Logger - How Vector loggers can be used with J1939

Version 1.2
2013-06-10
Application Note AN-IND-1-013

Author	Jochen Neuffer, Dagmar Martinek
Restrictions	Public Document

Abstract	This application note describes how the Vector logger family can be used to handle J1939 protocol specifics.

Table of Contents

1.0 Overview

The Vector/G.i.N. logger family was designed as general purpose loggers. However, it is also possible to handle J1939 protocol specifics very easily. Although there is no built-in support of the J1939 protocol, it is possible to address the most common use cases with the configuration language LTL (Log Task Language).

1.1 Use Cases

- The most common use case is to record network data without actively participating in the network. This means the logger does not actually simulate a network node i.e. it does not send J1939 messages. The number of channels which can be logged can vary dependent on the used logger type.
- Another common use case is to provide a gateway between J1939 and standard CAN.

This document will give hints and tips to achieve both uses cases with the Vector logger family.

1.2 GL1000/GL2000/GL3000/GL4000 Families, CANlog3/CANlog4, Multilog

Transceivers:	High-speed transceivers are available
Sleep/wake-up:	With logger piggyback 1041 also for J1939 networks available
Bus parameter:	Baud rate adjustable
Database:	Import DBC
Filter:	Filter on extended ID, also filter range on extended ID, data reduction filter e.g. a certain message only every n milliseconds which can be useful on short messages.

Trigger on signals:	Only possible for messages with a maximum of 8 data bytes. Messages which exceed 8 data bytes cannot be interpreted directly.
Trigger on SPN:	See LTL code example below
Trigger on PGN:	See LTL code example below
Network management:	This is not supported since the logger is no active node in the network.
Address claiming:	This is not supported since the logger is no active node in the network. IDs have to be static.
Gateway:	Bidirectional gateway J1939 / standard CAN
Classification:	Classification functions such as Count, Flipcount, Time, Min/Max, Rainflow are available for messages with a maximum of 8 data bytes and messages which support multiplex signals.

Note for GL1000 family: Classification can be emulated partly with LTL.

Permanent logging:	Possible with/without overwrite of old data, start/stop/resume of the logger is also supported.
Triggered logging:	Via ring buffer
Format of log files:	ASC, BLF, MDF (for short messages ≤ 8 data bytes), TXT, CLF

LTL code examples are available on request. Customer-specific applications can also be done as a project work.

2.0 CANcaseXL log

Transceiver:	High-speed transceivers are available (CANpiggies)
Sleep/wake-up:	not supported
Bus parameter:	Baud rate adjustable
Database:	Import DBC
Filter:	Filter on extended ID, also filter range on extended ID, data reduction filter e.g. a certain message only every one out of n messages which can be useful on short messages.
Trigger on signals:	Only possible for messages with a maximum of 8 data bytes. Messages which exceed 8 data bytes cannot be interpreted directly.
Trigger on SPN:	Not supported
Trigger on PGN:	Not supported
Network management:	This is not supported since the logger is no active node in the network.
Address claiming:	This is not supported since the logger is no active node in the network. IDs have to be static.
Gateway:	Not supported
Classification:	Not supported
Format of log files:	BLF, MDF, TXT, XLX

3.0 Analysis

3.1 CANoe / CANalyzer

For analysis of logging files CANoe / CANalyzer with the option .J1939 can be used. All well-known analysis features like trace, graphics, data window, etc. are available. For files which contain permanent loggings, triggers and filters may be used. CANoe and CANalyzer are able to import ASC, BLF and CLF files.

3.2 CANape / CANgraph

CANape / CANgraph are able to import BLF, ASC, MDF, CLF and XLX. Also transport protocols BAM and NMEA2000 Fast Package are supported. With the built-in scripting language and data mining it is possible to automatically evaluate logging files.

4.0 LTL Sample Configuration

These LTL samples can be used for all GL loggers, CANlog and Multilog.

4.1 Trigger on SPN

Notes:	The following example is based on the CANoe.J1939 “DTC Monitor” demo . CAN1 of CANoe and CAN1 of GL1000 must be connected.

```
{ ---------------------------------------------------
  GL1000 Sample Configuration J1939 - Trigger on SPN
  (c) 2009-2013 Copyright Vector Informatik GmbH
  Version 1.0, 2013-06-10
  Reference: CANoe.J1939 Sample "DTC Monitor"
--------------------------------------------------- }
SYSTEM
  Can1Timing          = Timing250k  { baud rate 250 kbaud }
  Can1Output          = On
  CAN1KeepAwake       = On
  Logger1Size         = 5000
  SleepSeconds        = 5           { sleep mode after 5 seconds bus sleep }
  StandardDelay       = 2000
  Pause               = 0


CONST
  TriggerSPN_HighByte = 0x6E
  TriggerSPN_LowWord  = 0x000B

VAR
  Current_SPN_HighByte = FREE[8]
  Current_SPN_LowWord  = FREE[16]

  TriggerBit = FREE[1]
  BAM_Valid  = FREE[1]
```

```
EVENT
{ -------- DM_1 message: one DTC only ------- }
  ON RECEIVE (CAN1 XDATA 0x18FECA00 - 0x18FECAFF) BEGIN
    VAR
         Valid  = FREE[1]
         Current_SPN_HighByte = THIS [2(7:0)]
         Current_SPN_LowWord  = THIS [3(7:0) 4(7:0)]
    CALC
         Valid = (this.6 = 0xFF AND this.7 = 0xFF)
         TriggerBit = (Current_SPN_HighByte = TriggerSPN_HighByte AND
                       TriggerSPN_LowWord = Current_SPN_LowWord)
                       WHEN (Valid)
  END

{ -------- Sample for up to 5 DTCs  ------------- }
{ -- First message: BAM message ----------------- }
  ON RECEIVE (CAN1 XDATA 0x18ECFF00 - 0x18ECFFFF) BEGIN
    CALC
         { check whether received message is valid }
         BAM_Valid = (this.5 = 0xCA AND this.6 = 0xFE AND
                      this.3 > 1 AND this.3 < 4)
  END

{ -- Next messages: P messages ------------------ }
  ON RECEIVE (CAN1 XDATA 0x18EBFF00 - 0x18EBFFFF) BEGIN
    VAR
          TriggerBit1 = FREE[1]  TriggerBit2 = FREE[1]
          PackageNo = THIS[0]                { Package number 1..3 for up to 5 DTCs}
          Valid  = FREE[1]

          SPN_0_H = THIS[3(7:0)]             { SPN #0,  8 Bit (high byte) in package 1 }
          SPN_0_L = THIS[4(7:0) 5(7:0)]      { SPN #0, 16 Bit  (low word) in package 1 }

          SPN_1_H = THIS[7(7:0)]             { SPN #1,  8 Bit (high byte) in package 1 }
          SPN_1_L = THIS[1(7:0) 2(7:0)]      { SPN #1, 16 Bit  (low word) in package 2 }

          SPN_2_H = THIS[4(7:0)]             { SPN #2,  8 Bit (high byte) in package 2 }
          SPN_2_L = THIS[5(7:0) 6(7:0)]      { SPN #2, 16 Bit  (low word) in package 2 }

          SPN_3_H = THIS[1(7:0)]             { SPN #3,  8 Bit (high byte) in package 3 }
          SPN_3_L = THIS[2(7:0) 3(7:0)]      { SPN #3, 16 Bit  (low word) in package 3 }

          SPN_4_H = THIS[5(7:0)]             { SPN #4,  8 Bit (high byte) in package 3 }
          SPN_4_L = THIS[6(7:0) 7(7:0)]      { SPN #4, 16 Bit  (low word) in package 3 }
```

```
  CALC
      Valid = (PackageNo >= 1 AND PackageNo <= 3)  { package valid }

      Current_SPN_HighByte = SPN_0_H WHEN (BAM_Valid AND Valid AND PackageNo = 1)
      Current_SPN_LowWord  = SPN_0_L WHEN (BAM_Valid AND Valid AND PackageNo = 1)
      TriggerBit  = (Current_SPN_HighByte = TriggerSPN_HighByte AND
                     TriggerSPN_LowWord = Current_SPN_LowWord)
                     WHEN (BAM_Valid AND Valid AND PackageNo = 1)

      Current_SPN_HighByte = SPN_1_H WHEN (BAM_Valid AND Valid AND PackageNo = 1)
      Current_SPN_LowWord  = SPN_1_L WHEN (BAM_Valid AND Valid AND PackageNo = 2)
      TriggerBit1 = (Current_SPN_HighByte = TriggerSPN_HighByte AND
                     TriggerSPN_LowWord = Current_SPN_LowWord)
                     WHEN (BAM_Valid AND Valid AND PackageNo = 2)

      Current_SPN_HighByte = SPN_2_H WHEN (BAM_Valid AND Valid AND PackageNo = 2)
      Current_SPN_LowWord  = SPN_2_L WHEN (BAM_Valid AND Valid AND PackageNo = 2)
      TriggerBit2 = (Current_SPN_HighByte = TriggerSPN_HighByte AND
                     TriggerSPN_LowWord = Current_SPN_LowWord)
                     WHEN (BAM_Valid AND Valid AND PackageNo = 2)
      TriggerBit = (TriggerBit1) OR (TriggerBit2)
                     WHEN (BAM_Valid AND Valid AND PackageNo = 2)

      Current_SPN_HighByte = SPN_3_H WHEN (BAM_Valid AND Valid AND PackageNo = 3)
      Current_SPN_LowWord  = SPN_3_L WHEN (BAM_Valid AND Valid AND PackageNo = 3)
      TriggerBit1 = (Current_SPN_HighByte = TriggerSPN_HighByte AND
                     TriggerSPN_LowWord = Current_SPN_LowWord)
                     WHEN (BAM_Valid AND Valid AND PackageNo = 3)

      Current_SPN_HighByte = SPN_4_H WHEN (BAM_Valid AND Valid AND PackageNo = 3)
      Current_SPN_LowWord  = SPN_4_L WHEN (BAM_Valid AND Valid AND PackageNo = 3)
      TriggerBit2 = (Current_SPN_HighByte = TriggerSPN_HighByte AND
                     TriggerSPN_LowWord = Current_SPN_LowWord)
                     WHEN (BAM_Valid AND Valid AND PackageNo = 3)
      TriggerBit = (TriggerBit1) OR (TriggerBit2)
                     WHEN (BAM_Valid AND Valid AND PackageNo = 3)
  END

{--- Stop/Start Ring buffer ---}
STOP  1 (TriggerBit)
START 1 (NOT TriggerBit)
```

```
{--- LED Display ---}
OUTPUT
  LED1 = (_10msec < 50)          { Flashing LED for operating mode }
  LED2 = (TriggerBit)            { LED for trigger occurrence }
END
{ Program end }
```

4.2 Trigger on PGN

Notes:	The following example is based on the CANoe.J1939 “System Demo”. The PGN 0xFDD1 is triggered on channel 2. Therefore for logging CAN2 of CANoe and CAN2 of GL1000 must be connected. LTL only supports a maximum of 16 bit variables. Therefore to keep the example simple only the lower 16 bits are evaluated. This will be sufficient in most cases e.g. in the CANoe.J1939 System Demo all PGN are ≤ 16bit. However, it is also possible to evaluate 18 bit values with LTL ‘AND’ operations.

```
{ ------------------------------------------------
  GL1000 Sample Configuration J1939 - Trigger on PGN
  (c) 2009-2013 Copyright Vector Informatik GmbH
  Version 1.0, 2013-06-10
  Reference: CANoe.J1939 Sample "System Demo"
------------------------------------------------ }

SYSTEM
  CAN1Timing          = Timing250K    { baud rate 250 kbaud }
  CAN1Output          = On
  CAN1KeepAwake       = On
  CAN2Timing          = Timing250K    { baud rate 250 kbaud }
  CAN2Output          = On
  CAN2KeepAwake       = On
  Logger1Size         = 5000
  SleepSeconds        = 10            { sleep mode after 10 seconds bus sleep }
  StandardDelay       = 2000
  Pause               = 0

VAR PGN      = FREE[16]
VAR Trigger  = FREE[1]
```

```
{--- On receive of any message check its PGN ---}
EVENT ON RECEIVE (CAN2 XDATA 0 - 0x1FFFFFFF) BEGIN
  CALC PGN = ((LOWBYTE(This.IDhi) * 0x100) + HIGHBYTE (This.ID))
       Trigger = (1) WHEN (PGN =  0xFDD1)
END

{--- After a ring buffer was triggered reset the trigger flag ---}
EVENT ON SET (Stopped1) BEGIN
  CALC Trigger = (0)
END

{--- Stop/Start Ring buffer ---}
STOP  1 (Trigger)       { trigger when PGN FDD1 was received }
START 1 (NOT Trigger)   { start new recording when the trigger flag was reset }

{--- LED Display ---}
VAR TriggerLED = FREE[1]
EVENT ON SET (Trigger) BEGIN
  CALC TriggerLED = (1)
END
TIMER tTriggerLED TIME = 1000 (TriggerLED)
EVENT ON SET (tTriggerLED) BEGIN
  CALC TriggerLED = (0)
END

OUTPUT
  LED1 = (_10msec < 50)  { Flashing LED for operating mode }
  LED2 = (TriggerLED)    { LED for trigger event }
  LED3 = (0)
  LED4 = (FlashFull)     { SD card is full, oldest records are overwritten }
END
{ Program end }
```

4.3 Filter

Notes:	The following example is based on the CANoe.J1939 “System Demo”. CAN1 of CANoe and CAN1 of GL1000 must be connected.

```
{ -------------------------------------------------------------------
  GL1000 Sample Configuration J1939 - Long-term Logging with Filter
  (c) 2009-2013 Copyright Vector Informatik GmbH
  Version 1.0, 2013-06-10
  Reference: CANoe.J1939 Sample "System Demo"
------------------------------------------------------------------- }
SYSTEM
  Can1Timing          = Timing250k  { baud rate 250 kbaud, powertrain }
  Can1Output          = On
  CAN1KeepAwake       = On
  Logger1Size         = 5000
  SleepSeconds        = 5           { sleep mode after 5 seconds bus sleep }
  StandardDelay       = 0
  Pause               = 0

{--- Filter conditions ---}
RECORDFILTER
  EXCLUDE CAN1 XDATA 0 - 0x1FFFFFFF             { stop filter for all extended IDs on CAN1 }
  INCLUDE CAN1 XDATA 0x18FECA00 - 0x18FECAFF    { pass filter for DM1 messages }
  INCLUDE CAN1 XDATA 0x0C000000 - 0x0C00FFFF    { pass filter for TSC messages }

{--- Long-term logging ---}
VAR ShutdownTrigger = FREE[1]
EVENT ON SYSTEM (Shutdown) BEGIN
  CALC ShutdownTrigger = (1)
END
STOP  1 (NOT LoggerAtEnd1 DELAY = 0) OR (ShutdownTrigger DELAY = 10)
START 1 (NOT ShutdownTrigger)

OUTPUT
  LED1 = (_10msec < 50)  { Flashing LED for operating mode }
  LED2 = (0)
  LED3 = (0)
  LED4 = (FlashFull)     { SD card is full, oldest records are overwritten }
END
{ Program end }
```

5.0 Contact

Germany
and all countries not named below:

Vector Informatik GmbH
Ingersheimer Str. 24
70499 Stuttgart
GERMANY
Phone: +49 711-80670-0
Fax: +49 711-80670-111
E-mail: info@de.vector.com

France, Belgium, Luxemburg:

Vector France S.A.S.
168, Boulevard Camélinat
92240 Malakoff
FRANCE
Phone: +33 1 42 31 40 00
Fax: +33 1 42 31 40 09
E-mail: information@fr.vector.com

Sweden, Denmark, Norway, Finland, Iceland:

VecScan AB
Theres Svenssons Gata 9
41755 Göteborg
SWEDEN
Phone: +46 31 764 76 00
Fax: +46 31 764 76 19
E-mail: info@se.vector.com

United Kingdom, Ireland:

Vector GB Ltd.
Rhodium, Central Boulevard
Blythe Valley Park
Solihull, Birmingham
West Midlands B90 8AS
UNITED KINGDOM
Phone: +44 121 50681-50
Fax: +44 121 50681-69

E-mail: info@uk.vector.com

China:

Vector Automotive Technology (Shanghai) Co., Ltd.
Sunyoung Center
Room 1701, No.398 Jiangsu Road
Changning District
Shanghai 200050
P.R. CHINA
Phone: +86 21 6432 53530
Fax: +86 21 6432 5308
E-mail: info@cn.vector.com

India:

Vector Informatik India Pvt. Ltd.
4/1/1/1, Sutar Icon, Sus Road,
Pashan, Pune - 411 021
INDIA

Phone: +91 20 2587 2023
Fax: +91 20 2587 2025

E-mail: info@in.vector.com

USA, Canada, Mexico:

Vector CANtech, Inc.
39500 Orchard Hill Place, Suite 550
Novi, MI 48375
USA

Phone: +1 248 449 9290
Fax: +1 248 449 9704
E-mail: info@us.vector.com

Japan:

Vector Japan Co. Ltd.
Tennozu Yusen Bldg. 16F
2-2-20 Higashi-shinagawa,
Shinagawa-ku,
Tokyo 140-0002
JAPAN
Phone: +81 3 5769 7800
Fax: +81 3 5769 6975
E-mail: info@jp.vector.com

Korea:

Vector Korea IT Inc.
5F, Gomoas bldg.
12 Hannam-daero 11-gil, Yongsan-gu
Seoul, 140-889
REPUBLIC OF KOREA

Phone: +82 2 807 0600
Fax: +82 2 807 0601
E-mail: info@kr.vector.com