

This is a new standard that has not yet been approved by the NMRA Board of Directors.

| | |
|-----------------------------|---------|
| NMRA STANDARD | |
| Electrical Standards | |
| For NMRAnet™ Bus, | |
| All Scales | |
| DRAFT May 2010 | S 9.5.1 |

1. NMRAnet™ CAN Standard

CAN Standard Version 2.0 A&B shall be adhered to for all signals on the CAN bus. The CAN Extended frame shall be used by NMRAnet to transfer data in all modes of operation.

All NMRAnet modules shall support the Fully Operational Mode.

NMRAnet modules may support the Simple Mode of Operation. When a module is first enabled it looks for the Network Manager sending of the Layout_ID Frame (at least once per second), if it does not see a Layout_ID packet within 2 seconds it can assume simple mode operation. This simple mode of operation is intended for use by small layouts, probably less than 5-10 modules, but allow the user to grow his layout to a more robust and capable fully operational network with a greater amount of user tools.

The baud rate shall be 125khz.

2. Extended Frame Fully Operational Mode Definition:

In the Extended frame the CAN header is 29 bits. The format then defines the length of the data 0-8 bytes.

NMRAnet defines the frame as follows:

<P:1><Message_Type:4><Command/Channel:8><Addr/Event/Seq#:16><DNL:4>{data}

There are two additional bits in the Header; IDE and RTR.

IDE = 1 for Extended frame (shall be set)

RTR = 0 for Normal Frame

RTR = 1 for Special Frame used in requesting a new Node_ID

<P:1> = IDE28 Priority bit for use by bridges or gateways to clear their buffers. Recessive priority = 1, Dominate priority = 0.

<Message_Type:4> = IDE27-24 Message Type usage

<Command/Channel:8> = IDE23-16 Definition depends on message type

<Addr/Event/Seq#:16> = IDE15-0 Definition depends on message type

<DNL:4> Data Length from 0-8

{data} up to 8 bytes of data

Message Types

| | | |
|----|------|-----------------------------------|
| | 0000 | Network Manager => Slaves |
| | 0001 | Network Manager <= Slaves |
| 35 | 0010 | Configuration Tool => Slave |
| | 0011 | Configuration Tool <= Slave |
| | 0100 | Reserve |
| | 0101 | Reserve |
| | 0110 | Reserve |
| 40 | 0111 | Reserve |
| | 1000 | Producer/Consumer Event broadcast |
| | 1001 | Reserve |
| | 1010 | Train Control => Throttles |
| | 1011 | Train Control <= Throttles |
| 45 | 1100 | Reserve |
| | 1101 | Data Streaming |
| | 1110 | Reserve |
| | 1111 | Invalid- never used |

50 The above definition of the Frame Message Type allows for optional use of the CAN filters found in all CAN modules. As the number of modules increases along with an increase of producers, peak Frame activity can be high. The use of the CAN filters can reduce the chance of overload. For example; a simple accessory module can filter the bus messages with its filter to only receive; <0000> NM=>, < 0010> CT=>, and <1000> Events. This will greatly reduce its processing overhead by selecting only
55 those messages that could concern it.

3. Command / Channel

For message types Network Management, Configuration Tool and Train Control these 8 bits are defined as the Command. For message types Events or Data Streaming these bits are the Channel number.

60 4. Address / Event # / Sequence

For message types Network Management, Configuration Tool and Train Control these 16 bits are the Node_ID of the Slave being talked to or sending back information. Address 0 is reserve for a broadcast address to all slave Nodes.

For message type Event, the 16 bits are the Event number.

65 For message type Data Streaming, the 16 bits are the sequence number of the data packet.

5. Network Management

There shall be a Network Manager in the Fully Operational Mode on each Layout with the following functions:

70 Providing a 32 bit unique Layout_ID for the layout. This is made up of the 8 bit manufactures code plus a 24 bit sequence number.

Power up sequencing of all modules to an operational state.

Assigning Node_IDs to all new nodes to the layout

Provide feedback to the USER of the state of the layout

75 Provide a means for the USER to save the current state of the Consumers (push button)

Perform Node Guarding. This function has the NM continuously (every 1/4 to 1 sec) querying each module for its status.

80 Notification of any network errors and provide addition information on these errors.

85 The Network Management functions can either be a separate module or combined with other network functions; gateways, command stations, etc. However, due to its interaction with the USER it must have visibility by the user. The Network Manager could use simple LEDs to reflect the status of the network or include a display device (LCD). This is also the ideal location to include a USB connection to a computer, to enable programs on the computer to assist in any debugging of network errors. This connection can then be used by the Configuration Tool to program the network devices.

6. Network Management Commands

90 NM =>Slave

Layout_ID and request for Node Status;

Command= 0

```
<1><0><0><Node_ID><5>{layout_ID:32}<network_state:8}
```

95 If the Node_ID is 0, this is a broadcast request for all valid nodes on the layout to report their status. If the Node_ID is other than 0 then just that node responds. At this time the network_state is defined:

| | |
|---------------|-----------------|
| 010000000 | Operational |
| 001000000 | Pre-Operational |
| 100 000100000 | Halted |

Every Network Management function has a unique 32 bit identification number. This number is used by the modules attached to the network to identify the layout and if new to the layout, request a unique Node_ID for this layout.

105 All nodes on this layout for the first time will identify itself using the special RTR and its Long_ID. The Long_ID is a 24 bit value generated by the manufacture of the module with the manufactures 8 bit code plus a 16 bit sequence number.

Command = 1

110 Setting a New Node's Node_ID: <1><0><1><new Node_ID><3><Long_ID>

The Network Manager addresses the new Node using the supplied Long_ID and provides a unique Node_ID. At this time the new Node saves the given Node_ID and the Layout_ID for use in any further interaction on the network.

Move to State;

115 Command = 2

Operational: <1><0><2><Node_ID><0>

Command = 3

Pre-Operational: <1><0><3><Node_ID><0>

Command = 4

120 Halted: <1><0><4><Node_ID><0>

Command = 5

Reset: <1><0><5><Node_ID><0>

if Node_ID = 0, broadcast to all nodes.

125 **NM <= Slave**

Return the Node's status in response to a broadcast or Node directed request:

<1><1><0><Node_ID><3><node_state><status0><status1>

The node's state bits are defined as follows:

Bit7 = fixed node(0) or mobile node(1)

130 Bit6 = Operational

Bit5 = Pre-Operational

Bit4 = Halted

Bit3-0 Reserved

The Node's status byte reflect its error status

135 Special response to request a new Node_ID when the Node is new to the Network:

<1><1><manufacture#><Manuf. Sequence #><DNL>

RTR = 1

DNL can be from 0-8 as a special indicator to the type of Node

140 A DNL = 0 is a standard fixed Node

The Manufacture # and the manuf. Sequence # is referred to as the node's Long_ID

145 ##Note; the Manufacture will assign a sequence number to each module. If more than 65536 sequence numbers are used then the sequence will start over from 0.

7. Network Management sequencing at Power up:

After a 1 second delay the NM will send out a broadcast Layout_ID Frame which also requests all valid nodes to report their status. The NM uses this to confirm the active nodes on the layout and to compare it with the saved nodes on the layout.

150 If a Node does not compare with the its saved Layout_ID it requests a New Node_ID
The NM assigns a unique Node_ID to each New Node. Each New Node upon receiving a Node_ID saves both the Layout_ID and the Node_ID for all further communication on the network. After receiving a new Node_ID, the Node responds with a Node Status Frame.

155 The NM then moves the Network to the Operational State if all conditions on the network are good.

##Note; If the USER has made any changes which result in Network Errors, this is the point where the worst of these type errors will be noted. The NM can also report any module which is not reporting.

160 8. Network Management Node Guarding:

Node Guarding is used to check and report on any errors that take place on the network during operation. The NM sends out a Layout_ID, Node_ID status request every second. The Node_ID must respond with its status indicating any errors it encountered since the last Status request. The NM continues to cycle through all the Nodes.

165 ##Note; This repeated Frame with the Layout_ID is also used by any node plugging into the Layout (mobile nodes or late power up nodes) to identify the layout. A node powering up on an operational layout will ID itself to the NM with a Node Status response if it is a valid node. A new node will request a new node identification.

170 9. Configuration Tool

All Modules shall have within it's programming a compressed XML file which describes it capability and how the Configuration Tool will interact with it. A Configuration Tool after identifying the Nodes on a layout would request this compressed XML file. From this point on the CT would interact with the Node as is described in that Node's XML file.

175 The minimum set of Configuration commands all modules shall support:

CT request for all Nodes to report status: Command = 0
<1><2><0><Node_ID><0> if Node_ID = 0 broadcast request to all nodes.

180 CT request to a Node to supply its Compressed XML: Command = 1
<1><2><1><Node_ID><0> must be a valid Node_ID

Nodes response to a CT status request Command = 0
<1><3><0><Node_ID><3>{<node_state><status0><status1>}

Node's response to CT request for the Compressed XML file;

185 The Node will supply the XML file to the CT using Channel 0 of the Data Streaming Message_Type:

<1><13><0><seq#><1-8>{data} Channel 0 is reserved for CT use

At the completion of the transfer an ACK frame is given: Command = 1

<1><3><1><Node_ID><0>

190 To complete the command set, a Node can respond with a NACK frame:
Command = 2

<1><3><2><Node_ID><0>

The above is the minimum set that shall be supported, but to provide an example on further interactions that could be established by the onboard XML file, the following was used during prototyping:

195 **CT => Node**

CT => Read a variable from a Node's Device: Command = 2

<1><2><2><Node_ID><2>{<Device_Offset:8><Variable_Offset:8>}

CT => Write a variable to a Node's Device: Command = 3

<1><2><3><Node_ID><3-8> {<Device_Offset:8><Variable_Offset:8>[data]}

200 CT => Read Event Table: Command = 4

<1><2><4><Node_ID><0>

CT => Erase an Event Tie from the Event Table: Command = 5

<1><2><5><Node_ID><3>{<Event:16><Device_Offset>}

CT => Add Event Tie to a Consumer in the Event Table: Command = 6

205 <1><2><6><Node_ID><4> {<Event:16><Device_Off:8><Device_Act_Code:8>}

CT => Erase the entire Consumer Event Table: Command = 7

<1><2><7><Node_ID><0>

210 **CT<= Node**

CT<= from Node for a read: Command = 4

<1><3><4><Node_ID><3-8>{<Device_Off:8><Variable_Off:8> [data]}

215 After any requested action the Node will respond with data on a read or for writes an ACK or NACK. The request for the Event Table is transferred using Channel 0 of Data Streaming and is completed with an ACK.

The below would be defined in the compressed XML file:

Device_Offset:8 from 1-255. Can be either a Producer or Consumer and its capability is defined in a group.

220 Variable_Offset:8 from 0-255. Size and usage defined in the XML table for each group of devices.

Device_Action_Code:8 Defined for each Consumer Group in the XML table.

10. **Producer / Consumer Events**

225 At this time only Events without data are defined. Channel 0 is used to identify this type of Event. Events without data can be broadcasted by more than one Node without conflict. The other channels can be defined in the future to handle Events with data.

<1><8><0><Event#><0>

230 A Normal bi-state Producer can have an Event # tied to each state. This Event # is then broadcasted to the Network in the Operational state of the Network. An Event # of 0x0000 or 0xFFFF is a null Event and will not be broadcasted. This allows for a push button Producer to be tied to a single Event or an Event for each state.

235 The Consumer is more complex and needs to be able to be tied to multiple Events. In order for multiple Event Ties across all Consumers, a Consumer Event Table within each node is suggested. This allows for Consumer Ties up to the depth of the Table (table entry #). This is also the quickest method to locate a tie or reject an Event. An entry could be;

Event#:16, Device_Offset:8, Device_Action_Code:8, a 4 byte entry.

11. **Data Streaming**

240 Up to 256 channels of data streaming are possible. These channels would have to be assign by the Configuration Tool or some other manager. Channel 0 is reserved for the Configuration Tool's use.