PROFIBUS DP

BC service channel

Version: **1.10 (April 2013)**
Model no.:

All information contained in this manual is current as of its creation/publication. B&R reserves the right to change the contents of this manual without notice. The information contained herein is believed to be accurate as of the date of publication; however, Bernecker + Rainer Industrie-Elektronik Ges.m.b.H. makes no warranty, expressed or implied, with regard to the products or documentation contained within this manual. In addition, Bernecker + Rainer Industrie-Elektronik Ges.m.b.H. shall not be liable for any incidental or consequential damages in connection with or arising from the furnishing, performance or use of the product(s) in this documentation. Software names, hardware names and trademarks are registered by their respective companies.

Chapter 1 • Introduction

The BC service channel can be used to access module data directly. This may include system information such as the serial number or firmware version or acyclic register data on I/O modules. Read and write access to acyclic I/O registers can be used to modify module parameters during program execution.

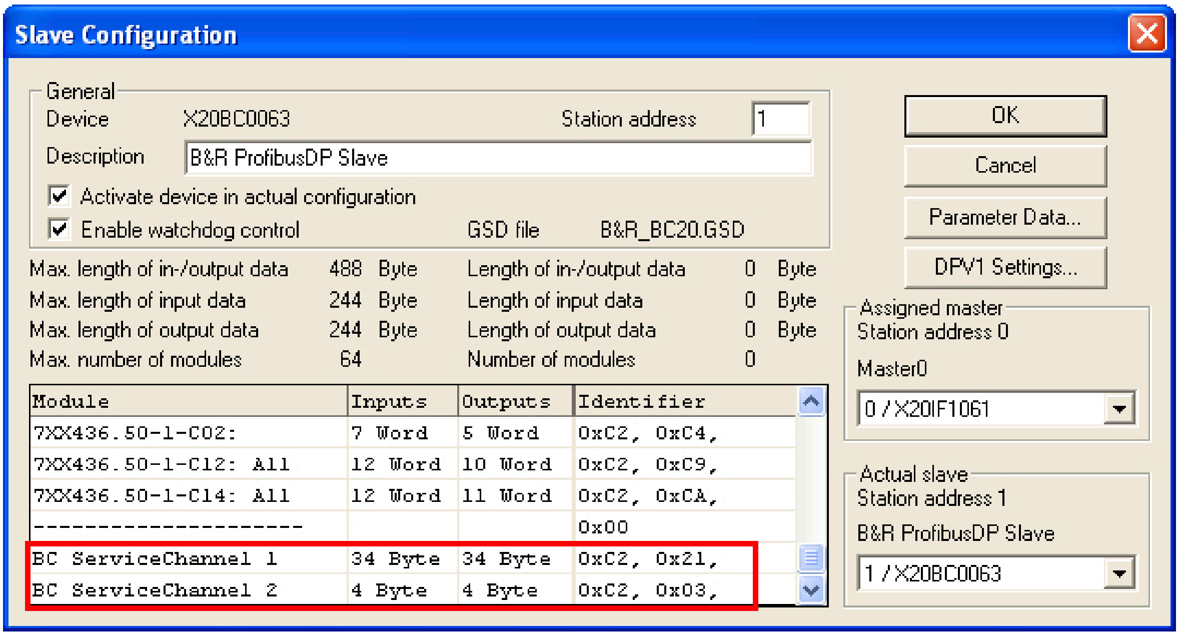
In order to perform this acyclic communication, the BC service channel reserves a number of bytes in the cyclic I/O data. The required number can be defined in the PROFIBUS DP master development environment by inserting virtual modules as placeholders in the I/O module configuration.

The size of the BC service channel can be adapted to accommodate the amount of data that needs to be transferred. This is done by combining as many virtual modules as necessary.

Information:

BC service channel modules must always be inserted after the last I/O module in the configuration!

GSD file entries for two types of virtual BC service channel modules:



Device X20BC0063 Station address 1

Description B&R ProfibusDP Slave

☑ Activate device in actual configuration

☑ Enable watchdog control GSD file B&R_BC20.GSD

Max. length of in-/output data	488	Byte
Max. length of input data	244	Byte
Max. length of output data	244	Byte
Max. number of modules	64	

Length of in-/output data	0	Byte
Length of input data	0	Byte
Length of output data	0	Byte
Number of modules	0	

Assigned master: Station address 0, Master0, 0 / X20IF1061

Actual slave: Station address 1, B&R ProfibusDP Slave, 1 / X20BC0063

Module	Inputs	Outputs	Identifier
7XX436.50-1-C02:	7 Word	5 Word	0xC2, 0xC4,
7XX436.50-1-C12: All	12 Word	10 Word	0xC2, 0xC9,
7XX436.50-1-C14: All	12 Word	11 Word	0xC2, 0xCA,
---------------------			0x00
BC ServiceChannel 1	34 Byte	34 Byte	0xC2, 0x21,
BC ServiceChannel 2	4 Byte	4 Byte	0xC2, 0x03,

Figure 1: Inserting virtual service channel modules as placeholders

Example

11 bytes are required for acyclic access to an I/O register. The BC service channel should be configured so that it is able to transfer this number within one bus cycle.

Inserting 3 virtual modules with the type "BC Service Channel 2" gives us 3 * 4 = 12 bytes for the transfer. One byte goes unused.

Information:

The bus controller can only handle requests individually. It is therefore not possible to configure the BC service channel to allow multiple requests per cycle.

1 BC service channel numbers

There are two BC service channels with the numbers 1 and 2. They have the following differences:

BC Service Channel 1	BC Service Channel 2
Should not be used in the application	Can be used in the application as often as needed
Used to transfer larger volumes of data	Used to access acyclic data
Smallest possible size: 34 bytes	Smallest possible size: 4 bytes
Poor utilization of the available bandwidth (many unused bytes)	Good utilization of the available bandwidth (no more than 3 unused bytes)

Generally, BC Service Channel 2 should be preferred.

Information:

BC service channels 1 and 2 must not be used together in the same application!

Chapter 2 • Overview of commands

The BC service channel protocol supports various commands for register access.

- Two commands for reading system information. (See Chapter 3 "Command 6 - System information" on page 7)
- One command for accessing the acyclic I/O module register. (See Chapter 4 "Command 7 - Asynchronous register access to I/O modules" on page 11)

Information:

All other command codes are reserved by the system for internal use and must not be used for register access.

1 Communication process

For an example of the communication process, see 4 "Examples" on page 10.

The first byte of the protocol frame is the command (C) that defines the frame's function.

Once the master system has been initialized, all output data is set to 0x00 by default. The first byte of the service request is the same as the first byte output on the BC service channel. As long as this byte is 0x00, the bus controller (PROFIBUS DP slave) returns received data (= BC service channel output data) back to the master (= BC service channel input data).

After the command (C), the second byte is the service cycle counter (SCC). (See 2 "Service cycle counter" on page 6)

When a new request is sent, the application needs to check the master system to see whether the sent data and received data are identical. This can be accomplished using the "memcmp" command, for example. As soon as this is confirmed, the first byte of output data (request frame) can be set to the respective command.

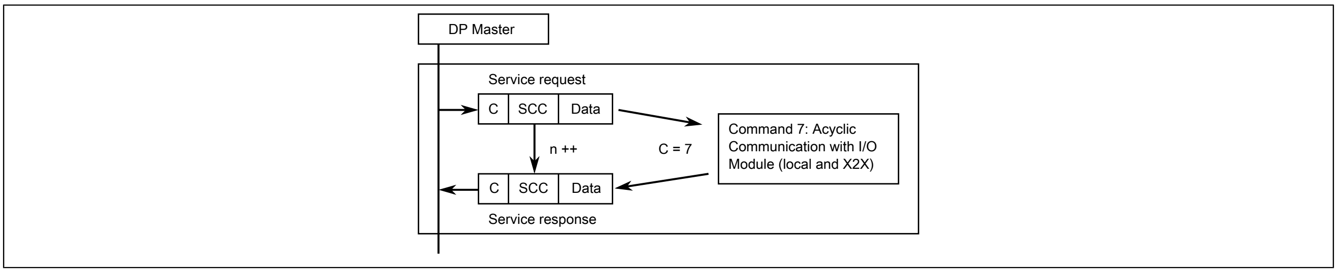
If the request sent was valid (e.g. Command 6), but the first byte of the response frame is 0x00, this means that another request is still being processed. Once the other request is finished processing, the first byte will be something other than 0x00. The application on the master system is responsible for verifying the received data by comparing the "Service cycle counter" values.

2 Service cycle counter

The service cycle counter (SCC) is always the second byte of the received data and is responsible for checking the validity of the received data.

Data can be considered valid and consistent when the following two conditions are met:

1 The command byte is not 0x00
2 The SCC is at least 1 higher than the SCC of the request



Chapter 3 • Command 6 - System information

Used to read system information from the bus controller or connected I/O modules.

1 Request frame

<table>
<tr><th>Byte offset</th><th>Description</th></tr>
<tr><td>0</td><td>Command (0x06)</td></tr>
<tr><td>1</td><td>I/O station number (decimal 0 to 63 or 0x00 to 0x3F)
<table>
<tr><td>0</td><td>The IP67 bus controller's integrated I/O module or the power supply module (X20PS94xx) on the IP20 bus controller (X20BC0063)</td></tr>
<tr><td>1</td><td>First I/O module after power supply module or IP67 bus controller</td></tr>
<tr><td>2</td><td>Second I/O module after power supply module or IP67 bus controller</td></tr>
<tr><td>3</td><td>...</td></tr>
</table></td></tr>
<tr><td>2, 3</td><td>Parameter number (Offset 2 = high byte, 3 = low byte)
1. Read system information from the bus controller.
When these parameter numbers are used, the I/O station number (byte offset 1) is ignored.
<table>
<tr><th>Dec.</th><th>Hex</th><th>Description</th></tr>
<tr><td>0</td><td>0x0</td><td>Current value of the node number switch (PROFIBUS station number)</td></tr>
<tr><td>1 - 20</td><td>0x1 - 0x14</td><td>Reserved</td></tr>
<tr><td>21</td><td>0x15</td><td>Active firmware version (see 4 "Examples" on page 10)</td></tr>
<tr><td>22</td><td>0x16</td><td>Boot block (flash block of active firmware)</td></tr>
<tr><td>23</td><td>0x17</td><td>Firmware version of boot block 0 (default firmware)</td></tr>
<tr><td>24</td><td>0x18</td><td>Firmware version of boot block 1 (update firmware)</td></tr>
<tr><td>25</td><td>0x19</td><td>Version of FPGA0 in flash memory</td></tr>
<tr><td>26</td><td>0x1A</td><td>Version of FPGA1 in flash memory</td></tr>
<tr><td>27</td><td>0x1B</td><td>Serial number (low word) (firmware V.1.28 and higher)</td></tr>
<tr><td>28</td><td>0x1C</td><td>Serial number (high word) (firmware V.1.28 and higher)</td></tr>
<tr><td>29</td><td>0x1D</td><td>Version of the active FPGA (firmware V.1.29 and higher)</td></tr>
<tr><td>30</td><td>0x1E</td><td>Hardware version (firmware V.1.40 and higher)</td></tr>
</table>
2. Read parameters from the I/O module.
When these parameter numbers are used, the desired module is determined by the I/O station number (byte offset 1).
<table>
<tr><th>Dec.</th><th>Hex</th><th>Description</th></tr>
<tr><td>8000</td><td>0x1F 40</td><td>Operating mode (see 2 "Response frame" on page 8)</td></tr>
<tr><td>8001</td><td>0x1F 41</td><td>Hardware ID</td></tr>
<tr><td>8002</td><td>0x1F 42</td><td>Serial number (low word)</td></tr>
<tr><td>8003</td><td>0x1F 43</td><td>Serial number (high word)</td></tr>
<tr><td>8004</td><td>0x1F 44</td><td>I/O module firmware version</td></tr>
<tr><td>8500</td><td>0x21 34</td><td>Module info string (> 2 bytes of data)</td></tr>
</table></td></tr>
</table>

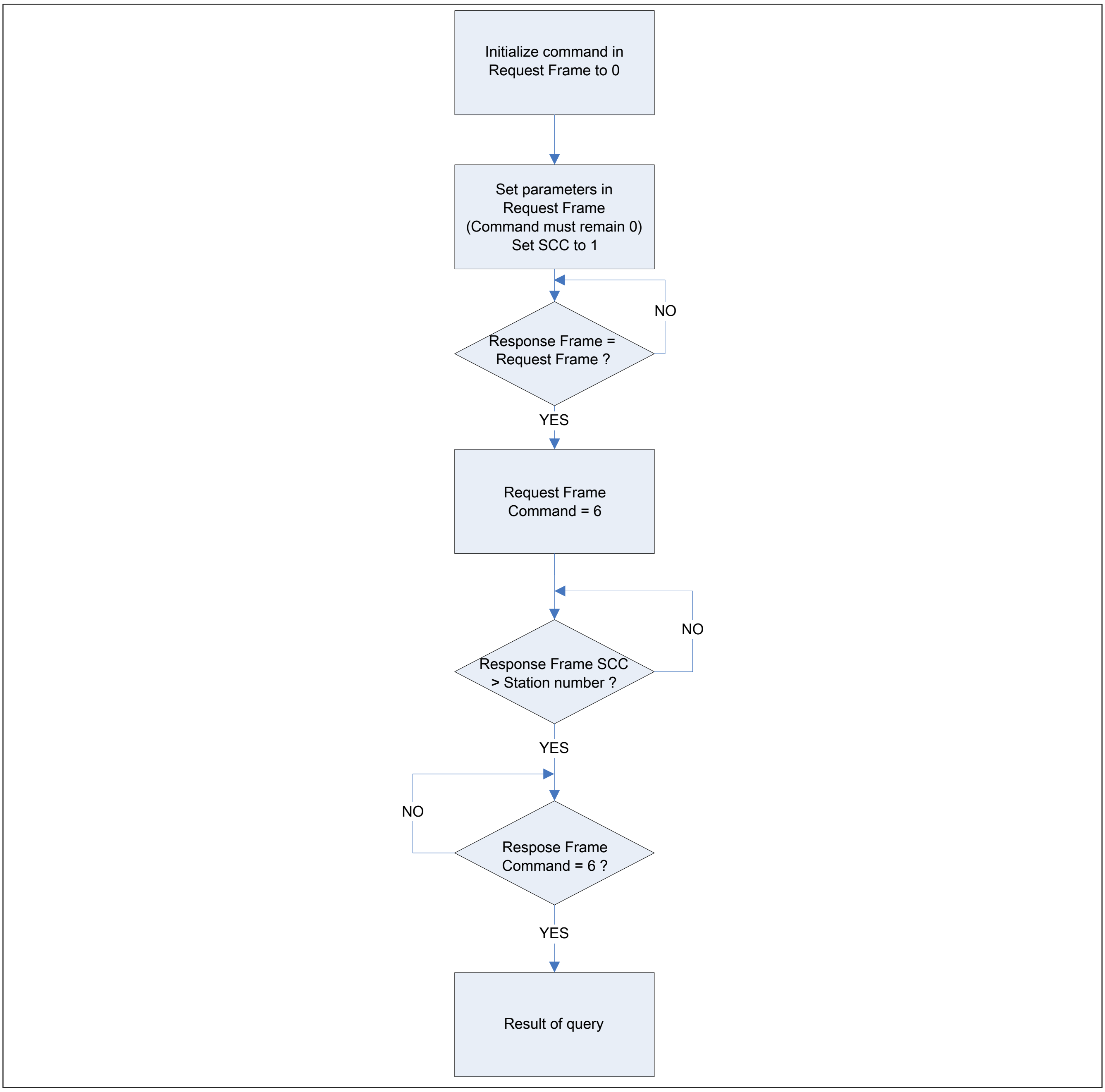
2 Response frame

Byte offset	Description
0	Command (0x00, 0x06 or 0x63) 0x00 Waiting/Busy 0x06 Request completed successfully
1	Service cycle counter (see 2 "Service cycle counter" on page 6)
2, 3	Parameter value (Offset 2 = high byte, 3 = low byte)

Possible response parameter values for the I/O module request "operating mode"

Value	Description
0x00 '0'	No X2X Link station found for this slot
0x42 'B'	Boot process (The boot loader tests whether the firmware/OS can be loaded)
0x43 'C'	Module being configured
0x44 'D'	Firmware download active
0x4E 'N'	X2X station found, but the I/O module can't be started **Cause**: No I/O module power supply, or the module is not connected to the X2X ASIC module
0x50 'P'	Preoperational (module ready for "Run mode")
0x52 'R'	Run mode (Operational)
0x55 'U'	Boot process (uploading IDs)
0x70 'P'	Preoperational (module ready for "Run mode")
0xE0	Error: Module firmware missing
0xE1	Error: Module firmware invalid
0xE2	Error: Configuration problem (module can't be activated) or possible configuration error (invalid function model, etc.)
0xE3	Error: Configuration problem (register can't be mapped, possible configuration error, etc.)
0xE4	Error: Module won't boot (internal error)
0xE5	Error: Module won't boot (X2X frame too small. **Solution:** increase X2X cycle time if necessary)
0xE6	Error: Module won't boot (incorrect module type configured for slot)

3 Flow chart - System information query


Initialize command in Request Frame to 0
Set parameters in Request Frame (Command must remain 0) Set SCC to 1
NO
Response Frame = Request Frame ?
YES
Request Frame Command = 6
NO
Response Frame SCC > Station number ?
YES
NO
Respose Frame Command = 6 ?
YES
Result of query

4 Examples

Reading the firmware version of the bus controller

Steps	Schedule	Request / Response	Byte offset 0 (Command)	Byte offset 1	Byte offset 2	Byte offset 3
1		Request	0x00	XX[1]	0	0x15
2		Response	0x00	n[2]	0	0x15
3	Request → Response	Request	0x06	XX[1]	0	0x15
4	Response SCC[3] <> n[2]	Response	0x06	n++[4]	0	YY[5]

[1] Any byte (ignored)
[2] Current value of the service cycle counter
[3] Service cycle counter
[4] Service cycle counter value increased by 1
[5] Firmware version of the bus controller (e.g. 0x83 = dec. 131 → V1.31)

Reading the Hardware ID of I/O Module 5

Module 5 is the fifth module after the power supply for the IP20 bus controller X20BC0063. With the IP67 version it is the fifth module after the bus controller. In this case, Module 0 is the I/O module integrated in the bus controller itself.

Parameter number 8001 (Hardware ID) is 0x1F41.

Steps	Schedule	Request / Response	Byte offset 0 (Command)	Byte offset 1	Byte offset 2	Byte offset 3
1		Request	0x00	0x05	0x1F	0x41
2		Response	0x00	n[1]	0x1F	0x41
3	Request → Response	Request	0x06	0x05	0x1F	0x41
4	Response SCC[2] <> n[1]	Response	0x06	n++[3]	0	ZZ[4]

[1] Current value of the service cycle counter
[2] Service cycle counter
[3] Service cycle counter value increased by 1
[4] Hardware ID of Module 5 (e.g. 0x1B45 = dec. 7077)

As part of the serial number, the Hardware ID is printed on the electronic module. The Hardware ID is the first four digits of the serial number.

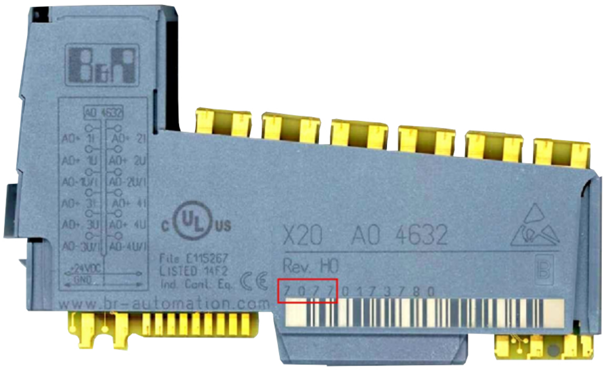


Figure 2: Example of module label

Information:

IDs higher than 9999 are shown as hexadecimal numbers.

Chapter 4 • Command 7 - Asynchronous register access to I/O modules

Chapter 4
Command 7 - Asynchronous register access to I/O modules

This command is used to send and receive asynchronous I/O registers in the application.

1 Request frame

Byte offset	Description
0	Command (0x07)
1	0x01 = Wait for RX data (always TRUE)
2	I/O station number (decimal 0 to 63 or 0x00 to 0x3F) 0 The IP67 bus controller's integrated I/O module or the power supply module (X20PS94xx) on the IP20 bus controller (X20BC0063) 1 First module after power supply or IP67 bus controller 2 Second module after power supply or IP67 bus controller 3 ...
3	Channel = 0x20 (number of the asynchronous X2X channel)
4	Transfer size = 0x06 (2 bytes for the register address and 4 bytes for the value. See 3 "Sent data" on page 11)
5 - x	Transfer data (see 3 "Sent data" on page 11)

2 Response frame

Byte offset	Description
0	Command (0x00, 0x07 or 0x63) 0x00 Waiting/Busy 0x07 Request completed successfully 0x63 An error has occurred (see Chapter 5 "Error messages" on page 16)
1	Service cycle counter (see 2 "Service cycle counter" on page 6)
2	I/O station number (decimal 0 to 63 or 0x00 to 0x3F) 0 The IP67 bus controller's integrated I/O module or the power supply module (X20PS94xx) on the IP20 bus controller (X20BC0063) 1 First module after power supply or IP67 bus controller 2 Second module after power supply or IP67 bus controller 3 ...
3	Channel (see 4 "Received data " on page 12 for values other than 0x20)
4	Received size
5 - x	Received data (see 4 "Received data " on page 12)

3 Sent data

<table>
<tr><th>Byte offset</th><th>Description</th></tr>
<tr><td>5</td><td>Register address high byte and and "<b>access bit</b>" (Bit 6 = TRUE for write access, FALSE for read access)
<table>
<tr><th>Bit 7</th><th>Bit 6</th><th>Bit 5</th><th>Bit 4</th><th>Bit 3</th><th>Bit 2</th><th>Bit 1</th><th>Bit 0</th></tr>
<tr><td>Reserved</td><td><b>Access bit</b></td><td>Address bit 13</td><td>Address bit 12</td><td>Address bit 11</td><td>Address bit 10</td><td>Address bit 9</td><td>Address bit 8</td></tr>
</table></td></tr>
<tr><td>6</td><td>Register address low byte</td></tr>
<tr><td>7</td><td>Configuration value of LBLW (low byte, low word)</td></tr>
<tr><td>8</td><td>Configuration value of HBLW (high byte, low word)</td></tr>
<tr><td>9</td><td>Configuration value of LBHW (low byte, high word)</td></tr>
<tr><td>10</td><td>Configuration value of HBHW (high byte, high word)</td></tr>
</table>

For write access to acyclic I/O registers, the access bit (Bit 6) of Byte 5 must be set to "1".

Example

The channel type of theX20AI4622 module in Register 18 (0x0012) should be addressed for write access.

Byte 5 = 0x40 (instead of 0x00)
Byte 6 = 0x12

4 Received data

<table>
<tr><th>Byte offset</th><th>Description</th></tr>
<tr><td>5</td><td>Register address high byte, access and error bits
<table>
<tr><th>Bit 7</th><th>Bit 6</th><th>Bit 5</th><th>Bit 4</th><th>Bit 3</th><th>Bit2</th><th>Bit 1</th><th>Bit 0</th></tr>
<tr><td>Error bit</td><td><b>Access bit</b></td><td>Address bit 13</td><td>Address bit 12</td><td>Address bit 11</td><td>Address bit 10</td><td>Address bit 9</td><td>Address bit 8</td></tr>
</table></td></tr>
<tr><td>6</td><td>Register address low byte</td></tr>
<tr><td>7</td><td>Register value of LBLW (low byte, low word)</td></tr>
<tr><td>8</td><td>Register value of HBLW (high byte, low word)</td></tr>
<tr><td>9</td><td>Register value of LBHW (low byte, high word)</td></tr>
<tr><td>10</td><td>Register value of HBHW (high byte, high word)</td></tr>
</table>

When read/write access is successful, bytes 5 and 6 are set to the same values as the corresponding bytes in the request frame. When the access bit in the response frame is TRUE, this indicates that write access has been performed.

When the error bit is TRUE, this indicates that the desired register address in the module can't be accessed.

Errors can therefore easily be identified by comparing bytes 5 and 6 in the response frame.

Example

An attempt to read Register **0x0001** in the power supply module X20PS9400, which doesn't exist in the firmware. (The I/O station number is 0x00)

Request frame:

Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7	Byte 8	Byte 9	Byte 10	Byte 11
0x07	0x01	0x00	0x20	0x06	**0x00**	**0x01**	XX[1)]	XX[1)]	XX[1)]	XX[1)]	0x00[2)]

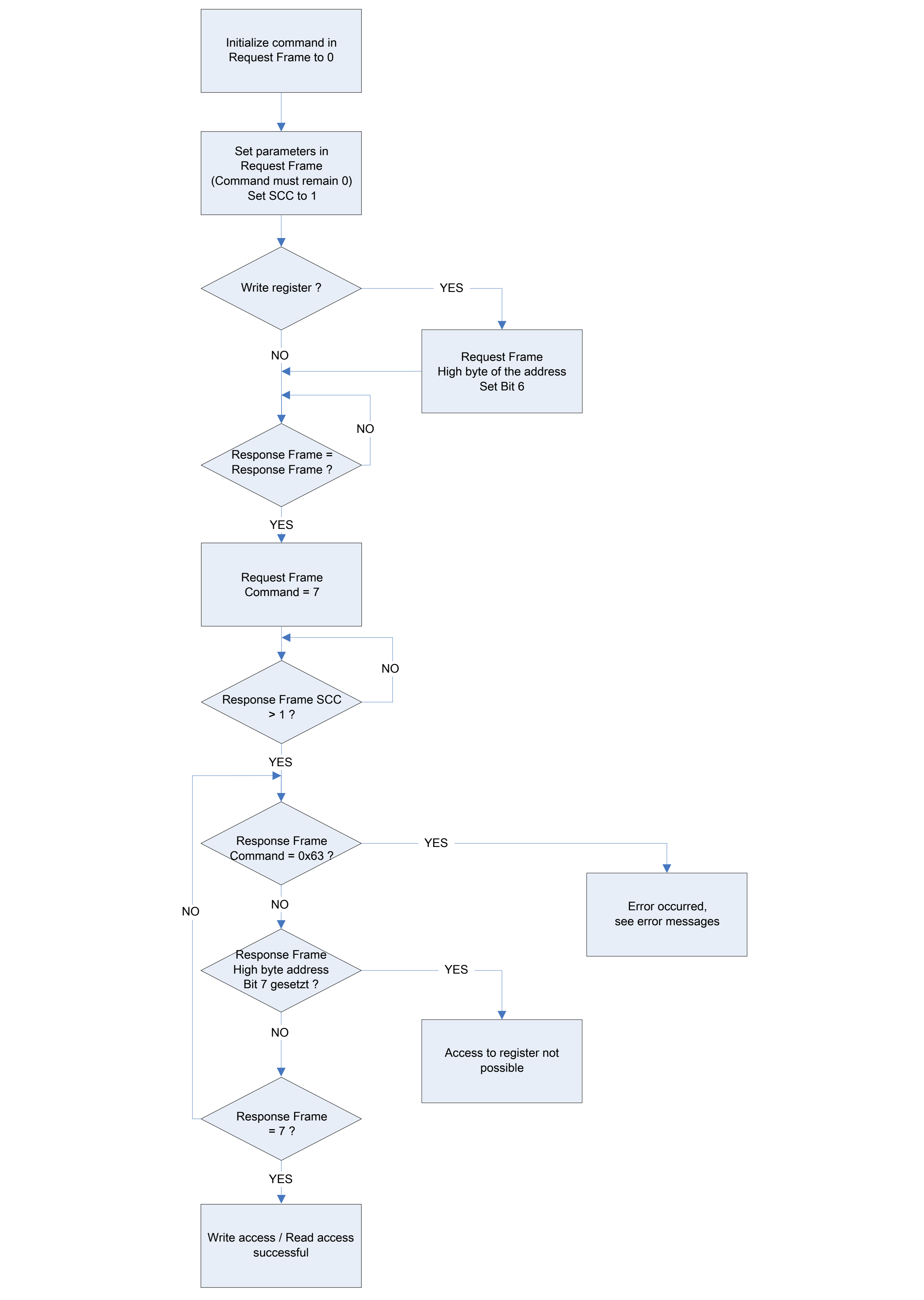
[1)] XX any byte (ignored)
[2)] Filler byte

Response frame:

Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7	Byte 8	Byte 9	Byte 10	Byte 11
0x07	SCC[1)]	0x00	0x20	0x06	**0x80**	**0x01**	0x00	0x00	0x00	0x00	0x00[2)]

[1)] Service cycle counter
[2)] Filler byte

5 Flow chart - Register access


Initialize command in Request Frame to 0
Set parameters in Request Frame (Command must remain 0) Set SCC to 1
Write register ?
YES
Request Frame High byte of the address Set Bit 6
NO
NO
Response Frame = Response Frame ?
YES
Request Frame Command = 7
NO
Response Frame SCC > 1 ?
YES
Response Frame Command = 0x63 ?
YES
Error occurred, see error messages
NO
NO
Response Frame High byte address Bit 7 gesetzt ?
YES
Access to register not possible
NO
Response Frame = 7 ?
YES
Write access / Read access successful

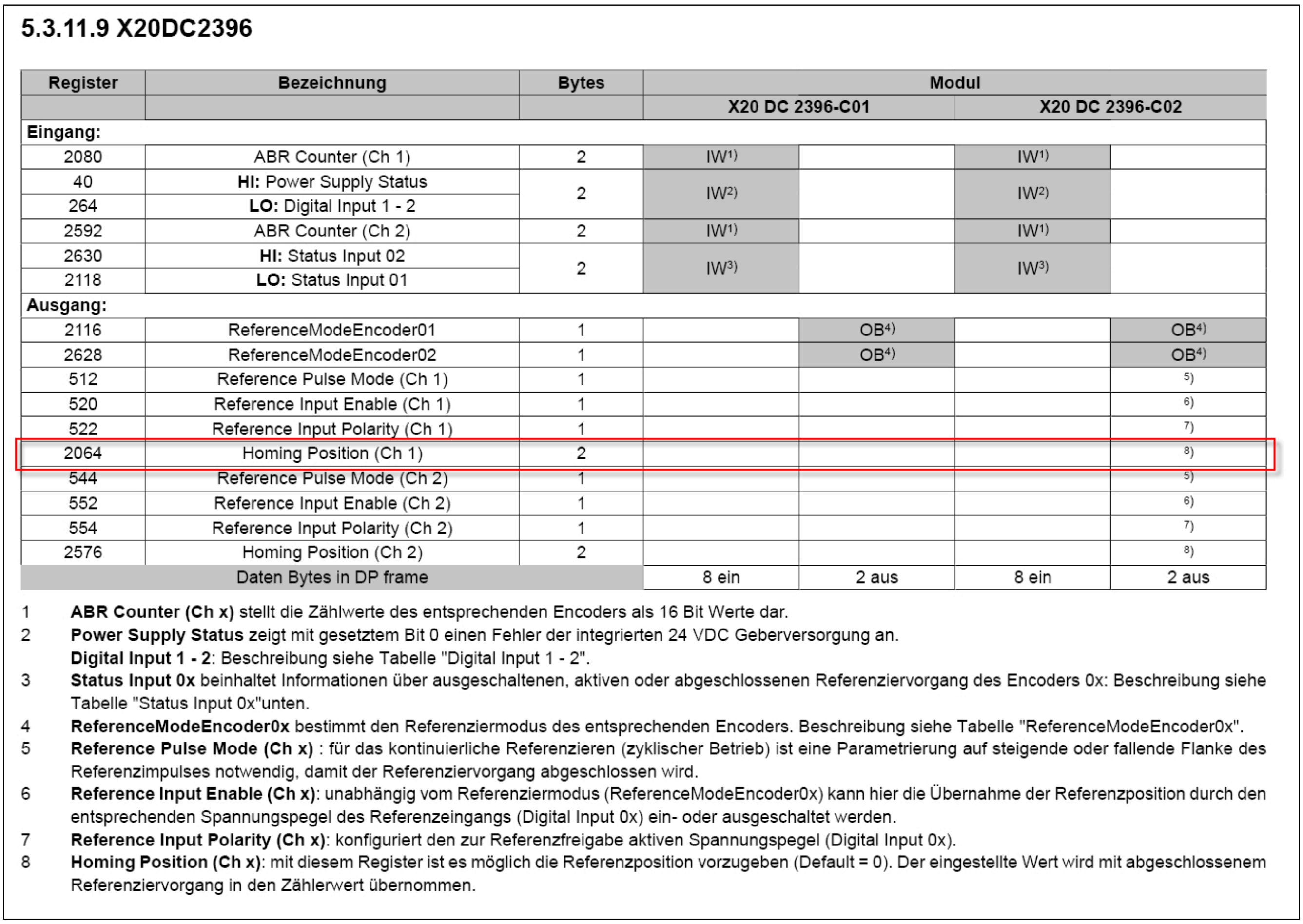
6 Example

Set the "Homing position (CH 1)" of the X20DC2396 module.

Hardware used:

- X20BC0063
- X20PS9400
- X20DC2396

The register number can be found in the module's register description. This is included in the "PROFIBUS User's Manual" and can be downloaded from the B&R website (www.br-automation.com).



5.3.11.9 X20DC2396

Register	Bezeichnung	Bytes	Modul			
			X20 DC 2396-C01		X20 DC 2396-C02	
Eingang:						
2080	ABR Counter (Ch 1)	2	IW1)		IW1)	
40	**HI:** Power Supply Status	2	IW2)		IW2)	
264	**LO:** Digital Input 1 - 2					
2592	ABR Counter (Ch 2)	2	IW1)		IW1)	
2630	**HI:** Status Input 02	2	IW3)		IW3)	
2118	**LO:** Status Input 01					
Ausgang:						
2116	ReferenceModeEncoder01	1		OB4)		OB4)
2628	ReferenceModeEncoder02	1		OB4)		OB4)
512	Reference Pulse Mode (Ch 1)	1				5)
520	Reference Input Enable (Ch 1)	1				6)
522	Reference Input Polarity (Ch 1)	1				7)
2064	Homing Position (Ch 1)	2				8)
544	Reference Pulse Mode (Ch 2)	1				5)
552	Reference Input Enable (Ch 2)	1				6)
554	Reference Input Polarity (Ch 2)	1				7)
2576	Homing Position (Ch 2)	2				8)
	Daten Bytes in DP frame		8 ein	2 aus	8 ein	2 aus

1 **ABR Counter (Ch x)** stellt die Zählwerte des entsprechenden Encoders als 16 Bit Werte dar.

2 **Power Supply Status** zeigt mit gesetztem Bit 0 einen Fehler der integrierten 24 VDC Geberversorgung an.
Digital Input 1 - 2: Beschreibung siehe Tabelle "Digital Input 1 - 2".

3 **Status Input 0x** beinhaltet Informationen über ausgeschaltenen, aktiven oder abgeschlossenen Referenziervorgang des Encoders 0x: Beschreibung siehe Tabelle "Status Input 0x"unten.

4 **ReferenceModeEncoder0x** bestimmt den Referenziermodus des entsprechenden Encoders. Beschreibung siehe Tabelle "ReferenceModeEncoder0x".

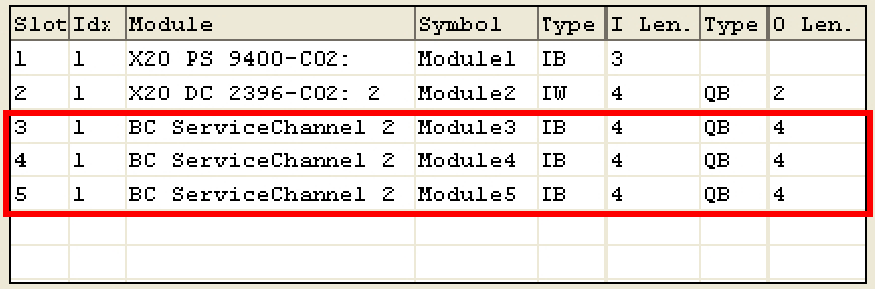
5 **Reference Pulse Mode (Ch x)** : für das kontinuierliche Referenzieren (zyklischer Betrieb) ist eine Parametrierung auf steigende oder fallende Flanke des Referenzimpulses notwendig, damit der Referenziervorgang abgeschlossen wird.

6 **Reference Input Enable (Ch x)**: unabhängig vom Referenziermodus (ReferenceModeEncoder0x) kann hier die Übernahme der Referenzposition durch den entsprechenden Spannungspegel des Referenzeingangs (Digital Input 0x) ein- oder ausgeschaltet werden.

7 **Reference Input Polarity (Ch x)**: konfiguriert den zur Referenzfreigabe aktiven Spannungspegel (Digital Input 0x).

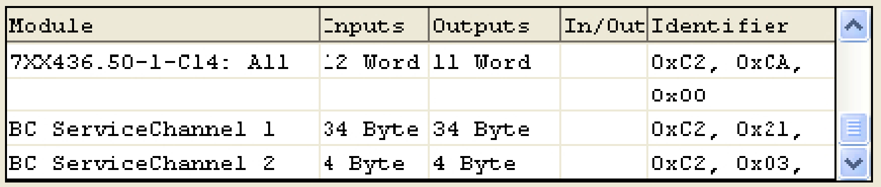
8 **Homing Position (Ch x)**: mit diesem Register ist es möglich die Referenzposition vorzugeben (Default = 0). Der eingestellte Wert wird mit abgeschlossenem Referenziervorgang in den Zählerwert übernommen.

Insert three virtual BC ServiceChannel modules as cyclic data in the development environment.



Slot	Idx	Module	Symbol	Type	I Len.	Type	O Len.
1	1	X20 PS 9400-C02:	Module1	IB	3		
2	1	X20 DC 2396-C02: 2	Module2	IW	4	QB	2
3	1	BC ServiceChannel 2	Module3	IB	4	QB	4
4	1	BC ServiceChannel 2	Module4	IB	4	QB	4
5	1	BC ServiceChannel 2	Module5	IB	4	QB	4

11 bytes of data are needed in order to write asynchronous I/O registers. It is therefore recommended to create three instances of the second entry "BC ServiceChannel 2" → 3 * 4 = 12 bytes. This minimizes the amount of I/O data on the PROFIBUS network, since "BC ServiceChannel 1" occupies 34 bytes of input and output data.



Module	Inputs	Outputs	In/Out	Identifier
7XX436.50-1-C14: All	12 Word	11 Word		0xC2, 0xCA, 0x00
BC ServiceChannel 1	34 Byte	34 Byte		0xC2, 0x21,
BC ServiceChannel 2	4 Byte	4 Byte		0xC2, 0x03,

The 11 byte structure is now written to the 12 bytes of the BC service channel. The last byte is not needed and goes unused.

Writing the register value

To set "Homing Position (Ch 1)" to 4660, the value 4660 (*0x1234*) needs to be written to Register 2046 (0x0810).

To write data, the MSB of the register address needs to make a logical OR operation with the access bit that has been set to TRUE. (0x0810 OR 0x4000 → **0x4810)**

In our example, the X20DC2396 module is the first module after the power supply, so the I/O station number is 0x01.

Information:

I/O station number (decimal 0 to 63 or 0x00 to 0x3F)

0	**The IP67 bus controller's integrated I/O module or the power supply module (X20PS94xx) on the IP20 bus controller (X20BC0063)**
1	**First I/O module after power supply module or IP67 bus controller**
2	**Second I/O module after power supply module or IP67 bus controller**
3	**...**

Request frame:

Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7	Byte 8	Byte 9	Byte 10	Byte 11
0x07	0x01	0x01	0x20	0x06	**0x48**	**0x10**	*0x34*	*0x12*	*0x00*	*0x00*	0x00

Response frame:

Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7	Byte 8	Byte 9	Byte 10	Byte 11
0x07	SCC[1]	0x01	0x20	0x06	**0x48**	**0x10**	*0x34*	*0x12*	*0x00*	*0x00*	0x00

[1] Service cycle counter

Reading the register value

Here, the "access bit" needs to be set to FALSE, so in this case the register address remains **0x0810**.

Request frame:

Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7	Byte 8	Byte 9	Byte 10	Byte 11
0x07	0x01	0x01	0x20	0x06	**0x08**	**0x10**	XX[1]	XX[1]	XX[1]	XX[1]	0x00

[1] Any bytes (ignored)

Response frame:

Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7	Byte 8	Byte 9	Byte 10	Byte 11
0x07	SCC[1]	0x01	0x20	0x06	**0x08**	**0x10**	*0x34*	*0x12*	*0x00*	*0x00*	0x00

[1] Service cycle counter

Evaluation of errors

- For errors in the response frame due to inaccessible registers, see 4 "Received data " on page 12
- For all other errors (Byte 0 in the response frame = 0x63), see Chapter 5 "Error messages" on page 16

Chapter 5 • Error messages

<table>
<tr><th>Byte offset</th><th>Description</th></tr>
<tr><td>0</td><td>0x63</td></tr>
<tr><td>1</td><td>Service cycle counter (see 2 "Service cycle counter" on page 6)</td></tr>
<tr><td>2, 3</td><td>Service error code
Offset: 2 = high byte, 3 = low byte
<table>
<tr><th colspan="2">Code</th><th rowspan="2">Description</th></tr>
<tr><th>Dec.</th><th>Hex</th></tr>
<tr><td>1</td><td>0x01</td><td>The configured DP buffers are too small for the access attempt</td></tr>
<tr><td>2</td><td>0x02</td><td>Incorrect checksum</td></tr>
<tr><td>3</td><td>0x03</td><td>Block number too high</td></tr>
<tr><td>4</td><td>0x04</td><td>Data read from flash memory doesn't match written data</td></tr>
<tr><td>5</td><td>0x05</td><td>Timeout while waiting for TX confirmation</td></tr>
<tr><td>6</td><td>0x06</td><td>Timeout while waiting for RX response from X2X Link</td></tr>
<tr><td>7</td><td>0x07</td><td>Timeout while waiting for write procedure in flash memory</td></tr>
<tr><td>8</td><td>0x08</td><td>NAK received, but requested module is not available</td></tr>
<tr><td>9</td><td>0x09</td><td>Acyclic communication is not possible in service mode</td></tr>
<tr><td>10</td><td>0x0A</td><td>Write/read command exceeds size of flash block</td></tr>
<tr><td>11</td><td>0x0B</td><td>Odd value entered for offset or invalid length</td></tr>
<tr><td>12</td><td>0x0C</td><td>Requested parameter number is invalid or doesn't exist</td></tr>
<tr><td>13</td><td>0x0D</td><td>I/O module download has size 0 or odd value. (not a multiple of a block size)</td></tr>
<tr><td>14</td><td>0x0E</td><td>Command not permitted while I/O manager is active</td></tr>
<tr><td>15</td><td>0x0F</td><td>File to large (Command 5)</td></tr>
<tr><td>16</td><td>0x10</td><td>Module has no SMB flash or is write protected</td></tr>
<tr><td>17</td><td>0x11</td><td>Write access not permitted to blocks with a size other than 16 bytes</td></tr>
<tr><td>18</td><td>0x12</td><td>Serial number can't be written if it is already unequal to 0xFFFFFFFF</td></tr>
</table>
</td></tr>
</table>