PanelPilot ACE

Design Studio | **CAN bus Protocol** | User Guide

PanelPilot ACE

Electronic Design & Manufacture

Hardware, Software and IoT Capabilities Under One Roof

Are you looking for a genuine one-stop-shop for your product design and manufacturing requirements? Or perhaps you need just one element of the product mix?

From electronic and mechanical hardware design to full custom Cloud platforms, smartphone Apps, local-built prototypes to high volume Asian production, Lascar brings your ideas to life.

Design · Prototype · Sourcing · Manufacture · Supply

Core Competencies

Displays and Display Integration

We've been designing and integrating smart displays for five decades. Our display capabilities include capacitive, resistive and e-paper technologies.

IoT & Wireless Solutions

Hardware is just the first step. Our IoT team can create fully fledged Cloud platforms, Apps and supporting software.

Contents

Contact us for more PanelPilotACE products, services and support:

UK & Europe
www.lascarelectronics.com
sales@lascar.co.uk
+44(0)1794 884567

Americas
www.lascarelectronics.com
us-sales@lascarelectronics.com
(814) 835 621

Asia
www.lascarelectronics.com
us-sales@lascarelectronics.com
(814) 835 621

PanelPilotACE Design Studio **CAN bus Protocol** User Guide

1. Introduction

A CAN bus (Controller Area Network) is a message-based communications protocol. It was originally used within the automotive industries. CAN bus allows microcontrollers and devices to communicate with each other without the need for a host PC.

As well as the Automotive industry, CAN bus can be used in many other areas including Aviation, Industrial Automation and Mechanical Control, Building Automation and Medical Equipment.

The CAN bus Protocol has been added to the SGD 70-A PanelPilotACE colour capacitive touch display.

The set-up and control of PanelPilotACE devices with CAN bus added is implemented through the PanelPilotACE Design Studio.

The CAN bus Protocol has been added to Design Studio software version 4.0.1.4050 onwards.
This and many other PanelPilotACE resources can be downloaded free from:

www.lascarelectronics.com/panelpilotace-university

1.1 SAE J1939 CAN bus Standards

The PanelPilotACE CAN bus adapter has been created to comply with the Society of Automotive Engineers (SAE) set of standards, specifically SAE J1939 and the following:

i) J1939-21 Data Link Layer

ii) J1939-71 Vehicle Application Layer

iii) J1939-81 Network Management

SAE Standards J1939-73 Application layer “Diagnostics” and J1939-74 Application Configurable Messaging are NOT supported by the PanelPilotACE CAN bus.

Standard J1939 defines a device connected and operated via CAN bus as an *Electronic Control Unit* (*ECU*) or node.

J1939-21 uses *Parameter Group Numbers* (*PGN*) and *Suspect Parameter Numbers* (*SPN*). Each SPN specifies the properties of a single parameter such as data length, resolution, offset, range and unit. PGN specifies which SPNs belong to it. J1939-21 also provides a user defined PGN range named “Proprietary A”, “Proprietary A2” and “Proprietary B”

J1939-71 specifies that all PGN and SPNs can be used in a vehicle.

PanelPilotACE CAN bus J1939 Elements provide a flexible way to create SPNs and PGNs. Users can create any PGN or SPN covered by J1939-71 and J1939-75 (Application Layer - Generator Sets and Industrial) or Proprietary PGN.

Data can be regularly reported by an ECU or by a Request PGN from another ECU.

Commanding ECUs can also send queries to other ECUs to check the support feature. Users can create PGNs by using the PanelPilotACE CAN Frame Builder in Design Studio. This will automatically setup the response of the Request PGN and acknowledge available features.

Each ECU has its own 8 bit address and a 64 bit ECU Name. Public, preferred addresses for ECUs can be found in the J1939 Appendix that is available to view at www.sae.org.

The preferred address is the default one used by ECUs on a network however, conflicts can occur. Standard J1939-81, includes a procedure to resolve these conflicts (view the tutorials in section 11 & 12).

The PanelPilotACE J1939 CAN bus Element also provides the ability to reconfigure it's address according to a "Commanding Address" message. However, note that it is not able to issue this message.

Each PGN contains 8 bytes. A PGN with a data length of more than 8 bytes will be carried by a transport protocol message as defined by J1939-21. Both Broadcast and RTS/CTS transport protocols are supported.

2. Getting Started with CAN bus

2.1 Installing the S70-CAN Adapter

The S70-CAN adapter is available as an Add-on board for the PanelPilotACE SGD 70-A model. The following are the instructions for installing the adapter.

1. Make sure the SGD 70-A is powered-off.
2. Plug the S70-CAN Adapter into the SK2 Pin Header on the reverse of the SGD 70-A device (see Figure 1).

The CAN bus adapter can be wired point to point or by multi-drop wiring.

An example of wiring an S70-CAN to a DB9-F connector can be seen in Figure 3.

Figure 2 shows a multi-drop wiring to system with multiple ECUs or SGD-70-A.

The PanelPilotACE SGD 70-A with S70-CAN Adapter.

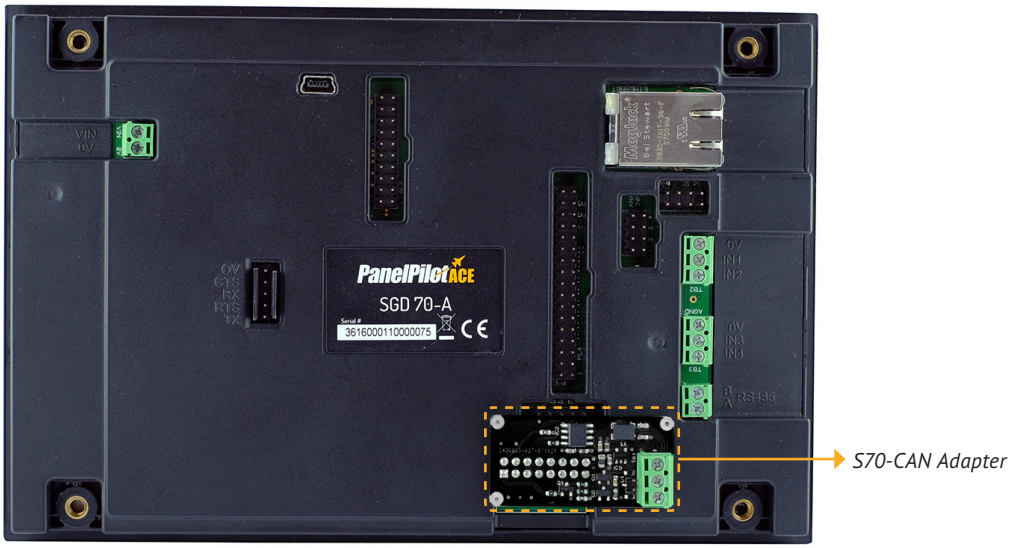

Figure 1. Plugging in the S70-CAN Adapter.

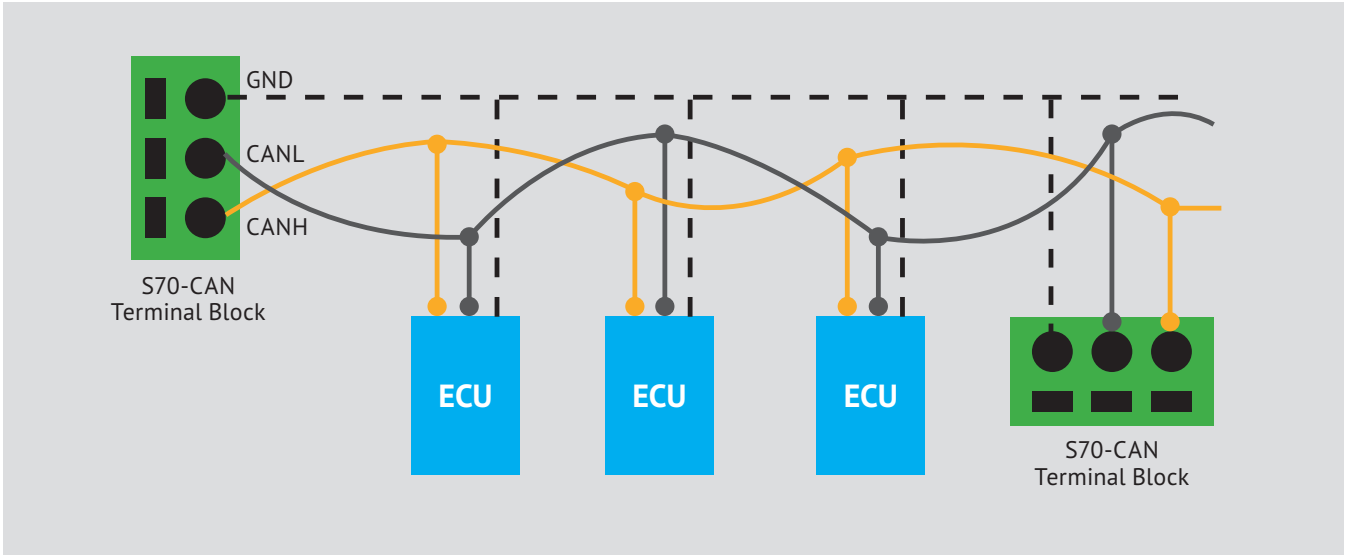

Figure 2. Multiple ECU connections diagram.

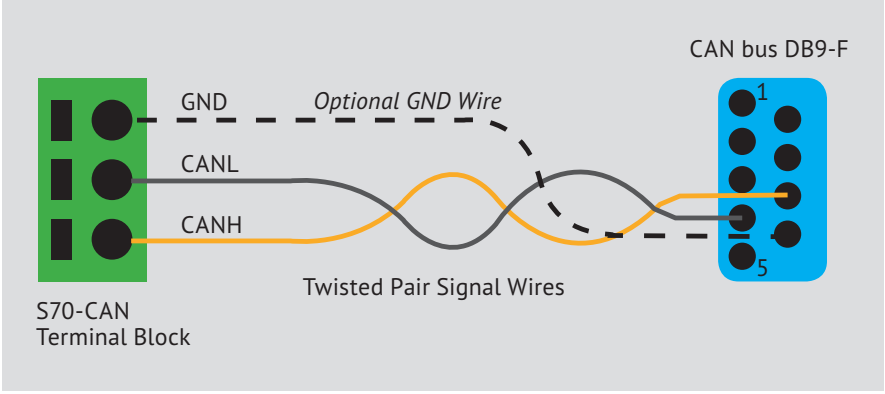

Figure 3. S70-CAN to DB9-F wiring diagram.

3. Design Studio Elements for CAN bus

The PanelPilotACE Design Studio contains two *Elements* and one *Project Variable* for a user to enable the CAN bus Protocol.

1. **CAN bus Device Hardware Element**
   This Hardware Element includes properties to control the S70-CAN hardware port and setup the ECU information.

2. **CAN Frame Builder Function Element**
   This Function Element provides the construct service. It can create J1939 CAN frame data encode, data decode and mapping to other Design Studio resources such as Project Variables or Element Properties.

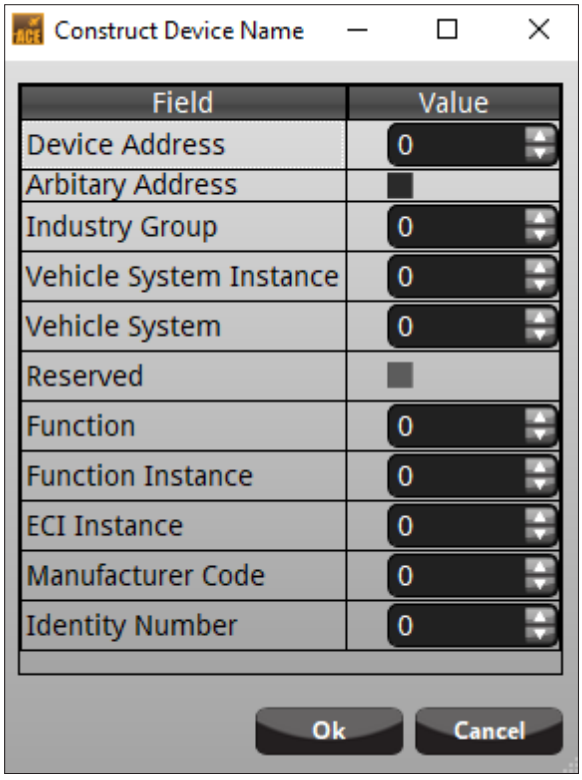

Figure 5. The ECU Construct Name dialogue box.

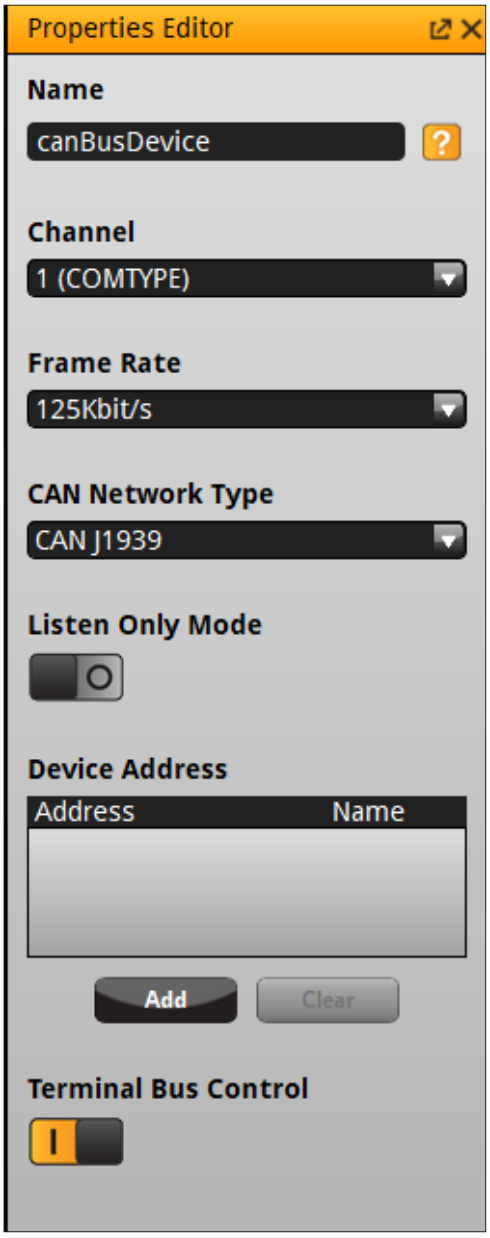

Figure 4. The Properties Editor showing CAN bus Device Properties.

3. **CAN Project Variable**
   This Variable provides a flexible way to construct CAN frame data in terms of PGNs and SPNs. It can be used to create a simulated ECU (Provide Service) or to remote control another ECU (Request Service).

4. CAN bus Device Hardware Element Properties

The configuable properties of the CAN bus Device Element can be found in the Properties Editor within the Design Studio. The Properties Editor is located in the top right-hand side of the Design Studio workspace (see Figure 4). Table 1 describes which of the values and fields within the Properties Editior that can be configured.

Property Name	Description
Name	Element Name
Channel	S70-CAN includes a signal CAN bus port. This is always Channel 1
Frame Rate	The Frame Rate can be set as 125, 256, 512 or 1 Mbits. All devices using CAN bus use the same frame rate. *Note: Do not connect a PanelPilotACE device to an active CAN bus with an unknown frame rate. Mismatched frame rates may cause network nodes to switch to passive or bus-off mode.*
CAN Network Type	This allows selection of the version of the CAN bus Protocol to be used. *Note: Only CAN J1939 Protocol is functional at this time. Open CAN and Native mode are reserved for future expansion.*
Listen Only Mode	Every ECU controller connected to a CAN bus will process all data frames on the bus and de-assert the ACK bit if any controller is found with errors on a data frame. The original controller will resend data frames until the accumulated error count exceeds the threshold that activates “Passive” or “Bus-off’ mode. A user can configure the controller to operate in “Listen Only Mode”. It does not de-assert the ACK bit if an ECU is just used as a monitor or to display an SPN parameter (i.e. read only).
Device Address	Each ECU on a network will have at least one 64 bit Name and one address assigned to it. A complex ECU can have multiple addresses and names. The Device Address, PGN and Message Priority form the CAN ID. This affects the message priority. Addresses can be allocated statically or dynamically using the “Address Claim” arbitration process. An address with a higher priority is dependant on the 64bit Name (small values on the device address have higher priority). Device Addresses are reconfigurable in runtime. This is why it is sometimes known as the Preferred Address. The Device Address can be reconfigured using the following steps: ● In the Properties Editor, click the Add button below the Device Address field. This will add an Emulated ECU to the Element. Double-click the field for the Construct Device Name dialogue box to appear. The “Arbitrary Address” field determines whether the address is configurable or fixed. When the “Arbitrary Address” field is set to True the PanelPilotACE will retry a new address using the address claim process if the address is already taken by something with a higher priority. If the PanelPilotACE fails to claim a new address then it will report an Address Claim Fail Error (see section 10 for error handler details). See J1939 Appendix & J1939-81 for further details.
Bus Terminate Control	The S70-CAN Adapter hardware provides the bus termination. The Bus Terminate Control Property can control the Bus Termination Enable/Disable. If a PanelPilotACE is the last ECU on a bus wire and the network wiring is long, terminating the bus can improve signal quality.

Table 1.

5. The CAN Variable

The J1939 data frame structure called a Protocol Data Unit (PDU) is displayed in Figure 6.

It shows the PDU data structure. The Priority, Extended Data Page, Data Page, PDU Format (PF) and PDU specific (PS) formatted the 29 bits extended.

The CAN ID is used in the CAN data link layer. PF, PS/Group Extension (GE) create the PGN.

There are two PDU formats PDU1 and PDU2. PDU1 has a PF value of less than 240. and the PS bytes store the Destination Address (DA). PDU2 has a PF value of more than or equal to 240 and PS bytes the Group Extension (GE).

Each PGN is defined by the individual parameter specified in terms of SPN. The data field itself does not carry any information or properties for decoding raw data such as resolution or offset. It only carries the parameters value.

For example, if a PDU with CAN ID 0x18FE0710, Data field is 0xE0, 0x2E, 0xC8, 0x32, 0xFF, 0xFF, 0xFF, 0xFF.

PF = 0xFE, Since PF > 240, the second byte is GE not SA and the value is 0x07.

Source Address (SA) = 0x10.

Therefore, the PGN = 0xFE07= 65031.

According to specification, PGN 65031 have 2 SPNs, SPN 2433 and SPN 2434.

SPN 2433 is Right Manifold Exhaust Gas Temperature a 16 bit, resolution 0.03125, offset -273°C. SPN 2434 which is the left side temperature.

J1939 PDU						
P	EDP	DP	PF	PS	SA	- - - - - Data Field
3	1	1	8	8	8	0 - 64

Key: P = Priority, EDP = Extended Data Page, DP = Data Page, PF = PDU Format, PS = PDU Specific, SA = Source Address.

Figure 6: J1939 PDU

SPN 2433, SPN 2434 Data Value = 16 bit raw data * 0x03125 + (- 273) unit in °C.

The first 2 data bytes belong to SPN2433, value 0x2EE0 decoded value is 102°C. The next 2 bytes belong to SPN2434, value 32C8 decoded as 133.25°C.

CAN Variables should be used to store PGN and SPN information.

A user can search the J1939 spec to locate the parameter required. These are to send or receive, find the SPN and PGN and input to the CAN bus variables.

Alternatively a user can create their own “Proprietary” PGN. Figures 7 and 8 show examples of PGN 65169 and SPN as found in the spec.

pgn65169 - Fuel Leakage - FL			
Transmission Repetition Rate		1s	
Data Length		8 bytes	
Data Page		0	
PDU Format		254	
PDU Specific		145	
Default Priority		7	
Parameter Group Number		65169 ($00FE91_{16}$)	
Bit Start Position/Bytes	Length	SPN Description	SPN
1.1	2 bits	Fuel Leakage 1	1239
1.3	2 bits	Fuel Leakage 2	1240

Figure 7. PGN65169 data structure

spn1239 - Fuel Leakage - 1 - status signal indicates fuel leakage in the fuel rail of the engine. The location can be either before or after the fuel pump.	
00 - no leakage detected	
01 - leakage detected	
Bit Length	2 bits
Type	Status
Suspect Parameter Number	1239
Parameter Group Number	[65169]
spn1240 - Fuel Leakage - 2 - status signal indicates fuel leakage in the fuel rail of the engine. The location can be either before or after the fuel pump.	
00 - no leakage detected	
01 - leakage detected	
Bit Length	2 bits
Type	Status
Suspect Parameter Number	1240
Parameter Group Number	[65169]

Figure 8. SPN1239 and SPN1240 meanings

The CAN Variable Properties found in the Properties Editor are listed in Table 2 (below).

Property Name	Description
Name	Variable Name
PGN	Parameter Group Number
Priority	Message default priority
Source Address	If this address is matched to the Preferred Device Address in the CAN bus Device Element, then the PGN will be processed as an out-going PGN and used to Encode the CAN frame. If a CAN Frame Builder links to this CAN variable it can send a CAN Frame to another device. If this address is not matched in CAN bus Device Element then the PGN will be processed as an incoming PGN and used to Decode CAN frame. If a CAN Frame Builder links to this CAN variable it can capture CAN frames from another device with source address matched.
Total Length	If the length is larger than 8 bytes then the PanelPilotACE device will automatically process the data using a Transport Message.
SPN List	This stores the SPN information belonging to the PGN. Click the "Add" button to activate a blank SPN, or click the SPN row to edit it. Figure 8 shows the dialogue box to add and edit SPNs. *Note: Input order of the SPN in the list must match the J1939 spec.*

Table 2.

The SPN Editor Dialogue Box shows SPN Properties that can be configured. Table 3 contains a description of these properties.

Property Name	Description
SPN	Suspect Parameter Number
Byte Positon	SPN data byte start position within the PGN. Start from 1
Bit Position	SPN data bit start position within the PGN. Start from 1
Length	Data size in number of bits
Offset	Date value offset
Range	Data value range
Scale	Scale between integer value and actual value of the parameter
Reverse	Checked means the scale value is inverse value
Unit	Unit field only for reference, PanelPilotACE do not use this field.

Table 3.

In J1939 CAN Frame, the real numbers are not Encoded/Decoded in the IEEE floating point data format. It uses "Resolution" to scale integer values into a real numbers with decimal places.

If the Resolution Number has several decimal places, you can check the "Reverse" check box. This uses an inverse value to keep the Resolution Property as an integer and does not lose decimal places.

e.g. SPN1135 resolution can either input as 0.03125 or 32 with the Reverse check box enabled.

Note: Users can also set the scale as 1 and use a Maths Builder or a Logic Builder to process the parameter externally.

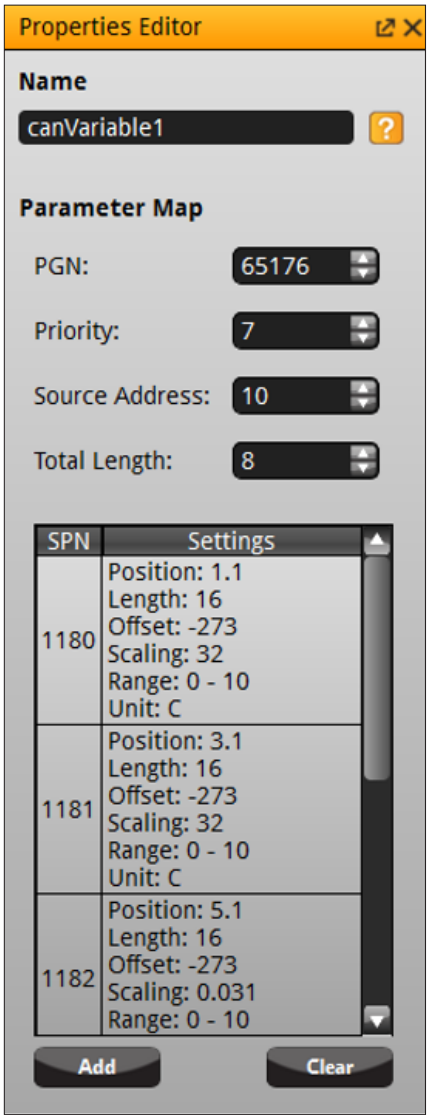

Figure 7. CAN Variable Properties

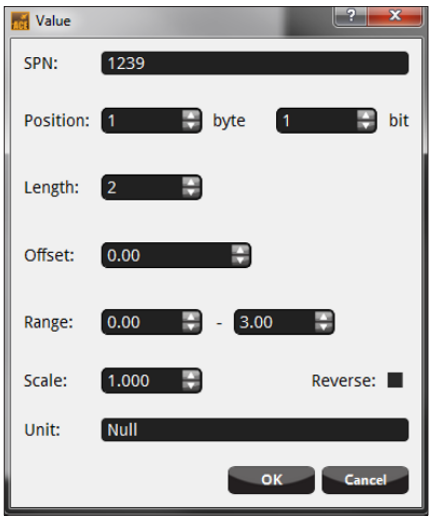

Figure 8. SPN Editor Dialogue Box

6. CAN Frame Builder

The CAN Frame Builder creates the connection between a CAN Variable and other PanelPilotACE project resources such as Project Variables, Element Properties or Actions (see Figure 9). A description of the available properties for this Element are listed in the Table 4 below:

Property Name	Description
Name	CAN Frame Builder Name.
CAN bus Port	Which CAN bus Port / CAN bus Device is selected for use.
CAN bus Variable	Which CAN bus Variable is being used. CAN Frame Builder will use the CAN bus Variable PGN and SPN information to Encode/Decode the CAN Frame.
Destination Address	This is the Target device address.
Record Count	A PGN can be a record type. Each record contains 1 set of SPNs. If the PGN is not a record type, set the record count to 1. If the PGN is a record type, configure the PanelPilotACE device with the number of records it is required to process. If the PGN is set as "Incoming" it will only process the number of records it has been configured for. For example, if the device receives 10 records, but the record count is set to 4, it will only process the first 4 records. The remaining 6 records will be ignored. If the PGN is set as "Outgoing", the record count sets the exact record write. When the record count is set to more than 1, more properties will appear in the Property Editor as seen in Figure 10.
Enable Count Field	Set "Enable Count Field" to true if the PGN has a first SPN to store the record count.
Enable FIFO Access	A PanelPilotACE device does not have array or record type variables, therefore PanelPilotACE will not process all records. Set the "Enable FIFO Access" to choose either process start from first or last records. Enable FIFO is also a Runtime Property; a user can Enable/Disable FIFO access using set rule action.
Parameters/Data – CAN Data Field	Configure Resource link to SPN. After selecting the CAN bus Variable, it will automatically check the number of SPNs contained in the PGN and reserve the same amount of un-configured links as "?". Click the "?" and select the resource that link to SPN. The first item will links to first SPN in the CAN bus variable. *Note: If the record count is 2 and the PGN has 4 SPN as a record, it will reserve 8 CAN Data fields and a total of 8 resources to link.*
Update Frequency	Some ECU (e.g. Engine, Throttle Position) will keep exchanging data with a very high repetition rate (e.g. 20-50ms). If you process all the data and perform screen updates, the screen may overload the PanelPilotACE CPU. Update Frequency controls how many repeated PGN captures will trigger the Post Action. Update and perform a Post Action. e.g. Set update frequency to 5 means update the project Variable/Property value once after target PGN captured 5 times.
Post Action	Action taken when a target PGN is captured or transmitted.
Error Handler	Select which Error Handler Element will process the error generated by this CAN Frame Builder.

Table 4.

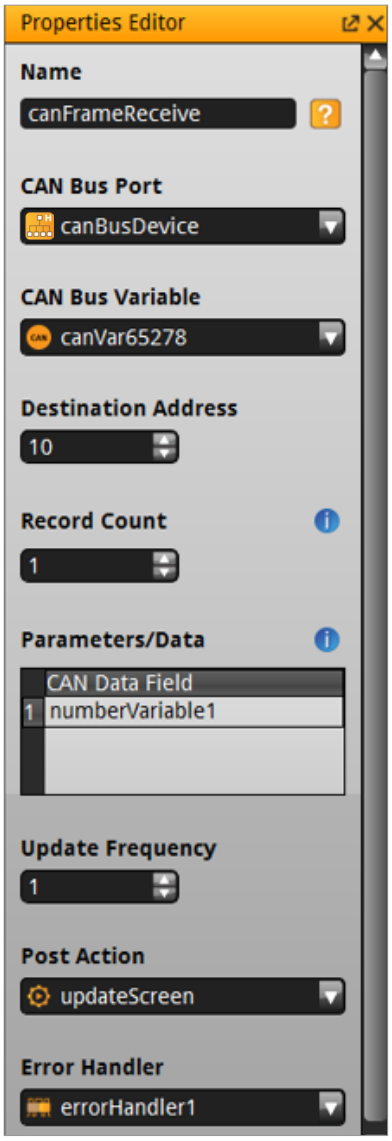

Figure 9. CAN Frame Builder Properties

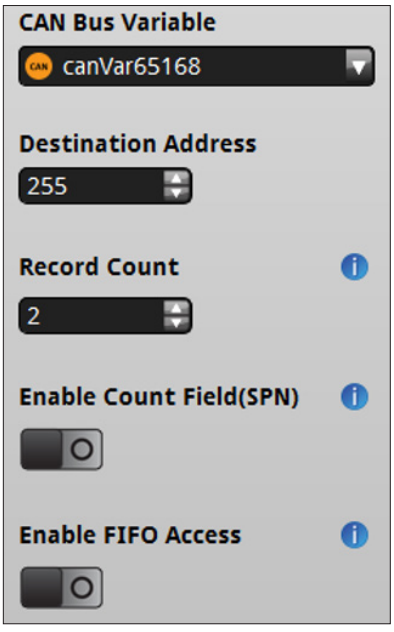

Figure 10. More CAN Variable Properties

7. Runtime & Error Handling

ECU Hardware controllers include an Error Counter to detect transmission faults. The Counter will reduce when the transmission returns to normal. If the Counter exceeds 127, the device will enter an Error Passive State. If the Counter exceeds 255, the device will enter a Bus-Off State and will stop all transmissions until the controller has been reset.

7.1 Active State

Transmissions will continue when the controller is in a Passive State.

The CAN bus Device Element includes a runtime property called "Active State" that monitors the status of the controller. The status is either "Active" or "Passive".

A user can add an Element Property Trigger to capture any status changes (see Figure 11). It is possible to configure an action in a Property Trigger to update a Visual Element to indicate the status as an indication of signal quality.

7.2 Comms Error Handler

The Communication Error Handler can be configured to manage CAN bus errors.

Figure 12 shows an example of this set-up in the Property Editor. The properties have been configured as shown in the Table 5.

Property Name	Description
Name	errorHandler
Communication Port	Select CAN bus Port/Device to be used (e.g. CANBusPort1)
Communication Type	CAN bus Communication
Error Handlers	Add the link for individual Error to Action.

Table 5.

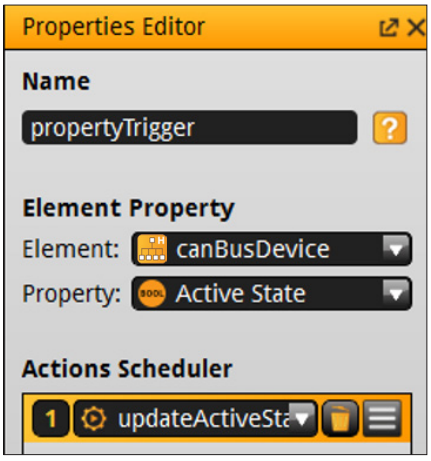

Figure 11. Values for the Property Trigger.

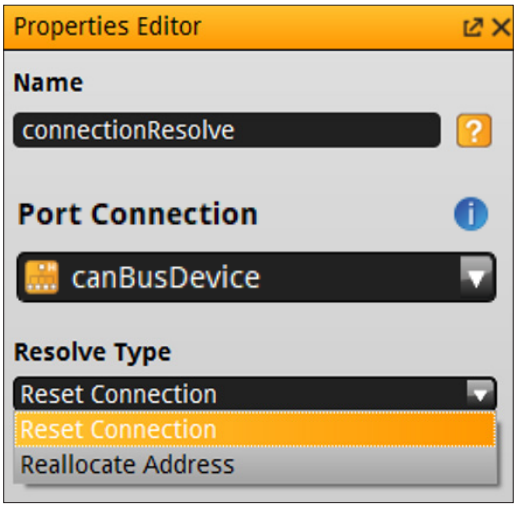

Figure 13. Values for Resolve Connection.

Figure 12. Values for the Communications Error Handler.

All CAN Frame Elements can link to a Communication Error Handler. If a CAN Frame Element cannot connect to the Communication Error handler then an Error Dialogue message will appear to notify the user.

7.3 Resolve Connection Action

The Resolve Connection Action controls different communication ports to resolve connection failures including CAN bus. Figure 13 shows an example configuration set-up in the Property Editor. The properties have been configured as shown in Table 6.

Property Name	Description
Name	connectionResolve
Port Connection	Select the CAN bus Port/Device to be used.
Resolve Type	Two Resolve options need to be set: i) Reset Connection - Resets the CAN bus port hardware. ii) Reallocate Address - Retries the Address Claim procedure and tries to claim the device address if the Address Claim has failed before.

Table 6.

8. Address Limitations

PanelPilotACE devices include the functions Address Claim and Self-configurable Address. Following power-up, the device will send an Address Claim PGN for each address in the Device Address table. It will also record other ECU devices' claimed addresses (whether or not another ECU has claimed success).

If such conflicts occur and other ECUs attempt to claim the address that already belongs to a PanelPilotACE device, the PanelPilotACE will follow the rules defined by J1939-81 which is to compare the ECU name. If the ECU has a higher priority, then it will add the address to an "occupied address list". The PanelPilotACE will randomly pick a new address that is not in the "occupied address list" and apptempt to claim a new address. If the claim is successful the Source Address for CAN Frame used for write will be updated to the new address.

PanelPilotACE does not trace other ECU address claims, therefore the Destination Address of CAN Frame used for write and Source Address of CAN Frame used for Read must be reconfigured in the Design Studio if the target ECU has changed address.

PanelPilotACE devices also manage "Request PGN" for address claims and "commanded address" from other ECUs. It will automatically send "Request PGN" for address claims when powered-on if Configurable Address is enabled. PanelPilotACE devices do not provide a "Commanded Address". A user can build one using the CAN frame builder function.

pgn0 - Torque/ Speed Control 1 - TSC1			
Transmission Repetition Rate		When active: 10 ms to engine - 50 ms to retarder	
Data Length		8 bytes	
Data Page		0	
PDU Format		0	
PDU Specific		DA	
Default Priority		3	
Parameter Group Number		0 (000000_{16})	
Bit Start Position/Bytes	Length	SPN Description	SPN
1.1	2 bits	Override Control Mode	695
1.3	2 bits	Requested Speed Control Conditions	696
1.5	2 bits	Override Control Mode Priority	897
2-3	2 bytes	Requested Speed/Speed Limit	898
4	1 byte	Requested Torque/ Torque Limit	518
Note: Retarder may be disabled by commanding a torque limit of 0%. Use of the limit mode allows the use of the retarder only up to the limit specified in the request. This can be used to permit retarding of up to 50%. For example, if that limit is required by some device such as ABS, or it can disable the use of the retarder by others when an ABS controller detects wheel slip.			

Figure 14. Table for pgn0 - Torque/ Speed Control 1 - TSC1.

9. Tutorial: Basic Data Transfer

This section is a tutorial that describes how to use a PanelPilotACE with CAN bus enabled to perform a basic data transfer.

9.1 Send PDU1 type data frame to bus

PDU1 format data has a PF byte value of less than 240 and a PS byte store for the Destination address.

Choose a PGN that has SPNs with different types. In the example above "PGN0 – Torque/Speed Control 1 – TSC1". Details of the PGN0 can be found in the J1939-71 spec. See Figure 14. for this information.

9.2 Create a CAN bus device in the Design Studio

1. Add a "CAN bus device" Hardware Element into a Design Studio project. Configure its Properties as follows:

Property Name	Description
Name	canBusDevice
Channel	1 (COMTYPE)
Frame Rate	125kbit/s
CAN Network Type	CAN J1939
Listen Only Mode	Off
Bus Terminate Control	On

Table 7.

2. Click the "Add" button to add a new ECU address with an ECU name with the default value.

3. Click on the newly added row and edit the address and name (see Figure 15 for the values).

4. Set the Device Address to 12 and check the “Arbitrary Address” box. Select OK to close the dialogue box. A new ECU name will be displayed (see Figure 16).

9.3 Add a CAN bus Variable to a Project

1. Go to the Project Browser in the top left-hand area of the workspace and select Project Variables from the tree. Right-click and select “Add CAN Variable” from the context menu (see Figure 17).

2. Name the new Variable “canVariablePGN0”.

3. Go to the Properties Editor and input the values as shown in Table 8 below:

Property Name	Description
Name	canVariablePGN0
PGN	0
Priority	3
Source Address	12
Total Length	8

Table 8.

The PanelPilotACE will know that this CAN Variable is used to write data to the bus.

PGN 0 has 5 SPNs inside. These are: SPN 695, SPN 696, SPN 897, SPN 898, SPN 518.

5. Add 5 new SPN settings.

6. Click-on the first SPN row to edit the first SPN setting.

7. Combining the data displayed in Figures 14 and 19, SPN 695 will start with byte/bit position 1.1 and size 2 bits. Input SPN 695, Byte Position 1, Bit Positon 1, Length 2 bits. Offset to 0 and scale to 1

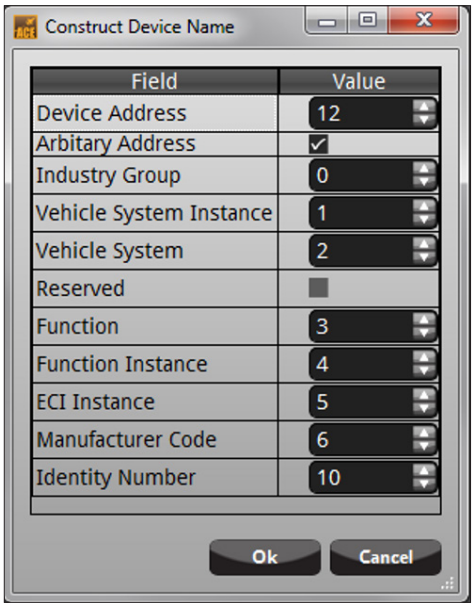

Figure 15. Values for Construct Device Name.

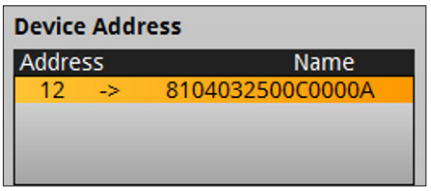

Figure 16. New ECU Name.

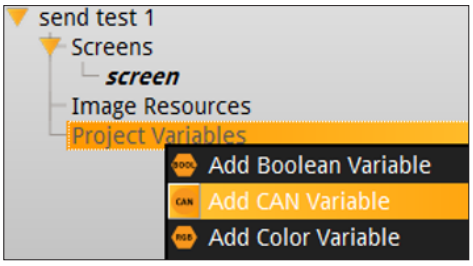

Figure 17. Adding a CAN Variable in the Project Browser.

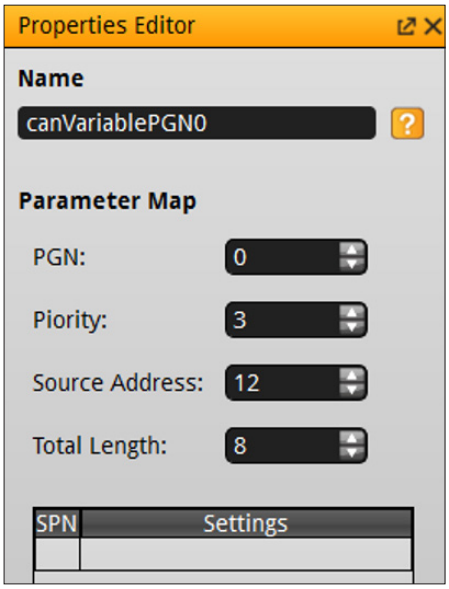

Figure 18. Values for Construct Device Name.

spn695 - Override Control mode - The override control mode defines which sort of command is used:		
00	Override disabled	Disable any existing control commanded by the source of this command.
01	Speed control	Govern speed to the included ‘desired speed’ value
10	Torque control	Control torque to the included ‘desired torque’ value
11	Speed/torque limit control	Limit speed and/or torque based on the included limit values. The speed limit governer is a droop governor where the speed limit value defines the speed at the maximum torque available during this operation.
Bit Length	2 bits	
Type	Status	
Suspect Parameter Number	695	
Parameter Group Number	[0]	

Figure 19. SPN695 data.

8. Set the Range as 0 to 3 and make sure Reverse is unchecked (see Figure 20).
9. Use the same method to edit SPN 696 as position 1.3, and SPN 897 as position 1.5.
10. Set SPN 898 as position 2.1, 16 bits long, offset 0, Range 0 to 8031.88, scale 0.125. (see Figure 21).
11. Input SPN 518 as position 4.1, 8bits long, offset -125, Range +/- 125, Scale 1 (see Figure 22).

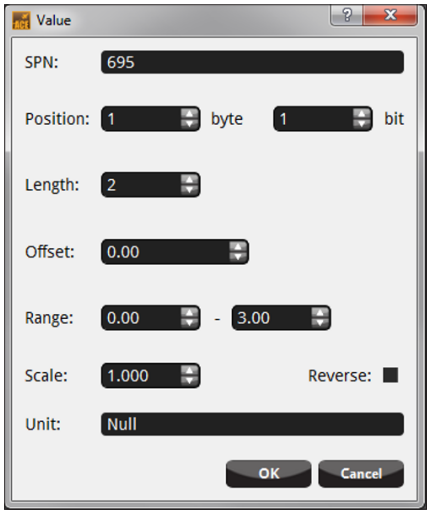

Figure 20. Values for SPN 695.

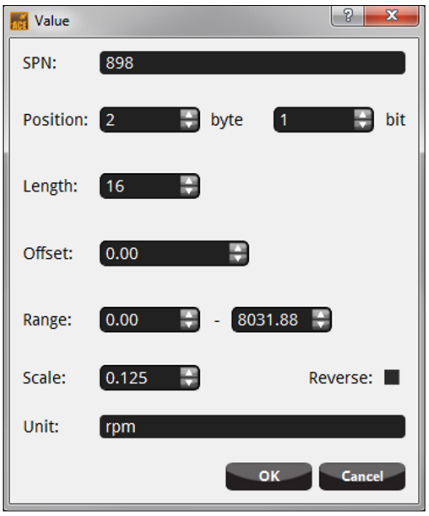

Figure 21. Values for SPN 898.

9.4 Add a CAN Frame Builder to a project

The following steps explain how to add a CAN Frame Builder to a project (see Figure 23).

1. Add a CanFrameBuilder Element from the Library.
2. Go to the Properties Editor and select "canBusDevice for the CAN bus Port field that was created in section 9.2.
3. Next choose "canVariablePGN0" as the CAN bus Variable.
4. Once you have selected a CAN bus Variable, the correct number of rows will appear in the "Parameters/Data" table.
5. For this example, input constant values 1, 0, 2, 1500.5, 89 in correct order as seen in Figure 23.
6. Input 50 in the Destination Address field. Other fields can be left with their default values.

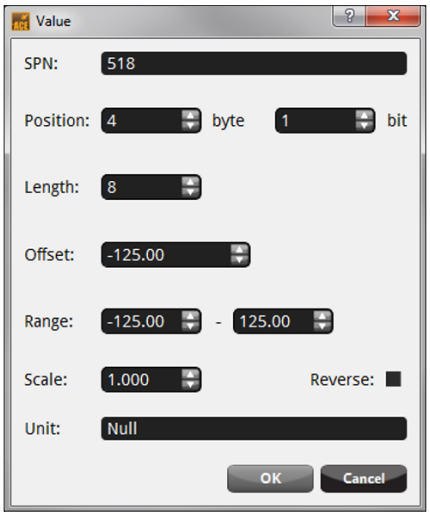

Figure 22. Values for SPN 518.

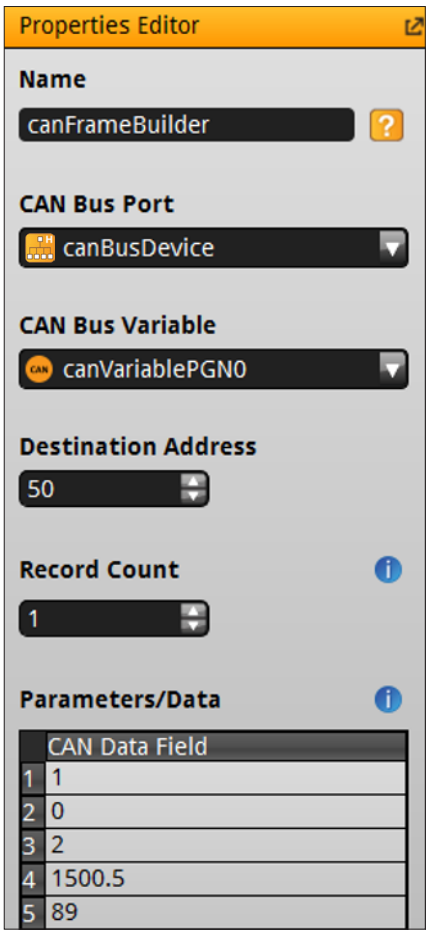

Figure 23. Values for CAN Frame Builder.

9.5 Link to a button

Add a Rectangle Visual Element to the screen. Configure the Rectangle as a Button Image and configure the button to perform an Action to the CAN Frame Builder as just created in step 9.2.

The button can be previewed in the PanelPilotACE Emulator (press F5) or upload the project to a SGD 70-A device.

9.6 Examine results with a CAN bus Analyser

We suggest using a CAN bus Analyser to capture RAW CAN frames on the bus. USBCAN-E-min (USBCAN-E-min can also decode CAN frames). Microchip APGDT002 CAN bus analysers are recommended for this task.

Power-up the SGD 70-A and both analysers. They will capture 2 CAN Frames from the SGD 70-A. (Figures 24 and 25).

The First message sent is "Request PGN 60928". This requests the other ECU to start an Address Claim.

Next is the Address Claim SGD 70-A owned address "0x0C' (i.e. Device address 12).

Press the "Rectangle" button on SGD 70-A to trigger it to send out the designed CAN frame.

You will see that one new frame has been captured from Device Address 12 to Device Address 50 (Figure 25). Select that message. The detail of the SPN data will show in the lower part of the window (Figure 25). The Analysers decode the SPN and values as 1,0,2,1500.5 and 89 matching the CAN Frame Builder.

9.7 Examine results with another PanelPilotACE device

If the CAN Variable has a Source Address that is not the same as the Device Address and linked to a CAN Frame Builder, the CAN Frame Builder will be automatically configured to capture incoming frames as described in Figure 27. To do this, take the following steps:

1. Save the project in the Design Studio as a new project.
2. Change the Device Address to 50 in the CAN bus Device Element. Also change the Name as seen in Figure 28.
3. Create 4 Integers and 1 float Project Variable as seen in Figure 29.
4. Replace the values in "Parameter/Data" in the CAN Frame Builder with the Project Variables shown in Figure 29. (i.e. items 1,2,3,5 use integer variables, item 4 uses the float variable).
5. Add a Text Box and a Set Rule Action "logicActions" to display the value on the screen (see Figure 30).
6. Go back to the CAN Frame Builder Post Action Property, select "logicActions".
7. Upload the project to a second PanelPilotACE and connect it to the CAN bus.
8. Press the "Button" on the first PanelPilotACE. The second PanelPilotACE will update the text box with the value and send from the first PanelPilotACE.

Num	Direction	Time	Name	ID(H)	AD Src(H)	AD Dest(H)	Type	Format	Length	Data(H)
0	Receive	18959.6842	RQST	18EA0CFE	FE	0C	Extend	Data Frm	3	00 EE 00
1	Receive	18961.6827	ACL	18EEFF0C	0C	FF	Extend	Data Frm	8	0A 00 C0 00 25 03 04 81

Figure 24. CAN bus Analyser output example.

TRACE	ID	DLC	DATA 0	DATA 1	DATA 2	DATA 3	DATA 4	DATA 5	DATA 6	DATA 7	TIME STAMP (sec)	TIME DELTA (sec)
RX	0x18EEFF0Cx	8	0x0A	0x00	0xC0	0x00	0x25	0x03	0x04	0x81	1637.2693	1.998
RX	0x18EA0CFEx	3	0x00	0xEE	0x00						1635.2714	75.914

Figure 25. APGDT002 Analyser example.

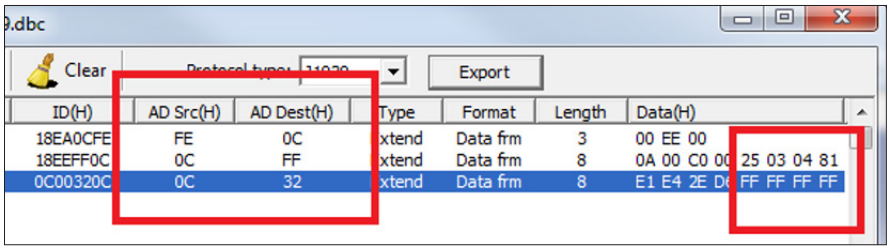

Figure 26. Incomming frame captured.

Num	Name	Physical value	Description	Raw value	StartBit	BitWi...	Factor	Offset
0	EngOverrideCtrlMode	1.00	peed control Govern speed to t...	1	0	2	1.000000	0.000000
1	EngRequestedSpeedCtrlConditions	0.00	ransient Optimized for driveline...	0	2	2	1.000000	0.000000
2	OverrideCtrlModePriority	2.00	Medium priority	2	4	2	1.000000	0.000000
3	EngRequestedSpeed_SpeedLimit	1500.50rpm	--	12004	8	16	0.125000	0.000000
4	EngRequestedTorque_TorqueLimit	89.00%	--	214	24	8	1.000000	-125.000000

Figure 27. Name values.

9.8 Auto reply "Request PGN"

The CAN Frame Builder operates as an "Outgoing CAN Frame" sending a CAN frame to a bus via an Action trigger.

It also automatically replies for "Request PGN". A user can broadcast (DA=255) Request or Send Request to other PanelPilotACE devices with a Destination Address by using Request PGN (RQST, PGN 59904). Figure 32 shows the Request PGN process. To activate, take the following steps:

1. Open the Design Studio project as created in section in 9.3.
2. Input the CAN variable "canCVariable1" with the values in Table 9:

canCVariable1	Value
PGN	59904
Priority	6
Source Address	50
Data Length	3bytes
SPN	2540, Position 1.1, Length 24, Offset 0, Scale 1, Range 0-65535.

Table 9.

Note: SPN Values are only labels used in the Design Studio. They do not affect the CAN frame Encode/Decode. Request PGN does not have an SPN. The data field only stores PGNs. Inputting 0 or 2540 as the SPN value will have the same result.

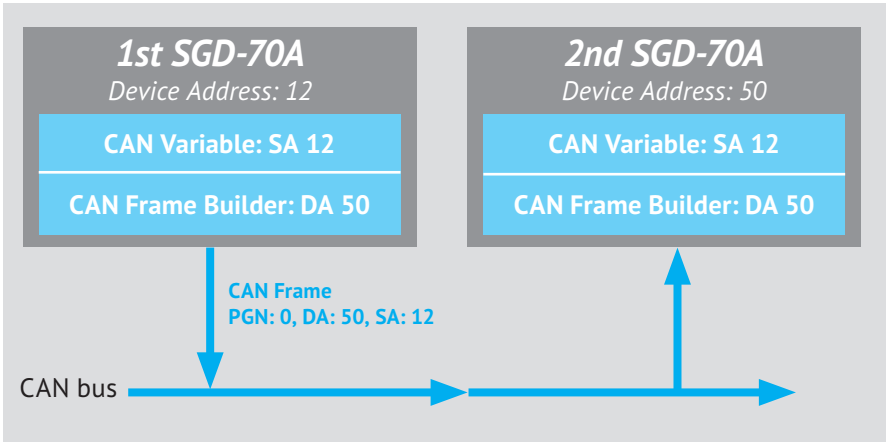

Figure 26. Capture the CAN Frame using SGD 70-A.

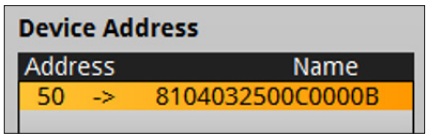

Figure 28. New Name Value.

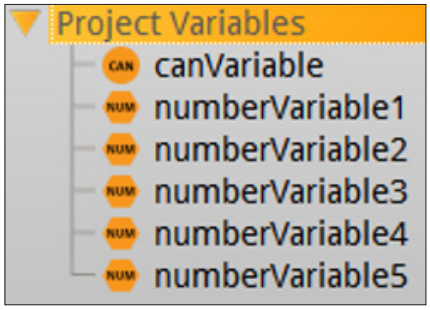

Figure 29. Creating new Integers and a new float Variable.

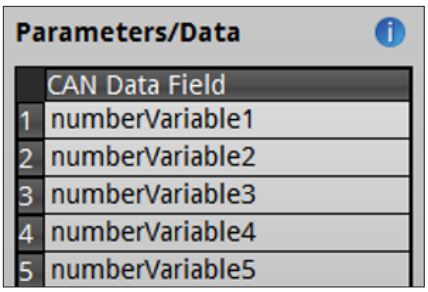

Figure 30. Parameters/Data

4. Add a new “canFrameBuilder1”. Link this to “canVariable1”. Set the Destination Address to 12. The CAN Data field should be 0 as this is the PGN to be requested. (see Figure 33).

5. Add a Visual Element and configue it as a Button. Set the “On Clicked Perform” property to “canFrameBuilder1”.

6. Upload the Design Studio project to the 2nd PanelPilotACE.

7. Start the Project on the first PanelPilotACE device. Click the button on 2nd PanelPilotACE. It will receive the PGN 0 from 1st PanelPilotACE and display this on the screen.

9.9 Acknowledge Supported Feature

The J1939 Group Function includes a Proprietary Function, Network Management Function and Transport Functions.

When sending a request with a PGN of a Group Function, that ECU will reply directly with data or reply an acknowledge to tell you whether it supports that feature or not.

e.g. if a device needs to know if the PanelPilotACE device supports Transport Data it will send “Request PGN 60416” or “Request PGN 60160”. The PanelPilotACE will reply “Acknowledge” back.

Test this feature by modifying the Design Studio Project already used in this tutorial with a different PGN. Monitor the PanelPilotACE replies with a CAN bus analyser. Replies should be as displayed in Table 10.

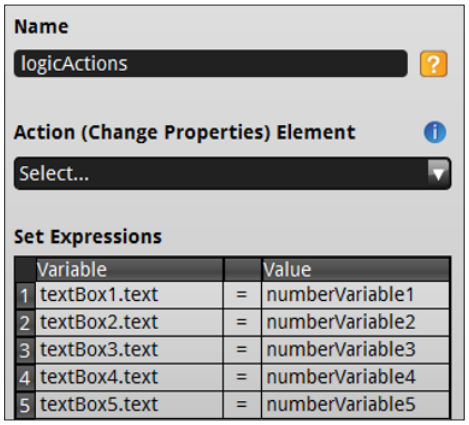

Figure 31. Logic Actions.

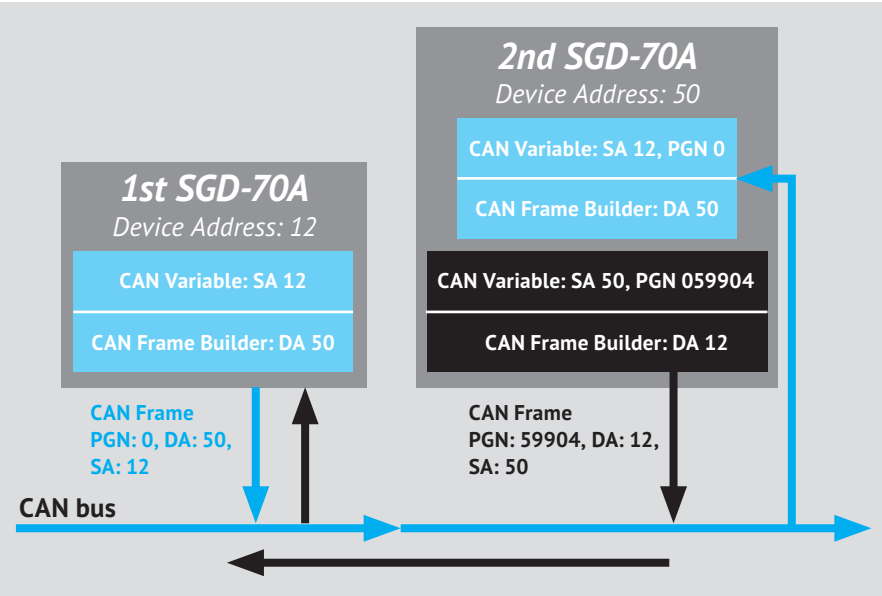

Figure 32. The Request PGN process.

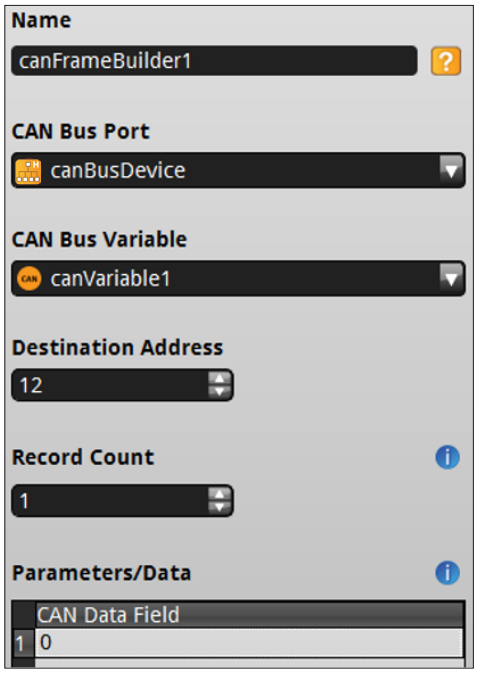

Figure 33. Values for camFrameBuilder1.

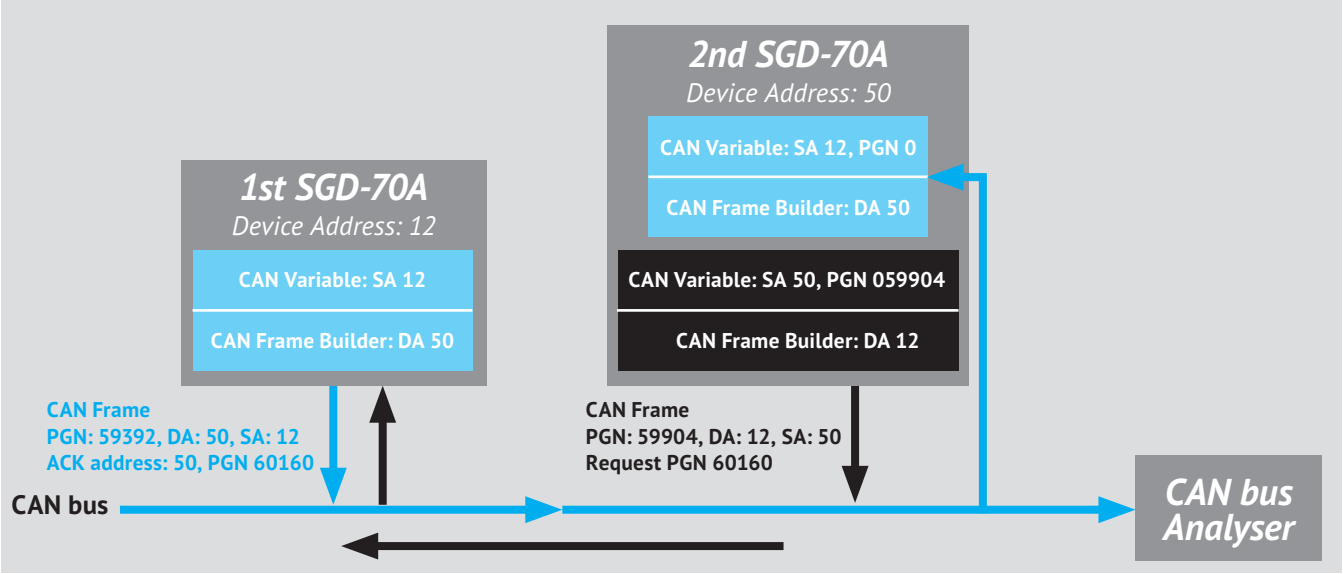

Figure 34. Acknowledge Supported Feature

PGN	Description	Reply
PGN 59904	Request	ACK
PGN 59392	Acknowledge	ACK
PGN 61184	Proprietary A	NACK
PGN 126720	Proprietary A2	NACK
PGN 65280	Proprietary B	NACK
PGN 51456	Request 2	NACK
PGN 51712	Transfer	NACK
PGN 60416	Transport Connection Manage	ACK
PGN 60160	Transport Data Transfer	ACK

Table 10.

Using a CAN bus Analyser, click on the receive frame to check "ControlByte". Here you can see the reply is either ACK or NACK.

Alternatively use a PanelPilotACE CAN Variable, set PGN to 59392, with the first byte set SPN 2556 which is the Control Byte containing the ACK. The other 7 bytes are not processed. Only map the first byte to the number variable in the CAN Frame builder. The reply is ACK if the value is 0.

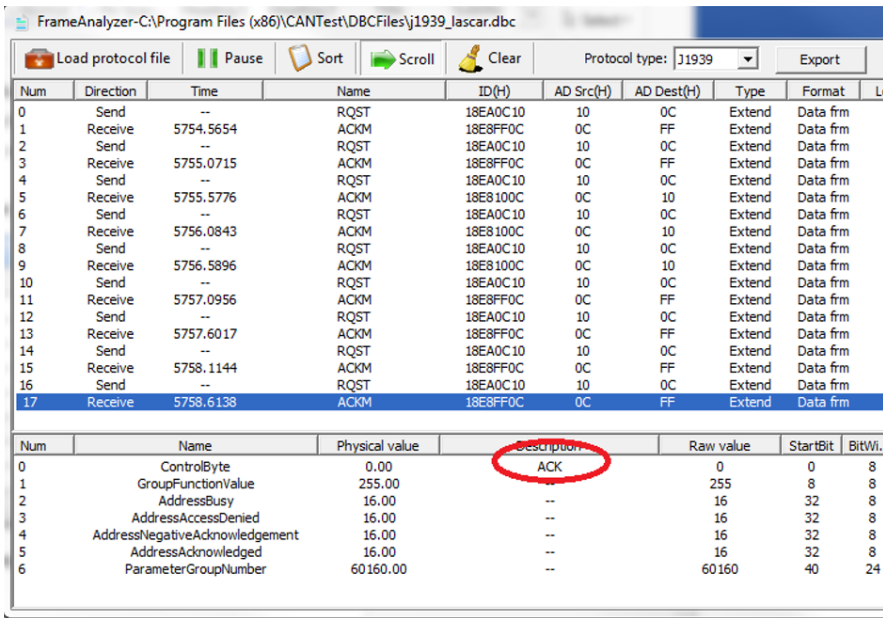

Figure 35. Use CAN Bus Analyser to check the reply

9.10 Encode/Decode CAN Frame with String Data Type

In J1939 some PGNs have SPNs with String Data Types. PGN 61445 is an example of one of these. If using this type, then the CAN Frame Builder set the SPN scale to 0 in the CAN Variable as shown in Figure 36.

Open the existing tutorial project and edit the CAN variable as shown in Table 11.

String Data Type	Value
PGN	61445
Priority	6
Source Address	12
Total Length	8

Table 11.

Note: String Data Types do not need Offset or Range. Input 0 in the SPN Editor dialogue box.

9.11 Change the linked Project Variable default value with different values

1. Upload the tutorial project for sending a CAN Frame to the 1st PanelPilotACE with the Device Address set to 12.
2. Upload the project for capturing a CAN Frame with the 2nd PanelPilotACE with the Device Address set to 50.
3. Click the Button on the 1st PanelPilotACE. The 2nd PanelPilotACE will receive the CAN frame and update the received data on the screen.

   The data length of the SPNs 162 and 163 is only 2 characters. PanelPilotACE will delete the first 2 characters from the String Variable in order to send and store only 2 characters when receiving the frame.
4. Alternatively users can use a CAN bus Analyser. Input a CAN frame with frame ID=18F0050C, data=C8 E9 2E E6 20 52 4E 31 and send it to the bus as seen in Figure 38.
5. The 2nd PanelPilotACE will show value 75, 12.009, 105, " R", "N1" on the screen.

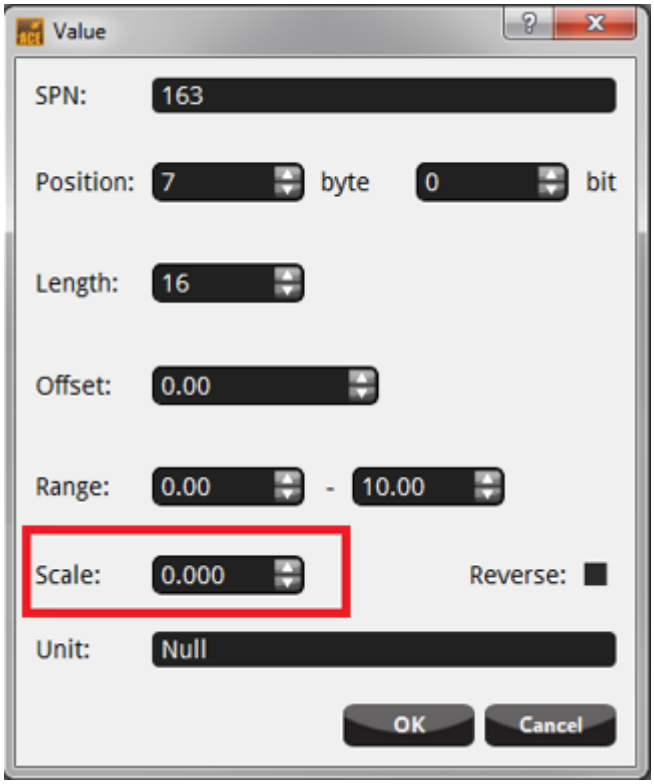

Figure 36. Setting the scale to 0 to signify the SPN as string type data.

SPN	Position	Length (bits)	Offset	Scale	Range	Linked Project Variable
524	1.1	8	-125	1	-125/125	Number Variable
526	2.1	16	0	0.001	0/62.25	Number Variable
523	4.1	8	-125	1	-125/125	Number Variable
162	5.1	16	0	0	0	String Variable
163	7.1	16	0	0	0	String Variable

Figure 37. SPN Data.

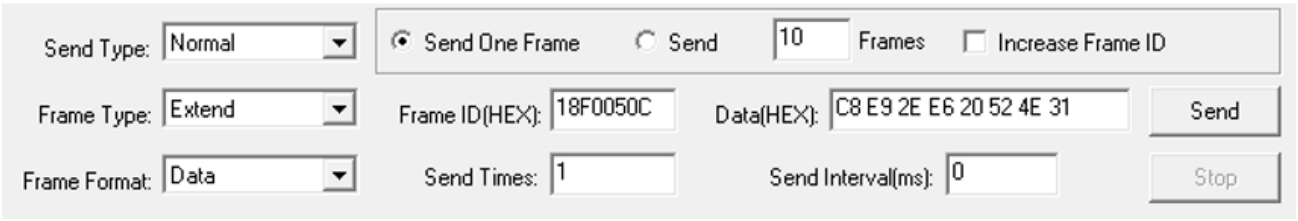

Figure 38. Use CAN Bus Analyser to send the PGN 61445 to PanelPilotACE.

10. Tutorial: Transport Data

If a PGNs data length is more than 8 bytes, J1939 will use a Transport Protocol to segment the data. Transport Protocols consist of 2 flow control methods:

a) **BAM**: Broadcast Announce Message.

b) **RTS/CTS**: Request to Send/Clear to Send.

In the case of RTS/CTS control the Sender will send the RTS signal and wait for CTS before replying with the Transport Data.

Some PDUs' do not have a Destination Address field. In this case the PanelPilotAce will automatically use a BAM to the handle the Transport Data.

10.1 Transport Data with BAM

To inform a node that Transport Data will be following, there is no need to ACK each Transport Data segment.

To understand how Transport Data works with BAM, try the following example.

1. Add a CANbus device to a new Design Studio project.
2. Create a CAN Variable called "canVariable65212" with the values shown in Table 12.

canVariable65212	Value
PGN	65212
Priority	7
Source Address	12
Total Length	16

Table 12.

3. Create a CAN Frame Builder with CANbus variable set to "canVariable65212" and the Parameters/Data properties to 12000, 26000.5, 4000100.1, 60000.

SPN	Position	Length (bits)	Offset	Scale	Range	Linked Project Variable
990	1.1	32	0	0.125	0/526385151.9	Number Variable (float)
991	5.1	32	0	0.125	0/526385151.9	Number Variable (float)
992	9.1	32	0	0.125	0/526385151.9	Number Variable (float)
993	13.1	32	0	1	0/4227858431	Number Variable (float)

Figure 39. SPN Data 2.

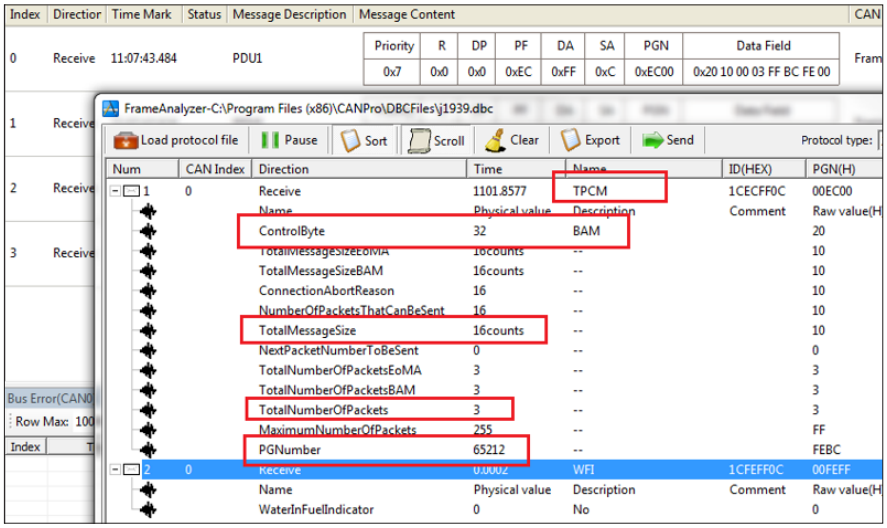

Figure 40. BAM message content.

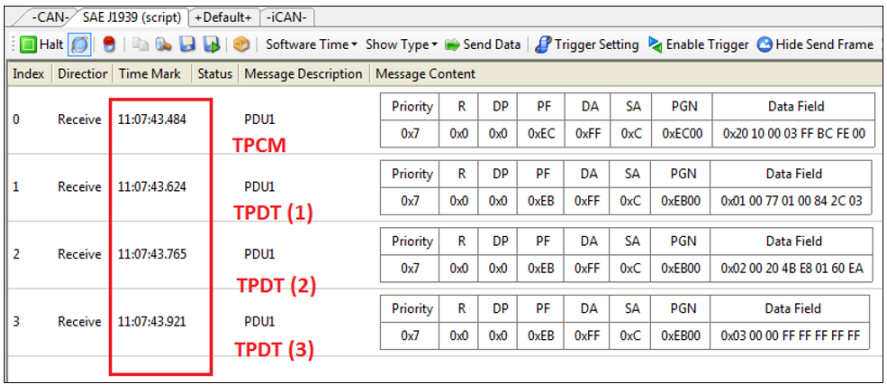

Figure 41. Transport Data follow the BAM.

4. Add a button image. Give the button the action function "On Checked Perform" to the CAN frame.
5. Upload the project to a SGD 70-A device and set an analyser to begin capturing data.
6. If the button on the SGD 70-A is pressed the analyser will capture 4 frames. The first frames detail shows that the "TPCM" message is BAM, with PGN 65212, a total of 3 packets and data length as 16 (see Figure 41.)
7. 3 TPDP frames follow the BAM carrier the data as seen in Figure 40.

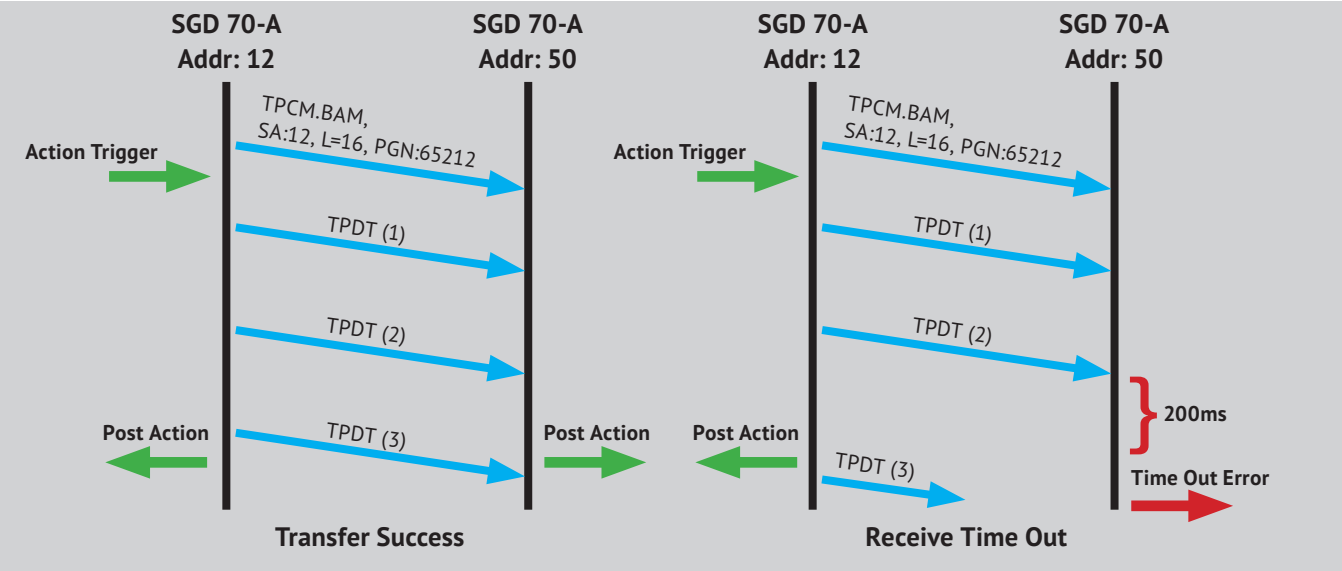

Figure 41. BAM Transport.

8. Modify the CAN Variable for receiving and upload this to the 2nd PanelPilotACE device. It will capture the data and decode it automatically. Post Action only performs when all transport is completed.

If a missing transport data frame or a data frame interval is more than 200ms, the device will ignore the data and trigger "Transport Receive Time Out" Error Message as seen in Figure 41.

10.2 RTS/CTS: Request to Send & Clear to Send

A Sender sends RTS and waits for the CTS to reply before a Transport Data transfer. Transport Data also ends with a End Of Message ACK as seen in Figure 44. Take the following steps to begin an RTS/CTS transfer:

1. Use the existing tutorial project and upload to the 1st PanelPilotACE device.
2. Save the project as a new project and change the Destination Address to 50.

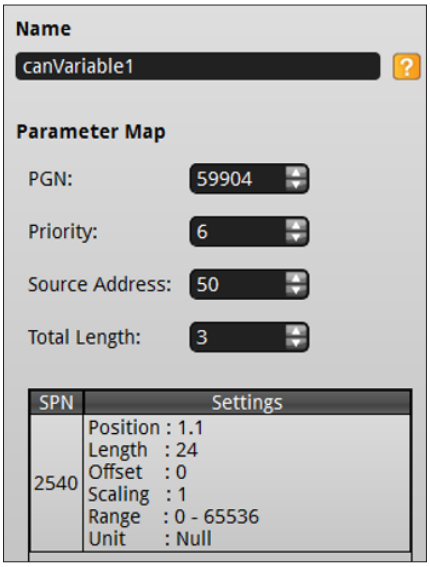

Figure 42. CanVariable1 Properties.

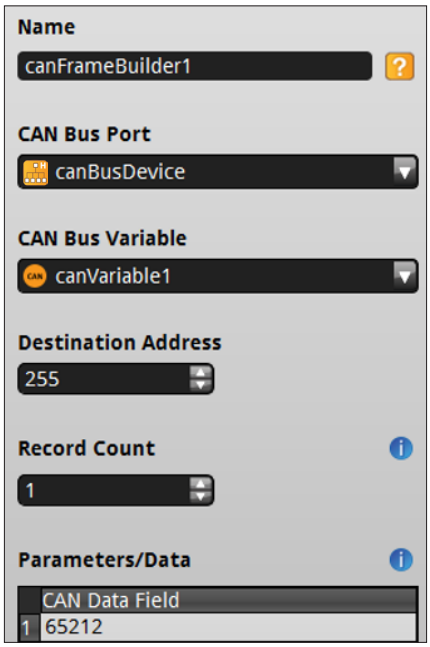

Figure 43. CanFrameBuilder1 Properties.

3. Add a "request PGN" to the project. Modify it to request PGN 65212 with SA=50 as seen in Figures 42 and 43.
4. Modify the button "On Clicked Perform" to the CAN Frame Builder that sends the request.
5. Upload this to the 2nd PanelPilotACE, Press the button on the 2nd PanelPilotACE device to send the request.

6. The 2nd PanelPilotACE will capture and update PGN 65212 to the value on-screen. Each transport data frame carries 7 bytes of data. The PGNs total length is 16 bytes. Therefore, it will segment into 3 transport data frames.

7. Request, RTS, CTS, EOMA and CAN bus analyser will capture a total of 7 CAN frames as seen in Figure 45.

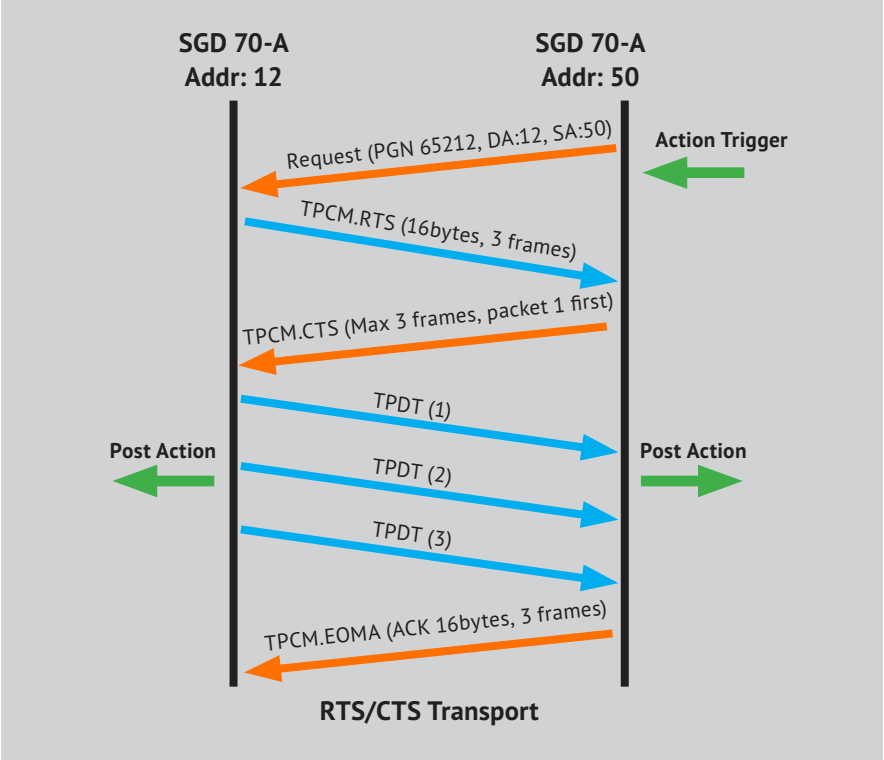

Figure 44. RTS/CTS Transport diagram.

-CAN- | SAE J1939 (script) | +Default+ | -iCAN-

Halt | Software Time | Show Type | Send Data | Trigger Setting | Enable Trigger | Hide Send Frame

Index	Direction	Time Mark	Status	Message Description	Message Content
0	Receive	17:55:50.434		PDU1	Priority: 0x6 · R: 0x0 · DP: 0x0 · PF: 0xEA · DA: 0xC · SA: 0x32 · PGN: 0xEA00 · Data Field: 0xBC FE 00
1	Receive	17:55:50.441		PDU1	Priority: 0x7 · R: 0x0 · DP: 0x0 · PF: 0xEC · DA: 0x32 · SA: 0xC · PGN: 0xEC00 · Data Field: 0x10 10 00 03 03 BC FE 00
2	Receive	17:55:50.442		PDU1	Priority: 0x7 · R: 0x0 · DP: 0x0 · PF: 0xEC · DA: 0xC · SA: 0x32 · PGN: 0xEC00 · Data Field: 0x11 03 01 FF FF BC FE 00
3	Receive	17:55:50.587		PDU1	Priority: 0x7 · R: 0x0 · DP: 0x0 · PF: 0xEB · DA: 0x32 · SA: 0xC · PGN: 0xEB00 · Data Field: 0x01 20 03 00 00 40 06 00
4	Receive	17:55:50.733		PDU1	Priority: 0x7 · R: 0x0 · DP: 0x0 · PF: 0xEB · DA: 0x32 · SA: 0xC · PGN: 0xEB00 · Data Field: 0x02 00 2C 01 00 00 90 01
5	Receive	17:55:50.878		PDU1	Priority: 0x7 · R: 0x0 · DP: 0x0 · PF: 0xEB · DA: 0x32 · SA: 0xC · PGN: 0xEB00 · Data Field: 0x03 00 00 FF FF FF FF FF
6	Receive	17:55:50.898		PDU1	Priority: 0x7 · R: 0x0 · DP: 0x0 · PF: 0xEC · DA: 0xC · SA: 0x32 · PGN: 0xEC00 · Data Field: 0x13 10 00 03 FF BC FE 00

Figure 45. Captured CAN bus frames.

11. Tutorial: Non-reconfigurable Address Claim

Previous sections have explained the address claim used in J1939. In this section different use cases for the address claim are demonstrated.

1. Open the previous tutorial project that contains a single PGN write triggered by button.
2. Modify the project by unchecking the “Arbitrary Address” Check Box in Device Name.
3. Upload this to the 1st PanelPilotACE. The device address will be 12. The Name has become “0104032500C0000A”. Power-off the device..
4. Save the project as a new project. Change the device Identity Number to 5 and upload the project to the 2nd PanelPilotACE. This device address is also 12, the name has become “0104032500C00005”. Power-off the device.

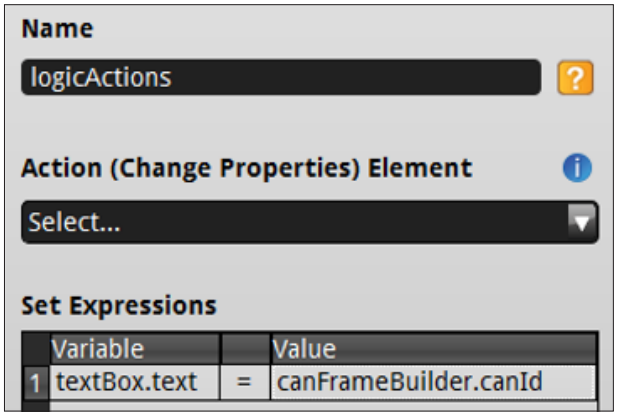

Figure 46. Logic Actions.

Figure 47. Edit Identity Number.

11.1 Address Claim conflict: A new node with a higher priority name

1. Power-up the 1st PanelPilotACE device. The CAN bus analyser will capture an Address Claim Request as seen in Figure 49. When the PGN equals 0xEE00 (Address Claim) then the DA=0xFF (i.e. broadcast) Data Field reads “0x0A 00 C0 25 03 04 01”. This reverses the order of the “name”.

Figure 48. Error Dialogue Box.

0 Receive 12:29:04.874 PDU1

Priority	R	DP	PF	DA	SA	PGN	Data Field
0x6	0x0	0x0	0xEE	0xFF	0xC	0xEE00	0x0A 00 C0 00 25 03 04 01

Figure 49. CAN bus Analyser results.

2. Press the button to send a PGN. The LCD will update the value 201326604 that is the CAN ID. Convert this into hex to become 0xC000000C.
3. The last byte is 0x0C, equal to 12 that is the device address.
4. Power-up the 2nd PanelPilotACE (name "0104032500C00005"). This device will send an address claim. The 1st PanelPilotACE has the same Device Address and is not reconfigurable and so its device address became invalid.

An Error dialogue box will appear if the address claim is unsuccessful (see Figure 48). The 1st PanelPilotACE write CAN frame access rights will be disabled. Press the button on 1st PanelPilotACE and it will not respond.

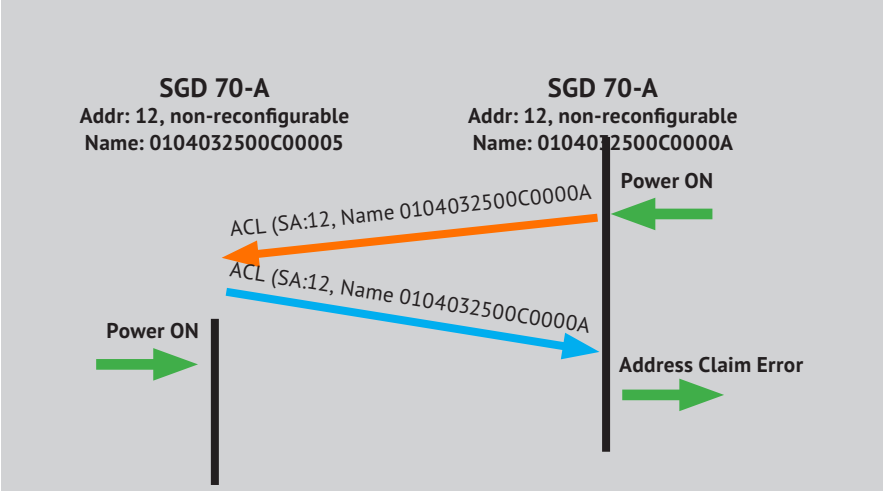

Figure 50. Multiple ECU Connection.

11.2 Address Claim conflict: A new node with a lower priority name

1. Reverse the power-up sequence: Power-up the PanelPilotACE device with the name "0104032500C00005" first.
2. Now Power-up the other PanelPilotACE device with the name "0104032500C0000A".

"0104032500C0000A" will fail to claim an address again because it's name is now a lower priority (see Figure 51).

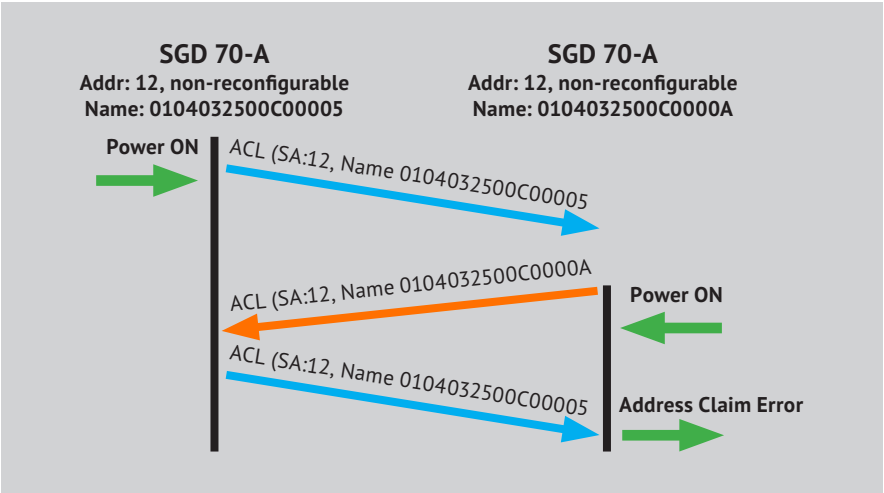

Figure 51. Multiple ECU Connection.

12. Tutorial: Reconfigurable Address Claim

When a PanelPilotACE CAN bus Device is configured with a reconfigurable address, it will send a “Request for Address Claim” (PGN 59904, data 60928) with the Destination Address as its’ prefered device address following power-up.

If the bus line already has an ECU using Device Address 12, that ECU will reply the request PGN. The Reply is by an Address Claim Frame.

The ECU will know that the address was occupied and by who. There will be one of two outcomes:

1. If the ECUs name is lower priority than itself, it will send the address claim to use its’ own prefered address and name to force the other ECU to change it’s address.
2. If the ECU name has a higher priority than itself, it will change to a new address and try to claim it.

After the address claim is successful any further PGN CAN Frames will be sent using the new address. Figure 52 shows the CAN frame in this process.

Note: A Request for Address Claim can be broadcast in J1939. All nodes on the CAN bus will reply Address Claim therefore it is the fastest way to discover unused Device Addresses. The PanelPilotACE system does not use this method because it generates too much traffic on the BUS.

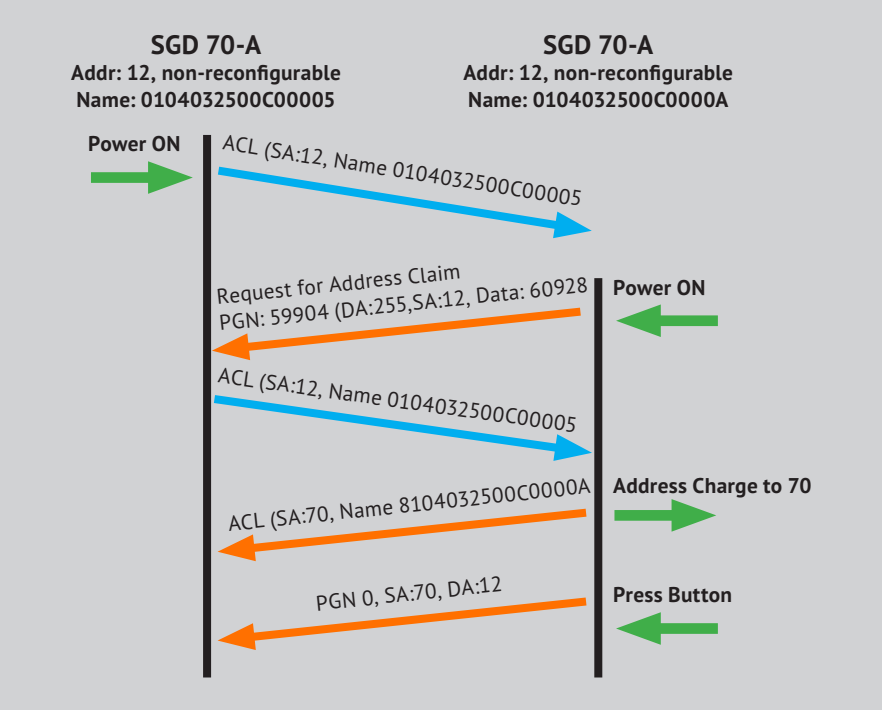

Figure 52. Reconfigurable Address Claim Process.

				Priority	R	DP	PF	DA	SA	PGN	Data Field
1	Receive	15:18:51.420	PDU1	0x6	0x0	0x0	0xEA	0xC	0xFE	0xEA00	0x00 EE 00
2	Receive	15:18:51.422	PDU1	0x6	0x0	0x0	0xEE	0xFF	0xC	0xEE00	0x05 00 C0 00 25 03 04 01
3	Receive	15:18:51.485	PDU1	0x6	0x0	0x0	0xEE	0xFF	0x46	0xEE00	0x0A 00 C0 00 25 03 04 81

Figure 53. Reconfigurable Address Claim.

13. Tutorial: Error Handling

The Design Studio provides a "Communication Error Element" to handle communication errors.

This section describes how to configure this Element to handle CAN bus J1939 errors.

13.1 Using Communication Error Element

1. Add a Communication Error Element to the project
2. Select the CAN bus Device in Communication Port Property.
3. Select "CAN bus Communication" in the Communication Type Property.
4. Click Add to include an Error Handler.
5. Select the handle in the "Error" combo box as seen in Figure 55.
6. Select the action that will be triggered when the error occurs in the "Action" combo box as seen in Figure 56.
7. Go to the properties for the CAN Frame Builder. Choose the created Error Handler in the "Error Handler" property.

13.2 Disable Error Dialog for a specific error

An Error will show in the dialogue box by default if an error handler link to the CAN Frame Builder has not been set.

If a user doesnt want to display an Error Dialogue box on screen. Add an Error Handler but leave Action empty. This specific error will become muted.

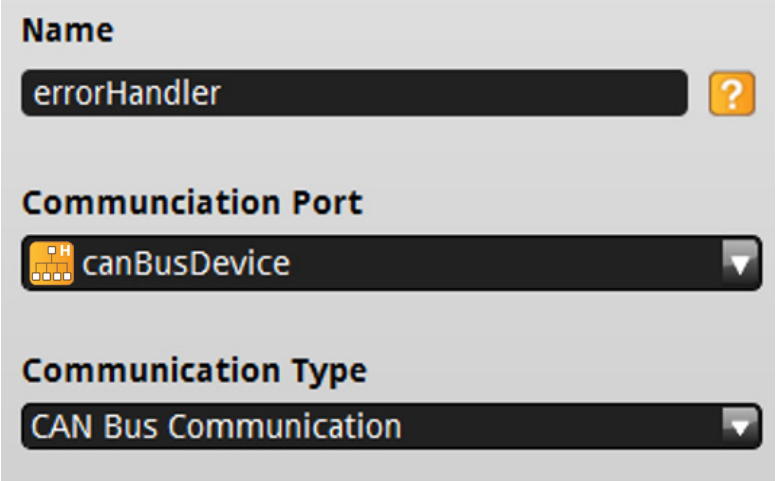

Figure 54. Reconfigurable Address Claim Process.

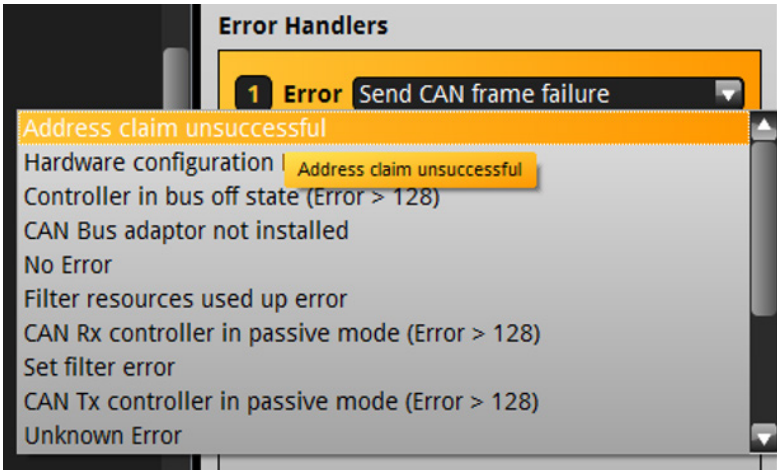

Figure 55. Reconfigurable Address Claim Process.

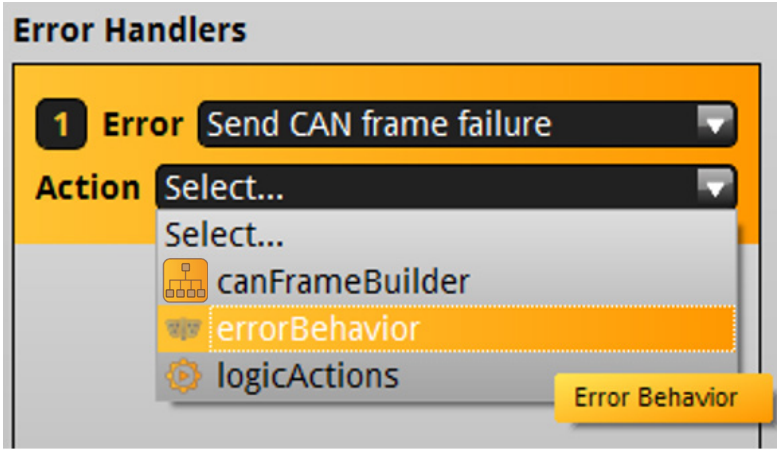

Figure 56. Reconfigurable Address Claim Process.

PanelPilotACE

Contact us for more
PanelPilotACE products,
services and support:

UK & Europe
www.lascarelectronics.com
sales@lascar.co.uk
+44(0)1794 884567

Americas
www.lascarelectronics.com
us-sales@lascarelectronics.com
(814) 835 621

Asia
www.lascarelectronics.com
us-sales@lascarelectronics.com
(814) 835 621

LASCAR electronics

PanelPilotACE
CAN bus Protocol User Guide
Issue 1 04/2019