neoECU-12



User's Guide

Version 1.0 – March 17, 2020





TABLE OF CONTENTS

- 1 Introduction and Overview
 - 1.1 Introduction
 - 1.2 Package Contents
 - 1.3 Operational Overview
 - 1.4 Block Diagram
 - 1.5 Summary of Key Features
 - 1.6 Hardware and Software Requirements
- 2 A Tour of neoECU-12 Hardware
 - 2.1 Case and Overall Design
 - 2.2 Front side of case
 - 2.3 Back side of case
- 3 Hardware and Software Setup
 - 3.1 Vehicle Spy and Driver Installation and Setup
 - 3.2 Hardware Connections
 - 3.3 Termination
 - 3.4 Vehicle Network and Power Connections
 - 3.5 PC Connection
- 4 Device Configuration
 - 4.1 Starting and Using neoVI Explorer
 - 4.2 Update Firmware
 - 4.3 Hardware Setup
 - 4.4 CAN Setup
 - 4.5 LIN Setup
 - 4.6 MISC IO Setup
 - 4.7 Program CoreMini
- 5 Function Block Scripts
 - 5.1 Setting MISC IO as inputs
 - 5.2 Send MISC IO status in CAN message
 - 5.3 Setting MISC IO as outputs
 - 5.4 Setting the MISC IO as PWM outputs
 - 5.5 Setting the Analog Inputs
 - 5.6 Send Analog Input value in CAN message
 - 5.7 Controlling the Tri-Color LEDs
 - 5.8 LIN example
 - 5.9 CAN to CAN-FD Gateway example
- 6 Support Contact Information
 - 6.1 ICS United States Headquarters
 - 6.2 ICS International Offices

1 Introduction and Overview

1.1 Introduction

Thank you for purchasing an Intrepid Control Systems neoECU-12 low-cost embedded ECU. The neoECU-12 is a scriptable, low cost node for CAN/CAN-FD, and LIN. The neoECU-12 provides two programmable CAN/CAN-FD channels. The CAN/CAN-FD channels can be selected for SW CAN, or LSFT CAN, or CAN/CAN-FD. The neoECU-12 also provides one channel of LIN, seven channels of MISC I/O (six at 3.3V, one at 5.0V), four channels of Analog Inputs, and five channels of PWM. The five channels of PWM are multiplexed with the first five channels of MISC I/O.

The neoECU-12 is the next generation of the neoECU-10 and adds CAN-FD capability.

1.2 Package Contents

Your neoECU-12 package includes both hardware and software.

Hardware

Upon opening the neoECU-12 box, you should find a neoECU-12 Firmware Guide on top with the device itself secured under a transparent plastic film in a cardboard holder. Remove the guide, the device and the cardboard, and you'll see the following items:



1.3 Operational Overview

The neoECU-12 is a compact but powerful tool for working with vehicle networks. Its operation can be broken into multiple network communication, analog and digital inputs, and PWM and digital outputs. This device is ideal for CAN and/or LIN node communication and provides numerous connection points to interface to external circuits for command and control.

Using Vehicle Spy software you can define CAN and LIN transmit messages with custom data and send them on a periodic schedule or send them based on certain conditions. Using Intrepid's Scripting language built into Vehicle Spy you can create custom intelligent scripts that control the input and output ports. The data from the input and output ports can be inserted as signals into the CAN or LIN messages or used to trigger CAN or LIN messages. The custom scripts will be compiled and programmed into the internal memory (called CoreMini) of the neoECU-12.



1.4 Block Diagram

Please note HSCAN 1 H, LSFT CAN 2 H, and SW CAN 2 share the same pin (14) so only one can be selected. HSCAN 1 L and LSFT CAN 2 L share the same pin (15). You select which network you want to use in neoVI Explorer.

To force the neoECU-12 into bootloader mode put a "low" onto the Hard Start pin (before you power-up the neoECU-12). Otherwise leave it open since the neoECU-12 has an internal pull-up resistor keeping the pin "high".

To force the neoECU-12 into reset mode put a "low" onto the Reset pin. Otherwise leave it open since the neoECU-12 has an internal pull-up resistor keeping the pin "high".

1.5 Summary of Key Features

The neoECU-12 is a low-cost embedded ECU with CAN FD and LIN. It has two channels of CAN/CAN FD which can also be configured for SW CAN and LSFT CAN. It has digital inputs and outputs, analog inputs, and PWM outputs. There are four internal programmable buttons and five internal tri-color LEDs that are visible on the back side of the unit.

Construction

- Compact design: 2.2" x 3.7" x 1.0" (5.6 X 9.4 x 2.5 cm)
- Light weight: less than 2 oz. (57 g)
- Plastic casing
- 25-pin male DB 25 connector

Power and Performance

- Fourth generation neoVI architecture
- 32 MB memory for custom scripts
- Field upgradeable firmware
- 4.5V to 40V input power on VBatt
- Lower power sleep mode

Network Interfaces and Features

- Two selectable CAN channels: CAN/CAN-FD or SW CAN or LSFT CAN
 - Software-configurable CAN termination
 - o Software enable/disable, baud rate and other parameters
- 1 LIN / K-Line channel
 - Software enable/disable, baud rate and other parameters
- Six selectable MISC DIO or PWM output channels. NOTE: PWM 3 waveform is mirrored on PWM 6. Frequency range is 5 Hz to 20 KHz. Maximum frequency is 100 KHz when using 50% duty cycle.
- Four Analog inputs. 12-bit A/D resolution (0-4095 counts), 5V maximum input voltage.
- Hard start pin. If a "low" voltage is detected on this pin then the neoECU-12 starts in bootloader mode.
- Reset pin. If a low voltage is inserted on this pin then the neoECU-12 is reset.

1.6 Hardware and Software Requirements

- A vehicle network, either within an actual vehicle or a test bench environment.
- A DC power supply capable of providing 4.5V (minimum) to 40V (maximum), with a nominal current of 70 mA at 12V. Your network setup must include wiring capable of providing this power on pin 25 and ground on pin 13.
- In order to program the neoECU-12 you will need a ValueCAN or neoVI device connected to HS CAN 1 and a licensed copy of Vehicle Spy (Pro or Enterprise version) 3.9.1.13 or higher running on a Windows-based PC or laptop.

2 A Tour of neoECU-12 Hardware

2.1 Case and Overall Design

The neoECU-12 is enclosed in a lightweight plastic case. The device has been designed and tested for invehicle use, and is operational in a temperature range from -40°C to +85°C.

The pinout for the DB25 male connector is on the bottom of the case.

DB	25 Pin Out	14	HS CAN 1 H
1	SW CAN 1		SW CAN 2
2	HARD START	15	HS CAN 1 L
3	LSFT CAN 1 H	23	LSFT CAN 2 L
4	LSFT CAN 1 L	16	HS CAN 2 H
5	MISC IO 1	17	HS CAN 2 L
6	MISC IO 2	18	MISC IO 3
7	MISC IO 7	19	MISCIO 4
8	LIN 1	20	MISC 10 5
9	•	21	MISCIO 6
10	AIN 1	22	AIN 3
11	AIN 2	23	AIN 4
12	040	24	RESET
13	GND	25	VBATT

2.2 Front side of Case

The front side of the case contains the DB25 male connector.



2.3 Back side of Case

The back side of the case contains the serial number and tri-color LEDs.



3 Hardware and Software Setup

- 3.1 Vehicle Spy Installation
 - 1 Run the Vehicle Spy installer from the CD-ROM or from the download link
 - 2 Select Language
 - 3 Start Vehicle Spy setup wizard
 - 4 Review and accept license agreement
 - 5 Select installation type (new or repair ... most cases will be new)
 - 6 Select destination location (we recommend the default location)
 - 7 Select Start Menu folder (we recommend the default location)
 - 8 Review installation options and begin installation
 - 9 Install VCP Drivers
 - 10 Install WinPcap
 - 11 Install ICS port Drivers

3.2 Hardware Connections

The following diagram will show the minimum connection to the neoECU-12 in order to configure it and program the CoreMini of the device. Connect to HS CAN 1 of the neoECU-12.



Termination. Please note the neoECU-12 does not have built-in selectable termination resistors. Therefore, an external resistor (120 ohm) is required on the CAN-H and CAN-L pins as shown below. The device you are connecting to also needs the termination resistor. Most of the Intrepid Control System devices have built-in selectable termination resistors that can be enabled in via the hardware setup for that device. If this option is not available, then add the external resistor as shown below.



4 Device Configuration

4.1 Internal momentary pushbutton switches

In order to open the case use a small screwdriver and put into the slot near the front and twist. Do this for both sides and you will see the case is hinged in the back. Now you have access to the momentary pushbutton switches.



Press and hold one of these switches to get in that particular mode when applying power to the neoECU-12. After power is applied the LEDs will light red in sequence from 1 to 5 and repeat.

Switch	Description
Switch 1	Prevents neoECU from Automatically running the CoreMini Script
Switch 2	Sets application settings to default states
Switch 3	Forces neoECU into Boot-Loader Mode (This can also be done by holding Pin 2 of the 25 pin connector to ground)
Switch 4	Forces neoECU into Boot-Loader mode and sets application settings to default states

S1, S2, S3, S4



4.2 Update firmware

Open Vehicle Spy 3.

You will see the version number in the lower left corner (i.e. 3.9.1.15) and the level in red text (i.e. Basic, Pro, or Enterprise). This is the device connected to the computer via the USB cable that will program the neoECU-12 over the CAN Bus.

Open	Help	
Tutorials		Configure Hardware.
 <u>Tutorial 1 : Basics</u> <u>Tutorial 2 : Trans</u> <u>Tutorial 3 : Decod</u> <u>More Tutorials</u> 	of Vehicle Spy mitting Messages ing Signal Data	Vehicle Network Interface ValueCAN4-2 V20892
3.9.1.15	Enterprise	

Next, select "CoreMini Console" from the Tools pull-down menu and wait for the CoreMini Console window to open.

Click on the neoECU CAN tab and note the neoECU-12 is in bootloader mode.

🔤 Core	Mini Executable G	ienerator					×
Build	SD Card Partition	Advanced Settings	User Files	Security Settings			Help
	Compiled for 32 bit Warning wait t Warning wait t Warning wait t Flash Optimiza Remaining Free Compilation de Also compiled 16 bi	architecture ime expressions will u ime expressions will u ime expressions will u tion turned off, select ce Usage: 4200 bytes e Space: 12184 bytes tails it architecture for Wiv	se ms time o se ms time o se ms time o ted neoVI do s (25.635 % c (74.365 % /I export	dock. Hide this warn dock. Hide this warn dock. Hide this warn dock. Hide this warn oesn't support it. %)	ing by forcing time prec ing by forcing time prec ing by forcing time prec ing by forcing time prec	ision. ision. ision.	
Compile (Wireless Expor	CoreMini Compiled with Wa neoVI t a .wivi package	arnings at UTC 201 for use with Wirele	9/09/05 1 ess neoVI	2:18:17:25000.	Compile	Export	to Clipboard
Downloa	ad to Device						
Select	(USB) neoECU (C/ tool to communicate) neoECU12 (NE00	AN) RS232/UART with neoECU Value 40) App Version 1.55	CAN4-2 V20 - Bootloade	1892 ·	HS CAN Configure Device	Poll for neoE	CUs
					Firmware is up to date (Flash Firmware	(1.55)
	Send	Clear		Show A	dvanced Settings		
Success	fully erased.						

Click on the "Flash Firmware" button to update the firmware and get it out of bootloader mode.

Download to Device	
neoVI (USB) neoECU (CAN) RS232/UART	
Select tool to communicate with neoECU ValueCAN4-2 V20892	HS CAN Poll for neoECUs
neoECU12 (NE0040) App Version 1.55 - Bootloader Mode	Configure Device Hardware Setup
	Firmware is up to date Flash Firmware (1.55)
Send	Show Advanced Settings

After the device is successfully updated you will see that it no longer shows "Bootloader Mode" and LED 3 is blinking a magenta color. During the flash sequence LED 3 blinks an orange color.

Download to Device	
neoVI (USB) neoECU (CAN) RS232/UART	
Select tool to communicate with neoECU ValueCAN4-2 V20892	HS CAN Poll for neoECUs
meoECU12 (NE0040) App Version 1.55	Configure Device By Hardware Setup
	Firmware is up to date Flash Firmware (1.55)
Send Clear	Show Advanced Settings
•	
Successfully bootloaded.	

If the neoECU tab is blank as shown below click on the "Poll for neoECUs button. If the neoCU-12 does not show then there is probably a baud rate mismatch.

Download to Device	
neoVI (USB) neoECU (CAN) RS232/UART	
Select tool to communicate with neoECU ValueCAN4-2 V20892	HS CAN Poll for neoECUs
	Configure Device 🕮 Hardware Setup
	If firmware is out of date, Flash Firmware
Send Clear	Show Advanced Settings
blank	

4.3 Hardware Setup

Next, click on the "Hardware Setup" button to set the parameters for the HS CAN 1, HS CAN 2, and LIN Busses and for all the peripherals on the neoECU-12.

Download to Device neoVI (USB) neoECU (CAN) RS232/UART	
Select tool to communicate with neoECU ValueCAN4-2 V20892	HS CAN Poll for neoECUs
meoECU12 (NE0040) App Version 1.55	Configure Device Hardware Setup
	Firmware is up to date Flash Fighware (1.55)
Send Clear	Show Advanced Settings
Successfully bootloaded.	

The neoVI Explorer window should pop-up. The neoECU-12 should be listed. If not, click on the "Search for Devices' button at the bottom of this window. Select the neoECU-12 and click the "Connect" button.

Connect	Disconnect	Serial No: N/A
Read Settings	Write Settings	Firmware Version
Load Defa	ult Settings	
System Settings		
neoECU12 (NEOO	40) App Version 1.55	

After connecting you will see the neoECU-12 Serial Number and Firmware version.

🔤 neoVI Explorer		
File	Disconnect	Serial No: NE0040
Read Settings	Write Settings	License: neoECU Firmware Versions
Load Defa	MPIC: 1.55	
– System Settings – Available Firmware		
 ⊢-1⊜ neoECU12 (NE004 General Settings Product Details 	UJ App Version 1.55	

Next click on "Network Enables".





In this example you can see HS CAN 1, HS CAN 2, and LIN is enabled.

Now if you select "LSFT CAN" as well you can see it is enabled under the Selectable Networks 2 but HS CAN1 remains selected as well. Since they share the same pin the scenario is not desirable.



Now click on LSFT CAN1. You will see that it is actually disabled.



The best method for selecting the network is to click on "Selectable Networks 1" or "Selectable Networks 2" and from the pull-down menu make your selection.

File		T.
Connect	Disconnect	
Read Settings	Write Settings	Owcan
Load Defa	ult Settings	LISETCAN
Solectable Notwo Sole	the 2	

4.4 CAN Setup

Click on HS CAN1 (or HS CAN2) then click on the Enabled box to enable CAN and click on the CAN FD Enabled box to enable CAN FD. Set the baud rate appropriately. Click on the Write Settings button after you make any changes.

Connent] [Disconnect	HS CAN1		CAN FD	150
Read Settings	Write Settings	Baud Rate	😨 Specify by Baud	Baud Rate	
Load Default	Seitings	TQ SEGL 63	Sync 16	TQ SEGI 15	Sync 4
System Settings	/	10 5067 16	SEF-1 1	10 5052 4	BEF-3 1
Available Finnware R nenECU12 (NE0040)	Ann Version 1 55	TQ Prop 0	(Clock is 80 MHz)	TQ Prop 0	TDC 0
- General Settings		Bit Rat	e Calculator		
→ CLSFT CAN2 → CLSFT CAN2 → Selectable Networks → HS CAN2 → LSFT CAN1 → SW CAN1	12				
S- LIN 					
S LIN 	l Settings	wedECU12 (NECO4O) Ap	p Version 1.55 settings have be	een road.	

4.5 LIN Setup

Click on LIN1 then click on the Enabled box to enable LIN. Set the baud rate appropriately. Most cases the baud rate is 10417 or 19200. If the neoECU-12 is going to act as the LIN Master click on the Master Resistor On box. Click on the Write Settings button after you make any changes.

Connect Disconnect		
Connect Disconnect		
T. successives	LIN 1	
Read Settings Write Settings	Baud Rate	
Load Default Settings		
System Settings Available Finnware ConceCU12 (NE0040) App Version 1.55 General Settings Product Details Selectable Networks 1 Concerning Selectable Networks 2 Selectable Networks 2 Selectable Networks 3 Concerning Selectable Networks 3 Concerning Selectable Networks 3 Concerning Selectable Networks 3 Concerning Selectable Networks 3 Selectable Net	Normal Mode • W Master Resistor On Advanced Options	LIN Master node requires Master Resistor On box to be checked
MISC ID Initial Values Advanced Analog Settings Iso 15785-2 Test API Network Enables	weaECU12 (NECO4O) App Version 1.55 settings hav	re been read.
Search For Devices		

4.6 MISC IO Setup

Click on Initial Values then click on the pull-down menu to select Digital Output (Initially Off), Digital Output (Initially On), or Digital Input.

e			
Cunnect	Disconnect	Initial Setup	
Read Settings	Write Settings	MISC 1 Digital Output (Initially Off)	
Load Defaul	Soltings	HISC 3 Digital Input	
System Settings Assiluble Frances		MISC 4 Digital Output (Initially Off)	
IncoECU12 (NE0040	App Version 1.55	MISC 5 Digital Output (Initially Off) +	
 General Settings Product Details Selectable Meteory 	3	MISC 7 Digital Output (Initially Off) +	
HS CAN1			
Selectable Network HS CAN2 LSFT CAN1 SW CAN1	*2		
	/		
HISC 10	g Settings	neoECU12 (NEC040) App Version 1.55 setting: have been read.	
– Text API – Network Enables			

4.7 Program CoreMini

Next you will program the neoECU-12 with your custom script(s) that you created. But first click on the "Advanced Settings" tab and make sure the Architecture is set to <u>32 Bit</u> and not 16 Bit.

CoreMini Executable Generator	X
Build SD Card Partition Advanced Settings User Files Security Settings	Help
Reserved memory for J1939 Transport Protocol : 1024 Bytes	
Optimization Level : On - Use Flash	
Read Data Step : Max Values Per File Line : 6 (Default)	
Allow device to sleep when USB is connected :	
Num Records Persistent Logging : 2000 👻	
Architecture: 32 Bit 💌	
Compile CoreMini	
Complied with Warnings at 01C 2019/09/05 12:18:17:25000.	Copy to Clipboard
Wireless neoVI	
Export a .ww package for use with Wireless neovi	Export WW File
Download to Device	1
Select tool to communicate with peoFCU_ValueCAN4.2 V20802	Poll for peoECLIs
medecular (NE0040) App Version 1.55	
Configure Device	Hardware Setup
Firmware is up to date	Flash Firmware (1.55)
Send Clear Show Advanced Settings	13
Current fills have been dead	
Successfully bootloaded.	

After selecting 32-bit click back to the Build tab. Click on the "Compile" button, followed by clicking on the "Clear" button, followed by clicking on the "Send" button. Once the progress bar is complete you have programmed the CoreMini of the device. The message at the bottom of the screen will show "Successfully downloaded" and the neoECU 12 shows "Running CoreMini".

CoreMini Executable Generator						
Build SD Card Partition Advanced Settings User Files Security Settings Help						
 ✓ Compiled for 32 bit architecture Warning wait time expressions will use ms time dock. Hide this warning by forcing time precision. Warning wait time expressions will use ms time dock. Hide this warning by forcing time precision. Warning wait time expressions will use ms time dock. Hide this warning by forcing time precision. Warning wait time expressions will use ms time dock. Hide this warning by forcing time precision. Warning wait time expressions will use ms time dock. Hide this warning by forcing time precision. Warning wait time expressions will use ms time dock. Hide this warning by forcing time precision. Warning wait time expressions will use ms time dock. Hide this warning by forcing time precision. Warning wait time expressions will use ms time dock. Hide this warning by forcing time precision. Warning wait time expressions will use ms time dock. Hide this warning by forcing time precision. Warning wait time expressions will use ms time dock. Hide this warning by forcing time precision. Warning wait time expressions will use ms time dock. Hide this warning by forcing time precision. Warning time expressions will use ms time dock. Hide this warning by forcing time precision. Warning time expressions will use ms time dock. Hide this warning by forcing time precision. Warning time expressions will use ms time dock. Hide this warning by forcing time precision. Warning time expressions will use ms time dock. Hide this warning by forcing time precision. Warning time expressions will use ms time dock. Hide this warning by forcing time precision. Warning time expressions will use ms time dock. Hide this warning by forcing time precision. Warning time expressions will use ms time time expressions. Warning time expressions will use ms time time express						
Compile CoreMini Compiled with Warnings at UTC 2019/09/05 13:00:47:65200. Compile]					
Wireless neoVI Export a .wivi package for use with Wireless neoVI Export WiVI File	ו					
Download to Device neoVI (USB) neoECU (CAN) RS232/UART	٦,					
Select tool to communicate with neoECU ValueCAN4-2 V20892 HS CAN Poll for neoECUs]					
Configure Device B Hardware Setup						
Firmware is up to date Flash Firmware (1.55)						
Send Clear Show Advanced Settings						
Successfully downloaded.						

5.0 Function Block Scripts

Next you will learn to program the neoECU-12 Function Block Scripts. From the Scripting and Automation pull-down menu select Function Blocks.

le Setup Spy N	letworks Measurement Embedded Tools	Scripting and Automation	Run Tools Help
• Offline	📖 🖫 📴 Platform: (None)	Function Blocks	📔 🚷 🞯 Desktop
		😡 C Code Interface	
	Logon Name	😂 Application Signals	DataLogger
■ Go	neoFCU-12 examples	🛛 🚾 Text API Terminal	cleScape DAO

Next click on the "+" symbol and select "Script".

File Setup Spy N	etworks Measurement	Emb
▶ ▼ Offline	🕎 🐻 🌬 Plat	form:
Function Blocks		
+ - X 🖻 🖬	1 × 🕈 🖊 🛞	7
🧾 Script	•	Туре
Capture	7	5
Playback		

Click on the "Function Block 1" Description to rename it. For example, rename it to "MISC IO as inputs". This is not mandatory but it is good coding practice to name your scripts that are meaningful for debugging and code re-use.

🚍 Function Blocks								
+ - % 🖻 🖻 ∽ 🛧 🖊 ⊗ 🗗 🛩 🔚 🔍								
Key Descripti	ion	Туре	Start Type	Running		1	Status	
	7	Y		Y				
tst0 Function	Block 1	Script	Immediate	Stopped			Function	
	Volick to	ranama						
Script Start	Notes	remaine	Functi	ion Block 1				
🕂 After	🕈 After 🗣 Before 💻 👗 🛍 🛍 🗠 🗊 No Errors							
Step	Description	Value					Comment	
1	// TODO: Add step commands h	here						
2								
3								

5.1 MISC IO as input(s)

Next double-click on the empty cell row 2 column Description to bring up a pull-down menu of all the scripting commands. Select "Set Value".



Next click on the empty cell row 2 column Value. Note the red text "The SetValue expression could not compile" message. Please note all error messages needs to be corrected before the script can be compiled and programmed into CoreMini. Once we set the value correctly the error message will go away.

Script	Start	Notes	MISC IOx as inputs	
- A	fter	🕈 Before 📃 🖁	🗈 🛍 🗠 🗊 The SetValue expression could not	t compile.
	Step	Description	Value	Comment
	1	// TODO: Add step commands here		
	2	Galue Set Value	Click For Setup	
	3			

Next double-click where it shows "Click For Setup" The Expression Builder pop-up window will show. Click on "Physical IO" followed by "Misc IO" followed by the "+" symbol.

ntar Expression for sign	al		Help OC Cancel
No. of the particular		Cator	Format Etvaluate as text
Value To Set		Farat	+
Expression		til line	Piece
Clear		+	Chiprete Values
5 Expression Builder	W		
Rx Messages	Set Syl + Nois bet halve Add To Digression (Add Operator	Calculator Panel
Database	Red Dear /		
Signal Groups	H 🗊 An VIET - 5		7 8 9 +
E DAO	III 🔨 Asalog Irputs 👩	¢	4.1.8.1.8.
Dobs		2	
TApp Signals	Canada and		La Mandatatata
P:Networks	D T LEOR		P. Handination County
Nodes	H A HC D	itO(Naros
Misc	E Novi File Misc	iit1(iit26	
Prunction Blocks	B Power Management	its(and .
	B D we Suply	itt itt	Walder Street Street St
Cameras	It we operate	it6((c) the sala-generated term
User Signals	Property a	nit7(10%m
	b	ytel(- NEW C
		iyte2(
	[a]	WC034	

Next click on "MISC IO 1" followed by double-clicking on the "IS Output" property. This adds MISC IO 1 (Is Output) to the "Value to Set" box. Then put "0" for the Expression and click the OK button. "0" = input and "1" = output.

anaw sapreción					-	
Enter Expression for signa Vela Ta Set (MSC KO L (Ja Ou Expression of Clear	n aputi :maal mib-1-mikra(0)) 		E Carton Forest Inste	Parnet E	Columba and b Columba and b Columb	aut Carcel
& Expression Builder	22					
**Rx Messages **Rx Messages **Tx Messages **Signal Groups **DAQ **DAQ **Dobs **App Signals **Nodes **Nodes **Misc **Misc **Misc **Disca for **Cameras **User Signals	Set By: - Rick Set Hakas Add To Expression Find One If An VIET If Add To Expression If An VIET If Add To Expression If A Add To Expression<	Add Ope / = > < >= >= >= >= >= >= >= >= >= bit0(bit1(bit2(bit2(bit4(bit5(bit5(bit5(bit7(bytca)(retur E	Celouis 7. 4. 7. 4. 7. 4. 7. 4. 7. 7. 4. 7. 7. 4. 7. 7. 7. 7. 7. 7. 7. 7. 7. 7. 7. 7. 7.	or Panel (+ 4 9 0 3 0 2 3 4 3 stor Sguk	t i

Note the Value column is now filled and the error message was replaced by "No Errors".

Script	Script Start Notes MISC IOx as input					
🕈 Af	🕇 After 🕇 Before 💻 🕺 🖹 🛍 🛍 🗠 🗊 No Errors					
	Step	Description	Value	Comment		
	1	// TODO: Add step commands here				
	2		$\{MISC IO 1 (Is Output) : neo0-mi0-1-index(0)\} = 0$			
	3					

Next add the appropriate comment by double-clicking in the comment box to the right.

1	// TODO: Add step commands here		
2	Set Value	{MISC IO 1 (Is Output) :neo0-mi0-1-index(0)} = 0	// MISC IO 1 set as input
3			

Use the same format to setup the remaining MISC IO as inputs:

1	// TODO: Add step commands here		
2	Set Value	$\{MISC IO 1 (Is Output) : neo0-mi0-1-index(0)\} = 0$	// MISC IO 1 set as input
3	-•• Set Value	MISC IO 2 (Is Output) :neo0-mi1-1-index(0) = 0	// MISC IO 2 set as input
4	-•• Set Value	{MISC IO 3 (Is Output) :neo0-mi2-1-index(0)} = 0	// MISC IO 3 set as input
5	- O Set Value	{MISC IO 4 (Is Output) :neo0-mi3-1-index(0)} = 0	// MISC IO 4 set as input
6	- O Set Value	MISC IO 5 (Is Output) :neo0-mi4-1-index(0) = 0	// MISC IO 5 set as input
7	- O Set Value	MISC IO 6 (Is Output) :neo0-mi5-1-index(0) = 0	// MISC IO 6 set as input
8	-• Set Value	{MISC IO 7 (Is Output) :neo0-mi6-1-index(0)} = 0	// MISC IO 7 set as input

The next section is optional.

Add variables (Application Signals) to your script. From the Scripting and Automation pull-down menu select Application Signals.

😇 Vehicle Spy 3 Enterprise		
File Setup Spy Networks Measurement Embedded To	ols Scripting and Automation	Run Tools Help
Offline Platform: (None) Eurotion Blocks ····	Function Blocks	🔹 🔌 🞯 Desktop 1
	Application Signals	
▼ ─ ゐ ≝ ⊯ ♥ ♥ ♥ ♥ ₩ ₩	Text API Terminal	
Key Description Type S	tart Type Running	📂 🥮 🛝 🞚 Status

Next click on the "+" symbol to add "Application Signals". These are like variables in C code. Clicking on the "+" symbol again to add additional application signals and clicking on the "-" symbol will delete the application signal that is highlighted.

File Setup Spy	Networks	Measurement	Embe
Diffline		🔲 🌬 Platf	orm: (I
Function Block	s 🖾 🗳	Application Signal	s 🔛
😹 + 📒 🐰	B	v 🖻 🖬	
Filter	1		
	-		

Next rename the variable to something that is more meaningful. Click on the description and backspace to rename. Since we are creating a variable for MISC IO 1 input let's name it "var_misc_io_1_input" for example.



Description	
app_sig_misc_io_1	
Signal Type Digital	Format True/False -
The Appli Analog	Array Size 8
Value Type State Encoded	
Text This value is used as a general	purpose variable
	Initial Value 0

Next set the Signal Type to "Digital". When we read the port it will be "low" or "high".

Do this for the remaining MISC IO inputs. Note the Format is defaulted to "True/False" but you can switch to "On/Off", "Yes/No", "Passed/Failed", "Open/Closed", etc...

🛃 + 🗕 👗 🖻 💼 🗠 🕞 🖼	Description
Filter app_sig_misc_io_1 (sig4) app_sig_misc_io_2 (sig5) app_sig_misc_io_3 (sig6) app_sig_misc_io_4 (sig7) app_sig_misc_io_5 (sig8) app_sig_misc_io_6 (sig9) app_sig_misc_io_7 (sig10)	app_sig_misc_io_1 Signal Type Digital Format True/False The Application Signal is an array Array Size 8 Value Type This value is used as a general purpose variable Initial Value 0

Next, let's read the ports and assign the status to their respective variables. Select "Set Value" as the command double-click in the Value column. Click on App Signals then double-click on "app_sig_misc_io_1".

🔚 Enter Expression				a a market a
Enter Expression for signa Value To Set (app_rig_mic_io, Expression 	al ,1 (/mhai) :mg+0)	4	EGuston Formet (Units (Help DK Cancel tormat Cvaluate as text True=1Folse= + Image: 0 true=1Folse= + Image: 0 Image: 0 Image: 0
A Expression Builder Rx Messages Database Tx Messages Signal Groups DAQ DOS App Signals App Signals App Signals Misc Function Blocks Physical IO Logger Cameras User Signals	Sort By: Pick Set Value Add To Expression Picd Obsi ** exp.sp.ser.1 ** exp.sp.ser.3 ** exp.sp.ser.3 ** exp.sp.ser.3 ** exp.sp.ser.6.2 ** exp.sp.ser.6.3 ** exp.sp.ser.6.3 ** exp.sp.ser.6.3 ** exp.sp.ser.6.3 ** ** exp.sp.ser.6.3 ** ** exp.sp.ser.6.5 ** ** exp.sp.ser.5.5 ** ** exp.sp.ser.5.5 ** ** exp.sp.ser.5.5 ** ** exp.sp.sp.ser.5.7 **	Add Op / - > < < < >= >> < >> < bit0(bit1(bit2(bit3(bit4(bit5(bit6(bit7(byte0(byte1(byte3(one(and	rator *	Calculator Panel

Next, click on "Physical IO" followed by "MISC IO 1" followed by "Value" followed by "Add to Expression". Then click "OK".



Repeat for all MISC IO. Below shows the MISC IO ports to be read into their respective variable.

10) Set Value		// read misc io 1 into app signal 1
11	Set Value		// read misc io 2 into app signal 2
12	2 Set Value	{app_sig_misc_io_3 (Value) :sig6-0} = {MISC IO 3 (Value) :neo0-mi2-0-index(0)}	// read misc io 3 into app signal 3
13	3 Set Value		// read misc io 4 into app signal 4
14	Set Value	{app_sig_misc_io_5 (Value) :sig8-0} = {MISC IO 5 (Value) :neo0-mi4-0-index(0)}	// read misc io 5 into app signal 5
15	5 Set Value	{app_sig_misc_io_6 (Value) :sig9-0} = {MISC IO 6 (Value) :neo0-mi5-0-index(0)}	// read misc io 6 into app signal 6
16	5 Set Value	{app_sig_misc_io_7 (Value) :sig10-0} = {MISC IO 7 (Value) :neo0-mi6-0-index(0)}	// read misc io 7 into app signal 7

End of optional section.

5.2 Send MISC IO port status in CAN message.

From the Spy Networks pull-down menu select Message Editor.



Click on the Transmit button then click on the "+" symbol to create a new HS CAN message.

📑 Function Blocks 🖾 🗳 Application Signal			gnals 🔝 🕫	Messages Edito	r 🔜 🖳	Tx Panel 🔀 🕼 I	Messages	23			
Edit	•v• <u>Receive</u>	므 <u>Transmit</u>		abase o	n Networ	k HS CAN			+		
Кеу	Description	Ty	/pe	Arb ID	Multi	Len	B1	52	B3	B4	
	Y		7	7	Y	7		Y	\mathbf{A}	Y	2
out1	Tx Message HS CAN 1	C/	AN Std 11 bit		None	· · · · · ·					

Click on "Tx Message HS CAN 1" to rename it (i.e. misc_io_input_status) and set the Arb ID to a value not being used (i.e. 101).

🙄 A	📽 Application Signals 🔯 ∾ Messages Editor 🜌 🙆 Messages 🖾 🚍 Function Blocks 🖾 📴 Signal Plot 🔯									
Edit	<u> ∾v¤ Receive</u>	📙 <u>Trans</u>	mit 📄 🗊 Databa	se on Networ	k HS CAN		• +			
Key	Description	1	Туре	Arb ID Multi	Len	B1 B2	2 B3	B4		
	7	/	Y	7	Y	YY	Y	A.		
out1	misc io input status		CAN Std 11 bit	101 None						

Add seven 1-bit CAN signals to this message. Click on the down arrow next to "+8" and select "+1". Then click the "+1" six more times.



Signals in Message								
♣ 8 ▼ − ♠	ŧ							
Signals in Message		Byte 1	Byte 2					
Description	Туре	76543210	765432					
can sig misc io 7 input	Digital	0						
can_sig_misc_io_6_input	Digital	0						
can_sig_misc_io_5_input	Digital	0						
can_sig_misc_io_4_input	Digital	0						
can_sig_misc_io_3_input	Digital	0						
can_sig_misc_io_2_input	Digital	0						
can_sig_misc_io_1_input	Digital	0						

Next rename the signals to something more meaningful (i.e. can_sig_misc_io_1_input).

Next modify the script to get the MISC IO port value and copy it into its respective CAN signal. In this example the port is read and copied into the CAN signals then the CAN message is sent every 50 ms.

*NOTE – If you had done the optional section above and created application signals and read the MISC IO into the application signals then you would copy the application signals to their respective can signals. The image shown below is copying the MISC IO directly to their respective can signals. Either way works.

26 -•• Set Value	{can_sig_misc_io_1_input (Value) :out1-sig6-0} = {MISC IO 1 (Value) :neo0-mi0-0-index(0)}	// copy misc io 1 to can signal 1 $% \left(1 \right) = \left(1 \right) \left(1 \right$
27 -•• Set Value	{can_sig_misc_io_2_input (Value) :out1-sig5-0} = {MISC IO 2 (Value) :neo0-mi1-0-index(0)}	// copy misc io 2 to can signal 2
28 -•• Set Value	{can_sig_misc_io_3_input (Value) :out1-sig4-0} = {MISC IO 3 (Value) :neo0-mi2-0-index(0)}	// copy misc io 3 to can signal 3
29 -•• Set Value	{can_sig_misc_io_4_input (Value) :out1-sig3-0} = {MISC IO 4 (Value) :neo0-mi3-0-index(0)}	// copy misc io 4 to can signal 4
30 -•• Set Value	{can_sig_misc_io_5_input (Value) :out1-sig2-0} = {MISC IO 5 (Value) :neo0-mi4-0-index(0)}	// copy misc io 5 to can signal 5
31 Set Value	{can_sig_misc_io_6_input (Value) :out1-sig1-0} = {MISC IO 6 (Value) :neo0-mi5-0-index(0)}	// copy misc io 6 to can signal 6
32 Set Value	{can_sig_misc_io_7_input (Value) :out1-sig0-0} = {MISC IO 7 (Value) :neo0-mi6-0-index(0)}	// copy misc io 7 to can signal 7
33 🙁 🖄 Wait For	= 50 ms	// delay 50 ms
34 🖳 Transmit	misc io input status (out1)	// transmit status message
35 🔟 Jump To	Step 10	// read misc io inputs again

Program CoreMini with this script. Connect another Intrepid tool to the neoECU-12 and monitor the HS CAN bus. You should see the status of all seven input ports.

*NOTE – If you do not connect a voltage to the pins they tend to float high (3.3V) except for IO7 which is a 5V tolerant IO pin. It floats low as you cen see from the image below. All of the other pins floated high except IO3 which was connected to GND.

	Count	Time (abs/rel)	Тх	Er	A Description	ArbId/Header	Len	DataBytes
Filter								
- •4•	267	56.006 ms			misc io input status	101	1	76
*	can_sig_misc	_io_7_input			= False [0]			
*	can_sig_misc	_io_6_input			= True [1]			
*	can_sig_misc	_io_5_input			= True [1]			
*	can_sig_misc	_io_4_input			= True [1]			
*	can_sig_misc	:_io_3_input			= False [0]			
*	can_sig_misc	_io_2_input			= True [1]			
*	can_sig_misc	io_1_input			= True [1]			

5.3 MISC IO as output(s)

Setting the MISC IO as outputs is the same procedure as setting the port to an input but the expression is set to "1" instead of "0".

1	// TODO: Add step commands here		
2	Set Value	MISC IO 1 (Is Output) := 1	// MISC IO 1 set as output
3	Set Value	MISC IO 2 (Is Output) := 1	// MISC IO 2 set as output
4	Set Value	MISC IO 3 (Is Output) := 1	// MISC IO 3 set as output
5	Set Value	MISC IO 4 (Is Output) := 1	// MISC IO 4 set as output
6	Set Value	MISC IO 5 (Is Output) := 1	// MISC IO 5 set as output
7	Set Value	MISC IO 6 (Is Output) := 1	// MISC IO 6 set as output
8	Galue Set Value	$\{ MISC \ IO \ 7 \ (Is \ Output) : neo0-mi6-1-index(0) \} = 1$	// MISC IO 7 set as output

To write to the port set the MISC IO value property. In the example below the ports are alternated "high" and "low".

9		
10 🔤 Set Value	$\{MISC IO 1 (Value) : neo0-mi0-0-index(0)\} = 1$	// MISC IO 1 is set high
11 Set Value	{MISC IO 2 (Value) :neo0-mi1-0-index(0)} = 0	// MISC IO 2 is set low
12 🔤 Set Value	{MISC IO 3 (Value) :neo0-mi2-0-index(0)} = 1	// MISC IO 3 is set high
13 🔤 🕫 Set Value	{MISC IO 4 (Value) :neo0-mi3-0-index(0)} = 0	// MISC IO 4 is set low
14 Set Value	{MISC IO 5 (Value) :neo0-mi4-0-index(0)} = 1	// MISC IO 5 is set high
15 🔤 Set Value	{MISC IO 6 (Value) :neo0-mi5-0-index(0)} = 0	// MISC IO 6 is set low
16 🔤 Set Value	{MISC IO 7 (Value) :neo0-mi6-0-index(0)} = 1	// MISC IO 7 is set high
17		

5.4 MISC IO as PWM outputs

There are five PWM Outputs. Please note channel 3 PWM is mirrored on channel 6 PWM.

Setting the MISC IO as PWM outputs is like setting the MISC IO as outputs. After selecting the "Set Value" command double-click where it shows "Click For Setup". The Expression Builder pop-up window will show. Click on "Physical IO" followed by "PWM Outputs" followed by the "+" symbol. Select PWM Output 1 and then double-click on "PWM Frequency". Set the frequency in the Expression box. In this example, the frequency is set to 100 Hz.



Continue setting the PWM Output 2 thru 5 to 100Hz as shown below.

2	Set Value	{PWM Output 1 (PWM Frequency (Hz)) :neo0-po0-1-index(0)} = 100	// PWM 1 freq set to 100 Hz
3	-• Set Value	{PWM Output 2 (PWM Frequency (Hz)) :neo0-po1-1-index(0)} = 100	// PWM 2 freq set to 100 Hz
4	-• Set Value	{PWM Output 3 (PWM Frequency (Hz)) :neo0-po2-1-index(0)} = 100	// PWM 3 freq set to 100 Hz
5	-• Set Value	{PWM Output 4 (PWM Frequency (Hz)) :neo0-po3-1-index(0)} = 100	// PWM 4 freq set to 100 Hz
6	-• Set Value	{PWM Output 5 (PWM Frequency (Hz)) :neo0-po4-1-index(0)} = 100	// PWM 5 freq set to 100 Hz
7			

Repeat the same steps for setting the Duty Cycle for all five PWM channels.



In this example the duty cycle is 10%, 30%, 50%, 70%, and 90%, respectively for the five PWM channels.

2	•• Set Value	{PWM Output 1 (PWM Frequency (Hz)) :neo0-po0-1-index(0)} = 100	// PWM 1 freq set to 100 Hz
3	Set Value	{PWM Output 2 (PWM Frequency (Hz)) :neo0-po1-1-index(0)} = 100	// PWM 2 freq set to 100 Hz
4	Set Value	{PWM Output 3 (PWM Frequency (Hz)) :neo0-po2-1-index(0)} = 100	// PWM 3 freq set to 100 Hz
5	Set Value	{PWM Output 4 (PWM Frequency (Hz)) :neo0-po3-1-index(0)} = 100	// PWM 4 freq set to 100 Hz
6	Set Value	{PWM Output 5 (PWM Frequency (Hz)) :neo0-po4-1-index(0)} = 100	// PWM 5 freq set to 100 Hz
7			
8	- Set Value	{PWM Output 1 (PWM Duty (%)) :neo0-po0-2-index(0)} = 10	// PWM 1 duty cycle set to 10%
9	•• Set Value	{PWM Output 2 (PWM Duty (%)) :neo0-po1-2-index(0)} = 30	// PWM 2 duty cycle set to 30%
10	- Set Value	{PWM Output 3 (PWM Duty (%)) :neo0-po2-2-index(0)} = 50	// PWM 3 duty cycle set to 50%
11	•• Set Value	{PWM Output 4 (PWM Duty (%)) :neo0-po3-2-index(0)} = 70	// PWM 4 duty cycle set to 70%
12		{PWM Output 5 (PWM Duty (%)) :neo0-po4-2-index(0)} = 90	// PWM 5 duty cycle set to 90%
13			

Below is a screenshot of the five PWM channels. Each measured 10%, 30%, 50%, 70%, 90% duty cycles, respectively.

· Control		0.e.d.e.		
atart			+30.46	
00 PMH 1	0.45	6		
ET PANY 2	0.15			
AZ PWM3	0.15			ī
US PAINA	0 IF			
Die PWM 6	• 5	10 U		

Note: The PWM output is 3.3V signal driven by a series 10K resistor. Please buffer accordingly to drive higher current loads.

5.5 MISC IO as Analog Inputs

There are four 0-5V Analog inputs. These are connected to 12-bit A/D Converters so the maximum value read from the A/D Converter is 4095 counts.

The next section is optional (just like it was for the MISC IO as inputs).

Add variables (Application Signals) to your script. From the Scripting and Automation pull-down menu select Application Signals. Add 4 variables and rename them appropriately. Note the Signal Type is set to "Analog" and the Max Value is set to "4095".

🙄 Application Signals 🖾 💀 Messages Edi	tor 🖾 🐵 Messages 🖾 🚍 Function Blocks 🖾
🚰 + - X 🖻 🖻 ∽ 🚔 🖬 Filter	Description app_sig_ain_1
app_sig_ain_1 (sig0) app_sig_ain_2 (sig1) app_sig_ain_3 (sig2)	Signal Type Analog Format Units Min 0.0000000 Max 4095.0000000
app_sig_ain_4 (sig3) app_sig_misc_io_1 (sig4) app_sig_misc_io_2 (sig5) app_sig_misc_io_3 (sig5)	The Application Signal is an array Array Size 8
app_sig_misc_io_5 (sig0) app_sig_misc_io_4 (sig7) app_sig_misc_io_5 (sig8) app_sig_misc_io_6 (sig9)	 This value is used as a general purpose variable Initial Value

Now read the analog inputs into the variables.

2	-•• Set Value	{app_sig_ain_1 (Value) :sig0-0} = {Analog Input 1 (Value) :neo0-ai0-0-index(0)}	// read ain 1 into app signal 1
3	-•• Set Value	{app_sig_ain_2 (Value) :sig1-0} = {Analog Input 2 (Value) :neo0-ai1-0-index(0)}	// read ain 2 into app signal 2
4	-•• Set Value	{app_sig_ain_3 (Value) :sig2-0} = {Analog Input 3 (Value) :neo0-ai2-0-index(0)}	// read ain 3 into app signal 3
5	-•• Set Value	{app_sig_ain_4 (Value) :sig3-0} = {Analog Input 4 (Value) :neo0-ai3-0-index(0)}	// read ain 4 into app signal 4

End of optional section.

5.6 Send Analog input values in CAN message

From the Spy Networks pull-down menu select Message Editor.



Click on the Transmit button then click on the "+" symbol to create a new HS CAN message.

📑 F	unction Blocks 📧 😫	Application Sign	nals 🔝 🚾 Me	ssages Edito	r 🔜 🖳	Tx Panel 🔀 🔋 I	Messages	23			
Edit	o∿o <u>Receive</u>	<u> Transmit</u>	Databas	se o	n Networ	k HS CAN		•	+		
Кеу	Description	Тур	e	Arb ID	Multi	Len	B1	22	B3	B4	
	Y		Y	Y	Y	7		Y	\mathbf{A}	Y	ľ
out1	Tx Message HS CAN 1	CAI	N Std 11 bit		None	· · · · ·					

Click on "Tx Message HS CAN 3" to rename it (i.e. ain_status) and set the Arb ID to a value not being used (i.e. 103).

🙄 A	🍄 Application Signals 🔀 💀 Messages Editor 🔀 🕼 Messages 🔀 🚍 Function Blocks 🔀												
Edit	<u> P∿¤ Receive</u>	📙 <u>Trans</u> i	<u>mit</u>	Databas	<u>se</u> 0	n Networl	k	HS CAN			• •		
Кеу	Description	1	Туре		Arb ID	Multi	Len	1	B1	B2	B3	B4	
	Y			7	7	7	/	Y	Y	∇	\mathbf{A}	Y	
out1	misc io input status		CAN Std	11 bit	101	None							
out2	misc io output straus		CAN Std	11 bit	102	N je							
out0	ain_status		CAN Std	11 bit	103	None							

Add four 16-bit CAN signals to this message. Click on the down arrow next to "+8" and select "+16". Then click the "+1" three more times.

1	Sig	nak	s in Message
	÷	8	- + +
Si	÷	1	Add 1 Bit Signal
D	¢	8	Add 8 Bit Signal
	÷	16	Add 16 Bit Signal

Next rename the signals to something more meaningful (i.e. can_sig_ain_1).

Signals in Message									
◆ 8 ★ ↓				Equation (F	iaw Value] (0,1/	0, 3 6	fa.	Edit	e Edit
Signals in Massage	L.	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7	Byte 8
Description	Type				3 8 7 8 5 4 7 1		32245455	10201401	
can_sk_an_1	Analog			1 A CONTRACTOR	and an track of		and a protocol of the	No. of the local division of the	A CONTRACTOR OF A
can sig an 2	Anabg	and the second	California (California)	C = +	*******	10 5 30 11			
can_sig_ain_3	Analog			100000		5 5 0 C 3 A	5 8 7 6 2 4 5 2	10	
cansilation.4	Analog							1 2 2 2 4	

Or re-size the 16-bit signal values to 12-bit values by dragging the edge of the blue box inwards.

Signals in Message									
• a + ↓				Equation	(Raw Value) (0,1,4,1	2	10	Edit	Live Edit
Signals in Neesago	31	Byte 1	8yta 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7	Byte 8
Description	Type	7634		7 1 5 #		7654	30268433	107654	**********
ran_sig_an_t	Analog	1000			COLUMN THE COLUMN		the state of the s		Service and the service of the servi
can_sp_an_2	Analog		 A.S. S. S. S. S. S. S. S. S. S. S. 	S - 1		0			
can_sig_an_3	Analog						28762528	1 0	
can_sig_ain_4	Analog								5-31703+3111

Next modify the script to get the analog in value and copy it into its respective CAN signal. In this example the port is read and directly copied into the CAN signals then the CAN message is sent every 10 ms.

2	-•• Set Value	{can_sig_ain_1 (Value) :out0-sig0-0} = {Analog Input 1 (Value) :neo0-ai0-0-index(0)}	// copy ain 1 to can signal 1
3	-•• Set Value	{can_sig_ain_2 (Value) :out0-sig1-0} = {Analog Input 2 (Value) :neo0-ai1-0-index(0)}	// copy ain 2 to can signal 2
4	-•• Set Value	{can_sig_ain_3 (Value) :out0-sig2-0} = {Analog Input 3 (Value) :neo0-ai2-0-index(0)}	// copy ain 3 to can signal 3
5	Set Value	{can_sig_ain_4 (Value) :out0-sig3-0} = {Analog Input 4 (Value) :neo0-ai3-0-index(0)}	// copy ain 4 to can signal 4
6	🗏 Transmit	ain_status (out0)	
7	🖄 Wait For	= 10 ms	

Optionally, if you used a variable and read the analog input port into a variable then you would assign the variable to the appropriate CAN signal as shown below.

7	-•• Set Value	{can_sig_ain_1 (Value) :out0-sig0-0} = {app_sig_ain_1 (Value) :sig0-0}	// copy app sig 1 to can signal 1
8	oset Value	$\label{eq:can_sig_ain_2 (Value) :out0-sig1-0} = \{app_sig_ain_2 (Value) : sig1-0\}$	// copy app sig 2 to can signal 2
9	oset Value	{can_sig_ain_3 (Value) :out0-sig2-0} = {app_sig_ain_3 (Value) :sig2-0}	// copy app sig 3 to can signal 3
10	oset Value	{can_sig_ain_4 (Value) :out0-sig3-0} = {app_sig_ain_4 (Value) :sig3-0}	// copy app sig 4 to can signal 4
11	🗏 Transmit	ain_status (out0)	
12	🖄 Wait For	= 10 ms	

Program CoreMini with this script. Connect another Intrepid tool to the neoECU-12 and monitor the HS CAN bus. You should see the status of all four analog input ports as shown in the example below.

		Count	Time (abs/rel)	Тх	Er	A Description	ArbId/Header	Len	DataBytes	Network
F	ilter									
Ę	- o% o	77128	11.992 ms	;		analog_in_status	101	8	06 7A 06 6B 08 D3 09 10	HS CAN
	*	analog_in_1	level			= 1658 [67A]				
	*	analog_in_2	level			= 1643 [66B]				
	*	analog_in_3	level			= 2259 [8D3]				
	*	analog_in_4	level			= 2320 [910]				

The circuit that was used in the example was four LEDs connected to each AIN port. A yellow LED was connected to AIN 1, a red LED connected to AIN 2, a green LED connected to AIN 3, and a blue LED connected to AIN 4. From the signal plot you can see the forward voltage drop on the yellow and red LED are about the same and the forward voltage drop on the green and blue are about the same. To calculate the forward voltage drop of the LED take the measured counts, divide by 4096 (for 12-bit A/D) then multiply by 5 (for 5V maximum voltage).

LED	calculation	
yellow	1658 / 4096 * 5 =	2.02 V
red	1642 / 4096 * 5 =	2.00 V
green	2259 / 4096 * 5 =	2.76 V
blue	2320 / 4096 * 5 =	2.83 V

Test circuit used:





5.7 Controlling the Tri-Color LEDs

The neoECU-12 has default behavior for the first two tri-color LEDs. The 1^{st} LED is for HS CAN 1 and the 2^{nd} LED is for HS CAN 2.

Interpretation of RGB LED Colors

These are "RGB" LEDs because they contain separate red, green and blue elements. For networks, each indicates a different aspect of the device's overall status:

- Green: Device is transmitting messages on this channel.
- Blue: Device is receiving messages on this channel.
- Red: Device is detecting errors on this channel.

It is possible for more than one LED component to be lit, producing the following results:

- Green+Blue (Cyan): Device is transmitting and receiving on this channel.
- Green+Red (Yellow): Device is transmitting and detecting errors on this channel.
- Blue+Red (Magenta): Device is receiving and detecting errors on this channel.

• **Green+Blue+Red (White):** Device is transmitting, receiving and detecting errors on this channel.

LED3 blinks magenta when a function block script is running.

In bootloader mode, all five LEDs blink red in succession.

To override the default behavior, use the command "Set Value" and set "LED x (Auto) = 0" as shown below. To restore to default set "LED x (Auto) = 1". To set the color of the LED set the red/green/blue property of the LED to 255 as shown below. 255 is the maximum intensity. A value of 50 would be a very low intensity. You can also mix the red/green/blue property to create additional colors.

Script Start Notes		Notes	LED Control		
- A	fter	🕈 Before 📃 🖁	🗈 🛍 🗠 🗊 No Errors		
	Step	Description	Value	Comment	
	1	// TODO: Add step commands here			
	2	Set Value	$\{LED 1 (Auto) : neo0-ld0-3-index(0)\} = 0$	// user control of LED 1 $$	
	3	Galue Set Value	$\{LED 2 (Auto) : neo0-ld1-3-index(0)\} = 0$	// user control of LED 2	
	4	Galue Set Value	{LED 3 (Auto) :neo0-ld2-3-index(0)} = 0	// user control of LED 3	
	5	oset Value	{LED 4 (Auto) :neo0-ld3-3-index(0)} = 0	// user control of LED 4	
	6	oset Value	{LED 5 (Auto) :neo0-ld4-3-index(0)} = 0	// user control of LED 5	
	7				
	8	oset Value	{LED 1 (Green) :neo0-ld0-5-index(0)} = 255	// set LED 1 to green	
	9	oset Value	{LED 2 (Green) :neo0-ld1-5-index(0)} = 255	// set LED 2 to green	
	10	-® Set Value	{LED 3 (Green) :neo0-ld2-5-index(0)} = 255	// set LED 3 to green	
	11	- Set Value	{LED 4 (Green) :neo0-ld3-5-index(0)} = 255	// set LED 4 to green	
	12	Set Value	{LED 5 (Green) :neo0-ld4-5-index(0)} = 255	// set LED 5 to green	
	13				

The proper way to set the LED color is to assign values to the red/green/blue property. This example shows how to set LED 1 to green color.

	8	• Set Value	$\{LED \ 1 \ (Red) : neo0 - Id0 - 4 - index(0)\} = 0$	// set LED 1 to green
	9	- Set Value	{LED 1 (Green) :neo0-ld0-5-index(0)} = 255	// set LED 1 to green
	10	• Set Value	{LED 1 (Blue) :neo0-ld0-6-index(0)} = 0	// set LED 1 to green

This example shows how to set LED 1 to yellow color.

	8	Set Value	{LED 1 (Red) :neo0-ld0-4-index(0)} = 255	// set LED 1 to yellow
	9	-• Set Value	{LED 1 (Green) :neo0-ld0-5-index(0)} = 255	// set LED 1 to yellow
	10	-• Set Value	{LED 1 (Blue) :neo0-ld0-6-index(0)} = 0	// set LED 1 to yellow

	8	Set Value	{LED 1 (Red) :neo0-ld0-4-index(0)} = 255	// set LED 1 to purple
	9	Set Value	$\{LED 1 (Green) : neo0-ld0-5-index(0)\} = 0$	// set LED 1 to purple
	10	Set Value	{LED 1 (Blue) :neo0-ld0-6-index(0)} = 255	// set LED 1 to purple

This example shows how to set LED 1 to purple color.

This example shows how to set LED 1 to cyan color.

	8	-•• Set Value	{LED 1 (Red) :neo0-ld0-4-index(0)} = 0	// set LED 1 to cyan
	9	-•• Set Value	{LED 1 (Green) :neo0-ld0-5-index(0)} = 255	// set LED 1 to cyan
	10	-•• Set Value	{LED 1 (Blue) :neo0-ld0-6-index(0)} = 255	// set LED 1 to cyan

This example shows how to set LED 1 to white color. (note even though the LED is white for the most part you can see the red filament of the LED turn on)

	8	Set Value	{LED 1 (Red) :neo0-ld0-4-index(0)} = 255	// set LED 1 to white
	9	-• Set Value	{LED 1 (Green) :neo0-ld0-5-index(0)} = 255	// set LED 1 to white
	10	-•• Set Value	{LED 1 (Blue) :neo0-ld0-6-index(0)} = 255	// set LED 1 to white

This example shows how to set LED 1 to orange color. Note the intensity changes from 255 to 100 on the green property.

	8	-•• Set Value	{LED 1 (Red) :neo0-ld0-4-index(0)} = 255	// set LED 1 to orange
	9	-• Set Value	{LED 1 (Green) :neo0-ld0-5-index(0)} = 100	// set LED 1 to orange
	10	-• Set Value	{LED 1 (Blue) :neo0-ld0-6-index(0)} = 0	// set LED 1 to orange

Recommendation: Since LED 1 thru LED 3 are predefined start with using LED 4 and LED 5 for user control.

5.8 LIN example

Below is a simple LIN Master schedule table. Each message is transmitted with 20 ms delay between messages.

Script	Start	Notes	LIN Master schedule table				
+ A	🕈 After 🕈 Before 💻 🐰 🖹 🛍 🛍 ស 🛐 No Errors						
	Step	Description	Value	Comment			
	1	// TODO: Add step commands here					
	2	🚇 Transmit	LIN Master msg1 (out7)	// transmit LIN Master msg1			
	3	🔆 Wait For	= 20 ms	// delay 20 ms			
	4	📇 Transmit	LIN Master msg2 (out8)	// transmit LIN Master msg2			
	5	🔆 Wait For	= 20 ms	// delay 20 ms			
	6	📇 Transmit	LIN Master msg3 (out9)	// transmit LIN Master msg3			
	7	🔆 Wait For	= 20 ms	// delay 20 ms			
	8						

In this example, three LIN Transmit messages were created. "LIN Master msg1" is a message that contains data that will be sent to the LIN slaves. The signals were pre-defined as "01" thru "08" but can be modified in the script. The signals in "LIN Master msg1" were named "Data1" thru "Data8" as shown below. Note the Type is "Master". For LIN messages that will contain data from the slaves the Type needs to be set to "Header Only".

2 F	Function Blocks 🔝 🕻	🗧 Applicatio	n Signals 🔞 🍳	👷 Messages	Editor 🔛 🐰	🗏 Tx Panel 🖇	3 🖲 Messages	8	来	del Ho	ka 🕅	ŝ		
Edit	oho Receive	🗏 Irac	ant 🛛 🖬 G	Jetabose	on Notu	ork LIN		-	•		1 %	6	8	-1
Key.	Description		Гуре		ID CheckSi	um Len	B1	82	83	84	85	86	87	68
	7		7	2	7 7		7 7	7	7	7	7	7	7	7
out7	LIN Master msg1		Master		01 Enhance	ed	01	02	03	04	05	06	07	08
outs.	LIN Master mag2		Header Only		02 Enhance	ed								
out9	LIN Master meg3		Header Only		03 Enhance	ed								
Senals	in Message	12	Byte 1	Byte 2	Byte 3	Byte 4	Byte S	Byte 6	Je La	Byte	7	B	/w 8	
Description	ton	Type				IN REAL PROPERTY.								
Data1	91	Analog	22022											
Data2		Analog			1 1 H									
Data3		Analog			7 4 5 4 2	2 X B								
Data4		Analog				14213								
Data5		Analog					16243330	_	-					
Data6		Analog						260	222	5		-		
Data7		Analog								- 5	8 - 1 - 1 - 2	20		-
1.0152		Analog.									112 6		신민이지	

If we program this example into the neoECU-12 and then use another Intrepid device to monitor the LIN Bus these are the messages on the LIN Bus. Note there is an error for LIN message "LIN 02" and "LIN 03". This is because the monitoring or slave device has not sent back any data to the LIN Master (neoECU-12).

	Count	Time (abs/tel)	Tah	Th5	i Ir	21 Description	Arbid/Header	DataOytes	ChkSum	Thate	SyStatus	Network
Filter	1	1		1	1			and a second	100000			
E mo	FERZ	60.970	ns			LIN Master msg 1	01 (0xC1)	01 02 03 04 05 06 07 08	1A (Enh)	ūu5	Ok	LDN
0-	Detal					- t [t]				2-97.		
既	Dets2					- z [Z]						
-	Data3					- 3 [3]						
	Data4					- 4 [4]						
-	DataS					- S [5]						
0-	Dete6					- 6 [6]						
、既	Data7					- 7 [7]						
歌	DataB					= 8 [8]						
DAD -	1050	60.971	ns			LIN 03; No Slave Data	03 (0x03)		00 (Error)	DuS		LDN
the second	1175	61,056	ne:			LIN 02; No Slave Data	02 (0x42)		00 (Error)	0.45		1.04

Next have the monitoring or slave device add data to "LIN Master msg2" and "LIN Master msg3". Note the data can be any length from 1 to 8 bytes. It is now up to the LIN Master to process the data bytes sent back from the LIN Slave.

	Count	Time (abs/fel)	Tah	TX5	Er	2 Description	Arbid/Header	DataOytes	ChkSum	Trate	SyStatus	Network
Filter		1		-	1			to shake o	10000			
E 54.0	1233	60.970 m	6		ini -	LIN Master meg t	01 (0xC1)	01 02 03 04 05 06 07 08	1A (Enh)	Bus	OR SO	LD4
27	Data1					- t [1]						
BL-	Date2					= 2 [2]						
- At	Data3					- 3 [3]						
**	Data4					- 4 [4]						
79	Date5					= S [S]						
01-	Data6					= 6 [6]						
	Data7					- 7 [7]						
-	DataB					- 8 [8]						
白昌	1003	60.969 m		۵		LIN Master mag2	02 (0x42)	AA 88 CC DD	AC (E-h)	0 uS	Chidum	104
0+	meg2_Detail					= 170 [AA]						
**	meg2_Data2	91				- 187 [88]						
0+	msg2_Data3					- 204 [CC]						
	msg2_Data4	6				- 221 [DD]						
日星	103	51.056 m	8	۲		LIN Master msg3	02 (0×03)	CE FF	05 (brh)	0.05	ID Panty	LUN
-	meg3_Data1	-				= 238 [EE]						
-	msg3_Data2	15.				- 255 [FF]						

To see the LIN columns in the message view select LIN from the Columns button at the bottom of the messages view.

To test the LIN Master is receiving the LIN Slave data a simple script was written to control LED 4 and LED 5 based on receipt of the LIN Slave data. If the "msg2_Data1" = 170 then LED 4 is blue otherwise it is red. If the "mg3_Data1" = 238 the LED 5 is blue otherwise it is red.

Script	Start	Notes	LIN proces	ss slave data
+ A	fter	🕈 Before 🛛 🗕 🖁 🐰	🗈 🛍 🗠 🛐 No Errors	
	Step	Description	Value	Comment
	1	// TODO: Add step commands here		
	2		{LED 4 (Auto) :neo0-ld3-3-index(0)} = 0	// user control of LED 4
	3		{LED 5 (Auto) :neo0-ld4-3-index(0)} = 0	// user control of LED 5
	4	📑 If	{msg2_Data1 (Value) :in1-sig0-0} = 170	// is msg2_Data1 = 170?
	5	-•• Set Value	{LED 4 (Red) :neo0-ld3-4-index(0)} = 0	// yes, set LED 4 to blue
	6	-•• Set Value	$\{LED 4 (Green) : neo0-ld3-5-index(0)\} = 0$	// yes, set LED 4 to blue
	7	-•• Set Value	{LED 4 (Blue) :neo0-ld3-6-index(0)} = 255	// yes, set LED 4 to blue
	8	🔯 Else		
	9	Set Value	{LED 4 (Red) :neo0-ld3-4-index(0)} = 255	// no, set LED 4 to red
	10	Set Value	$\{LED 4 (Green) : neo0-ld3-5-index(0)\} = 0$	// no, set LED 4 to red
	11	-•● Set Value	{LED 4 (Blue) :neo0-ld3-6-index(0)} = 0	// no, set LED 4 to red
	12	📑 End If		
	13	📑 If	{msg3_Data1 (Value) :in2-sig0-0} = 238	// is msg3_Data1 = 238?
	14	Set Value	{LED 5 (Red) :neo0-ld4-4-index(0)} = 0	// yes, set LED 5 to blue
	15	-•● Set Value	$\{LED 5 (Green) : neo0-ld4-5-index(0)\} = 0$	// yes, set LED 5 to blue
	16	-•● Set Value	{LED 5 (Blue) :neo0-ld4-6-index(0)} = 255	// yes, set LED 5 to blue
	17	🔯 Else		
	18	-•● Set Value	{LED 5 (Red) :neo0-ld4-4-index(0)} = 255	// no, set LED 5 to red
	19		$\{LED 5 (Green) : neo0-ld4-5-index(0)\} = 0$	// no, set LED 5 to red
	20		{LED 5 (Blue) :neo0-ld4-6-index(0)} = 0	// no, set LED 5 to red
	21	📑 End If		
	22	🖄 Wait For	= 100 ms	// delay 100 ms
	23	🔟 Jump To	Step 4	// check data gain

Also note for this example the "LIN Master msg2" and "LIN Master msg3" were copied from the Transmit table and added to the Receive table. Then "msg2_Data1" thru "msg2_Data4" signals and "msg3_Data1" thru "msg3_Data2" signals were added to their respective messages. The signals needed to be defined so they could be processed in the script.

F F	unction Blocks 🔝 🕻	Application Sign	als 83 ava	Messages E	iditor 🖾 🔒	🗏 Tx Panel 🛛	8 🖲 Mes	19988	8	汗 ,)	letro	ka 🕅	El .		
Edit	aba Receive	🗏 Iranamit	C Reta	bose	on Netv	ork LIN				•	-	1 2	6	8	-1
Key	Description	Тур	e		ID CheckS	um Len		B1	82	83	84	85	86	87	68
	7		7	3	7		7	7	7	7	7	7	7	7	7
in1	LIN Master men2	Hea	der Only		02 Enhance	ed								_	
in2	LIN Master meg3	Hea	der Only		03 Enhance	ed									
4 6	als in Message				Equation	(Raw Value) (0, 1	,0,8		1.0	<u>A 50</u>	L.	Live	Edit		-
Soralsi	in Message	13	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5		syte 6		Byta	ž	Br	ta B	
Descript	tion	Type	*******	10100		238788X3	76543	3107	6 2 4	511	1 7 1 1	432	187	8 3 4 3	230
mag2_0	tatia i	Analog	1222	all search as	1.1.2										1916
mag2_D	vata2	Analog		246822											
mag2_D	Ester	Analog			103+3	2 2 2 3	Markel .								
nisg2_D	kata-t	Analog				1 1 5 + 3	1.1.0								

Receive table – "LIN Master msg2" and signals

Receive table - "LIN Master msg3" and signals

-	Function Blocks 🔯 🕯	Signals 53	🕐 🗠 Messages Editor 🔝 🗮 Ta Panel 😒 🗹 Messages 😂 🔆 Networks 😒														
Edil	aba Receive	📕 🗏 Irana	at E	Detabose		-0	n Networl	LI	. 1.			•	-	1 2	-	8	-1
Key	Description		Туре		-	ID	CheckSum	Len		B1	82	83	84	85	86	87	68
	7		7		7		7		7	7	7	7	7	7	7	7	7
in1	LIN Master meg2	397764453	Header Only	/		02	Enhanced				-		-			_	
in2	LIN Master meg3		Header Only	ć –		03	Enhanced										

Signals in Message									
♣ 8 ▼ = ♠ ➡				Equation	{Raw Value} 0,1,0,8		<i>f</i> ∗ <u>Edit</u>	Live Edit	t
Signals in Message		Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7	Byte 8
Description	Туре	7 6 5 4 3 2 1 0	76543210	76543	2 1 0 7 6 5 4 3 2 1 0	7 6 5 4 3 2 1 0	7 6 5 4 3 2 1 0	7 6 5 4 3 2 1 0	7 6 5 4 3 2 1 0
msg3_Data1	Analog	7 6 5 4 3 2 1 0							
msg3_Data2	Analog		7 6 5 4 3 2 1 0						

5.9 CAN to CAN-FD Gateway example

In this example, HS CAN 1 will be configured for standard CAN 2.0B. HS CAN 2 will be configured for HS CAN 2. The baud rate will be set to 500 Kbps and 2000 Kbps, respectively.

Create 3 HS CAN 1 Receive messages and their respective signals as shown below.

Edit	aba Receive	A Trans	at 8	Detabase	on Nets	work HS C	AN			•		3	Ga 1	8	-7
Key	Description		Type		Arb ID Mutti	Len /	-	Bt	82	83	84	85	86	87	68
100	7		7		7 7	1	7	7	7	7	7	7	7	7	7
			an an the	10		-		,						1.1	
ino -	HECANI - KX Messa	ge 1	CAN Std LLD	1	300 None										
113	HSCAN1 - RX Messa	ge 2	CAN Sto 110	1	301 None	/									
104	HOCANI - RX PRESS	geo	CAN SEE LID	u,	302 None										
Sign	als in Message														
	+ +				Equation	(Raw Value) (0	,1,0,8			le ta	L	Live	Edit		
Signals	n Nessage	AL.	Dyte 1	Byte 2	Dyte 3	Dyte 4	Dyte 5		Dyte 0		Dyte	7	Dy	teð	
Description	tion	Type	2228				******	2220	7004					ПТ	FT I
unt n	leteo, toe	Analog		4 4 4	Service and services										
con L m	Seteb_1ge	Analog		7 8 2		1.1.2									
unl n	ng L_clain 3	Arailog													
cen1_m	eg1_clate4	Analog					1 1 1 1 1								
cen1_m	ng1_clain5	Analog					- + 2	1128							
cani,n Cani,n	ngl_dwin5 agl_dala6 agl_dala7	Analog Analog Analog					- + 3			000		ana			
ani,n ani,n ani,n	egl_dete5 egl_deta5 egl_deta7 egl_deta7	Analog Analog Analog								660		202			mas
canijn canijn canijn canijn	ngi,dalas ngi,dalas ngi,dala7 ngi,dalai	Analog Analog Analog Analog							000		BCB	801	00 00		
can1_m can1_m can1_m can1_m Sign	ng1_data5 ng1_data6 ng1_data7 ng1_data7 ng1_data1 als in Message	Analog Analog Analog Analog						10008		1919 0	866	808	80		1000
can1_n can1_n can1_n can1_n Sign	ngl_denta5 agl_data6 agl_data7 sgl_data7 alls in Message s + = ♠ ♣	Analog Analog Analog Analog			Equation	(Furw Value) (0	, 1,0,8	0000		100 0 A 50			Edit		
sen1,m can1,m can1,m can1,m Sign Signals	ng1_data5 ng1_data6 ng1_data7 ng1_data7 ng1_data1 alls in Message	Analog Analog Analog Analog	Byta 1	Byta 2	Equation Byte 3	(Farw Value) (D Byte 4	1,0,8 8yte 5		Byte 6	/ 10	Lu	1000 Elwe 7	Edit By	tu 8	
sen1_m can1_m can1_m can1_m Sign Signals Descrip	ng1_dete5 ag1_data5 ag1_data7 sg1_data7 alls in Message a + - + +	Analog Analog Analog Analog Type	Byta 1	Byta 2	Equation Byte 3	(Faw Value) (D Byte 4	1,0,8 Byte 5		Byte 6	<u>A 50</u>	Byta 17 14 5	UNE 7	Edit By	tu 8	230
sen1_m can1_m can1_m can1_m Sign 9 c Signals Descrip	ng1_dete5 ag1_data5 ag1_data7 sg1_data7 alls in Message in Nessage in Nessage ton too	Analog Analog Analog Analog Type Analog	Byta 1	Byte 2	Equation Byte 3	(Faw Value) (D Byte 4	1,0,8 Byte 5 B 2 2 2 3 3 4	5310	Byte 6	A 10	Byta 17 1 5	000 01we 7	Edit By	tu B	
Sign Sign Sign Sign Sorals Descrip	ng1_dete5 ag1_data6 ag1_data7 sg1_data7 alls in Message s + - + + in Nessage ton reconstruct ag2_data2	Analog Analog Analog Analog Type Analog Analog Analog	Byta 1 2 4 5 4 1 DOEDE	Byte 2 12 2 5 2 5 5 12 10 10 12 10 10	Equation Byte 3 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2	(Raw Value) (D Byte 4	1,0,8 Byte 5 B 2 2 0 1 4 5 4	5310	Byte 6	A 50	Byta 17 t s	UME 7 14 3 2	Edit By 1 9 7 4	ta B	
Sign Sign Sign Signis Decolo Cantur Cantur	ng1_dete5 ag1_dete6 ag1_dete7 ag1_dete7 ag1_dete7 ag1_dete1 ag1_dete7 ag1_dete1 in Message ton module1 ag2_dete3	Analog Analog Analog Analog Analog Analog Analog Analog Analog	Byta 1 2 4 5 4 5	Byte 2 1 2 0 2 5 5 5 1 6 6	Equation Byte 3	(Raw Yalus) (D Byte 4	1,0,8 Byto 5	5310	Byte 6 7 6 9 9	A 22	8yta 1 7 4 5	UNE 7 [4]3]2	Edit By	ta 8	230
sentun cantun cantun cantun sign sign sign cantun cantun cantun	ng1_dete5 ag1_data6 ag1_data7 sg1_data7 alls in Message in Nessage ton tool_tata1 sg2_data2 sg2_data3 sg2_data4	Analog Analog Analog Analog Typo Analog Analog Analog Analog Analog	Byta 1 2 4 5 4 1 DEEDE	Byte 2 12 2 5 2 5 5 12 12 1 12 1 1 1 1	Equation Byte 3 C C C C C C C C C C C C C C C C C C C	(Raw Value) (D Byte 4 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2	1,0,8 Byte 5 1 1 1 1 1 1 1 1 1	10000	Byte 6	A 10	Byta z 7 e S	000 0.0we 7 14312	Edit By 3 9 7 4	ta: B (5) 4 3	230
sign Sign Sign Sign Sign Sign Sign Sign S	ng1_dete5 ng1_data6 ng1_data7 sq1_data7 sq1_data1 als in Message in Nessage ton ng2_data2 sq2_data3 sq2_data4 sq2_data4 sq2_data4	Analog Analog Analog Analog Analog Analog Analog Analog Analog	Byta 1	Byte 2 1 2 5 2 2 2 2 1 2 0 1 2	Equation Byte 3 C C C C C C C C C C C C C C C	(Faw Value) (D Byte 4	1,0,8 8)to 5 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1	15 2 1 0	Byte 6	A 10	8 yta 1 7 1 5	7 [4]3]3	Edit By 3 9 7 4	ta 8	230
Signi Signi Signi Signi Signi Signi Signi Signi Signi Signi Signi Signi	ng1_detes ng1_data5 ng1_data7 ng1_data7 ng1_data7 ng1_data7 ng1_data1 ng2_data1 ng2_data1 ng2_data2 ng2_data3 ng2_data3 ng2_data4 ng2 ng2_data4 ng2 ng2 ng2 ng2 ng2 ng2 ng2 ng2	Analog Analog Analog Analog Analog Analog Analog Analog Analog	Byta 1 Ze 2 a 2 DCCCC	Byte 2 1 2 6 2 6 1 1 0 0 0 1 0 0 1 0 0 1 0 0	Equation Byte 3 Equation Equation	(Fuzw Value) (D Byte 4	1,0,8 8,to 5 1110 7 6 5 4	· 5 2 [] 0	Byte 6	f~ Edi	Byta 17 († 5	Live	Edit Edit	tu 8 5 4 5	
ant, n cant, n cant, n cant, n Sign of Social cant, n cant, n	ng1_detes: ng1_detes: ng1_data? ng1_data? ng1_data? ng1_data? ng1_data? ng2_data?	Analog Analog Analog Analog Analog Analog Analog Analog Analog Analog	Byte 1	Byte 2	Equation Byte 3 Colored Equation Byte 3	(Raw Value) (D Byte 4 (Raw Value) (O Byte 4	, 1,0,8 8,10 5 7 5 5 1 1,0,8 8,10 5		8 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9	F Edi		Uwe 7 1 4 3 2	Edit Byj Edit Edit	ta 8	

can1_msg3_data2

7 6 5 4 3 2 1

7 6 5 4 3 2 1 7 6 5 4 3 2 1

Analog

Analog

Create 3 HS CAN 2 Transmit messages and their respective signals as shown below. Hint: go to the HS CAN 1 Receive table and copy the 3 messages. Then go to the HS CAN 2 Transmit table and paste the messages. Paste into the Description column. This will copy over the signals as well! Change the Type to "CAN FD Std 11 bit" for all 3 messages.

Fur	ation Blocks 🔯 🕻	Application S	ignals 83	e're Mes	sages Ed	itor 🕰	昌 TxPanel	23 (A M	estages	8	75 N	leteror	ka N	81		
Edit	ete Receive	🗏 Iranan	a. 81	🖹 Detabes		on Netv	work HS (CA112			•		1 26	6		*7
Key I	Neacription		Гуре		A/b	ID Multi	Len		B1	82	83	84	85	86	87	68
	7		5	7	7	7	7	7	7	7	7	7	7	7	7	7
outta +	ECAN1 - Rx Message	1 0	CAN FO Sto	111 bit	3	00 None	1		-		-			-	_	-
outi4 +	SCAN1 - Ry Messare	2	CAN ED Str	titbit -	-	Ci Mana										
butt5 H	SCANT - Rx Mensage	ã U	CAN FD Str	d 11 bit	3	02 None										
	-		the Approximation of the													
Signal	s in Message															
• 6	+ +					Equation	(Raw Value) (0	0, 1,0,8			fa Ed		- Uve	Edit		
Signals in I	Message	N.	Byte 1	Byt	#2	Byte 3	Byte 4	Byte 5		Byte 6		Byte	2	B	1.8	
Description		Тури	205						4 7 7 8 8			1 1 1				
cont_rup	Lowin	Analog	1 4 2	• z z s g	May No.	1.1	electron of the				1000	1014	111			-
can1_mag	1_claim2	Analog			7 4 7 2 3		1 1 1									
can1_mg	L_data3	Analog.				7 6 5 4	1111									
cani_mig	L_data4	Analog					14 2 4	3 2 3 3								
can1_mg	L_data5	Analog						1.5	4 3 3 8 9							
can1_mg	1_data6	Analog							122	751	222	6				
cari1_risp	L_dats7	Analog										7.6	- 5 2	11	i.	المحد
can1_nsg	L data8	Analog										10 10	1.4PP	111 7	631	121
Signals	in Message															
e 8	+ +					Cquetion	(Raw Value) (0	1,1,0,8			h 10	L	- Ure	Edt		_
Signals in P	lessage	1	Byte 1	Byt	e 2	Byte 3	Byte 4	Byte 5	i	Byte 6		Byte	7	. IBA	de B	
Description		Type		TTTTT I					13230	7 6 5 4	321	0763	432	1 8 7	634	2 2 1 1
cent reso	commit-	Analog	100	- alt da	COLUMN ST	21 10 Y 11 Y			deletetete	100	1000	-	1			
can1_msqu	data2	Analog .		1	5+11	a sister										
can1_msg2	(data3	Analog				1 1 2 1	113.0									
can1_msg2	clate4	Analog					1223	3230								
Signals	in Message															
4 8	+ +					Equation	{Raw Value} 0	,1,0,8			<i>f</i> ≈ <u>Edi</u>	<u>t</u>]	Live	Edit		
Signals in N	lessage		Byte 1	Byt	e 2	Byte 3	Byte 4	Byte 5		Byte 6		Byte	7	By	te 8	
Description	1	Туре	765	4 3 2 1 0 7 6	54321	07654	3 2 1 0 7 6 5 4	3 2 1 0 7 6 5	4 3 2 1 0	7654	321	0 7 6 5	4 3 2	107	654	321
can1_msg3	_data1	Analog	765	4 3 2 1 0												
can1 mso3	3 data2	Analog		7 6	5 4 3 2 1	0										

1	// TODO: Add step commands here	
2	📑 If	{HSCAN1 - Rx Message 1 (Present) :in0-0}
3		$\{HSCAN1 - Rx Message 1 (Present) : in0-0\} = 0$
4	-• Set Value	{can1_msg1_data1 (Value) :out13-sig3-0} = {can1_msg1_data1 (Value) :in0-sig3-0}
5	-• Set Value	{can1_msg1_data2 (Value) :out13-sig4-0} = {can1_msg1_data2 (Value) :in0-sig4-0}
6	Set Value	{can1_msg1_data3 (Value) :out13-sig5-0} = {can1_msg1_data3 (Value) :in0-sig5-0}
7	Set Value	$\label{eq:can1_msg1_data4} (Value) :out13-sig6-0 \label{eq:can1_msg1_data4} (Value) :in0-sig6-0 eq:can1_msg1_d$
8	Set Value	{can1_msg1_data5 (Value) :out13-sig7-0} = {can1_msg1_data5 (Value) :in0-sig7-0}
9	Set Value	$\label{eq:can1_msg1_data6} (Value) :out13-sig8-0 \end{tabular} = \end{tabular} \end{tabular} = \end{tabular} \end{tabular} \end{tabular}$
10	Set Value	$\label{eq:can1_msg1_data7} $$ $ $ $ $ $ $ $ $ $ $ $ $ $ $ $ $ $ $
11	Set Value	$\label{eq:can1_msg1_data8 (Value) :out13-sig10-0} = \label{eq:can1_msg1_data8 (Value) :in0-sig10-0}$
12	🗏 Transmit	HSCAN1 - Rx Message 1 (out13)
13	📑 End If	
14	🛃 If	{HSCAN1 - Rx Message 2 (Present) :in3-0}
15	Set Value	{HSCAN1 - Rx Message 2 (Present) :in3-0} = 0
16	-•● Set Value	$\label{eq:can1_msg2_data1} $$ $ $ $ $ $ $ $ $ $ $ $ $ $ $ $ $ $ $
17	-•● Set Value	$\label{eq:can1_msg2_data2} \label{eq:can1_msg2_data2} \label{eq:can2_msg2_data2} eq:can2_msg2$
18	-•● Set Value	{can1_msg2_data3 (Value) :out14-sig2-0} = {can1_msg2_data3 (Value) :in3-sig2-0}
19	-•● Set Value	{can1_msg2_data4 (Value) :out14-sig3-0} = {can1_msg2_data4 (Value) :in3-sig3-0}
20	Transmit	HSCAN1 - Rx Message 2 (out14)
21	📑 End If	
22	📑 If	{HSCAN1 - Rx Message 3 (Present) :in4-0}
23	0 Set Value	{HSCAN1 - Rx Message 3 (Present) :in4-0} = 0
24	0 Set Value	{can1_msg3_data1 (Value) :out15-sig0-0} = {can1_msg3_data1 (Value) :in4-sig0-0}
25	0 Set Value	$\label{eq:can1_msg3_data2} \label{eq:can1_msg3_data2} \label{eq:can2_msg3_data2} \label{eq:can3_msg3_data2} eq:can3_msg3$
26	🖳 Transmit	HSCAN1 - Rx Message 3 (out15)
27	📑 End If	

Next, create a script to copy the signals from the Receive messages into the Transmit messages.

The following screenshot shows the comments for each script line item ...

// is HSCAN1 - Rx Message 1 present?
// yes, dear present flag
// copy byte 1 from HSCAN1 rx message to HSCAN2 tx message
// copy byte 2 from HSCAN1 rx message to HSCAN2 tx message
// copy byte 3 from HSCAN1 rx message to HSCAN2 tx message
// copy byte 4 from HSCAN1 rx message to HSCAN2 tx message
// copy byte 5 from HSCAN1 rx message to HSCAN2 tx message
// copy byte 6 from HSCAN1 rx message to HSCAN2 tx message
// copy byte 7 from HSCAN1 rx message to HSCAN2 tx message
// copy byte 8 from HSCAN1 rx message to HSCAN2 tx message
// transmit HSCAN2 message 1
// is HSCAN1 - Rx Message 2 present?
// yes, dear present flag
// copy byte 1 from HSCAN1 rx message to HSCAN2 tx message
// copy byte 2 from HSCAN1 rx message to HSCAN2 tx message
// copy byte 3 from HSCAN1 rx message to HSCAN2 tx message
// copy byte 4 from HSCAN1 rx message to HACAN2 tx message
// transmit HSCAN2 message 2
// is HSCAN1 - Rx Message 3 present?
// yes, dear present flag
// copy byte 1 from HSCAN1 rx message to HSCAN2 tx message
// copy byte 2 from HSCAN1 rx message to HSCAN2 tx message
// transmit HSCAN2 message 3

When the script is complete program it into the neoECU-12.

When creating the script make sure the Tx Message is selected followed by HS CAN2 followed by the 1st data byte in message 1. Next, make sure the Rx Messages is selected followed by HS CAN followed by the 1st data byte in message 1 then click "Add to Expression" button. Repeat for all signals in all 3 messages.

🔤 Enter Expre	ession			
Enter Expre	ssion for signal			
Value To Set	{can1_msg1_data1	(Value) :out13-sig3-0}		
Expression				
Clear		1		
f* Express	ion Builder			
•••Rx Mes	ssages	Sort By: Networks	▼ Pick Set Value	Add To Expression
Databa	ise	Find		Clear
	sages	🖂 🔩 CAN		
	Groups	표 하는 HS CAN		
		□ □ HS CAN2	Mechane 1 (out13) [300]	
	anals	Can1_msg	1_data1	
Retwo	rks	Can1_msg	1_data2	
°∿∎II		Can 1_msg	1_data3	
🔤 Enter Expre	ssion			
Enter Expre	ssion for signal			
Weber To Get	C 1 1 1 1 1	((-),-),,,,,,,,,,,,-		
Value To Set	{can1_msg1_data1	(value) :InU-sig3-0}		
Expression	{can1_msg1_data1	L (Value) :in0-sig3-0}		
Clear				
f* Express	ion Builder			
Rx Mes	sages	Sort By: Networks	Pick Set Value	Add To Expression
Databa	ise	Find		Add TO Expression
BTx Mes	sages	rina	/	Clear
Signal	Groups			<u>^</u>
₽DAQ			Message 1 (10) [300]	
Jobs		can1_msg	1_data1	
🖀 App Si	gnals	can1_msg1	1_data2	
l	rks		1_00(0)	

Now use another Intrepid tool that is CAN/CAN-FD capable to generate CAN traffic on HS CAN 1. Transmit 3 CAN messages with ArbID 300, 301, and 302. The 1st message should contain 8 data bytes, the 2nd message should contain 4 data bytes, and the 3rd message should contain 2 data bytes.

are h	Aessages Editor 🌇 🕮 (Nessages 🔛 🖗	5 Networks 😒	昌 1:	Panel		Function Black	8 8 1							
Edit	ete Receive	🗏 Iranami	Database	1	n Netwo	nk.	HS CAN		-	•	-	3	9	8	-1
Key.	Description	Гуре		Arb ID	Multi	Le	n.	B1	82	83	84	85	86	87	68
1	7		7	7	7		7	7	7	7	7	7	7	7	7
out2	HSCAN1 - Tx Message 1	CAN Std	Ltbit	300	None			01	02	03	04	05	06	07	CB.
out3	HSCAN1 - Tx Message 2	CAN Std	11 bit	301	None			AA	68	CC	DD.				
out4	HSCANT - Tx Message 3	CAN Str	111 bit	302	None			臣	FF						

In the Tx Panel set the rate to be 100 ms and periodic. This is a test only. The ECU will have it's own rates for each message or you can duplicate the rates here.

104 Messages Editor (23) (9: Messages (23) (7: Networks (23) 📥 TK Pana 🛃 Function Blocks (23)															
📇 Edit Transmit Nessages 🛛 🗙 Disable A		Tx	Protocol:	Al 🔻											
Description	Auto Tx	Rate (s)	Arb ID	Len	B1	82	B3	B4	85	66	87	88	Network	Color	
7		3	7	7	\mathbf{Y}	7	7	3	\mathbf{Z}	7	7.	7	3	7	
HSCAN1 - Tx Message 1		Periodic	0.100000	300		01	02	03	04	05	06	07	08	HS CAN	1
HSCAN1 - Tx Message 2	10	Periodic	0.100000	301		AA	88	CC	DD					HS CAN	
HSCAN1 - Tx Message 3		Periodic	0.100000	302		EE	FF							H5 CAN	

Next transmit these messages and monitor both HS CAN 1 and HS CAN 2. Each time the message was received on HS CAN 1 it was forwarded to HS CAN 2. In this case, the same ArbID and all of the data bytes were forwarded as is and not changed or scaled. Note the FDF bit (CAN-FD) bit is set and the BRS bit (Bit Rate Switch) is set. If you were to put a scope on the HS CAN 2 line you would see the data bytes are transmitted at the higher bit rate. To see the CAN-FD columns in the message view select CAN FD from the Columns button at the bottom of the messages view.

	Count	Tone (abs/rel)	īκ	6	2 Description	Arbid/Header	Len	DetaBytes	Network	Node	ac	FDF	BRS	ESI
Filter			1		1200000000000		1000	020203000						
200	144	302,986 ms	1		HS CAN2 \$300	300	8	010203040505050708	H5 CAN2		8	t	1	0
	144	302.994 mg			HS.CAN2 \$301	301	4	AA 88 CC 00	HS CAND		4	ı	1	2
7	144	102,985 ins			HS CAN2 \$302	302	2	EE FF	HS CANZ		2	1	1	0
昌	144	202.930 mp	۲		HSCAN1 - Tx Memage 1	300	a	01 02 03 04 05 06 07 08	HS CAN		5	_		
8	144	102.931 ms	۲		HSCAN1 - Tx Meisage 2	301	+	AA 68 CC DO	HS CAN		4			
2	144	302.931 mt	-		HSCAN1 - Tx Message 3	302	2	EFF	H5 CAN		2			

Check out our new Gateway Builder in Vehicle Spy Enterprise version. Drag and Drop GUI makes it super easy to build a custom Gateway! It generates the scripts for you. (3)

6.0 Support Contact Information

6.1 ICS United States Headquarters and West Coast office

Intrepid Control Systems, Inc. 31601 Research Park Drive Madison Heights, MI 48071

Phone: (800) 859-6265 or (586) 731-7950 Fax: (586) 731-2274 Email: <u>Email ICS</u>

Intrepid Control Systems, Inc. 1925 Winchester Blvd, Ste 200 Campbell, CA 95008 USA

Phone: 1-800-859-6265 or (586) 731-7950 Fax: 586-731-2274 Email: <u>Email Sales</u>

6.3 ICS International Offices

European Union

Haid-und-Neu-Straße 7 76131 Karlsruhe Germany Phone: +49 721 1803083-0 Fax: +49 721 1803083 -9 Email: <u>Email Sales</u>

United Kingdom

Serviced Office Suite 1.03 MIRA Technology Park Technology Centre NW05 Watling Street Nuneaton Warwickshire CV10 0TU Phone: +44 24 7718 0296 Email: <u>Email Sales</u>

China (Beijing)

https://www.intrepidcs.net.cn

Room 214, No.45 Chengfu Road, Zhongguancun Zhizao Street Block C, Haidian District, Beijing P.R.China 100083 Phone: +86 176 8014 3205 Email: <u>Email Sales</u>

China (Shanghai)

https://www.intrepidcs.net.cn

Room 902, Building 16 No. 1000 Jinhai Road City of Elite Pudong, Shanghai P.R.China 201206 Phone: 021-61637366 Fax: 021-61637366 * 600 Email: <u>Email Sales</u>

China (Shenzhen)

https://www.intrepidcs.net.cn

Room 22-YZ, Block A, Che Kung Temple Fortune Plaza No.5 Xianglin Road Shenzhen, 518040 CHINA Phone: +86 0755 82723212 Email: <u>Email Sales</u>

India

Office 306, B Building, Third Floor, GO Square, Wakad, Pune, MH, 411057 India Phone: Sales: +91 77 55 99 00 70 Tech Support: +91 77 55 99 00 64 All Others: +91 77 55 99 00 74 Email: <u>Email Sales</u>

South Korea

http://www.intrepidcs.co.kr/

#1310, Keurancheu Techno B/D, 388, Dunchon-daero, Jungwon-gu, Seongnam-si, Gyeonggi-do, 13403, South Korea Phone/전화: +82 31 698 3460

Fax/팩스: +82 31 698 3461 Email: Sales

Japan

http://www.intrepidcs.jp

164-0003 Shinryou Building 3F Higashinakano 1-59-6 Nakano-Ku: Tokyo 電話番号: +81 03-5937-1523 ファックス: +81 03-5937-2524 営業 / 一般的なお問い合わせ: <u>icsjapan@intrepidcs.com</u> 技術サポート: <u>icsjapansupport@intrepidcs.com</u>

Australia

67A Hardiman St Kensington, VIC 3031 Australia Phone: 03 9466 4948 (International callers: +61 3 9466 4948) Email: <u>Julian Merritt</u>