option for ECU behavior if a second tester request is received prior to ECU responding to the initial request.
v2003.0	2.2.3.8	2.2.3.8	Added allowance for an ECU to perform a reset upon expiration of the Lack of Diagnostic Dialog Timer if services $2F, $31, or $B1 were executed in the default diagnostic session ($81).
v2003.0	2.2.3.8	2.2.3.8	Added clarification for the use of vehicle speed to suspend "timed" diagnostic functionality.
v2003.0	N/A	2.2.3.13	Added clarification that any special procedures shall be documented in the SSDS.
v2003.0	N/A	2.2.3.13	Added requirement for explicit approval in order for support of any special or unique tester involvement necessary to perform certain diagnostic functionality
v2003.0	N/A	2.2.3.14	Added requirements for powertrain modules to require diagnostic tool control of outputs and to provide parameters for monitoring both inputs and outputs.
v2003.0	N/A	2.2.3.15	Added requirements for FNOS related PIDs.
v2003.0	N/A	2.2.3.16	Added requirement for gateway ECUs to support diagnostics on both networks.
v2003.0	2.3.2.2.1	2.3.2.2.1	Changed section title from "Performance Parameter Definitions" to "Performance Parameter Definitions (ref ISO 15765-2, 7.6)" to illustrate

Release	Previous Section	Current Section	Change Description
			where the parameters are defined.
v2003.0	Table 45	Table 46	Changed N_Bs timeout value from 75ms to a range of 75ms – 120ms
v2003.0	Table 45	Table 47	Changed N_Cr timeout value from 75ms to a range of 75ms – 120ms
v2003.0	A.2	A.2	Clarification on usage of NRCs $21 and $78
v2003.0	A.2.17	A.2.17	Added an additional reason of incorrect dataFormatIdentifier to negative response code $31 for service $34 (requestDownload)
v2003.0	A.2.18	A.2.18	Added an additional reason of incorrect dataFormatIdentifier to negative response code $31 for service $35 (requestUpload)
v2003.0	A.2.21	A.2.21	Added an additional negative response code of $10 (generalReject) to service $3B (writeDataByLocalID).
v2003.0	A.2.25	A.2.25	Added an additional reason of requested data rate does not match active data rate to negative response code $22 (conditionsNotCorrect) for service $A0 (requestDiagnosticDataPacket).
v2003.0	A.2.27	A.2.27	Added an additional reason of all prerequisite conditions are not satisfied to negative response code $22 (conditionsNotCorrect) for service $B1 (diagnosticCommand).

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

1 Introduction

1.1 Purpose / Scope

The purpose of this document is to define a methodology and specify requirements for performing enhanced link based diagnostics on the Controller Area Network (CAN) protocol. It references requirements from the Ford Global Diagnostic Specification (Part One), "Keyword Protocol 2000 Implementation of Diagnostic Services Recommended Practice" (German origin), and ISO 15765-2 specifications.

This document is intended to provide guidance and requirements to architects, system designers, application developers, and suppliers that are to deliver Controller Area Network (CAN) compliant diagnostics to Ford production vehicle programs. It shall serve as the Cross Car Line compliance reference for implementation of CAN diagnostics in all Ford, Think!, and Mazda vehicles. Adherence to the specifications contained within this document is required to ensure compatibility within the corporate product line from vehicle to vehicle, model year to model year and maintain compatibility with the industry driven standards. Most CAN diagnostic protocol conformance requirements shall be referenced from the industry standard recommended practice: "Keyword Protocol 2000 Implementation of Diagnostic Services Recommended Practice" (German origin) and the ISO 15765-2 standard. Only the requirement sections that are specifically referenced from these standards by this document shall be applicable. In the case of conflict, this document shall override the Ford Global Diagnostic Specification (Part One), "Keyword Protocol 2000 Implementation of Diagnostic Services Recommended Practice" (German origin), and the ISO 15765-2 documents.

Note: Ford reserves the right to examine all related hardware and software designs related to this document for validation and verification of compliance.

1.2 Deviations

Any non-compliance to this specification, for any ECU, is a deviation and shall follow the appropriate process for resolution. Any deviation shall be fully documented and adequately explained in the Subsystem Specific Diagnostic Specification (Part 2) for the ECU. In addition to the normal deviation process, any deviations must be approved by FCSD / DSP. Note that failure to comply with the requirements outlined in this specification may result in lack of diagnostic availability in time to sell/ship vehicles. A Subsystem Specific Diagnostic Specification, commonly referred to as SSDS – Part 2, shall be generated to document the diagnostic message requirements and capabilities for each ECU.

1.3 Use of this Document

The CAN network is utilized to transfer information from an ECU to an off-board Tester and vice-versa. The information transferred consists of module fault information, parameter reporting, test commands, calibration values and many other types of data used either in performing module diagnostics, re-programming or any other data transfer not associated with normal mode communications.

This document is to be used by module developers and any other entity utilizing the diagnostic capabilities specified or referenced in this document.

This document is the controlling specification for implementing diagnostics on CAN. All diagnostic requirements shall be either specifically stated in this specification or specifically referenced as a requirement from another specification. Only the specific sections of the Ford Global Diagnostic Specification (Part One), "Keyword Protocol 2000 Implementation of Diagnostic Services Recommended Practice" (German origin), and ISO 15765-2 specifications that are referenced from this document are applicable.

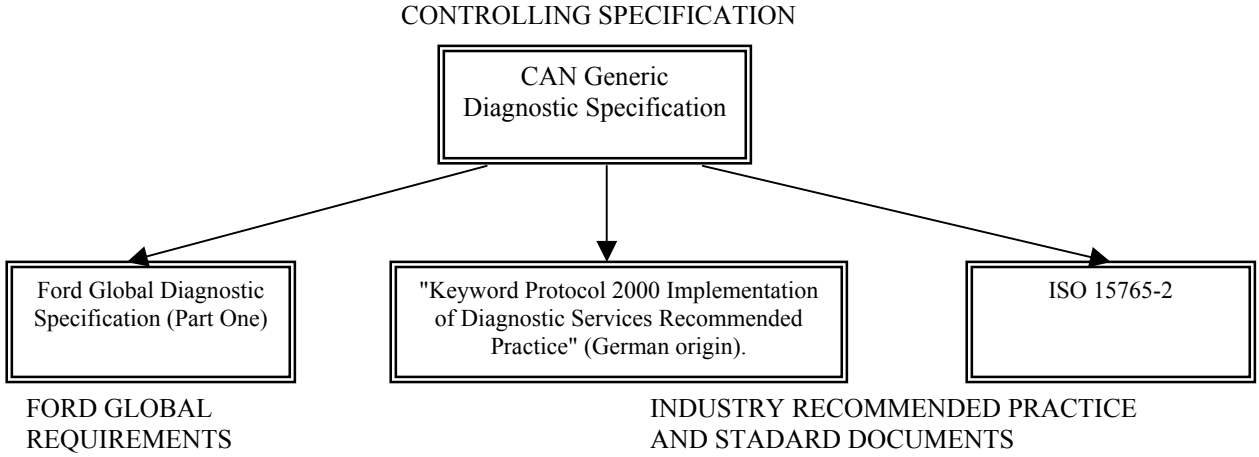


Figure 1: CAN Diagnostic Specification Relationships

This specification addresses diagnostic strategies relevant to the CAN diagnostic interface. It does not address normal or fault modes relating to the ECU to ECU communications over CAN. The "HS/MS-CAN ECU and Subsystem Requirements Specification" and "HS/MS-CAN System Level and Distributed Applications Requirements Specification" address implementation of CAN networking for normal and fault mode communications. FCSD / DSP additional requirements are contained within the Communication "Linked Based" Service Diagnostic Requirements specification (see Table 1). Additional powertrain requirements are contained within the CAN Generic Diagnostic Specification Assembly Plant Powertrain Configuration and Testing Supplement (see Table 1).

Though an attempt was made to cover all aspects of implementing CAN networks for Ford there may still be design attributes that may lack detail or are overlooked. As these and other attributes are further studied requirements will be added or modified to reflect the latest industry and corporate directions and trends.

1.4 Applicable Documents

The following documents are either referenced by this specification, or contain information that is relevant to this specification.

Table 1: Applicable Documents Table

Source	Title	Document Reference No.
Ford	CAN Generic Diagnostic Test Procedure Specification	CDTP-v2003.0
Ford	Global Diagnostic Specification (Part 1) v2003.0	P1-v2003.0
Ford	Module Programming and Configuration Specification v2003.0	MC-v2003.0
Ford	HS/MS-CAN ECU and Subsystem Requirements Specification	
Ford	HS/MS-CAN System Level and Distributed Applications Requirements Specification	
SAE	Glossary of Vehicle Networks for Multiplex and Data Communication	J1213
SAE	Electrical/Electronic Systems Diagnostic Terms, Definitions, Abbreviations, and Acronyms	J1930
SAE	Diagnostic Connector	J1962
SAE	Diagnostic Trouble Code Definitions	J2012
SAE	Enhanced E/E Diagnostic Test Modes	J2190
SAE	E/E Data Link Security	J2186
SAE	High Speed CAN (HSC) Implementation For Passenger Vehicle Applications	J2284
ISO	Road Vehicles – Diagnostics on CAN – Part 2: Network layer	15765-2

	services, Date: 11-30-1999	
ISO	Road Vehicles – Communication Between Vehicle and Test Equipment	15031
ISO	Data Processing Systems - Open Systems Interconnection - Standard Reference Model	7498
ISO	Road Vehicles - Interchange of digital information -- Controller Area Network (CAN) for high speed	11898:1993/Amd.1:1995(E)
Europe (Germany)	Keyword Protocol 2000 Implementation of Diagnostic Services Recommended Practice, v1.5, Date: October 1, 1997	KWP-GRP-1.5
Ford	Communication "Linked Based" Service Diagnostic Requirements	
Ford	CAN Generic Diagnostic Specification Assembly Plant Powertrain Configuration and Testing Supplement v2003.0	

Ford requirements documents, such as this one may be accessed on the Ford Intranet at the R&VT/EESE on-line documentation site (https://f1.ford.com/eRoom/EESE/NetCom).

Information for obtaining SAE documents may be found on the external Internet: “http://www.sae.org”.
Information for obtaining ISO documents may be found on the external Internet: “http:// www.iso.ch”.

1.5 Abbreviations / Acronyms

The following abbreviations are used throughout this specification:

CAN:	**C**ontroller **A**rea **N**etwork
DLC:	**D**ata **L**ink **C**onnector (SAE J1962)
DSP:	**D**iagnostic **S**ervice **P**lanning
ECU:	**E**lectronic **C**ontrol **U**nit
E/E:	**E**lectrical and **E**lectronic
EESE:	**E**lectrical/**E**lectronic **S**ystems **E**ngineering
FAO:	**F**ord **A**utomotive **O**perations
FCSD:	**F**ord **C**ustomer **S**ervice **D**ivision
FNOS:	**F**ord **N**etwork **O**perating **S**ystem
GDS:	**G**lobal **D**iagnostic **S**pecification
ID:	**Id**entifier
ISO:	**I**nternational **S**tandards **O**rganization
MRDB:	**M**aster **R**eference **D**ata **B**ase
NGS:	**N**ew **G**eneration **S**tar **T**ester
OSI:	**O**pen **S**ystem **I**nterconnect
PCM:	**P**owertrain **C**ontrol **M**odule
SAE:	**S**ociety of **A**utomotive **E**ngineers
SSDS:	**S**ubsystem **S**pecific **D**iagnostic **S**pecification (Part 2)
ECATS:	**E**nd-Of-Line **C**onfiguration and **T**est **S**ystem
WDS:	**W**orldwide **D**iagnostic **S**ystem
R&VT:	**R**esearch and **V**ehicle **T**echnology
WCR:	**W**orldwide **C**ustomer **R**equirements
OBD:	**O**nboard **D**iagnostics
EOBD:	**E**uropean **O**nboard **D**iagnostics
eCATS:	**E**lectronic **C**onsumer **A**cceptance **T**est **S**ystem
CM:	**C**ontinuous **M**onitor
M:	**M**andatory
O:	User **O**ptional; support is application specific or dependent on dynamic usage
VO:	**V**ehicle **O**perations

2 Diagnostic Protocol Implementation Requirements

Section 2 and its subsections specify design requirements for all Ford Motor Company (FMC), subsidiary, and affiliate production intent implementations of CAN (Controller Area Network) diagnostics. The requirements consist of references, clarifications, and embellishments of existing recommended practice "Keyword Protocol 2000 Implementation of Diagnostic Services Recommended Practice" and ISO 15765-2 plus requirements that are specific to FMC.

The requirements are structured and ordered using the organizational framework of ISO's 7498 7-layer network abstraction model. Figure 2 depicts the framework along with enhancements to help conceptualize the breakdown of requirements and their respective origins. In the left margin and moving from the bottom up the "CAN Standard Behavior" layers specify industry wide requirements, while the "Cross Vehicle Behavior" specifies corporate or FMC wide requirements, and the "Specific Vehicle Behavior" specifies platform specific requirements. In the right margin the figure depicts bulleted sample requirements for greater insight into the type of requirements that are associated with each layer. Between layers detailed generalizations of information interfaces between layers are also set forth. The ISO 15765-2 specifies the Network Layer and assumes direct interfacing to the application layer that is defined in the KWP-GRP-1.5.

As part of the testing to validate conformance to this specification, suppliers are required to provide initial test results from CDTP-v2003.0 and corresponding output trace file(s) to the Netcom Application Engineer prior to CP software freeze. EESE Core Netcom has developed test scripts which automate the test procedures defined in CDTP-v2003.0 (available at https://f1.ford.com/eRoom/EESE/NetCom). Suppliers are encouraged to use them. Questions relating to accessing or running the test scripts should be directed to the appropriate Netcom Application Engineer.

2.1 Document Precedence for the CAN Diagnostic Protocol

In the case of any requirement conflicting between either this document or any of the referenced documents, this document shall be the governing document. All requirements specified in this document shall supersede any requirement specified in any other document. This specification shall be followed and the requirements specified in this document shall not be negotiable.

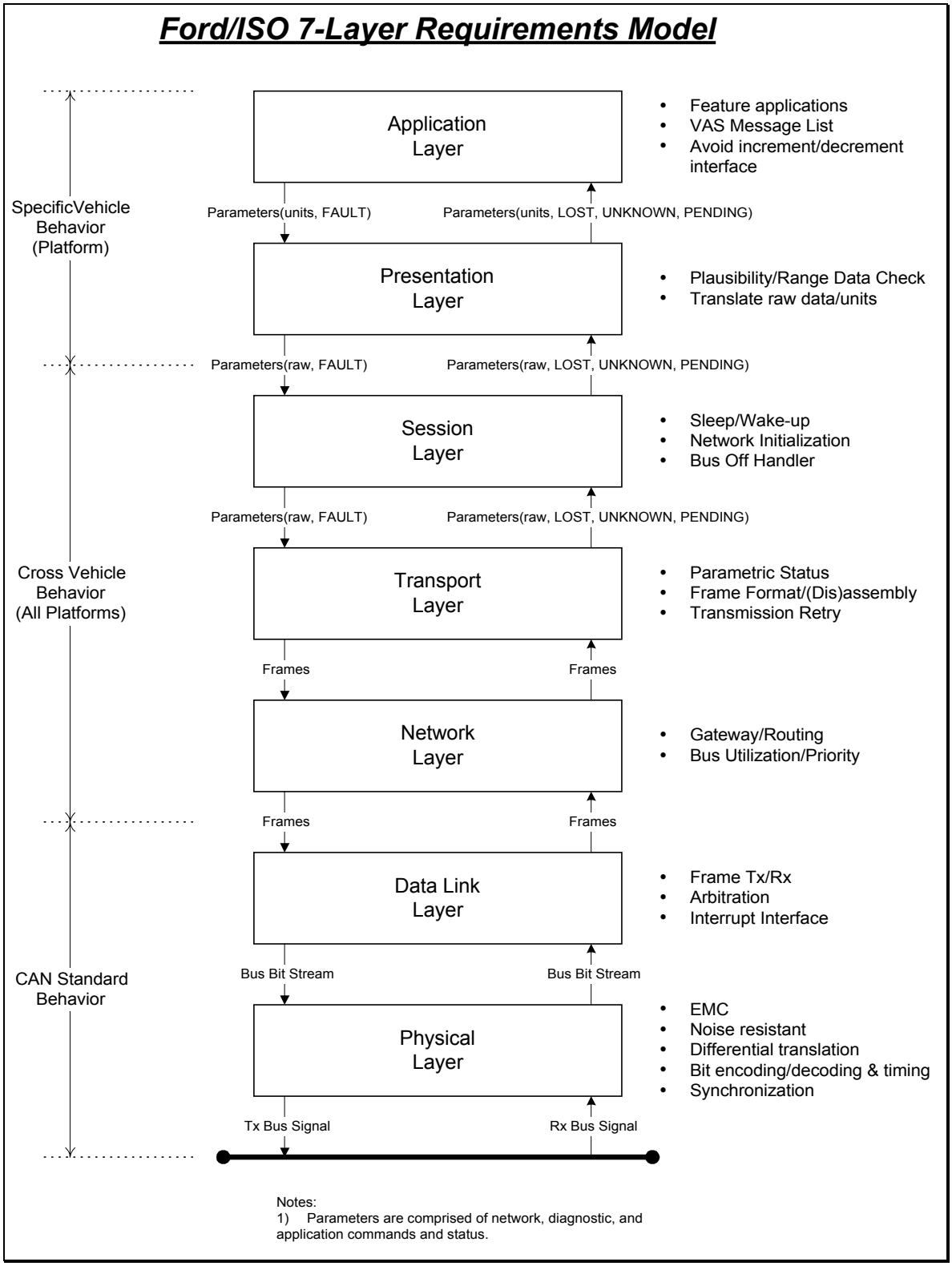


Figure 2: Ford/ISO 7-Layer Requirements Model

2.2 Application Layer

2.2.1 Information References

The following references below are for informational purposes only, relating to the KWP-GRP-1.5 recommended practice.

Table 2: Diagnostic Application Layer Informational References

Item	Document	Section	Subsections
[1]	KWP-GRP-1.5	Introduction	All
[2]	KWP-GRP-1.5	1. Scope	All
[3]	KWP-GRP-1.5	2. Normative reference	All
[4]	KWP-GRP-1.5	5. General implementation rules	5.1
[5]	P1-v2003.0	3. Link-Based Diagnostic System Overview	3.1, 3.2, 3.2.1, 3.2.2, 3.2.3-6

2.2.2 Referenced Requirements

All Ford CAN diagnostic Application Layer implementations shall comply with requirements referenced throughout this section.

Table 3: Diagnostic Application Layer Referenced Requirements

Item	Document	Section	Subsections
[6]	P1-v2003.0	2. Diagnostic Development Process	2.4
[7]	P1-v2003.0	5. General Requirements	5.3, 5.4, 5.5.1, 5.5.2, and 5.7
[8]	P1-v2003.0	6.3. DTC Requirements	All except 6.3.2.3
[9]	P1-v2003.0	9.4. Security Levels	All
[10]	P1-v2003.0	16. Parameter Identifier (PID) and Data Packet Requirements	All except 16.4, 16.5, 16.10.3 and Table 16.4
[11]	P1-v2003.0	10.4.4. On-demand Fault PIDs	All
[12]	P1-v2003.0	10.5.1. On-demand Self-test ($02) – Mandatory	All
[13]	P1-v2003.0	10.5.2. Assembly Self-test ($11) – Optional	All
[14]	P1-v2003.0	15. Diagnostic Command Mode Requirements	15.1, 15.2, 15.3, 15.4 and 15.7
[15]	KWP-GRP-1.5	3. Definitions and abbreviations	All except 3.2.2
[16]	KWP-GRP-1.5	4. Conventions	4.4
[17]	KWP-GRP-1.5	5. General implementation rules	5.3.1.1 and 5.3.1.4.1
[18]	KWP-GRP-1.5	6.1. startDiagnosticSession service	All except 6.1.5.2.2, 6.1.5.2.3 and 6.1.5.2.4
[19]	KWP-GRP-1.5	6.3. securityAccess service	All except 6.3.5
[20]	KWP-GRP-1.5	6.4. testerPresent service	All except 6.4.5.1
[21]	KWP-GRP-1.5	6.5. ECUReset service	6.5.1, 6.5.2, 6.5.3 and 6.5.4
[22]	KWP-GRP-1.5	7.1. readDataByLocalIdentifier service	All except 7.1.5.1, 7.1.5.2, and Table 7.1.1
[23]	KWP-GRP-1.5	7.2. readDataByCommonIdentifier service	All
[24]	KWP-GRP-1.5	7.3. readMemoryByAddress service	All except 7.3.5.1, 7.3.5.2, and Table 7.5.1
[25]	KWP-GRP-1.5	7.5. writeDataByLocalIdentifier service	All
[26]	KWP-GRP-1.5	7.6. writeDataByCommonIdentifier service	All

[27]	KWP-GRP-1.5	7.7. writeMemoryByAddress service	All except 7.7.5.1 and 7.7.5.2
[28]	KWP-GRP-1.5	8.2. readDiagnosticTroubleCodesByStatus service	All except 8.2.1.1.2, Fig. 4, Fig. 5, 8.2.5.2.3
[29]	KWP-GRP-1.5	8.4. readFreezeFrameData service	All except 8.4.5.1
[30]	KWP-GRP-1.5	8.5. clearDiagnosticInformation service	All except 8.5.5
[31]	KWP-GRP-1.5	9.2. inputOutputControlByCommonIdentifier service	All except 9.2.5
[32]	KWP-GRP-1.5	10. Remote Activation of Routine functional unit	All except 10.2, 10.3.5, 10.4, 10.5.5.2 and 10.6
[33]	KWP-GRP-1.5	11.1. requestDownload service	All except 11.1.5
[34]	KWP-GRP-1.5	11.2. requestUpload service	All except 11.2.5
[35]	KWP-GRP-1.5	11.3. transferData service	All except 11.3.4 and 11.3.5
[36]	KWP-GRP-1.5	11.4. requestTransferExit service	All except 11.4.1 and 11.4.5

2.2.2.1 Automatic Clearing of DTCs (Alternative method) (ref. P1-v2003.0, 6.3.2.1)

Continuous DTCs may be optionally cleared using a time based aging counter. The time counter shall start at zero and begin counting upward as soon as the fault that has been validated is not detected by the ECU anymore. The counter shall only be incremented while the module is in operation, as defined in its SSDS. The DTC shall be aged (deleted) after 75 hours of operation has occurred since the last validation of the logged DTC.

If the fault is detected while the counter is being incremented then the counter shall be reset to zero.

2.2.2.2 Fault PID and Fault PID DTC Requirements (ref. P1-v2003.0, 16.9, 10.4.4)

Every fault DTC defined in the MRDB shall have the following prefix added to the description: "Fault DTC:".

Every fault PID defined in the MRDB shall have the following prefix added to the description: "Fault PID (CM): or "Fault PID (OD):".

2.2.2.3 Inapplicable K-Line Protocol References (ref. KWP-GRP-1.5, All Sections)

No K-Line references in the KWP-GRP-1.5 shall be applicable to this specification. All the network and data link layer definitions shall be referenced in Section 2.4 of this document that reference the ISO 15765-2 document referring to CAN protocol elements. Also, all references to the ISO 14230-2 specification are not applicable to any Ford requirements.

2.2.2.4 Header Bytes (ref. KWP-GRP-1.5, All Sections)

Header formats specified in KWP-GRP-1.5 are not applicable to the Ford CAN diagnostic messaging definitions. See Section 2.3.2 for information relating to CAN identifiers and references to the messaging formats for CAN. Included in this omission are all references to format, target, source and length bytes.

2.2.2.5 Request Messages/Services

The modes of operation and the services supported in each of those modes are listed in Table 4.

Table 4: Service Identifier Value Summary Table

Column Number	1	2	3	4	5	6	7	8
Diagnostic Service Name	**Mnemonic (3.3.1)**	**Default / Std Diag / OBD II Mode ($81)**	**ECU Prog Mode ($85)**	**ECU Adj. Mode ($87)**	**Req. Value**	**Pos. Rsp. Value**	**Neg. Rsp. Value**	**Service (Column 5) is Mandatory if any of the listed service(s) are supported (per session)**
SAE J1979 Diag. Test Modes	**J1979M**	■		■	**00-0F**	**40-4F**	**7F**	--
startDiagnosticSession	**STDS**	④	◆	◆	**10**	**50**	**7F**	--
ECUReset	**ER**	①	◆	①	**11**	**51**	**7F**	$34
readFreezeFrameData	**RFFD**	②		②	**12**	**52**	**7F**	--
clearDiagnosticInformation	**CDI**	◆	①	◆	**14**	**54**	**7F**	--
readDTCByStatus	**RDTCBS**	◆	◆	◆	**18**	**58**	**7F**	--
readDataByLocalId	**RDBLI**	①		①	**21**	**61**	**7F**	$3B
readDataByCommonId	**RDBCI**	◆	◆	◆	**22**	**62**	**7F**	--
readMemoryByAddress	**RMBA**	①	①	①	**23**	**63**	**7F**	$3D
reqCommonIDScalingMasking	**RCISM**	①	①	①	**24**	**64**	**7F**	--
securityAccess	**SA**	①	①	①	**27**	**67**	**7F**	--
communicationControl	**CC**	◆	◆	◆	**28**	**68**	**7F**	--
writeDataByCommonId	**WDBCI**	②	①	②	**2E**	**6E**	**7F**	--
InputOutputCtrlByCommonId	**IOCBCI**	①		①	**2F**	**6F**	**7F**	--
startRoutineByLocalIdentifier	**STRBLI**	◆	①	◆	**31**	**71**	**7F**	$32, $33
stopRoutineByLocalIdentifier	**SPRBLI**	◆	①	◆	**32**	**72**	**7F**	$31, $33
Req.RoutineResultsByLocalId	**RRRBLI**	◆	①	◆	**33**	**73**	**7F**	$31, $32
requestDownload	**RD**		◆	①	**34**	**74**	**7F**	--
requestUpload	**RU**		①	①	**35**	**75**	**7F**	--
transferData	**TD**		◆	①	**36**	**76**	**7F**	$34, $35
requestTransferExit	**RTE**		◆	①	**37**	**77**	**7F**	$34, $35
writeDataByLocalIdentifier	**WDBLI**	①		①	**3B**	**7B**	**7F**	--
writeMemoryByAddress	**WMBA**	①	①	①	**3D**	**7D**	**7F**	--
testerPresent	**TP**	◆	◆	◆	**3E**	**7E**	**7F**	--
controlDTCSetting	**CDTCS**	◆	◆	◆	**85**	**C5**	**7F**	--
requestDiagnosticDataPacket	**RDDP**	②		②	**A0**	**E0**	**7F**	$A1
dynamaicallyDefineDiagnostic DataPacket	**DDDDP**	②		②	**A1**	**E1**	**7F**	$A0
diagnosticCommand	**DC**	①	①	①	**B1**	**F1**	**7F**	--

◆ **Essential service in this diagnostic mode!**
■ **Essential service in this diagnostic mode if ECU supports OBD II requirements!**
① **Selection of service is vehicle manufacturer/project specific; the data content is project specific!**
② **Selection of service requires additional detail to be specified by vehicle manufacturer/system supplier; the data content/structure and/or method is project specific!**
③ **Service shall be reserved mainly for legislative purposes (e.g. emission related test data; future EOBD)!**
④ **Selection of service is vehicle manufacturer/project specific; the service is mandatory if any diagnostic mode other than default diagnostic mode ($81) is supported.**

All diagnostic services not specifically called out in Table 4 shall be considered never supported.

2.2.2.6 Response Code $78 Logic (ref. KWP-GRP-1.5, 5.3.1.4.1)

Refer to Annex A.2 for conditions where response code $78 may be used.

When utilizing response code $78 logic, the initial negative response with response code $78 shall be sent according to the timing requirements as specified in section 2.2.3.7. The timing parameters between each successive response code $78 negative response shall not exceed 4000 ms. An ECU shall respond to a tester request with a valid positive response or negative response (other than response code $78) within 60 seconds of the time of the original tester request. Any deviations to the 60 seconds requirement shall be documented in the ECU's Subsystem Specific Diagnostic Specification.

If the tester does not receive an additional negative response code $78 (or the final positive or negative response), within 5 seconds of the previous negative response code $78, or if the tester does not receive a final positive or negative response (other than response code $78) within 60 seconds (or the time specified in the ECU's Part 2 SSDS) of the original tester request, then the tester shall assume that a response will not be received from the ECU. The tester may initiate any appropriate recovery action and begin other diagnostic sessions and strategies after the timeout period.

While the tester awaits the final positive or negative response from the ECU, it shall continually send any necessary diagnostic messages to the ECU in order to keep the lack of diagnostic dialog timer (see section 2.2.3.8) from timing out. When any valid diagnostic frame is received by an ECU during its response code $78 handling, the ECU shall reset the appropriate timers and respond to the diagnostic frame with another service $7F, response code $78 message containing the same Request Service Identifier (see section 2.2.3.4) that is already being handled with the current response code $78. Note that this Request Service Identifier is not necessarily the same as contained in the diagnostic frame causing the reset of the timers. Any deviations to this requirement shall be detailed in the ECU's Subsystem Specific Diagnostic Specification.

Note: For certain ECUs, the transmission and reception of messages is not possible while performing specific actions. For example, an ECU that is commanded to perform a flash memory erase may be unable to transmit and/or receive messages during this time. Under these circumstances, the ECU shall respond to the initial tester request with a single negative response code $78, and respond when complete with the appropriate positive or negative response. If a second response code $78 (or the final positive or negative response) is not received by the tester within 5 seconds, the tester shall assume that the ECU is in a no transmit/receive state. The tester shall wait a duration of 60 seconds (or the time specified in the ECU's Part 2 SSDS) for the final response. The tester shall not abort the transaction due to a lack of additional response code $78s while waiting. This use of the single response code $78 by an ECU for tasks that take longer than 4 seconds is only allowed while performing Method 3 programming (ref. MC-v2003.0) and requires approval by the R&VT EESE Core Network Communication Section. The specific circumstances where the single response code $78 strategy is utilized shall be detailed in the ECU's Subsystem Specific Diagnostic Specification.

Example of Tester – ECU dialog during service $7F, response code $78 handling	
Message Transfer	**Event or Action**
Tester to ECU	Request diagnosticCommand ($B1) with commandCommonIdentifier of $00B2 (Flash Erase)
ECU to Tester	Response of service $7F, request service identifier $B1, response code $78. *Note: This message is transmitted while the ECU is performing the flash erase if the procedure can not be completed within 50ms. This message must be sent by the ECU at least every 4 seconds until the final positive or negative response for service $B1 is transmitted (after completing the flash erase).*
Tester to ECU	Request testerPresent ($3E) *Note: The reception of this message by the ECU shall cause the appropriate ECU diagnostic timers to be reset.*
ECU to Tester	Response of service $7F, request service identifier **$B1**, response code $78. *Note: This message is in response to the testerPresent ($3E) tester request, but the request service identifier of $B1 is used because the ECU is still servicing the diagnosticCommand ($B1) request.*
ECU to Tester	Positive response ($F1) to request diagnosticCommand ($B1) with commandCommonIdentifier of $00B2. *Note: At this point the task of erasing flash has completed execution, and the ECU is capable of responding to the original tester request for diagnosticCommand ($B1)..*

2.2.2.7 Diagnostic Services (ref. KWP-GRP-1.5, 6-7)

The following sub-sections contain additional information and requirements that either supplement, replace, remove or clarify referenced service requirements in other specifications.

2.2.2.7.1 startDiagnosticSession (ref. KWP-GRP-1.5, 6.1.1)

The parameters values supported for the diagnosticMode parameter of the startDiagnosticSession service are listed in Table 5. No other diagnosticMode values shall be supported.

Table 5: startDiagnosticSession diagnosticMode Values Supported

Parameter (Hex)	Mandatory or Optional	Description	Reference
81	M	DefaultMode-StandardDiagnosticMode-OBDIIMode	ref. KWP-GRP-1.5, Table 6.1.1
85	O	ECUProgrammingMode	ref. KWP-GRP-1.5, Table 6.1.1
87	O	ECUAdjustmentMode	ref. KWP-GRP-1.5, Table 6.1.1
89-F9	O	vehicleManufacturerSpecific	ref. KWP-GRP-1.5, Table 6.1.1
FA-FE	O	systemSupplierSpecific	ref. KWP-GRP-1.5, Table 6.1.1

The baudrateIdentifier parameter shall not be used with this or any service.

One and only one diagnostic session shall be active in an ECU at all times. Diagnostic session $81 shall be active by default upon power-up of an ECU (i.e., a tester request message shall not be required). A tester shall have the capability of changing from any one diagnostic session to another without performing any type of security access (refer to section 2.2.2.7.6 for additional details).

Diagnostic session $87 (ECUAdjustmentMode), if implemented, shall be an extended diagnostic session that is a superset of the diagnostic functionality supported in diagnostic session $81.

Diagnostic session $85 (ECUProgrammingMode), if implemented, shall be used only for Method 3 file download and shall be the implementation of the Method 3 programming requirements as described in MC-v2003.0.

When an ECU transitions from any diagnostic session to another diagnostic session, the ECU shall reset all active diagnostic functionality that is not supported in the new diagnostic session (e.g., security access, I/O control), with the exception of changes written to long term memory. For example, if an ECU only supports service $2F (inputOutputControlByCommonID) in diagnostic session $87 then any actively controlled inputs or outputs shall revert back to the normal value as determined by the control system upon a transition from session $87 to session $81.

An ECU is allowed a 750ms re-initialization period upon returning to the default diagnostic session ($81) from any other diagnostic session. During this re-initialization period the ECU is not required to respond to any diagnostic requests.

The only defined vehicleManufacturerSpecific diagnostic session is $F0 and is defined as EOLExtendedDiagnosticSession. Any ECU that needs to grant special privileges to an End of Line test tool shall implement diagnostic session $F0 to provide access to these privileges. For example, an ABS module that normally exits to the defaultDiagnosticSession when vehicle speed is greater than a given value may maintain the EOLExtendedDiagnosticSession (despite the vehicle speed). Diagnostic session $F0 shall be reserved only for use by End of Line test tools in assembly plants and shall be implemented only when needed to verify communication with an End of Line tester. This EOLExtendedDiagnosticSession shall grant access to all functionality (e.g., diagnostic services, privileges, input/output control, etc.) that is necessary during the assembly testing and shall contain a superset of the diagnostic functionality supported in diagnostic session $87 (ECUAdjustmentMode).

SystemSupplierSpecific diagnostic sessions shall not be supported by Ford test tools. All implemented systemSupplierSpecific diagnostic sessions shall be specified by the module designer and documented in the ECU's Subsystem Specific Diagnostic Specification (SSDS/Part 2).

2.2.2.7.2 ECUReset (ref. KWP-GRP-1.5, 6.5)

Only the powerOn value of the resetMode (RM_) parameter shall be supported. The resetStatus (RS_) parameter shall not be supported.

The positive response to an ECUReset (service $11) request shall occur before the ECU performs the reset. An ECU is allowed a 750ms re-initialization period after providing a positive response to an ECUReset request. During this re-initialization period the ECU is not required to respond to any diagnostic requests.

2.2.2.7.3 clearDiagnosticInformation (ref. KWP-GRP-1.5, 8.5)

This service shall clear all groups of DTCs in the ECU, including powertrain, chassis, body and network.

Only the value of $FF00 shall be supported for the groupOfDiagnosticInformation (GODI_) parameter, specifying that all groups shall be cleared in every clearDiagnosticInformation service request.

2.2.2.7.4 readMemoryByAddress (ref. KWP-GRP-1.5, 7.3)

This service is mainly intended for development purposes only. Most service and end-of-line equipment do not support the use of this service.

Every instance of this message shall include all four address bytes as specified in Table 6. If the ECU supports less than four address bytes, then every byte that is not used shall be set to zero in the request message with data byte #5 always corresponding to the LSB of the ECU's address.

Table 6: readMemoryByAddress

Data Byte	Parameter Name	Cvt	Hex Value	Mnemonic
#1	**readMemoryByAddress Request**	**M**	**23**	**RMBA**
#2	memoryAddress { MSB of the ECU's address – byte 3}	M	xx	**MA_...**
#3	memoryAddress { ECU's address – byte 2}	M	xx	
#4	memoryAddress { ECU's address – byte 1}	M	xx	
#5	memoryAddress { LSB of the ECU's address – byte 0}	M	xx	
#6	MemorySize (MSB)	M	xx	**MS_...**
#7	MemorySize (LSB)	M	xx	**MS_...**

The MemorySize parameter shall be limited to a maximum of 4094 bytes.

2.2.2.7.5 writeMemoryByAddress (ref. KWP-GRP-1.5, 7.7)

This service is mainly intended for development purposes only. Most service and end-of-line equipment do not support the use of this service.

Every instance of this message shall include all four address bytes as specified in Table 7. If the ECU supports less than four address bytes then every byte that is not used shall be set to zero in the request message with data byte #5 always corresponding to the LSB of the ECU's address.

The MemorySize parameter shall be limited to a maximum of 4088 bytes.

Table 7: writeMemoryByAddress Request Message

Data Byte	Parameter Name	Cvt	Hex Value	Mnemonic
#1	**writeMemoryByAddress Request**	**M**	**3D**	**WMBA**
#2	memoryAddress { MSB of the ECU's address – byte 3}	M	xx	**MA_...**
#3	memoryAddress { ECU's address – byte 2}	M	xx	
#4	memoryAddress { ECU's address – byte 1}	M	xx	
#5	memoryAddress { LSB of the ECU's address – byte 0}	M	xx	
#6	MemorySize (MSB)	M	xx	**MS_...**
#7	MemorySize (LSB)	M	xx	**MS_...**
#8	RecordValue#1	M	xx	**RV_…**
.	.	.	.	.
.	.	.	.	.
#n	RecordValue#m	U	xx	**RV_…**

Table 8: writeMemoryByAddress Positive Response Message

Data Byte	Parameter Name	Cvt	Hex Value	Mnemonic
#1	**writeMemoryByAddress Positive Response**	**M**	**7D**	**WMBAPR**
#2	memoryAddress { MSB of the ECU's address – byte 3}	M	xx	**MA_...**
#3	memoryAddress { ECU's address – byte 2}	M	xx	
#4	memoryAddress { ECU's address – byte 1}	M	xx	
#5	memoryAddress { LSB of the ECU's address – byte 0}	M	xx	

2.2.2.7.6 securityAccess (ref. KWP-GRP-1.5, 6.3.2 – ref. P1-v2003.0, 9.4)

Security access requirements are specified in both the KWP-1.5 and P1-v2003.0 specifications.

The referenced 10 second time delay from KWP-GRP-1.5 (after two unsuccessful key submittals and after ECU power on) is not applicable on Ford ECUs due to the number of combinations in the required three byte seed and key.

The sequence of events required when a given diagnostic session requires security access in order to

perform secured functions is as follows:

1) startDiagnosticSession service
2) securityAccess service

The above sequence does not necessarily prohibit a tester from unsecuring an ECU before entering a given diagnostic session, but it does require the ECU to allow the tester the capability to move from any one diagnostic session to another without performing security access.

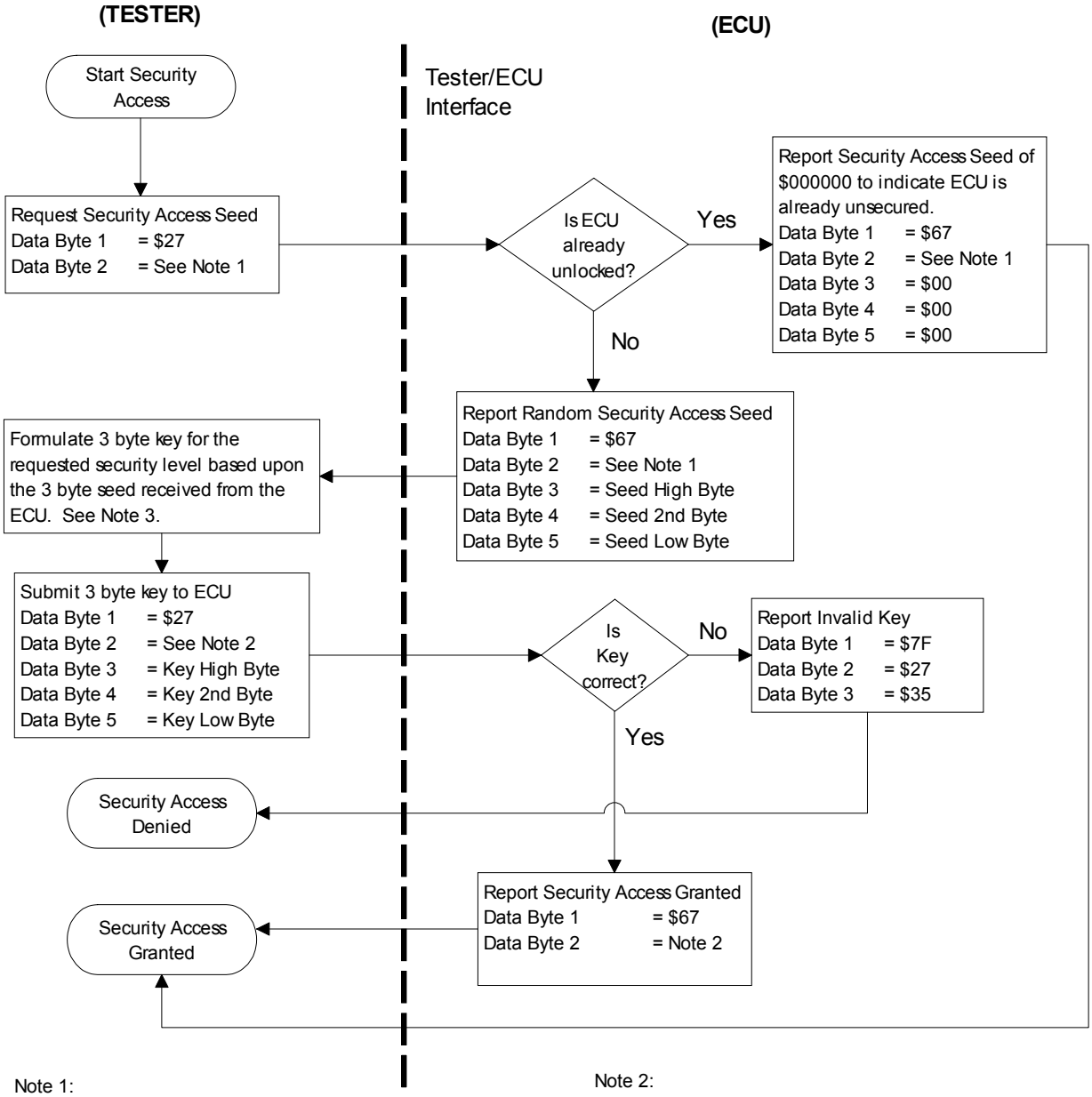
The securityAccessStatus parameter shall not be used, see Figure 3 for more details.

2.2.2.7.6.1 Seed and Key Procedure

The security access procedure shown in Figure 3 shall be used to "unlock" the ECU and grant security access. Upon receipt of a valid seed request, the ECU shall respond with a randomly chosen security seed. This seed shall remain valid or "active" in the ECU until one of the following conditions occur:

- A new validly formatted seed request is received by the ECU (regardless of whether it is for the same or a different security level and regardless of whether the security level is supported by the ECU or not).
- A security key (correct or incorrect) is received by the ECU (regardless of whether it is for the same or a different security level and regardless of whether the security level is supported by the ECU or not).
- Lack of Diagnostic Dialog timer expires, as defined in Section 2.2.3.8 of this document.

An ECU shall reject any security key sent by a test tool for a given security level if a valid or "active" seed does not exist for that security level, with a service $7F and response code $22.



Note 3:
Security algorithm may be obtained upon request from Core Network Communications.
The module Part 2 specification shall define which security levels are supported as well as the constants used in the security algorithm for each security level supported. Constants used in the security algorithm shall all be non-zero values.

Figure 3: CAN Security Access Procedure

2.2.2.7.7 readDataByCommonIdentifier (ref. KWP-GRP-1.5, 7.2)

This service is equivalent to the reading of PIDs as defined in P1-v2003.0. CommonIDs (PIDs) are not limited to a maximum of four bytes on the CAN protocol. Definitions of all CommonIdentifiers, except for those in the supplier reserved range, must be approved by the R&VT EESE Core Network Communication Section. All approved CommonIDs (PIDS) and their descriptions are defined and maintained in the MRDB.

2.2.2.7.8 readDiagnosticTroubleCodesByStatus (ref. KWP-GRP-1.5, 8.2.1.1)

The parameter values supported for the statusOfDTC (for the request message service $18) parameter are listed in Table 9, no other parameter values shall be supported.

Table 9: readDiagnosticTroubleCodesByStatus statusOfDTC Values Supported

Hex	Description	Man / Opt	Mnemonic
00	**requestIdentifiedBCDDTCAndStatus (SAE J2012 format)** This value is used by the client (tester) to indicate to the server(s) (ECU(s)) that the server(s) (ECU(s)) shall respond with all diagnostic trouble codes (DTCs) which have been identified to cause a problem at the time of the request (i.e., stored in non-volatile memory).	M	**RIDTCAS**
01	**requestSupportedBCDDTCAndStatus (SAE J2012 format)** This value is used by the client (tester) to indicate to the server(s) (ECU(s)) that the server(s) (ECU(s)) shall respond with all supported diagnostic trouble codes (DTCs) regardless of their status. Even if no DTC is stored in non-volatile memory (no DTC causes a problem at the time of the request) the server shall report all DTCs which are supported. This allows the client (tester) to test the DTCReadinessFlag in the statusOfDTC parameter of each supported DTC.	O	**RSUDTCAS**
F0-F9	**VehicleManufacturerSpecific** This range of values is reserved for vehicle manufacturer specific use.	O	**VMS**
FA-FE	**SystemSupplierSpecific** This range of values is reserved for system supplier use and shall not be supported by Ford test tools.	O	**SSS**

A value of $00 in the statusOfDTC in the **request** message shall trigger all codes to be reported in the positive response message that have the DTCStorageState values of either b01 (b = binary) "**DTCNotPresent at time of request**" or b11 "**DTCPresent at time of request**". The storage state of "DTCNotPresent at time of request" means that the DTC is currently being aged. The storage state of "DTCPresent at time of request" means that the fault(s) that set the DTC is still present.

Only the value of $FF00 shall be supported for the groupOfDTC (GODTC_) parameter specifying that all groups shall be requested in every requestDiagnosticTroubleCodes service request.

The only allowable DTCFaultSymptoms for the statusOfDTC parameter in the response message $58 are 0b0000 and 0b1111. No other values shall be allowed. Both values represent that the DTC identified does not require any additional fault symptoms (due to the specifics of the DTC descriptions in the MRDB). A value of 0b0000 shall represent that all values in the statusOfDTC parameter are supported. A value of 0b1111 shall represent that the entire statusOfDTC parameter is not supported.

NOTE: This service is only associated with requesting continuous DTCs. DTCs are not required to be reported in the same sequence as they have been detected.

Definitions of all DTCs must be approved by the R&VT EESE Core Network Communication Section. All approved DTCs and their descriptions are defined and maintained in the MRDB. A complete description of the logic used by the ECU to set each supported DTC shall be documented within the ECU's SSDS.

NOTE: If an ECU is reporting more than 255 DTCs in the response message $58, the numberOfDTC

parameter shall be capped at 255 ($FF). All DTCs shall still be reported and the first frame data length (FF_DL) contained within the PCI shall still accurately reflect the total number of relevant message bytes.

2.2.2.7.9 readFreezeFrameData (ref. KWP-GRP-1.5, 8.4.1)

Each freeze frame can specify different types of information relating to the DTC that it is associated with. All freeze frame information shall be specified by the module designer and documented in the ECU's Subsystem Specific Diagnostic Specification (SSDS/Part 2). If the freeze frame data is not defined by embedded PID references, then once a freeze frame number (FFNR_) is defined for a DTC in an ECU, its format shall be the same for any DTC on that ECU that chooses to implement that freeze frame number (reference P1-v2003.0, 6.3.2.5 for additional details). If the freeze frame data is defined by embedded PID references, then a freeze frame number (FFNR_) format can contain different data for different DTCs.

Only a value of $00 or $04 shall be used for the RAMI_ parameter.
The value of the RI_ parameter shall specify the two byte DTC that the freeze frame is associated with.

2.2.2.7.9.1 Response Message Definition

The number of positive responses returned based upon a readFreezeFrameData request is determined by the RAMI_ parameter, the freeze frame number (FFNR_), and the number of DTCs with freeze frames stored in the ECU. The cases detailed below illustrate the appropriate responses.

The following simplified example assumes only two DTCs are supported in the ECU. DTC $1234 stores freeze frame numbers $00 and $01. DTC $5678 stores freeze frame numbers $01 and $02.

Case 1 Request Message: RAMI_ = $04, FFNR_ = $00, RI_ = $1234

- If DTC $1234 is **not** stored in non-volatile memory, the ECU shall respond with a single negative response with response code $31 (request out of range) to indicate that the DTC does not have freeze frame data associated with it.
- If DTC $1234 is stored in non-volatile memory, the ECU shall respond with a single positive response containing the appropriate data.

Case 2 Request Message: RAMI_ = $04, FFNR_ = $02, RI_ = $1234

- The ECU shall respond with a negative response with response code $31 (request out of range) to indicate that the DTC does not support this freeze frame number.

Case 3 Request Message: RAMI_ = $04, FFNR_ = $FF, RI_ = $1234

- If DTC $1234 is **not** stored in non-volatile memory, the ECU shall respond with a single negative response with response code $31 (request out of range) to indicate that the DTC does not have freeze frame data associated with it.
- If DTC $1234 is stored in non-volatile memory, the ECU shall respond with two separate positive responses (one response containing freeze frame number $00 data and the other response containing freeze frame number $01 data).

Case 4 Request Message: RAMI_ = $00, FFNR_ = $00, RI_ = Not Present in Request

- If DTC $1234 is **not** stored in non-volatile memory, the ECU shall return a single positive response with FFNR_ = $01, RAMI_ = $00, FFD = Not Present in Response, RI_ = $0000.
- If DTC $1234 is stored in non-volatile memory, the ECU shall respond with a single positive response containing the appropriate data.

Case 5 Request Message: RAMI_ = $00, FFNR_ = $01, RI_ = Not Present in Request

- If neither DTC $1234 nor DTC $5678 are stored in non-volatile memory, the ECU shall return a single positive response with FFNR_ = $01, RAMI_ = $00, FFD = Not Present in Response, RI_ = $0000.
- If either DTC $1234 or DTC $5768 are stored in non-volatile memory, the ECU shall respond with a separate positive response message for each DTC stored in non-volatile memory that supports freeze frame number $01.

Case 6 Request Message: RAMI_ = $00, FFNR_ = $03, RI_ = Not Present in Request

- The ECU shall return a single negative response with response code $31 (request out of range) to indicate that freeze frame number $03 is never supported in the ECU.

Case 7 Request Message: RAMI_ = $00, FFNR_ = $FF, RI_ = Not Present in Request

- If neither DTC $1234 nor DTC $5678 are stored in non-volatile memory, the ECU shall return a single positive response with FFNR_ = $FF, RAMI_ = $00, FFD = Not Present in Response, RI_ = $0000.
- If either DTC $1234 or DTC $5768 are stored in non-volatile memory, the ECU shall respond with a separate positive response message for each freeze frame number for each DTC stored in non-volatile memory.

2.2.2.7.10 inputOutputControlByCommonIdentifier (ref. KWP-GRP-1.5, 9.2.1)

The inputOutputCommonIdentifier values shall refer to Parameter Identifiers (PIDs) specified in the MRDB.

Table 10: Allowable inputOutputControlParameter values

Hex	Description	Man or Opt	Mnemonic
00	**ReturnControlToECU** This value shall indicate to the server (ECU) that the client (tester) no longer has control about the input signal, internal parameter or output signal referenced by the inputOutputCommonIdentifier.	M	**RCTECU**
05	**FreezeCurrentState** This value shall indicate to the server (ECU) that it is requested to freeze the current state of the input signal, internal parameter or output signal referenced by the inputOutputCommonIdentifier.	O	**FCS**
07	**ShortTermAdjustment** This value shall indicate to the server (ECU) that it is requested to adjust the input signal, internal parameter or output signal referenced by the inputOutputCommonIdentifier in RAM to the value(s) included in the controlOption parameter(s). (e.g. set Idle Air Control Valve to a specific step number, set pulse width of valve to a specific value/duty cycle).	M	**STA**

The controlState (CS_) parameter shall only be supported in $2F request messages with an inputOutputControlParameter of 0x07 "shortTermAdjustment" and is mandatory.

The controlState (CS_) parameter shall only be supported in $6F positive response messages with an inputOutputControlParameter of 0x05 "freezeCurrentState" or 0x07 "shortTermAdjustment" and is mandatory.

Definitions of all inputOutputCommonIdentifiers, except for those in the supplier reserved range, must be approved by the R&VT EESE Core Network Communication Section. All approved inputOutputCommonIdentifiers and their descriptions are defined and maintained in the MRDB. Each inputOutputCommonIdentifier shall only use the encoding in the MRDB and other encoding values shall not be allowed.

All implemented inputOutputCommonIdentifiers shall be documented in the ECU's Subsystem Specific Diagnostic Specification, including supplier reserved identifiers.

2.2.2.7.11 startRoutineByLocalIdentifier (ref. KWP-GRP-1.5, 10.1.2)

This service shall not use the REYS_ parameter.

The routineEntryOption (REYO_) parameter can be of any needed length in bytes required by an ECU and shall specify additional information relating to a routine. The REYO_ parameter shall be specified in the SSDS and is associated with each routine number. The REYO_ parameter is mandatory and shall have a length of one and a value of $00 when it is not used by a specific execution routine (i.e., no additional information is added to the routineLocalId parameter through the REYO_ parameter).

The mandatory routine local identifier of $02, on-demand self-test, shall always be supported in the default diagnostic session ($81).

Definitions of all routines must be approved by the R&VT EESE Core Network Communication Section. All approved routines and their descriptions are defined and maintained in the MRDB.

2.2.2.7.12 stopRoutineByLocalIdentifier (ref. KWP-GRP-1.5, 10.3.1, 10.3.2)

The mandatory and only allowable RETO_ parameter values are specified in Table 11 and are only one byte in length.

Table 11: Allowable RETO_ values

Value (Hex)	Description
00	**Exit immediately**

The Ford defined RETS_ parameter values are specified in Table 12. RETS_ is used in the positive response to the stopRoutineByLocalIdentifier and stopRoutineByAddress services and is used to indicate the status of the routine exit.

Table 12: Allowable RETS_ values

Hex	Description	Mnemonic
63	AbnormalExitWithResultsAvailable	**AEWRA**
64	AbnormalExitWithoutResultsAvailable	**AEWORA**
FA – FE	systemSupplierSpecific This range of values is reserved for system supplier specific use.	**SSS**

2.2.2.7.13 requestRoutineResultsByLocalIdentifier (ref. KWP-GRP-1.5, 10.5.2)

All DTCs logged by any execution routine shall be stored and aged as continuous codes that can be retrieved by service $18. Also, all codes that are logged by an execution routine shall be available by service $33 as long as the diagnostic session in which the test was ran is still active. An ECU shall not maintain the on-demand results for more than one diagnostic test at any given time. The tester is responsible for obtaining the results to a specific diagnostic test prior to the execution of another test. All codes logged by an execution routine shall not be available by service $33 after leaving the diagnostic session in which the test was ran, unless they are logged again by running another or the same execution routine.

The ECU shall always respond with a negative response code of busy-repeatRequest ($21) to a requestRoutineResultsByLocalIdentifier request that is received while the requested Local ID routine is currently executing.

The RRS_ parameter shall consist of a routineStatus byte followed by an optional routineDataResults parameter. The possible values that the routineStatus byte can be are listed in Table 13.

Table 13: Allowable routineStatus Byte values

Hex	Description
00	Routine completed and passed (e.g., no on-demand DTCs detected during self-test)
01	Routine completed and failed (e.g., one or more on-demand DTCs detected during self-test)
02	Routine aborted

The routineDataResults parameter for the local identifiers of $02 (on-demand self-test) and $11 (assembly self-test) shall be a list of 2 byte DTCs. The format of the routineDataResults parameter for all execution routines shall be documented in the ECU's SSDS. Table 14 shows the layout of the requestRoutineResultsByLocalIdentifier positive response message.

Table 14: Allowable Routine Status Byte values

Data Byte	Parameter Name	Cvt	Hex Value	Mnemonic
#1	**requestRoutineResultsByLocalIdentifier Pos. Resp. Service Id**	**M**	**73**	**RRRBLIPR**
#2	RoutineLocalIdentifier	M	xx	**RELI_...**
#3	routineStatus	M	00, 01, or 02	
:	routineDataResults#1	O	:	} **RRS_**
:	:	:	:	
#n	routineDataResults#m	O	xx	

2.2.2.7.14 requestDownload (ref. KWP-GRP-1.5, 11.1.2)

The requestDownload service parameters are listed in Table 15

Table 15: requestDownload Service Message

Data Byte	Parameter Name	Cvt	Hex Value	Mnemonic
#1	**requestDownload Request Service Id**	**M**	**34**	**RD**
	transferRequestParameter=[			**TRTP_**
#2	memoryAddress { MSB of the ECU's address – byte 3}	M	xx	**MA_**
#3	memoryAddress { ECU's address – byte 2}	M	xx	**MA_**
#4	memoryAddress { ECU's address – byte 1}	M	xx	**MA_**
#5	memoryAddress { LSB of the ECU's address – byte 0}	M	xx	**MA_**
#6	dataFormatIdentifier	M	xx	**DFI**
#7	unCompressedMemorySize { High Byte }	M	xx	**UCMS**
#8	unCompressedMemorySize { Middle Byte }	M	xx	**UCMS**
#9	unCompressedMemorySize { Low Byte }]	M	xx	**UCMS**

It is recommended that the dataFormatIdentifier (DFI) parameter shall always have the value of $00, specifying that compression and encryption are not used. However, if compression or encryption is required, the test tool shall not be responsible for any compression or encryption of the data.

2.2.2.7.15 requestDownload Positive Response(ref. KWP-GRP-1.5, 11.1.2)

The requestDownload positive response shall consist of the message as shown in Table 16.

Table 16: requestDownload Positive Response Service Message

Data Byte	Parameter Name	Cvt	Hex Value	Mnemonic
#1	**requestDownload Positive Response Service Id**	**M**	**74**	**RDPR**
#2	transferResponseParameter=[maxNumberOfBlockLength – MSB]	M	xx	**TREP_ MNROBL**
#3	transferResponseParameter=[maxNumberOfBlockLength – LSB]	M	xx	**TREP_ MNROBL**

The transferResponseParameter shall specify the number of data bytes that the ECU will be able to receive within each transferData request message, so that the ECU may always respond with a BlockSize value of 0 in the flow control frame (ref. ISO 15765-2) when the number of bytes specified in the first frame of the transferData request message is less than or equal to the transferResponseParameter.

2.2.2.7.16 requestUpload (ref. KWP-GRP-1.5, 11.2.2)

The requestUpload service parameters are listed in Table 17

Table 17: requestUpload Service Message

Data Byte	Parameter Name	Cvt	Hex Value	Mnemonic
#1	**requestUpload Request Service Id**	**M**	**35**	**RU**
	transferRequestParameter=[			**TRTP_**
#2	memoryAddress { MSB of the ECU's address – byte 3}	M	xx	**MA_**
#3	memoryAddress { ECU's address – byte 2}	M	xx	**MA_**
#4	memoryAddress { ECU's address – byte 1}	M	xx	**MA_**
#5	memoryAddress { LSB of the ECU's address – byte 0}	M	xx	**MA_**
#6	dataFormatIdentifier	M	xx	**DFI**
#7	unCompressedMemorySize { High Byte }	M	xx	**UCMS**
#8	unCompressedMemorySize { Middle Byte }	M	xx	**UCMS**
#9	unCompressedMemorySize { Low Byte }]	M	xx	**UCMS**

It is recommended that the dataFormatIdentifier (DFI) parameter shall always have the value of $00, specifying that compression and encryption are not used. However, if compression or encryption is required, the test tool shall not be responsible for any compression or encryption of the data.

2.2.2.7.17 requestUpload Positive Response(ref. KWP-GRP-1.5, 11.2.2)

The requestUpload positive response shall consist of the message as shown in Table 18.

Table 18: requestUpload Positive Response Service Message

Data Byte	Parameter Name	Cvt	Hex Value	Mnemonic
#1	**requestUpload Positive Response Service Id**	**M**	**75**	**RDPR**

If upload functionality is supported, the upload flowchart in Figure 5 shall be used to transfer data from an ECU's memory to an off-board tester.

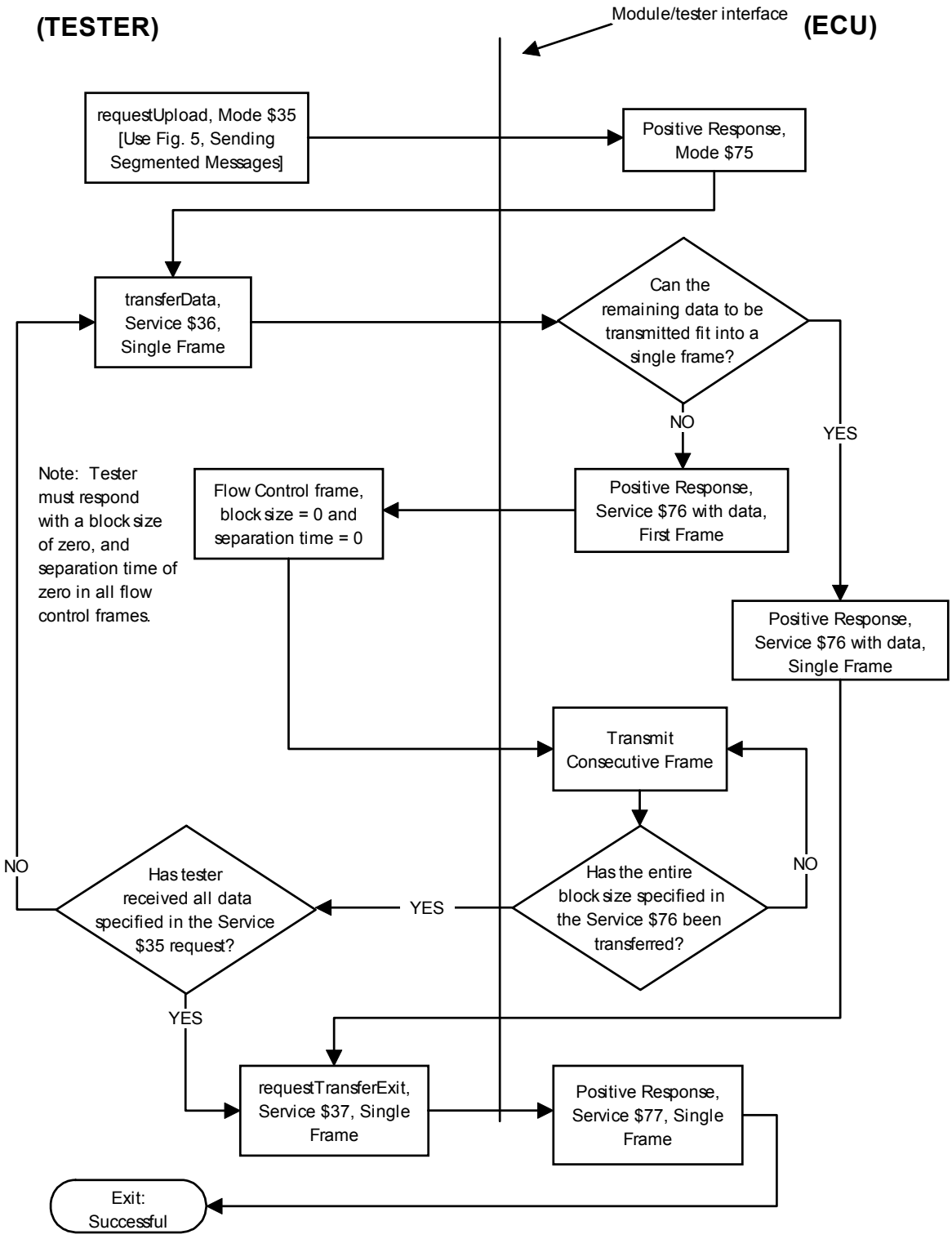


Figure 4: CAN Upload Flowchart

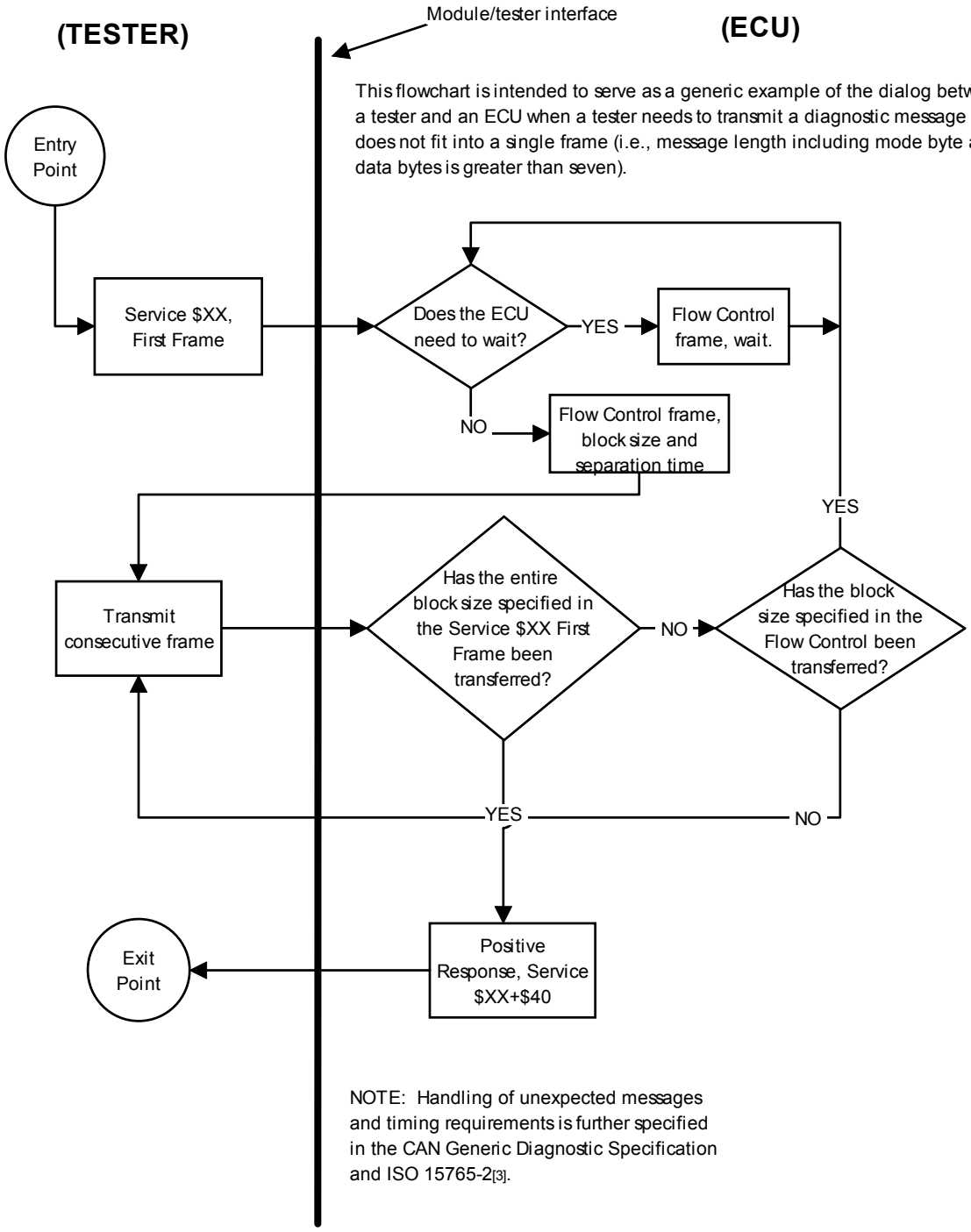


Figure 5: Sending Segmented Messages on CAN

2.2.2.7.18 transferData (ref. KWP-GRP-1.5, 11.3.1)

The TRTP_ parameter is the data transferred in the download operation and the TREP_ parameter is the data transferred in the upload operation. These parameters shall have a maximum length of 4094 bytes.

2.2.2.7.19 requestTransferExit (ref. KWP-GRP-1.5, 11.4.2)

The TRTP_ and TREP_ parameters shall not be used for this service. The only parameter in the request and positive response messages for this parameter shall be the RTE parameter.

2.2.2.7.20 testerPresent (ref. KWP-GRP-1.5, 6.4)

The responseRequired parameter shall be supported for all ECUs. Only the values of $01 (Yes) and $02 (No) shall be supported for the responseRequired parameter.

Table 19: testerPresent Request Message

Data byte	Parameter Name	Cvt	Hex Value	Mnemonic
#1	**TesterPresent Request Service Id**	**M**	**3E**	**TP**
#2	sub-function = [responseRequired]	M	Xx	RRD_

2.2.2.7.21 diagnosticCommand (ref. P1-v2003.0, 15)

The diagnosticCommand request message is shown in Table 20.

Table 20: diagnosticCommand Request Message

Data Byte	Parameter Name	Cvt	Hex Value
#1	**diagnosticCommand Request Service Identifier**	**M**	**B1**
#2	CommandCommonIdentifier (High Byte)	M	Xx
#3	CommandCommonIdentifier (Low Byte)	M	Xx
#4	Command parameter #1 (Optional – must follow MRDB definition)	O	Xx
#5	Command parameter #2 (Optional – must follow MRDB definition)	O	Xx
#6	Command parameter #3 (Optional – must follow MRDB definition)	O	Xx
#7	Command parameter #4 (Optional – must follow MRDB definition)	O	Xx

Definitions of all CommandCommonIdentifiers, except for those in the supplier reserved range, must be approved by the R&VT EESE Core Network Communication Section. All approved CommandCommonIdentifiers and their descriptions are defined and maintained in the MRDB. Each CommandCommonIdentifier shall only use the length and encoding as specified in the MRDB. Note that the $B1 service request has a variable length, dependent upon the data length of the requested CommandCommonIdentifer.

The positive response for diagnostic commands is listed in Table 20.

Table 21: diagnosticCommand Positive Response Message

Data Byte	Parameter Name	Cvt	Hex Value
#1	**diagnosticCommand Positive Response Service Id**	**M**	**F1**
#2	commandCommonIdentifier (High Byte)	M	Xx
#3	commandCommonIdentifier (Low Byte)	M	Xx

The negative response for diagnostic commands is listed in Table 22.

Table 22: diagnosticCommand Negative Response Message

Data Byte	Parameter Name	Cvt	Hex Value
#1	**negativeResponse service Id**	**M**	**7F**
#2	**diagnosticCommand Service Id**	**M**	**B1**
#3	ResponseCode	M	Xx

2.2.2.7.22 dynamicallyDefineDiagnosticDataPacket [A1] (ref. P1-v2003.0, 16.10)

Table 23, Table 24 and Table 25 list the message parameters for this service.
All references in P1-v2003.0 shall be changed from $2C to $A1.

The ECU shall not accept configurations for any DPID that would require the reporting of the DPID to be larger than a single frame.

2.2.2.7.22.1 Request Message Definition

Table 23: dynamicallyDefineDiagnosticDataPacket Message

Data Byte	Parameter Name	Cvt	Hex Value	Mnemonic
#1	**DynamicallyDefineDiagnosticDataPacket Request Service Id**	**M**	**A1**	**DDDDP**
#2	Data Packet Identification (DPID) Number	M	xx	DPIDN
#3	DPID Format Byte	M	xx	DPIDFB
#4	Reference Number 1	M	xx	RN1
#5	Reference Number 2	M	xx	RN2
#6	Reference Number 3	C1	xx	RN3
#7	Reference Number 4	C1	xx	RN4

C1 = Parameter is only present if the DPID is being defined by memory address.

2.2.2.7.22.1.1 Request Message Data Parameter Definitions

The DPID Number (DPIDN) parameter is a single byte identifier used to specify the logical DPID number that is being defined.

The DPID Format Byte (DPIDFB) contains the DPID definition details, including the method of definition, starting position of data, and size. The DPID Format Byte (DPIDFB) shall use only the values and definitions in Table 24 to define a DPID.

Table 24: DPID Format Byte (DPIDFB) Parameter Values

Bit Number	Bit Description
7-6	**DPID Definition Method** 0b00 = Undefined 0b01 = Define by PID / CommonID 0b10 = Define by Memory Address 0b11 = Reserved for Manufacturer defined values
5-3	**DPID Parameter Position** This value specifies the position of the defined DPID parameter in the message frame where 0b001 represents the data bye position after the DPID #. Valid range is 0b001 – 0b101.
2-0	**DPID Parameter Number of Bytes** This value specifies the number of data bytes for the DPID parameter (i.e., the data length of the DPID being defined). Valid range is 0b001 – 0b101.

The Referenced Number parameter contains the number for the item being added to the DPID. If the DPID is being defined by PID / CommonID (i.e., DPID Definition Method of 0b01), then RN1 shall correspond to the PID / CommonID most significant byte, RN2 shall correspond to the PID / CommonID least significant byte, and RN3 and RN4 shall not be included in the request. If the DPID

is being defined by Memory Address (i.e., DPID Definition Method of 0b10), then RN1 shall correspond to the most significant byte of the memory address, RN2 shall correspond to the 2nd most significant byte, RN3 shall correspond to the 3rd most significant byte, and RN4 shall correspond to the least significant byte of the memory address. If less than four address bytes are supported by an ECU, every Reference Number parameter that is not used shall be set to zero with RN4 always corresponding to the least significant byte of the memory address.

A DPID can be cleared by specifying zero ($00) as the DPID format byte of a service $A1 request.

2.2.2.7.22.2 Positive Response Message Definition

Table 25: dynamicallyDefineDiagnosticDataPacket Positive Response Message

Data Byte	Parameter Name	Cvt	Hex Value	Mnemonic
#1	**DynamicallyDefineDiagnosticDataPacket Response Service Id**	**M**	**E1**	**DDDDPPR**
#2	Data Packet Identification (DPID) Number	M	xx	DPIDN

2.2.2.7.22.2.1 Positive Response Message Data Parameter Definitions

The DPID Number (DPIDN) parameter is an echo of the DPID Number parameter from the request message.

2.2.2.7.23 requestDiagnosticDataPacket(s) [A0] (ref. P1-v2003.0, 16.10.2)

Table 26, Table 27, and Table 28 list the message parameters for this service.
All references in P1-v2003.0 shall be changed from $2A to $A0.

2.2.2.7.23.1 Request Message Definition

Table 26: requestDiagnosticDataPacket(s) Message

Data Byte	Parameter Name	Cvt	Hex Value	Mnemonic
#1	**RequestDiagnosticDataPacket(s) Request Service Id**	**M**	**A0**	**RDDP**
#2	Data Rate	M	xx	DR
#3	DPID Number 1	C1	xx	DPID1
#4	DPID Number 2	C2	xx	DPID2
#5	DPID Number 3	C2	xx	DPID3
#6	DPID Number 4	C2	xx	DPID4
#7	DPID Number 5	C2	xx	DPID5

C1 = Parameter is only present if the Data Rate is equal to $00, $01, $02, $03, $04 or $09.
C2 = Parameter may only be present if DPID Number 1 parameter is present. The existence of this parameter depends upon dynamic usage by the requester.

2.2.2.7.23.1.1 Request Message Data Parameter Definitions

The Data Rate contains the action to be performed by this request. The Data Rate shall use only the values and definitions in Table 27.

Table 27: Data Rate (DR) Parameter Values

Hex Value	Description	Man or Opt.
00	Stop sending data for listed DPIDs in this message	O
01	Send 1 response for each listed DPID in this message	C1
02	Repeat continuously at slow rate for listed DPIDs in this message	C1
03	Repeat continuously at medium rate for listed DPIDs in this message	C1
04	Repeat continuously at fast rate for listed DPIDs in this message	C1
05	Stop sending data for all defined DPIDs in the ECU	M
06	Repeat continuously at slow rate for all defined DPIDs in the ECU	C1
07	Repeat continuously at medium rate for all defined DPIDs in the ECU	C1
08	Repeat continuously at fast rate for all defined DPIDs in the ECU	C1
09	Stop Sending and clear all listed DPIDs in this message from memory	O
0A	Stop Sending and clear all defined DPIDs in the ECU from memory	M

C1 = A minimum of one of the data rates ($01, $02, $03, $04, $06, $07, $08) shall be supported if this service is implemented.

The DPID Number parameter contains a single byte identifier used to specify the logical DPID number that is being requested.

2.2.2.7.23.2 Positive Response Message Definition

Table 28: requestDiagnosticDataPacket(s) Positive Response Message

Data Byte	Parameter Name	Cvt	Hex Value	Mnemonic
#1	**RequestDiagnosticDataPacket(s) Response Service Id**	**M**	**E0**	**RDDPPR**
#2	Data Packet Identification (DPID) Number	C1	xx	DPIDN
#3	DPID Data Byte 1	C2	xx	DPIDDB1
#4	DPID Data Byte 2	C2	xx	DPIDDB2
#5	DPID Data Byte 3	C2	xx	DPIDDB3
#6	DPID Data Byte 4	C2	xx	DPIDDB4
#7	DPID Data Byte 5	C2	xx	DPIDDB5

C1 = Parameter is only present when the Data Rate in the request message is $01, $02, $03, $04, $06, $07, or $08.

C2 = Parameter is only present when the DPIDN parameter is present and this parameter is needed to report all of the requested DPID data bytes (i.e., it is dependent on the defined size of the requested DPID Number).

2.2.2.7.23.2.1 Positive Response Message Data Parameter Definitions

The DPID Number (DPIDN) parameter is an echo of the DPID Number parameter from the request message. The DPID Data Byte parameters correspond to the actual data contained in the requested defined DPID.

2.2.2.7.23.3 General Reporting Requirements

All reporting of DPIDs from an ECU (positive response $E0) shall fit within a single frame.

Each DPID Number and its corresponding data shall be transmitted in a separate positive response as shown in Table 28. For example, a service $A0 request with more than one DPID Number and a Data Rate of $01, $02, $03, $04, $06, $07, or $08 shall result in a separate service $E0 response for each DPID Number.

An ECU shall report rapid data packets in one of the following ways. The ECU shall either report the DPIDs in the order the DPIDs were requested from the tester or the ECU shall report the DPIDs in ascending numerical order based upon DPID number.

The positive response to a service $A0 tester request with a data rate set to $00, $05, $09, or $0A shall be a single message consisting of only $E0 for the service byte (no additional data bytes, except padding).

2.2.2.7.23.4 Periodic Positive Responses

Periodic positive diagnostic data packet responses ($E0) to service $A0 (request diagnostic data packet) requests shall be transmitted on separate CAN transmission IDs than all other diagnostic responses. These periodic diagnostic IDs are defined and maintained in the MRDB as CAN "Rapid Data" IDs for each ECU where applicable. Some ECUs may require more than one periodic CAN rapid data transmit ID. All diagnostic requests shall still be received on the normal ECU CAN reception ID. Note that non-periodic positive $E0 responses, such as a positive response to a service $A0 request with data rate set to $00, $01, $05, $09, or $0A shall still be transmitted on the regular CAN transmission ID.

For example, the PCM shall receive service $A0 requests on CAN ID $7E0, but shall transmit periodic $E0 responses on CAN IDs $6A0 and $6A1, as defined in the MRDB.

An ECU shall continue to transmit the periodic positive for a particular DPID until one of the following conditions occur:

1) The lack of diagnostic dialog timer expires (see section 2.2.3.8)
2) A request is received to stop sending the particular DPID (i.e., a service $A0 request with a data rate of $00 or $09 is received for the particular DPID or a service $A0 request with a data rate of $05 or $0A is received).

2.2.2.8 Diagnostic Gateway Requirements

2.2.2.8.1 Purpose / Scope

This section specifies the diagnostic requirements for the design of systems that utilize a gateway module(s). Diagnostic gateway modules translate diagnostic messages from one protocol to another to enable a tester to communicate to modules that only communicate on protocols not supported by a tester; see an example of a gateway topology in Figure 6.

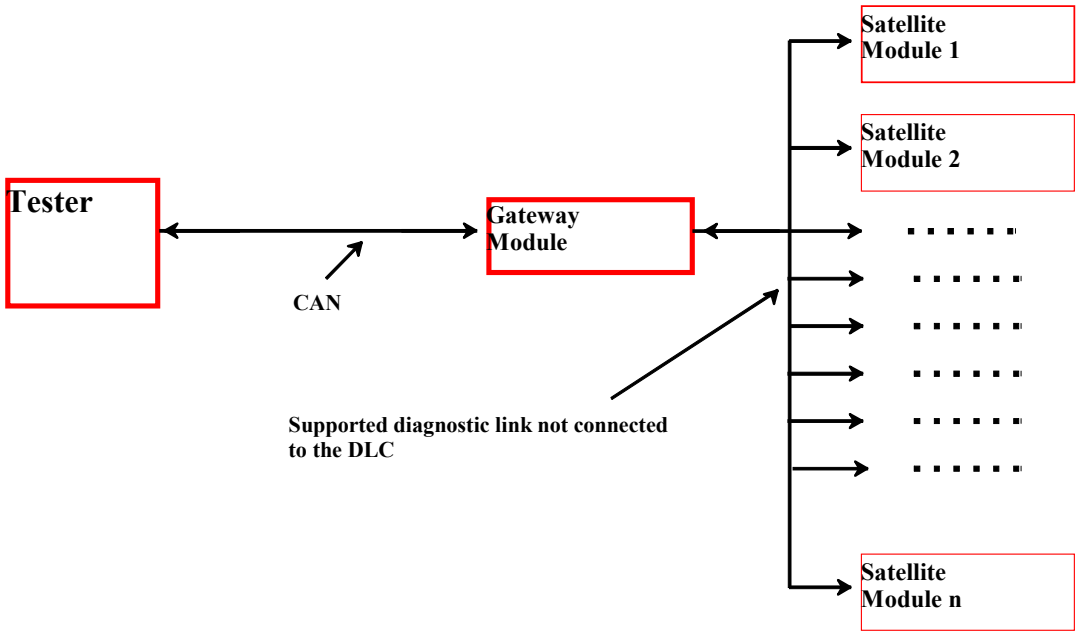


Figure 6: Example of Multi-Module Gateway System

2.2.2.8.2 Entering Gateway State

To enter the gateway state, the gateway module shall meet the following entry requirements:

- The gateway module shall be in a session that supports entry to gateway state.
- No subsystem specific condition exists that prevents the gateway module from entering the gateway state.
- The gateway module receives a correctly formatted diagnostic command $0300 message with valid data. Data Byte #4 of the $0300 command message shall contain the 1 Byte address of the satellite ECU and Data Byte #5 shall contain $00 to enter the gateway state of operation.

2.2.2.8.3 Gateway State Message Handling

Once a gateway module is in the gateway state, the following rules for tester-gateway-satellite communications apply.

All diagnostic request messages received from a tester, except for command $0300, shall be sent to the satellite module. The gateway module's satellite module protocol translation algorithm shall be capable of translating the collective set (equivalent to a logical OR) of all the messages supported by the satellite module. The gateway module shall also translate response messages received from a satellite module into the tester supported protocol and return those responses to the tester.

2.2.2.8.3.1 Messages: Tester to Gateway

- All request messages received by a gateway module from a tester that are formatted correctly, except for command $0300 "Gateway State Access," shall be translated and sent to the satellite module.
- If the gateway module receives command $0300 "Gateway State Access" with data byte 5 = $00, it shall return a Positive Response (service $F1) as shown in Table 21, and switch its pointer to the satellite module identified in data byte 4 of the command $0300 request message.

- If the gateway module receives command $0300 "Gateway State Access" with data byte 5 = $81, it shall return a Positive Response (service $F1) as shown in Table 21 and exit the gateway state to the default diagnostic mode.

NOTE: Upon returning to the default diagnostic session, the gateway module shall have 500 milliseconds to reinitialize. During this time period, the gateway module is not required to respond to requests received from a tester.

- If the gateway module receives command $0300 "Gateway State Access" with data byte 5 not equal to $00 or $81, it shall return a Negative Response (service $7F) of RequestOutOfRange (code $31).
- If the gateway module receives command $0300 "Gateway State Access" with data byte 4 containing a value that is not a recognized satellite module address, it shall return a Negative Response (service $7F) of RequestOutOfRange (code $31).
- If the gateway module receives a service ID that is not supported in the translation algorithm of the gateway module, it shall return a Negative Response (service $7F) of ServiceNotSupported (code $11).

2.2.2.8.3.2 Messages: Gateway to Satellite

Satellite modules that receive translated requests from a gateway module shall handle the various translated requests as follows:

- If the satellite supports the request, it shall formulate a proper response and return it to the gateway module.
- If the satellite receives a service that is not supported, it shall respond with a Negative Response (service $7F) of ServiceNotSupported (code $11) to the gateway module.
- If the satellite receives a data byte that is not valid, it shall respond with a Negative Response (service $7F) of RequestOutOfRange (code $31) to the gateway module.

2.2.2.8.3.3 Messages: Satellite to Gateway

Gateway modules shall process satellite module responses as follows:

- Correctly formatted satellite responses shall be translated to the communication protocol linking the gateway module and the tester.
- Gateway modules shall ignore a satellite's response if it is an invalid message.

2.2.2.8.4 Gateway State Timing Requirements

The response time between the trailing edge of the last bit of a request message issued by a tester and the leading edge of the response message returned to the tester shall not exceed 275 milliseconds; the tester shall wait 300 milliseconds before logging or indicating an error.

2.2.2.8.5 Satellite Modules

Satellite modules shall adhere to all requirements specified within this specification. Satellite modules are the indirect target of tester request messages when a gateway module is in the gateway state.

NOTE: All tester messages are sent to the gateway module address, not the satellite module address.

2.2.2.8.6 Satellite Module in a Diagnostic Session

All satellite modules shall implement a Lack of Diagnostic Dialog timer (see section 2.2.3.8).

2.2.2.8.7 Satellite Module Mandatory PIDs

All satellite modules shall support all of the (PID/common ID) requirements specified or referenced in this specification and also support (PID/common ID) $D12D "Node Address."

2.2.2.8.8 Redirecting Communication

To redirect communication to another satellite module when the gateway module is already in gateway state, the tester shall send command $0300 “Gateway State Access” message, with the new satellite module node ID specified in data byte 4 and data byte 5 = $00 for commanding the gateway module to either stay or enter into the gateway state.

2.2.2.9 CAN Diagnostic Specification Version CommonID/PID (ref. P1-v2003.0, 16.7)

In addition to the referenced mandatory CommonID/PIDs, a CAN diagnostic implementation version number shall reside within the ECU memory and this version shall be reported via the mandatory CAN version CommonID/PID $E6F3. CommonID/PID $E6F3 is state encoded with each value referencing a valid CAN diagnostic implementation. For the most up to date state encoded values, please refer to the Ford diagnostic MRDB.

Table 29: CAN Diagnostic Specification Version CommonID/PID $E6F3

State Encoded Value	Description
$0A	CAN Generic Diagnostic Specification (v2001.0)
$0B	CAN Generic Diagnostic Specification (v2001.1)
$0C	CAN Generic Diagnostic Specification (v2003.0)

2.2.3 Additional Requirements

The requirements in this section are specified for Ford Motor Company and are additional to the requirements referenced in other industry standard documents. These requirements are generic and apply to all ECUs that perform network diagnostics.

2.2.3.1 Diagnostic Services

The following sub-sections contain information on services that are not referenced from the KWP-GRP-1.5 or P1-v2003.0 but are supported for CAN diagnostic communications and services.

2.2.3.1.1 communicationControl (28 hex) service

This service is used to suspend the transmission of normal mode messages in an ECU (i.e., inter-module messages that do not use the diagnostic range of identifiers). Diagnostic messaging, including periodic positive diagnostic data packet responses (rapid data), shall not be affected by this service.

The Lack of Diagnostic Dialog timeout requirements from Section 2.2.3.8 shall apply to this service.

2.2.3.1.1.1 Request Message Definition

Table 30: communicationControl Request Service Message

Data Byte	Parameter Name	Cvt	Hex Value	Mnemonic
#1	**communicationControl Request Service Id**	**M**	**28**	**CC**
#2	sub-function = [responseRequired]	M	xx	LEV_ RESPREQ_
#3	ControlByte	M	xx	CB
#4	communicationType	M	xx	CTP

2.2.3.1.1.1.1 Request Message Sub-Function Parameter LEV_ Definition

The sub-function parameter responseRequired is used by the communicationControl request message to specify if the server shall send a response message; this parameter shall always have the value of $01 corresponding to the ECU being required to respond to the request.

2.2.3.1.1.1.2 Request Message Data Parameter Definitions

The controlByte parameter contains information on how the ECU shall modify the communication referenced in the communicationType parameter. The information contained in the controlByte parameter is bit oriented and allows for enabling/disabling of the transmission and/or the reception of certain messages of the server. The controlByte (CB) parameter shall use only the values and definitions in Table 31 to control whether to enable or disable communication transmissions in an ECU.

Table 31: controlByte (CB) Parameter Values

Bit number	Bit Definition	'0' Value Definition	'1' Value Definition
0	Control transmission of messages	Enable	Disable
1-7	Reserved by document	N/A	N/A

The communicationType parameter shall indicate which type of messages are either enabled or disabled by this service. Only the values specified in Table 32 shall be supported for the communicationType (CTP) parameter.

Table 32: communicationType (CTP) Parameter Values

Hex Value	Description	Man. or Opt.
01	**Application** This value references all application-related communication	M
02	**NetworkManagement** This value references all network management related communication	O

2.2.3.1.1.2 Positive Response Message Definition

Table 33: communicationControl Positive Response Message

Data Byte	Parameter Name	Cvt	Hex Value	Mnemonic
#1	**CommunicationControl Response Service Id**	**M**	**68**	**CCPR**

2.2.3.1.2 controlDTCSetting (85 hex) service

The controlDTCSetting service shall be used by an ECU to stop and start the setting of continuous DTCs.

2.2.3.1.2.1 Service Description

The controlDTCSetting request message can be used to stop the setting of diagnostic trouble codes in an individual ECU. The ECU being addressed shall respond with a controlDTCSetting positive response message or, if unable to stop the setting of diagnostic trouble codes, respond with a controlDTCSetting negative response message indicating the reason for the reject.

The Lack of Diagnostic Dialog timeout requirements from Section 2.2.3.8 shall apply to this service.

2.2.3.1.2.2 Request Message Definition

Table 34: controlDTCSetting Request Message Definition

Data Byte	Parameter Name	Cvt	Hex Value	Mnemonic
#1	**controlDTCSetting Request Service Id**	**M**	**85**	**CDTCS**
#2	sub-function = [DTCSettingMode]	M	xx	**LEV_ DTCSM_**

2.2.3.1.2.2.1 Request Message Sub-Function Parameter $Level (LEV_) Definition

The sub-function parameter DTCSettingMode is used by the controlDTCSetting request message to indicate to ECUs whether diagnostic trouble code setting shall be either enabled or disabled. The only values that shall be used are defined in Table 35.

Table 35: controlDTCSetting Sub-function Values

Hex	Description	Cvt	Mnemonic
01	**On** The ECU shall resume the setting of diagnostic trouble codes according to normal operating conditions.	M	ON
02	**Off** The server(s) shall stop the setting of diagnostic trouble codes.	M	OFF

2.2.3.1.2.3 Positive Response Message Definition

Table 36: controlDTCSetting Positive Response Message

Data Byte	Parameter Name	Cvt	Hex Value	Mnemonic
#1	**ControlDTCSetting Response Service Id**	**M**	**C5**	**CDTCSPR**
#2	DTCSettingMode	M	xx	**DTCSM_**

2.2.3.1.2.3.1 Positive Response Message Data Parameter Definition

Table 37: Response Message Data Parameter Definition

Definition
DTCSettingMode This parameter is an echo of the sub-function parameter from the request message.

2.2.3.1.3 requestCommonIDScalingMasking (24 hex) service

The requestCommonIDScalingMasking service shall be used to obtain which bits of a bit-mapped common ID are supported.

2.2.3.1.3.1 Service Description

The requestCommonIDScalingMasking request message can be used by a tester to obtain the specific bits of a bit-mapped common ID which are supported for an individual ECU. The ECU being addressed shall respond with a requestCommonIDScalingMasking positive response message or, if unable to perform the request, respond with a requestCommonIDScalingMasking negative response message indicating the reason for the reject.

2.2.3.1.3.2 Request Message Definition

Table 38: requestCommonIDScalingMasking Request Message Definition

Data Byte	Parameter Name	Cvt	Hex Value	Mnemonic
#1	**requestCommonIDScalingMasking Request Service Id**	**M**	**24**	**RCISM**
#2	commonIDOffset	M	xx	**CIOFF_**
#3	recordCommonIdentifier (High Byte)	M	xx	**RCI_**
#4	recordCommonIdentifier (Low Byte)	M	xx	

Only the value of $FF shall be supported for the commonIDOffset specifying that the offset is not applicable.

2.2.3.1.3.3 Positive Response Message Definition

Table 39: requestCommonIDScalingMasking Positive Response Message

Data Byte	Parameter Name	Cvt	Hex Value	Mnemonic
#1	**requestCommonIDScalingMasking Response Service Id**	**M**	**64**	**RCISMPR**
#2	commonIDOffset	M	xx	**CIOFF_**
#3	recordCommonIdentifier (High Byte)	M	xx	**RCI_**
#4	recordCommonIdentifier (Low Byte)	M	xx	
#5	scalingMaskingFormat	M	xx	**SMF_**
#6	scalingMaskingValue#1	M	xx	**SMV_**
	:	:	xx	**:**
#n	scalingMaskingValue#m	U	xx	**SMV_**

2.2.3.1.3.3.1 Positive Response Message Data Parameter Definition

Table 40: Response Message Data Parameter Definition

Definition
CommonIDOffset This parameter is an echo of the commonIDOffset parameter from the request message.
RecordCommonIdentifier This parameter is an echo of the recordCommonIdentifier parameter from the request message.

ScalingMaskingFormat
This parameter is used to specify the scaling/masking of the requested recordCommonIdentifier. The high order nibble defines the type of information encoding which is used to represent the parameter and the low order nibble can be read directly to determine the number of bytes used to represent the parameter.

Bits 7 – 4:	Bits 3 – 0:
0000 – 0001 Reserved by document	0000 Reserved by document
0010 Report bit mapped mask	0001 1 Byte
0011 – 1111 Reserved by document	0010 2 Bytes
	0011 3 Bytes
	0100 4 Bytes
	0101 – 1111 Reserved by document

ScalingMaskingValue
These values are used to return the scaling and masking information for a recordCommonIdentifier. In the case of bit-mapped records, scalingMaskingValue#1 will represent the validity (supported) bit mask for recordCommonIdentifier recordValue#1 as reported with readDataByCommonIdentifier positive response ($62), and scalingMaskingValue#2 will represent the validity (supported) bit mask for recordCommonIdentifier recordValue#2, etc.

2.2.3.2 Network Fault Strategy

CAN network faults consist of missing message, invalid message and bus off errors.

2.2.3.2.1 Missing Message Network Faults

The range of DTCs from $C100 - $C2FF is defined in the MRDB as Lost Communication with "Module X", where "Module X" is assigned a specific description for each DTC (e.g., DTC $C151 is defined as Lost Communication with Restraints Control Module). A DTC within this "Lost Communication" range shall be logged by an ECU for each message that it fails to receive that is expected from another ECU. Filtering shall be applied before setting each specific lost communication DTC to make sure that it is set only when customer noticeable problems **can** occur. Filtering shall be chosen so that each specific lost communication DTC shall be logged only after the message is missed for a certain timed duration or a number of missed intervals, and that a customer noticeable failure **can** occur from the missed message. The time and/or duration of the filter values that shall be applied to each network message shall be calculated by the ECU developer and listed in the ECU's Subsystem Specific Diagnostic Specification (Part 2). For example, if the vehicle speed from the PCM is missing for 5 seconds the speedometer on the cluster may display erratic readings; under these conditions, DTC $C100 (Lost Communication with ECM/PCM) shall be set.

An ECU shall not log a lost communication DTC while the vehicle ignition status is crank. Additionally, if an ECU can detect its supply voltage, then it shall not log a lost communication DTC when the supply voltage drops below 10 volts. Differences in ECU power-up times on a vehicle should be considered when determining the criteria for logging lost communication DTCs.

Only the lost communication DTCs defined in the range from $C100 - $C2FF shall be logged by an ECU to identify that a message(s) was missing.

2.2.3.2.2 Invalid Data Faults

Invalid data faults consist of the entire set of signal values that are defined as out of range in a CAN message. ECUs that receive invalid data in a CAN message shall not log DTCs for invalid data. The transmitter of invalid data shall be responsible for logging a code relating to the invalid data that it transmits to other ECUs.

NOTE: The receiver of invalid data can set DTCs triggered by invalid data for conditions such as the ECU needing to suspend some sort of functionality because the data received in a message was out of range. Invalid data faults shall not be set by the receiving message pointing to the faulty data in the message or the ECU sending the invalid data. The recommended generic DTC for the receiver, if necessary, is DTC $E023 "Fault Received from External Node."

2.2.3.2.3 **Bus Off Faults**

Bus off failures occur for a variety of reasons as specified in the HS/MS-CAN ECU and Subsystem Requirements Specification and HS/MS-CAN System Level and Distributed Applications Requirements Specification.

DTC $C073 "Control Module Communication Bus Off" may be set when bus off conditions arise. It is up to the system developer (D&R) to define and implement effective diagnostics associated with DTC $C073. For example, some ECUs may require that the ECU is bus off for a certain duration before DTC $C073 is logged. Also, DTC $C073 should not be logged unless a customer noticeable problem occurs from the bus off condition.

2.2.3.2.4 **Private Network Faults**

A public network shall be considered any communication bus that is brought out to the J1962 diagnostic connector. A private network shall be considered any communication bus that is not brought out to the J1962 diagnostic connector. An ECU on the public network that is responsible for logging communication DTCs against a private network shall utilize the range of DTCs ($C028 – $C072) as allocated in the MRDB. The DTC range consists of faults for five separate private links (Vehicle Communication Bus 'A' through Vehicle Communication Bus 'E'). Each vehicle communication bus DTC group consists of generic, performance, and electrical faults. The type of private link that each vehicle communication bus fault group refers to (e.g., Vehicle Communication Bus 'D') shall be ECU specific. If more than one ECU on the public network logs these communication DTCs against the same physical private network, they shall both use the same vehicle communication group (e.g., both ECUs use Vehicle Communication Bus 'A' faults or both ECUs use Vehicle Communication Bus 'B' faults).

2.2.3.3 Reporting of Logged DTCs

Criteria for logging DTCs in an ECU shall be determined by the module designer unless referenced or specified differently in this specification. All DTCs that are logged in an ECU shall be reported by the readDiagnosticTroubleCodesByStatus service.

2.2.3.4 Negative Response Messages (Service $7F)

The message data byte structure for negative response messages shall follow the structure specified in Table 41.

Table 41: Negative Response Data Byte Layout

Type	Parameter Name	Hex Value
Data Byte 1	negativeResponse Service Identifier	7F
Data Byte 2	<Service Name> Request Service Identifier	xx
Data Byte 3	responseCode=[KWP2000ResponseCode]	xx=[00-8F, 90-FF]

Refer to Annex A for all applicable negative response codes per diagnostic service.

2.2.3.5 Safety and Damage Prevention Requirements

All diagnostic operations shall have safety and damage protection associated with them. Any operation that can cause a safety hazard or damage the ECU or any of the vehicles system shall have protection associated with it.

For example, performing a service $2F to drive ABS coils too long can cause damage to the ABS system by burning the coils. For conditions such as this, the ABS module shall only allow the coils to be energized for a finite amount of time to avoid any damage to the ABS sub-system.

2.2.3.6 Powertrain KOEO Selftest $02

Execution routine $02 shall be designated "KOEO Self Test" for powertrain control systems and "On-Demand Self Test" for non-powertrain related systems. KOEO stands for Key On Engine Off and requires the engine to be in a specific pre-condition state before the test can be properly executed.

2.2.3.7 Intermessage Gap Requirements

Transactions are composed of messages that are sequentially transmitted. Specification of intermessage response time sets the maximum and minimum transaction time with a 1 millisecond tolerance. The period between the completion of the tester request message and the start of the next message is defined as follows:

ECU Response Following A Tester Request:

$$0 \leq \textbf{Intermessage Gap Time} \leq \textbf{50 milliseconds}$$

This condition occurs when the ECU responds with a single response message consisting of one or more frames to a tester query/request. The tester must be able to receive frames back-to-back for an indefinite period. If the initial ECU response is not returned within 75 milliseconds (150% of 50ms), the tester may time out and shall assume that no response will be sent. Refer to section 2.3.2.2 for detailed performance and timeout requirements while transmitting or receiving segmented messages.

If an ECU, after receiving a tester request, receives one or more additional tester requests before the response to the initial tester request is transmitted on the bus, the ECU shall behave as follows:

1) If the second request is received within 15ms of the first request the ECU may either:
   a) Ignore the 2nd tester request, and all additional tester requests until the response to the initial tester request is processed and transmitted.
   b) Process all tester requests and send a response to each. Note that the requests shall be responded to in the order they were received.
   c) Disregard the 1st tester request and only process the second tester request
2) If the second request is received after 15ms of the first request the ECU may either:
   a) Ignore the 2nd tester request, and all additional tester requests until the response to the initial tester request is processed and transmitted.
   b) Process all tester requests and send a response to each. Note that the requests shall be responded to in the order they were received.

An ECU shall never perform actions based upon a tester request without giving an appropriate response, unless a sub-function of response required is supported for the request and the sub-function is equal to “No”.

ECU Response Following another ECU Response in a Sequence:

$$0 \leq \textbf{Intermessage Gap Time} \leq \textbf{50 milliseconds}$$

This condition is not common but occurs when the ECU responds with multiple response messages to a single tester query/request (e.g., the multiple positive $E0 responses to a service $A0 request with a data rate of $01 for more than one DPID). If each of the messages is not returned within 75 milliseconds (150% of 50ms) of the preceding message, the tester may time out and shall assume that no more messages will be sent. The time-out timer for the 75 milliseconds (150% of 50ms) time-out period shall be reset after each message in the sequence from the ECU is received.

2.2.3.8 Timed Operation Timeout

Specific operations in the ECU may be timed, depending on the operation's function. Every ECU shall support the **Lack of Diagnostic Dialog** timeout timer for each timed operation. In addition to the **Lack of Diagnostic Dialog** timeout, a **Damage and Safety Timeout** may be supported for an operation if it the operation needs to timeout before the **Lack of Diagnostic Dialog** timeout. If either

of the two following timers expires, then normal ECU control of the operation that was timed out shall resume.

1. **Damage and Safety Timeout:** An ECU needs to suspend the operation of certain timed operations to avoid damage to the ECU or to avoid compromising the safety of any system.

2. **Lack of Diagnostic Dialog:** A 5 second timer shall be supported and reset each time the ECU receives a diagnostic request from a tester. The ECU shall suspend all diagnostic service functionality (see Table 42) upon expiration of the timer. Additionally, if the ECU is not already in the default diagnostic session ($81) when the timer expires, the ECU shall return to this default session. When receiving a segmented message, the timer shall be suspended when a first frame indication occurs in the ECU and shall be reset whenever a successful reception of a complete message occurs or an error is received from the network layer while receiving the message. When transmitting a segmented message, the timer shall be suspended when the diagnostic message is passed to the network layer for transmission, and shall resume when the network layer indicates the successful transmission of the complete message or indicates any error while transmitting the message. Note that if the Lack of Diagnostic Dialog timer expires when the default diagnostic session ($81) is active and timed service $2F (inputOutputControlByCommonID), $31 (startRoutineByLocalID), or $B1 (diagnosticCommand) has been successfully performed during this same active diagnostic session, the ECU is allowed to perform a reset to ensure a successful continuation of normal operation.

A **Damage and Safety** timer shall be set up for each service that needs to automatically time out before the **Lack of Diagnostic Dialog** timer expires (e.g., certain diagnosticCommands). The value of each **Damage and Safety** timer that is allocated to any diagnostic operation shall be documented in the ECU's SSDS. The value of the **Damage and Safety** timeout can vary depending on the operation that is initiated by a tester. For example, a specific diagnosticCommand that drives ABS coils may need to be de-energized before the coil burns up.

See Table 42 for the services that shall support the **Lack of Diagnostic Dialog** timeout timer.

Table 42: Services that Support the Lack of Diagnostic Dialog Timer

Hex Value	Diagnostic Services
10	startDiagnosticSession
27	securityAccess
28	communicationControl
2F	InputOutputCtrlByCommonID.
31	startRoutineByLocalIdentifier
85	controlDTCSetting
A0	requestDiagnosticDataPacket
B1	diagnosticCommand

If vehicle speed is available to an ECU, it may suspend all timed diagnostic functionality (see Table 42) and transition to the default diagnostic session when vehicle speed exceeds a value specified in the Subsystem Specific Diagnostic Specification. This shall only be utilized in situations where safety is the primary need for the timeout.

2.2.3.9 Memory Modification and Output Control

WriteMemoryByAddress, writeDataByLocalID and writeDataByCommonID shall be used for writing to memory that shall not be automatically modified by the ECU's software, these services shall not directly modify or control any ECU output signal in any way.

InputOutputControlByCommonIdentifier shall be used for temporarily bypassing inputs (e.g., real world sensors) and for directly controlling output devices only. The substituted value shall be used

only for the duration of this timed service (see section 2.2.3.8). When the ECU is returned to normal operation or the data value control is returned to the ECU, the substituted value shall revert back to the normal value as determined by the control system.

2.2.3.10 Local ID Differentiation

Local IDs are used for both execution routines and logical memory referencing and used, for example, with services $31 and $21 respectively. The local IDs associated specifically for services $31 and $21 are not the same and shall be treated as separate entities. Local IDs associated with service $31, $32, and $33 are execution routine local IDs and are specified and stored in the MRDB. Local IDs associated with service $21 and $3B are used to reference memory in ECUs by using a logical number that gets translated to a physical address in each ECU that uses them. These local IDs are typically used for configuration (reference MC-v2003.0 for additional details). Any use of the local IDs associated with service $21 and $3B for purposes other than configuration shall not interfere with the assignment of local IDs as described in MC-v2003.0.

2.2.3.11 Diagnostic Mode (PID/common ID)

(PID/common ID) $D100 shall be supported with the states as defined in Table 43.

Table 43: Mandatory (PID/common ID) $D100 States

State (HEX)	Description
81	Standard Diagnostic Mode
85	ECU Programming Mode
87	ECU Adjustment Mode

2.2.3.12 Module Configuration and Programming

Any data that is necessary to be configured or programmed into an ECU by Ford test tools shall follow one of the methods as defined in the Module Programming and Configuration Design Specification (MC-v2003.0). The use of any method not detailed in MC-v2003.0 will not be supported for configuration or programming by Ford tools.

2.2.3.13 Special Procedures and Requirements

Special functions, such as ABS service bleed, IVD calibration, etc., shall be documented in the ECU's SSDS. Additionally, any diagnostic commands, command timing, status PIDs, etc. used to monitor the failure or completion of the procedure and any instructions to be displayed on the service tool shall be clearly documented in the ECU's SSDS.

Prerequisite conditions for performing any diagnostic service shall never include involvement from the diagnostic tester, with the exception of transitioning to the required diagnostic session (service $10), performing security access (service $27), or configuring an upload (service $35) or download (service $34). Any other prerequisite conditions requiring tester involvement shall require explicit approval by the appropriate Network Communications Engineer as well as FCSD / DSP and VO Test Strategies.

Additionally, any special ECU requirements which necessitate specific tester support over and beyond what is documented in this specification (e.g., a requirement for unique and/or special tester involvement in order to maintain a timed diagnostic service in addition to the requirements detailed in section 2.2.3.8) shall require explicit approval by the appropriate Network Communications Engineer as well as FCSD / DSP and VO Test Strategies. If approved, these requirements shall be clearly documented in the ECU's SSDS.

2.2.3.14 Powertrain Specific Requirements

All powertrain control module outputs require individual diagnostic tool control. Additionally, parameters shall be provided to enable monitoring of each physical input and output.

2.2.3.15 FNOS Mandatory PIDs

In addition to the referenced mandatory CommonID/PIDs, the FNOS related PIDs as detailed in Table 44 shall be mandatory if the corresponding FNOS component is utilized by the ECU.

Table 44: FNOS CommonIDs/PIDs

CommonID/PID	Description
$E400	FNOS CAN Driver Version Number
$E401	FNOS OSEK NM Version Number
$E402	FNOS NM Junior/Node Management Version Number
$E403	FNOS Interaction Layer Version Number
$E404	FNOS Network Initialization Version Number
$E405	FNOS Transport Layer Version Number
$E406	FNOS Diagnostics Version Number
$E407	FNOS Generation Tool Version Number
$E408	FNOS Bootloader Version Number
$E409	FNOS Database Version Number

2.2.3.16 Inter-module Communication Gateway

Any ECU functioning as a gateway for application related inter-module communication messages between the public HS-CAN and public MS-CAN network shall support full diagnostics on both networks, but not simultaneously.

2.3 NETWORK LAYER

2.3.1 Referenced Requirements

All Ford CAN diagnostic Network Layer implementations shall comply with requirements referenced below.

Table 45: Diagnostic Network Layer Referenced Requirements

Item	Document	Section	Subsections
1	ISO 15765-2	1. Scope	All
2	ISO 15765-2	2. Normative references	All
3	ISO 15765-2	3. Term(s) and definition(s)	All
4	ISO 15765-2	4. Symbols (and abbreviated terms)	All
5	ISO 15765-2	5. Network Layer Overview	All
6	ISO 15765-2	7. Network layer protocol	All

2.3.1.1 Addressing Format (ref ISO 15765-2, 5.2, 7.3.5.1, 7.4.1, 7.4.2, 8.3)

All CAN diagnostic implementations shall use only the normal addressing formats referenced. Extended and mixed addressing shall not be supported or designed into any Ford ECUs.

2.3.1.2 Single Frame Transmission (ref ISO 15765-2, 7.1, 7.2)

All CAN single frame messages shall have an available data payload of seven bytes associated with normal addressing.

2.3.1.3 Definition of BlockSize (BS) Parameter (ref ISO 15765-2, 7.4.4.1)

All flow control frames transmitted from a tester shall have a BlockSize of 0 specified to allow the maximum number of blocks to be sent to the tester from the ECU. An ECU shall assume, upon receiving a validly formatted flow control from the tester, that the BlockSize parameter is zero without further validation.

It is strongly recommended that all flow control frames transmitted from an ECU should have a BlockSize value of 0 to allow the maximum number of blocks to be sent to the ECU from the tester.

2.3.1.4 Definition of SeparationTime (STmin) Parameter (ref ISO 15765-2, 7.4.4.2)

All flow control frames transmitted from a tester shall have a SeparationTime of 0 specified to allow back to back frames to the tester from the ECU. An ECU shall assume, upon receiving a validly formatted flow control from the tester, that the SeparationTime parameter is zero without further validation.

It is strongly recommended that all flow control frames transmitted from an ECU should have the following SeparationTime values:

- STmin ≤ 5ms while not in the bootloader
- STmin = 0 while in the bootloader

2.3.1.5 Half and Full Duplex (ref ISO 15765-2, 7.6.2.2)

Only half-duplex shall be supported.

2.3.1.6 Maximum number of Flow Control Wait frame transmissions (ref ISO 15765-2, 7.5)

The N_WFTmax parameter for any tester shall be zero, signifying that the tester shall only use flow control frames with a FlowStatus of 0 (ContinueToSend).

The N_WFTmax parameter for an ECU shall be less than or equal to 50, signifying that the tester shall wait this maximum number of times before aborting message transmission. However, it is strongly recommended that an ECU only transmit flow control frames with a FlowStatus of 0 (ContinueToSend).

2.3.2 Additional Requirements

2.3.2.1 CAN Diagnostic Messages

In all cases, diagnostic frames shall be full-length. The Data Length Code (DLC) parameter that is associated with every CAN frame shall always be 8 for diagnostic CAN frames. Each frame shall be full, consisting of 8 data bytes that can be either transmitted data or pad bytes. Any CAN diagnostic frame with a DLC other than 8 shall be considered invalidly formatted and ignored by the recipient (i.e., no response shall be elicited and no action shall be taken). In the event that the relevant diagnostic information does not completely fill a CAN frame, the remaining bytes within the frame shall be padded with zero ($00). An ECU shall not reject a received diagnostic request with non-zero pad bytes however.

Note that the term message in CAN refers to the entire set of diagnostic information, whereas frames are the individual units transmitted on CAN bus. For example, if an ECU was to transmit 10 DTCs, the entire set of 10 DTCs with service byte constitutes the message. However, to transmit this data on the CAN bus, the message is broken into frames.

2.3.2.2 CAN Diagnostic Messages Error Handling

2.3.2.2.1 Performance Parameter Definitions (ref ISO 15765-2, 7.6)

Table 46: Segmented Message Performance Parameter Definitions

Parameter	Timeout Value	Performance Requirement Value
N_Bs	75 ms – 120 ms	-
N_Ar + N_Br	-	< 50 ms
N_As + N_Cs	-	< 50 ms
N_Cr	75 ms – 120 ms	-

2.3.2.2.2 Tester to ECU Segmented Message Error Handling

Each segmented message that is initiated by the tester shall have the following error handling associated with it:

Table 47: Tester to ECU Segmented Error Handling

Error Condition	Error Handling
N_Ar + N_Br parameter exceeded	Product development shall fail the ECU during testing and fix the problem.
N_Bs timeout exceeded	Test equipment shall assume the ECU will not respond to the request. The tester can initiate other requests.
N_Cr timeout exceeded	ECU shall send a negative response to the first frame with response code of $72 "transferAborted".

2.3.2.2.3 ECU to tester Segmented Message Error Handling

Each segmented message that is initiated by the ECU shall have the following error handling associated with it:

Table 48: ECU to Tester Segmented Error Handling

Error Condition	Error Handling
N_Bs timeout exceeded	The ECU shall timeout and terminate the remainder of the message.
N_As + N_Cs parameter exceeded	Product development shall fail the ECU during testing and fix the problem.
N_Cr timeout exceeded	Test equipment shall assume the ECU will not send any more frames within this message. The tester can initiate other requests.

2.3.2.3 CAN Diagnostic Message ID

Because each ECU (including the tester) must transmit a unique CAN frame, each ECU shall support two (2) CAN message IDs—one for receiving diagnostic requests and one for transmitting diagnostic responses. The N_AI parameter referenced in Section 7.3.5 of the ISO 15765-2 document shall consist of the 11-bit physical diagnostic IDs defined in this section.

The 11-bit CAN IDs $700 - $7FF shall be reserved for diagnostics. Furthermore, within this range, the range $7E0 - $7EF shall be reserved for emissions related diagnostics, and the range $7F0 - $7FF shall not be used. Table 49 provides the list of reserved diagnostic ID pairs.

Table 49: Diagnostic Request / Response Message IDs

ECU Reception ID (Tester Transmit Request)	ECU Transmission ID (Tester Receive Response)
$720	$728
$721	$729
• • •	• • •
$7C5	$7CD
$7C6	$7CE
$7C7	$7CF
$7E0	$7E8
$7E1	$7E9
• • •	• • •
$7E6	$7EE
$7E7	$7EF

For example, to place an ECU into a different diagnostic session, the tester would transmit the request message using CAN ID $7E0 and the ECU would respond on ID $7E8. Officially designated IDs are stored in the corporate MRDB and listed in the vehicle specific EESS.

CAN "Rapid Data" IDs, as defined in the corporate MRDB, shall be used for all periodic $E0 DPID responses in place of the ECU transmission ID. Please reference section 2.2.2.7.23.4 for further details.

2.4 Data Link Layer

2.4.1 Referenced Requirements

All Ford CAN Data Link Layer implementations shall comply with requirements referenced below.

Table 50: Diagnostic Data Link Layer Referenced Requirements

Item	Document	Section	Subsections
1	**ISO 15765-2**	**8. Data link layer usage**	**All, except 8.3.1.1, 8.3.2 and 8.3.3**

Annex A

A.1 Service Response Code Tables

This annex contains tables of every service available for use in this document. Each table lists all of the allowable response codes associated with each service. No additional response codes shall be applied to any service if they are not shown in each service's table except for the general ones that apply to all services.

A.2 General Response Codes

General response codes may be used for any service where necessary. As an example, the general response code of busy ($21) shall be supported by an ECU in response to a request to obtain routine results while the execution routine is running, but may never be used by an ECU in response to a CommonID (PID) request.

The general response code of busy-repeatRequest ($21) is returned by the ECU if the ECU is temporarily too busy to perform the requested operation. The response code indicates that the requested action was not performed, but that a later repetition of the request will eventually result in a positive response. In this circumstance the tester may perform repetition of the "identical request message" or "another request message". The length of time an ECU may respond with a negative response code of $21 to a tester request shall always be finite.

The general response code of reqCorrectlyRevd-RspPending ($78) is returned by the ECU if the time required to complete the request takes longer than 50ms. See Section 2.2.2.6 for more information related to response code $78. General response code $78 shall **only** be used when necessary and **only** under the following conditions:

- Long duration memory writes when responding to diagnostic service $14, $2E, $3D, $3B, or $36
- Checksum calculations and/or application validation when responding to diagnostic service $11 or $37
- Erasure of flash memory when responding to diagnostic service $B1 with a CommandCommonID of $00B2 (Flash Memory Erase)

In all other cases other than described above, an ECU is **not** allowed to respond with a negative response code of $78 unless explicit approval is obtained by EESE Core Network Communications.

Response code $72 handling shall be performed as specified in Section 2.3.2.2.2 for segmented messages initiated by the test equipment. In addition, response code $72 shall be used when an ECU receives a consecutive frame from a tester with a sequence number not in the correct order.

If an ECU receives a first frame from a tester with a first frame data length (FF_DL) less than eight, the ECU shall respond with a negative response code of illegalByteCountInBlockTransfer ($75).

A.2.1 ($10) startDiagnosticSession

Hex	Description
11	**ServiceNotSupported** This service is not supported by the ECU under any circumstances
12	**SubFunctionNotSupported-InvalidFormat** 1. The length of the message is wrong 2. All required data byte(s) were not supplied 3. Extra data byte(s) were supplied 4. The requested session is not supported by the ECU
22	**ConditionsNotCorrect** The criteria for entering the requested session was not met

A.2.2 ($11) ECUReset

Hex	Description
11	**ServiceNotSupported** This service is not supported by the ECU under any circumstances
12	**SubFunctionNotSupported-InvalidFormat** 1. The length of the message is wrong 2. All required data byte(s) were not supplied 3. Extra data byte(s) were supplied 4. The requested resetMode parameter is not supported by the ECU
22	**ConditionsNotCorrect** 1. Some vehicle conditions exist that prohibit this service from being executed 2. Not supported in current diagnostic session

A.2.3 ($12) readFreezeFrameData

Hex	Description
11	**ServiceNotSupported** This service is not supported by the ECU under any circumstances
12	**SubFunctionNotSupported-InvalidFormat** 1. The length of the message is wrong 2. All required data byte(s) were not supplied 3. Extra data byte(s) were supplied
22	**ConditionsNotCorrect** Not supported in current diagnostic session
31	**RequestOutOfRange** 1. The freezeFrameNumber is not supported 2. The recordAccessMethodIdentifier is not supported 3. The DTC number (RI_) either is not supported by the ECU or does not have freeze frame data associated with it
33	**SecurityAccessDenied-securityAccessRequested** The module has not been unlocked and needs to be in order for this request to work

A.2.4 ($14) clearDiagnosticInformation

Hex	Description
12	**SubFunctionNotSupported-InvalidFormat** 1. The length of the message is wrong 2. All required data byte(s) were not supplied 3. Extra data byte(s) were supplied
22	**ConditionsNotCorrect** 1. Internal conditions in the ECU prevent clearing of DTCs 2. Not in the correct diagnostic session
31	**RequestOutOfRange** The requested groupOfDiagnosticInformation parameter is not supported by the ECU

A.2.5 ($18) readDTCByStatus

Hex	Description
12	**SubFunctionNotSupported-InvalidFormat** 1. The length of the message is wrong 2. All required data byte(s) were not supplied 3. Extra data byte(s) were supplied 4. The requested statusOfDTC parameter is not supported by the ECU
22	**ConditionsNotCorrect** Not in the correct diagnostic session
31	**RequestOutOfRange** The requested groupOfDTC parameter is not supported by the ECU

A.2.6 ($21) readDataByLocalID

Hex	Description
11	**ServiceNotSupported** This service is not supported by the ECU under any circumstances
12	**SubFunctionNotSupported-InvalidFormat** 1. The length of the message is wrong 2. All required data byte(s) were not supplied 3. Extra data byte(s) were supplied
22	**ConditionsNotCorrect** Not in the correct diagnostic session
31	**RequestOutOfRange** The requested local ID is not supported
33	**SecurityAccessDenied-securityAccessRequested** The module has not been unlocked and needs to be in order for this request to work

A.2.7 ($22) readDataByCommonID

Hex	Description
12	**SubFunctionNotSupported-InvalidFormat** 1. The length of the message is wrong 2. All required data byte(s) were not supplied 3. Extra data byte(s) were supplied
22	**ConditionsNotCorrect** The requested item is a fault PID that is not currently active
31	**RequestOutOfRange** The requested common ID is not supported
33	**SecurityAccessDenied-securityAccessRequested** The module has not been unlocked and needs to be in order for this request to work

A.2.8 ($23) readMemoryByAddress

Hex	Description
11	**ServiceNotSupported** This service is not supported by the ECU under any circumstances
12	**SubFunctionNotSupported-InvalidFormat** 1. The length of the message is wrong 2. All required data byte(s) were not supplied 3. Extra data byte(s) were supplied
22	**ConditionsNotCorrect** Not in the correct diagnostic session
31	**RequestOutOfRange** The requested address or length to read from is not supported
33	**SecurityAccessDenied-securityAccessRequested** The module has not been unlocked and needs to be in order for this request to work

A.2.9 ($24) requestCommonIDScalingMasking

Hex	Description
11	**ServiceNotSupported** This service is not supported by the ECU under any circumstances
12	**SubFunctionNotSupported-InvalidFormat** 1. The length of the message is wrong 2. All required data byte(s) were not supplied 3. Extra data byte(s) were supplied 4. The commonIDOffset is not equal to $FF
22	**ConditionsNotCorrect** Not in the correct diagnostic session
31	**RequestOutOfRange** 1. The requested common ID is not supported 2. The requested common ID does not have scaling/masking information
33	**SecurityAccessDenied-securityAccessRequested** The module has not been unlocked and needs to be in order for this request to work

A.2.10 ($27) securityAccess

Hex	Description
11	**ServiceNotSupported** This service is not supported by the ECU under any circumstances
12	**SubFunctionNotSupported-InvalidFormat** 1. The length of the message is wrong 2. All required data byte(s) were not supplied 3. Extra data byte(s) were supplied
22	**ConditionsNotCorrect** 1. Not in the correct diagnostic session 2. A 'sendkey' parameter is received by the ECU without a valid or active seed present for that security level.
31	**RequestOutOfRange** The security level requested is not supported by the ECU
35	**InvalidKey** The key is incorrect

A.2.11 ($28) communicationControl

Hex	Description
12	**SubFunctionNotSupported-InvalidFormat** 1. The length of the message is wrong 2. All required data byte(s) were not supplied 3. Extra data byte(s) were supplied 4. The requested responseRequired parameter is not supported by the ECU
22	**ConditionsNotCorrect** 1. Not in the correct diagnostic session 2. Some vehicle conditions exist that prohibit this service from being executed
31	**RequestOutOfRange** 1. The controlByte parameter is out of range 2. The communicationType parameter is out of range
33	**SecurityAccessDenied-securityAccessRequested** The module has not been unlocked and needs to be in order for this request to work

A.2.12 ($2E) writeDataByCommonId

Hex	Description
11	**ServiceNotSupported** This service is not supported by the ECU under any circumstances
12	**SubFunctionNotSupported-InvalidFormat** 1. The length of the message is wrong 2. All required data byte(s) were not supplied 3. Extra data byte(s) were supplied
22	**ConditionsNotCorrect** Not in the correct diagnostic session
31	**RequestOutOfRange** 1. The requested common ID is not supported 2. Any data transmitted in the request message after the common ID is invalid
33	**SecurityAccessDenied-securityAccessRequested** The module has not been unlocked and needs to be in order for this request to work

A.2.13 ($2F) inputOutputCtrlByCommonId

Hex	Description
11	**ServiceNotSupported** This service is not supported by the ECU under any circumstances
12	**SubFunctionNotSupported-InvalidFormat** 1. The length of the message is wrong 2. All required data byte(s) were not supplied 3. Extra data byte(s) were supplied
22	**ConditionsNotCorrect** 1. Not in the correct diagnostic session 2. All prerequisites for input/output control of the CommonID are not satisfied.
31	**RequestOutOfRange** 1. The requested common ID is not supported for I/O control 2. The requested inputOutputControlParameter is not supported 3. Any data transmitted in the controlState parameter is invalid
33	**SecurityAccessDenied-securityAccessRequested** The module has not been unlocked and needs to be in order for this request to work

A.2.14 ($31) startRoutineByLocalIdentifier

Hex	Description
12	**SubFunctionNotSupported-InvalidFormat** 1. The length of the message is wrong 2. All required data byte(s) were not supplied 3. Extra data byte(s) were supplied
22	**ConditionsNotCorrect** 1. Not in the correct diagnostic session 2. All routine entry requirements are not satisfied
31	**RequestOutOfRange** 1. The requested local ID is not supported 2. The requested routineEntryOption parameter is not supported
33	**SecurityAccessDenied-securityAccessRequested** The module has not been unlocked and needs to be in order for this request to work

A.2.15 ($32) stopRoutineByLocalIdentifier

Hex	Description
12	**SubFunctionNotSupported-InvalidFormat** 1. The length of the message is wrong 2. All required data byte(s) were not supplied 3. Extra data byte(s) were supplied
22	**ConditionsNotCorrect** 1. An execution routine can be executed in this session but is not currently running 2. Not in the correct diagnostic session
31	**RequestOutOfRange** 1. This local ID is not supported 2. The RETO_ parameter is not supported by the ECU

A.2.16 ($33) requestRoutineResultsByLocalID

Hex	Description
12	**SubFunctionNotSupported-InvalidFormat** 1. The length of the message is wrong 2. All required data byte(s) were not supplied 3. Extra data byte(s) were supplied
22	**ConditionsNotCorrect** 1. The execution routine is supported but has not been ran in the current diagnostic session 2. Not in the correct diagnostic session
31	**RequestOutOfRange** This local ID is not supported
33	**SecurityAccessDenied-securityAccessRequested** The module has not been unlocked and needs to be in order for this request to work

A.2.17 ($34) requestDownload

Hex	Description
10	**GeneralReject** Some internal condition is preventing a download from occurring (e.g., flash memory is not blank)
11	**ServiceNotSupported** This service is not supported by the ECU under any circumstances
12	**SubFunctionNotSupported-InvalidFormat** 1. The length of the message is wrong 2. All required data byte(s) were not supplied 3. Extra data byte(s) were supplied
22	**ConditionsNotCorrect** Not in the correct diagnostic session
31	**RequestOutOfRange** The address, length, or dataFormatIdentifier is not correct
33	**SecurityAccessDenied-securityAccessRequested** The module has not been unlocked and needs to be in order for this request to work

A.2.18 ($35) requestUpload

Hex	Description
11	**ServiceNotSupported** This service is not supported by the ECU under any circumstances
12	**SubFunctionNotSupported-InvalidFormat** 1. The length of the message is wrong 2. All required data byte(s) were not supplied 3. Extra data byte(s) were supplied
22	**ConditionsNotCorrect** Not in the correct diagnostic session
31	**RequestOutOfRange** The address, length, or dataFormatIdentifier is not correct
33	**SecurityAccessDenied-securityAccessRequested** The module has not been unlocked and needs to be in order for this request to work

A.2.19 ($36) transferData

Hex	Description
11	**ServiceNotSupported** This service is not supported by the ECU under any circumstances
12	**SubFunctionNotSupported-InvalidFormat** 1. The length of the message is wrong 2. All required data byte(s) were not supplied 3. Extra data byte(s) were supplied
22	**ConditionsNotCorrect** 1. Not in the correct diagnostic session 2. Request not preceded by a valid service $34 or service $35 request
72	**TransferAborted** 1. An error occurred during the transfer 2. The transfer was interrupted by the tester 3. General response code $72 handling (see Section A.2)

A.2.20 ($37) requestTransferExit

Hex	Description
11	**ServiceNotSupported** This service is not supported by the ECU under any circumstances
12	**SubFunctionNotSupported-InvalidFormat** 1. The length of the message is wrong 2. Extra data byte(s) were supplied
22	**ConditionsNotCorrect** 1. Not in the correct diagnostic session 2. A download or upload was not performed in the current diagnostic session
72	**TransferAborted** 1. An error occurred during the transfer 2. The transfer was interrupted by the tester 3. General response code $72 handling (see Section A.2)
79	**IncorrectByteCountDuringBlockTransfer** The number of bytes sent did not match the number expected.

A.2.21 ($3B) writeDataByLocalId

Hex	Description
10	**GeneralReject** Some internal error occurred (e.g., battery voltage too low)
11	**ServiceNotSupported** This service is not supported by the ECU under any circumstances
12	**SubFunctionNotSupported-InvalidFormat** 1. The length of the message is wrong 2. All required data byte(s) were not supplied 3. Extra data byte(s) were supplied
22	**ConditionsNotCorrect** Not in the correct diagnostic session
31	**RequestOutOfRange** 1. This local ID is not supported 2. Any data transmitted in the request message after the local ID is invalid
33	**SecurityAccessDenied-securityAccessRequested** The module has not been unlocked and needs to be in order for this request to work

A.2.22 ($3D) writeMemoryByAddress

Hex	Description
11	**ServiceNotSupported** This service is not supported by the ECU under any circumstances
12	**SubFunctionNotSupported-InvalidFormat** 1. The length of the message is wrong 2. All required data byte(s) were not supplied 3. Extra data byte(s) were supplied
22	**ConditionsNotCorrect** Not in the correct diagnostic session
31	**RequestOutOfRange** This address or length to write to is not supported
33	**SecurityAccessDenied-securityAccessRequested** The module has not been unlocked and needs to be in order for this request to work

A.2.23 ($3E) testerPresent

Hex	Description
12	**SubFunctionNotSupported-InvalidFormat** 1. The length of the message is wrong 2. Extra data byte(s) were supplied 3. The requested responseRequired parameter is not supported by the ECU

A.2.24 ($85) controlDTCSetting

Hex	Description
12	**SubFunctionNotSupported-InvalidFormat** 1. The length of the message is wrong 2. All required data byte(s) were not supplied 3. Extra data byte(s) were supplied 4. The DTCSetting parameter is not supported by the ECU
22	**ConditionsNotCorrect** Not in the correct diagnostic session
33	**SecurityAccessDenied-securityAccessRequested** The module has not been unlocked and needs to be in order for this request to work

A.2.25 ($A0) requestDiagnosticDataPacket(s)

Hex	Description
11	**ServiceNotSupported** This service is not supported by the ECU under any circumstances
12	**SubFunctionNotSupported-InvalidFormat** 1. The length of the message is wrong 2. All required data byte(s) were not supplied 3. Extra data byte(s) were supplied 4. The requested data rate is not supported by the ECU
22	**ConditionsNotCorrect** 1. Not in the correct diagnostic session 2. The requested data rate is different from the data rate of the DPIDs that are currently being reported
31	**RequestOutOfRange** DPID identification number is not supported or defined
33	**SecurityAccessDenied-securityAccessRequested** The module has not been unlocked and needs to be in order for this request to work

A.2.26 ($A1) dynamicallyDefineDataPacket

Hex	Description
11	**ServiceNotSupported** This service is not supported by the ECU under any circumstances
12	**SubFunctionNotSupported-InvalidFormat** 1. The length of the message is wrong 2. All required data byte(s) were not supplied 3. Extra data byte(s) were supplied
22	**ConditionsNotCorrect** 1. Not in the correct diagnostic session 2. DPID identification number is currently being periodically reported.
31	**RequestOutOfRange** 1. DPID identification number is not supported or statically defined 2. DPID format is not supported 3. Reference number is out of range
33	**SecurityAccessDenied-securityAccessRequested** The module has not been unlocked and needs to be in order for this request to work

A.2.27 ($B1) diagnosticCommand

Hex	Description
10	**GeneralReject** Some internal error occurred
11	**ServiceNotSupported** This service is not supported by the ECU under any circumstances
12	**SubFunctionNotSupported-InvalidFormat** 1. The length of the message is wrong 2. All required data byte(s) were not supplied 3. Extra data byte(s) were supplied
22	**ConditionsNotCorrect** 1. Not in the correct diagnostic session 2. The necessary code to complete this command functionality has not been downloaded yet (e.g., the secondary bootloader is not present when a request is received to erase flash memory). 3. All prerequisites for the execution of the diagnosticCommand are not satisfied
31	**RequestOutOfRange** 1. The requested Command Common Identifier is not supported by the ECU 2. The requested Command Parameter is not supported by the ECU
33	**SecurityAccessDenied-securityAccessRequested** The module has not been unlocked and needs to be in order for this request to work