DELTA ELEKTRONIKA B.V.
DC POWER SUPPLIES

Vissersdijk 4, 4301 ND
Zierikzee, the Netherlands

www.DeltaPowerSupplies.com
Tel. +31 111 413656

IMPLEMENTATION GUIDE

Anybus® Carrier Interface

Firmware Update

It is strongly recommended to first perform a power-supply firmware update before further operation. Detailed instructions are provided in the Quick-Start Manual of the power supply.

Driver & Example Software

Drivers and example projects for a range of applications and interfaces can be downloaded from our website. **Products/Power Supply Series/Downloads**

INT-MOD-ANY
PSC-ANY-EXT

Document 202507-9
Firmware P0230 (SM15K)
Firmware P0170 (SM3300)
Firmware P0100 (PSC-ANY)

CONTENTS

1 Information on this document

1.1 Validity

This document is valid for the following products:

- INT-MOD-ANY (Pluggable Anybus interface module)
  - SM3300-series power supply with firmware version as stated on the cover or higher.
  - SM15K-series power supply with firmware version as stated on the cover or higher.
- PSC-ANY-EXT (External Anybus interface unit)
  - With firmware version as stated on the cover or higher.

1.2 Target Group

This equipment must be operated only by qualified personnel who understand the instructions and safety instructions provided with the equipment. If the equipment must be operated by unqualified personnel, then he/she must be supervised by qualified personnel.

1.3 Trademarks

Anybus® is a registered trademark of HMS Industrial Networks AB. All other trademarks mentioned in this document are the property of their respective holders.

1.4 Content and Structure

Please consider the safety instructions of all products involved before operating or connecting. This supplementary guide explains how to implement a supported Anybus module in combination with one of the Anybus carrier interfaces with a power supply into a specific fieldbus.

1.5 Additional information

The following additional product related documentation is available on our website. Please visit the website to find a list of the latest supplementary materials.

- INT-MOD-ANY: **Products > Interfaces > Anybus module > Downloads**
- PSC-ANY-EXT: **Products > Interfaces > Anybus unit (external) > Downloads**

Document name	Product	Type
Product Manual INT MOD ANY	INT-MOD-ANY	Manual
Python Example Code INT-MOD-ANY	INT-MOD-ANY	Software
CANopen EDS INT-MOD-ANY	INT-MOD-ANY	EDS file
EtherCAT ESI INT-MOD-ANY	INT-MOD-ANY	ESI file
Ethernet/IP EDS INT-MOD-ANY	INT-MOD-ANY	EDS file
PROFIBUS GSD INT-MOD-ANY	INT-MOD-ANY	GSD file
PROFINET GSDML INT-MOD-ANY	INT-MOD-ANY	GSDML file

Document name	Product	Type
Product Manual PSC ANY EXT	PSC-ANY-EXT	Manual

Please consult the website for the latest document versions and additional information.

1.6 Contact

In case of the need for additional guidance or in case of any remarks or feedback. Please visit our website and fill out one the **contact forms**. A structured technical contact form is available to enter descriptions and to upload files.

In case of malfunction or breakdown, please fill out the **RMA form** to get the product serviced.

2 General

2.1 Introduction

This document provides a guide in setting up a fieldbus with the INT-MOD-ANY and PSC-ANY-EXT products. The fieldbus implementation for a power supply using an Anybus module for a specific protocol is explained in three steps following the preparation in as described in Section 2.2. The three steps are illustrated in Figure 2.1

The first step is to get an overview of available commands and data available. A limited set of commands is available for communication through an Anybus module with a power supply. An overview of the available commands depending on the carrier-product and power supply series is shown in Section 2.4.

The second step is to choose a data format. Two data formats are available to send commands and retrieve data from the Anybus module connected to a power supply. One of either data formats must be chosen before implementing a fieldbus. Both data formats have their specific advantages and disadvantages. These two global data formats are discussed in Section 2.5.

Step three is implementing specific protocol mapping, based on the previously chosen data format. The protocol specific implementation is discussed in Chapter 3.

Be Aware

When toggling the data format, RSD will be set active. This is a safety precaution.

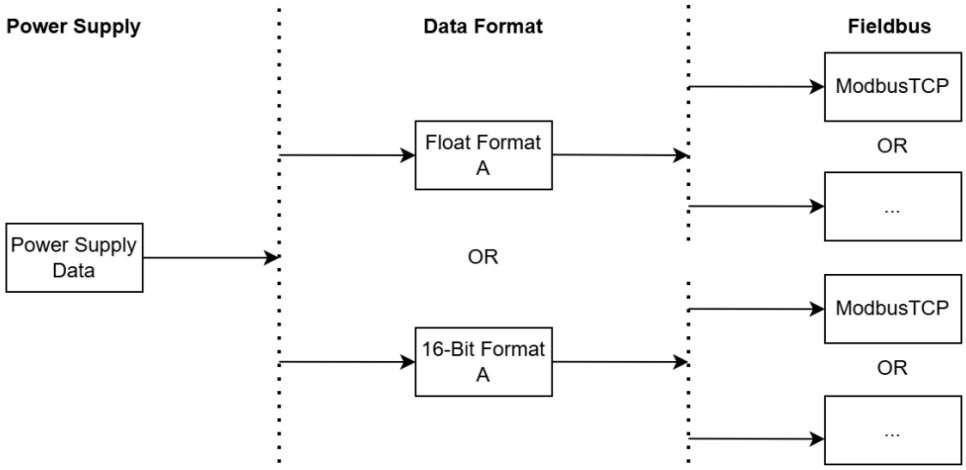

Figure 2.1: Flow of fieldbus implementation

2.2 Preparation

Please make sure to complete the setup of all products as explained in the product manuals before implementing a fieldbus using this document. The combination of power supply, INT-MOD-ANY or PSC-ANY-EXT and an Anybus module should be configured as far as it is explained in the product manuals.

Setup instructions can be found in each product manual, which can be downloaded from our website.

- INT MOD ANY: **Products > Interfaces > Anybus module > Downloads**
- PSC ANY EXT: **Products > Interfaces > Anybus unit (external) > Downloads**

The next step will be to implement a fieldbus using the equipment.

2.3 Example code

Several Python implementation examples are available for download on the website. These examples provide low threshold examples as reference for other fieldbus implementations. The examples can be found on the website through the same links as mentioned above for the product manuals in Section 2.2.

2.4 Power supply data

An overview of all commands using the INT-MOD-ANY or PSC-ANY-EXT and compatibility with a power supply series is shown in the table below, see Table 2.1. For reference, a table with command and status descriptions is listed in Appendix A1. The table below, Table 2.1, lists a Refresh Counter. This is a counter that will increment each time the parameters are updated. When the counter hits 65536, it returns to 0.

The registers mentioned in Table 2.1, are implemented as binary words of 16-bit length. Each bit in this word represents a binary status or setting. The position of each bit is denoted by a bit weight value. For example, the RSD bit in RemCTRL is represented by the first bit, position 0. Please note that the RemCTRL word writes multiple settings at once.

Compatible

Not compatible

Future update

Parameter	Bit Weight	R/W	INT-MOD-ANY		PSC-ANY-EXT				
			SM15K	SM3300	SM6000	SM1500	SM800	ES300	ES150
CVprg	-	R/W	■	■	■	■	■	■	■
CCprg		R/W	■	■	■	■	■	■	■
CCneg		R/W	▤						
CPprg		R/W	▤						
CVmon		R	■	■	■	■	■	■	■
CCmon		R	■	■	■	■	■	■	■
CPmon		R	▤						
CVlim		R	■	■					
CClim		R	■	■					
CClimneg		R	■	■					
CPlim		R	▤						
CPlimneg		R	▤						
Refresh Counter		R	■	■	■	■	■	■	■
Remote Control Register (RemCTRL)									
├ **RSD**	1	W	■	■	■	■	■	■	■
├ **Output**	2	W	■	■					
├ **RemCV**	512	W	■	■					
├ **RemCC**	1024	W	■	■					
└ **RemCP**		W	▤						
Status Register A									
├ **CV**	1	R	■	■					
├ **CC**	2	R	■	■	■	■	■	■	■
├ **CP**	4	R	■						
├ **V_{lim}**	8	R	■	■	■	■	■		
├ **I_{lim}**	16	R	■	■					
├ **P_{lim}**	32	R	■						
├ **DcfVolt**	64	R	■	■	■	■	■		
├ **DcfCurr**	128	R	■	■					
├ **OT**	256	R	■	■	■	■	■		
├ **PSOL**	512	R	■	■	■	■	■		
├ **ACF**	1024	R	■	■	■	■	■		
├ **Interlock**	2048	R	■	■					
├ **RSD**	4096	R	■	■	■	■	■	■	■
├ **Output**	8192	R	■	■					
└ **Frontpanellock**	16384	R	■	■					
Status Register B									
├ **Rem CV**	1	R	■	■					
├ **Rem CC**	2	R	■	■					
├ **Rem CP**	4	R	■						
├ **ProgRunning**	8	R	■	■					
├ **WaitForTrigger**	16	R	■	■					
├ **Master**	32	R	■	■					
├ **Slave**	64	R	■	■					
├ **V_{output} overload**	128	R	■	■	■	■	■		
├ **I_{output} overload**	256	R	■	■	■	■	■		
├ **Sensebreak**	8192	R	■	■					
├ **PROT**	16384	R	■	■					
└ **ProgOpenEndError**	32768	R	■	■					

Table 2.1: Available commands and compatibility with power supply series

2.5 Data formats

In the previous section an overview was given on compatible commands per power supply series. At this point a data format should be chosen that will be used to send the commands and receive data.

Two data formats exist, a float and a binary number representation. These are called Float Format A and 16-Bit Format A. When compared to each other, both formats have their advantages and disadvantages, these are compared and shown in Table 2.2.

The main difference between the two formats is processing speed internally at the interface of the HMS Anybus module. Which is achieved by sending and requesting smaller packages consisting of parameters with shorter word lengths, which can be seen in Table 2.3 and Table 2.4. The 16-bit format packages are smaller and therefore processed faster.

Data format	Advantages	Disadvantages
Float Format A	• Human readability (float representation) • Includes CVlim, CClim and Status Register B.	• Slower processing speed
16-Bit Format A	• Faster processing speed	• Does not include CVlim, CClim and Status Register B. • Less human readable (binary representation)

Table 2.2: Data format comparison

For each data format, a table is shown below that indicates the datatype for each command and its value range. This is shown in Table 2.3 and Table 2.4. The Chapter 3 of this guide will cover implementation for each specific protocol.

2.5.1 Float Format A

The parameters in the float format, beside the refresh counter and the status registers are of type REAL32. The word size is 16 bit, so two words are needed for the float type parameters. The register type parameters are all of single word length. This can be seen in Table 2.3.
The allowed range for the float parameters, as described in the Range column, are the maximum and minimum values of the power supply model.

Parameter	R/W	Datatype	Range
CVprg	Read/Write	REAL32	[0 .. V_{max}]
CCprg	Read/Write	REAL32	[0 .. I_{max}]
CVmon	Read	REAL32	[0 .. V_{max}]
CCmon	Read	REAL32	[-I_{max} .. I_{max}]
CVlim	Read	REAL32	-
CClim	Read	REAL32	-
Refresh Counter	Read	UINT16	-
Remote Control Register (RemCTRL)	Write	UINT16	-
Status Register A	Read	UINT16	-
Status Register B	Read	UINT16	-

Table 2.3: Float Format A Datatype and Value Range

2.5.2 16-Bit Format A

All parameters in the 16-bit format are of single word length, this can be seen in Table 2.4. Status Register B, CVlim and CClim are not present in this data format to keep the total data package as small as possible for best processing speed. The parameters like CVprg and CCmon, which are not representing a binary status register, need to be scaled to be interpreted correctly. To program these parameters, the values are scaled to the Nominal parameter value as shown in the Nominal column in Table 2.4. For example, to program a SM66-AR-110 (max. 66 V, max. 110 A) power supply, to a CV setting of 33 V. The value to send to the Anybus module is (62500/66)*33. See the equations below that illustrate the scaling operation.

$$Proggraming\ value\ = \left(\frac{Nominal\ Parameter\ value}{Maximum\ value\ of\ the\ power\ supply}\right) * Actual\ desired\ output\ value$$

$$Actual\ Measured\ output\ value = \left(\frac{Maximum\ value\ of\ the\ power\ supply}{Nominal\ parameter\ value}\right) * Monitoring\ value$$

Parameter	**R/W**	**Datatype**	**Range**	**Nominal (100% output)**
CVprg	Read/Write	UINT16	[0 .. 65000]	62500
CCprg	Read/Write	UINT16	[0 .. 32500]	31250
CVmon	Read	UINT16	[0 .. 65535]	62500
CCmon	Read	UINT16	[-32768 .. 32767]	31250
Refresh Counter	Read	UINT16	[0 .. 65535]	-
Remote Control Register (RemCTRL)	Write	UINT16	[0 .. 65535]	-
Status Register A	Read	UINT16	[0 .. 65535]	-

Table 2.4: 16-Bit Format A Datatype and Value Range

3 Fieldbus implementation

3.1 Modbus-TCP - Read/Write Registers

Implementation of the Anybus module into a Modbus-TCP network is done using Modbus registers. The Modbus registers are 16-bit wide. They are referenced by an address defined by the device address map. This map is shown for each data format in Table 3.1 and Table 3.2. The data type of each parameter was previously mentioned in section 2.5. The meaning of each parameter is briefly described in Appendix A1. The minimum processing speed that can be obtained is mentioned in the data sheet of the interface. Together with the network latency, this processing speed defines the maximum polling rate. According to the Modbus-TCP network guide from HMS Networks, the module can have up to four simultaneous connections. This information however needs to be verified by the user at HMS, as this aspect is not within our scope of influence. The Anybus module has its own web interface, showing current parameter values, which can be used to aid implementation and debugging.

3.1.1 Float Format A

Parameter	Read		Write		Size
	Address	Function Code	Address	Function Code	
CVprg	0x800 + 0x801	0x03	0x000 + 0x001	0x10	2 Words
CVmon	0x802 + 0x803	0x03	-	-	2 Words
CCprg	0x804 + 0x805	0x03	0x002 + 0x003	0x10	2 Words
CCmon	0x806 + 0x807	0x03	-	-	2 Words
CVlim	0x808 + 0x809	0x03	-	-	2 Words
CClim	0x80A + 0x80B	0x03	-	-	2 Words
RemCTRL	-	-	0x004	0x06	1 Word
Refresh Counter	0x80C	0x03	-	-	1 Word
Status Register A	0x80D	0x03	-	-	1 Word
Status Register B	0x80E	0x03	-	-	1 Word

Table 3.1: Float format A Modbus-TCP device address map

3.1.2 16-Bit Format A

Parameter	Read		Write		Size
	Address	Function Code	Address	Function Code	
CVprg	0x800	0x03	0x000	0x06	1 Word
CVmon	0x801	0x03	-	-	1 Word
CCprg	0x802	0x03	0x001	0x06	1 Word
CCmon	0x803	0x03	-	-	1 Word
RemCTRL	-	-	0x002	0x06	1 Word
Refresh Counter	0x804	0x03	-	-	1 Word
Status Register A	0x805	0x03	-	-	1 Word

Table 3.2: 16-Bit format A Modbus-TCP device address map

3.2 Ethernet/IP – Implicit Messaging

When writing to the Anybus module, a full array of bytes (BYTE[x]) must be sent. This must include CVprg, CCprg, and RemCTRL even when only one of them is adjusted. For example, In 'Float Format A', this is a BYTE[10] (array of ten bytes), In '16-Bit Format A' this is a BYTE[6].

3.2.1 Float Format A

Parameter	Read object: 0x04, instance: 0x64, attribute: 0x03	Write object: 0x04, instance: 0x96, attribute: 0x03	Size
CVprg	0d000 .. 0d031	BYTE[4+4+2]	2 Words
CVmon	0d032 .. 0d063	-	2 Words
CCprg	0d064 .. 0d095	BYTE[4+4+2]	2 Words
CCmon	0d096 .. 0d127	-	2 Words
CVlim	0d128 .. 0d159	-	2 Words
CClim	0d160 .. 0d191	-	2 Words
RemCTRL	-	BYTE[4+4+2]	1 Word
Refresh Counter	0d192 .. 0d207	-	1 Word
Status Register A	0d208 .. 0d223	-	1 Word
Status Register B	0d224 .. 0d239	-	1 Word

3.2.2 16-Bit Format A

Parameter	Read *object: 0x04,* *instance: 0x64,* *attribute: 0x03*	Write object: 0x04, instance: 0x96, attribute: 0x03	Size
CVprg	0d000 .. 0d015	BYTE[2+2+2]	1 Word
CVmon	0d016 .. 0d031	-	1 Word
CCprg	0d032 .. 0d047	BYTE[2+2+2]	1 Word
CCmon	0d048 .. 0d063	-	1 Word
RemCTRL	-	BYTE[2+2+2]	1 Word
Refresh Counter	0d064 .. 0d079	-	1 Word
Status Register A	0d080 .. 0d095	-	1 Word

3.3 CANopen – SDO

Note that the CANopen module does not have an internal termination resistor and that it should be added externally if required.

3.3.1 Float Format A

Parameter	Read		Write		Size
	Index	**Subindex**	**Index**	**Subindex**	
CVprg	0x200C	0x0001	0x200A	0x0001	2 Words
CVmon	0x200C	0x0002	-	-	2 Words
CCprg	0x200C	0x0003	0x200A	0x0002	2 Words
CCmon	0x200C	0x0004	-	-	2 Words
CVlim	0x200C	0x0005	-	-	2 Words
CClim	0x200C	0x0006	-	-	2 Words
RemCTRL	-	-	0x200B	0x0000	1 Word
Refresh Counter	0x200D	0x0000	-	-	1 Word
Status Register A	0x200E	0x0000	-	-	1 Word
Status Register B	0x200F	0x0000	-	-	1 Word

3.3.2 16-Bit Format A

Parameter	Read		Write		Size
	Index	**Subindex**	**Index**	**Subindex**	
CVprg	0x2016	0x0001	0x2014	0x0001	1 Word
CVmon	0x2016	0x0002	-	-	1 Word
CCprg	0x2016	0x0003	0x2014	0x0002	1 Word
CCmon	0x2016	0x0004	-	-	1 Word
RemCTRL	-	-	0x2015	0x0000	1 Word
Refresh Counter	0x2017	0x0000	-	-	1 Word
Status Register A	0x2018	0x0000	-	-	1 Word

3.4 EtherCAT - SDO

3.4.1 Float Format A

Parameter	Read		Write		Size
	Index	Subindex	Index	Subindex	
CVprg	0x200C	0x0001	0x200A	0x0001	2 Words
CVmon	0x200C	0x0002	-	-	2 Words
CCprg	0x200C	0x0003	0x200A	0x0002	2 Words
CCmon	0x200C	0x0004	-	-	2 Words
CVlim	0x200C	0x0005	-	-	2 Words
CClim	0x200C	0x0006	-	-	2 Words
RemCTRL	-	-	0x200B	0x0000	1 Word
Refresh Counter	0x200D	0x0000	-	-	1 Word
Status Register A	0x200E	0x0000	-	-	1 Word
Status Register B	0x200F	0x0000	-	-	1 Word

3.4.2 16-Bit Format A

Parameter	Read		Write		Size
	Index	Subindex	Index	Subindex	
CVprg	0x2016	0x0001	0x2014	0x0001	1 Word
CVmon	0x2016	0x0002	-	-	1 Word
CCprg	0x2016	0x0003	0x2014	0x0002	1 Word
CCmon	0x2016	0x0004	-	-	1 Word
RemCTRL	-	-	0x2015	0x0000	1 Word
Refresh Counter	0x2017	0x0000	-	-	1 Word
Status Register A	0x2018	0x0000	-	-	1 Word

3.5 PROFIBUS – DPV1

3.5.1 Float Format A

Parameter	Read		Write		Size
	Slot	Index	Slot	Index	
CVprg	0	11	0	9	2 Words
CVmon	0	11	-	-	2 Words
CCprg	0	11	0	9	2 Words
CCmon	0	11	-	-	2 Words
CVlim	0	11	-	-	2 Words
CClim	0	11	-	-	2 Words
RemCTRL	-	-	0	10	1 Word
Refresh Counter	0	12	-	-	1 Word
Status Register A	0	13	-	-	1 Word
Status Register B	0	14	-	-	1 Word

3.5.2 16-Bit Format A

Parameter	Read		Write		Size
	Slot	Index	Slot	Index	
CVprg	0	21	0	19	1 Word
CVmon	0	21	-	-	1 Word
CCprg	0	21	0	19	1 Word
CCmon	0	21	-	-	1 Word
RemCTRL	-	-	0	20	1 Word
Refresh Counter	0	22	-	-	1 Word
Status Register A	0	23	-	-	1 Word

3.6 PROFINET IRT

PROFINET GSDML-files are available on our website. The files can be downloaded at the following pages:

- INT MOD ANY: **Products > Interfaces > Anybus module > Downloads**
- PSC ANY EXT: **Products > Interfaces > Anybus unit (external) > Downloads**

These configuration files can be imported into the desired PLC programming software.

3.6.1 Float Format A

Module	**Submodule**
DE_INT_MOD_ANY_PRG	CVprg
	CCprg
DE_INT_MOD_ANY_STS_PRG	RemCTRL
DE_INT_MOD_ANY_MON	CVprg
	CVmon
	CCprg
	CCmon
	CVlim
	CClim
DE_INT_MOD_ANY_REFRESH	Refresh Counter
DE_INT_MOD_ANY_STS_MON_A	Status Register A
DE_INT_MOD_ANY_STS_MON_B	Status Register B

3.6.2 16-Bit Format A

Module	**Submodule**
DE_INT_MOD_ANY_PRG16	CVprg
	CCprg
DE_INT_MOD_ANY_STS_PRG	RemCTRL
DE_INT_MOD_ANY_MON16	CVprg
	CVmon
	CCprg
	CCmon
DE_INT_MOD_ANY_REFRESH	Refresh Counter
DE_INT_MOD_ANY_STS_MON_A	Status Register A

A. Appendix

A1. Command and Status Descriptions

Parameter	Read/ Write	Description
CVprg	R/W	Read or write the Constant Voltage (CV) setting
CCprg	R/W	Read or write the positive Constant Current (CC) setting
CCneg	R/W	Read or write the negative Constant Current (CC) setting
CPprg	R/W	Read or write the Constant Power (CP) setting
CVmon	R	Read the measured Voltage
CCmon	R	Read the measured Current
CPmon	R	Read the measured Power
CVlim	R	Read the Constant Voltage (CV) Limit setting
CClim	R	Read the Constant Current (CC) positive Limit setting
CClimneg	R	Read the Constant Current (CC) negative Limit setting
CPlim	R	Read the Constant Power (CP) Limit setting
Refresh Counter	R	Read the Refresh Counter value of the Anybus module
Remote Control (RemCTRL)		Register of multiple settings
├ **RSD**	W	Set Remote Shut Down (1:ON)
├ **Output**	W	Set Output, enable or disable (1:ON, 0:OFF)
├ **RemCV**	W	Set Constant Voltage (CV) programming source to Remote (1:ON)
├ **RemCC**	W	Set Constant Voltage (CC) programming source to Remote (1:ON)
└ **RemCP**	W	Set Constant Voltage (CP) programming source to Remote (1:ON)
Status Register A		Register of multiple statuses (ON or OFF)
├ **CV**	R	Read CV-mode status
├ **CC**	R	Read CC-mode status
├ **CP**	R	Read CP-mode status
├ **V_{lim}**	R	Read Voltage-Limit Alert status
├ **I_{lim}**	R	Read Current-Limit Alert status
├ **P_{lim}**	R	Read Power-Limit Alert status
├ **DcfVolt**	R	Read DC-fail Voltage status
├ **DcfCurr**	R	Read DC-fail Current statusd
├ **OT**	R	Read Over-Temperature Alert status
├ **PSOL**	R	Read Power Supply Overload status
├ **ACF**	R	Read AC-Failure Alert status
├ **Interlock**	R	Read Interlock status
├ **RSD**	R	Read Remote Shutdown Status
├ **Output**	R	Read Output ON/off status
└ **Frontpanellock**	R	Read Front Panel Lock status
Status Register B		Register of multiple statuses (ON or OFF)
├ **Rem CV**	R	Read Constant Voltage (CV) programming source Status
├ **Rem CC**	R	Read Constant Voltage (CC) programming source Status
├ **Rem CP**	R	Read Constant Voltage (CP) programming source Status
├ **ProgRunning**	R	Read Sequencer Program status
├ **WaitForTrigger**	R	Read Sequencer Trigger status
├ **Master**	R	Read Master status, for Master/Slave operation
├ **Slave**	R	Read Slave status, for Master/Slave operation
├ **V_{output} overload**	R	Read Output voltage overload status
├ **I_{output} overload**	R	Read Output current overload status
├ **Sensebreak**	R	Read Sense break status of remote sensing
├ **PROT**	R	Read Self Protect status
└ **ProgOpenEndError**	R	Read Sequencer Program Open End Error status

Additional notes:

- RemCV, RemCC, RemCP set the programming source of a power supply to remote. The SM3300 and SM15K power supplies will automatically set the programming source to the Slot with the Anybus carrier module inserted.
- RSD and Output (ON/off) settings are similar in behavior. However, individual behavior can be customized in case of a power sink option or natively bidirectional power supply like the SM15K-series. Please consult the manuals of the power supply and or power sink option.
- For SM3300 and SM15K power supplies, monitoring and status register parameters can only be read if the INT-MOD-ANY is selected as the programming source. The programming source is selected either via the front, web interface of the power supply or through the RemCV, RemCC and RemCP commands from the Anybus module.