



# AN1203

## Automatic start of CANopen slave devices

This application note explains the procedure to start the cyclic transmission of Process Data Objects (PDOs) automatically after power on.

---

## Contents

<b>1</b>	<b>Introduction</b> . . . . .	<b>3</b>
1.1	Example configuration . . . . .	3
<b>2</b>	<b>Device configuration</b> . . . . .	<b>4</b>
2.1	Identifier configuration . . . . .	4
2.2	Cyclic timer configuration . . . . .	4
2.3	Automatic startup configuration . . . . .	5
2.4	Storing the configuration . . . . .	5
<b>3</b>	<b>References</b> . . . . .	<b>6</b>
<b>4</b>	<b>Revision history</b> . . . . .	<b>7</b>

# 1 Introduction

For an increasing number of applications CANopen slave devices are used without a CANopen network manager. But even without the missing network management (NMT) messages the CANopen slave devices can transmit „Process Data Object“ (PDO) messages after power up, just by configuration of some parameters.

## 1.1 Example configuration

The following example explains the configuration of a CANopen slave device with the node address 127 (7F<sub>h</sub>).

With „Service Data Objects“ (SDO) the access to a device object dictionary is provided. By means of a SDO a peer-to-peer communication channel between two devices is established. The data length of a SDO message is fixed to eight bytes.

The values for the SDO identifiers are calculated according to the formula:

- ID (SDO-Tx) = 580<sub>h</sub> + node address (CAN message from device)
- ID (SDO-Rx) = 600<sub>h</sub> + node address (CAN message to device)

With the assumed address of 127 the SDO identifiers are:

- ID (SDO-Tx) = 5FF<sub>h</sub> (CAN message from device)
- ID (SDO-Rx) = 67F<sub>h</sub> (CAN message to device)

In the example configuration the device is programmed with a identifier value of 330<sub>h</sub> for the 1<sup>st</sup> transmit PDO, which is send with a 100ms period just after power up.

## 2 Device configuration

The device setup according to the requirements is done in four steps:

- configuration of PDO identifier
- configuration of PDO transmission type and transmission rate
- configuration of the autostart behaviour
- storing the data in the non-volatile memory of the device

### 2.1 Identifier configuration

The identifier for the 1<sup>st</sup> transmit PDO is configured via index 1800h, sub-index 1 of the communication profile. The device is set to pre-operational state first.

No	DIR	ID (hex)	DLC	Data (hex)	Comment
1	Tx	000	2	80 7F	Preop. node ID 127
2	Tx	67F	8	23 00 18 01 <b>30 03</b> 00 40	SDO write, 1800h, sub1
3	Rx	5FF	8	60 00 10 00 00 00 00 00	SDO response, OK

*Trace 1: Identifier setup*

The data bytes 5 and 6 of message no. 2 define the identifier (330<sub>h</sub> in this example). Please note that the value is transmitted LSB first.

### 2.2 Cyclic timer configuration

In the next step the 1<sup>st</sup> transmit PDO is configured as event triggered and the associated cyclic timer is programmed.

No	DIR	ID (hex)	DLC	Data (hex)	Comment
4	Tx	67F	8	2F 00 18 02 FE 00 00 00	SDO write, 1800h, sub2
5	Rx	5FF	8	60 00 18 02 00 00 00 00	SDO response, OK
6	Tx	67F	8	2B 00 18 05 <b>64 00</b> 00 00	SDO write, 1800h, sub5
7	Rx	5FF	8	60 00 18 05 00 00 00 00	SDO response, OK

*Trace 2: Cyclic timer setup*

The data bytes 5 and 6 of message no. 6 define the timer repetition rate in multiples of 1 millisecond (64<sub>h</sub> = 100 ms). Please note that the value is transmitted LSB first.

## 2.3 Automatic startup configuration

Now the device is configured to start PDO data transmission just after power up (no NMT start command required).

No	DIR	ID (hex)	DLC	Data (hex)	Comment
8	Tx	67F	8	23 80 1F 00 08 00 00 00	SDO write, 1F80h
9	Rx	5FF	8	60 80 1F 00 00 00 00 00	SDO response, OK

*Trace 3: NMT startup configuration*

## 2.4 Storing the configuration

In the final step the configuration is stored in the device.

No	DIR	ID (hex)	DLC	Data (hex)	Comment
10	Tx	67F	8	23 10 10 01 73 61 76 65	SDO write, 1010h
11	Rx	5FF	8	60 10 10 01 00 00 00 00	SDO response, OK

*Trace 4: Storing the configuration*

### **3**                    **References**

- /1/    CiA 301, CANopen Application Layer and Communication Profile, Version 4.2, CAN in Automation (CiA) e.V.,  
<http://www.can-cia.org>
  
- /2/    CiA 302-2, CANopen additional application layer functions - Part 2: Network management, Version V4.0, CAN in Automation (CiA) e.V.,  
<http://www.can-cia.org>

## 4 Revision history

<i>Revision</i>	<i>Date</i>	<i>Description</i>
01	28.10.2008	Initial version

### Disclaimers

Life support — Products and software described in this application note are not designed for use in life support appliances, devices, or systems where malfunction of these products can reasonably be expected to result in personal injury. MicroControl customers using or selling these products for use in such applications do so at their own risk and agree to fully indemnify MicroControl for any damages resulting from such application.

Right to make changes — MicroControl reserves the right to make changes in the products - including circuits and/or software - described or contained herein in order to improve design and/or performance. MicroControl assumes no responsibility or liability for the use of any of these products, conveys no licence or title under any patent, copyright, or mask work right to these products, and makes no representations or warranties that these products are free from patent, copyright, or mask work right infringement, unless otherwise specified.

### Copyright

No part of this application note may be copied, transmitted or stored in a retrieval system or reproduced in any way including, but not limited to, photography, magnetic, optic or other recording means, without prior written permission from MicroControl GmbH & Co. KG.

© 2008 MicroControl GmbH & Co. KG, Troisdorf

  
**MicroControl**  
Systemhaus für Automatisierung

MicroControl GmbH & Co. KG  
Lindlastr. 2c  
53842 Troisdorf  
Germany  
Fon: +49 / 2241 / 25 65 9 - 0  
Fax: +49 / 2241 / 25 65 9 - 11  
<http://www.microcontrol.net>