

ITS Data Bus—Application Message Layer

TABLE OF CONTENTS

1.	Scope	5
2.	References	5
2.1	Applicable Publications	5
2.1.1	SAE Publications	5
2.2	Related Publications	5
2.2.1	SAE Publications	5
2.2.2	ISO Publications	6
3.	Definitions	6
3.1	AHdr (Application Header)	6
3.2	ALP (Application Layer Primitives)	6
3.3	AMI-C (Automotive Multimedia Interface Collaboration)	6
3.4	APDU (Application Protocol Data Unit)	6
3.5	Audio Arbiter	6
3.6	Audio Bus	6
3.7	Audio Bus Capability	6
3.8	Audio Bus Device	6
3.9	Byte	6
3.10	CEA (Consumer Electronics Association)	6
3.11	Clear	6
3.12	ID	6
3.13	ITS	6
3.14	Logical Node	6
3.15	MC (Message Code)	6
3.16	Mute	7
3.17	Octet	7
3.18	OEM (Original Equipment Manufacturer)	7
3.19	OSI	7
3.20	Output Level	7
3.21	Physical Node	7
3.22	PMODE (Power Mode Line)	7
3.23	Primitives	7
3.24	Receiver	7

SAE Technical Standards Board Rules provide that: "This report is published by SAE to advance the state of technical and engineering sciences. The use of this report is entirely voluntary, and its applicability and suitability for any particular use, including any patent infringement arising therefrom, is the sole responsibility of the user."

SAE reviews each technical report at least every five years at which time it may be reaffirmed, revised, or cancelled. SAE invites your written comments and suggestions.

TO PLACE A DOCUMENT ORDER: +1 (724) 776-4970 FAX: +1 (724) 776-0790
SAE WEB ADDRESS <http://www.sae.org>

Copyright 2002 Society of Automotive Engineers, Inc.
All rights reserved.



Documents

Document Sharing Hub

3.25	SAPID (Service Access Point Identifier)	7
3.26	Set	7
3.27	Transmitter.....	7
4.	Protocol Overview	8
4.1	Basic Concepts	8
4.1.1	SAPID for Message Interpretation.....	8
4.1.2	SAPID for Service Announcement.....	8
4.1.3	Physical Nodes and Logical Nodes	9
4.1.4	Physical Node IDs and Logical Node IDs	9
4.1.5	Logical Node Status	9
4.2	Application Layer Primitives.....	10
4.2.1	Send Application Message	10
4.2.2	Receive Application Message	10
4.2.3	Change Status Request.....	11
4.2.4	Get Status Information.....	11
4.3	Application Common Header Elements.....	11
4.3.1	Control	12
4.3.2	SAPID	13
4.3.3	Address.....	13
4.3.4	Sequence	14
4.3.5	MsgLen (MSB).....	14
4.4	Application Message Layer SAPIDs	14
4.4.1	Application SAPIDs.....	14
4.4.2	Standard APDU Encoding Rules	14
4.4.3	Encapsulated Protocols	20
5.	Standard Messages Supported by All Logical Nodes	22
5.1	APLNewStatus and APLReqStatus	22
5.2	APLReqLogicalNodeInfo and APLDevInfo	23
5.3	APLReqMaxMsgLen and APLMaxMsgLen	24
5.4	APLPing and APLPingReply	24
5.5	APLDiag and APLDiagReply	24
5.6	APLResults	24
6.	Audio Arbitration	24
6.1	Introduction	24
6.2	Audio Bus Messages SAPID	25
6.3	Responsibilities	25
6.3.1	Audio Arbiter	25
6.3.2	Transmitter.....	25
6.3.3	Receiver.....	25
6.4	Audio Arbitration Sequences	25
6.4.1	Audio Arbiter Identification.....	25
6.4.2	Transmitter Activation	26
6.5	Audio Bus Messages	28
6.5.1	ABCapabilityReq and ABCapabilityReply.....	28
6.5.2	ABTransmitRequest	30
6.5.3	ABSetTransmitParms	30
6.5.4	ABAudioStart	30
6.5.5	ABAudioDone	30
6.5.6	ABDeviceDisconnect	30
6.5.7	ABMute	30

7.	Example Message Construction	31
Appendix A ITS Data Bus Application Message Layer Lexicon		33
A.1	Scope	33
A.2	IDB Application Message Layer PDU Elements	33
A.2.1	ClassMap	33
A.2.2	Action	33
A.2.3	Modifier	35
A.2.4	Object	37
A.2.5	Feature	39
A.2.6	UnitType and UnitType Lists	41
A.2.6.1	PositionList	45
A.2.6.2	HeadingList	47
A.2.6.3	ResultList	47
A.2.6.4	AudioList	48
A.2.6.5	StatusCodeList	49
A.2.6.6	TransmissionStateList	49
A.2.6.7	FuelTypesList	50
A.2.6.8	WarningList	50
A.2.6.9	EntPlayerStatusList	51
A.2.6.10	Languages	51
A.2.6.11	DisplayStatusList	51
A.2.6.12	Encapsulated Protocols	51
A.3	Explanations and Rationale	52
A.3.1	PositionList	52
A.3.2	HeadingList	52
A.3.3	ResultList	52
A.3.3.1	RAcknowledge	52
A.3.3.2	RAAlreadyInUse	52
A.3.3.3	RCRCError	52
A.3.3.4	RDecodeError	52
A.3.3.5	RDeviceNotPresent	52
A.3.3.6	RDeviceNotReady	52
A.3.3.7	RDeviceReady	52
A.3.3.8	RFailure	52
A.3.3.9	RInProcess	52
A.3.3.10	RInvalidAction	52
A.3.3.11	RInvalidFeature	53
A.3.3.12	RInvalidLength	53
A.3.3.13	RInvalidMessage	53
A.3.3.14	RInvalidModifier	53
A.3.3.15	RInvalidObject	53
A.3.3.16	RInvalidUnitType	53
A.3.3.17	RInvalidValue	53
A.3.3.18	RNoMedia	53
A.3.3.19	RNotSupported	53
A.3.3.20	ROutOfRange	53
A.3.3.21	RSecurityError	53
A.3.3.22	RSuccess	53
A.3.3.23	RTooFrequent	53
A.3.3.24	RUnknown	53

A.3.4	AudioList	53
A.3.5	StatusCodeList	53
A.3.6	TransmissionStateList	53
A.3.7	FuelTypesList	53
A.3.8	WarningList	53
A.3.9	EntMediaInfo.....	53
A.3.10	EntPlayerStatusList	54
A.3.11	EntTrackInfo	54
A.3.12	Language	55
A.3.13	DisplayStatusList	55
A.3.14	Encapsulated Protocols	55
A.4	Notes	55

List of Figures

Figure 1	Application Message Layer Message Format.....	12
Figure 2	APDU Format	15
Figure 3	Standard MC Packet Format	16
Figure 4	StatusCode Format.....	19
Figure 5	Encapsulated Protocol Example	21
Figure 6	Transmitter Activation Sequence	27
Figure 7	Transmit Request Rejected Sequence	28
Figure 8	Set Trip Origin Message	31
Figure 9	Request Current Vehicle Speed Message	31
Figure 10	Vehicle Speed Response Message.....	31
Figure 11	Request Periodic Speed Report Message	32
Figure 12	Periodic Speed Report Message	32
Figure 13	Update Rate Too High Response Message	32

List of Tables

Table 1	SAPID Size	13
Table 2	SAPID Categories and Document Cross-Reference	14
Table 3	ClassMap Element of MC	16
Table 4	Valid Action Elements.....	17
Table 5	Valid Modifier Elements	17
Table 6	Valid Object Elements	18
Table 7	Valid Feature Codes	18
Table 8	Valid UnitType and Corresponding Value Elements.....	18
Table 9	Supported Encapsulated Protocols	21
Table 10	Application Messages Supported by All Logical Nodes	22
Table 11	Audio Bus Common Application Messages.....	28
Table 12	Audio Bus Use Categories	30
Table A1	ClassMap	33
Table A2	Action	33
Table A3	Modifier	35
Table A4	Object	37
Table A5	Feature	39
Table A6	UnitType and UnitType Lists	41
Table A7	PositionList Names and Values	45
Table A8	HeadingList Names and Values	47
Table A9	ResultList Names and Values.....	47
Table A10	AudioList Names and Values	48

Table A11	StatusCode List Names and Values	49
Table A12	Transmission StateList Names and Values	49
Table A13	FuelTypesList Names and Values	50
Table A14	WarningList Names and Values	50
Table A15	EntPlayerStatusList Format	51
Table A16	Language Names and Values	51
Table A17	DisplayStatusList Names and Values	51
Table A18	Supported Encapsulated Protocols	51
Table A19	EntMediaInfo Format	53
Table A20	EntTrackInfo Format	54

1. Scope

NOTE— SAE J2366-7LX—ITS Data Bus Application Message Layer Lexicon is now combined (as an appendix) with SAE J2366-7 to provide for a quicker revision process in the future.

This SAE Surface Vehicle Recommended Practice defines an Application Message Layer, which may be used as part of a complete protocol stack with the other protocol layers in the SAE J2366 family.

The Application Message Layer provides application message support for devices that are interconnected via a bus or network. Design of the messages and headers has stressed flexibility, expandability, economy (in terms of octets on the bus), and reusability.

The Application Message Layer is independent of the underlying network used, and may be used on any network. Sections of this document make specific reference to the use of SAE J2366-7 with other layers of the SAE J2366 family of protocols, and may not apply when SAE J2366-7 is used with other networks.

In addition, this document defines an audio arbitration scheme that may be used with the SAE J2366-1L Low Impedance Stereo Audio (LISA) bus, or any other comparable analog audio bus.

2. References

2.1 Applicable Publications—The following publications form a part of this specification to the extent specified herein. Unless otherwise indicated, the latest issue of SAE publications shall apply.

2.1.1 SAE PUBLICATIONS—Available from SAE, 400 Commonwealth Drive, Warrendale, PA 15096-0001.

SAE J1760—Data Security Services (Under Development at time of this document's publication)
 SAE J2366-2—ITS Data Bus Protocol Link Layer
 SAE J2590—PMode for In-Vehicle Networks

2.2 Related Publications—The following publications are provided for information purposes only and are not a required part of this specification.

2.2.1 SAE PUBLICATIONS—Available from SAE, 400 Commonwealth Drive, Warrendale, PA 15096-0001.

SAE J2355—ITS Data Bus Architecture Reference Model Information Report
 SAE J2366-1—ITS Data Bus Protocol Physical Layer
 SAE J2366-1L—Low Impedance Stereo Audio (LISA) bus
 SAE J2366-4—ITS Data Bus Protocol Thin Transport Layer

2.2.2 ISO PUBLICATIONS—Available from ANSI, 25 West 43rd Street, New York, NY 10036-8002.

ISO/IEC 8824:1998 (ITU-T X.680) Information Technology – Abstract Syntax Notation One (ASN.1):
Specification of Basic Notation

ISO 7498:1990, Information Processing Systems – Open System Interconnect – Basic Reference Model

3. Definitions

- 3.1 **AHdr (Application Header)**—This element identifies the contents of Application Messages and identifies how the data are to be interpreted.
- 3.2 **ALP (Application Layer Primitives)**—These are generally function call parameters and are “out-of-band” signals between an application and the Application Message Layer.
- 3.3 **AMI-C (Automotive Multimedia Interface Collaboration)**—<http://www.ami-c.org>
- 3.4 **APDU (Application Protocol Data Unit)**—This packet of data contains the specific application messages. One or more specific messages may be contained within a single APDU.
- 3.5 **Audio Arbiter**—The function that determines whether and at what levels devices may access the analog Audio Bus. The Audio Arbiter function may be present on any node on the network, and does not have to be associated with a Transmitter or Receiver, nor does it require direct connection to the Audio Bus.
- 3.6 **Audio Bus**—An audio bus that allows Transmitters and Receivers to share one or more audio channels. The Audio Bus is assumed to support individual output gain settings for Transmitters and audio mixing of multiple Transmitter outputs. The Audio Bus is assumed to be capable of accommodating up to a maximum of 16 connected Receivers, and a maximum of 16 connected Transmitters. When used with the SAE J2366 series of protocols, the total number of physical connections is limited to 16.
- 3.7 **Audio Bus Capability**—A device that is a Transmitter and/or a Receiver is considered to have Audio Bus Capability.
- 3.8 **Audio Bus Device**—A device that uses or controls access to the Audio Bus channels. A single physical device may host one or more of the following device functions: Audio Arbiter, Transmitter, Receiver. There can be a maximum of 16 Audio Bus Devices on the network.
- 3.9 **Byte**—In this document, bytes and octets are synonymous. See the definition for octet for more information.
- 3.10 **CEA (Consumer Electronics Association)**—<http://www.ce.org>
- 3.11 **Clear**—The word “clear” is used in referring to one or more bits to indicate that they each have a value of '0'B.
- 3.12 **ID**—Identifier
- 3.13 **ITS**—Intelligent Transportation Systems
- 3.14 **Logical Node**—This is used to indicate an application running on a Physical Node or on another device that is directly attached to the Physical Node, and for which the Physical Node is acting as a “proxy” on the network. A Logical Node has a Logical ID within that Physical Node. Logical Node ID '0'H refers to the Physical Node itself.
- 3.15 **MC (Message Code)**—Each MC represents a single Application Layer message. Multiple MCs may be chained together in a single APDU.

- 3.16 Mute**—This action, when requested of a Transmitter, causes it to apply silence to the Audio Bus. This may, in some cases, be equivalent to an Output Level of zero. This action, when requested of a Receiver, causes it to silence the Receiver output.
- 3.17 Octet**—An octet is a unit of data transmission that consists of eight bits. On most commonly used microprocessors, such an eight-bit quantity is commonly referred to as a “byte.” The bits within an octet may be subdivided and/or grouped, to provide specific meanings.
- 3.18 OEM (Original Equipment Manufacturer)**—In this document it refers to the manufacturer of a vehicle.
- 3.19 OSI**—Officially known as the “ISO Reference Model of Open System Interconnection.” It refers to a model, or way of thinking about communication protocols as being divided into (usually) seven distinct layers.
- 3.20 Output Level**—When sent to a Transmitter, defines the output level that the Transmitter is to present to the Audio Bus. This parameter takes a value of 0-255, which represents a range of 0-100%.
- 3.21 Physical Node**—A Physical Node is a device that makes a single physical connection to the network. Each Physical Node has at least one Physical Node ID. Each Physical Node ID may have associated with it up to 15 additional “virtual” node IDs, called Logical Node IDs.
- 3.22 PMODE (Power Mode Line)**—This is a signal defined in SAE J2590 that indicates the power mode of the network. There are three basic power modes: KEY ON, KEY OFF, and KEY PULSE. Of these modes, KEY ON and KEY OFF are the only ones generally pertinent to an application.
- 3.23 Primitives**—These are parameters necessary for the operation of a protocol but are generally not part of the data stream between layers of a protocol stack. These parameters are passed through various methods including function call variables, private messages, and global variables.
- 3.24 Receiver**—An Audio Bus device that “sinks” audio from one or more channels of the Audio Bus. Such devices might include: Amplifier, Voice Recognition System. There can be at most 16 connected Receivers on the Audio Bus.
- 3.25 SAPID (Service Access Point Identifier)**—There are two primary uses / definitions of SAPID.
- In the body of an APLNewStatus message, the SAPID value is used to specify a group of messages, representing a service being provided, which are supported by a Logical Node.
- In the AHdr, the SAPID value specifies the interpretation to be placed upon the message. The value represented by this element identifies which “interpreter” should be used to decode an APDU.
- This release of the SAE J2366-7 protocol supports a large number of SAPIDs. SAPIDs in the range 0 to 32767 are interpreted according to the rules laid out in this document. The SAPID value may be used to differentiate (for example) between “unlock vehicle doors” and “unlock CD changer door.” Although the APDU itself may contain the same message octets for both messages, one SAPID will specify that the message is a vehicle related message while the other SAPID will specify that the message is a CD changer related message. SAPID values above 32767 are intended for use by encapsulated protocols. The SAPID specifies what protocol material is contained in the APDU, allowing it to be interpreted correctly.
- 3.26 Set**—The word “set” is used in referring to one or more bits to indicate that they each have a value of '1'B.
- 3.27 Transmitter**—An Audio Bus device that produces audio onto one or more channels of the Audio Bus. Such devices might include: Text to Speech generator, Radio Tuner, Cassette Player, CD Changer, Mobile Phone. There can be at most 4 simultaneously active Transmitters on the Audio Bus.

4. Protocol Overview—SAE J2366-7 is designed to meet the varied and challenging requirements for in-vehicle data communications. Some noteworthy features of the protocol:

- a. The protocol is independent of the underlying protocol layers.
- b. The protocol follows the OSI protocol layering concepts.
- c. It provides the capability to transmit messages of up to 2051 bytes in length.
- d. It provides information necessary for the application to maintain the type of device associated with each node present on the network.
- e. It provides a set of node diagnostic messages to allow remote verification of the node's "sanity" at Physical through Application Layers and provides node statistical information.

4.1 Basic Concepts—Application messages consist of two parts: the AHdr and the APDU. The Application Header contains a SAPID that identifies how the APDU is to be interpreted. This SAPID may indicate different types of Application Messages or it may indicate that the information is encapsulated information from another protocol.

All field values or bits that are not explicitly defined by this document, or explicitly referenced to other documents are reserved for future definition by a later version of SAE J2366-7. They are not available for arbitrary use by specific implementations.

4.1.1 SAPID FOR MESSAGE INTERPRETATION—The term "SAPID" comes from the "Service Access Point" concept. Here it is used within a SAE J2366-7 message to specify the method by which the message is to be interpreted.

Some SAPID values specify a particular protocol that is being encapsulated within SAE J2366-7 messages. An example of this would be SAPID 35072 ('8900'H) for the Kermit protocol. Some SAPID values specify that the message is a message intended to be interpreted by a particular type of device or service. An example of this would be SAPID 10 ('0A'H), from the "Vehicle" category.

It is possible for a single Physical Node to support several types of Logical Node. The SAPID allows such a Physical Node to direct the incoming message to the correct interpreter of the message contents.

4.1.2 SAPID FOR SERVICE ANNOUNCEMENT—The other use for a SAPID is to announce the availability of a particular type of Logical Node or service to other Logical Nodes on the network. Used in this way, it allows other Logical Nodes to know which destination is able to perform a particular function or respond with particular information.

The mechanism for this announcement is the Standard Message APLNewStatus, which specifies the SAPID value(s) that correspond to services being provided by that Logical Node. Each Logical Node may announce 1-16 SAPIDs in its APLNewStatus message.

When a Logical Node is prepared to transmit and receive SAE J2366-7 layer messages, it transmits an APLNewStatus message as a broadcast. This may take place an arbitrary amount of time following the Logical Node's connection to the network. When a Logical Node receives a broadcast APLNewStatus message, it responds to the message with a point-to-point APLNewStatus message that includes its identification. A Logical Node does not transmit an APLNewStatus in response to a point-to-point APLNewStatus.

All Logical Nodes support the messages specified later in this document that use a SAPID value of seven (7), therefore SAPID 7 shall not be announced. A Logical Node that provided access to certain vehicle control or information via messages using SAPID 10 ('0A'H), from the "Vehicle" category, would advertise this availability by including a SAPID 10 value in an APLNewStatus message. Other Logical Nodes that desire to access the capabilities provided by that Logical Node, and that send messages using SAPID 10, do not advertise SAPID 10 within their own APLNewStatus message, because they are not providing the service.

The standardized message set documents for various types of devices will specify the SAPID value(s) associated with those devices.

- 4.1.3 PHYSICAL NODES AND LOGICAL NODES—The SAE J2366-2 Data Link Layer supports up to 16 Physical Node addresses. As stated previously, each Physical Node may have associated with it up to 16 Logical Nodes. This provides for up to 256 different Logical Nodes operating on a single network.

Each Physical Node shall have an application responsible for the overall control of the Physical Node. This application shall be the first application to connect to the network and it shall register the Physical Node's basic capabilities by transmitting its supported SAPIDs via the APLNewStatus message.

- 4.1.4 PHYSICAL NODE IDs AND LOGICAL NODE IDs—In most Physical Nodes, the application is responsible for notifying the lower layers of the protocol when the Physical Node may join and should leave the network. It is also the application's responsibility to identify the primary capabilities of the Physical Node by assigning it one or more SAPIDs.

SAE J2366-7 supports up to 16 Logical Nodes per Physical Node. Logical Node number 0 is assigned to the Physical Node itself, and Logical Node numbers 1 through 15 are available for assignment to the separate application capabilities supported by the various Logical Nodes on the Physical Node. Each Logical Node on a Physical Node shall individually use primitives to connect or disconnect from the lower protocol layer (e.g., SAE J2366-4—Thin Transport Layer) and shall be assigned, on connection, a Logical ID. The first connection request shall be made by the Physical Node application, which is assigned a Logical ID of 0 (zero). Subsequent Logical IDs may be assigned in any order.

Logical ID 0 has special properties. It affects all other Logical Nodes on a Physical Node. If Logical Node 0 becomes inactive, then all other Logical Nodes on that Physical Node are also considered inactive. The inverse, however, is not necessarily true. When Logical Node 0 is active, not all other Logical Nodes on the Physical Node need necessarily be active. While Logical Node 0 is active, individual Logical Nodes 1 through 15 may become active or inactive at the request of the application assigned to that Logical Node.

- 4.1.5 LOGICAL NODE STATUS—When a Node wishes to announce the activation of a new Logical Node, the first message it is responsible for sending is a broadcast APLNewStatus message, which identifies the new Logical Node to all other Logical Nodes on the network.

The APLNewStatus message may also be sent in response to an APLReqStatus broadcast message received by a Logical Node. The APLNewStatus message will include the status of all active Logical Nodes operating on the Physical Node.

With the exception of Logical ID 0 (which is an alias for the Physical Node itself), the only states for a Logical Node are active and inactive. A Logical Node may be active only when the Physical Node (Logical Node 0) is active and the application bound with or connected to the Logical Node has marked its local state as active and has sent the APLNewStatus broadcast onto the network.

The status of each Logical Node on a Physical Node is controlled by the application. When this status changes, the Logical Node shall broadcast an APLNewStatus message for each change. When the status of a Physical Node (Logical Node 0) changes from active to inactive, all Logical Nodes greater than 0 become inactive. (Note: When the entire Physical Node status changes, its Logical Nodes are not required to broadcast an APLNewStatus message, however, they may do so, if possible.)

4.2 Application Layer Primitives—Primitives are interfaces that allow information (usually control or address information) to pass from one layer to another layer. This information is generally necessary for the targeted layer to set a state or properly process a message. This section provides an example of such primitive operations between an Application and the Application Message Layer.

Four Primitives are described between the Application and the Application Message Layer. These are provided to describe the parameters passed between the application and the Application Message Layer. Specific implementations are at liberty to modify these to suit their needs, as long as the information that is eventually transmitted over the network conforms to the protocol specifications.

4.2.1 SEND APPLICATION MESSAGE—This primitive is the interface that permits an Application to send an Application Message Layer message to other Logical Nodes on the network. This primitive, provided by the Application Message Layer, allows the Application to provide the following parameters:

- a. The destination Logical Node ID – encoded into the AHdr as DLAddr. The Logical Node ID is used rather than a SAPID to identify the destination Logical Node. This is because the Destination Physical Node may have more than one Logical Node that uses that SAPID, e.g., a Physical Node with two tuners, and there would be no way to determine which of the two is the intended destination Logical Node.
- b. The destination Physical Node ID – passed to the next lower layer.
- c. Network Broadcast flag – passed to the next lower layer to indicate that this message is to be sent to all Logical Nodes on the network.
- d. Application Acknowledge Request – encoded into the AHdr as fAckReq. This permits an application to request acknowledgement of a point-to-point or broadcast message. This is useful, for example, when a Logical Node wishes to send a message to multiple instances of the same application and requires an acknowledgement from one or more of them.
- e. SAPID – encoded into the AHdr; indicates to the receiving Application how the APDU is to be interpreted.
- f. Message Length – encoded into the AHdr as MsgLen; provides the total length of the APDU. This version of SAE J2366-7 limits the length to 2051 octets.
- g. Delivery Priority of the message – used by the Application Message Layer in determining the correct sequencing of outgoing messages. The highest priority is 0; the lowest is 3.
- h. Application Priority of the message – encoded into the AHdr as DestPriority; used only by the receiving application and may differ from the Delivery Priority. The highest priority is 0; the lowest is 3.
- i. A sequence number – encoded into the AHdr as Sequence with a value of 0-127; assigned by the Application so that it is able to identify and match up responses and/or track messages. The most significant bit of the sequence number octet is reserved for future use related to Data Security Services, and shall be '0'B at this time.

4.2.2 RECEIVE APPLICATION MESSAGE—This primitive is the interface that permits an Application to receive an Application Message Layer message sent from Logical Nodes on other Physical Nodes or from Logical Nodes on the same Physical Node. This primitive, provided by the Application Message Layer, returns the following information to the receiving Application:

- a. Logical ID of the Logical Node originating the message – decoded from the AHdr (SLAddr).
- b. The source Physical Node address – passed up from the next lower layer.
- c. Application Acknowledge Request flag – decoded from the AHdr; informs the receiver that the sender has requested an acknowledgement.
- d. SAPID – decoded by the Application Message Layer from the AHdr, indicates to the receiving Application how the APDU is to be interpreted.
- e. Total length of the APDU – decoded from the AHdr. For this version of SAE J2366-7 the maximum length is 2051 octets.
- f. Application level priority of the message – decoded from the AHdr; used only by the receiving application. The highest priority is 0; the lowest is 3.

- g. A sequence number – decoded from the AHdr, assigned by the sending Application and to be used in any reply. This number may be a value between 0 and 127. The most significant bit of the sequence number octet is reserved for future use related to Data Security Services, and shall be '0'B at this time.

4.2.3 **CHANGE STATUS REQUEST**—This primitive is used by an application to control how and when a Physical Node joins and/or leaves the network, and how and when a Logical Node becomes active or inactive. Only one parameter is required and may have the following values:

- a. NotPresent – tells the lower layers (e.g., SAE J2366-2—Link Layer) to remove the Physical Node from participation on the network.
- b. Active – allows / requests the Physical Node to join the network if not already joined and to become an actively participating Physical Node. This primitive will return with a SUCCESS or FAILURE depending on the outcome of the request.
- c. Quiescent – allows / requests an active Physical Node to become Quiescent or Sleeping. If the Physical Node is NotPresent when this primitive receives the request, it will return FAILURE. If the command was successfully executed, the primitive will return SUCCESS.
- d. Join – allows / requests a Logical Node to begin participating on the network. If the Physical Node is not already Active, this primitive will return FAILURE, otherwise it will return SUCCESS.
- e. Leave – allows / requests a Logical Node to be excluded from participating on the network. This primitive does not affect the status of the Physical Node.

NOTE— When SAE J2366-7 is used with other networks, the mechanism to join/leave the network or change state may be different.

4.2.4 **GET STATUS INFORMATION**—This primitive is used by a Logical Node to get the current status information about its Physical Node. Only two parameters are returned: RunState and PMode. RunState may have the following values:

- a. NotPresent – indicates that the Physical Node is not participating on the network, i.e., the Physical Node has no Physical Node ID.
- b. Active – indicates that the Physical Node is actively participating on the network.
- c. Quiescent – indicates that the Physical Node has a Physical Node ID but is not actively participating in passing the token.

PMode may have the following values:

- a. KEY ON – indicates that the PMode signal is in the KEY ON state.
- b. KEY OFF – indicates that the PMode signal is in the KEY OFF state.

NOTE— When SAE J2366-7 is used with other networks, the mechanism to determine Physical Node status may be different.

4.3 **Application Common Header Elements**—Figure 1 shows the format of the Application Message Layer messages.

All Application Layer messages begin with a common header, called the Application Header (AHdr). This header is further divided into five components: Control, SAPIID, Address, Sequence, and MsgLen. This header is not necessarily visible to the application. It is a “wrapper” placed in front of the APDU by the Application Message Layer. The combination of the AHdr and the APDU form the message that is transmitted application layer-to-application layer.

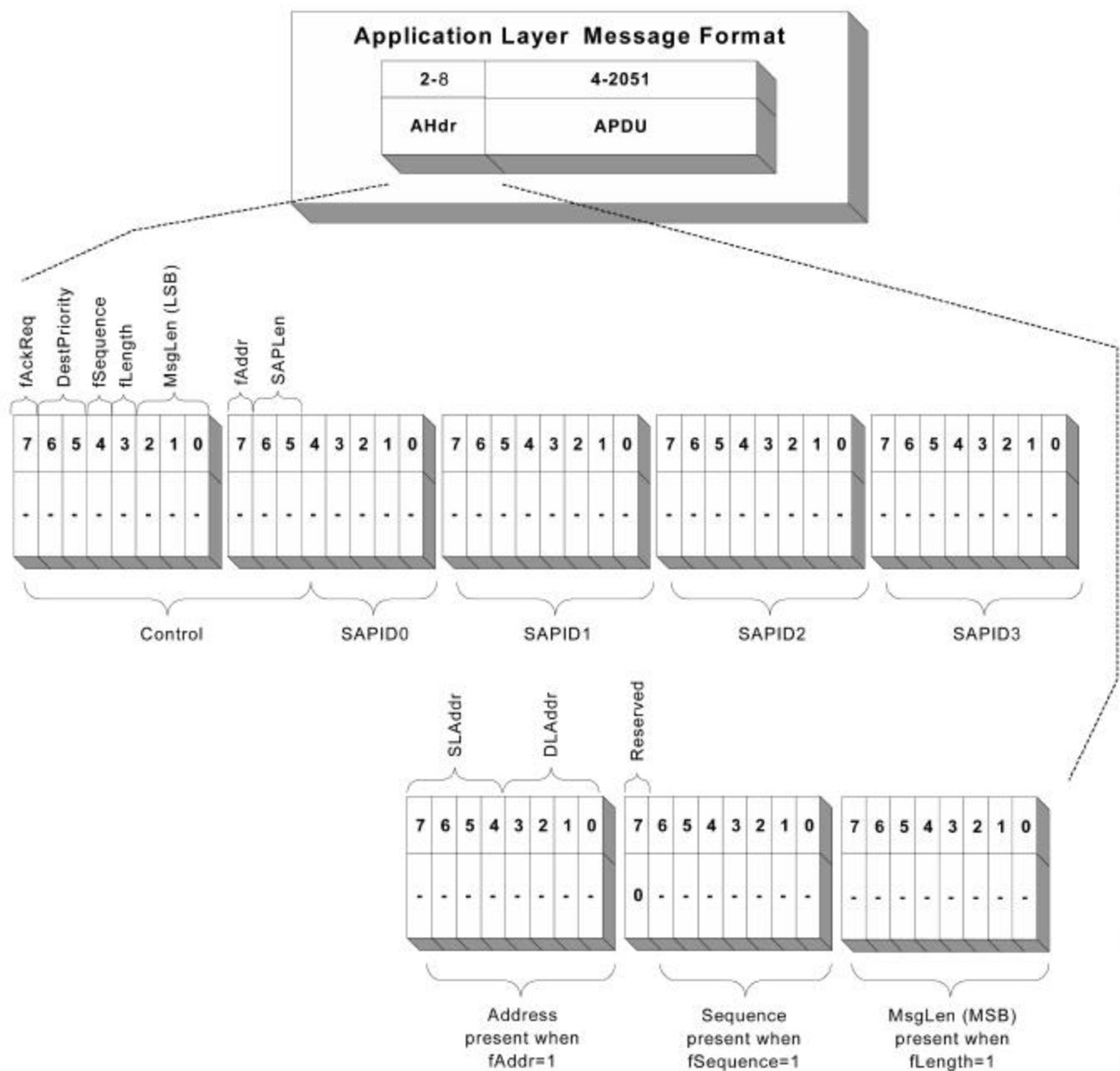


FIGURE 1—APPLICATION MESSAGE LAYER MESSAGE FORMAT

- 4.3.1 **CONTROL**—This element identifies key components of the AHdr and indicates how the AHdr is to be decoded. This element is broken down into seven sub-components, each of which is described as follows.
- 4.3.1.1 *fAckReq*—This sub-component, when set, indicates that the application is requesting an acknowledgement to this message, as described in 4.2.1.
- 4.3.1.2 *DestPriority*—This sub-component indicates the Application Priority of the message to the destination Application Layer. The highest priority is 0; the lowest is 3. Applications should restrict use of highest priority for genuinely urgent messages.

- 4.3.1.3 *fSequence*—This sub-component is a flag that indicates whether the Sequence component is present in the AHdr. When set, it indicates that it is present. When cleared, it indicates that it is not present.
- 4.3.1.4 *fLength*—This sub-component is a flag that indicates whether the MsgLen (MSB) octet is present in the AHdr. When set, it indicates that it is present. When cleared, it indicates that it is not present.
- 4.3.1.5 *MsgLen (LSB)*—This sub-component consists of 3 bits, which, together with MsgLen (MSB) defines the number of octets in the APDU. This allows a maximum APDU length of 2047 + 4 octets in the MC (described in 4.4.2) for a total of 2051 octets. If the number of octets in the APDU is between 4 and 11, MsgLen (MSB) is not required and fLength is cleared. This component is communicated to and from the Application via Application-to-Application Message Layer primitives.
- 4.3.1.6 *fAddr*—This sub-component is a flag that indicates whether the Address component is present in the AHdr. When set, it indicates that it is present. When cleared, it indicates that it is not present.
- 4.3.1.7 *SAPLen*—This identifies the length of the SAPID component of the AHdr. The correlation is described in Table 1.

TABLE 1—SAPID SIZE

SAPLen (bits)	SAPID Size (Bits)
00	5
01	13
10	21
11	29

- 4.3.2 **SAPID**—The SAPID component indicates the encoding format of the message. This component is always present, but its length may vary from 5 to 29 bits according to Table 1. The Application Message Layer format initializes the SAPLen to the shortest length that can indicate the SAPID. This component is communicated to and from the Application via Application-to-Application Message Layer primitives.

When multiple SAPID octets are required, they are transmitted in Network Byte Order, i.e., SAPID 0 (most significant bits) is transmitted first.

SAPIDs of 0-32767 are reserved for Logical Node categories. SAPIDs 0-9 are reserved for future protocol diagnostic services. SAPIDs greater than 32767 are reserved for encapsulated protocols.

- 4.3.3 **ADDRESS**—The Address component indicates the logical origin and destination of the message. This address is broken into two sub-elements: SLAddr and DLAddr. This component is included in all point-to-point messages from this Node and in Network Broadcast messages that require a response. This component only appears in the AHdr if the fAddr sub-component is set. This component is communicated to and from the Application via Application-to-Application Message Layer primitives.

- 4.3.3.1 *SLAddr (Source Logical Address)*—This element contains the Logical ID of the Logical Node sending the message.

- 4.3.3.2 *DLAddr (Destination Logical Address)*—This element contains the Logical ID of the Logical Node to which the message is directed.

NOTE— If a message for a particular SAPID is sent and the Address component is not present, the implied DLAddr is the lowest numbered Logical Node ID that has published that SAPID value.

4.3.4 SEQUENCE—The Sequence octet is divided into two parts, a Reserved bit and the Sequence sub-component:

4.3.4.1 *Reserved*—The most significant bit of the sequence number octet is reserved for future use related to Data Security Services, and shall be '0'B at this time.

4.3.4.2 *Sequence*—This sub-component may have a value between 0 and 127.

4.3.5 MSGLEN (MSB)—This component, together with MsgLen (LSB) described in 4.3.1.5, indicates the number of octets contained in the APDU. This allows a maximum APDU length of 2047 + 4 octets for a total of 2051 octets. If the number of octets in the APDU is between 4 and 11, MsgLen (MSB) is not required and fLength is cleared. This component is communicated to and from the Application via Application-to-Application Message Layer primitives.

4.4 Application Message Layer SAPIDs

4.4.1 APPLICATION SAPIDs—As previously mentioned, SAPIDs 7 and 10 through 32767 use the same basic message format. The Application Message SAPIDs are grouped into major “categories” as defined in Table 2. Logical Nodes or applications identify themselves as belonging to a particular SAPID, to specify their type of Logical Node or application. It is intended that the document references in Table 2 will be added as the documents are published. Work is being done to provide for one or more documents for each category that describes the Application Messages supported for that category.

NOTE—The Data Security Services SAPID Range is expected to be defined by the work of the SAE J1760 committee.

TABLE 2—SAPID CATEGORIES AND DOCUMENT CROSS-REFERENCE

Category	SAPID Range	Document
Commerce	2560-2815	
Communications	256-511	
Computation	512-767	
Diagnostic	768-1023	
Emergency/Public Safety	1280-1535	
Entertainment	1536-1791	
Reserved for SAE J2366 Protocols	0-9	
Navigation	1792-2047	
Other	3072-3839	
Public Service	4096-4351	
Security	2048-2303	
Data Security Services	4352-4607	
Storage	2304-2559	
Traveler Information	3840-4095	
User Interface	1024-1279	
Vehicle	10-255	
Warning	2816-3071	

4.4.2 STANDARD APDU ENCODING RULES—All messages with a SAPID of 7 or 10 through 32767 use the same encoding rules and message format for the APDU. The APDU may include one or more MC packets as illustrated in Figure 2. This permits applications to send multiple commands or to respond with multiple data to a request without incurring unnecessary protocol overhead.

NOTE— The names and values provided in this document for each of the MC's elements are only to support the basic message set specified in Section 5. The SAE J2366-7LX Application Message Layer Lexicon contains the latest information on element names and values.

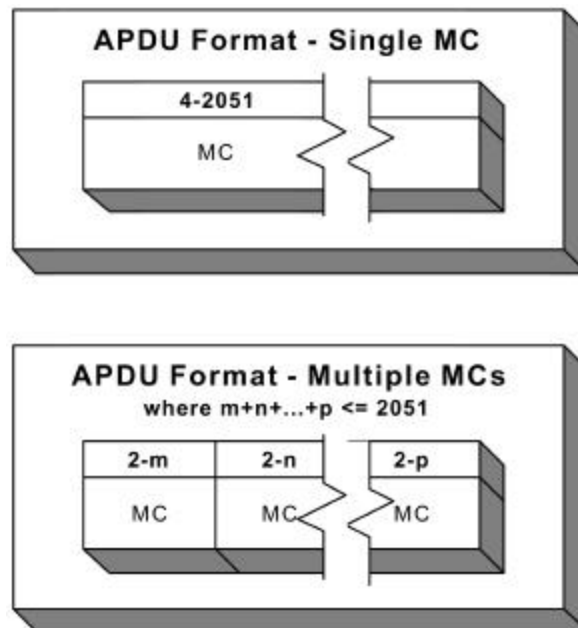


FIGURE 2—APDU FORMAT

The format for the MC is illustrated Figure 3. The minimum length of an MC is 2 octets, viz., ClassMap and Action. If there is only one MC in the APDU and it has fewer than 4 octets, it shall be padded to 4 octets with '00'H octets. If it already contains 4 or more octets, no padding is required. If there are 2 or more MCs in the APDU, each may be 2-n octets long, as long as $\sum n$ (the total length of the APDU) does not exceed 2051 octets.

A single SAE J2366-7 message shall have no more than one Action element, but may have multiples of other elements. This is achieved by adding additional MCs with the Action element of AContinue. Any MC with the AContinue Action has the same bMsgClass value as the initial MC of the message. However, the other bits of the ClassMap octet correspond to MC elements present within the continuation MC. This means that a continuation MC might contain an additional Modifier, Object, Feature, UnitType, and/or Value element.

An APDU whose first MC has an Action of AContinue shall be ignored. Any MC that contains an Action that is not AContinue shall begin a new message.

- 4.4.2.1 ClassMap Element**—This element identifies each message as belonging to one of the four Message Classes. It is further broken down into two sub-components: bMsgClass and bMsgMap. The bMsgClass identifies the general message class and the bMsgMap identifies which of five elements is present in the message. The current standard defines only 4 general message classes, as shown in Table 3, but has room to expand to 8. All unassigned values are reserved for future use. All messages have an Action element.

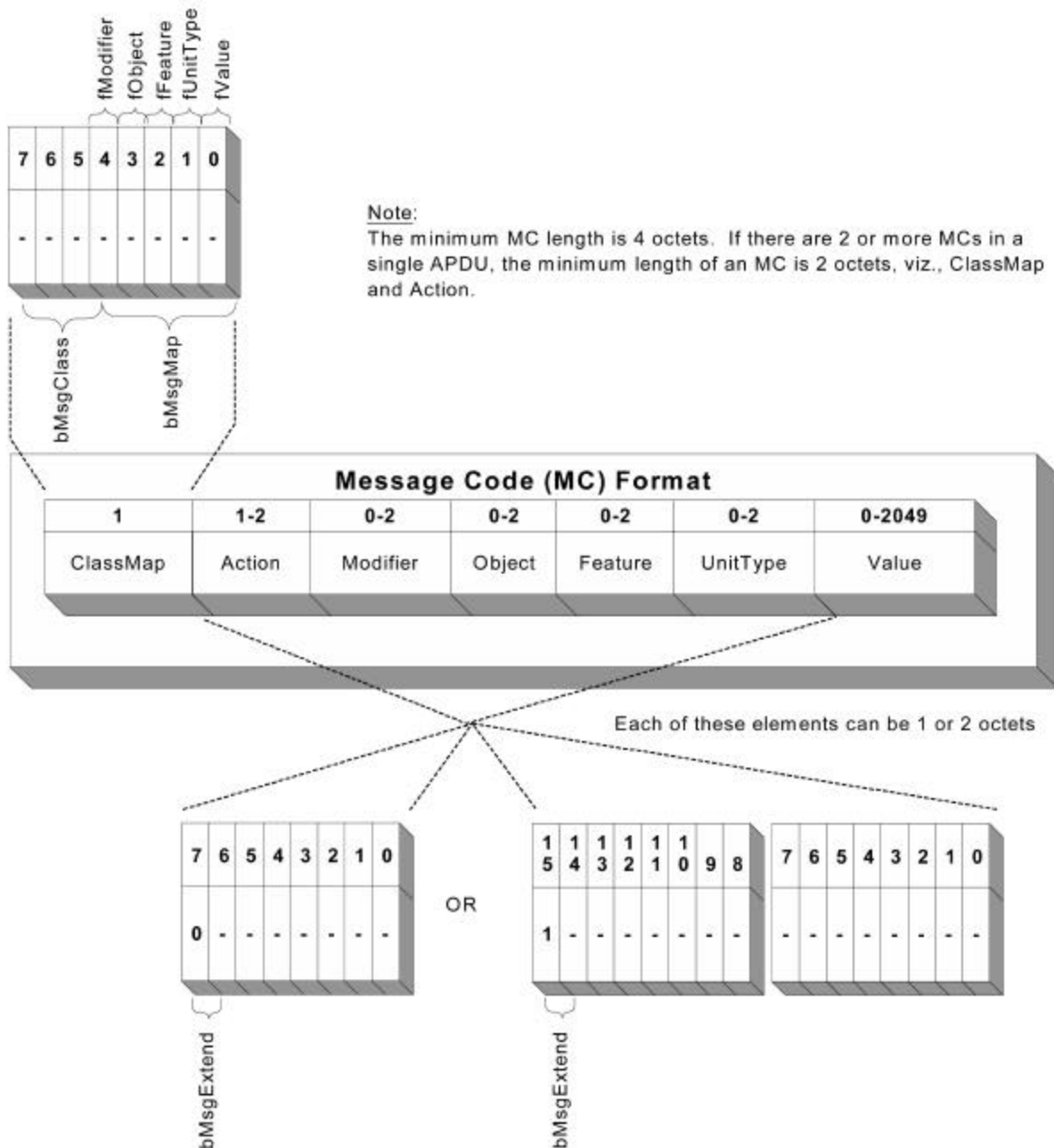


FIGURE 3—STANDARD MC PACKET FORMAT

TABLE 3—CLASSMAP ELEMENT OF MC

BMsgClassName	bMsgClass
Request	'000'B
Command	'001'B
Reply	'010'B
Datagram	'011'B

- 4.4.2.1.1 Request—This type of message may be used to request information from another Logical Node. The result of this type of message is typically a Reply message.
- 4.4.2.1.2 Command—These messages are used to issue a command. Examples of commands are setting the volume level of a particular Logical Node, turning a Logical Node on or off, specifying an address for a route calculation.
- 4.4.2.1.3 Reply—A Reply message is sent in response to a Request or Command message. The response shall either be the data requested or a status or error code indicating why the request or command failed.
- 4.4.2.1.4 Datagram—Datagrams are unsolicited or periodic application messages. The Sequence element of the AHdr shall be set to zero in Datagram messages. A periodic message might be one requested by another Logical Node.
- 4.4.2.2 *Action Element*—This element is present in all messages but has particular meaning within Request and Command message classes. It generally indicates an action to be carried out by the recipient of the Request or Command. When the bMsgExtend bit of this element is set then this element occupies 2 octets, otherwise, it occupies 1 octet. The Action elements required to support the minimal functionality listed in Section 5 are listed in Table 4.

TABLE 4—VALID ACTION ELEMENTS

Action Name	Value
ANone	0
AEcho	1
AREport	2
AContinue	32

- 4.4.2.3 *Modifier Element*—This element is used within Request and Command message classes and generally modifies or expands an action that is to be carried out by the recipient of the Request or Command. When the bMsgExtend bit of this element is set, this element occupies 2 octets, otherwise, it occupies 1 octet.

The Modifier element does not always appear in a message. When used, it shall be paired with an Action element other than ANone. As with most of the message elements, the presence of the element is indicated when the fModifier sub-component is set within the ClassMap element. The Modifier elements required to support the minimal functionality listed in section Section 5 are listed in Table 5.

TABLE 5—VALID MODIFIER ELEMENTS

Name	Value
MReserved	0
Max	16

- 4.4.2.4 *Object Element*—This element may be used with all classes of messages. However, it is primarily used with the Reply and Datagram message classes and generally identifies a general Logical Node or capability. If the bMsgExtend bit of this element is set, then this element occupies 2 octets, otherwise, it occupies 1 octet.

The Object element is present when the fObject sub-component is set in the ClassMap element. In most cases, this element is paired with a Feature element. The Object elements required to support the minimal functionality listed in Section 5 are listed in Table 6.