CiA 801

Application note

Automatic bit-rate detection

.

Version: 1.0.0

01 January 2005

History

Date	Changes
2005-01-01	*Publication of version 1.0 as* application note

General information on licensing and patents

CAN in AUTOMATION (CiA) calls attention to the possibility that some of the elements of this CiA application note may be subject of patent rights. CiA shall not be responsible for identifying any or all such patent rights.

Because this application note is licensed free of charge, there is no warranty for this application note, to the extent permitted by applicable law. Except when otherwise stated in writing the copyright holder and/or other parties provide this application note "as is" without warranty of any kind, either expressed or implied, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose. The entire risk as to the correctness and completeness of the specification is with you. Should this application note prove failures, you assume the cost of all necessary servicing, repair or correction.

Trademarks

CANopen® and CiA® are registered community trademarks of CAN in Automation. The use is restricted for CiA members or owners of CANopen vendor ID. More detailed terms for the use are available from CiA.

CAN in Automation e. V.
Kontumazgarten 3
DE - 90429 Nuremberg, Germany
Tel.: +49-911-928819-0
Fax: +49-911-928819-79
Url: www.can-cia.org
Email: headquarters@can-cia.org

Contents

1 Scope

This technical report describes the recommended practice and gives application hints for implementing automatic bit-rate detection in CANopen devices. With the Layer Setting Services (LSS) it is possible to change the bit-rate in CANopen networks. However, this mechanism fails in certain situations. Some low-cost devices that do not support LSS at all could also benefit from the recommended automatic bit-rate detection method. This technical report discusses an approach for automatic bit-rate detection in CANopen networks. As introduction the possible solutions to detect an unknown bit-rate for CAN controllers (Software / Hardware) are presented. The technical report will focus on situations where automatic bit-rate detection fails (no traffic on the bus, error frames) and how to avoid these deadlocks.

The primary objective of this application note is to recommend a solution for automatic bit-rate detection on any CANopen device. A motivating requirement for the solution is for trouble-free operation of LSS-only devices and to avoid deadlock situations. The demands are:

- LSS-only devices can be configured within a network (no point-to-point connection necessary)
- A bus-off situation is handled autonomously
- LSS-only devices can run with CANopen modules that do not support LSS
- All defined CANopen bit-rates shall be supported
- Error frames, caused by wrong bit-rates, shall be avoided
- No specific identifier or data field for the CAN message is required

2 References

2.1 Normative references

/CiA301/	CiA DS 301, CANopen application layer and communication profile
/CiA305/	CiA DS 305, CANopen layer setting services and protocol
/ISO11898-1/	ISO 11898-1:2003, Road vehicles - Controller area network (CAN) – Part 1: Data link layer and physical signaling

2.2 Informative references

/AP292501/	Tobias Wenzel: CAN baudrate detection with Infineon CAN devices. AP 292501, Infineon.

3 Abbreviations and definitions

3.1 Abbreviations

CAN	Controller area network
CAN-ID	CAN identifier
COB	Communication object
CRC	Cyclic redundancy check
LMT	Layer management
LSS	Layer setting services and protocols
NMT	Network management

3.2 Definitions

The definitions given in /CiA301/, /CiA305/ and /ISO11898-1/ apply to this application note, too.

4 General introduction

CANopen devices which do not have a hardware interface (e.g. DIP-switches, RS-232) to setup the CAN bit-rate and the node address use the LSS protocol for that purpose. LSS is described in /CiA305/. Two identifiers are used for communication between the LSS master (CAN-ID: $7E5_h$) and the LSS slave (CAN-ID: $7E4_h$). In order to identify a certain CANopen node, the LSS address of that node is used. The LSS address consists of four 32-bit keys (vendor-, product-, revision- and serial-number), so that every CANopen device has an unique 128-bit key for identification. By means of the “configure bit timing parameters” service the LSS master sets the new bit timing on a LSS slave. For CANopen devices, the following bit timings are defined:

Table 1: CANopen bit-rates

Bit-rate	***Sample point location***	***Range***
1,000 kbit/s	87.5%	75.0% - 90.0%
800 kbit/s	87.5%	75.0% - 90.0%
500 kbit/s	87.5%	85.0% - 90.0%
250 kbit/s	87.5%	85.0% - 90.0%
125 kbit/s	87.5%	85.0% - 90.0%
50 kbit/s	87.5%	85.0% - 90.0%
20 kbit/s	87.5%	85.0% - 90.0%
10 kbit/s	87.5%	85.0% - 90.0%

Configuring a wrong bit-rate for a CANopen device is a critical issue. Depending on the data stream stuff-bit, CRC, form or acknowledgement error occurs and the CANopen device will signal this by transmitting an error frame. The process of error detection and signaling stops, when the device with the incorrect bit-rate goes into bus-off state.

A mis-configured CANopen device that supports bit-rate setup only via LSS is in a deadlock situation now: it is not accessible via the network. The mis-configured device shall be removed from the network and the stored bit-rate shall be evaluated by “try-and-error”, using a point-to-point connection between a LSS master and the LSS slave.

In order to avoid the described deadlock situation two solutions seem reasonable:

- Use of a default fallback bit-rate
- Use of automatic bit-rate detection

5 Deadlock avoiding solutions

5.1 Default fallback bit-rate

The *default fallback bit-rate* approach may be implemented on any CAN controller by software: after detecting the bus-off state the device resumes with a low default bit-rate (e.g. 20 kbit/s). This simple solution has several drawbacks, however. First, all devices in the CANopen network shall go through the bus-off state and they all shall be configured for the correct bit-rate afterwards. Second, devices that do not support the *default fallback bit-rate* mechanism, will persist in the bus-off state. And finally, even if the reason for the bus-off condition was not raised by a mis-configured LSS device, the network will be forced to a “default” bit-rate. As this is not acceptable, this solution is not suitable and we have a closer look to automatic bit-rate detection.

5.2 Automatic bit-rate detection

5.2.1 General

In order to meet the requirement that a running CAN network is not disturbed by error frames, it is not suitable to “test” the reaction of the network by sending out messages on all possible bit-rates. The method for bit-rate detection should be non-reactive for the complete CAN network. The two possible methods are:

- Measuring the time of a single bit
- Suppressing transmission of error frames (listen-only mode)

5.2.2 Measuring the bit-time of a single bit

Measuring the time of a single bit does not require a CAN controller, but a good and fast timer module inside a microcontroller. The concept has already been described in /AP292501/. During the detection phase the CAN controller is switched off. The timer module measures in a consecutive process the length of a single bit inside the data stream. The main problem of the method is to find one single dominant bit – and only one – inside the frame. There is always one dominant bit inside a CAN frame which is surrounded by two recessive bits: the acknowledgement bit. But this is only true if there are at least two CAN nodes on the bus. Assuming only one active node (e.g. the CANopen manager is scanning the bus), this method requires a specified identifier field or data field, otherwise it will fail. Also the method requires a timer module, whereas the clock frequency should be higher than 10 MHz. For a clock frequency of 10 MHz the difference between 800 kbit/s and 1 Mbit/s is just 2 timer ticks (refer to table 2).

Table 2: Timer ticks for measuring a single dominant bit

f_{Timer} [MHz]	CAN bit-rate [kbit/s]	Expected timer ticks
10	500	20
10	800	12
10	1,000	10
24	500	48
24	800	30
24	1,000	24

Due to the drawback that at least two active nodes are required and the high demands on the timer module this approach does not seem suitable.

5.2.3 Analyzing in listen-only mode

The listen-only mode is a feature that has already been implemented in several CAN controllers[1]. In listen-only mode, the CAN controller only listens to the CAN receive line without acknowledging the received messages on the bus. It does not send any messages in this mode. However, the error flags are updated so that the bit timing may be adjusted until no error occurs.

The necessary software algorithm is shown in Figure 1. The CAN controller is initialized for acceptance of all messages (i.e. the global / local mask is set to 0). The bit timing values of the first possible CANopen bit-rate (10 kbit/s) is loaded and the controller is switched into “Listen-Only” mode. Assuming that there is traffic on the network and the bit-rate is correct, the message is accepted. The error registers will not change and the flag for message reception is set inside the CAN controller. This means the correct bit-rate has been detected.

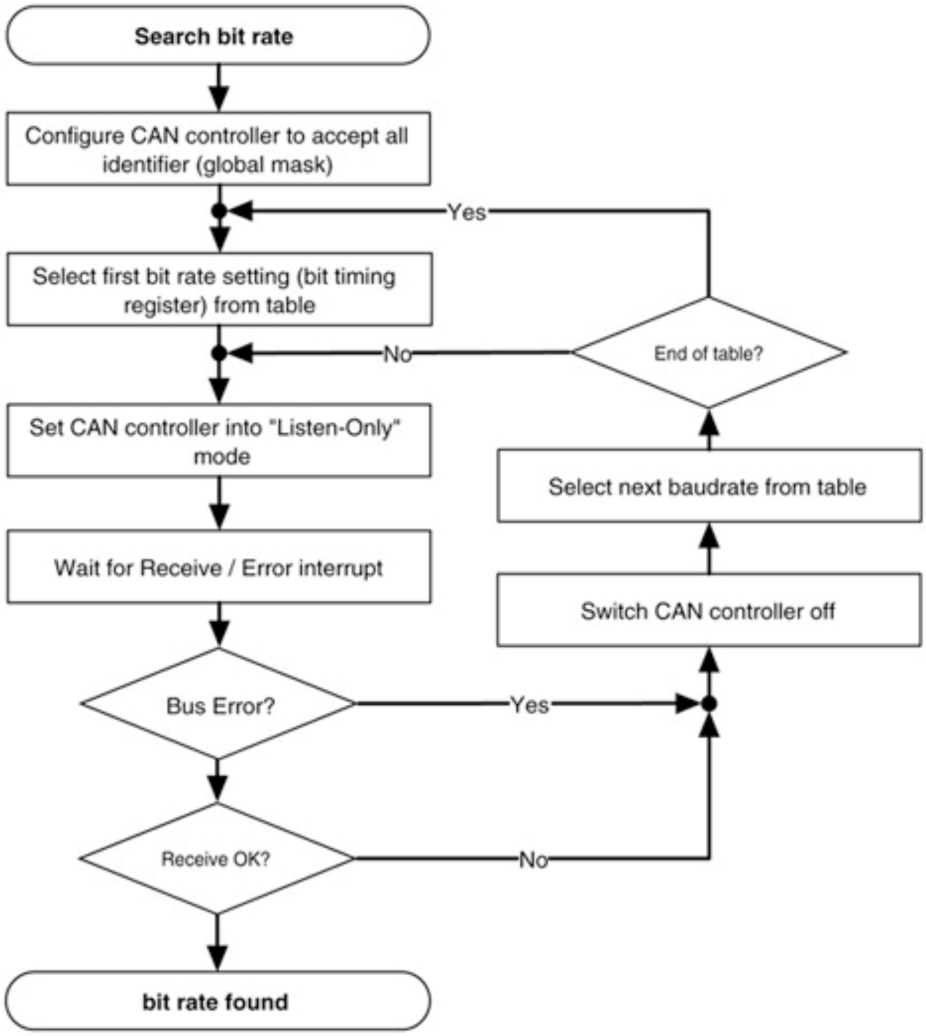

Figure 1: Algorithm for bit-rate detection in “Listen only” mode

Assuming the bit-rate is not correct, the error flags will be updated (stuff-, CRC or form-error). In this case the CAN controller is switched off and the next possible bit timing values are loaded from the bit-rate table.

The minimum number of required messages to find the correct bit-rate is eight (i.e. the number of defined CANopen bit-rates). However, it shall be taken into account that a delay between the CAN messages is required, as shown in Figure 2. The delay T_{cc} is the time to configure the CAN controller (switch CAN controller off, select next bit-rate, set CAN controller into “Listen-Only” mode). Depending on the interframe space and the time to re-configure the CAN controller, the number of required CAN frames may be higher. In addition to that, a CAN frame might be corrupted when the detection on the correct bit-rate is in progress. This means the algorithm will exit in the „bus error“ branch. One

[1] e.g. the SJA1000 (Philips), the TwinCAN (Infineon), or the CANary family (Atmel)

improvement of the algorithm is to check two consecutive times for a bus error, leading to a more stable behavior even in a disturbed environment.

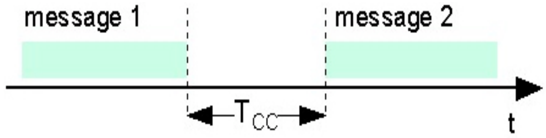

Figure 2: Delay between CAN messages

5.2.4 CAN controller not supporting listen-only mode

CAN controller without a built-in listen-only mode require an additional external hardware. The external hardware is shown in Figure 2 and consists of an AND-Gate and an OR-Gate.

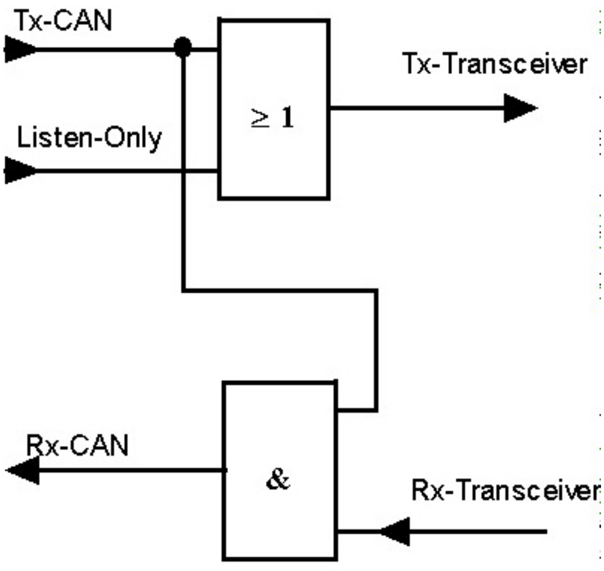

Figure 3: External hardware

For the given circuit it is assumed that the dominant bit is transmitted as a low level between the CAN controller and the CAN transceiver, a recessive bit is transmitted as an high level. Most CAN controllers use this setup as default. CAN controller with a configurable output stage[2] shall be configured in that way.

An additional output pin of the microcontroller is used to select between normal operation and "Listen-Only" mode. In "Listen-Only" mode the corresponding pin of the OR-gate is held on high level. The output of the OR-gate is tied to a high level, meaning that a recessive level is at the TX-input of the CAN transceiver all the time. But the CAN controller shall detect its own frames (acknowledgement, error frames) in order to handle the internal state machine correctly. This is achieved by feeding the Tx-CAN line into an AND-gate. The Tx-CAN line is AND-ed with the receive signal of the CAN transceiver. The result is shown in Table 3.

Table 3: Rx-CAN with additional hardware

Tx-CAN	*Rx-Transceiver*	*Rx-CAN*
Recessive	Recessive	Recessive
Recessive	Dominant	Dominant
Dominant	Recessive	Dominant
Dominant	Dominant	Dominant

[2] e.g. AN82527 (Intel)

By means of the AND-gate the CAN controller detects the frames on the bus and its own responses (e.g. acknowledgement). The circuit is switched in normal operation when the “Listen-only” pin is held on a low level. For detection of the unknown bit-rate the algorithm from Figure 1 is used.

6 Bus traffic requirements

In order to detect the unknown bit-rate, at least one "good" reception is required. The number of necessary frames on the bus depends on the configuration of the CANopen network, as shown in Figure 4.

Configuration 1 consists of a CANopen manager and devices with automatic bit-rate detection. When the CANopen manager starts to configure the network, none of the CANopen slaves sends an acknowledge. This means the very first message is repeated until the devices have found the correct bit-rate.

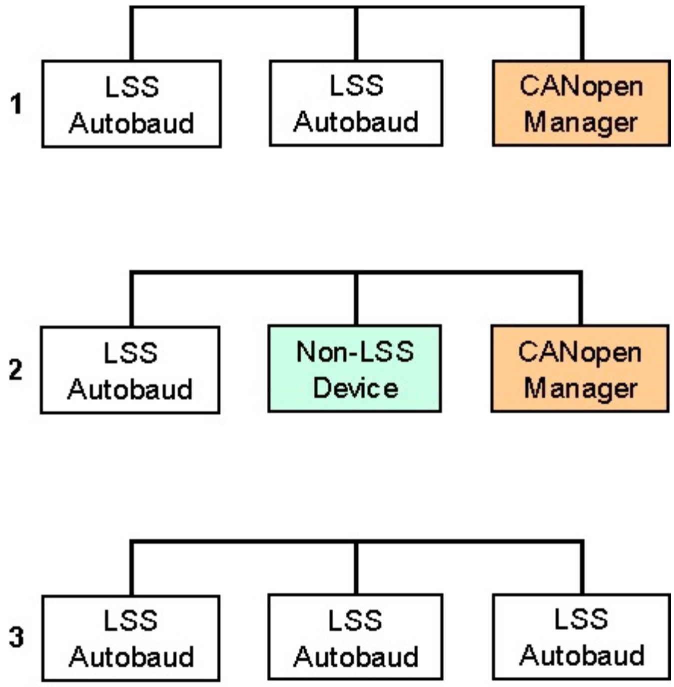

Figure 4: Possible network configurations

Configuration 2 consists of a CANopen manager and devices that support LSS as well as devices that do not have LSS at all (hardware setup of bit-rate). In that case the scanning algorithm of the CANopen manager may fail, depending on the bus traffic (unless the non-LSS device is capable of, and configured for, automatic bit-rate detection). As we have already seen, at least 8 CAN messages are required to find the correct bit-rate. The LSS automatic bit devices will send the Boot-up message after the correct bit-rate has been detected. This is the signal for the CANopen manager to configure the LSS automatic bit devices.

Configuration 3 consists of CANopen slave devices with LSS Automatic bit detection only, whereas one device is a "Mini-Master". The "Mini-Master" functionality is configured via the object $1F80_h$ (NMT startup) in the object dictionary. By this object a CANopen device may enter the *operational state* autonomously and it may start other nodes with the NMT start command. A CANopen device that is configured in that way may not use automatic bit-rate detection, because the network will not start at all.

There are still some open questions: what kind of messages (CANopen services) are used to produce traffic? And how many messages are required to setup the correct bit-rate?

Using a CANopen master, there are three possible services that may be used to produce a constant traffic that does not influence the network behavior.

- Heartbeat: the heartbeat message of the master (if implemented) is used for traffic generation.

- SDO request: the SDO manager sends requests to the indices 1000_h, 1001_h and 1018_h.

- SYNC: the CANopen master sends SYNC messages

It is also possible to use any combination of these three methods. The total number of required CAN frames is 24 (i.e. 1 message lost during controller configuration, 2 messages for bit-rate identification, 8 possible bit-rates). Since many applications use the bit-rates 125 kbit/s to 500 kbit/s, it is a good choice to test these bit-rates first.

7 Impact of error frames

The automatic bit-rate detection as shown in Figure 1 expects correct message reception. If the CAN message is corrupted by any reason, a CAN error frame is sent. This behavior has an influence on the algorithm in Figure 1. Suppose that the CAN controller is configured on the correct bit-rate and waiting for a message. But the incoming message is corrupted, causing an error frame. Then the algorithm will select the next possible bit-rate from the table, thus expanding the required time to find the correct bit-rate.

8 Conclusion

The presented method for automatic bit-rate detection in CANopen networks makes the integration of devices without a hardware interface quite simple. As a result an additional entry to the bit-rate table in /CiA305/ will be added, that denotes “automatic detection” for an LSS device (*see* Table 4).

Table 4: Revised CiA bit timing table

Bit-rate	***Index***
1,000 kbit/s	0
800 kbit/s	1
500 kbit/s	2
250 kbit/s	3
125 kbit/s	4
reserved	5
50 kbit/s	6
20 kbit/s	7
10 kbit/s	8
Automatic bit-rate detection	9

Practical tests in automotive test stands have shown that devices with automatic bit-rate detection are easier to handle than those where the bit-rate setup is made via the LSS protocol. Manufacturers of CANopen devices with LSS support should provide the automatic bit-rate detection feature, since a better handling of those devices is achieved.

9 Acknowledgements

This technical report is based on the iCC paper *Automatic bit-rate detection in CANopen networks* presented by *Uwe Koppe*, and it has been discussed in the CANopen Interest Group's (IG) Task Force LSS.