GM	WORLDWIDE ENGINEERING STANDARDS	General Specification Electrical/Electronic	PPEI 3.7 GMW8772

Platform to Powertrain Electrical Interface (PPEI) Specification
Serial Data Architecture Subsystem

1 Introduction

This standard defines the Serial Data Architecture between Platform and Powertrain.

1.1 Applicability.

The **GMW8762** PPEI (Platform to Powertrain Electrical Interface) Standard Specification includes: General Information, On-Board Diagnostics and Electrical Requirements and GMLAN Serial Data Signal Definitions and Framing for the following nineteen PPEI subsystems standard specifications:

1. **GMW8763** Power and Ground
2. **GMW8764** Four Wheel Drive/All Wheel Drive Controls
3. **GMW8765** Displays and Gauges
4. **GMW8766** Engine Power Management
5. **GMW8767** Starter Control
6. **GMW8768** Vehicle Theft Deterrent
7. **GMW8769** Cruise Control
8. **GMW8770** Cooling Fan Control
9. **GMW8771** Air Conditioning Compressor Control
10. **GMW8772** Serial Data Architecture
11. **GMW8773** Brakes and Traction Control
12. **GMW8774** Enhanced Evaporative Emissions and Fuel
13. **GMW8775** Exhaust After-Treatment
14. **GMW8776** Suspension Control
15. **GMW8776** Transmission
16. **GMW8778** Generator Control
17. **GMW8779** Post Collision Operation
18. **GMW8780** Power Take-Off and Fast Idle Control
19. **GMW8781** Vehicle Speed and Rough Road Sensing

Each of the nineteen PPEI subsystem standard specifications contains the hardware, serial data, algorithms and calibrations for the named subsystem.

The master PPEI document and all nineteen PPEI subsystem standard specifications are required to define the complete set of PPEI requirements.

2 References

Note: Only the latest approved standards are applicable unless otherwise specified.

2.1 External Standards/Specifications.

ISO 11898	SAE J1962
ISO 15031	SAE J2284

2.2 GM Standards/Specifications.

GMW3001	GMW8769
GMW3059	GMW8770
GMW3122	GMW8771
GMW3173	GMW8773
GMW7349	GMW8774
GMW8762	GMW8775
GMW8763	GMW8776
GMW8764	GMW8777
GMW8765	GMW8778
GMW8766	GMW8779
GMW8767	GMW8780
GMW8768	GMW8781

2.3 Additional References.

TL 01.58.2041-R5

3 Subsystem Requirements

3.1 Functional Overview.

Powertrain shall be interfaced with Platform via only one serial data link, which shall be high speed GMLAN. Powertrain shall not be a gateway between nodes on the link.

High speed CAN is part of the GM Local Area Network (GMLAN) communication strategy and has been approved for diagnostics by CARB. High speed GMLAN will be used to meet OBD/EOBD requirements for diagnostic serial data communication.

3.2 Hardware Overview.

The Platform and the Powertrain Electronics shall communicate via high speed CAN according to GMW3122 - GMLAN Dual Wire CAN Physical Layer Interface Specification, and GMW3104 - GMLAN Communication Strategy Specification. Reference the following serial data documents:

- ISO 11898 Road Vehicles - Interchange of Digital Information - CAN for High Speed Communication
- SAE J-2284 Dual Wire, High Speed CAN
- ISO 15031 – Road Vehicles – Communication Between Vehicle and External Equipment for Emissions-Related Diagnostics (for OBD/EOBD legislated diagnostics)
- GMW3110 – Enhanced Diagnostic Test Mode Specification (for non-legislated diagnostics)
- GMW3173 GMLAN Architecture and Bus Wiring Requirements
- The ECM shall contain one termination for the high speed CAN link. The termination shall be connected between CAN_H and CAN_L bus signals. The platform shall provide the other termination in a manner consistent with the electrical specifications in SAE J2284 and GMW3122.
- Each device on the high-speed link that does not contain a termination shall provide a serial data interface that includes two (2) connector pins per CAN bus signal (CAN_H and CAN_L, 4 pins total) that are adjacent and common to each other. Both pins per signal shall share EMC protection components. Refer to GMW3122 and GMW3173 for details.

The Powertrain Electronics and the Platform Immobilizer device shall be able to transmit and receive data on the high speed CAN link when the battery voltage at the controller input pins is 7.0 to 16.0 Vdc. (This implies that there is no requirement for serial data transmission during a low voltage crank event). In the range of 16.0 to 18.0 Vdc all devices shall be able to transmit and receive data sufficient to allow the engine to start and run and sufficient for functionality of all OBD diagnostic functions. These ranges result from the Voltage Requirements for Starting defined in GMW8767 Section 3.5.2 PPEI Starter Control Requirements. All devices shall be able to transmit and receive data on the high speed CAN link in the normal operating voltage range as defined in GMW8767 Section 3.5.1 PPEI Starter Control Requirements.

Note: The following paragraph states that the powertrain electronics shall not support two of the three methods for device wake-up identified in the GMLAN Strategy.

The Powertrain Electronics shall be capable of transmitting and receiving high-speed CAN frames when either the Accessory/Wakeup or Run/Crank signals are in the active state. The Powertrain Electronics shall be capable of communications within 100 ms of reaching either of these states. The Powertrain Electronics will not support wake-up via serial data traffic or wake-up using a dedicated discrete signal.

Platform shall provide a gateway from any other vehicle serial data link to the high speed CAN link if communication is required between devices connected to these links. This gateway shall synthesize the necessary input frames to Powertrain Electronics.

3.2.1 Block Diagram.

The following block diagram (Figure 1) depicts a typical mechanization for the serial data link. The electrical interface between Powertrain and Platform is the only standard defined.

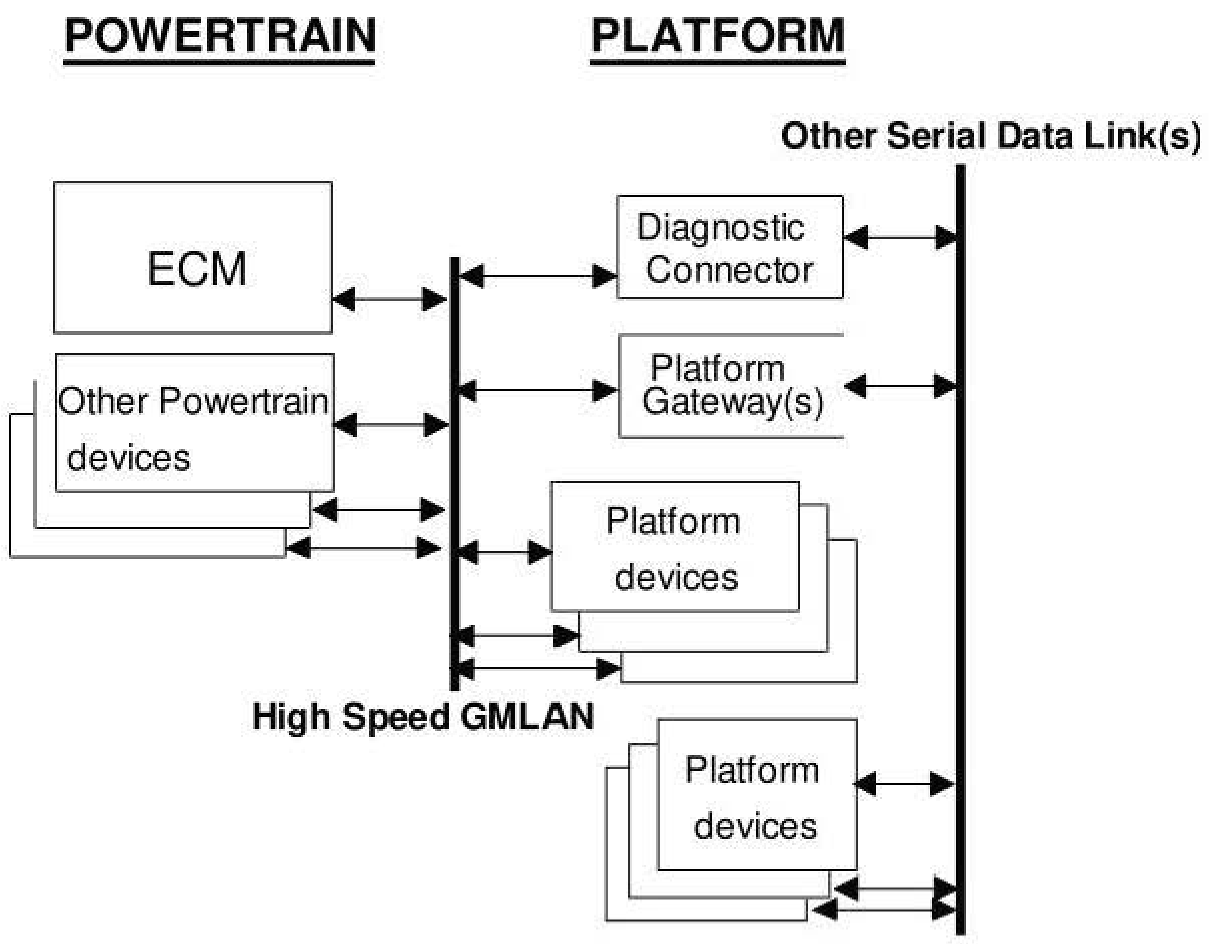

Figure 1: Serial Data Architecture Block Diagram

Table 1: Serial Data Architecture Serial Data Signals

Signal Name	Transmitter	Notes
Diagnostic Trouble Code Information Extended	Powertrain	Required

3.2.2 Diagnostic Connector.

The diagnostic connector shall conform to SAE J1962.

3.3 Interface Description.

3.3.1 High Speed CAN Data Link.

Reference GMW8762 Section 4 PPEI Serial Data Signal and Definitions and Framing for definitions of signals listed in Table 1.

3.3.1.1 GMLAN Signals.

GMLAN signals for the defined subsystem specifications shown in Section 1.1 are summarized in each respective GMW specification. The detailed definitions of GMLAN signals and GMLAN standard frames are contained in GMW8762 Section 4 PPEI Serial Data Signal and Definitions and Framing Requirements.

All GMLAN signals are defined in the GMLAN Signal Database.

Reference GMW8762 Section 4 PPEI Serial Data Signal and Definitions and Framing Requirements for definitions of signals.

There are several signals in PPEI that are optional (i.e., are not present on a vehicle if certain options are not present). An example is signals related to Traction Control, which would not be sent if ABS or Traction Control is not present. The details of which signals are not transmitted when a particular option is not present are located in the signal descriptions in PPEI GMW8762 Section 4 PPEI Serial Data Signal and Definitions and Framing Requirements. Note that even though a signal may be present on the vehicle (i.e., it is part of a GMLAN frame that is transmitted on the vehicle), it is possible that the signal may not actually be supported for a particular application. An example is the signals that are related to Traction Control on a vehicle equipped with an EBCM that does not have the traction control feature. The individual signal descriptions in GMW8762 Section 4 PPEI Serial Data Signal and Definitions and Framing Requirements indicate the data values that should be sent when a signal is present but not supported.

There are also several signals in PPEI that have the possibility of being transmitted by different devices depending on optional content. For example, most of the signals related to transmission status will be sent by the Transmission Control Module (TCM) on vehicles that have a TCM, but will be sent by the Engine Control Module (ECM) on vehicles that are not equipped with a TCM. This variation can even occur on a single platform (for example, TCM transmits the signals on a vehicle in the platform with an automatic transmission, while the ECM transmits those signals on a manual transmission vehicle). Both receiver and transmitter must support any configuration necessary to support this variability of transmitters.

3.3.1.2 GMLAN Network Management.

Network management on the high speed GMLAN serial data link shall be via the “Shared Input Activated Virtual Network” technique described in GMW3104. In particular, two shared input activated VN’s shall be defined whose activation criteria correspond to the status of the Accessory/Wakeup and Run/Crank inputs, respectively.

Powertrain electronics shall transmit all normal serial data signals whenever either of these VN’s is active. This implies that all Powertrain devices (ECM, TCM, and Transfer Case Control Module) MUST have both Accessory/Wakeup and Run/Crank hardwire inputs. Powertrain electronics shall expect to receive only those signals that it actually uses in the operating/power modes represented by the VN activation criteria. In general, Powertrain electronics will expect to receive all signals when the Run/Crank VN is active, and few (if any) signals when only the Accessory/Wakeup VN is active. The details of which signals are required in which VN’s are shown in GMW8762 Section 4 PPEI Serial Data Signal and Definitions and Framing Requirements.

Platform electronics shall use a shared input activated network management approach compatible with (but not necessarily identical to) the one described above. For example, platform electronics may only support a single virtual network that is the composition of the two VN’s described above. At a minimum, Platform shall transmit all signals required by Powertrain in the VN’s indicated in GMW8762 Section 4 PPEI Serial Data Signal and Definitions and Framing Requirements (i.e., from Powertrain’s perspective, the system shall operate as if it had the VN structure defined in the previous paragraph even if the actual network management structure is somewhat different).

3.3.1.3 Alive Rolling Count and Protection Value Requirements.

Certain critical signals require special handling within the serial data communication system. The following sections outline the requirements for two particular classes of signals, Alive Rolling Count Signals and Protection Value Signals.

3.3.1.3.1 Alive Rolling Count Signals.

3.3.1.3.1.1 General.

Alive Rolling Count signals are intended to protect against two types of faults:

1. A module's communication process continues (i.e., all periodic frames are sent) but the application software responsible for updating critical data is not executing (and thus the data is not being updated).
2. A module's communication process continues (i.e., all periodic frames are sent) but a communication controller fault prevents properly executing application software from updating the data transmitted on the communication link (and thus messages are being repeated with no update).

The reception of a message with a properly updated alive rolling count generally indicates that the message is not simply a repeat of a previous message but rather contains data that has been processed by the application software since transmission of the previous message.

Each Alive Rolling Count signal "covers" a certain set of PPEI signals or packets. The specific set of signals/packets covered by a particular Alive Rolling Count signal is specified in GMW8762 Section 4 PPEI Serial Data Signal and Definitions and Framing Requirements for that Alive Rolling Count signal.

3.3.1.3.1.2 Transmitter Requirements.

All Alive Rolling Count signals used in the PPEI shall be 2 bits in length and shall be of the type Unsigned Numeric. A properly operating transmitter shall update the value of the Alive Rolling Count signal every time a frame is transmitted, even if no other data in the frame has changed value. The count value shall be updated in the sequence 0, 1, 2, 3, 0, 1, 2, 3, etc. Upon startup, the initial value of the Alive Rolling Count signal shall be 0 (zero).

Alive Rolling Count signals shall be located in the same frame as the signals/packets that they cover. To ease processing burden, frames containing Alive Rolling Count shall use the periodic transmit model only (i.e., they should not use the event or event plus periodic transmit models).

The task of initiating an update of the Alive Rolling Count value shall be carried out by the application software module responsible for updating the data "protected" by the alive rolling count, and shall only be initiated after the data has been updated. The update process shall not be carried out exclusively by the communication handler or transport layer software modules. These requirements are intended to ensure that the Alive Rolling Count value will only be updated if the application software has also correctly updated the protected data.

3.3.1.3.1.3 Receiver Requirements – Determination of Alive Rolling Count Errors.

The receiver shall check the Alive Rolling Count value with every new received frame. An incorrect value (i.e., a value other than the expected value) shall cause the receiver to detect an **Alive Rolling Count Error**. In most cases, the data received in a frame with an Alive Rolling Count error is ignored. Refer to Section 3.3.1.3.3 for further information on error handling.

The determination of the "expected" value of an Alive Rolling Count signal in a frame shall be based on incrementing the value from the last time the frame was received (i.e., the alive rolling count value is continuously reset based on the value received in the previous frame).

The following example (Table 2) contains a sequence of received values that indicates the operation of this determination:

Table 2: Determination of Alive Rolling Count Errors

Alive Rolling Count Value	Expected Value	Error?
0	N/A	N/A
0	1	Yes
0	1	Yes
1	1	No
1	2	Yes
2	2	No
4	3	Yes
0	0	No

The following pseudocode describes the operation of Alive Rolling Code processing.

```
START  (Note:  Assumes value already stored in
            Expected_ARC_Value)
            ARC_Error = FALSE
WHILE  ARC Processing Required
            Wait for reception of new ARC frame
            New_ARC_Value = Received ARC Value
            IF
            New_ARC_Value = Expected_ARC_Value
            THEN
            ARC_Error = FALSE
            ELSE
            ARC_Error = TRUE
            Increment ARC error counters
            ENDIF
            Expected_ARC_Value =
            (New_ARC_Value + 1) mod 4
END WHILE
```

3.3.1.3.1.4 Alive Rolling Counter Initialization.

The processing of Alive Rolling Count values received immediately following the activation of a Virtual Network (i.e., at start up) or after a reset differs from the procedure described above. Due to the possibilities that different devices start up at different times, it is necessary to define the procedure by which receivers initially synchronize to the Alive Rolling Count of the transmitter. This procedure should not result in any errors under normal startup circumstances, but should ultimately detect errors that are actually present.

The general idea is that the receiver should synchronize to the Alive Rolling Count value that is initially received, and begin normal Alive Rolling Count processing. Data protected by the Alive Rolling Count signal should only be considered valid once the Alive Rolling Count has achieved synchronization and then subsequently has passed a normal Alive Rolling Count update check.

The following pseudo code describes the operation of Alive Rolling Count initialization.

```
            START
            Wait for reception of first ARC frame
            Old_ARC_Value = Received ARC Value
            Expected_ARC_Value = (Old_ARC_Value
            + 1) mod 4
            Begin normal ARC Processing
```

3.3.1.3.2 Protection Value Signals.

3.3.1.3.2.1 General.

Protection signals are intended to protect against faults that result in incorrect data being sent by a CAN controller. Such faults can arise from faults within the transmit buffer memory of the controller that would send out incorrect data, but would still generate a valid CRC (if, for example, the fault in the controller takes place before the computation of the CRC).

The reception of a message with a proper protection field usually indicates that there were no bit fault errors in the CAN controller. It does this by providing additional redundant values that may be checked to ensure integrity of the data.

There are multiple methods of protecting data from corruption within a CAN controller. The following paragraphs define the standard PPEI method. The existence of this standard protection method does not imply that PPEI may not use other methods on a case-by-case basis.

Each Protection Value signal "covers" a certain PPEI signal or packet. The specific signal/packet covered by a particular Protection Value signal is specified GMW8762 Section 4 PPEI Serial Data Signal and Definitions and Framing Requirements for that Protection Value signal.

3.3.1.3.2.2 Transmitter Requirements.

All protection signals in PPEI shall be equal in length to the signal or signals that they protect. The exception to this requirement on the length of protection signals is when a single bit signal has both a protection value and an alive rolling count (see below). In this case, the protection value shall be two bits in length. If more than one signal is protected by a single protection value then all protected signals shall be grouped into a packet. This grouping explicitly defines the bit order of the combined signals, reducing the possibility of implementation errors.

All protection signals shall be located in the same frame as the signal/packet that it serves to protect. To ease processing burden, frames containing a protection value shall use the periodic transmit model only (i.e., they should not use the event or event plus periodic transmit models).

For PPEI applications, if a signal has a protection value and a validity or failed signal then the protection value shall also include the validity/failed signal. The validity/failed signal shall be included in the same packet as the protected signal(s).

Unless otherwise specified, all PPEI signals that have protection values shall also have Alive Rolling Count values. These Alive Rolling Counts are used in the computation of the protection value, but do not increase its length (except in the case of single bit signals - see above)

The following examples indicates several cases:

- Protection signal to cover a single 8-bit signal, with an Alive Rolling Count signal. PPEI would define an 8-bit signal, a 2-bit Alive Rolling Count signal, and an 8-bit protection value.
- Protection signal to cover an 8-bit signal with a validity bit and an Alive Rolling Count signal. PPEI would define a 9-bit packet, a 2-bit Alive Rolling Count signal, and a 9-bit protection value.
- Protection signal to cover a single bit signal with an Alive Rolling Count signal. PPEI would define a 1-bit signal, a 2-bit Alive Rolling Count signal, and a 2-bit protection value.

The protection value associated with an n-bit signal/packet shall be computed using the following algorithm: Add the binary value of the protected signal or packet to the binary value of the Alive Rolling Count signal. This addition is done such that it produces an n-bit result (i.e., any carry bits generated as part of the addition are discarded). The protection value is the n-bit two's complement of the sum. The two's complement is computed by taking the bitwise inverse of the data, and then adding one. Again, the result is treated as an n-bit value (i.e., any carries generated are ignored).

The data used in the computation of the protection value is the binary number formed by the signal or the packet. This value may be different than the engineering units encoded by the signal that is protected. The binary value of a collection of signals in a packet is given by the concatenation of the individual signals in the order defined by the packet.

The following example demonstrates the computation of a protection value.

Signals to be protected:

A 14-bit packet, composed of three signals, with the following definition (see Table 3):

Table 3: Computation of a Protection Value

Signal	Length	Data Type	Range	Conversion
Example Powertrain Control Command	14	PKT	N/A	N/A
Example Powertrain Control Active	1	BLN	N/A	\$1 = True; \$0 = False
Example Powertrain Control Fast Mode Active	1	BLN	N/A	\$1 = True; \$0 = False
Example Powertrain Control Acceleration Request	12	SNM	-20.48 – 20.47 m/s^2	E = N * .01
Example Powertrain Control Command Protection	14	UNM	0 - 16383	E = N * 1
Example Powertrain Control Alive Rolling Count	2	UNM	0 - 3	E = N * 1

Assume that the values of the signals above are as follows:

Example Powertrain Control Active = True (binary 1)
Example Powertrain Control Fast Mode Active = False (binary 0)
Example Powertrain Control Acceleration Request = –2.0 m/s^2 (binary 1111 1110 1100)
Example Powertrain Control Alive Rolling Count = 2 (binary 10)

The binary value of the Example Powertrain Control Command packet would be

```
        1 0 1111 1110 1100      (hex $2FEC)
```

The sum of the packet and Alive Rolling count would be

```
        10 1111 1110 1100       Packet
+       00 0000 0000 0010       Alive Rolling Count
        10 1111 1110 1110       (hex $2FEE)
```

The two's complement would be computed by taking the inverse and adding one.

```
        01 0000 0001 0001       Inverse
+       00 0000 0000 0001
01 0000 0001 0010       Example Powertrain Control Command Protection  (hex $1012)
```

The transmitter shall update the Protection value every time the frame is transmitted. Obviously, the computation of the protection value shall use the updated value of the Alive Rolling Count signal.

3.3.1.3.2.3 Receiver Requirements – Determination of Protection Value Errors.

The receiver shall check the protection value with every new received frame. The check shall be performed using the identical algorithm used by the transmitter, basing the computations on the received value of the signal/packet and the Alive Rolling Count. An incorrect protection value (i.e., the received value of the protection signal is different than the protection value computed by the receiver) shall cause the receiver to detect a **Protection Value Error**. In most cases, the data received in a frame with a Protection Value error is ignored. Refer to Section 3.3.1.3.3 for further information on error handling.

3.3.1.3.3 Error Handling.

The previous sections defined two error conditions, an Alive Rolling Count Error and a Protection Value Error. If either or both of these conditions exist for a given received frame then the signals protected by the Alive Rolling Count and Protection Value is considered to have experienced a **Signal Verification Error** for that frame.

In most cases, data with a Signal Verification Error is ignored; however other application specific actions are allowed (accepting a "safe" state command such as cruise off even if the frame had a Signal Verification Error, for example). The details of how Signal Verification Errors are handled (i.e., whether the data is to be ignored, or if certain data may be accepted even in the presence of Signal Verification Errors) are specified in the individual signal or algorithm description sections of PPEI.

For various reasons, it is possible that a certain number of Signal Verification Errors will occur in an otherwise properly operating system. The simple frame-by-frame error handling described above will minimize the probability of misinterpreting the received data. Many applications, however, rely on consistent delivery of their operating data. As such, they may require a mechanism to determine conditions where Signal Verification Errors occur frequently enough to have the potential to cause issues with system operation. The following paragraphs define a general "sliding window" error checking procedure. The actual details of how a given set of signals handles these errors are specified in the signal or algorithm description section of PPEI (possibly by referring to the generic definition in this section). These specifications of error handling should make use of the terms defined in this section (i.e., Alive Rolling Count Error, Protection Value Error, Signal Verification Error, and Sliding Window Verification Error).

3.3.1.3.3.1 Sliding Window Verification Error Determination.

The receiver shall perform a "sliding window" check on signals covered by Alive Rolling Count and/or Protection Value signals. The existence of X Signal Verification Errors within a window of Y consecutive frames shall constitute a **Sliding Window Verification Error**. Typical values are X=3 and Y=10. The value of X is restricted by a requirement to detect "stuck bit" failures in the Alive Rolling Count parameter; this requires $X \leq Y / 2$ for Y even and $X \leq (Y+1) / 2$ for Y odd. The actual values of X and Y are specified in the signal or algorithm section of PPEI.

The following table (Table 4) gives an example of the determination of the existence of a Sliding Window Verification Error when X = 3 and Y = 10. The first column gives an index as to the number of received frames. The column marked "Signal Verification Error?" indicates whether or not there was a Signal Verification Error (either an Alive Rolling Count Error or a Protection Value Error) in the frame. The column marked "Window" indicates which frames are included in the "sliding" window of size Y = 10. The column entitled "Errors in Window" indicates the total number of frames that had Signal Verification Errors that are included in the window. Finally, the column marked "Sliding Window Verification Error?" indicates whether or not there have been X = 3 or more errors within the sliding window. For the purposes of this example, it is assumed that frames numbered from 1 – 9 have been received without any signal verification errors.

Table 4: Sliding Window Verification Error Determination

Number	Signal Verification Error?	Window	Errors in Window	Sliding Window Verification Error?
10		1 – 10	0	
11	Error	2 – 11	1	
12		3 – 12	1	
13		4 – 13	1	
14		5 – 14	1	
15		6 – 15	1	
16		7 – 16	1	
17		8 – 17	1	
18		9 – 18	1	
19	Error	10 – 19	2	
20		11 – 20	2	
21		12 – 21	1	
22	Error	13 – 22	2	
23		14 – 23	2	
24		15 – 24	2	
25		16 – 25	2	
26	Error	17 – 26	3	Error
27		18 – 27	3	Error
28		19 – 28	3	Error
29		20 - 29	2	
30		21 – 30	2	
31	Error	22 – 31	3	Error
32		23 – 32	2	
33		24 – 33	2	
34		25 - 34	2	

The following pseudocode defines the determination of X out of Y Sliding Window Verification Errors for a window size of Y and an error threshold of X.

Possible behavior on the detection of a Sliding Window Verification Error is to irreversibly place the associated function into a "safe" state (disengage the cruise control, for example) for the remainder of the driving cycle. Other application-specific behaviors are also allowed – the details of the handling of this type of error are described in the signal and algorithm definition sections of PPEI.

```
START
Error_Array[Y] = "False"
/* Zero-based array of Y elements of Booleans.  All Elements initially False */
Num_Frames_Received = 0
Errors_In_Window = 0
Sliding_Window_Verification_Error = "False"
WHILE Sliding Window Verification Processing Required
        IF
                Error_Array[Num_Frames_Received mod Y] = "True"
        THEN
                Errors_In_Window = Errors_In_Window – 1
        END IF
        WAIT for received Frame
        IF
                Signal Verification Error in received frame
        THEN
                Error_Array[Num_Frames_Received mod Y] = "True"
                Errors_In_Window = Errors_In_Window + 1
        ELSE
                Error_Array[Num_Frames_Received mod Y] = "False"
        END IF
        IF
                Errors_In_Window ≥ X
        THEN
                Sliding_Window_Verification_Error = "True"
        ELSE
                Sliding_Window_Verification_Error = "False"
        END IF
        Num_Frames_Received = Num_Frames_Received + 1
END WHILE
```

3.3.1.3.3.2 Sliding Window Verification Error Recovery.

Certain applications may desire to recover from the detection of a Sliding Window Verification Error within a single driving cycle. The details of this recovery (if used at all) are subsystem specific. This section presents a generic set of recover requirements. The detailed description of how a subsystem recovers from Sliding Window Verification Errors is described in the signal and algorithm sections of PPEI (possibly by referring to this generic description).

The generic method to recover from a Sliding Window Verification Error (if such a recovery is allowed) is to consider the subsystem faulted (and thus operating in a default state) until the receiver has received a number frames equal to two times the size of the sliding window without any Signal Verification Errors. For example, if the parameters for the determination of the sliding window were X=3 and Y=10, the system would need to receive 20 consecutive frames with neither an Alive Rolling Count Error nor a Protection Value Error before re-enabling normal operation of the system. During this time, the system continues to detect Signal Verification errors; if any occur, the counter is reset and the process begins again. Note that the system will cease to indicate a Sliding Window Verification Error long before it meets this recovery criterion.

3.3.1.3.3.3 Supervision Error Recovery.

Although not an error of the same type as those previously defined (i.e., Alive Rolling Count Error, Sliding Window Verification Error, etc.), recovery from a signal supervision error requires special handling. The details of this recovery (if used at all) are subsystem specific. This section does not require recovery from supervision errors. This section presents a generic recovery mechanism that may be used if an application desires to recover from such errors. The detailed description of how a subsystem recovers from signal supervision errors is described in the signal and algorithm sections of PPEI (possibly by referring to this generic description).

In general, recovery from a GMLAN supervision error should occur in a manner similar to procedures already described. For example, it is likely that the first reception of a rolling count signal following recovery from a supervision error would be inconsistent with the last value received before the supervision failure. In order to prevent the detection of a rolling count error when there really was none, the system should resynchronize the alive rolling count value in the same manner as the alive rolling count is synchronized at the initialization of a Virtual Network (refer to Section 3.3.1.3.1.4).

In addition, it is possible that certain types of faults that may cause supervision errors might experience periods of instability on recovery. Because of this, recovery should only take place after a period of correct continuous operation. The technique used should be similar to the technique described for sliding window verification error recovery (i.e., require a defined number of messages to be received with no Signal Verification Errors before the system is considered to have recovered.

3.3.1.4 Virtual Device Availability Signals.

PPEI is intended to allow applications where certain serial data communication signals do not originate from the High Speed GMLAN serial data link. In order for these signals to be received by Powertrain, they must pass through a gateway device, which receives the signals on their original communication link and retransmits them onto the High Speed GMLAN data link.

Normal GMLAN signal supervision is performed on the presence of a signal, i.e., as long as a signal continues to be received it is presumed that no failsoft action needs to take place. When a gateway lies between the original source of the data and the receiver, it is possible that the original transmitter of the data could fail but the gateway continues to operate normally. In this circumstance, messages would continue to be sent by the gateway, and thus receivers of the information would be unable to detect the failure of the original transmitter because they would not detect a supervision error.

GMLAN offers a procedure to rectify this situation, known as "supervision by value". In this mechanism, the data value of a signal, rather than the presence of a signal, determines the supervision state of another signal. Specifically, signals known as Virtual Device Availability (VDA) signals are used to indicate when the gateway has detected a supervision failure with the original transmitter of the signals.

Virtual Device Availability Signals are all single bit signals of type ENM, and have possible data values of "Unavailable" and "Available". These signals are generated by the gateway on detection of loss of supervision with the original transmitting device.

Not all signals transmitted by the gateway have associated VDA signals. In general, a VDA signal is not required if the any of the following conditions are true:

a. The signal represents an input that is required (for speed reasons, for example) to be read by the gateway. An example of such a signal is cruise control switch status.

b. The signal represents the results of an arbitration process executed by the gateway, even if this process is based on information received from another serial data link. It is expected that the platform arbitration process will take the necessary steps on loss of communication with the devices that provide input to the arbitration process.

c. The receivers of the signal all have a common view of the appropriate default action in the event of a serial data failure. In this case, the gateway can simply substitute the proper default value when a supervision error is detected.

d. Signals that do not meet any of these criteria (i.e., they do not originate in the gateway and the receivers need to actually detect the fault condition with the original transmitter) should have an associated Virtual Device Availability signal generated by the gateway.

e. Receivers of a signal with an associated VDA signal still need to supervise the signal by the normal "supervision by presence" mechanism – this allows detection of failures of the gateway. In addition to this supervision, the receiver needs to use the VDA signal as part of its failsoft strategy.

Note that the previous discussion has been focused on VDA signals associated with Platform signals received by Powertrain. VDA signals may also be required for Powertrain to Platform communications. These signals, however, would be sent from the gateway to another platform device, and therefore do not cross the PPEI interface and are not described in this document.

3.3.2 Calibrations.

The PPEI subsystem definition shall define a serial data signal or calibration(s) for communicating Platform information to Powertrain or Powertrain information to Platform. The serial data or calibration usage shall be determined by the following guidelines:

a. Data Identifier (DID) Usage Requirements

1). DIDs may be utilized to program "non-controlled" information, which will typically be updated in the service environment, emissions test or by an aftermarket upfitter.

2). DIDs may be utilized to program "non-controlled" information in the vehicle electronics, which are not bound by OBD II compliance regulations.

b. Calibration Usage Requirements

The PPEI subsystem definition shall define a calibration or calibration table in the Platform or Powertrain electronics based on the following criteria:

1). Calibrations shall contain data, which does not change dynamically during the normal operation of the vehicle.

2). The PPEI subsystem definition shall define calibrations, which reside in the Platform electronics, but are owned and calibrated by Powertrain, or jointly owned and calibrated by Powertrain and Platform.

3). The PPEI subsystem definition shall define calibrations, which reside in the Powertrain electronics, but are owned and calibrated by Platform, or jointly owned and calibrated by Platform and Powertrain

4). Calibrations which provide specific subsystem or vehicle configuration information (i.e., "option present" and "system type") shall reside in both Platform and Powertrain controllers that require the information unless:

(a) Extreme proliferation of calibration part numbers will result, where two or more electronic modules in the vehicle require the same information. In this case, the information may reside (be calibrated) in one module and transmitted via serial data to any other module(s) that required the data.
   (1) OBD-II related calibration data shall reside in an OBD compliant controller and transmitted via serial data to other modules as required.
   (2) Non-OBD II related calibration data shall reside with the natural owner of the information and transmitted via serial data to Platform and Powertrain as required.

(b) When non-OBD II related aftermarket re-calibration is required, DIDs may be utilized. DIDs shall be partitioned to the natural owner of the information and transmitted via serial data to Platform and Powertrain as required.

c. Serial Data Usage Requirements

The PPEI subsystem definition shall define serial data signals or messages in the Platform or Powertrain electronics based on the following requirements:

1). Serial data signals shall be used to communicate data, which changes dynamically during the normal operation of the vehicle.

2). Serial data shall be used to communicate common calibration data utilized by two or more electronic modules throughout the vehicle. The intent of this requirement is to eliminate the need to support a large number of calibrations in several vehicle modules, which contain identical information. Note: In general, serial data **shall not** be used as the source for vehicle configuration information, unless this configuration data can change dynamically during normal vehicle operation.

(a) Modules receiving configuration information shall determine their own default/fail-soft strategy.

(b) Received serial data information may be learned when received.

(c) Most recently learned information may be retained/stored over ignition cycles.

(d) Once information is learned, it may be latched until a battery disconnect or a "reset" service procedure is performed

(e) The descriptions of serial data used to communicate common calibration data shall reference the associated calibration by name.

3.3.3 Accessory/Wakeup.

This input to the Powertrain Electronics is used in the GMLAN Network Management (refer to Section 3.3.1.2). For details of the input refer to GMW8763 Section 3.3.2 PPEI Power and Ground Subsystem Accessory/Wakeup Requirements.

3.3.4 Run/Crank Relay Output.

This input to the Powertrain Electronics is used in the GMLAN Network Management (refer to Section 3.3.1.2). For details of the input refer to GMW8763 Section 3.3.3 PPEI Power and Ground Subsystem Run/Crank Relay Output Requirements.

3.4 Failure Modes and Diagnostics.

Each unit on the serial data link shall provide its own diagnostics and failsofts to survive serial data link failure.

In order to facilitate the proper setting of diagnostic trouble codes against communication faults, all devices that receive serial data frames that originate from the other side of the PPEI boundary (i.e., Powertrain devices receiving frames transmitted by Platform, and Platform devices receiving frames transmitted by Powertrain) shall be configurable as to the relationship between the physical transmitter of the frame and the frame itself. For example, an ECM receiving a platform frame containing wheel speed data shall be configurable as to the physical device (EBCM, Gateway, etc.), which transmits the frame. In certain cases, this configuration may need to be based on option content of the vehicle (presence or absence of ABS, for example) and thus may need to be done on a vehicle by vehicle basis (at End of Line configuration, for example).

Failsoft behavior of signals that have an associated Virtual Device Availability Signal shall take into account the value of the VDA signal, but shall also still supervise on presence. Refer to Section 3.3.1.4 for more details.

Each unit shall support transmitting of DTC information via the GMLAN signal Diagnostic Trouble Code Information Extended. Refer to GMW8762 section 4.2.5.1, Diagnostic Trouble Code Signal Rationale.

3.5 Electrical Characteristics.

3.5.1 High Speed CAN Data Link.

The termination in the ECM shall be a split termination per GMW3122.

3.5.2 Accessory/Wakeup.

Refer to GMW8763 Section 3.5.5 PPEI Power and Ground Subsystem Accessory/Wakeup Requirements.

3.5.3 Run/Crank Relay Output.

Refer to GMW8763 Section 3.5.6 PPEI Power and Ground Subsystem Run/Crank Relay Output Requirements.

4 Algorithm

Not Applicable.

5 Provisions for Shipping

Not Applicable.

6 Notes

6.1 Glossary

None.

6.2 Acronyms, Abbreviations, and Symbols.

See GMW8762 Appendix Section A.3

7 Additional Paragraphs

7.1 All materials supplied to this specification must comply with the requirements of GMW3001, **Rules and Regulations for Materials Specifications.**

7.2 All materials supplied to this specification must comply with the requirements of GMW3059, **Restricted and Reportable Substances for Parts.**

8 Coding System

This specification shall be referenced in other documents, drawings, VTS, CTS, etc. as follows:

GMW8772

9 Release and Revisions

9.1 Release. This general specification originated in June 2003; approved by The Global PPEI Core Team in December 2003 and initially published in February 2004 for the Global PPEI Version 3.4.

9.2 Revisions.

Rev	Approval Date	Description (Organization)
A	Aug 2004	Global PPEI Version 3.5 Release.
B	Jul 2005	Global PPEI Version 3.6 Release.
C	Mar 2006	Global PPEI Version 3.7 Release.

Appendix A

The following are approved Change Requests (CRs) for the Global PPEI Version 3.7 Release that impacted the GMW8772 Serial Data Architecture Subsystem:

Sections Changed	Description of Changes	Rationale/ Authorization
	No changes were made to GMW8772 Serial Data Architecture Subsystem for the Global PPEI Version 3.7.	

Deviations

None.