DBC and LDF Configuration

Technical Reference

for FGA

Version 1.7.2

Authors	Hannes Haas, Milena Shakir
Status	Released

Document Information

History

Author	Date	Version	Remarks
Hannes Haas, Klaus Emmert	2010-09-27	1.0.0	Created
Hannes Haas	2010-09-28	1.0.1	Added update bit for signal groups
Hannes Haas	2010-10-13	1.0.2	GenMsgNrOfRepetition description changed Removed restrictions of Tx Mode combination Added description to message routing
Hannes Haas	2010-10-15	1.0.3	Changed spec. for GenSigTimeoutTime
Hannes Haas	2010-11-08	1.0.4	GenMsgCycleTimeFast for Tx modes with repetitions
Hannes Haas	2010-12-17	1.1.0	Added LDF 2.1 rules Renamed NmNodeType to NwmNodeType
Hannes Haas	2011-01-03	1.1.1.	Improved Tx mode mappings by adding GenMsgCycleTimeFast to the examples Fixed issues in Send Type examples
Hannes Haas	2011-03-07	1.1.2	CONTROL Tx mode only supported for carry-back ECUs
Hannes Haas	2011-04-07	1.1.3	Release
Hannes Haas	2011-05-03	1.1.4	Changed definition of GenMsgNrOfRepetition to make it compliant with CANbedded Added TpOwnSystemEcuNumber for CANbedded projects Clarified handling of CONTROL signals
Hannes Haas	2011-07-27	1.2.0	Added Tx modes Dual Cycle with Trigger
Hannes Haas	2012-02-10	1.3.0	Added additional CAN baud rate attributes
Milena Shakir	2012-11-14	1.4.0	Added attributes for Class C Network Management With Wakeup
Milena Shakir	2013-02-28	1.5.0	Updated description for NwmNodeType and added chapter for diagnostic routing
Milena Shakir	2013-03-20	1.6.0	Updated chapter 3.2 regarding transmission mode control
Milena Shakir	2013-03-25	1.6.1	Minor adaptions after review (improvements)
Hannes Haas	2013-06-20	1.7.0	Clarified ClassC Nm attribute usage for networks without wakeup Added rules for RoE diagnostics
Hannes Haas	2013-07-01	1.7.1	Minor clarifications and examples

Hannes Haas	2013-09-11	1.7.2	Fixed RoE Example in chapter 3.7.1.1

Reference Documents

No.	Source	Title	Version
[1]	Vector	TechnicalReference_Asr_CanNm.pdf	
[2]	Vector	TechnicalReference_Asr_Com.pdf	
[3]	Vector	TechnicalReference_Asr_CanTp.pdf	
[4]	Vector	TechnicalReference_Asr_Dcm_Fiat.pdf	
[5]	LIN	LIN Standard	2.1
[6]	Vector	Application Note AN-AND-1-106: Basic CAN Bit Timing Available from the vector.com download center	2.0

Contents

Tables

1 Overview

This application note describes the CAN data base (DBC) attributes for the FGA AUTOSAR Stack. In the following chapter an introduction to the CAN data base and the CAN data base attributes is provided. In a further chapter the relevant CAN data base attributes for FGA are described in detail.

2 Introduction

The CAN data base (DBC for short) describes the CAN communication between all ECUs in one network. For the description of multiple CAN networks in a vehicle a separate DBC file for each CAN network is necessary.

Additionally the DBC format supports the possibility to describe any additional information via attributes, e.g. CAN baud rate.

The described CAN communication and the attributes within a DBC file are used to configure the embedded communication stack via the configuration tool GENy. GENy also generates the dynamic code part of this stack. Other Vector tools also use the DBC file as input format, e.g. the network simulation and analyzing tool CANoe.

This document focuses on the necessary attributes and their values for correct configuration and code generation via GENy.

2.1 Data Base Attributes

Data base attributes contain additional configuration-relevant information, e.g. the baud rate of a CAN network. As such information has always a specific context (e.g. baud rate is network specific) at creation time of an attribute the context (object type) has to be defined. The following list contains the relevant object types.

- **Network**
  Attribute is network-specific and has to be configured only once in the DBC (as a DBC contains only a single network)
- **Node**
  Attribute is node-specific and has to be set for each node in the network
- **Message**
  Attribute is message-specific and has to be set for each message in the network
- **Signal**
  Attribute is signal-specific and has to be set for each signal in the network

Info
The default value of a data base attribute is automatically set to all relevant objects at creation time of the attribute.

There are four relevant different value types for data base attributes that are described in the following list:

- **Enumeration (Enum)**
  Attribute can adopt a limited list of non-numeric values, e.g. No / Yes.

Caution
The names and the sorting order must be considered as defined in this document for each attribute (e.g. No = 0, Yes = 1).

- **Hex**
  Numeric attribute in a hexadecimal format
- **Integer**
  Numeric attribute in an integer format
- **String**
  Attribute can be any string

3 DBC Attributes and Conventions

This chapter describes the definition and usage of the attributes used for the FGA AUTOSAR stack. Please also refer to the attribute documentation in the component-specific technical references.

The described attribute rules are backward compatible with the existing CANbedded solution.

Info
This document just gives an overview of attributes for a data base for FGA. For more detailed description on the attributes, please also refer to the attribute documentation in the component specific technical references.

3.1 General Attributes

Attribute Name	Object Type	Type	Values and Ranges (**Bold =** default)	Description
Baudrate	Network	Integer	0..**500000**..1000000	CAN Baud rate of the network
SamplePointMin	Network	Integer	50..81..100	For details please refer to [6]: “Sample Point”. This parameter is optional.
SamplePointMax	Network	Integer	50..90..100	For details please refer to [6]: “Sample Point”. This parameter is optional.
SyncJumpWidthMin	Network	Integer	1..4	For details please refer to [6]: “Resynchronization Jump Width”. This parameter is optional.
SyncJumpWidthMax	Network	Integer	1..4	For details please refer to [6]: “Resynchronization Jump Width”. This parameter is optional.
NBTMin	Network	Integer	8..6..25	For details please refer to [6]: “Number of Time Quanta”. This parameter is optional.
NBTMax	Network	Integer	8..21..25	For details please refer to [6]: “Number of Time Quanta”. This parameter is optional.
Manufacturer	Network	String	**Fiat**	Indicates OEM FGA. Value must be „Fiat“.
DBName	Network	String	**CAN**	Specifies the name of the network. Must be different for any CAN (and LIN) network within one ECU.

Table 3-1 General Attributes

3.2 Attributes for COM Module

The following table describes the relevant attributes for the FGA Autosar COM.

Attribute Name	Object Type	Type	Values and Ranges (**Bold** = default)	Description
GenMsgILSupport	Message	Enum	**No** = 0 Yes = 1	Indicates that a message shall be handled by COM. If “Yes” is chosen the message will be handled by COM otherwise not. Please note for network management messages: For FGA Class C Nm messages (STATUS_C) this attribute shall be set to “Yes”. For FGA Class B NM or AUTOSAR NM messages that attribute shall be set to “No”.
GenMsgSendType	Message	Enum	Cyclic = 0, NotUsed, NotUsed, NotUsed, NotUsed, NotUsed, NotUsed, NotUsed, **NoMsgSendType = 8**	Specifies the Tx behavior if the I-PDU. Can be combined with any kind of GenSigSendType.
GenSigSendType	Signal	Enum	NotUsed, OnWrite = 1, OnWriteWithRepetition = 2, OnChange = 3, OnChangeWithRepetition = 4, IfActive = 5, IfActiveWithRepeition = 6, **NoSigSendType = 7**	Specified the Tx behavior if a Signal. OnChange and IfActive are only supported for signals <= 4 Byte. Please note: The combination of transmission types with repetition and without repetition will result in the message being transmitted with repetition at any time.
GenMsgCycleTime	Message	Integer	**0**.. 65535	Time in ms between each cyclic transmission of a message.
GenMsgCycleTimeFast	Message	Integer	**0**.. 65535	Relevant for DualCycle and BAF messages: Time in ms between each cyclic transmission of a message if at least one IfActiveSignal has a different value as its default value. Relevant for messages with repetitions (i.e. GenMsgNrOfRepetition > 0): Time between each repetition.
GenSigStartValue	Signal	Integer Hex	**0x00**.. 0x7FFFFFFF	This Value is the default value for the signal. The string value type can represent hexadecimal and integer

Attribute Name	Object Type	Type	Values and Ranges (**Bold** = default)	Description
				values.
GenSigInactiveValue	Signal	Integer Hex	**0x00**.. 0x7FFFFFFF	Indicates the inactive value of a signal. IfActive and IfActiveWithRepetition only supported for signals <= 4 Byte.
GenMsgDelayTime	Message	Integer	**0**.. 65535	This is the minimum time in ms between the transmissions of different messages with the same identifier.
GenMsgStartDelayTime	Message	Integer	**0**.. 65535	This defines the time in ms between Com_IpduGroupStart and the first transmission of the cyclic part of this I-PDU.
GenMsgNrOfRepetition	Message	Integer Hex	**0x00**.. 0xFF	Number of confirmed transmit requests upon a triggering event. If the value is 0 or 1 the PDU will be transmitted once. The time between the repetitions has to be defined using the dbc attribute GenMsgCycleTimeFast.
GenSigTimeoutTime_<Ecu>	Signal	Integer	**0**.. 65535	Timeout time in ms used for this signal received by a specific node. If different GenSigTimeoutTime values are configured for a message and update bits are not used, the lowest timeout time (strongest definition) is used for timeout monitoring. A dedicated attribute definition (GenSigTimeoutTime_<Ecu>) has to be provided for every ECU receiving this signal.

Table 3-2 List of COM Data Base Attributes

3.2.1 Mapping of FGA Transmission Modes to DBC Attributes

For all transmission modes the attribute

> GenMsgILSupport = Yes

shall be set.

Periodic

> GenMsgSendType = Cyclic

> GenMsgCycleTime = Cycle time

> GenMsgStartDelayTime = Optional

- GenMsgDelayTime = Optional
- GenSigTimeoutTime = Optional
- GenSigSendType = All signals have to be set to NoSigSendType

Periodic/Event CONTROL and Event CONTROL

These transmission modes are supported by carry-back ECUs only. These rules must be applied as soon as soon a message includes at least one CONTROL signal.

- NmType = Fiat_Class_C or FIAT
- GenMsgSendType = NoMsgSendType
- GenSigSendType = All signals have to be set to NoSigSendType including signals that are actually STATUS signals.

Application Task: The transmission has to be triggered by the application manually. For this purpose the application has to use the API Com_TriggerIPDUSend.

Periodic/Event STATUS and Cyclic on Change

- GenMsgSendType = Cyclic
- GenMsgCycleTime = Cycle time
- GenMsgStartDelayTime = Optional
- GenMsgDelayTime = Optional
- GenSigTimeoutTime = Optional
- GenSigSendType = NoSigSendType for signals that cannot trigger the transmission. Signals that can trigger the message transmission shall be set to OnChange or OnChangeWithRepetition. Within one message OnChange and OnChangeWithRepetition must not be combined.
- GenMsgNrOfRepetition = If signal sent type is OnChangeWithRepetition only.
- GenMsgCycleTimeFast = Time between each repetition. Only required if sent type is OnChangeWithRepetition.

Event STATUS and On Change

- GenMsgSendType = NoMsgSendType
- GenMsgDelayTime = Optional
- GenSigTimeoutTime = Optional
- GenSigSendType = NoSigSendType for signals that cannot trigger the transmission. Signals that can trigger the message transmission shall be set to OnChange or OnChangeWithRepetition. Within one message OnChange and OnChangeWithRepetition must not be combined.
- GenMsgNrOfRepetition = Required if signal sent type is OnChangeWithRepetition.

- GenMsgCycleTimeFast = Time between each repetition. Only required if sent type is OnChangeWithRepetition.

Spontaneous

- GenMsgSendType = NoMsgSendType
- GenMsgDelayTime = Optional
- GenSigTimeoutTime = Optional
- GenSigSendType = NoSigSendType for signals that cannot trigger the transmission. Signals that can trigger the message transmission shall be set to OnWrite or OnWriteWithRepetition. Within one message OnWrite and OnWriteWithRepetition must not be combined.
- GenMsgNrOfRepetition = Required if signal sent type is OnWriteWithRepetition.
- GenMsgCycleTimeFast = Time between each repetition. Only required if sent type is OnChangeWithRepetition.

Event on message reception

- GenMsgSendType = NoMsgSendType
- GenSigTimeoutTime = Optional
- GenSigSendType = One signal shall be OnWrite which is used to trigger the message transmission. All other signals shall be set to NoSigSendType.
- GenMsgNrOfRepetition = If signal sent type is OnChangeWithRepetition only
- GenMsgCycleTimeFast = Time between each repetition. Only required if send type is OnChangeWithRepetition.

Application Task: The Rx Confirmation of the request message has to be used to trigger the transmission of the EM message. By writing to the signal with Tx type OnWrite the transmission is triggered.

Multi Message

- GenMsgSendType = NoMsgSendType
- GenSigSendType = One signal shall be OnWrite which is used to trigger the message transmission. All other signals shall be set to NoSigSendType

Application Task: Application has to trigger the next segment within the TxConfirmation of a signal within this message. The message segmentation has to be implemented by the application. A Signal Group may be used in order to indicate that the signals have to be kept consistently.

By Active Function (BAF)

- GenMsgSendType = NoMsgSendType
- GenMsgCycleTimeFast = Cycle time if at least one signal with GenSidSendType IfActive has a value different to GenSigInactiveValue.

- GenMsgStartDelayTime = Optional
- GenMsgNrOfRepetition = Number of transmissions
- GenSigTimeoutTime = Optional
- GenSigInactiveValue = Default value of the signal
- GenSigSendType = IfActiveWithRepetition

Dual Cycle

- GenMsgSendType = Cyclic
- GenMsgCycleTime = Cycle time that is used if all signal with GenSidSendType IfActive have a value equal to GenSigInactiveValue.
- GenMsgCycleTimeFast = Cycle time if at least one signal with GenSidSendType IfActive has a value different to GenSigInactiveValue.
- GenMsgStartDelayTime = Optional
- GenSigTimeoutTime = Optional
- GenSigInactiveValue = Default value of the signal
- GenSigSendType = IfActive

Dual Cycle plus On Change (DCOC)

- GenMsgSendType = Cyclic
- GenMsgCycleTime = Cycle time that is used if all signal with GenSidSendType IfActive have a value equal to GenSigInactiveValue.
- GenMsgCycleTimeFast = Cycle time if at least one signal with GenSidSendType IfActive has a value different to GenSigInactiveValue.
- GenMsgStartDelayTime = Optional
- GenSigTimeoutTime = Optional
- GenSigInactiveValue = Default value of the signal
- GenSigSendType = Signals switching the cycle shall be set to IfActive. Signals that shall trigger the transmission shall be set to OnChange.

Dual Cycle plus On Change With Repetitions (DCOCrep)

- GenMsgSendType = Cyclic
- GenMsgCycleTime = Cycle time that is used if all signal with GenSidSendType IfActive have a value equal to GenSigInactiveValue.
- GenMsgCycleTimeFast = Cycle time if at least one signal with GenSidSendType IfActive has a value different to GenSigInactiveValue. Time between each repetition.
- GenMsgStartDelayTime = Optional

- GenMsgNrOfRepetition = Number of transmissions
- GenSigTimeoutTime = Optional
- GenSigInactiveValue = Default value of the signal
- GenSigSendType = Signals switching the cycle shall be set to IfActive. Signals that shall trigger the transmission shall be set to OnChangeWithRepetitions.

3.3 Attributes for AUTOSAR Network Management

The following table provides a list of data base attributes that shall be used if the CANNM is used on a network.

Attribute Name	Object Type	Type	Values and Ranges (**Bold** = default)	Description
NmType	Network	String	NmAsr	This attribute defines the type of the NM. Must be set to "**NmAsr**" for AUTOSAR NM networks.
NmAsrNode	Node	Enum	No = 0, **Yes** = 1	This attribute defines if the corresponding node uses the AUTOSAR NM ("**Yes**") on this channel or not ("**No**").
NmAsrTimeoutTime	Network	Integer	1..**1000**.. 65535	This attribute defines the NM Network Timeout Time
NmAsrWaitBusSleepTime	Network	Integer	1..**750**.. 65535	This attribute defines the Wait Bus Sleep Time
NmAsrRepeatMessageTime	Network	Integer	1..**800**.. 65535	This attribute defines the Repeat MessageTime
NmAsrMessage	Message	Enum	**No = 0,** Yes = 1	This attribute identifies a message as AUTOSAR NM message ("**yes**").
NmAsrMessageCount	Network	Integer	1..**64**..256	This attribute defines the maximum number of AUTOSAR NM messages received by the NM (message range) Value must be 2 to the power of n and n has to be natural number
NmAsrBaseAddress	Network	Hex	**0xE094000**.. 0x7FFFFFFF	Base address of the NM messages; Identifies together with NmAsrMessageCount the NM message range: <NmAsrBaseAddress>.. <NmAsrBaseAddress + NmAsrMessageCount - 1> Value must be an integer multiple of NmAsrMessageCount
NmAsrCanMsgCycleTime	Network	Integer	1..**200**.. 65535	NM message cycle time
NmAsrCanMsgReducedTime	Node	Integer	1..**100**.. 65535	Message time for Bus Load Reduction. This attribute is node specific and has to be greater or equal to ½ * NmAsrCanMsgCycleTime but less than NmAsrCanMsgCycleTime

Attribute Name	Object Type	Type	Values and Ranges (**Bold** = default)	Description
NmAsrCanMsgCycleOffset	Node	Integer	**0**.. 65535	NM message transmission offset. Has to be set less than NmAsrCanMsgCycleTime.
NmAsrNodeIdentifier	Node	Hex	**0x00**..0xFF	Default address of Arbitrary Address Capable nodes. Node ID as used in the NID of the NM message (if used)

Table 3-3 List of NM Data Base Attributes

3.4 Attributes for FGA Class C Networks (Non-Gateway, Without Wakeup)

Attribute Name	Object Type	Type	Values and Ranges (**Bold** = default)	Description
NmType	Network	String	Fiat_Class_C	This attribute defines the type of the NM. Must be set to "**Fiat_Class_C**" for ClassC networks. No further NM attributes are required. The STATUS_C message shall be assigned to COM (GenMsgILSupport = Yes).
NmNode	Node	Enum	No = 0, **Yes = 1**	Attribute must always be set to "**No**" for nodes without wakeup capability.
NwmNodeType	Node	Enum	kMaster = 0, kSlave15 = 1, kSlave30 = 2, **none = 3**	Attribute must always be set to "**none**".

Table 3-4 Class C Network Management

Additionally a cyclic STATUS_C message may be transmitted as application message (chapter 3.2):

- GenMsgILSupport == Yes
- NmMessage == No (or undefined)
- NmAsrMessage == No (or undefined)

3.5 Attributes for FGA Class C Network Management (with wakeup)

Attribute Name	Object Type	Type	Values and Ranges (**Bold** = default)	Description
NmType	Network	String	Fiat_Class_C	This attribute defines the type of the NM. Must be set to "**Fiat_Class_C**" for ClassC networks.

Attribute Name	Object Type	Type	Values and Ranges (**Bold** = default)	Description
NmNode	Node	Enum	No = 0, **Yes = 1**	This attribute defines if the corresponding node uses the FGA ClassC Network Management (with wakeup) or not
NwmNodeType	Node	Enum	kMaster = 0, kSlave15 = 1, kSlave30 = 2, **none = 3**	If the node is a ClassC Wakeup Master node the attribute must be set to “**kMaster**”. If the node is a Wakeup Slave Node the attribute shall be set “**kSlave30**”. Nodes without wakeup capability shall be defined according to chapter 3.4.
NmMessage	Message	Enum	**No = 0,** Yes = 1	This attribute defines that a message is a FGA Class C Network Management message. The ClassC NM Wakeup message must be set to “**Yes**”

Table 3-5 Class C Network Management

The WAKE_C NM message shall be defined as follows:

- NmMessage == Yes
- NmAsrMessage == No (or undefined)
- GenMsgILSupport == No
- Slave Nodes only: The message shall contain a signal named “WakeUpNodeAddress”. The initial value (GenSigStartValue) defines the nodes wakeup address. Master nodes do not require this signal.

Additionally a cyclic STATUS_C message may be transmitted as application message (chapter 3.2):

- GenMsgILSupport == Yes
- NmMessage == No (or undefined)
- NmAsrMessage == No (or undefined)

3.6 Attributes for FGA Class B Network Management (Slave)

The following rules are applicable for Clamp15 and Clamp30 ECUs as long as not stated otherwise.

Attribute Name	Object Type	Type	Values and Ranges (**Bold** = default)	Description
NmType	Network	String	FIAT	This attribute defines the type of the NM. Must be set to “**FIAT**” for ClassB networks.
NmNode	Node	Enum	No = 0, **Yes** = 1	Defines that the related node makes use of the NM module (“**Yes**”) or not (“**No**”).
NwmNodeType	Node	Enum	kMaster = 0, kSlave15 = 1, kSlave30 = 2, **none** = 3	Specifies the ClassB NM mode use on this channel (i.e. Slave Clamp 30 or Slave Clamp 15).
NmStationAddress	Node	Hex	0x00..0xFF	Station address of the NM node. Only relevant for Master nodes.
NmMessage	Message	Enum	**No = 0,** Yes = 1	Defines that the related message is a NM message (“**Yes**”).

Table 3-6 Class B Network Management

The user-data contained in the NM message will be written to the message by the NM module.

3.7 Attributes for Diagnostic (DCM)

The following data base attributes are utilized to configure the DCM and CANTP modules.

Attribute Name	Object Type	Type	Values and Ranges (**Bold** = default)	Description
DiagState	Message	Enum	**No** = 0, Yes = 1	Set to "**yes**" for: - Enhanced UDS Functional request - OBD request
DiagRequest	Message	Enum	**No** = 0, Yes = 1	Set to "**yes**" for: - Enhanced UDS Physical Request - Flow Control PDU used for OBD Responses
DiagResponse	Message	Enum	**No** = 0, Yes = 1	Set to "**yes**" for: - Enhanced UDS Response - OBD Responses
DiagRoeResponse	Message	Enum	**No** = 0, Yes = 1	Optional attribute required for networks with RoE service. Set to "**yes**" for: - RoE Response (transmitted by ECUs supporting RoE)
DiagRoeFlowControl	Message	Enum	**No** = 0, Yes = 1	Optional attribute required for networks with RoE service. Set to “**yes**” for: - RoE Flow Control (transmitted by Tester)
DiagConnection	Message	Integer	**0x00**..0xFFFF	Groups DiagRequest and

Attribute Name	Object Type	Type	Values and Ranges (**Bold** = default)	Description
		Hex		DiagResponse and RoE messages to a connection. RoE Response and RoE Flow Control message shall have the same DiagConnection value as the related enhanced UDS Request and Response. The value shall be derived as follows: <TesterAddress><ServerAddress>.
IsOBDrelevant	Node	Enum	**No = 0,** Yes = 1	Specifies if an ECU is OBD-relevant (“**Yes**”) or not (“**No**”). If set, the Source Address F1 will be interpreted as OBD Request. Otherwise Request, Response and Functional Request will be handled as enhanced UDS.
TpOwnSystemEcuNumber	Node	Integer Hex	**0x00**..0xFF	Optional Attribute. Used by the CANbedded (non AUTOSAR) tool chain only. Will be ignored by AUTOSAR tools. This value provides the ECU Number, necessary for setting up the Tx CAN ID using Normal Fixed TP addressing.

Table 3-7 List of DCM Data Base Attributes

3.7.1 Examples

Note
These examples assume that the enhanced UDS tester ID is 0xF4 as defined by FGA for AUTOSAR projects. Carry back projects use the tester ID 0xF1.

3.7.1.1 Enhanced UDS Communication

Example of DBC attributes for a communication between the Tester tool (0xF4) and one Node (0x12):

Enhanced UDS Request (also used as Flow Control for Response):

- DiagState = No
- DiagRequest = Yes
- DiagResponse = No
- DiagRoeResponse = No
- DiagRoeFlowControl = No
- DiagConnection = F412

Note: CANID is 0x18DA12F4

Enhanced UDS Response (also used as Flow Control for Request):

- DiagState = No
- DiagRequest = No
- DiagResponse = Yes
- DiagRoeResponse = No
- DiagRoeFlowControl = No
- DiagConnection = F412

Note: CANID is 0x18DAF412

Functional Request

- DiagState = Yes
- DiagRequest = No
- DiagResponse = No
- DiagRoeResponse = No
- DiagRoeFlowControl = No
- DiagConnection = F4FE

Note: CANID is 0x18DBFEF4

RoE Response (Optional):

- DiagState = No
- DiagRequest = No
- DiagResponse = No
- DiagRoeResponse = Yes
- DiagRoeFlowControl = Now
- DiagConnection = F412

Note: CANID is different to the enhanced UDS Response: 0x18DAF412

RoE Flow Control (Optional):

- DiagState = No
- DiagRequest = No

- DiagResponse = No
- DiagRoeResponse = No
- DiagRoeFlowControl = Yes
- DiagConnection = F412
- Note: CANID is different to the enhanced UDS Request: 0x18DA12F4

3.7.1.2 Legislative OBD Communication

Example of DBC attributes for a communication between the OBD tool (0xF1) and one Node (0x12):

OBD Request (Functional):

- DiagState = Yes
- DiagRequest = No
- DiagResponse = No
- DiagConnection = F133

Note: CANID is 0x18DB33F1

OBD Response:

- DiagState = No
- DiagRequest = No
- DiagResponse = Yes
- DiagConnection = F112

Note: CANID is 0x18DAF112

Flow Control Frame for OBD Response

- DiagState = No
- DiagRequest = Yes
- DiagResponse = No
- DiagConnection = F112

Note: CANID is 0x18DA12F1

3.8 Definition of XCP and application messages

If a message shall be handled by a non AUTOSAR module – such as XCP or another CDD – the message shall not be assigned to any of the layers such as COM, DCM or network management.

The following attributes must be set as defined:

- GenMsgILSupport = No
- NmMessage = No (or attribute not available)
- NmAsrMessage = No (or attribute not available)
- DiagState = No (or not available)
- DiagRequest = No
- DiagResponse = No

Furthermore one signal shall be defined in the message that spans over the complete message. The signal has to be used for the Rx mapping of the message in the DBC file.

3.9 Definition of Update Bits

In order to use update bits in the DBC file a naming convention has been defined. The update bit is therefore defined as an independent signal with some specific characteristics. Update bits can be used for signals and signal groups.

- GenMsgILSupport = Yes
- The update bit of the signal (or signal group) with the name <X> is configured as a further signal with the name <X>_UB
- The update bit <X>_UB signal must be in the same message as the signal (or signal group) <X>
- The update bit “signal” shall have a bit size of 1bit.
- GenSigSendType shall be NoSigSendType for the update bit.

In the system description this update bit signal will be modelled as an AUTOSAR update bit (i.e. the signal definition will be removed and an update bit will be added).

3.10 Definition of Invalid Values

If the AUTOSAR mechanism of an invalid value shall be used for a signal, the DBC value table has to be extended by a value description called “SNA”. The value defined for this description will be used as invalid value by all ECUs sending or receiving this signal.

If a signal has no SNA description in the value table, the invalid value feature will not be used for this signal.

4 LDF Rules and Conventions

The FGA AUTOSAR Stack shall use an LDF file according to LIN 2.1. In the LDF file, the following elements have to be set as follows (see [5]):

LIN_protocol_version = "2.1";

LIN_language_version = "2.1";

This chapter defines additional requirements to this standard in order to be able to configure some AUTOSAR related features that are not available in the LDF.

4.1 Definition of LIN Channel Name

The LIN Channel Name shall be defined in all LDF files by using the LDF description element ‘Channel_name’ (see [5]). The channel name must be unique over all CAN (DBName attribute) and LIN channels.

4.2 Definition of Update Bits

The LIN Master Node (AUTOSAR ECU) shall be able to support update bits (e.g. for CAN to LIN signal routing). In order to use update bits in the LDF file a naming convention has been defined. The update bit is defined as an independent signal with some specific characteristics.

- The update bit of the signal with the name <X> is configured as a further signal with the name <X>_UB
- The update bit <X>_UB signal must be in the same message as the signal <X>
- The update bit “signal” shall have a bit size of 1bit.
- The default value shall be 0 (required for the LIN Slaves that handle update bits as ordinary signals)

In the system description, this update bit signal will be modelled as an AUTOSAR update bit (i.e. the signal definition will be removed and an update bit will be added).

LIN Slaves will read the LDF directly and will handle the Update Bits as an ordinary signal which has to be evaluated manually before reading the actual signal.

4.3 Definition of invalid values

In order to allow the invalidation of LIN Signals by the LIN Master node the LDF element ‘Signal_encoding_types’ (see [5]) shall be used. The value that shall be used as invalid value shall have the name ‘SNA’. This is similar to the solution on CAN where value tables are used.

The LIN slaves will handle this “SNA” value as a normal signal value that must be considered as invalid. This will be handled by the LIN Slave application manually.

4.3.1 Example

```
Signal_encoding_types {
   FrontLeftSwitchStatus_Sig_Type {
      physical_value, 0, 2, 1, 0;
      logical_value, 2, "down";
      logical_value, 3, "SNA";
   }
}

Signal_representation {
   FrontLeftSwitchStatus_Sig_Type : FrontLeftSwitchStatus;
}
```

5 General Rules

These rules apply for LDF and DBC files in the same way.

5.1 Definition of AUTOSAR System Signals

AUTOSAR specifies a so called “Signal Fan-out” which is implemented by the RTE.

Using this mechanism allows the application to write to a single SWC data element which is then transmitted using several COM Signals. I.e. the signal is transmitted on e.g. within several messages on several networks.

In order to configure this mechanism using DBC and LDF files a naming convention is used:

If a node transmits several signals (n) with the same signal name, the same signal size and (if available) the same value table, n COM signals will be created but only a single system signal. If the RTE accesses the system signal a fan-out resp. fan-in will be generated that writes the signal value to all n COM signals.

The mechanism is supported for LDF and DBC (GenMsgILSupport = Yes) signals.

5.2 Definition of Routing Relations

Routing relations between messages and signals of DBC and LDF files are to be defined as follows.

5.2.1 Signal Routing (by COM)

Signal routing relations between two signals (in different messages) is defined by an equal signal name in the DBC or LDF fie.

When routing signal groups, a routing relation between any GroupSignal has to be established. The tooling will automatically detect a routing between the SignalGroup if applicable.

5.2.2 Message Routing (IF Layer Routing by PDUR)

Routing relations between two messages (on different networks) are defined by an equal message name in the DBC or LDF file. Please note that the database design must ensure that the two messages are similar with respect to the signal layout.

If a message routing has been detected that links two messages that contain signals with the same names, only a message routing between these two messages will be created. No signal routing will be created as this would be redundant.

Please note that message routing forwards received PDUs without changes to the payload to the destination network. There is no transmission mode conversion possible – i.e. the reception always triggers the transmission of the PDU on the destination.

5.2.3 Diagnostic Routing (TP Gateway Routing by PDUR)

Routing relations between two diagnostic messages (on different networks) are defined by an equal message name in the DBC. Please note that the database design must ensure that the two messages are similar with respect to the signal layout.

> Additionally, **Physical diagnostic requests** or **responses** will be identified by the attributes DiagRequest = Yes / DiagResponse = No or DiagRequest = No / DiagResponse = Yes. DiagState is set to No.

Diagnostic message type	Attributes
Physical diagnostic requests	DiagRequest = Yes DiagResponse = No DiagState = No
Physical diagnostic responses	DiagRequest = No DiagResponse = Yes DiagState = No

In case of two diagnostic messages with an equal message name a routing relation is created.

> Additionally, **Functional diagnostic requests** will be identified by the attribute DiagState = Yes.

Diagnostic message type	Attributes
Functional diagnostic requests	DiagRequest = No DiagResponse = No DiagState = Yes

In case of two diagnostic messages with an equal message name a routing relation is created.

> Please note, that, in case a functional diagnostic request shall not be routed from one bus to the other bus, the message should not have the same name, so that no routing relation is created.

6 Contact

Visit our website for more information on

- News
- Products
- Demo software
- Support
- Training data
- Addresses

www.vector.com