SAE The Engineering Society For Advancing Mobility Land Sea Air and Space® INTERNATIONAL
400 Commonwealth Drive, Warrendale, PA 15096-0001

SURFACE VEHICLE RECOMMENDED PRACTICE

Submitted for recognition as an American National Standard

SAE J2056/1

Issued 1993-06-14

CLASS C APPLICATION REQUIREMENT CONSIDERATIONS

FOREWORD

Three classes of vehicle communications have been defined by the SAE Vehicle Networking for Multiplexing & Data Communications Standards Committee. One of those classes, Class C applications, represents those communications which are intended for real-time control systems such as engine controls and anti-lock brakes in order to facilitate distributed control and further reduce vehicle wiring. The requirements for these applications are different from those required for either Class A or Class B applications. This paper describes those requirements specific to a Class C application. An example system is provided for consistency of discussion.

TABLE OF CONTENTS

Printed in U.S.A.
90-1202B EG

1. ***Scope***—This SAE Recommended Practice will focus on the requirements of Class C applications. The requirements for these applications are different from those required for either Class A or Class B applications. An overall example system is provided for consistency of discussion.

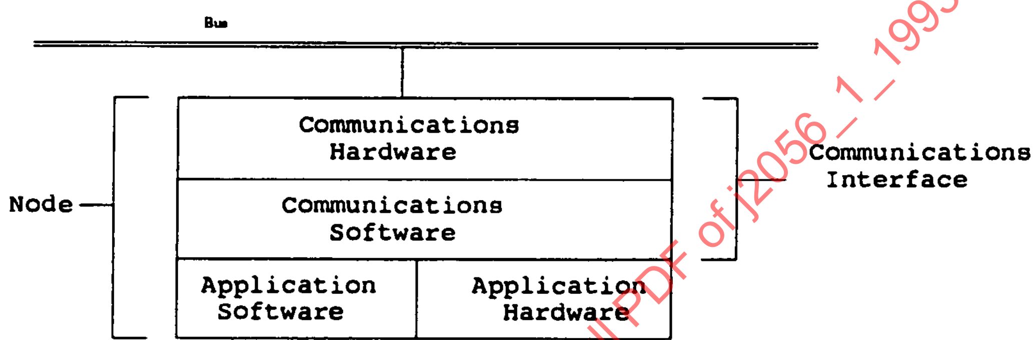

FIGURE 1—TYPICAL NODE BLOCK DIAGRAM

Figure 1 is a block diagram of a typical network node. The hardware and software for both the communication interface and the application itself are shown. For the purposes of this discussion the communication hardware and software will be considered as the communication interface. This paper will discuss the requirements for the communication interface (both hardware and software) without necessarily determining whether it will be accomplished by hardware or software. This choice is left as a subsequent trade-off. Thus requirements are presented from the perspective of the application.

The examples provided are for discussion purposes only and are in no way intended to be an endorsement or recommendation of how a specific application should be designed.

1.1 **Background**—Three classes of vehicle communications have been identified by the SAE Vehicle Networking for Multiplexing & Data Communications Standards Committee. These classes are defined as follows:

a. Class A—A potential multiplex system usage whereby vehicle wiring is reduced by the transmission and reception of multiple signals over the same signal bus between nodes that would have been accomplished by individual wires in a conventionally wired vehicle. The nodes used to accomplish multiplexed body wiring typically did not exist in the same or similar form in a conventionally wired vehicle.

b. Class B—A potential multiplex system usage whereby data is transferred between nodes to eliminate redundant sensors and other system elements. The nodes in this form of a multiplex system typically already existed as stand-alone modules in a conventionally wired vehicle.

c. Class C—A potential multiplex system usage whereby high data rate signals typically associated with real-time control systems, such as engine controls and anti-lock brakes, are sent over the signal bus to facilitate distributed control and further reduce vehicle wiring.

These three classes describe the various applications of communication that are anticipated to exist within a vehicle. Each class is intended to be able to support the lower level classes of applications also. That is, Class A systems are designed for basic low level switch multiplexing. Class B introduces the aspect of parametric data sharing while still providing for Class A applications. Class C introduces the aspect of real-time closed-loop feedback machine control but still allows Class B and Class A tasks to be performed. Issues such as cost, reliability, and performance will determine which link or combination of links are most appropriate for a given application.

It is believed there are significant benefits available to the automotive, component, and semiconductor manufacturers in developing a standard Class C communication network. The work performed toward standardization of the Class B network has provided insight into the magnitude of and potential methods for this effort, and has shown that this is a significant undertaking—one which must be initiated early. A discussion of the benefits of standardization are presented in Appendix A.

2. *References*

2.1 Applicable Document—The following publication forms a part of this specification to the extent specified herein. The latest issue of SAE publications shall apply.

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

SAE J1850—Class B Data Communication Network Interface

2.1.2 OTHER DOCUMENTS

Ford Motor Company, "ETX-I Final Report," Volume I

Patil, P. B., et. al., "Electric Transaxle System Design for an Advanced Electric Vehicle Powertrain," EVC Expo 83, Paper No. 8324, Dearborn, Michigan, October, 1983

Bates, B., et. al., "A Vehicle Control System for an Electric Vehicle," 19th IECEC, Paper No. 849441, San Francisco, California, August, 1984

Landman, R. G., et. al., "Control System Architecture for an Advanced Electric Vehicle Powertrain," SAE Paper No. 871552, Future Transportation Technology Meeting, Seattle, Washington, August, 1987

3. ***Example Description***—To clarify the communication requirements of a distributed control system, an electric vehicle drive- and brake-by-wire system is described. One version of the system was implemented in an advanced electric vehicle powertrain called ETX-I (ETX-Electric Trans-Axle). The system consisted of seven modules: the vehicle controller (V/C), the inverter/motor (I/M) controller, the instrument panel display, the transmission, the traction battery, brakes, and driver inputs.

The vehicle controller is the command center of the system. It electronically interprets all driver demands by monitoring the accelerator and brake pedals and the shift lever and provides the desired wheel torque response by appropriately controlling the inverter, motor, transmission, and brake operation. It also provides fault management and diagnostics. In the ETX implementation, two dedicated serial data links were used: one between the vehicle controller and I/M controller and the other between the vehicle controller and the display. All other signals, such as between the vehicle controller and the transmission or the I/M controller and the inverter, used hard wires. However, a network-based system configuration with intelligent nodes for each major subsystem as shown in Figure 2 is feasible and is used to illustrate the communication requirements of a distributed control system.

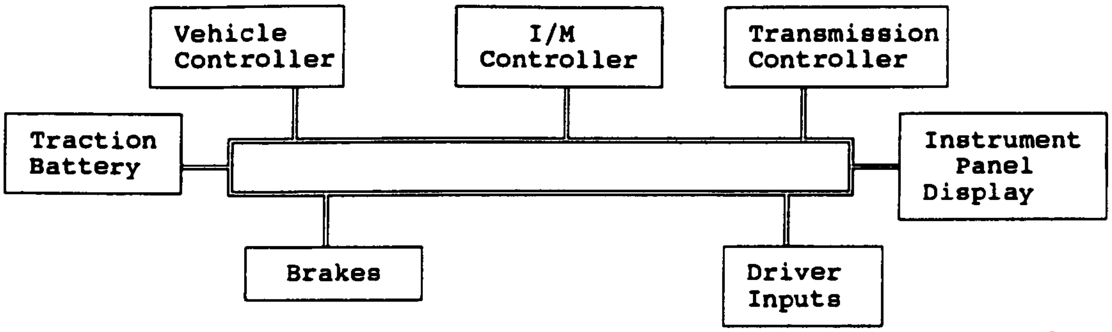

FIGURE 2—EXAMPLE BLOCK DIAGRAM

The operation of the system can best be described with reference to Figure 3 which shows the state diagram for the ETX-I system. The circles represent states and the interconnecting arrows represent possible transition paths. The logical expressions associated with each transition arrow describe the conditions necessary for the transition to occur. Tables 1 and 2 provide a list of the states and the events responsible for state transitions. Detailed description of the propulsion system, its operation and the design of the control system can be found in references 1 to 4. What follows is a brief description of certain aspects of system operation to illustrate the relationship between system operation and communication requirements.

State 0 in Figure 3 is the state to which the control system is initialized when the ignition key is turned on. Several messages are initiated as a result of this driver action: the vehicle controller needs to know the status of the shift lever, the friction clutches in the transmission, and the relay that locks the power on to the vehicle controller itself. If the transmission is not in "park" or "neutral," for example, the system goes to (fault) State 1 and displays an appropriate message. On the other hand, if the transmission is in "park" or "neutral," the friction clutches are disengaged and the power relay is locked on. When the key switch goes to the "start" position, the system transitions to State 2 and energizes the relay that provides power to the I/M Controller. Thus the driver action of turning the ignition key to "on" and "start" positions results in a "burst" of messages. These messages are not repetitive. However, they should have low, worst case latencies to prevent a perceptible delay between key turning and system initialization.

When the system is in State 14, the V/C is continuously sending a torque command to the I/M Controller at a 5 ms update rate and needs to receive the pedal position and the calculated torque value at the same rate. This is an example of repetitive data which must be updated at a rate fast enough to provide a smooth response. If the time required for acknowledgement of such data is a significant fraction of the update period, it may be desirable not to require an acknowledgement and use the old data until the next error-free data packet arrives. In State 14, the vehicle controller is also monitoring several other signals which are event driven and non-repetitive and whose receipt in certain combinations can lead to state transitions. These messages may have to be acknowledged to prevent faulty state transitions. Some of these messages need very short latencies to provide proper operation and fault management.

One of the ETX-I Control system state transitions is a gear shift from 1st to 2nd gear, State 19. Since the gear shift initiation and execution depend on the occurrence of certain conditions in a specific sequence within a given time interval, it is necessary for the communication system to meet requirements for data consistency, sequence, and latency. Also during this period the clutch pressure and motor speed signals need to be updated at a much faster rate (5 to 10 ms) than during normal driving in a particular gear. System design, must, therefore, address the issue of whether to allow for this high update rate and low latencies all of the time (high bandwidth) or to provide the means to dynamically change priorities and update rates.

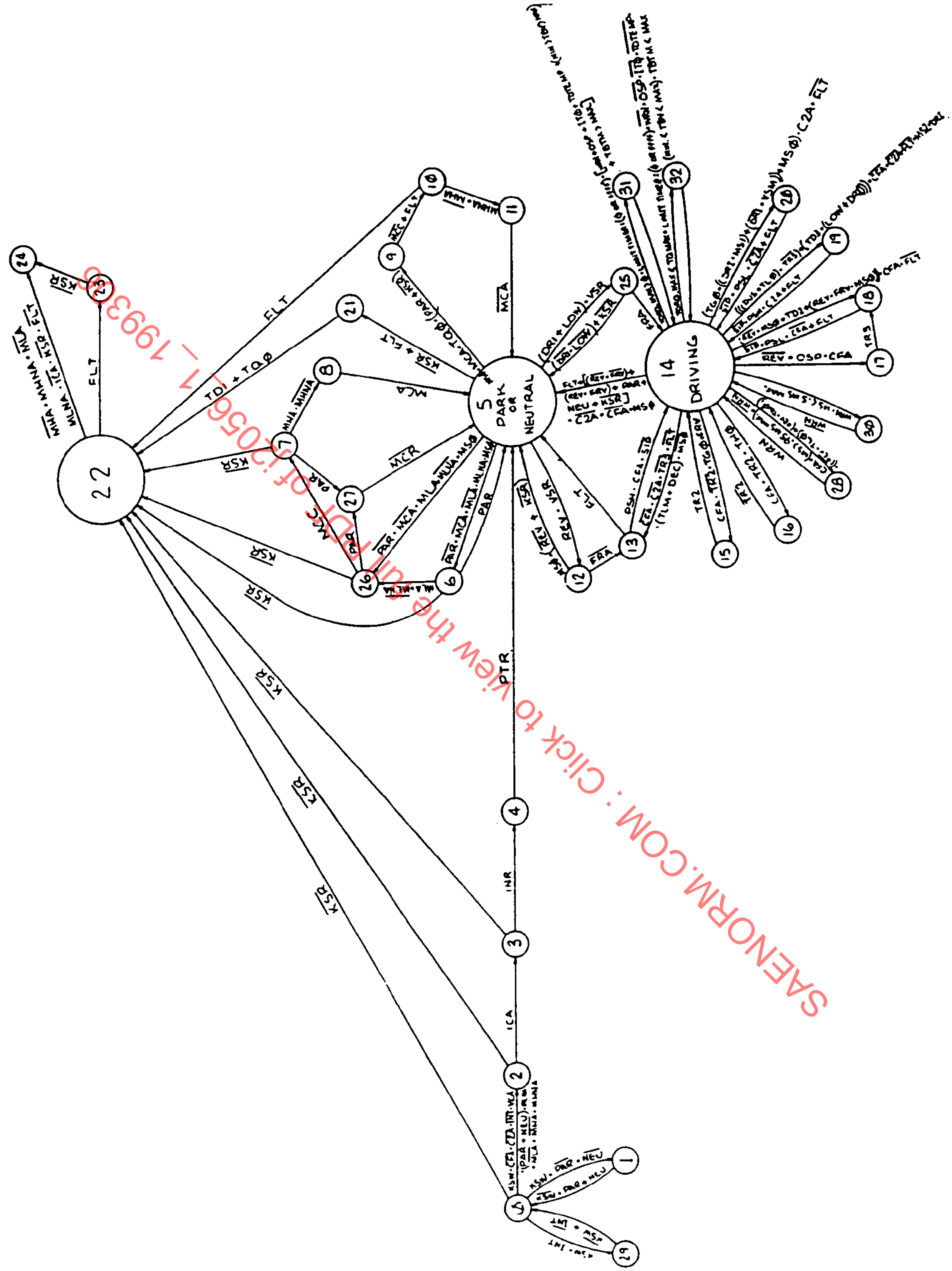

FIGURE 3—EXAMPLE STATE DIAGRAM

TABLE 1—ETX SOFTWARE STATE DIAGRAM DEFINITIONS

State	Definition
0	Initialize Vehicle Controller
1	Display "SHIFT NOT IN PARK OR NEUTRAL" Message
2	Close Inverter/Motor Controller Power Relay
3	Send Initialization Record to I/MC
4	Send PTR (Prepare to Run) message to I/MC
5	PARK or NEUTRAL - Output 0 torque
6	Close main contactor negative side. Start timeout for MLA.
7	Close main contactor positive side. Start timeout for MHA.
8	Send MCA
9	Send NOT(MCR) and start timeout for NOT(MCC)
10	Open Main Contactor and start timeout for NOT(MHNA)
11	Send NOT(MCA).
12	Send NOT(FRV) to I/MC
13	Energize 1st Gear Friction Clutch
14	DRIVING - Process Torque Schedule and output to I/MC continuously.
15	Start time delay 2 for regen to drive shift
16	Stop time delay 2
17	Start time delay 3 for clutch to clutch shift
18	De-energize 1st gear friction clutch
19	Energize 2nd gear clutch
20	De-energize 2nd gear clutch
21	Start timeout 1 for measured torque to equal 0
22	SHUTDOWN - Open inverter/motor controller power relay, main contactors, and clutch solenoids
23	Display diagnostic status
24	Open vehicle controller power relay
25	Send FRV to I/MC
26	Send MCR and start timeout for MCC
27	Send NOT(MCR)
28	Set motor overspeed warning flag
29	Display "CHARGER STILL PLUGGED IN" message
30	Clear motor overspeed warning flag
31	Decrement maximum power available to motor
32	Increment maximum power available to motor
33	Increase negative torque to limit motor speed
34	Decrease negative torque

TABLE 2—ETX STATE TRANSITION EVENTS

C2A	Second Gear Friction Gear Solenoid Acknowledge
CFA	First Gear Friction Gear Solenoid Acknowledge
DEC	Rate of change of accelerator pedal position (APP) function: (dAPP/dt < -Y and (APP < X)
DRI	Gear Shift in Drive Position
FLT	Fault Detected
FRA	Forward/Reverse Acknowledge from I/MC
FRV	Forward/Reverse Request to I/MC
INR	Initialization Record Received
ICA	Inverter Power Relay Acknowledged
KSR	Key Switched On (in Run or Start Position)
KSW	Key Switched to Start Position
LOW	Gear Shift in Low Position
MCC	I/MC ready for contactor closure
MCA	Contactor acknowledge sent to I/MC
MS1	Motor Speed for Scheduled Downshift
MS2	Motor Speed for Scheduled Upshift
NEU	Gear Shift in Neutral Position
OSP	Motor Over Speed Detected
PAR	Gear Shift in Park Position
PTR	Prepare to Run Signal sent to I/MC
REV	Gear Shift in Reverse Position
SDN	Shutdown signal from Inverter Motor Controller
TD1	Time Delay 1 Timed Out (Time for Measured Torque to Reach 0)
TR1	Time Delay 1 Running
TD2	Time Delay 2 Timed Out (Time to Shift from Regen to Drive)
TR2	Time Delay 2 Running
TD3	Time Delay 3 Timed Out (Delay for clutch to Clutch Shift)
TR3	Time Delay 3 Running
TG0	Torque Request - 0 + delta
TH0	Torque Request < 0 - delta
TL0	Torque Request < 0 - delta
TQ0	0 - delta < Measured Torque < 0 + delta
VCA	Vehicle Controller Power Relay Acknowledge
VSM	Maximum Allowable Vehicle Speed for Downshift
VSR	Allowable Vehicle Speed for Shift Into Reverse
INT	Charger Interlock (charger is plugged in)
MLA	Main Contactor Neg. Contact Closed
MHA	Main Contactor Pos. Contact Closed
MLNA	Main Contactor Neg. Contact Open
MHNA	Main Contactor Pos. Contact Open
MS0	Motor Speed Equals Zero
SIP	Shift in Progress
SIN	Shift Inhibit
WRN	Motor Near Overspeed
TBV	Traction Battery Voltage
TBTM	Traction Battery Temperature
ITO	Inverter Overtemp
TOTEMP	Transaxle Overtemp

In the ETX system the serial communication between the V/C and the I/M Controller was handled as described in the following. The controller was implemented in software. Information was passed as message packets of variable length from 3 to 7 bytes long. Each message packet began with a sync byte and ended with a checksum byte. These two features allowed the processors to determine with a high degree of confidence that they were interpreting the right string of bytes as a packet and described the information contained in the rest of the packet. Two types of data were passed. Non-periodic data, such as status information, were updated as required and had to be acknowledged by the receiving processor. If the acknowledgement was not received within a given period, the transmitting processor retransmitted the message packet. Other data were periodically updated. This type was not acknowledged. Since a packet with an incorrect checksum is ignored, the processor will continue using the previous value of the data until it is updated. This saves time on the data that are updated often (e.g., the torque command is updated every 5 ms) yet assures that valid information is always used.

Table 3 provides a list of signals for the ETX-I control system. The first part of the table lists the signals which used individual wires in the ETX implementation. However, these signals could be sent using the communication bus in a fully multiplexed system. In this case, the analog signals would have to be converted to digital ones as indicated by the dual entries under the "type" column in Table 3. The second part of the table shows signals that were sent using the serial link between the vehicle and the inverter/motor controllers described previously. The signals which do not have an update rate associated with them in Table 3 are "event driven" (i.e., driver action or a change in state) and their update rate can only be described statistically. Some of the signals in Table 3 can cause a state change and therefore cause other signals to be generated as was described for the key-on signal earlier.

4. ***Application Requirements***—The requirements of Class C applications associated with the communication characteristics will be described in the following sections. This section is grouped into five major areas: (a) regularity of information transfer, (b) performance, (c) dependability, (d) system issues, and (e) implementation. Within each of these sections, the specific requirements affecting the communication network will be discussed.

4.1 **Regularity of Information Transfer**—As defined, Class C applications are control-oriented. By the fundamentals of control system design, there is a high level of repetitiveness inherent in the system. This occurs in particular with digital control systems where sample times and processing loops are a fundamental part of the design. Periodic events also arise from periodic physical phenomena such as crankshaft position in an engine.

For the control applications currently being considered, parameters such as coolant temperature, throttle position, engine load level, and vehicle speed are typically updated at regular intervals. This information must then be communicated to other subsystems within the vehicle. Update intervals may range from milliseconds to seconds depending on the rate of change of the data and its intended use. In the example in Section 2.0, the discussion of State 14, where V/C is continuously sending a torque command to the I/M C at 5 msec update rate and needs to receive pedal position and the calculated torque value at the same rate, is an example of repetitive data (reference Table 3).

Despite the high degree of regularity of information, it will also be necessary to accommodate the transfer of irregular information and information bursts. This includes the transfer of information associated with irregular events such as driver-initiated mode changes, failures, or command information being transferred between command modules which triggers multiple messages. An example of such information was described in Section 3. In that case, in State 0 (initialization) a burst of messages is set once at start-up. These are considered to be irregular data transmissions in this case.

It is anticipated that a large percentage of the data transferred on a Class C network will be repetitive in nature. This data will represent sampled information and values calculated in control loops. This is in contrast to a Class B application, where bursty information is more prevalent.

TABLE 3—ETX VEHICLE CONTROL SYSTEM SIGNALS

	Type	Update Interval (ms)	Symbol	From	To
Traction Battery Voltage	A/D8	100	TBV	TBat[1]	V/C
Traction Battery Current	A/D8	100	TBI	TBat	V/C
Traction Battery Temp, Avg	A/D8	1000	TBTA	TBat	V/C
Traction Battery Temp, Max	A/D8	1000	TBTM	TBat	V/C
Auxiliary Battery Voltage	A/D8	100	ABV	ABat	V/C
Auxiliary Battery Current	A/D8	100	ABI	ABat	V/C
Accelerator Position	A/D8	5	APP	Driver	V/C
Brake Pressure, Master Cylinder	A/D8	5	BPM	Brkes	V/C
Brake Pressure, Line	A/D8	5	BPL	Brkes	V/C
Transaxle Lubrication Pressure	A/D8	100	PLT	Trans	V/C
Transaxle Clutch Line Pressure	A/D8	5	PCT	Trans	V/C
Vehicle Speed	Pulse train	100	WHS	Brake	V/C
Traction Battery Ground Fault	D1	1000	TGF	TBat	V/C
Hi & Lo Contactor Open/Close	D4	-	Table 2	TBat	V/C
Key Switch Run	D1	-	KSR	Driver	V/C
Key Switch Start	D1	-	KSW	Driver	V/C
Accelerator Switch	D2	-	ASW	Driver	V/C
Brake Switch	D1	-	BSW	Brake	V/C
Emergency Brake	D1	-	PBK	Driver	V/C
Shift Lever (PRNDL)	D3	-	Table 2	Driver	V/C
Motor/Trans Over Temperature	D2	1000	TOTEMP	Trans	V/C
Speed Control	D3	-	SPC	Driver	V/C
12 V Power Acknowledge					
Vehicle Controller	D1	-	VCA	ABat	V/C
Inverter	D1	-	ICA	ABat	V/C
I/M Controller	D1	-	IMCA	ABat	V/C
Brake Mode (Parallel/Split)	D1	-	BMD	Driver	V/C
SOC Reset	D1	-	SOCR	Driver	V/C
Interlock	D1	-	INT	TBat	V/C
High Contactor Control	Pulse train	10	MHC	V/C	TBat
Low Contactor Control	Pulse train	10	MLC	V/C	TBat
Reverse & 2nd Gear Clutches	D2	-	PC1/2	V/C	Trans
Clutch Pressure Control	Pulse train	5	PC1/2	V/C	TBat
DC/DC Converter	D1	1000	DDC	V/C	TBat
DC/DC Converter Current Control	Pulse train	-	DIC	V/C	TBat
12 V Power Relays	D1	-	APC/TPC	V/C	TBat
Traction Batt. Ground Fault Test	D2	1000	TGF	V/C	Brake
Brake Solenoid	D1	-	BSL	V/C	Brake
Back Up Alarm	D1	-	BUA	V/C	Brake
Warning Lights	D7	-	-	V/C	InstPnl
Key Switch	D1	-	KSW	V/C	I/M C
Main Contactor Close	D1	-	MCC	I/M C	V/C
Torque Command	D8	5 ms	TQC	V/C	I/M C
Torque Measured	D8	5 ms	TQM	I/M C	V/C
FWD/REV	D1	-	FRA	V/C	I/M C
FWD/REV Acknowledge	D1	-	FRA	I/M C	V/C
Idle	D1	-	IDL	V/C	I/M C Shift
Inhibit	D1	-	SIN	I/M C	V/C
Shift in Progress	D1	-	SIP	V/C	I/M C
Processed Motor Speed	D8	5 ms	PMS	I/M C	V/C
Inverter Temperature Status	D2	-	ITS	I/M C	V/C
Shutdown	D1	-	SDN	I/M C	V/C
Status/Malfunction (TBD)	D8	-	SML	I/M C	V/C
Main Contactor Acknowledge	D1	-	MCA	V/C	I/M C

[1] Abbreviations: TBat - Traction Battery
ABat - Auxiliary Battery
Trans - Transmission
A/D8 - Analog with 8 bits resolution
D4 - Digital 4 bit value

4.2 Performance—The performance of a communication system is typically discussed in terms of communications speed—how long it takes to get messages through the data link. The most common measures of communication speed are throughput and latency. Throughput rates are selected by the system designer within limits determined by cost and EMI considerations. Latency is the time delay between queuing a message for transmission and receipt at the destination. Latency is somewhat affected by bit rate choices but is more significantly affected by the protocol's logical structure—particularly the media access method.

The latency of data exchange may be considered from a number of different perspectives. From an application perspective, latency relates to the time delay associated with the transfer of information from one application program to another. In this respect it is necessary for the latency to be minimal (in some cases less than 1 ms) and predictable (as defined by the systems exchanging data). Predictability has to do with how the message latencies change over time—whether they follow a statistical distribution or are deterministic.

As a control system is designed, the delay resulting from information transfer is accounted for in such a way that the control algorithm is able to compensate for that fact. From this perspective the necessity of predictable latency (latency defined within a given tolerance) can be understood. Obviously, the need to minimize latency is also a function of the algorithms and acceptable limits for each application must be defined. In some cases such as chassis control systems, latencies less than 2 ms may be necessary.

Priority can be used as a means of reducing the latency associated with a particular message. In that sense, priority can be used to assure that critical messages remain within necessary latency bounds. In some cases, priority may be used to override normal operation of the system to allow certain messages to be sent immediately. Many forms of priority may be used, including dynamic alteration of the priority as a function of time in the message transmission queue. Class B systems may also utilize priority for a similar reason (in some cases it is considered an additional identifier field). In any case, the use of priorities can only improve latencies for the highest priority message at the expense of other messages. If the media access method is statistical by nature, priority methods cannot insure absolute guarantees on latencies.

For example, a message containing the vehicle speed and engine speed (rpm) may be sent at a normal priority level, where as the command from the brake system to the throttle system to reduce throttle for a given traction condition may be sent at the highest priority. An application of dynamic prioritization may be used for messages involving wheel speed. In this instance, the wheel speed message would be sent at normal priority, until the system entered a critical situation. At that time the priority for the wheel speed message might be raised to a higher level.

4.3 Dependability—Since a Class C data link is likely to be used in safety-critical systems, it must be designed so as to dependably perform its function. This need for dependability includes traditional component-oriented reliability requirements, but goes beyond this in ways that affect system reliability and safety. For example, a fault that could cause unsafe operation should be detected and the system put into a safe mode, perhaps with degraded performance, whenever possible. This implies that there should be means for defeating and recovering from communication failures such as corrupted messages, media faults, and failed nodes.

The communications protocol must be able to provide guaranteed delivery of critical messages. Some types of messages might not need this, but for those that do, various acknowledgement schemes can help. Immediate acknowledgement can be used to indicate correct receipt of a valid message. This might be automatically returned by the interface hardware itself or it might be generated by low-level communications software. In either case, the acknowledgement is based on the validity of the message framing, bit encoding, CRC codes, etc., that define syntactically correct messages.

Beyond this, the sending task may need to know that the message was successfully acted upon by the application program at the destination. In the example system, this occurs when the vehicle controller issues a torque request to the Inverter/Motor Controller and can observe the measured torque output to verify correct operation. This can be accomplished by returning status information, but may require special provisions in the protocol to ensure timely acknowledgement. If a fast acknowledgement is needed, it may not be acceptable to wait for normal media access methods. The initiating node might need to retain control of the medium and allow the destination node to transmit without arbitrating for access to the network.

The data link must also deliver the message intact. Various techniques such as message framing, checksums, and CRC fields can be used to validate messages. If a message can be sent to more than one recipient, then it is also necessary that the recognition of an invalid message at one node force all other receiving nodes to reject that message. Otherwise, data and parameters can have inconsistent values throughout the system. However, one node must not be able to coninuously invalidate messages used by other nodes. A receiving task may also need the ability to detect missed messages (e.g., sensor values).

This need for communications integrity applies to the time order of messages as well. If a sequence of messages is sent in a particular order, it should be received in that order as well. This requirement affects queuing disciplines used to buffer messages, particularly in bridge or gateway nodes. However, the same problems can arise in networks where nodes function as active relays with store-and-forward capacity. This can be accomplished by time stamping and sequence numbering the messages.

Obtaining adequate system dependability may require the use of fault tolerant computing techniques. This places special requirements on the communications interface. One such technique may be the use of special bit encoding techniques to allow the regeneration of the data in messages containing errors. The data link itself must also be fault tolerant with graceful degradation. The failure of individual nodes must not cause total loss of the communications capability. If the physical medium is damaged or severed, the remaining portions should be able to communicate within themselves.

If true fault tolerance is to be implemented, the system will need the capability to recognize faulty behavior, identify the faulty node, and reconfigure the system to provide the most acceptable performance possible. This implies that the nodes can monitor the network itself and identify other nodes that are not responding or transmitting correctly. There must then be some method for ignoring or isolating the failed nodes in order to reduce their impact on the system.

Functional reconfiguration means that the critical functions which were being performed at a node that failed, are shifted to remaining good nodes. To do this, the task-to-task communications must use functional addressing rather than physical. Additionally, it may be necessary to dynamically reconfigure the network in some cases.

4.4 Impact on Vehicle System—From the perspective of the vehicle or subsystem engineer, a data link is not a function in itself. Rather, it is a means to providing the features and functions desired for a control system. The decision to incorporate a data link is driven by the earlier decision to partition the system functionality among a number of nodes. The resulting need that data be shared between nodes having interrelated functions is the precursor that leads to adopting the use of a data link. However, the data link must also enable the system to meet the requirements that originally lead to the partitioning choice. These requirements fall into three general categories: cost, flexibility, and fault tolerance.

Clearly, any technical choices must be cost effective. The most tangible cost impact of a data link comes from the medium, the connectors (including assembly costs), and the hardware/software interfaces required to support the protocol. One immediate impact of the protocol design comes through requirements for timing accuracy and synchronization. If the timing requirements are strict enough to force use of tight-tolerance components such as crystal clocks, this will add to the cost of every node on the data link. These are easily measured, but represent only part of the costs.

The choice of a data link also impacts the overall costs of a vehicle system in less direct ways. The communications link itself places an overhead burden on the processor(s) at a node. This reduces the node's available computation capacity and may require a faster (more expensive) processor in order to meet the time-critical aspects of control. Similarly, the interface circuitry places demands on the node's electrical power.

The data link also impacts cost by affecting the design cycle. The hardware and software interfaces must be easy to use. It must support modular partitioning on the software and easy reconfiguration so that the system designer is not artificially constrained by the communications. In general, this can be achieved by making the communications link as transparent as possible to the application programs. Ideally, two tasks should be able to share data as easily over a data link as if they were running on the same processor. In particular, this implies the use of functional addresses rather than physical addresses.

The data link must also support test instrumentation used during prototype development. On the physical side, the instrumentation tools will need to be connected to the data link medium. This places extra drive loads on the electronics and may require automatic reconfiguration of the network to accommodate the instrumentation equipment as an additional node. There must be communications bandwidth to carry debugging and diagnostic messages in addition to the normal traffic. The development process may also require testing partial systems in a bench-top prototype form. This will be easier if the protocol does not require a master node in order to operate.

Similar capabilities are needed for service diagnostics. In addition, the diagnostics may need to monitor all data link traffic and verify correct operation. To do this effectively, the diagnostic equipment may need to determine the source of every message. One way to facilitate this is to have a source ID field in every message.

4.5 Physical Implementation—There are a number of issues associated with the implementation of a Class C network which may also impose requirements on any network solutions. These issues focus primarily on the physical interface between the application and the network. It is generally expected that between 2 and 30 nodes will reside on a Class C network. This number is dependent on the extent of control system distribution and the number of vehicle options incorporated. It is also expected that bus rates in excess of 100 kbits/s will be required and that they will likely reach 1 Megabit/s or greater. This, obviously, is dependent on issues discussed earlier associated with the performance requirements.

4.5.1 OPENNESS—An open system is defined as a system consisting of nodes interconnected by a common communications medium (signal bus) according to established standards, which will support temporary connections to manufacturing networks, diagnostics, and other local area networks. This may be extended to allow devices to be added or deleted to the network without significant physical alterations. Class C applications will require an open system allowing nodes to be added and deleted as freely as possible. As a minimum this will require a standard protocol for the physical and data link layers. This will allow nodes to communicate effectively on the network. It may also be desirable to establish an open interconnect (i.e., standardized interconnect) at higher levels (e.g., priority definitions, specific diagnostic messages); however, this is not essential.

4.5.2 LEVEL OF HARDWARE IMPLEMENTATION—The nature of the applications and performance requirements imply constraints on the hardware implementation. First, it is essential that the hardware implementation minimize the burden to the processor of the communication system. Thus, a controller attached to the communication bus should not be burdened with a complicated, processor-intensive hardware and/or software interface for communication with other devices connected to the bus.