A theoretical approach for node-ID negotiation in CANopen networks

Alexander Philipp, emotas embedded communication GmbH

The increasing complexity and size of modern CANopen and CANopen FD networks creates new challenges for system designers and device developers how to decide which CANopen node-ID should be used for their CANopen devices. This becomes more difficult in systems which are highly dynamic or having multiple devices of the same type in the network, like cascaded battery systems. Sometimes it is doesn't even matter for the system which node-ID a device has. This paper discusses a theoretical approach how devices in a CANopen or CANopen FD network could negotiate their node-ID by themselves.

Introduction

The assignment of a node-ID to a CANopen device is an essential requirement. Without a node-ID a CANopen device is unable to communicate over services defined in CiA 301 and CiA 1301.
Without a valid node-ID in the range of 1 to 127, a CANopen device would never leave the NMT sub-state reset communication. Since the node-ID has to be unique in the network, there has to be a way to configure the value for node-ID. Neither CiA 301 nor CiA 1301 specify a way to configure this value. There are various solutions in the market how this can be achieved.
Some manufactures use some hardware solution on their devices, like dip- or rotaryswitches or encode plugs. Others use some kind of proprietary software configuration. All of this solutions need more or less knowledge about the system where the devices are integrated.
Another way to configure the node-ID is specified in the document CiA 305, Layer Set t ing Services. This specificat ion describes services and protocols for identifying CANopen devices and assigning node-IDs, which can be used by a so called LSS Master or some configuration tool.
The CANopen Application profile for building door control, CiA 416, also specifies a procedure for claiming node-IDs by the devices themself. But besides that this procedure is slow, it is also patented.
The following theoretical approach for negotiation of node-IDs introduced by the engineers of the emotas embedded communication GmbH should be a starting point how to overcome most of the disadvantages of the other solutions.

Use cases, a short description

Masterless systems

Some systems are designed very flexible. That means that a system could be build with different combinations of devices, exactly meeting the needed requirements.
Sometimes it could be enough that only two devices are need, sometimes much more and sometimes there are multiple of the same device type in a network.
An example of such kind of a systems are heating systems based on CANopen. They can include multiple sensors and controllers depending location and size where the system is installed.
When creating the devices for such a system, it is not clear which kind of device could be unique in the system, so it can operate as CANopen Manager. The CANopen Manager is the CANopen Master, which is responsible as NMT master, plus additional functionality for example LSS Master.
One way to assign node-IDs without a LSS Master is when integrating the system. But this leads to static system, without a real plug and play possibility. And in case some parts are added, or have to be replaced, a n

unused node-ID has to be assigned to the replaced part, which might be impossible for service technician.
Another example are battery clusters, which are capable of plug and play. The only devices in this network are the batteries themself. There are physically all the same and have the same software. In such a system the node-ID is irrelevant to the system, but is required for CANopen communication.

Systems with a master

In generic CANopen systems with a master, there are also possibilities that the actual distribution of the node-IDs is not relevant for the functionality of the system.
Examples are the CANopen Application profile for energy management systems, CiA 454, or the CANopen Application profile for special-purpose car add-on devices, CiA 447.
These specifications use the LSS Fastscan, defined in CiA 305 to detect devices and assign node-IDs to them.
With LSS Fastscan devices can be only detected one after another, and in the worst case of completely unknown devices it will take 128 messages to verify the every single bit of the CANopen LSS address. With response-timeout of 10ms, it would take 1.28s to detect one single device.

New approach

The idea behind this approach is that, in systems which don not depend on the node-ID distribution, the devices negotiate there node-ID themself.
The idea is not new, for example in the document J1939™-81 of the SAE®, there is a description of a so called address claim procedure. In this procedure the devices in a CAN based network negotiate the Addresses, depending on values of the NAME field.
The problem here is that this procedure uses the 8 byte Data Field of the CAN frame. This lead to the fact, that in case two or more device are claiming the same Address at the could lead to collisions on the CAN bus.
In our approach we are trying to avoid that by not using the Data Field at all, all the information exchanged between devices are encoded in the Identifier Field.

Requirements

The main requirements for this procedure is that the software of the devices is able to send and receive CAN Data Frames in Classical Extended Frame Format, CEFF. Because most modern CAN FD controller support sending and receiving this kind of Data Frames, this approach works in CANopen FD as well.
That the software can distinguish between Extended and Basic Frame Format is important too.

Usage of the CAN Identifier

The CAN Identifier of an Extended Data Frame has 29 bits available. For our approach we only need 13 bits. The least significant 13 bits are used for the node-ID negotiation.
The 13 bits are divided into two fields, the least significant 8 bits are used as Data Field, and the most significant 5 bits as Data Code. The following table illustrates the usage of the CAN ID.

Table 1: 29 bit CAN ID usage

Bit 28-13	Bit 12-8	Bit 7-0
reserved	Data Code	Data Field

Using 13 bit of the 29 bits of and extended CAN ID means that 44% of the bits are required, but the following formula:

$$\frac{2^{13}}{2^{29}} = \frac{8.192}{536.870.912} = 0{,}00001525$$

shows that only 0,0015% of all possible extended CAN ID are reserved for this service.
The Data Code defines the meaning of the content of the Data field. If the most significant bit is set to 1, the Data Field contains parts of the CANopen object 0x1018, Identity object. For this procedure to work, all 4 sub-indices are required.

The following table shows which part of the Identity object is transmitted with which Data Code.

Table 2: Data Code 0x1018 correlation

Data Code	0x1018 Data
b10000	sub-index 1 - bit 0-7
b10001	sub-index 1 - bit 8-15
b10010	sub-index 1 - bit 16-23
b10011	sub-index 1 - bit 24-31
b10100	sub-index 2 - bit 0-7
b10101	sub-index 2 - bit 8-15
b10110	sub-index 2 - bit 16-23
b10111	sub-index 2 - bit 24-31
b11000	sub-index 3 - bit 0-7
b11001	sub-index 3 - bit 8-15
b11010	sub-index 3 - bit 16-23
b11011	sub-index 3 - bit 24-31
b11100	sub-index 4 - bit 0-7
b11101	sub-index 4 - bit 8-15
b11110	sub-index 4 - bit 16-23
b11111	sub-index 4 - bit 24-31

Due the binary coding of the Data Code, it is very easy to implement the algorithm for accessing the value of the Identity object. Bit 3 and 4 are the sub-index number - 1, and bit 0 and 1 are the byte number of this sub-index.
The following code fragment shows a simple implementation to create the corresponding CAN ID.

```
idx = byteNr / 4;
canId = identity[idx];
idx <<= 2;
idx += (byteNr % 4);
idx <<= 8u;

canId >>= ((byteNr % 4) * 8u);
canId &= 0xff;

canId |= idx;
canId |= 0x1000ul;
```

Two Data Code are for the flow control of the process and are listed in the following table.

Table 2: Data Code 0x1018 correlation

Data Code	Name
b00001	ReqUsedNodeId
b00010	ActUsedNodeId

The Data Code ReqUsedNodeId is used to start a new process, whereby the Data Field contains 0.
The resulting message with extended CAN ID 0x100 will trigger all devices with a valid node-ID to respond with ActUsedNodeId and their own node-ID in the Data Field.

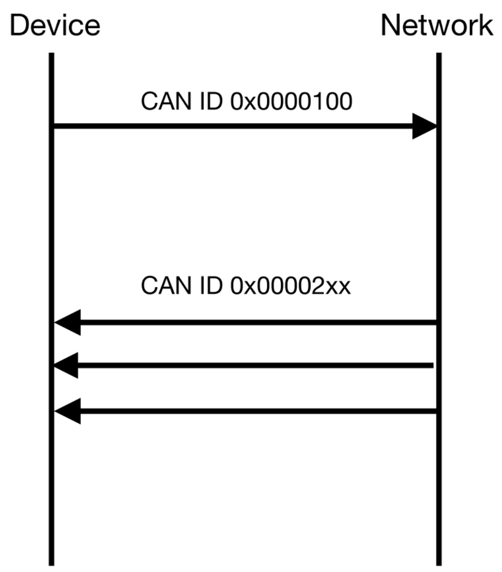
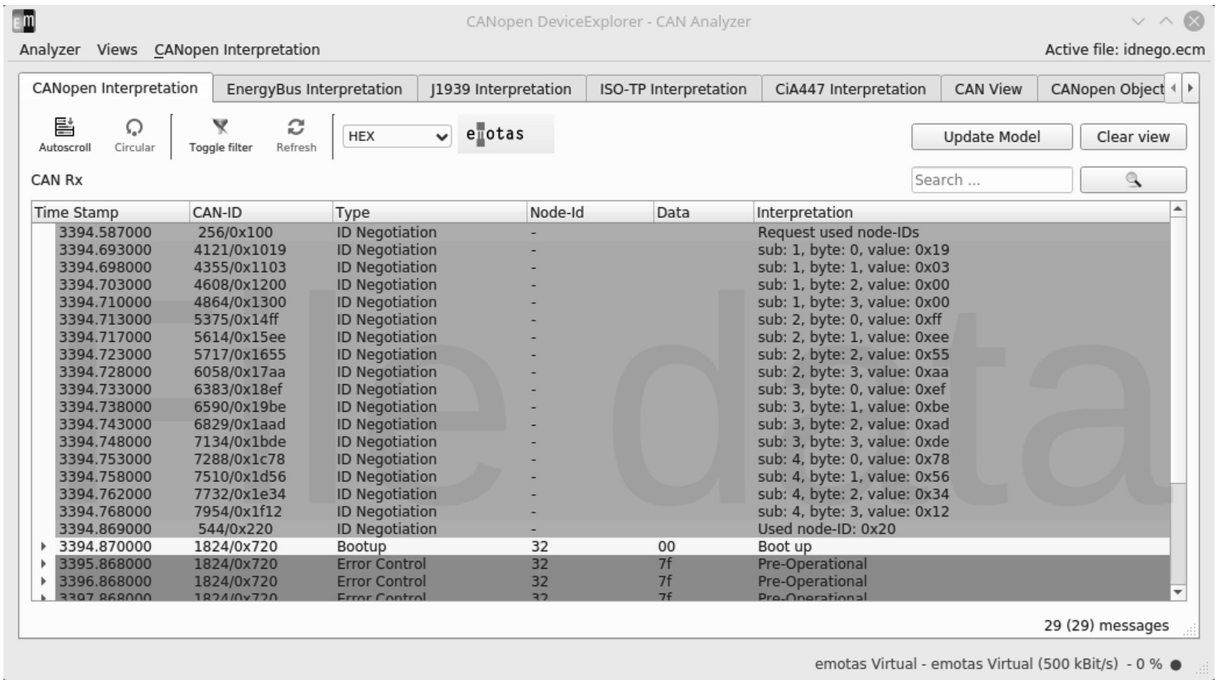


Figure 1: Initializing negotiation

Figure 1 shows the basic concept of asking the network for the node-IDs which are already used in the network.

Negotiation process

The negotiation process is only performed by nodes with an invalid node-ID.
All nodes with a valid node-ID are ignoring the messages with the most significant bit in the Data Code field set to one.
If a device participating in the procedure, receives a negotiation message it compares the value according to Table 2 with its own equivalent. In case the received value is higher than its own value, the device continues the negotiation.



Time Stamp	CAN-ID	Type	Node-Id	Data	Interpretation
3394.587000	256/0x100	ID Negotiation	-		Request used node-IDs
3394.693000	4121/0x1019	ID Negotiation	-		sub: 1, byte: 0, value: 0x19
3394.698000	4355/0x1103	ID Negotiation	-		sub: 1, byte: 1, value: 0x03
3394.703000	4608/0x1200	ID Negotiation	-		sub: 1, byte: 2, value: 0x00
3394.710000	4864/0x1300	ID Negotiation	-		sub: 1, byte: 3, value: 0x00
3394.713000	5375/0x14ff	ID Negotiation	-		sub: 2, byte: 0, value: 0xff
3394.717000	5614/0x15ee	ID Negotiation	-		sub: 2, byte: 1, value: 0xee
3394.723000	5717/0x1655	ID Negotiation	-		sub: 2, byte: 2, value: 0x55
3394.728000	6058/0x17aa	ID Negotiation	-		sub: 2, byte: 3, value: 0xaa
3394.733000	6383/0x18ef	ID Negotiation	-		sub: 3, byte: 0, value: 0xef
3394.738000	6590/0x19be	ID Negotiation	-		sub: 3, byte: 1, value: 0xbe
3394.743000	6829/0x1aad	ID Negotiation	-		sub: 3, byte: 2, value: 0xad
3394.748000	7134/0x1bde	ID Negotiation	-		sub: 3, byte: 3, value: 0xde
3394.753000	7288/0x1c78	ID Negotiation	-		sub: 4, byte: 0, value: 0x78
3394.758000	7510/0x1d56	ID Negotiation	-		sub: 4, byte: 1, value: 0x56
3394.762000	7732/0x1e34	ID Negotiation	-		sub: 4, byte: 2, value: 0x34
3394.768000	7954/0x1f12	ID Negotiation	-		sub: 4, byte: 3, value: 0x12
3394.869000	544/0x220	ID Negotiation	-		Used node-ID: 0x20
3394.870000	1824/0x720	Bootup	32	00	Boot up
3395.868000	1824/0x720	Error Control	32	7f	Pre-Operational
3396.868000	1824/0x720	Error Control	32	7f	Pre-Operational
3397.868000	1824/0x720	Error Control	32	7f	Pre-Operational

Figure 2: Complete negotiation cycle

In case the received value is smaller as its own equivalent, the negotiation is stopped on this device. And the device has to wait until the running negotiation is done. It can detect this by setting a timeout e.g. 100ms, and restarting this timeout every time it receives a negotiation message.

In case a device is able to transmit all 16 messages, which are needed to transmit all 128 bits of the Identity object, it will then announce the preferred node-ID to the network by using ActUsedNodeId Data Code message.

Figure 2 shows a complete negotiation cycle with timeout value of 100ms before and after the actual negotiation and an additional 5ms delay between every negotiation message.

Node-ID conflicts

In case a device successfully negotiated a node-ID, but another device indicates by the ActUsedNodeId message the very same node-ID, the receiving device sends out the same ActUsedNodeId and set its pending node-ID to the invalid node-ID, and enter the NMT state Reset Communication. Because of the other device receiving this message it looses the node-ID too.

Both devices can restart the negotiation process again. The loosing device of this negotiation shall alter the preferred node-ID for the next cycle to a node-ID not yet present in the system.

Example

Table 4 shows a simulation example of two nodes negotiating there node-IDs. The difference in the 0x1018 parameter Revision Number causes the one node to stop the negotiation and restarts again after the first one is done.

Table 4: CAN Trace

time	id	ide	l	d
0.000000	256/0x100	EXT	0	
0.105000	4121/0x1019	EXT	0	
0.106000	4121/0x1019	EXT	0	
0.110000	4355/0x1103	EXT	0	
0.111000	4355/0x1103	EXT	0	
0.115000	4608/0x1200	EXT	0	
0.115000	4608/0x1200	EXT	0	
0.119000	4864/0x1300	EXT	0	
0.121000	4864/0x1300	EXT	0	
0.125000	5375/0x14ff	EXT	0	
0.126000	5375/0x14ff	EXT	0	
0.130000	5614/0x15ee	EXT	0	
0.130000	5614/0x15ee	EXT	0	
0.135000	5717/0x1655	EXT	0	
0.136000	5717/0x1655	EXT	0	
0.140000	6058/0x17aa	EXT	0	
0.141000	6058/0x17aa	EXT	0	
0.145000	6383/0x18ef	EXT	0	
0.146000	6383/0x18ef	EXT	0	
0.149000	6590/0x19be	EXT	0	
0.155000	6829/0x1aad	EXT	0	
0.161000	7134/0x1bde	EXT	0	
0.165000	7201/0x1c21	EXT	0	
0.170000	7491/0x1d43	EXT	0	
0.174000	7781/0x1e65	EXT	0	
0.181000	8071/0x1f87	EXT	0	
0.279000	576/0x240	EXT	0	
0.280000	1856/0x740	_	1	0x00
0.282000	256/0x100	EXT	0	
0.283000	576/0x240	EXT	0	
0.387000	4121/0x1019	EXT	0	
0.392000	4355/0x1103	EXT	0	
0.397000	4608/0x1200	EXT	0	
0.402000	4864/0x1300	EXT	0	
0.406000	5375/0x14ff	EXT	0	
0.412000	5614/0x15ee	EXT	0	
0.417000	5717/0x1655	EXT	0	
0.422000	6058/0x17aa	EXT	0	
0.426000	6383/0x18ef	EXT	0	
0.432000	6590/0x19be	EXT	0	
0.437000	6829/0x1aad	EXT	0	
0.442000	7134/0x1bde	EXT	0	
0.447000	7423/0x1cff	EXT	0	
0.451000	7491/0x1d43	EXT	0	
0.456000	7781/0x1e65	EXT	0	
0.462000	8071/0x1f87	EXT	0	
0.561000	577/0x241	EXT	0	
0.562000	1857/0x741	_	1	0x00
1.280000	1856/0x740	_	1	0x7f
1.562000	1857/0x741	_	1	0x7f

Summary

The main goal is still to find a reliable, robust, and fast solution to assign node-IDs in plug and play systems to extend the network during its lifecycle. This theoretical approach should be only a basis for discussion of how to achieve this.
With the discussed process it is possible to assign node-IDs to devices without the need of a dedicated LSS Master. The usage of extended CAN IDs makes it possible that only 16 messages are needed, and that the process doesn't create CAN-BUS collisions.

Alexander Philipp
emotas embedded communication Gmbh
Fritz-Haber-Str. 9
DE-06217 Merseburg
www.emotas.de

References

[1] CiA 301, CANopen application layer and communication profile
[2] CiA 302, Framework for CANopen managers and programmable CANopen devices
[3] CiA 305, CANopen Layer setting services (LSS) and protocols
[4] CiA DSP 416, CANopen Application profile for building door control
[5] CiA 447, CANopen Application profile for special-purpose car add-on devices
[6] CiA 454, CANopen Application profile for energy management systems
[7] SAE J1939-81, Network Management
[8] EP1349348A1, Verfahren, Kommunikationseinrichtung und Kommunikationsmodul zur Ermittlung von Kommunikationsidentifizierern