PMBus™
Power System Management Protocol Specification
Part II – Command Language

Revision 1.3.1

13 March 2015

www.powerSIG.org

Filename: PMBus_Specification_Part_II_Rev_1_3_1_20150313A.docx
Last saved: 12 Mar 2015, 15:21

DISCLAIMER

This specification is provided “as is” with no warranties whatsoever, whether express, implied or statutory, including but not limited to any warranty of merchantability, non-infringement, or fitness for any particular purpose, or any warranty otherwise arising out of any proposal, specification or sample.

In no event will any specification co-owner be liable to any other party for any loss of profits, loss of use, incidental, consequential, indirect, or special damages arising out of this specification, whether or not such party had advance notice of the possibility of such damages. Further, no warranty or representation is made or implied relative to freedom from infringement of any third party patents when practicing the specification.

Other product and corporate names may be trademarks of other companies and are used only for explanation and to the owner’s benefit, without intent to infringe.

REVISION HISTORY

REV	DATE	DESCRIPTION	EDITED BY
1.0	28 Mar 2005	First public release.	Robert V. White Artesyn Technologies
1.1	5 Feb 2007	Second public release.	Robert V. White Astec/Artesyn
1.2	6 Sep 2010	Third public release	Robert V. White Embedded Power Labs
1.3	18 March 2014	Fourth public release	Robert V. White Embedded Power Labs
1.3.1	13 March 2015	Fifth public release	Robert V. White Embedded Power Labs

Table Of Contents

1. Introduction ... 11
1.1. Specification Scope ... 11
1.1.1. Specification Structure ... 11
1.1.2. What Is Included ... 11
1.1.3. What Is Not Included In the PMBus Specification ... 11
1.2. Specification Changes Since The Last Revision ... 11
1.3. Where To Send Feedback And Comments ... 11
2. Related Documents ... 11
2.1. Scope ... 11
2.2. Applicable Documents ... 12
2.3. Reference Documents ... 12
3. Reference Information ... 12
3.1. Signal and Parameter Names ... 12
3.2. Numerical Formats ... 12
3.2.1. Decimal Numbers ... 12
3.2.2. Floating Point Numbers ... 12
3.2.3. Binary Numbers ... 12
3.2.4. Hexadecimal Numbers ... 13
3.2.5. Examples ... 13
3.3. Bit And Byte Order ... 13
3.4. Bit And Byte Illustrations ... 13
3.5. Abbreviations, Acronyms And Definitions ... 15
4. Addressing And Grouping ... 17
4.1. Device Addresses ... 17
4.2. General Call Address (Global Broadcast) ... 17
4.3. Sending Commands To A Group ... 17
4.4. ZONE_READ And ZONE_WRITE ... 18
5. Commands ... 18
5.1. Commands And Command Codes ... 18
5.2. Command Extensions ... 18
5.3. Command Execution ... 18
5.4. Writing And Reading PMBus Devices ... 18
5.4.1. All Packets Start With A Write Address ... 18
5.4.2. Every Parameter That Can Be Written Must Be Readable ... 18
5.4.3. Commands May Be Read Only ... 19
6. Memory Model, Startup Behavior And Defaults ... 19
6.1. Order Of Memory Loading And Precedence ... 19
6.2. The Default And User Stores ... 21
7. Numeric Data Formats ... 22
7.1. Summary ... 22
7.2. Restrictions ... 23
7.3. LINEAR11 Numeric Format ... 23
7.4. DIRECT Data Format ... 24
7.4.1. Interpreting Received Values ... 24
7.4.2. Sending A Value ... 24

7.4.3. Obtaining The Value Of The m, b, And R Coefficients ... 24
7.5. IEEE-754 Floating Point ... 25
7.6. IEEE-754 Half Precision Floating Point ... 25
7.6.1. IEEE-754 Single Precision Floating Point Format ... 25
7.6.2. Floating Point Value Restrictions ... 25
7.7. Manufacturer Specific Numeric Data Formats ... 26
7.8. Accuracy ... 26
7.9. Resolution ... 26
8. Data Formats For The Output Voltage And Output Voltage Related Parameters ... 26
8.1. Restrictions ... 26
8.1.1. Positive And Negative Output Voltages ... 26
8.1.2. Floating Point Format ... 27
8.2. Two Step Process ... 27
8.3. VOUT_MODE Command ... 27
8.3.1. Mode Selection ... 27
8.4. Data Bytes For Output Voltage Related Commands ... 28
8.4.1. LINEAR16 Formats ... 29
8.4.1.1. ULINEAR16 Format ... 29
8.4.1.2. SLINEAR16 Format ... 29
8.4.2. VID Format ... 29
8.4.3. DIRECT Format ... 30
8.4.4. IEEE-754 Half Precision Floating Point ... 31
8.5. Absolute Value And Relative Value Voltage Related Commands ... 31
8.5.1. Absolute Value Mode Data Bytes ... 32
8.5.2. Relative Value ... 32
8.5.3. Relative Value Mode Notes ... 33
9. Setting And Monitoring The Output Voltage And Current ... 33
9.1. VOUT_SCALE_LOOP And VOUT_SCALE_MONITOR ... 33
9.2. Setting The Output Voltage ... 35
9.3. Switching Between PMBus and AVSBus Control Of The Output Voltage ... 37
9.4. Making And Calibrating Output Current Measurements ... 38
9.5. Deleted ... 39
10. Fault Management And Reporting ... 39
10.1. Monitoring Operation ... 39
10.2. General Description Of PMBus Device Fault Management ... 40
10.2.1. Warning Conditions ... 41
10.2.2. Fault Conditions ... 41
10.2.3. Clearing Warning Or Fault Bits ... 41
10.2.4. Clearing Individual Bits ... 42
10.2.5. Clearing Bits In The STATUS_BYTE And STATUS_WORD ... 42
10.2.5.1. General Rules ... 42
10.2.5.2. Exceptions ... 42
10.2.6. Immediate Reassertion After Clearing If Condition Is Still Present ... 43
10.3. Conceptual View Of How Status Bits And SMBALERT# Work ... 43
10.4. Setting Fault And Warning Thresholds ... 44
10.5. Setting The Response To A Detected Fault Condition ... 44
10.5.1. Response To Voltage, Temperature And TON_MAX Faults ... 45
10.5.2. Response To Current Faults ... 46

10.6. Reporting Faults And Warnings To The Host ... 48
10.6.1. SMBALERT# Signal And Process ... 48
10.6.2. Direct PMBus Device To Host Communication ... 49
10.7. Clearing A Shutdown Due To A Fault ... 49
10.8. Data Transmission Faults ... 49
10.8.1. Corrupted Data ... 50
10.8.2. Sending Too Few Bits ... 50
10.8.3. Reading Too Few Bits ... 50
10.8.4. Host Sends Or Reads Too Few Bytes ... 50
10.8.5. Host Sends Too Many Bytes ... 51
10.8.6. Reading Too Many Bytes ... 51
10.8.7. Device Busy ... 51
10.9. Data Content Faults ... 52
10.9.1. Improperly Set Read Bit In The Address Byte ... 52
10.9.2. Unsupported Command Code ... 52
10.9.3. Invalid Or Unsupported Data ... 52
10.9.4. Data Out Of Range Fault ... 53
10.9.5. Reserved Bits ... 53
11. Address, Memory, Communication And Capability Related Commands ... 54
11.1. WRITE_PROTECT ... 54
11.2. STORE_DEFAULT_ALL ... 54
11.3. RESTORE_DEFAULT_ALL ... 54
11.4. STORE_DEFAULT_CODE ... 55
11.5. RESTORE_DEFAULT_CODE ... 55
11.6. STORE_USER_ALL ... 56
11.7. RESTORE_USER_ALL ... 56
11.8. STORE_USER_CODE ... 56
11.9. RESTORE_USER_CODE ... 57
11.10. PAGE ... 57
11.11. PHASE ... 59
11.12. CAPABILITY ... 59
11.13. QUERY ... 60
11.14. PAGE_PLUS_WRITE ... 61
11.15. PAGE_PLUS_READ ... 61
11.16. Zone Operation Commands ... 62
11.16.1. ZONE_CONFIG ... 62
11.16.2. ZONE_ACTIVE ... 63
12. On, Off And Margin Testing Related Commands ... 64
12.1. OPERATION ... 64
12.1.1. OPERATION Command Bit [7] ... 64
12.1.2. OPERATION Command Bit [6] ... 65
12.1.3. OPERATION Command Bits [5:4] ... 65
12.1.4. OPERATION Command Bits [3:2] ... 65
12.1.5. OPERATION Command Bit [1] ... 65
12.1.6. OPERATION Command Bit [0] ... 66
12.1.7. OPERATION Command Invalid Data ... 66
12.2. ON_OFF_CONFIG ... 67
13. Output Voltage Related Commands ... 69

13.1. VOUT_MODE ... 69
13.2. VOUT_COMMAND ... 69
13.3. VOUT_TRIM ... 69
13.4. VOUT_CAL_OFFSET ... 69
13.5. VOUT_MAX ... 69
13.6. VOUT_MARGIN_HIGH ... 70
13.7. VOUT_MARGIN_LOW ... 70
13.8. VOUT_TRANSITION_RATE ... 70
13.9. VOUT_DROOP ... 70
13.10. VOUT_SCALE_LOOP ... 71
13.11. VOUT_SCALE_MONITOR ... 71
13.12. VOUT_MIN ... 71
14. Other Commands ... 72
14.1. COEFFICIENTS ... 72
14.2. POUT_MAX ... 72
14.3. MAX_DUTY ... 73
14.4. FREQUENCY_SWITCH ... 73
14.5. VIN_ON ... 73
14.6. VIN_OFF ... 73
14.7. INTERLEAVE ... 73
14.8. IOUT_CAL_GAIN ... 74
14.9. IOUT_CAL_OFFSET ... 75
14.10. FAN_CONFIG_1_2 ... 75
14.11. FAN_CONFIG_3_4 ... 76
14.12. FAN_COMMAND_n ... 77
14.13. POWER_MODE ... 77
15. Fault Related Commands ... 78
15.1. CLEAR_FAULTS ... 78
15.2. VOUT_OV_FAULT_LIMIT ... 78
15.3. VOUT_OV_FAULT_RESPONSE ... 79
15.4. VOUT_OV_WARN_LIMIT ... 79
15.5. VOUT_UV_WARN_LIMIT ... 79
15.6. VOUT_UV_FAULT_LIMIT ... 80
15.7. VOUT_UV_FAULT_RESPONSE ... 80
15.8. IOUT_OC_FAULT_LIMIT ... 80
15.9. IOUT_OC_FAULT_RESPONSE ... 80
15.10. IOUT_OC_LV_FAULT_LIMIT ... 81
15.11. IOUT_OC_LV_FAULT_RESPONSE ... 81
15.12. IOUT_OC_WARN_LIMIT ... 81
15.13. IOUT_UC_FAULT_LIMIT ... 81
15.14. IOUT_UC_FAULT_RESPONSE ... 82
15.15. DELETED ... 82
15.16. DELETED ... 82
15.17. OT_FAULT_LIMIT ... 82

15.18. OT_FAULT_RESPONSE 82
15.19. OT_WARN_LIMIT 83
15.20. UT_WARN_LIMIT 83
15.21. UT_FAULT_LIMIT 83
15.22. UT_FAULT_RESPONSE 83
15.23. VIN_OV_FAULT_LIMIT 84
15.24. VIN_OV_FAULT_RESPONSE 84
15.25. VIN_OV_WARN_LIMIT 84
15.26. VIN_UV_WARN_LIMIT 84
15.27. VIN_UV_FAULT_LIMIT 85
15.28. VIN_UV_FAULT_RESPONSE 85
15.29. IIN_OC_FAULT_LIMIT 85
15.30. IIN_OC_FAULT_RESPONSE 85
15.31. IIN_OC_WARN_LIMIT 86
15.32. POWER_GOOD Signal Limits 86
15.32.1. POWER_GOOD_ON 86
15.32.2. POWER_GOOD_OFF 86
15.33. POUT_OP_FAULT_LIMIT 87
15.34. POUT_OP_FAULT_RESPONSE 87
15.35. POUT_OP_WARN_LIMIT 87
15.36. PIN_OP_WARN_LIMIT 87
15.37. Other Fault And Warning Responses 88
15.38. SMBALERT_MASK Command 88
16. Output Voltage Sequencing Commands 88
16.1. TON_DELAY 88
16.2. TON_RISE 89
16.3. TON_MAX_FAULT_LIMIT 89
16.4. TON_MAX_FAULT_RESPONSE 89
16.5. TOFF_DELAY 89
16.6. TOFF_FALL 89
16.7. TOFF_MAX_WARN_LIMIT 90
17. Unit Status Commands 90
17.1. STATUS_BYTE 91
17.2. STATUS_WORD 92
17.3. STATUS_VOUT 93
17.4. STATUS_IOUT 93
17.5. STATUS_INPUT 94
17.6. STATUS_TEMPERATURE 94
17.7. STATUS_CML (Communications, Logic, And Memory) 95
17.8. STATUS_OTHER 95
17.9. STATUS_MFR_SPECIFIC 96
17.10. STATUS_FANS_1_2 96
17.11. STATUS_FANS_3_4 97
18. Reading Parametric Information 97
18.1. READ_VIN 98

18.2. READ_IIN ... 98
18.3. READ_VCAP ... 98
18.4. READ_VOUT ... 98
18.5. READ_IOUT ... 98
18.6. READ_TEMPERATURE_n ... 98
18.7. READ_FAN_SPEED_n ... 99
18.8. Deleted ... 99
18.9. READ_DUTY_CYCLE ... 99
18.10. READ_FREQUENCY ... 99
18.11. READ_POUT ... 99
18.12. READ_PIN ... 99
18.13. READ_EIN And READ_EOUT ... 99
18.14. READ_KWH_IN/READ_KWH_OUT ... 101
18.15. READ_KWH_CONFIG ... 102
19. Reserved ... 104
20. Reserved ... 104
21. Reserved ... 104
22. Manufacturer's Information ... 105
22.1. PMBUS_REVISION ... 105
22.2. Inventory Information ... 105
22.2.1. MFR_ID ... 106
22.2.2. MFR_MODEL ... 106
22.2.3. MFR_REVISION ... 106
22.2.4. MFR_LOCATION ... 106
22.2.5. MFR_DATE ... 106
22.2.6. MFR_SERIAL ... 106
22.2.7. IC_DEVICE_ID ... 106
22.2.8. IC_DEVICE_REV ... 106
22.3. Manufacturer Ratings ... 106
22.3.1. MFR_VIN_MIN ... 107
22.3.2. MFR_VIN_MAX ... 107
22.3.3. MFR_IIN_MAX ... 107
22.3.4. MFR_PIN_MAX ... 107
22.3.5. MFR_VOUT_MIN ... 107
22.3.6. MFR_VOUT_MAX ... 107
22.3.7. MFR_IOUT_MAX ... 107
22.3.8. MFR_POUT_MAX ... 107
22.3.9. MFR_TAMBIENT_MAX ... 107
22.3.10. MFR_TAMBIENT_MIN ... 107
22.3.11. MFR_EFFICIENCY_LL ... 107
22.3.12. MFR_EFFICIENCY_HL ... 108
22.3.13. MFR_PIN_ACCURACY ... 109
22.3.14. APP_PROFILE_SUPPORT ... 109
22.3.15. MFR_MAX_TEMP_1, _2, _3 ... 110
23. User Data And Configuration ... 110
24. Manufacturer Specific Commands ... 111
25. Command Extensions ... 111
25.1. MFR_SPECIFIC_COMMAND_EXT ... 111
25.2. PMBUS_COMMAND_EXT ... 111

APPENDIX I. Command Summary 112
APPENDIX II. Summary Of Changes 121

Table Of Figures

Figure 1. Bit Order Within A Byte 13
Figure 2. Conceptual View Of Possible PMBus Device Memory And Communication 20
Figure 3. Flowchart Of Conceptual Loading Operating Memory At Startup 21
Figure 4. LINEAR11 Numeric Format Data Bytes 23
Figure 5. Format Of Floating Point Format Data Bytes 25
Figure 6. VOUT_MODE Command Data Byte Structure 28
Figure 7. ULINEAR16 Format Data Bytes 29
Figure 8. VID Format Data Bytes 30
Figure 9. DIRECT Format Mode Data Bytes 31
Figure 10. Floating Point Format Mode Data Bytes 31
Figure 11. Output Voltage Sensing In A Typical Power Converter 34
Figure 12. Conceptual View Of The Application Of The VOUT_SCALE_LOOP Command 35
Figure 13. Conceptual View Of How Output Voltage Related Commands Are Applied 36
Figure 14. Permitted Changes Of Output Voltage Control 38
Figure 15. Generating READ_IOUT Concept 39
Figure 16. Status Register Map 40
Figure 17. Conceptual View Of Creating Bits In STATUS_BYTE And STATUS_WORD 42
Figure 18. Conceptual Schematic Of Status Bits And SMBALERT# 44
Figure 19. Interaction Of SMBALERT# And Status Registers 49
Figure 20. Packet Structure For PMBus Device To Notify Host 49
Figure 21. Conceptual View Of Paging Used For A Multiple Output PMBus Device 58
Figure 22. Conceptual View Of Using Paging With A PMBus To Non-PMBus Device Adapter 58
Figure 23. PAGE_PLUS_WRITE Command Example With Data Being Written And PEC 61
Figure 24. PAGE_PLUS_READ Command Example With Data Being Read And PEC 61
Figure 25. ZONE_CONFIG Command 63
Figure 26. ZONE_ACTIVE Command Used To Set The Active Write Zone And Active Read Zone 64
Figure 27. Using ZONE_ACTIVE Command To Read The Active Write Zone And Active Read Zone Stored In A Device 64
Figure 28. OPERATION Command Data Byte 64
Figure 29. Retrieving Coefficients Using PEC 72
Figure 30. Illustration Of The INTERLEAVE Command Function 74
Figure 31. SMBALERT_MASK Command Packet Format 88
Figure 32. Retrieving The SMBALERT_MASK Setting For A Given Status Register 88
Figure 33. Summary Of The Status Registers 91
Figure 34. READ_EIN Command Packet Format 101
Figure 35. READ_EOUT Command Packet Format 101
Figure 36. READ_KWH_IN/OUT Command Packet Format With Packet Error Checking 102
Figure 37. READ_KWH_IN/OUT Command Packet Format To Reset The Accumulator 104
Figure 38. APP_PROFILE_SUPPORT Packet Example 110

Table Of Tables

Table 1. Bit And Byte Symbols Used In This Specification 13
Table 2. Summary Of The VOUT_MODE Data Byte Format 28
Table 3. VID Types Supported By PMBus 30
Table 4. Voltage, Temperature And TON_MAX Faults Response Data Byte Details 45
Table 5. Current Fault Response Data Byte Details 46
Table 6. WRITE_PROTECT Command Data Byte 54
Table 7. CAPABILITY COMMAND Data Byte Format 59
Table 8. QUERY Command Returned Data Byte Format 60
Table 9. OPERATION Command Data Byte Contents 67
Table 10. ON_OFF_CONFIG Data Byte 68
Table 11. INTERLEAVE Data Bytes Format 74
Table 12. FAN_CONFIG_1_2 Data Byte Format 75
Table 13. FAN_CONFIG_3_4 Data Byte Format 76
Table 14. POWER_MODE Command Data Byte Format 78
Table 15. STATUS_BYTE Message Contents 91
Table 16. STATUS_WORD Message Contents 92
Table 17. STATUS_VOUT Data Byte 93
Table 18. STATUS_IOUT Data Byte 93
Table 19. STATUS_INPUT Data Byte 94
Table 20. STATUS_TEMPERATURE Data Byte 94
Table 21. STATUS_CML Data Byte 95
Table 22. STATUS_OTHER Data Byte 95
Table 23. STATUS_MFR_SPECIFIC Data Byte 96
Table 24. STATUS_FANS_1_2 Data Byte 96
Table 25. STATUS_FANS_3_4 Data Byte 97
Table 26. READ_KWH_CONFIG Data Byte Format 102
Table 27. PMBus Revision Data Byte Contents 105
Table 28. Data Format Of The MFR_EFFICIENCY_LL Command 108
Table 29. Data Format Of The MFR_EFFICIENCY_HL Command 109
Table 30. APP_PROFILE_SUPPORT First Data Byte Contents 110
Table 31. Command Summary 112

1. Introduction

The Power Management Bus ("PMBus™") is an open standard protocol that defines a means of communicating with power conversion and other devices.

For more information, please see the System Management Interface Forum Web site: www.powerSIG.org.

1.1. Specification Scope

1.1.1. Specification Structure

The PMBus specification is in three parts. Part I includes the general requirements, defines the transport, and defines the electrical interface and timing requirements of hardwired signals.

Part II, this document, describes the operation of commands, data formats, fault management and defines the command language used with the PMBus.

Part III defines the transport, electrical interface, timing requirements and command language for AVSBus.

1.1.2. What Is Included

This specification defines a protocol to manage power converters and a power system via communication over a digital communication bus.

1.1.3. What Is Not Included In the PMBus Specification

The PMBus specification is not a definition or specification of:

- A particular power conversion device or family of power conversion devices
- A specification of any individual or family of integrated circuits.

This specification does not address direct unit to unit communication such as analog current sharing, real-time analog or digital voltage tracking, and switching frequency clock signals.

1.2. Specification Changes Since The Last Revision

A summary of the changes between this revision and Revision 1.2 are shown in APPENDIX II.

1.3. Where To Send Feedback And Comments

Please send all comments by email to: techquestions@smiforum.org.

2. Related Documents

2.1. Scope

If the requirements of this specification and any of the reference documents are in conflict, this specification shall have precedence unless otherwise stated.

Referenced documents apply only to the extent that they are referenced.

The latest version and all amendments of the referenced documents at the time the power system is released to manufacturing apply.

2.2. Applicable Documents

Applicable documents include information that is, by extension, part of this specification.

[A01] PMBus Power System Management Protocol, Part I, General Requirements, Transport And Electrical Interface

[A02] PMBus Power System Management Protocol, Part III, AVSBus

[A03] System Management Interface Forum (SMIF), *System Management Bus (SMBus) Specification*, Version 3.0

[A04] *The I²C-bus specification and user manual*, Rev. 5, NXP Semiconductors, 9 October 2012

[A05] ISO/IEC 8859-1:1998*, 8-bit single-byte coded graphic character sets -- Part 1: Latin alphabet No. 1*, and all corrigenda, amendments published through the date of release of this specification.

[A06] PMBus Application Profile: Server AC-DC Power Supplies

[A07] PMBus Application Profile: DC-DC Converters For Microprocessor Power And Other Computer Applications

[A08] PMBus Application Profile: DC-DC Converters For General Purpose Use

2.3. Reference Documents

Reference documents have background or supplementary information to this specification. They do not include requirements or specifications that are considered part of this document.

[R01] IEEE-754-2008, IEEE Standard for Floating-Point Arithmetic

[R02] PMBus Application Note AN001, Using The ZONE_READ And ZONE_WRITE Protocols

3. Reference Information

3.1. Signal and Parameter Names

The names of signals and parameters are given in capital letters. Underscores are used to separate words rather than embedded spaces (example: SIGNAL_NAME).

The names of signals that are active low and parameters that are true when the value is 0 are indicated with an octothorpe (#) suffix (example: WRITE# means that the device can be written when the signal is low).

3.2. Numerical Formats

All numbers are decimal unless explicitly designated otherwise.

3.2.1. Decimal Numbers

Numbers explicitly identified as decimal are identified with a suffix of “d”.

3.2.2. Floating Point Numbers

Numbers explicitly identified as floating point are identified with a suffix of “f”.

3.2.3. Binary Numbers

Numbers in binary format are indicated by a suffix of “b”. Unless otherwise indicated, all binary numbers are unsigned.

All signed binary numbers are two's complement.

3.2.4. Hexadecimal Numbers

Numbers in hexadecimal format are indicated by a suffix of "h".

3.2.5. Examples

255d⇔ FFh ⇔ 11111111b

175d⇔ AFh ⇔ 10101111b

1.2f

3.3. Bit And Byte Order

As specified in the SMBus specification [A03]:

- When data is transmitted, the lowest order byte is sent first and the highest order byte is sent last.
- Within any byte, the most significant bit (MSB) is sent first and the least significant bit (LSB) is sent last.

3.4. Bit And Byte Illustrations

The transmission of bits, bytes and packets is illustrated in this section.

In all cases, the least significant bit is indicated as Bit 0. The most significant bit of a byte is always Bit 7, as shown below in Figure 1.

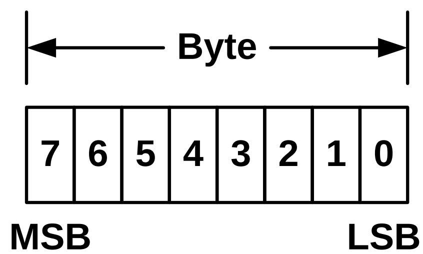
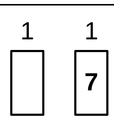
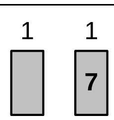
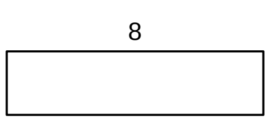


Figure 1. Bit Order Within A Byte

Within this specification, transactions over the PMBus are described. The symbols used to describe the details of those transactions and protocols are shown in Table 1.

Table 1. Bit And Byte Symbols Used In This Specification

Symbol	Meaning
	A unshaded vertical rectangle indicates a single bit sent from the host (bus master) to a slave
	A shaded vertical rectangle with a shaded interior indicates a bit sent from a slave device to the bus master.
	An unshaded rectangle with a number over it represents one or more bits, as indicated by the number, sent from the master to the slave. The name of the data or bit field may be included within the rectangle. If the data has a specific value, as might be shown in an example of a command, the

Symbol	Meaning
8 DATA NAME DATA VALUE	value is written below the data or bit field name.
8 [shaded rectangle] 8 DATA NAME DATA VALUE	A shaded rectangle with a number over it represents one or more bits, as indicated by the number, sent from the slave to the master. The name of the data or bit field may be included within the rectangle. If the data has a specific value, as might be shown in an example of a command, the value is written below the data or bit field name.
S	The START condition sent from a bus master device. The START condition is not a bit and does not have a number 1 over it.
Sr	A REPEATED START condition sent from a bus master device. The REPEATED START condition is not a bit and does not have a number 1 over it
1 A	An Acknowledge (ACK) condition send from the host
1 NA	A Not Acknowledge (NACK) condition sent from the host
1 A (shaded)	An ACKnowledge condition sent from a slave device
1 NA (shaded)	A NOT ACKnowledge condition sent from a slave device
P	A STOP condition sent by a bus master device. The STOP condition is not a bit and does not have a number 1 over it.
7 SLAVE ADDRESS	The first seven bits of the address byte, generally corresponding to the physical address of the device.
R	The bit [0] of the address byte with a value of 1, indicating the device is being addressed with a read.

Symbol	Meaning
W	The bit [0] of the address byte with a value of 0, indicating the device is being addressed with a write.
7 BROADCAST ADDRESS	The SMBus broadcast address to which all devices must respond. The value is 0000000b. This always used only with the bit [0] equal to 0 (write).
8 COMMAND CODE	A one byte value that indicates a command the slave device is to execute
8 LOW DATA BYTE	In a two byte value, the lower order byte (bits [7:0]).
8 HIGH DATA BYTE	In a two byte value, the higher order byte (bits [15:8]).
8 PEC	A byte with the Packet Error Check (PEC) value, if used.
● ● ●	The bit/byte/packet diagram is continued on the next line.

3.5. Abbreviations, Acronyms And Definitions

Term	Definition
ACK	ACKnowedge. The response from a receiving unit indicating that it has received a byte. See the SMBus specification [A03] for more information.
Assert, Asserted	A signal is asserted when the signal is true. For example, a signal called FAULT is asserted when a fault has been detected. See Negate.
AVS	Adaptive Voltage Scaling. AVS is used by a device to control its supply voltage, generally to minimize power consumption for a given operating condition.
AVSBus	AVSBus is an interface designed to facilitate and expedite point-to-point communication between an ASIC , FPGA, or other logic, memory, or processor devices and a POL control device on a system for the purpose of adaptive voltage scaling.
Bias, Bias Power	Power to the PMBus device's control circuit or ICs
Clear	When referring to a bit or bits, this means setting the value to zero.
Default Store	A non-volatile memory store most typically used by the PMBus device manufacturer to store default values
Disable, Disable Output	To instruct the PMBus device to stop the power conversion process and to stop delivering energy to the output. The device's control circuitry remains active and the device can communicate via the SMBus.
Enable, Enable Output	To instruct the PMBus device to start the power conversion process and to start delivering energy to the output.

Term	Definition
Host	A host is a specialized master that provides the main interface to the system's CPU. A host must be a master-slave and must support the SMBus host notify protocol. There may be at most one host in a system. See the SMBus specification [A03] for more information.
IIN	Input current
Inhibit	To stop the transfer of energy to the output while a give condition, such as excessive internal temperature, is present.
IOUT	Output current
LSB	Least significant bit
Master	A master is a device that issues commands, generates the clocks, and terminates the transfer. See the SMBus specification [A03] for more information.
MFR	Manufacturer
MSB	Most significant bit
NACK	Not ACKnowledge. The response from a receiving unit that it has received invalid data. See the SMBus specification [A03] for more information.
Negate, Negated	A signal is negated when the signal is false. For example, a signal called FAULT is negated when no fault has been detected. See Assert.
Negative Output Current	Current that flows into the converter’s output.
OC	Overcurrent
OP	Overpower
Operating Memory	The conceptual location where a PMBus maintains the data and parameters it uses operate.
OT	Overtemperature
OV	Overvoltage
PEC	Packet Error Checking. See the SMBus specification [A03] for more information.
PIN	Input power
Pin Programmed Values	Values entered into the PMBus device through physical pins. Values can be set, for example, by connecting a pin to ground, connecting a pin to bias power, leaving the pin unconnected or connecting the pin to ground or bias through a resistor.
Plain Text	Characters stored according to ISO/IEC 8859-1:1998 ([A05])
POL	Point-of-load
Positive Output Current	Current that flows out of the converter’s output.
POUT	Output power
Product Literature	Data sheets, product briefs, application notes or any other documentation describing the operation and application of a device.
Set	When referring to a bit or bits, this means setting the value to one.

Term	Definition
Shut Down	Disable or turn off the output. This generally implies that the output remains off until the device is instructed to turn it back on. The device's control circuit remains active and the device can respond to commands received from the SMBus port.
Sink (Current)	A power converter sinks current when current is flowing from the load into the converter's output. The current in this condition is declared to be negative.
Slave	A slave is a device that is receiving or responding to a command. See the SMBus specification [A03] for more information.
SMBus	System Management Bus – See the SMBus specification [A03] for more information.
Source (Current)	A power converter sources current when current is flowing from the converter's output to the load. The current in this condition is declared to be positive.
Turn Off	Turn Off means to "turn off the output", that is, stop the delivery of energy to the device's output. The device's control circuit remains active and the device can respond to commands received from the SMBus port. The same as Disable. See Turn On.
Turn On	Turn On means to "turn on the output", that is, start the delivery of energy to the device's output. The same as Enable. See Turn Off.
UC	Undercurrent (Excessive sink current by a synchronous rectifier)
User Store	A non-volatile memory store most often used by the PMBus device user to store an image, or snapshot, of the Operating Memory.
UT	Undertemperature
UV	Undervoltage
VIN	Input voltage
VOUT	Output voltage
X	When used to define a binary value X means that the value of that bit is "don't care".

4. Addressing And Grouping

4.1. Device Addresses

Individual PMBus devices are assigned a 7 bit address through a combination of manufacturer fixed bits and user assigned bits. This is described in the PMBus specification, Part I [A01].

4.2. General Call Address (Global Broadcast)

PMBus devices may respond to the General Call address (00h) as well as their own physical address.

4.3. Sending Commands To A Group

Commands may be sent to more than one PMBus device for simultaneous execution using the Group Command Protocol, as described in the PMBus specification, Part I [A01].

In any group command transaction, no more than one command can be sent to any one device.

4.4. ZONE_READ And ZONE_WRITE

PMBus devices may respond to a ZONE_READ to address 28h or a ZONE_WRITE to address 37h. Refer to the SMBus Specification [A03] and the PMBus specification, Part I [A01]. A device is configured for ZONE_READ and ZONE_WRITE using the ZONE_CONFIG (Section 11.16.1) and ZONE_ACTIVE 11.16.2 commands.

5. Commands

5.1. Commands And Command Codes

PMBus commands are one byte command codes. A listing of PMBus commands and their hexadecimal command codes are listed in APPENDIX I in Table 31.

Command codes are not register addresses in PMBus devices. The mapping of PMBus command codes to memory locations in a PMBus device is left to the PMBus device manufacturer.

5.2. Command Extensions

To provide more than the 256 commands possible with a one byte command code, the PMBus provides for two “command extensions” (Section 25). One of these extensions is made available to PMBus device manufacturers for manufacturer specific commands. The other is reserved for the future inclusion in the PMBus specifications.

The Command code extensions essentially make the command code two bytes long. The first byte transmitted is the Command Code Extension command code. The second byte of the extended command code identifies the action the PMBus device is to take.

These two command extensions use the Command Extension Protocols described in the PMBus specification, Part I [A01].

5.3. Command Execution

PMBus devices are to process and execute commands as soon as possible after the STOP condition is recognized. PMBus devices do not wait for a separate “Execute” command that launch the previously received command.

5.4. Writing And Reading PMBus Devices

5.4.1. All Packets Start With A Write Address

A device’s address byte that follows a START Condition (not the Repeated Start Condition or the SMBALERT# Response Address!) must always have bit [0] with a value of 0 (indicating a write).

If a device receives its own address in a byte directly after a START Condition (not a Repeated START Condition) with bit [0] equal to 1, then the device responds as described in Section 10.9.1.

5.4.2. Every Parameter That Can Be Written Must Be Readable

In general, any command that accepts a value for writing must also return that value when read. This can be used by the host to provide assurance that a transmitted value was received correctly. There are some exceptions.

Commands related to parametric values, such as VOUT_COMMAND or TON_DELAY may store the received value with fewer bits than the data format allows. For example, the data for the VOUT_COMMAND command is a 16 bit unsigned binary integer. The device, however, may store that as only a 12 bit integer. Reading the VOUT_COMMAND would then return the 12 bit value, not the 16 bit value. This is not considered an error.

The status commands (Section 17) behave different when reading and writing. When reading, a status command will return the results of that status register. When writing to a status command, the data byte is used to clear one or more bits in that register (Section 10.2.3).

5.4.3. Commands May Be Read Only

Not all commands must support writing parameters into a PMBus device. Some commands, such as those that read back parameters like output voltage, are inherently read only. PMBus device manufacturers may also make some commands available for reading, but not for writing. Examples might be the VOUT_MODE command (which sets the format of output voltage commands) and commands related to inventory information, such as MFR_MODEL (which can be used to retrieve the manufacturer’s model number).

6. Memory Model, Startup Behavior And Defaults

At the conceptual level, PMBus devices operate from values, such as the commanded output voltage, stored in volatile memory. This volatile memory, for purposes of describing the conceptual operation of a PMBus device, is called the Operating Memory. When bias power is applied and the PMBus device control circuitry starts operating, the Operating Memory is loaded from one or more of the following places:

- Values hard coded into an IC design (if any),
- Values programmed from hardware pins (if any),
- A non-volatile memory called the Default Store (if supported in the device),
- A non-volatile memory called the User Store (if supported in the device), or
- Communications from the SMBus.

The relationships between the conceptual Operating Memory and each of the possible sources for loading the Operating Memory are illustrated in Figure 2.

6.1. Order Of Memory Loading And Precedence

To illustrate the precedence of loading parameters into the conceptual Operating Memory, this section uses the conceptual model shown in Figure 2 and Figure 3. This model, and the discussion in this section, are only to illustrate the precedence of how parameters are set within the PMBus device. Any implementation is acceptable so long as it preserves the precedence described in this section.

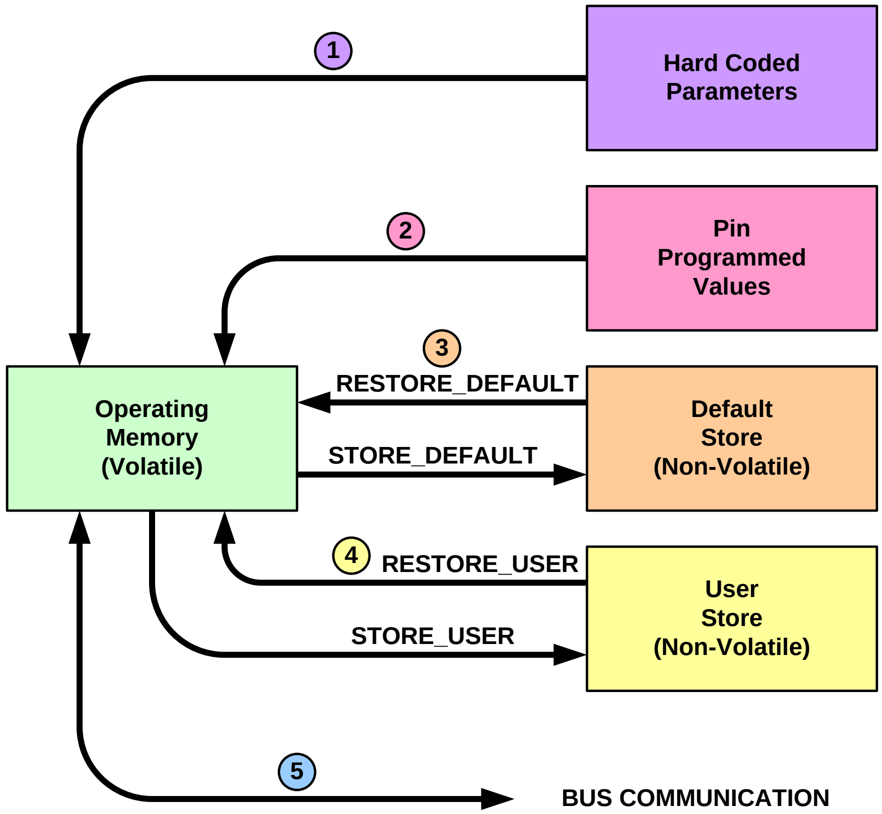


Figure 2. Conceptual View Of Possible PMBus Device Memory And Communication

The first parameters loaded into the Operating Memory are any hard coded parameters.

The second parameters loaded into the Operating Memory come from the pin programming. If any of the parameters programmed by the pins are the same as a parameter that was hard coded, the pin programmed value overwrites the previously loaded hard coded value.

This is the general rule: When parameters are loaded, they will overwrite the same parameter that is already in the Operating Memory.

The third set of parameters loaded comes from the optional non-volatile Default Store, if it exists. The values in the Default Store are usually programmed by the PMBus device manufacturer. The device manufacturer may or may not allow the user to overwrite the manufacturer provided values in the Default Store.

The fourth set of parameters loaded comes from the optional non-volatile User Store, if it exists. The User Store is most often used to store a “snapshot” of the Operating Memory once a device has been programmed and adjusted for operation. By storing a copy of the Operating Memory in the User Store, a device will resume operation with the last set of values stored by the User.

And finally, once the previous steps have finished, the PMBus device will start accepting commands from the SMBus. Note that this means that values written from the bus will overwrite all previous values, including those that were hard coded, pin programmed or copied from the Default and User Stores.

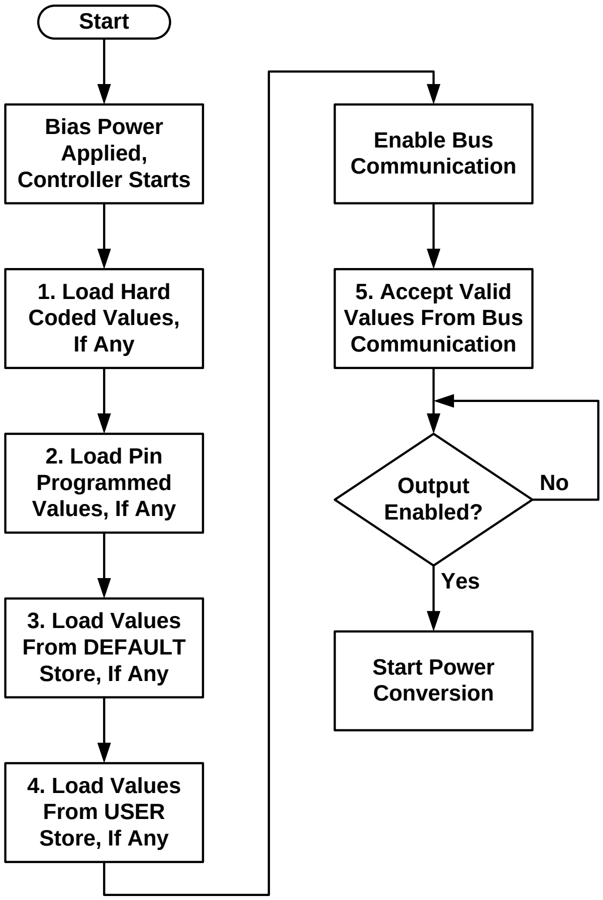


Figure 3. Flowchart Of Conceptual Loading Operating Memory At Startup

6.2. The Default And User Stores

The Default Store and User Store are optional.

Four commands are provided to manipulate the contents of these two non-volatile memory stores.

To copy the entire contents of Operating Memory into the Default Store, the STORE_DEFAULT_ALL command (Section 11.2) is used. To store just one parameter in the Default Store, the STORE_DEFAULT_CODE command (Section 11.4) is used. PMBus device manufacturers may not permit these operations. If STORE_DEFAULT_ALL or STORE_DEFAULT_CODE are permitted, they may generally be commanded when the PMBus device is operating and supplying power to the output. However, this may result in unpredictable and even catastrophic results. It is recommended that the output be disabled before issuing a STORE_DEFAULT_ALL or STORE_DEFAULT_CODE command.

To copy the entire contents of the Default Store into Operating Memory, the RESTORE_DEFAULT_ALL command (Section 11.3) is used. To copy just one parameter from the Default Store to Operating Memory, the RESTORE_DEFAULT_CODE command (Section 11.5) is used. These commands may generally be executed while the device is operating, but can result in unpredictable and even catastrophic results. It is recommended that the output be disabled before issuing a RESTORE_DEFAULT_ALL or RESTORE_DEFAULT_CODE command.

To copy the entire contents of Operating Memory into the User Store, the STORE_USER_ALL command (Section 11.6) is used. To store just one parameter in the User Store, the STORE_USER_CODE command (Section 11.8) is used. The STORE_USER_ALL or STORE_USER_CODE commands may generally be issued when the PMBus device is operating and supplying power to the output. However, this may result in unpredictable and even catastrophic results. It is recommended that the output be disabled before issuing a STORE_USER_ALL or STORE_USER_CODE command.

To copy the entire contents of the User Store into Operating Memory, the RESTORE_USER_ALL command (Section 11.7) is used. To copy just one parameter from the User Store to Operating Memory, the RESTORE_USER_CODE command (Section 11.9) is used. These commands may be generally be executed while the device is operating and supplying power to the output, but this can result in unpredictable and even catastrophic results. It is recommended that the output be disabled before issuing a RESTORE_USER_ALL or RESTORE_USER_CODE command.

7. Numeric Data Formats

7.1. Summary

PMBus devices may use one of several formats for numeric data depending on the requirements for device simplicity versus computational load on the system host and the range and resolution of the data being transmitted.

The LINEAR11 format provides a wide range of positive and negative values with good (10 bit) resolution. It may be used with commands that report values other than those related to the output voltage. For example, the LINEAR11 format may be used to report input voltage, output current, or temperature. The LINEAR11 format balances the computational and formatting burden between the PMBus device and the system host.

The ULINEAR16 format is used only with values related to the output voltage. It provides a wide range with fine resolution but is restricted to positive values.

The DIRECT format places the entire computation burden on the system host. For example, when the host reads data in the DIRECT format the host is getting essentially the output of an analog to digital (A2D) converter. In order to properly interpret data in the DIRECT format, the host will need to know the appropriate scaling factors and offset term to convert the raw binary value into a “real world value”.

The IEEE Half Precision Floating Point format provides the advantage of a standard format compatible with code written for system management processors. It also provides a finer resolution than the LINEAR11 format but with a narrower range of maximum and minimum values.

The IEEE Single Precision Floating Point format, with 32 bits, may be used in manufacturer specific commands.

Any parameters that do not use any of these formats have their data format described explicitly in the section describing the command that receives or transmits that parameter.

The product literature for each PMBus device shall describe which data format is used for each PMBus command the device supports.

7.2. Restrictions

If a PMBus device uses the IEEE Half Precision Floating Point Format for numerical data then it must use only the IEEE Half Precision Floating Point Format. This applies to commands both related, and unrelated, to the output voltage.

Conversely, if a PMBus device uses the LINEAR11, ULINEAR16, SLINEAR16, or Direct formats for any numerical data, then it may not use the IEEE Half Precision Floating Point Format for any command.

7.3. LINEAR11 Numeric Format

The LINEAR11 Numeric Format is typically used for commanding and reporting the parameters such as (but not only) the following:

- Output Current,
- Input Voltage,
- Input Current,
- Operating Temperatures,
- Time (durations), and
- Energy Storage Capacitor Voltage.

The LINEAR11 Numeric Format is a two byte value with:

- An 11 bit, two's complement mantissa and
- A 5 bit, two's complement exponent (scaling factor).

The format of the two data bytes is illustrated in Figure 4.

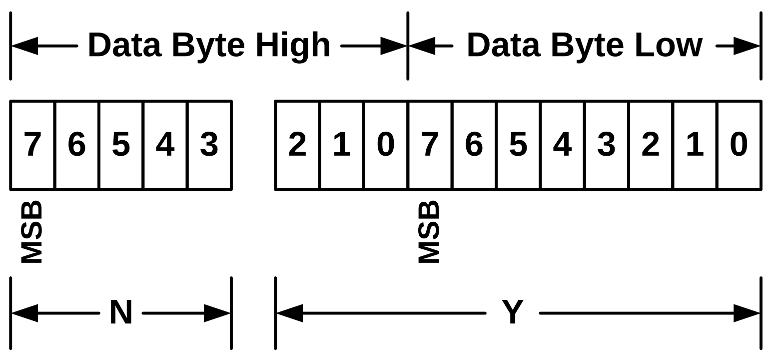


Figure 4. LINEAR11 Numeric Format Data Bytes

The relation between *Y*, *N* and the "real world" value is:

$X = Y \cdot 2^N$

Where, as described above:

X is the "real world" value;

Y is an 11 bit, two's complement integer; and

N is a 5 bit, two's complement integer.

Devices that use the LINEAR11 format must accept and be able to process any value of *N*.

7.4. DIRECT Data Format

If a PMBus device uses DIRECT form data, this shall be clearly described in the product literature.

7.4.1. Interpreting Received Values

The host system uses the following equation to convert the value received from the PMBus device into a reading of Volts, Amperes, degrees Celsius or other units as appropriate:

$$X = \frac{1}{m}(Y \times 10^{-R} - b)$$

Where:

X, is the calculated, "real world" value in the appropriate units (A, V, °C, etc.);

m, the slope coefficient, is a two byte, two's complement integer;

Y, is a two byte two's complement integer received from the PMBus device;

b, the offset, is a two byte, two's complement integer; and

R, the exponent, is a one byte, two's complement integer.

7.4.2. Sending A Value

To send a value, the host must use the equation in Section 7.4.1 solved for *Y*:

$$Y = (mX + b) \times 10^{R}$$

Where:

Y is the two byte two's complement integer to be sent to the unit;

m, the slope coefficient, is the two byte, two's complement integer;

X, a "real world" value, in units such as Amperes or Volts, to be converted for transmission;

b, the offset, is the two byte, two's complement integer; and

R, the exponent, is the decimal value equivalent to the one byte, two's complement integer.

7.4.3. Obtaining The Value Of The m, b, And R Coefficients

Before a host either sends information to or retrieves information from a PMBus device using DIRECT mode, it must know the value of the *m*, *b* and *R* coefficients. These values may either be:

- Retrieved from the device using the COEFFICIENTS command (Section 14.1)or
- Supplied by the device's manufacturer in the product literature. In this case, the host device must store the coefficients for all commands of interest.

Note that for a given parameter, such as output voltage, the coefficients used to set the value and to read the value may not be the same.

7.5. IEEE-754 Floating Point

7.6. IEEE-754 Half Precision Floating Point

Commands with 16 bits of numerical data, such as READ_VOUT, may use the IEEE-754 half precision floating point representation. Bit [15] is a sign bit, bits [14:10] are the exponent, and bits [9:0] are the mantissa. The details of the format are given in [R01].

For a PMBus transaction, the 16 bits are distributed within the high and low bytes as shown in Figure 5. The standard PMBus data format transmission rules of low byte first, most significant bit first are followed.

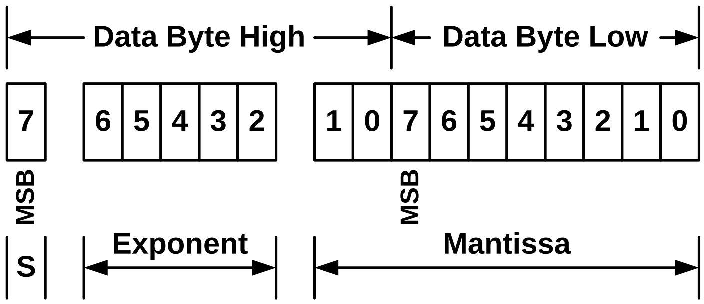


Figure 5. Format Of Floating Point Format Data Bytes

7.6.1. IEEE-754 Single Precision Floating Point Format

The 32 bit IEEE-754 Single Precision Floating Point format [R01] is one of two possible formats for the READ_KWH_IN and READ_KWH_OUT commands (Section 18.14). In this revision of the PMBus specification, no other standard commands use the Single Precision Floating Point Format.

PMBus device manufacturers may use the IEEE Single Precision Floating format in the manufacturer specific commands.

7.6.2. Floating Point Value Restrictions

PMBus devices that use either IEEE-754 Half Precision Floating Point or Single Precision format shall comply with the following restrictions:

- When the system host reads a value it has written to a PMBus device, the device shall return the exact IEEE-754 encoding that the master sent, including the NaN (Not A Number), +Inf, or -Inf encodings.
- If a PMBus device receives the NaN value when expecting a numeric value, the PMBus device shall treat this as invalid data, declare a communications fault, and respond as described in Section 10.8.
- PMBus devices shall interpret a received +Inf as a positive full scale.
- PMBus devices shall interpret a received –Inf as a negative full scale.
- PMBus devices may return NaN if the value is not available.
- PMBus devices shall return +Inf if the measurement channel was saturated in the positive direction.
- PMBus devices shall return -Inf if the measurement channel was saturated in the negative direction.

7.7. Manufacturer Specific Numeric Data Formats

For the Manufacturer Specific commands, the device manufacturer may specify any numeric format they choose. For example, for data requiring high precision or a very large range, manufacturers might specify IEEE-754 Single or Double Precision Floating Point format. Any manufacturer specific numeric data format must be fully described in the device product literature.

7.8. Accuracy

The accuracy of commanded and reported data shall be given in the PMBus device's product literature.

7.9. Resolution

PMBus devices may have an internal data resolution less than the transmitted value. For example, VOUT_COMMAND sends 16 bits in its data bytes. Yet a PMBus device might use only 10 of the 16 in commanding an output voltage. This is permitted and considered compliant.

When reading back information from a PMBus that uses a native resolution less than the number of bits used in the write version of the command, it is permissible for the PMBus device to return zero values for the lower order bits it does not support. In the example about, with the 10 bit resolution for output voltage, using the SMBus Read Word protocol with the VOUT_COMMAND command code would return the 10 highest order bits that were sent to the device. The six lowest order bits would be all zeros regardless of what was sent to the device with the original SMBus Write Word command with the VOUT_COMMAND command code. This behavior is considered compliant.

8. Data Formats For The Output Voltage And Output Voltage Related Parameters

Voltage data for commanding or reading the output voltage or related parameters (such as the overvoltage threshold) can be in one of four different formats depending on the type of device. PMBus device product literature shall clearly identify which of the formats the device is capable of supporting.

The formats for commanding and reporting voltage are:

- The ULINEAR16 and SLINEAR 16 formats use a two byte binary integer with a scaling factor (similar in concept to a mantissa and exponent),
- A Half-Precision Floating Point format that follows the IEEE-754 standard for representing magnitudes in 16 bits,
- A format that supports transmitting the VID codes of popular microprocessors via the PMBus, and
- The DIRECT format (7.4) that uses an equation and device supplied coefficients.

8.1. Restrictions

8.1.1. Positive And Negative Output Voltages

When using the ULINEAR16 or Direct Formats for output voltage related commands, the values are restricted to positive values as the data is unsigned. When using a IEEE-754 Floating Point Format, the voltages may be commanded as either positive or negative.

The restriction on only positive values for the ULINEAR16 and Direct Formats is not severe. Many power supplies and power converters are provided with the output(s) not referenced to a reference or ground. Such devices do not inherently have positive or negative output voltages. The end user creates positive or negative outputs when one terminal of the output is connected to a reference or ground. That is, output voltage related PMBus commands relate only to the difference between the most positive terminal and the most negative terminal, no matter which is connected to reference or ground. With the knowledge of which terminal is connected to reference or ground, the system power manager can manage negative outputs.

8.1.2. Floating Point Format

If a PMBus device uses the IEEE Floating Point Format for numerical data then it must use only the IEEE Floating Point Format. This applies to commands both related, and unrelated, to the output voltage.

Conversely, if a PMBus device uses any of the ULINEAR16 or Direct formats for any numerical data, then it may not use the IEEE Floating Point Format for any command.

8.2. Two Step Process

Commanding or reading an output voltage or output voltage related parameter requires two steps.

The first step is to set or read which of the allowable formats (ULINEAR16, Half-precision IEEE-754, VID, DIRECT) the device uses for output voltage related data. This is done with the VOUT_MODE command (Section 8.3).

The VOUT_MODE command is only issued when the format of the output voltage data changes. For some devices, this may be written only once in the device's life.

After the VOUT_MODE command is used to set or read the format of the output voltage data, other commands are used to set, adjust or read back output voltage related information. For example, the VOUT_COMMAND is used to set the voltage to which the device should set the output. The VOUT_OV_FAULT_LIMIT command is used to set the output overvoltage fault threshold.

8.3. VOUT_MODE Command

8.3.1. Mode Selection

The data byte for the VOUT_MODE command is one byte that consists of a three bit Mode and a five bit Parameter as shown in Figure 6. The three bit Mode sets whether the device uses the ULINEAR16, Half-precision IEEE 754 floating point, VID or DIRECT modes for output voltage related commands. The five bit Parameter provides more information about the selected mode, such as which manufacturer's VID codes are being used.

Sending the VOUT_MODE command with the address set for writing sets the Mode and Parameter into the PMBus device, if it accepts changes to these values.

PMBus devices may have the Mode and Parameter set at the time of manufacture and may not permit the user to change these values. In this case, if a host sends a VOUT_MODE command for a write to a PMBus device, the device shall reject the VOUT_MODE command, declare a communication fault for invalid data, and respond as described in section 10.2.2.

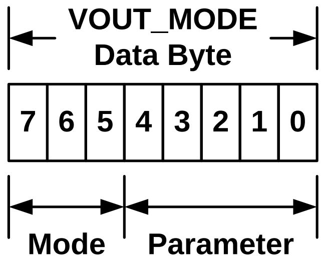


Figure 6. VOUT_MODE Command Data Byte Structure

If a device accepts the VOUT_MODE command, the Mode and Parameter are retained until changed with another VOUT_MODE command or until the bias power is removed.

Sending the VOUT_MODE command using the SMBus Read Byte protocol returns one byte with the Mode and Parameter as shown in Figure 6.

Table 2 shows the permitted values and format of the VOUT_MODE data byte. More information on the VOUT_MODE command is used with output voltage related commands is given below in Section 8.4.

Table 2. Summary Of The VOUT_MODE Data Byte Format

Mode	Bit [7]	Bits [6:5]	Bits [4:0] (Parameter)
ULINEAR16 [Note 1]	X	00b	Five bit two's complement exponent for the mantissa delivered as the data bytes for an output voltage related command.
VID	X	01b	Five bit VID code identifier per Table 3
Direct	X	10b	Always set to 00000b
IEEE Half Precision Floating Point	X	11b	Always set to 00000b
Absolute	0	XX	XXXXXb
Relative	1	XX	XXXXXb

Note 1: When the VOUT_COMMAND uses the ULINEAR16 format this setting for VMODE shall be used. It is understood that in this case some output voltage related commands, such as VOUT_TRIM, will use the SLINEAR16 format.

8.4. Data Bytes For Output Voltage Related Commands

There are several commands that either set or adjust the output voltage, or a related parameter, of a device that supports the PMBus protocol. Some examples are:

- VOUT_COMMAND which causes the device to set its output voltage to the commanded value;
- VOUT_TRIM, which is available to the device user to trim the output voltage; and
- VOUT_OV_FAULT_LIMIT, which sets the output voltage above which an output overvoltage fault is declared.

All output voltage related commands use two data bytes. The contents of those data bytes depend on the voltage data format in use (set by the VOUT_MODE command) and are described below.

8.4.1. LINEAR16 Formats

8.4.1.1. ULINEAR16 Format

Commands that directly set the output voltage, such as VOUT_COMMAND, use the ULINEAR16 format, which is a 16 bit unsigned integer.

The data bytes for the VOUT_MODE and VOUT_COMMAND when using the ULINEAR16 voltage data format are shown in Figure 7.

Note that the VOUT_MODE command is sent separately from output voltage related commands and only when the output voltage format changes. VOUT_MODE is not sent every time an output voltage command is sent.

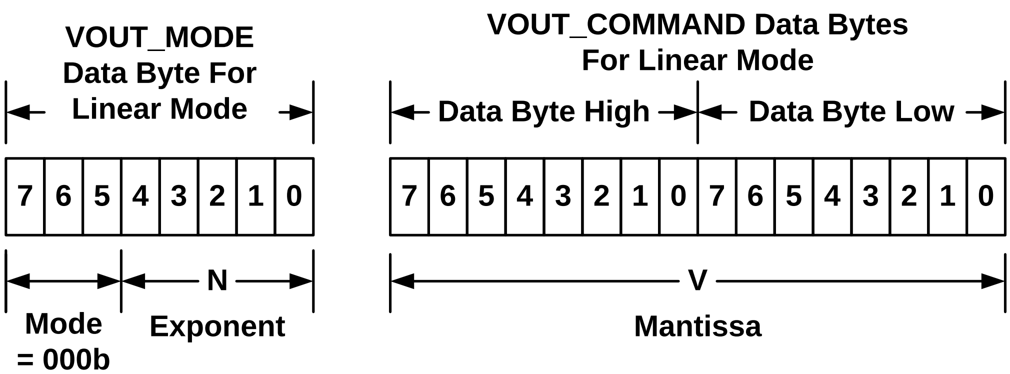


Figure 7. ULINEAR16 Format Data Bytes

The Mode bits are set to 000b.

The Voltage, in Volts, is calculated from the equation:

$$Voltage = V \times 2^N$$

Where:

Voltage is the parameter of interest in Volts;

V is a 16 bit unsigned binary integer; and

N is a 5 bit two's complement binary integer.

8.4.1.2. SLINEAR16 Format

Commands that add or subtract from the output voltage, such as VOUT_TRIM, use the SLINEAR16 format, which is a 16 bit two's complement integer.

8.4.2. VID Format

The data bytes for the VOUT_MODE and VOUT_COMMAND when using the VID voltage data format are shown in Figure 8. Note that the VOUT_MODE command is sent separately from output voltage related commands and only when the output voltage format changes. VOUT_MODE is not sent every time an output voltage command is sent.

The Mode bits are set to 001b. The VID Code Type is an unsigned binary integer. The defined values of VID Code Type are given below in Table 3. Any VID Code Types not listed in Table 3 are reserved for future use and shall not be used until listed in a future revision of this specification.

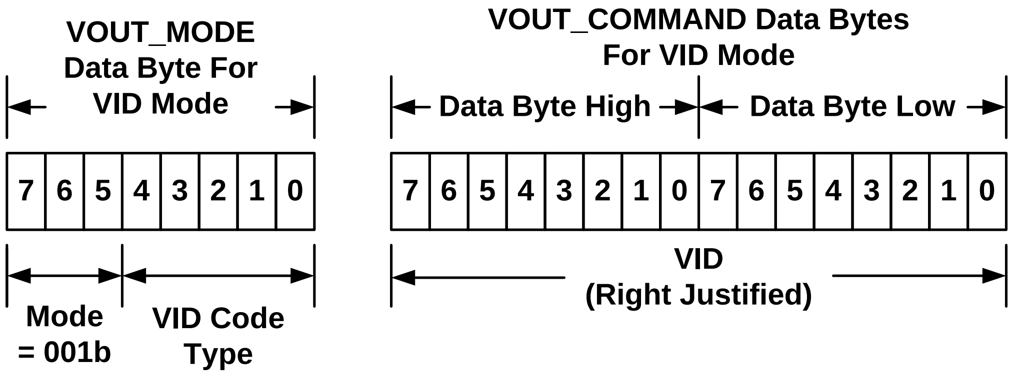


Figure 8. VID Format Data Bytes

Table 3. VID Types Supported By PMBus

VID Code Type	Microprocessor Family
00h	Not Used
01h	Reserved For A Future Generation Intel Microprocessor
02h	Reserved For A Future Generation Intel Microprocessor
03h	Reserved For A Future Generation Intel Microprocessor
04h	Reserved For A Future Generation Intel Microprocessor
10h	Reserved For A Future Generation AMD Microprocessor
11h	Reserved For A Future Generation AMD Microprocessor
1Ch	Reserved For Future Use
1Dh	Reserved For Future Use
1Eh	PMBus Device Manufacturer Specific
1Fh	PMBus Device Manufacturer Specific

VID Code Types 1Eh and 1Fh are provided so that PMBus device makers can provide customized or manufacturer specific VID codes. The details of the relationship between the VID codes and output voltage shall be provided in the PMBus device product literature.

Within the output voltage related command data bytes, the VID code shall be right justified with VID0 in bit 0 of the lower data byte, VID1 in bit 1 of the lower byte and so forth until all applicable VID bits are used. Any unused bits in the data bytes shall be filled with zeroes.

8.4.3. DIRECT Format

The DIRECT data format can also be used to command or read output voltage related values. See Section 7.4 for the details on this data format is used.

When the DIRECT format is used to set the output voltage, the coefficients *m*, *b* and *R* are generally chosen by the PMBus device manufacturer so that the minimum voltage to be commanded results in a value of 0 for *Y*. The result of the equation for the maximum value to be commanded generally results in a value of 2^{16}-1. The result of the calculation is converted to a 16 bit unsigned binary integer and transmitted as the data bytes of a VOUT_COMMAND command.

The *Y* shown in the VOUT_COMMAND data byte in Figure 9 is the value used in conjunction with the coefficients *m*, *b* and *R* to calculate the desired value. See Section 7.4 for the details.

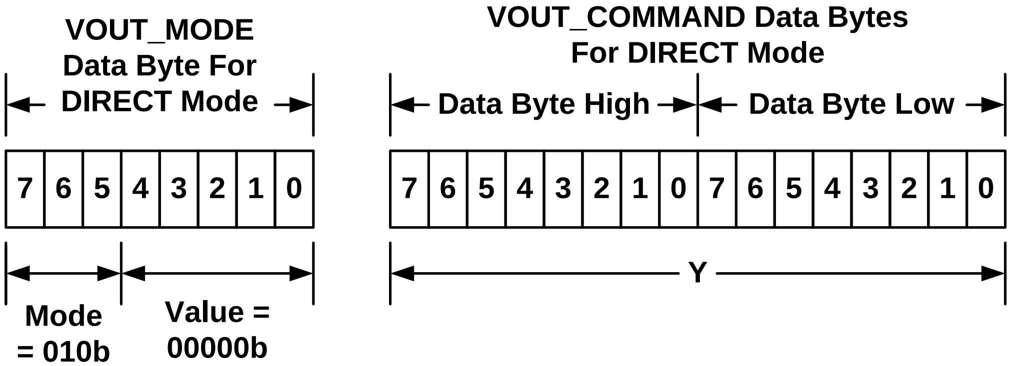


Figure 9. DIRECT Format Mode Data Bytes

8.4.4. IEEE-754 Half Precision Floating Point

The IEEE-754 Half Precision Floating Format, as described in Section 7.5, may also be used for output voltage related commands.

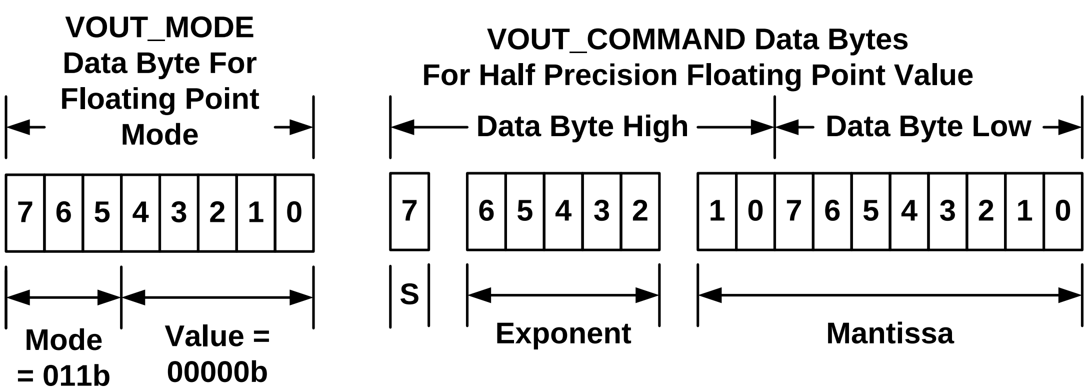


Figure 10. Floating Point Format Mode Data Bytes

8.5. Absolute Value And Relative Value Voltage Related Commands

For many output voltage related commands, such as the commands that set the margin limits, the user may want to set them to an absolute value or a percentage of the nominal output voltage (value set by VOUT_COMMAND). The Absolute and Relative bit is used to set this behavior. Setting or clearing this bit sets the Absolute or Relative value setting for all of the commands below. It is not permitted to mix Absolute and Relative value mode within the same PMBus device.

The commands for which this option can be set are:

- VOUT_MARGIN_HIGH
- VOUT_MARGIN_LOW
- VOUT_OV_FAULT_LIMIT
- VOUT_OV_WARN_LIMIT
- VOUT_UV_WARN_LIMIT

- VOUT_UV_FAULT_LIMIT
- POWER_GOOD_ON
- POWER_GOOD_OFF

8.5.1. Absolute Value Mode Data Bytes

If the commands listed above are set for Absolute Value mode, then these commands directly set the voltage level for that command. The data bytes of the commands are two bytes that may be in any of the formats permitted in Section 8.4 as permitted by the current setting of the VOUT_MODE command.

8.5.2. Relative Value

If these commands are set as Relative, then as the nominal value changes, the value of these commands will change in proportion (“tracking”).

If the commands listed above are set to Relative Value mode, the data bytes are in the same format as VOUT_COMMAND,interpreted with the current setting of the VOUT_MODE command.

In Relative Value mode, the voltage associated with a command, such as VOUT_MARGIN_HIGH, is the Relative value multiplied by the value set by VOUT_COMMAND (the nominal value). The relative value is always a positive value.

As an example, consider the case where:

- The nominal output voltage is 3.3 V
- The ULINEAR16 format is used for VOUT_COMMAND
- In VOUT_MODE, N is set to -10 for an output voltage resolution of 2^{-10} V/bit (= 977 µV/bit).
- It is desired to margin the nominal voltage up to 110% of the nominal value and down to 90% of the nominal value.

The value of the data for the VOUT_COMMAND command for the 3.3 V nominal output is:

$$V\_OUT\_COMMAND_{NOMINAL} = round\left(\frac{V_{OUT\_NOMINAL}}{2^{-N}}\right) = round\left(\frac{3.3}{977 \times 10^{-6}}\right) = 3379\text{d} = \text{ODO1h}$$

The desired Relative values for the VOUT_MARGIN_HIGH command is:

$$MARGIN\_HIGH\_RELATIVE = \frac{110\%}{100\%} = 1.1$$

The desired Relative value for the VOUT_MARGIN_LOW command is:

$$MARGIN\_LOW\_RELATIVE = \frac{90\%}{100\%} = 0.9$$

The Relative value for the VOUT_MARGIN_HIGH and VOUT_MARGIN_LOW commands must now be expressed using same scaling as used for the output voltage. The value of the data for the VOUT_MARGIN_HIGH command is found by:

$$VOUT\_MARGIN\_HIGH = round\left(\frac{MARGIN\_HIGH\_RELATIVE}{2^{-N}}\right)$$
$$= round\left(\frac{1.1}{977\times10^{-6}}\right)$$
$$= 1126\text{d} = 0466\text{h}$$

The output voltage would then be margined high by sending the VOUT_MARGIN_HIGH command with the data value 1126d (= 0466h) followed by the OPERATION command with bits [5:4] set to 10b. Due to the quantization of the Relative value the actual relative value for the margin high operation is not exactly 1.1:

$$MARGIN\_HIGH\_RELATIVE_{ACTUAL} = 1126\text{d}\times 2^{-N} = 1.0996$$

The value of the data for the VOUT_MARGIN_LOW command is found by:

$$VOUT\_MARGIN\_LOW = round\left(\frac{MARGIN\_LOW\_RELATIVE}{2^{-N}}\right)$$
$$= round\left(\frac{0.9}{977\times10^{-6}}\right)$$
$$= 922\text{d} = 039\text{Ah}$$

The output voltage would then be margined low by sending the VOUT_MARGIN_LOW command with the data value 922d (= 039Ah) followed by the OPERATION command with bits [5:4] set to 01b. Due to the quantization of the Relative value the actual relative value for the margin low operation is not exactly 0.9:

$$MARGIN\_LOW\_RELATIVE_{ACTUAL} = 922\text{d}\times 2^{-N} = 0.90039$$

8.5.3. Relative Value Mode Notes

The relative values for the output voltage warning limits, output voltage fault limits, and power good signal limits are not affected by the use of an output voltage droop characteristic. The margin limits can be affected by the output voltage droop setting. See Sections 9.2 and 13.9 for more information.

The Relative Value option is not available when a VID format is used to set the output voltage.

One consequence of Relative mode is that it is not possible to set an output voltage that will trigger an over or undervoltage warning or fault as the thresholds will change with nominal value set by the VOUT_COMMAND command.

9. Setting And Monitoring The Output Voltage And Current

There are several commands that affect how a PMBus device responds to output voltage related commands. This section provides a conceptual description of how those commands work. The actual implementation is left to the PMBus device manufacturers.

9.1. VOUT_SCALE_LOOP And VOUT_SCALE_MONITOR

In typical devices the output voltage is sensed through a resistive voltage divider, as illustrated in Figure 11. The resistive divider reduces, or scales, the output voltage so that when the output voltage is correct, the value supplied to the control circuit is equal to the reference voltage.

Many devices supporting the PMBus protocol will have a resistive voltage divider between the output and the input to the device's control circuit or IC. However, commands sent over the PMBus command the output voltage, not the reference voltage. To allow PMBus devices to map between the commanded voltage (such as 3.3 V), and the voltage at the control circuit input (perhaps 3.3 V divided down to match a reference voltage of 1.2 V), the VOUT_SCALE_LOOP (Section 13.10) command is used.

Figure 11 shows a conceptual view of how the VOUT_SCALE_LOOP command works. The output voltage, VOUT, is processed through a resistive divider with a ratio of output to input equal to K_R. Suppose, for example, the output voltage was 3.3 V and that the desired input to the PMBus device is 1.2 V. Then K_R is calculated as follows:

$$K_R = \frac{1.2\text{V}}{3.3\text{V}} = 0.3636...$$

The PMBus device needs to take account of the external resistive divider when processing output voltage related commands. The simplest concept is simply to think of the voltage command being scaled by the same amount as the actual output voltage. This shown by the 16 bit VOUT_COMMAND being applied to a gain block labeled as VOUT_SCALE_LOOP. If the gain of that block, K, is the same as the resistive divider ration, K_R, then in concept, the values applied to the control circuitry from the output voltage sensing network and the voltage command input, will be the same when the output is at the desired value.

This discussion illustrates the concept and use of the VOUT_SCALE_LOOP Command for setting the output voltage and output voltage related values. PMBus device users are instructed to consult the PMBus device manufacturer's product literature for information on how this command is implemented in any devices of interest.

In devices that provide an independent path for sensing the output voltage, such as for the output overvoltage protection circuit or the circuit that processes the sensed output voltage for the READ_VOUT command, a second scale factor, VOUT_SCALE_MONITOR (Section 13.11), is provided. This scale factor, in concept, works the same as the VOUT_SCALE_LOOP command.

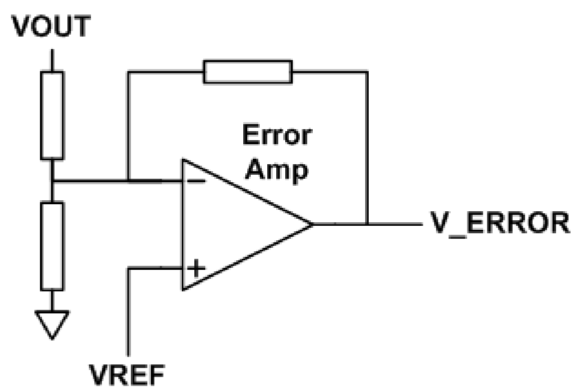


Figure 11. Output Voltage Sensing In A Typical Power Converter

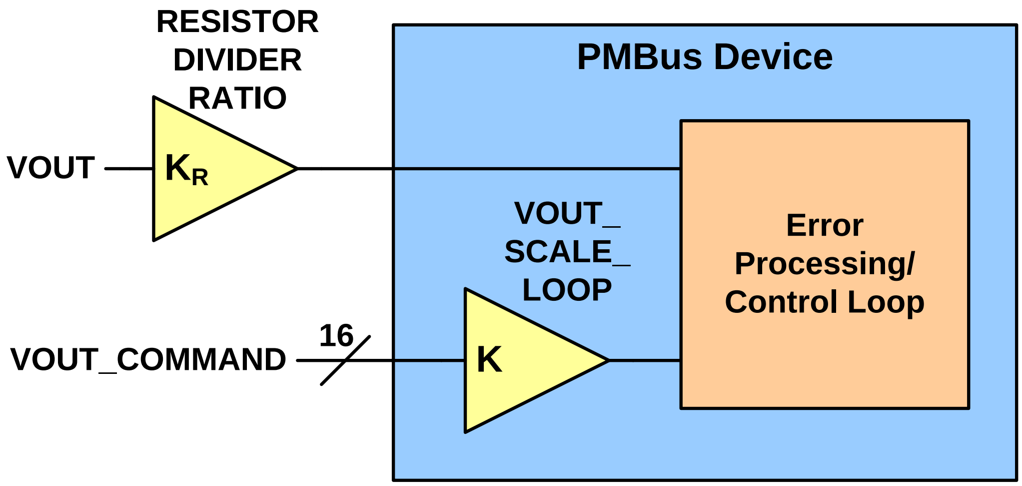


Figure 12. Conceptual View Of The Application Of The VOUT_SCALE_LOOP Command

When generating the value reported in response to the READ_VOUT command, the sensed value should be divided by the value of VOUT_SCALE_MONITOR. For example, using the same resistor divider ratio as above (0.3636…), a voltage at the monitoring pin of 1.25 V would result in the value 3.41 being returned in response to a VOUT_READ command.

For monitoring the output for overvoltage, the value set by the VOUT_OV_FAULT_LIMIT command should be multiplied by VOUT_SCALE_MONITOR, and the result of that calculation compared to the voltage at the sense pin. Continuing the example above, suppose the desired overvoltage fault threshold is 3.63 V (3.3 V + 10%). This is commanded by the VOUT_OV_FAULT_LIMIT command. Then a voltage of 1.32 V (VOUT_OV_FAULT_LIMIT × VOUT_SCALE_MONITOR = 3.63 V × 0.3636) at the monitoring pin would trigger an overvoltage fault.

PMBus device users are directed to the manufacturer's literature for information on how the VOUT_SCALE_COMMAND is used in any devices of interest.

9.2. Setting The Output Voltage

There are several commands that are used in commanding the output voltage of a device with a PMBus interface. These include:

- VOUT_MODE (Section 8.3),
- VOUT_COMMAND (Section 8),
- AVSBus Target Rail Voltage command (PMBus Specification, Part III, [A02])
- VOUT_TRIM (Section 13.3),
- VOUT_CAL_OFFSET (Section 13.4),
- VOUT_MAX (Section 13.5),
- VOUT_MIN (Section 13.12),
- VOUT_MARGIN_HIGH (Section 13.6),
- VOUT_MARGIN_LOW (Section 13.7),
- VOUT_DROOP (as a function of IOUT) (Section 13.9), and
- VOUT_SCALE_LOOP (Sections 13.10 and 9.1).

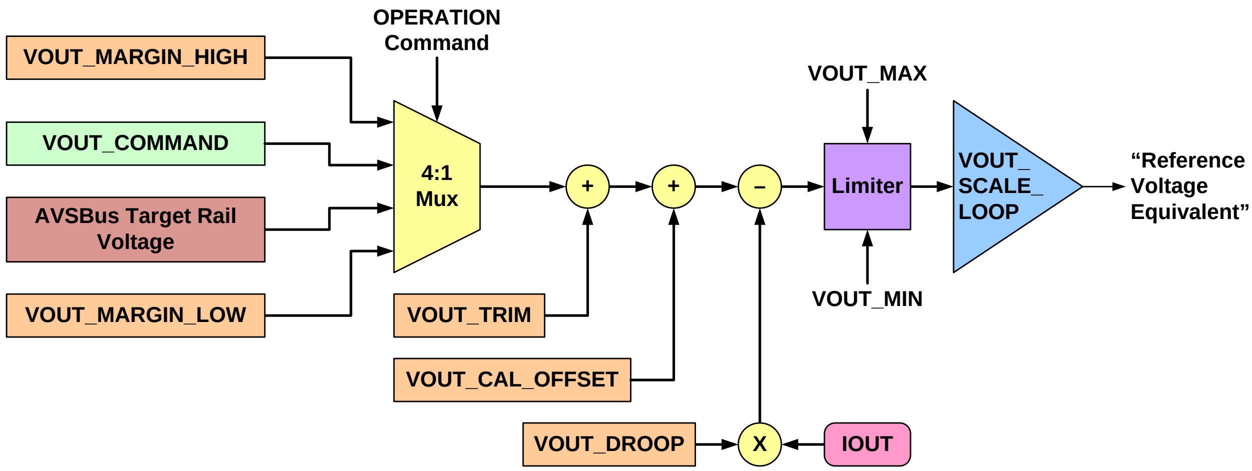


Figure 13. Conceptual View Of How Output Voltage Related Commands Are Applied

Figure 13 shows a conceptual view of how these commands are used to control the output voltage. The actual implementation is left to the PMBus device makers so long as the overall behavior is the same as shown in Figure 13.

In Figure 13, the values of the various parameters may come from:

- Hard coded values embedded in the PMBus device,
- Pin programming,
- The conceptual non-volatile Default Store,
- The conceptual non-volatile User Store, or
- Commands received from the SMBus port.

This process of loading parameters was described in Section 6.

The process of setting the output voltage starts with selecting one of four inputs as the source for the nominal voltage of the output: VOUT_COMMAND, VOUT_MARGIN_HIGH, VOUT_MARGIN_LOW, or the AVSBus Target Voltage. One of these four values is selected by the OPERATION command (Section 12.1) and passed on to the rest of the output voltage command processing.

The next step is to add the value in the VOUT_TRIM register to the output of the conceptual multiplexer. The value in the VOUT_TRIM register is a two's complement number that can either add to or subtract from the value from the conceptual multiplexer. The VOUT_TRIM register will typically be used by the end user to adjust the output voltage once the PMBus device is assembled into the end user's system. This might be done, for example, to adjust the voltage at the pins of a critical IC to optimize its performance.

Next, the value from the VOUT_CAL_OFFSET register is added. This is also a two's complement number and can add to or subtract from the voltage command value. The VOUT_CAL_OFFSET register will typically be used by the PMBus device manufacturer to adjust the output voltage in their factory.

Next, if the PMBus device has an output voltage droop characteristic, it is applied. The VOUT_DROOP coefficients are always greater than or equal to zero. The value of the VOUT_DROOP coefficient and the value of output current are multiplied and the result is always subtracted from the voltage command. This means that the output voltage decreases with increasing output current and increases with decreasing with output current. The droop calculation applies even if the device is sinking current (negative output current).

Note that if the margin testing is being performed, a non-zero value of VOUT_DROOP will change the margin voltages proportional to the output current.

The next step is to compare the commanded voltage developed so far with the output voltage limits set by the VOUT_MAX and VOUT_MIN commands. If the calculated voltage command would create an output voltage greater than the VOUT_MAX value or less than the VOUT_MIN value, the PMBus device limits the command voltage passed to the controller to the VOUT_MAX or VOUT_MIN value, as appropriate. It also sets an alarm as described in Section 13.5.

The next step is to apply the same scaling factor to the calculated voltage command as is applied to the external output voltage by a resistive divider. This is done by multiplying the calculated voltage command by VOUT_SCALE_LOOP.

At this point, the device now has a calculated value that is used as the equivalent to the reference voltage in standard analog controller. This is the value to which the sensed output voltage is compared when making decisions about adjusting the device's duty cycle.

9.3. Switching Between PMBus and AVSBus Control Of The Output Voltage

Figure 13 shows in concept that the output voltage can be controlled by commands from the PMBus (VOUT_COMMAND, VOUT_MARGIN_HIGH, and VOUT_MARGIN_LOW) or the AVSBus (AVSBus Target Rail Voltage). In principle, control of the output voltage can be switched between PMBus and the AVSBus. However, there are constraints on how this change of control is made.

When switching from PMBus control to AVSBus control, the requirement is that the output voltage does not change. Conceptually this means that the VOUT_COMMAND value must always be loaded into the AVSBus Target Rail Voltage before the multiplexor shifts its output to be equal to the AVSBus Target Rail Voltage.

When switching from AVSBus to PMBus control, the user can choose whether the VOUT_COMMAND value is updated with the AVSBus Target Rail Voltage or not. This option is selected, or not, by setting or clearing bit [1] in the OPERATION command data (see Section 12.1.5).

The following changes of control of the output voltage are permitted while the PMBus device is operating:

- From VOUT_COMMAND to AVSBus Target Rail Voltage
- From AVSBus Target Rail Voltage to VOUT_COMMAND
- From VOUT_COMMAND to VOUT_MARGIN_HIGH
- From VOUT_MARGIN_HIGH to VOUT_COMMAND
- From VOUT_COMMAND to VOUT_MARGIN_LOW
- From VOUT_MARGIN_LOW to VOUT_COMMAND
- From VOUT_MARGIN_HIGH to VOUT_MARGIN_LOW
- From VOUT_MARGIN_LOW to VOUT_MARGIN_HIGH

The following changes of control of the output voltage are prohibited while the PMBus device is operating:

- From AVSBus Target Rail Voltage to VOUT_MARGIN_HIGH
- From VOUT_MARGIN_HIGH to AVSBus Target Rail Voltage
- From AVSBus Target Rail Voltage to VOUT_MARGIN_LOW
- From VOUT_MARGIN_LOW to AVSBus Target Rail Voltage

Figure 14 shows the permitted changes to the source of the nominal output voltage. Changes to or from the AVSBus Target Rail Voltage source of the output voltage value and either the margin states are specifically not permitted.

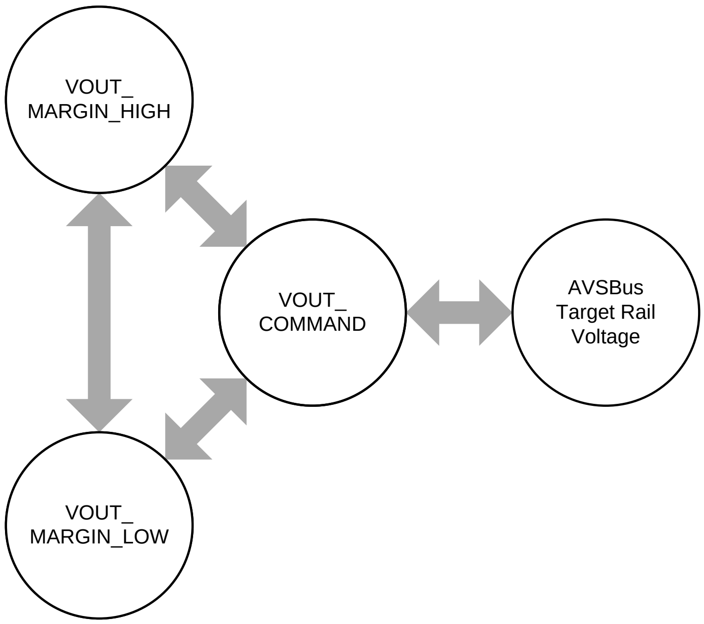


Figure 14. Permitted Changes Of Output Voltage Control

9.4. Making And Calibrating Output Current Measurements

The READ_IOUT command (Section 18.5) can be used to measure the PMBus device's output current.

Two commands are provided to improve the accuracy of output current measurements through single or two point calibrations: IOUT_CAL_GAIN (for gain calibration, Section 14.8) and IOUT_CAL_OFFSET (for offset calibration, Section 14.9).

These two commands are used to prepare the value returned in response to a READ_IOUT command as shown in the equation:

$$READ\_IOUT = \left(\frac{V_{MEASURED}(I_{OUT})}{IOUT\_CAL\_GAIN} \right) + IOUT\_CAL\_OFFSET$$

where $V_{MEASURED}$ (I_{OUT}) is a voltage proportional to the current being sensed.

The key point is that gain adjustment is applied first, followed by the offset adjustment. This sequence is illustrated in concept in Figure 15.

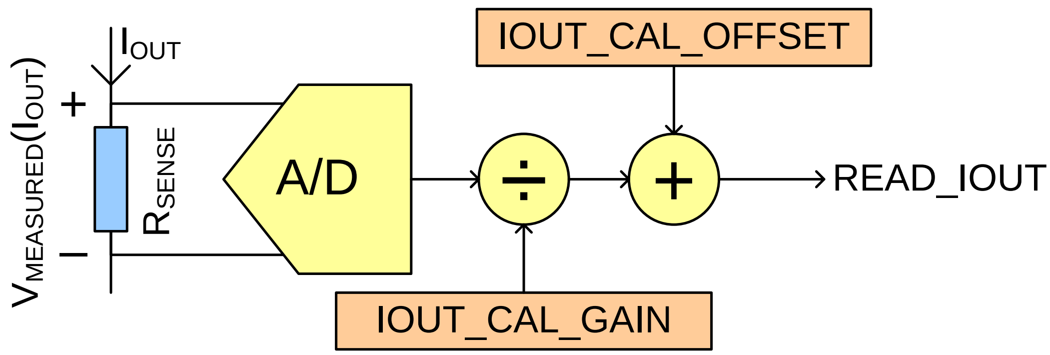


Figure 15. Generating READ_IOUT Concept

To minimize the error in values returned by READ_IOUT, automatic test equipment can be used to make measurements, calculate the best values of IOUT_CAL_GAIN and IOUT_CAL_OFFSET, and then load those values into the device.

For example, automatic test equipment could load a device to a precisely known output current. It would then use the READ_IOUT command to determine what current the device is reporting. A second measurement at a different load current would also typically be taken. Using the known currents drawn by the test equipment and the two currents reported by the device, the test equipment can then calculate the best values of IOUT_CAL_GAIN and IOUT_CAL_OFFSET to minimize the error in the current sensing circuit.

9.5. Deleted

10. Fault Management And Reporting

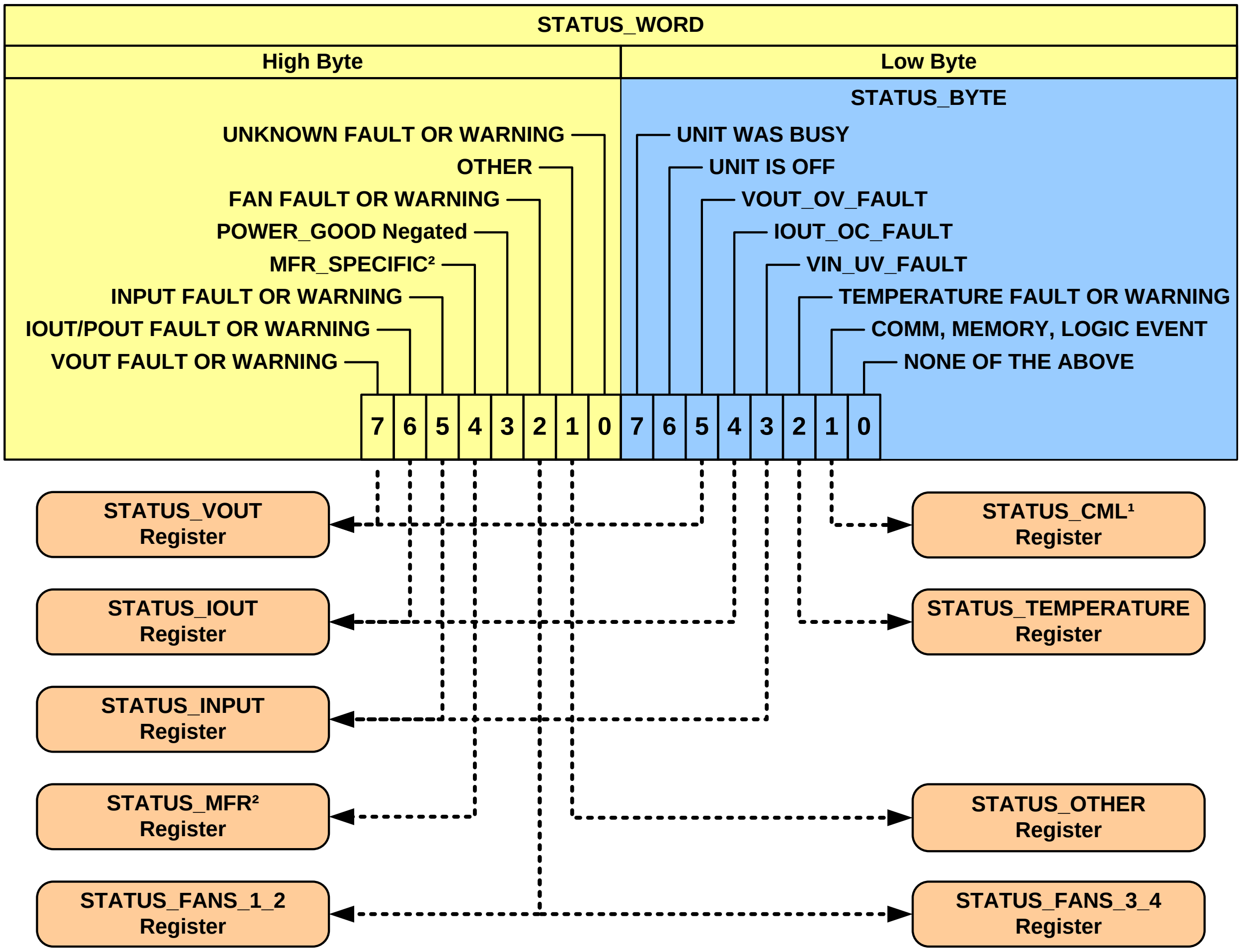
The PMBus protocol provides a comprehensive set of tools for monitoring the operation of and managing the faults in a PMBus device. Provisions are made for a host or power system manager to read a wide range of parametric values, such as the output voltage or output current. The PMBus protocol also includes the ability to program fault or warning levels for every important aspect of a power conversion device.

10.1. Monitoring Operation

The host or power system manager can use READ commands to ask a PMBus device about its current state. To simplify the PMBus devices, there is one READ command for each parameter, such as output voltage or device temperature. The details of the READ commands are given in Section 17.10.

PMBus devices can provide status information in two forms. One form (parametric information) is returned as a value, such as output voltage or output current. The details of these commands are given in Section 18.

The other form (binary OK/Not OK) is in the form of status bits and registers. The PMBus protocol provides three levels of status registers. This allows host or power system managers to retrieve the most important information in a fast, one byte transaction. Based on this information the host can act or request more detailed information. Figure 16 shows the relationship between the STATUS_BYTE register, the STATUS_WORD register and the more detailed status registers.



[1]: CML: Communication, Memory, Logic
[2]: MFR SPECIFIC: Manufacturer Specific

Figure 16. Status Register Map

As shown in Figure 16, the STATUS_BYTE register contains the most important fault and warnings. This allows the most basic PMBus devices to provide the most critical information at the lowest cost. The STATUS_WORD includes the STATUS_BYTE as its lower byte. In the higher byte of the STATUS_WORD, there are additional bits providing more information about the status of the PMBus device.

In more advanced PMBus devices, there are seven registers with even more detailed information about the status of the unit. The host or power system manager knows which of these to read based on which bits are set in the STATUS_BYTE or STATUS_WORD.

The details of the STATUS_BYTE, STATUS_WORD and other status registers are given in Section 17.

10.2. General Description Of PMBus Device Fault Management

The PMBus protocol supports setting warning (minor alarm) and fault (major alarm) thresholds for nearly every possible event.

If the PMBus device detects that one of these thresholds has been exceeded, a bit corresponding to the condition is latched.

10.2.1. Warning Conditions

Warning conditions are an indication that the device has a problem but can continue operating.

When the PMBus device detects a warning condition, the device sets the corresponding bit(s) in the status registers. When a bit is set, it remains set until cleared as described in Section 10.2.3.

Depending on what the PMBus device supports, it will:

- Simply set the warning condition bit(s) and wait for the host or power system manager to poll it, or
- The PMBus device may notify the host that a warning condition has occurred (Section 10.6).

10.2.2. Fault Conditions

Fault conditions are more serious than a warning condition. Depending on the severity of the fault condition and whether there is risk of damage to the load or the device, a fault may cause the PMBus device to disable the output and stop the transfer of energy to the output.

For many fault conditions (Section 15), the PMBus device can be programmed with a wide range of responses such as shut down immediately and latch off, shut down and retry or continue to operate for a specified delay time before shutting down. The possible fault responses are described in Section 10.5.

In addition, the PMBus device will set the corresponding fault bit(s) in the status registers. This bit remains (or bits remain) set until cleared as described in Section 10.2.3.

Depending on what the PMBus device supports, it will:

- Simply set the fault condition bit(s) and wait for the host or power system manager to poll it or
- The PMBus device may notify the host that a fault condition has occurred (Section 10.6).

10.2.3. Clearing Warning Or Fault Bits

Almost all of the warning or fault bits set in the status registers remain set, even if the fault or warning condition is removed or corrected, until one of the following occur:

- The bit is individually cleared (Section 10.2.4),
- The device receives a CLEAR_FAULTS command (Section 15.1),
- A RESET signal (if one exists) is asserted,
- The output is commanded through the CONTROL pin, the OPERATION command, or the combined action of the CONTROL pin and OPERATION command, to turn off and then to turn back on, or
- Bias power is removed from the PMBus device. Removing the bias power usually means that the input power has been removed long enough that the voltage to the control circuit has decayed to zero. However, in some devices, the input power and the power to the control circuitry are separate. In this case, removing the bias power means removing the input power to the control circuitry.

The two exceptions to the rule that status bits remain set are the OFF and PG_STATUS# bits. These bits always reflect the current state of the device and the POWER_GOOD signal (if present).

10.2.4. Clearing Individual Bits

Any or all of the bits in any status register except STATUS_BYTE and STATUS_WORD can be directly cleared by issuing the status command with one data byte that is written. The data byte is a binary value. A 1 in any bit position indicates that bit is to be cleared, if set, and unchanged if not set. Examples of data bytes:

- 00010000b indicates that bit [4] is to be cleared and all other bits are to be unchanged,
- 01100010b indicates that bits [6], [5], and [1] are to be cleared and all other bits are to be unchanged.
- 11111111b, or FFh, indicates all bits are to be cleared.

10.2.5. Clearing Bits In The STATUS_BYTE And STATUS_WORD

10.2.5.1. General Rules

Most bits in the STATUS_BYTE and STATUS_WORD are cleared by clearing the bit or all of the bits that cause the bit in STATUS_BYTE or STATUS_WORD to be set.

In general, one can think of the bits in STATUS_BYTE and STATUS_WORD as a logical OR of the bits in a lower level status register. Figure 17 shows this concept.

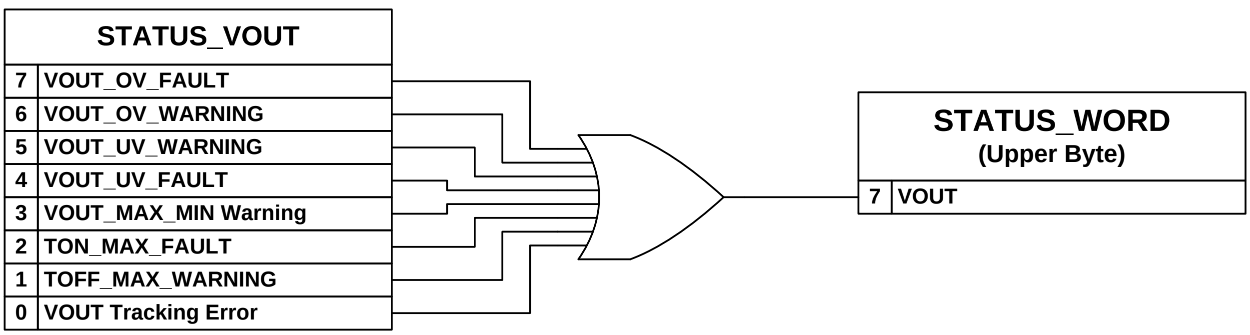


Figure 17. Conceptual View Of Creating Bits In STATUS_BYTE And STATUS_WORD

For example, if the VOUT_OV_FAULT bit in the STATUS_VOUT register is set, then the VOUT bit in the STATUS_WORD is also set. When the VOUT_OV_FAULT bit in the STATUS_VOUT register is cleared, the VOUT bit in the STATUS_WORD will be cleared at the same time provided no other bit in STATUS_VOUT is set.

For another example, suppose both VOUT_OV_WARNING and VOUT_OV_FAULT bits are set in the STATUS_VOUT register. Clearing just VOUT_OV_WARNING (or VOUT_OV_FAULT) and leaving VOUT_OV_FAULT (or VOUT_OV_WARNING) set will not cause the VOUT bit in the STATUS_VOUT register to clear. Only when no bits are set in STATUS_VOUT will the VOUT bit in STATUS_WORD be cleared.

10.2.5.2. Exceptions

There are two bits in STATUS_BYTE and STATUS_WORD that can be cleared directly. The BUSY bit in STATUS_BYTE is cleared by sending the STATUS_BYTE command with the data byte 80h using the WRITE BYTE protocol. The UNKNOWN bit in STATUS_WORD is cleared by sending the STATUS_WORD command with the data bytes 00h (low order byte) followed by 01h (high order byte) using the WRITE WORD protocol.

As noted above, the OFF and PG_STATUS# bits cannot be cleared as they always reflect the current state of the device.

10.2.6. Immediate Reassertion After Clearing If Condition Is Still Present

If the warning or fault condition is present when the bit is cleared, the bit is immediately set again. The device shall respond as described in Section 10.2.1 or Section 10.2.2 as appropriate.

Note that one effect is that if the SMBALERT# signal had been cleared before the status bit was cleared, the SMBALERT# will also be asserted again immediately after the status bit is cleared. The SMBALERT_MASK command can be used to prevent this behavior.

10.3. Conceptual View Of How Status Bits And SMBALERT# Work

When some warning or fault event is detected a latch is set. The output of this latch becomes the status bit in one of the lower level status register (such as STATUS_VOUT). The latch output may also be used, either by itself or OR'ed with other status bits, to create the corresponding bit in STATUS_BYTE or STATUS_WORD and to affect SMBALERT#.

The output of the latch passes through a gate controlled by the corresponding SMBALERT_MASK bit. If this bit is set, the output of the latch is blocked from driving the SMBALERT# circuit. If the SMBALERT_MASK bit is cleared, the latch output is allowed to pass and drive the SMBALERT# circuit. Figure 18 gives a conceptual illustration of how the SMBALERT# signal is generated.

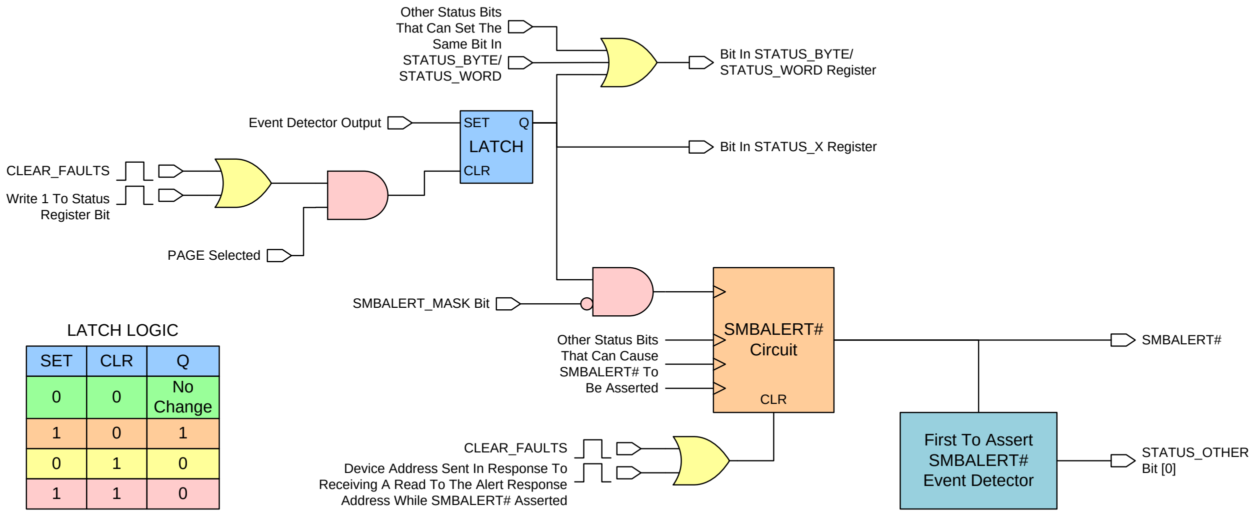
When the SMBALERT# circuit detects the rising edge of the latch output it asserts the SMBALERT# signal (output goes low).

If SMBALERT# was high (not asserted by another device on the bus), prior to the device asserting SMBALERT#, then it also sets bit [0] of the STATUS_OTHER register. Understanding the timing is not completely deterministic, in the case when many devices are asserting SMBALERT#, this provides a way for the host to know which device was probably the first to have a problem.

The SMBALERT# signal remains asserted until is cleared. It is cleared when the device successfully transmits its address in response to receiving the Alert Response Address. It is also cleared by a CLEAR_FAULTS command.

Note the behavior of the latch output and the SMBALERT_MASK. If the latch output is set, and the SMBALERT_MASK bit is changed from set to clear, the latch output is passed to the SMBALERT# circuit and will cause the SMBALERT# signal to assert. If this is not desired, the latch must be cleared before clearing the SMBALERT_MASK bit.

As described above, the latch can be cleared by writing a 1 to corresponding bit in the status register. It can also be cleared with the CLEAR_FAULTS command.



SET	CLR	Q
0	0	No Change
1	0	1
0	1	0
1	1	0

Figure 18. Conceptual Schematic Of Status Bits And SMBALERT#

Commands to clear a bit are gated by the PAGE command. The CLEAR_FAULTS can be made to clear all faults on all pages by setting the page command to FFh.

Conceptually the bit clearing commands act as pulses, driving the reset pin on the latch only momentarily.

This means that if the event is ongoing (the event detector is still active) the output latch will immediately set again. As described above, this will cause the SMBALERT# to reassert if it had been previously cleared (and the SMBALERT_MASK bit is not set).

This also means that a host won't be able to see the status bit get cleared. If a host sends the command to clear a status bit and then reads the bit as set, it should interpret that to mean that the event is still happening (assuming the PAGE is set properly).

10.4. Setting Fault And Warning Thresholds

Section 15 includes a comprehensive list of commands to set fault and warning thresholds that PMBus devices may support.

Not all PMBus devices will support all of the fault detection, reporting and management functions and features. The PMBus device product literature shall indicate which features and functions it supports.

10.5. Setting The Response To A Detected Fault Condition

Commands are provided to set the response to each fault condition. These commands have one data byte that describes how the device should respond to the fault. Each of the fault response commands requires that the user set three parameters that determine how the device will respond to the fault condition.

The first parameter is called the Response. The choice to be made is whether or not the device is to continue operating, shutdown or disable the output while the fault condition is present (Inhibit).

The second parameter is the Retry Setting. The choice to be made for the Retry Setting is whether or not to attempt to restart operation if the device shuts down in response to a fault.

The third parameter is Delay Time. The choice to be made here depends on the choices for the Response and Retry Settings. The device user must choose either:

- The period of time the unit continues to operate without shutting down after a fault is detected, or
- The time between retry attempts.

The details are given in the following sections.

10.5.1. Response To Voltage, Temperature And TON_MAX Faults

The data byte specifying the response to a voltage or temperature fault is detailed in Table 4.

Table 4. Voltage, Temperature And TON_MAX Faults Response Data Byte Details

<table>
<tr><th>Bits</th><th>Description</th><th>Value</th><th>Meaning</th></tr>
<tr><td rowspan="4">7:6</td><td rowspan="4">Response

For all values of bits [7:6], the device:
• Sets the corresponding fault bit in the status registers and
• If the device supports notifying the host (Section 10.6), it does so.

The fault bit, once set, is cleared only in accordance with Section 10.2.3 and not when the fault condition is removed or is corrected.</td><td>00</td><td>The PMBus device continues operation without interruption.</td></tr>
<tr><td>01</td><td>The PMBus device continues operation for the delay time specified by bits [2:0] and the delay time unit specified for that particular fault. If the fault condition is still present at the end of the delay time, the unit responds as programmed in the Retry Setting (bits [5:3]).</td></tr>
<tr><td>10</td><td>The device shuts down (disables the output) and responds according to the retry setting in bits [5:3].</td></tr>
<tr><td>11</td><td>The device’s output is disabled while the fault is present. Operation resumes and the output is enabled when the fault condition no longer exists.</td></tr>
<tr><td>5:3</td><td>Retry Setting</td><td>000</td><td>A zero value for the Retry Setting means that the unit does not attempt to restart. The output remains disabled until the fault is cleared (Section 10.7).</td></tr>
</table>

Bits	Description	Value	Meaning
		001-110	The PMBus device attempts to restart the number of times set by these bits. The minimum number is 1 and the maximum number is 6. If the device fails to restart (the fault condition is no longer present and the device is delivering power to the output and operating as programmed) in the allowed number of retries, it disables the output and remains off until the fault is cleared as described in Section 10.7. The time between the start of each attempt to restart is set by the value in bits [2:0] along with the delay time unit specified for that particular fault.
		111	The PMBus device attempts to restart continuously, without limitation, until it is commanded OFF (by the CONTROL pin or OPERATION command or both), bias power is removed, or another fault condition causes the unit to shut down.
2:0	Delay Time	XXX	The number of delay time units, which vary depending on the type of fault. This delay time is used for either the amount of time a unit is to continue operating after a fault is detected or for the amount of time between attempts to restart.

10.5.2. Response To Current Faults

The data byte specifying the response to a current fault is detailed in Table 5.

Table 5. Current Fault Response Data Byte Details

Bits	Description	Value	Meaning
7:6	Response For all values of bits [7:6], the device: • Sets the corresponding fault bit in the status registers and	00	The PMBus device continues to operate indefinitely while maintaining the output current at the value set by IOUT_OC_FAULT_LIMIT (Section 15.8) without regard to the output voltage (known as constant-current or brickwall limiting).

Bits	Description	Value	Meaning
	• If the device supports notifying the host (Section 10.6), it does so. The fault bit, once set, is cleared only in accordance with Section 10.2.3 and not when the fault condition is removed or is corrected.	01	The PMBus device continues to operate indefinitely while maintaining the output current at the value set by IOUT_OC_FAULT_LIMIT (Section 15.8) as long as the output voltage remains above the minimum value specified by IOUT_OC_LV_FAULT_LIMIT (Section 15.10). If the output voltage is pulled down to less than that value, then the PMBus device shuts down and responds according to the Retry setting in bits [5:3].
		10	The PMBus device continues to operate, maintaining the output current at the value set by IOUT_OC_FAULT_LIMIT (Section 15.8) without regard to the output voltage, for the delay time set by bits [2:0] and the delay time units for specified in the IOUT_OC_FAULT_RESPONSE (Section 15.9). If the device is still operating in current limiting at the end of the delay time, the device responds as programmed by the Retry Setting in bits [5:3].
		11	The PMBus device shuts down and responds as programmed by the Retry Setting in bits [5:3].
5:3	Retry Setting	000	A zero value for the Retry Setting means that the unit does not attempt to restart. The output remains disabled until the fault is cleared (Section 10.7).

Bits	Description	Value	Meaning
		001-110	The PMBus device attempts to restart the number of times set by these bits. The minimum number is 1 and the maximum number is 6. If the device fails to restart (the fault condition is no longer present and the device is delivering power to the output and operating as programmed) in the allowed number of retries, it disables the output and remains off until the fault is cleared as described in Section 10.7. The time between the start of each attempt to restart is set by the value in bits [2:0] along with the delay time unit specified for that particular fault.
		111	The PMBus device attempts to restart continuously, without limitation, until it is commanded OFF (by the CONTROL pin or OPERATION command or both), bias power is removed, or another fault condition causes the unit to shut down.
2:0	Delay Time	XXX	The number of delay time units, which vary depending on the type of fault. This delay time is used for either the amount of time a unit is to continue operating after a fault is detected or for the amount of time between attempts to restart.

10.6. Reporting Faults And Warnings To The Host

PMBus devices may support notifying the host if a fault or warning is detected.

There are two means available for a PMBus device to notify the host of a warning or fault condition: the SMBALERT# signal and direct communication from the PMBus device to the host. PMBus devices shall support at most one of the two methods.

10.6.1. SMBALERT# Signal And Process

The SMBALERT# process is described in the SMBus specification [A03].

Figure 19 shows how the status bits work in concert with the SMBALERT# signal. The basic principle is that if the host has already been notified by a device that it has a fault or warning condition, but the host has not yet read the status of the device, then there is no need for another SMBALERT# signal to the host.

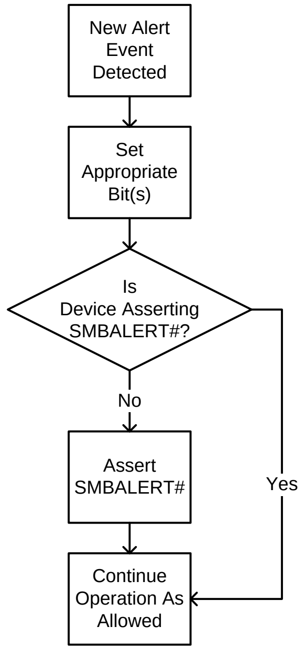


Figure 19. Interaction Of SMBALERT# And Status Registers

10.6.2. Direct PMBus Device To Host Communication

PMBus devices may temporarily become bus masters, as permitted in the SMBus specification [A03], in order to send notice to the host that a fault or an error has occurred. The format of the packet is shown in Figure 20.

The data bytes are the same as the STATUS_WORD command (Section 17.1).

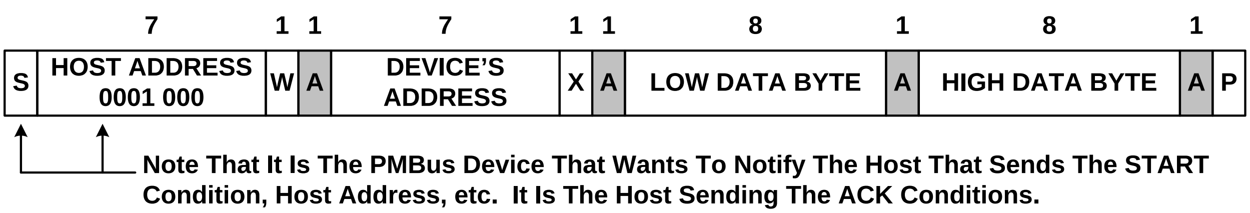


Figure 20. Packet Structure For PMBus Device To Notify Host

10.7. Clearing A Shutdown Due To A Fault

Any device that has shut down due to a fault condition remains off until:

- A RESET signal (if one exists) is asserted,
- The output is commanded through the CONTROL pin, the OPERATION command, or the combined action of the CONTROL pin and OPERATION command, to turn off and then to turn back on, or
- Bias power is removed from the PMBus device.

10.8. Data Transmission Faults

A data transmission fault occurs when information is not properly transferred between two devices. There are several possible data communication faults. This section describes these faults and how a PMBus device is to respond to each of them.

10.8.1. Corrupted Data

If the value of one or more bits in a packet is changed in transmission, this can be detected using the Packet Error Checking mechanism (SMBus specification [A03]).

If a PMBus device detects that the Packet Error Code (PEC) calculated by the receiving device does not match the received PEC, the preferred response is to NACK the PEC byte. Some PMBus devices may not be able to calculate the PEC and respond in time to NACK the last byte. This is acceptable behavior.

Whenever a PMBus device detects that the received and calculated PEC bytes do not match, whether or not the device was able to NACK the PEC byte, the device shall respond as follows:

- Not respond to or act upon the received command,
- Flush or ignore the received command code and any received data,
- Set the CML bit in the STATUS_BYTE,
- Set the Packet Error Check Failed bit in the STATUS_CML register (if supported), and
- Notify the host as described in Section 10.2.2.

10.8.2. Sending Too Few Bits

PMBus (and SMBus) transactions are carried out one byte at time. If a START or STOP condition interrupts the transmission while a device is writing to a PMBus device before a complete byte has been sent, this is a data transmission fault.

When a PMBus device detects this fault, it shall respond as follows:

- Flush or ignore the received command code and any received data,
- Set the CML bit in the STATUS_BYTE,
- Set bit [1] (“Other” fault) bit in the STATUS_CML register (if supported), and
- Notify the host as described in Section 10.2.2.

10.8.3. Reading Too Few Bits

PMBus (and SMBus) transactions are carried out one byte at time. If a START or STOP condition interrupts the transmission while a device is reading from a PMBus device before a complete byte has been read, this is a data transmission fault.

When a PMBus device detects this fault, it shall respond as follows:

- Flush or ignore the received command code and any received data,
- Set the CML bit in the STATUS_BYTE,
- Set bit [1] (“Other” fault) bit in the STATUS_CML register (if supported), and
- Notify the host as described in Section 10.2.2.

10.8.4. Host Sends Or Reads Too Few Bytes

If the host ends a PMBus packet with a STOP Condition before the host has transmitted all the bytes the PMBus device expected to receive, it is preferred not to treat this as an error. It is presumed that the host knows what it is doing and that it intentionally ended the transaction.

Similarly, if the host ends a PMBus packet with a STOP Condition before the host has read all the bytes the PMBus device expected to send, it is preferred not to treat this as an error. It is presumed that the host knows what it is doing and that it intentionally ended the transaction.

To support legacy devices, it is permissible for the PMBus device to treat this as a data communications fault and respond as follows:

- Set the CML bit in the STATUS_BYTE,
- Set bit [1] ("Other" fault) bit in the STATUS_CML register (if supported), and
- Notify the host as described in Section 10.2.2.

Note that declaring a communications fault for a transaction that the host terminates early will not be allowed in future versions of the PMBus specification.

10.8.5. Host Sends Too Many Bytes

If while writing to a PMBus device, the host sends more bytes than the device is expecting, this is a data transmission fault.

Sending a PEC byte to a device that does not support Packet Error Checking is included in this fault.

When a PMBus device detects this fault, it shall respond as follows:

- If possible, NACK all of the unexpected bytes as they are received (until the next STOP condition is received),
- Flush or ignore the received command code and any received data,
- Set the CML bit in the STATUS_BYTE,
- Set the Invalid Or Unsupported Data Received bit in the STATUS_CML register (if supported), and
- Notify the host as described in Section 10.2.2.

10.8.6. Reading Too Many Bytes

If while reading from a PMBus device, the host tries to read more bytes than the device is expecting to send, this is a data transmission fault.

Trying to read a PEC byte from a device that does not support Packet Error Checking is included in this fault.

When a PMBus device detects this fault, it shall respond as follows:

- Send all ones (FFh) as long as the host keeps clocking and acknowledging,
- Set the CML bit in the STATUS_BYTE,
- Set bit [1] ("Other" fault) bit in the STATUS_CML register (if supported), and
- Notify the host as described in Section 10.2.2.

10.8.7. Device Busy

A data transmission fault can occur if the receiving device is too busy to respond to communication on the bus. If a PMBus device is too busy to accept and process a command being sent to it over the bus, it shall respond as follows:

- ACK the address byte as all SMBus devices must ACK their own address,
- If possible, NACK the command byte and data bytes as they are received,
- If the host is attempting to read from the device, send all ones (FFh) as long as the host keeps clocking and acknowledging,

- Set the BUSY bit in the STATUS_BYTE, and
- Notify the host as described in Section 10.2.2.

10.9. Data Content Faults

If data is transferred without corruption from the host to a PMBus device, but the PMBus device is not able to process the received data, this is a data content fault. There are several possible Data Content Faults. This section describes these faults and how a PMBus device is to respond to each of them.

10.9.1. Improperly Set Read Bit In The Address Byte

Commands sent to individual PMBus devices all start with writing a command code. No command sent to an individual PMBus device starts with the R/W# bit set for read (value equal to 1). Starting a transaction with a PMBus device with the R/W# bit set to 1 is a Data Content Fault.

Note that there is one case when the R/W# bit should be set to 1. This is when the address is the SMBus Alert Response Address (0001 100b).

When a PMBus device receives a packet at its own address with the R/W# bit set to 1, it shall responds as follows:

- ACK the address byte as all SMBus devices must ACK their own address,
- If possible, NACK the command byte and data bytes as they are received,
- Send all ones (FFh) as long as the host keeps clocking and acknowledging,
- Set the CML bit in the STATUS_BYTE,
- Set bit [1] ("Other" fault) bit in the STATUS_CML register (if supported), and
- Notify the host as described in Section 10.2.2.

10.9.2. Unsupported Command Code

If a PMBus device receives a command that it does not support, including those command codes identified as Reserved, the device shall respond as follows:

- If possible, NACK the unsupported command code and all data bytes received before the next STOP condition,
- Flush or ignore the received command code and any received data,
- Set the CML bit in the STATUS_BYTE register,
- Set the Invalid Or Unsupported Command Received bit in the STATUS_CML register (if that register is supported), and
- Notify the host as described in Section 10.2.2.

10.9.3. Invalid Or Unsupported Data

There are two kinds of invalid or unsupported data. The first is a data that is totally unsupported by a device. An example of this is sending a VOUT_MODE command that attempts to set the output voltage mode to VID when the device only supports Direct Mode data for output voltage related commands.

The second kind of invalid or unsupported data fault can occur when there are multiple options for a given command code. For example, there are several possible responses to an output overvoltage fault. If a device only supports shut down and latch off, trying so set the fault response to "Inhibit Operation Only While the Fault Is Present" is treated as invalid or unsupported data. Another example of this kind of invalid or unsupported data is trying to command a device to execute a margin test when that device does not support the margin test options of the OPERATION command. Yet another example of this kind of invalid or unsupported data is attempting to send a value that is not defined for the command (example: sending the value FFH as the data byte for an OPERATION command).

If a PMBus device receives unsupported data, the preferred response is that the device shall:

- If possible, NACK the unsupported data bytes received before the next STOP condition,
- Flush or ignore the received command code and any received data,
- Set the CML bit in the STATUS_BYTE,
- Set the Invalid Or Unsupported Data Received bit in the STATUS_CML register (if supported), and
- Notify the host as described in Section 10.2.2.

In order to accommodate legacy devices, an acceptable response to unsupported data of the second kind is to convert the data to the nearest valid value (as defined by the device manufacturer). The command is then executed and no fault is declared. Note that this behavior will not be permitted in future revisions of the PMBus specification.

10.9.4. Data Out Of Range Fault

An example of a Data Out Of Range fault is an attempt to set the output of a typical board mounted point-of-load converter to 1000 V.

It is optional for a PMBus device to detect an attempt to set a parameter to a value that the device cannot realize. How a device knows that a value is out of range is left to the discretion of the device manufacturer.

If a device does support detecting data that is out of the range of the device, it shall respond as follows:

- If possible, NACK the unsupported data bytes received before the next STOP condition,
- Flush or ignore the received command code and any received data,
- Set the CML bit in the STATUS_BYTE,
- Set the Invalid Or Unsupported Data Received bit in the STATUS_CML register (if supported), and
- Notify the host as described in Section 10.2.2.

10.9.5. Reserved Bits

In several of the command definitions, bits are identified as reserved. PMBus devices shall ignore these bits, even if set. It is not a fault if a bit described as reserved is received as set (value equal to one).

11. Address, Memory, Communication And Capability Related Commands

11.1. WRITE_PROTECT

The WRITE_PROTECT command is used to control writing to the PMBus device. The intent of this command is to provide protection against accidental changes. This command is not intended to provide protection against deliberate or malicious changes to a device's configuration or operation.

All supported commands may have their parameters read, regardless of the WRITE_PROTECT settings.

This command has one data byte, described in Table 6.

Table 6. WRITE_PROTECT Command Data Byte

Data Byte Value	Meaning
1000 0000	Disable all writes except to the WRITE_PROTECT command
0100 0000	Disable all writes except to the WRITE_PROTECT, OPERATION and PAGE commands
0010 0000	Disable all writes except to the WRITE_PROTECT, OPERATION, PAGE, ON_OFF_CONFIG and VOUT_COMMAND commands
0000 0011	Manufacturer specified
0000 0010	Manufacturer specified
0000 0001	Manufacturer specified
0000 0000	Enable writes to all commands.

If a device receives a data byte that is not listed in Table 6, then the device shall treat this as invalid data, declare a communications fault and respond as described in Section 10.8.

11.2. STORE_DEFAULT_ALL

The STORE_DEFAULT_ALL command instructs the PMBus device to copy the entire contents of the Operating Memory to the matching locations in the non-volatile Default Store memory. Any items in Operating Memory that do not have matching locations in the Default Store are ignored.

It is permitted to use the STORE_DEFAULT_ALL command while the device is operating. However, the device may be unresponsive during the copy operation with unpredictable, undesirable or even catastrophic results. PMBus device users are urged to contact the PMBus device manufacturer about the consequences of using the STORE_DEFAULT command while the device is operating and providing output power.

This command has no data bytes.

This command is write only.

11.3. RESTORE_DEFAULT_ALL

The RESTORE_DEFAULT_ALL command instructs the PMBus device to copy the entire contents of the non-volatile Default Store memory to the matching locations in the Operating Memory. The values in the Operating Memory are overwritten by the value retrieved from the Default Store. Any items in Default Store that do not have matching locations in the Operating Memory are ignored.

It is permitted to use the RESTORE_DEFAULT_ALL command while the device is operating. However, the device may be unresponsive during the copy operation with unpredictable, undesirable or even catastrophic results. PMBus device users are urged to contact the PMBus device manufacturer about the consequences of using the RESTORE_DEFAULT_ALL command while the device is operating and providing output power.

This command has no data bytes.

This command is write only.

11.4. STORE_DEFAULT_CODE

The STORE_DEFAULT_CODE command instructs the PMBus device to copy the parameter whose Command Code matches the value in the data byte, from the Operating Memory to the matching location in the non-volatile Default Store memory.

If the device does not permit saving this parameter in the Default Store, or if the device does not support the Command Code specified in the data byte, then the device must notify the host that the command failed, as described in the PMBus specification, Part I [A01].

It is permitted to use the STORE_DEFAULT_CODE command while the device is operating. However, the device may be unresponsive during the copy operation with unpredictable, undesirable or even catastrophic results. PMBus device users are urged to contact the PMBus device manufacturer about the consequences of using the STORE_DEFAULT_CODE command while the device is operating and providing output power.

This command has one data byte, formatted as an unsigned binary integer.

This command is write only.

11.5. RESTORE_DEFAULT_CODE

The RESTORE_DEFAULT_CODE command instructs the device to copy the parameter whose Command Code matches the value in the data byte from the non-volatile Default Store memory to the matching location in the Operating Memory. The value in the Operating Memory is overwritten by the value retrieved from the Default Store.

If the device does not save this parameter in the Default Store, or if the device does not support the Command Code specified in the data byte, then the device must notify the host that the command failed, as described in the PMBus specification, Part I [A01].

It is permitted to use the RESTORE_DEFAULT_CODE command while the device is operating. However, the device may be unresponsive during the copy operation with unpredictable, undesirable or even catastrophic results. PMBus device users are urged to contact the PMBus device manufacturer about the consequences of using the RESTORE_DEFAULT_CODE command while the device is operating and providing output power.

This command has one data byte, formatted as an unsigned binary integer.

This command is write only.

11.6. STORE_USER_ALL

The STORE_USER_ALL command instructs the PMBus device to copy the entire contents of the Operating Memory to the matching locations in the non-volatile User Store memory. Any items in Operating Memory that do not have matching locations in the User Store are ignored.

It is permitted to use the STORE_USER_ALL command while the device is operating. However, the device may be unresponsive during the copy operation with unpredictable, undesirable or even catastrophic results. PMBus device users are urged to contact the PMBus device manufacturer about the consequences of using the STORE_USER_ALL command while the device is operating and providing output power.

This command has no data bytes.

This command is write only.

11.7. RESTORE_USER_ALL

The RESTORE_USER_ALL command instructs the PMBus device to copy the entire contents of the non-volatile User Store memory to the matching locations in the Operating Memory. The values in the Operating Memory are overwritten by the value retrieved from the User Store. Any items in User Store that do not have matching locations in the Operating Memory are ignored.

It is permitted to use the RESTORE_USER_ALL command while the device is operating. However, the device may be unresponsive during the copy operation with unpredictable, undesirable or even catastrophic results. PMBus device users are urged to contact the PMBus device manufacturer about the consequences of using the RESTORE_USER_ALL command while the device is operating and providing output power.

This command has no data bytes.

This command is write only.

11.8. STORE_USER_CODE

The STORE_USER_CODE command instructs the PMBus device to copy the parameter whose Command Code matches value in the data byte from the Operating Memory to the matching location in the non-volatile User Store memory.

If the device does not permit saving this parameter in the User Store, or if the device does not support the Command Code specified in the data byte, then the device must notify the host that the command failed, as described in the PMBus specification, Part I [A01].

It is permitted to use the STORE_USER_CODE command while the device is operating. However, the device may be unresponsive during the copy operation with unpredictable, undesirable or even catastrophic results. PMBus device users are urged to contact the PMBus device manufacturer about the consequences of using the STORE_USER_CODE command while the device is operating and providing output power.

This command has one data byte, formatted as an unsigned binary integer.

This command is write only.

11.9. RESTORE_USER_CODE

The RESTORE_USER_CODE command instructs the PMBus device to copy the parameter whose Command Code matches the value in the data byte from the non-volatile User Store memory to the matching location in the Operating Memory. The value in the Operating Memory is overwritten by the value retrieved from the User Store.

If the device does not save this parameter in the User Store, or if the device does not support the Command Code specified in the data byte, then the device must notify the host that the command failed, as described in the PMBus specification, Part I [A01].

It is permitted to use the RESTORE_USER_CODE command while the device is operating. However, the device may be unresponsive during the copy operation with unpredictable, undesirable or even catastrophic results. PMBus device users are urged to contact the PMBus device manufacturer about the consequences of using the RESTORE_USER_CODE command while the device is operating and providing output power.

This command has one data byte, formatted as an unsigned binary integer.

This command is write only.

11.10. PAGE

The page command provides the ability to configure, control and monitor through only one physical address either:

- Multiple outputs on one unit, or
- Multiple non-PMBus devices through a PMBus device to non-PMBus device adapter or bridge.

Figure 21 and Figure 22 illustrate these concepts.

Each PAGE contains the Operating Memory (and at the option of the device manufacturer, User Store and Default Store) for each output. Each page may offer the full range of PMBus commands available for each output or non-PMBus device.

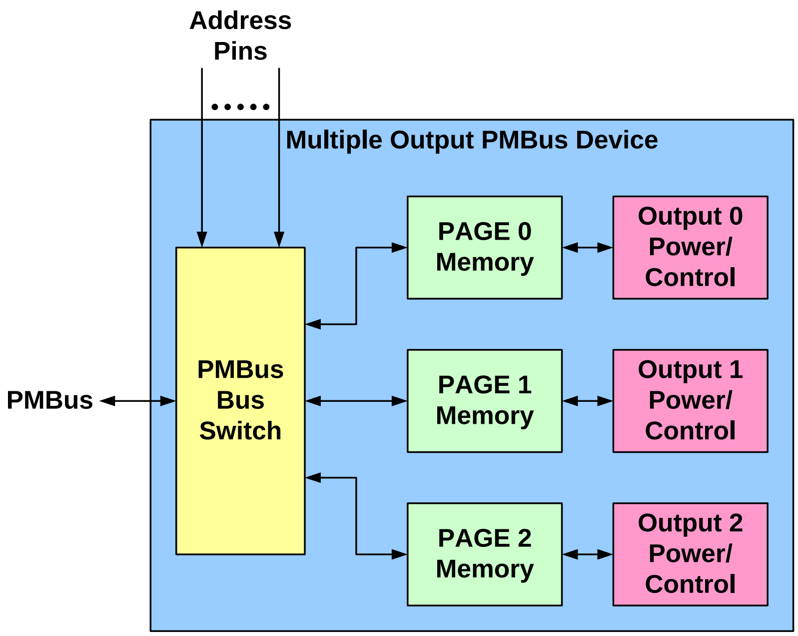


Figure 21. Conceptual View Of Paging Used For A Multiple Output PMBus Device

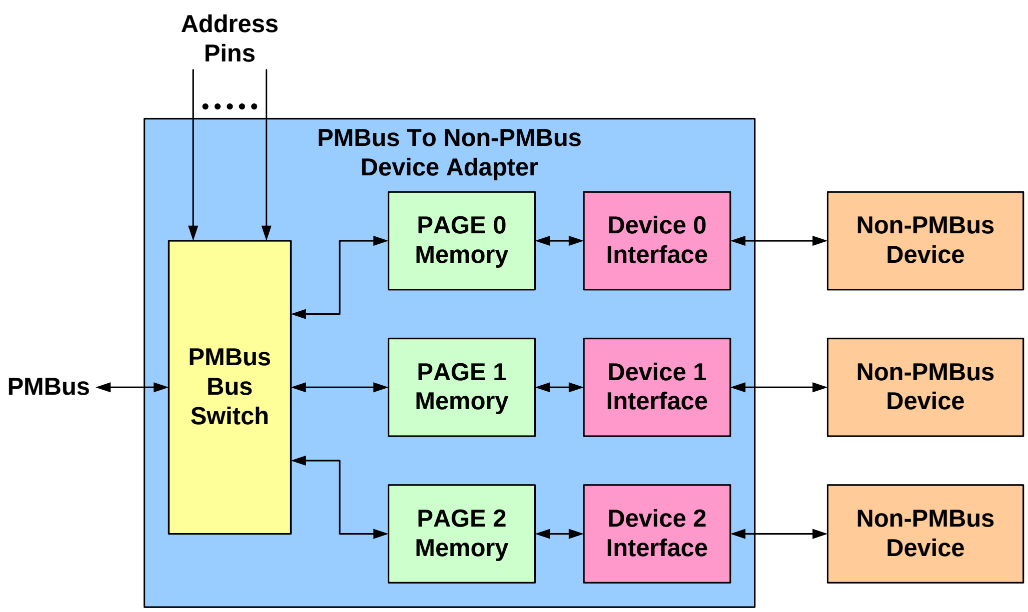


Figure 22. Conceptual View Of Using Paging With A PMBus To Non-PMBus Device Adapter

PMBus device manufacturers may also use multiple pages within a single PMBus device to offer additional commands or memory space for one or more outputs.

The data byte for the PAGE command is an unsigned binary integer.

Pages 00h through 1Fh are reserved specifically for multiple outputs on a device with a single physical address.

Setting the PAGE to FFh means that all subsequent commands are to be applied to all outputs.

Some commands, such as READ_TEMPERATURE, may use a common sensor but be available on all pages of a device. Such implementations are the decision of each device manufacturer or are specified in a PMBus Application Profile. Consult the manufacturer's documents or the Application Profile specification as needed.

11.11. PHASE

The PHASE command provides the ability to configure, control, and monitor multiple phases on one PMBus unit.

Each PHASE contains the Operating Memory (and at the option of the device manufacturer, User Store and Default Store) for each phase output. The phase selected by the PHASE command will be used for all subsequent phase-dependent commands.

The data byte for the PHASE command is an unsigned binary integer.

Phases 00h through 7Fh are reserved specifically for multiple phase outputs on a device with a single physical address.

Setting the PHASE to FFh means that all subsequent commands are applied to all the phase outputs. The default value will be set to FFh, allowing backward compatibility with single-phase commands.

It is possible to create PMBus devices that have multiple pages, each of which may control one or more phases. The PMBus device product literature shall clearly state the relationship between the PAGE and PHASE for the device.

Examples of functions that could take advantage of the PHASE command are READ_IOUT, IOUT_CAL_GAIN, and IOUT_CAL_OFFSET.

Some commands, such as READ_TEMPERATURE, may use a common sensor but be available on all phases of a device. Such implementations are the decision of each device manufacturer or are specified in a PMBus Application Profile. Consult the manufacturer's documents or the Application Profile specification as needed.

11.12. CAPABILITY

This command provides a way for a host system to determine some key capabilities of a PMBus device.

There is one data byte formatted as shown in Table 7.

This command is read only.

Table 7. CAPABILITY COMMAND Data Byte Format

Bits	Description	Value	Meaning
7	Packet Error Checking	0	Packet Error Checking not supported
		1	Packet Error Checking is supported
6:5	Maximum Bus Speed	00	Maximum supported bus speed is 100 kHz
		01	Maximum supported bus speed is 400 kHz
		10	Maximum supported bus speed is 1 MHz

Bits	Description	Value	Meaning
		11	Reserved
4	SMBALERT#	0	The device does not have a SMBALERT# pin and does not support the SMBus Alert Response protocol
		1	The device does have a SMBALERT# pin and does support the SMBus Alert Response protocol
3	Numeric Format	0	Numeric data is in LINEAR11, ULINEAR16, SLINEAR16, or DIRECT format
		1	Numeric data is in IEEE Half Precision Floating Point Format
2	AVSBus Support	0	AVSBus Not Supported
		1	AVSBus Supported
1:0	Reserved	X	Reserved

11.13. QUERY

The QUERY command is used to ask a PMBus device if it supports a given command, and if so, what data formats it supports for that command. This command uses the Block Write-Block Read Process Call described in the SMBus specification [A03].

For the write portion of the process call, the one data byte is the command code of the command being investigated.

For the read portion of the process call, the one data byte is an unsigned binary integer with values as follows:

Table 8. QUERY Command Returned Data Byte Format

Bits	Value	Meaning
7	0	Command is not supported
	1	Command is supported
6	0	Command is not supported for write
	1	Command is supported for write. This value is returned if the devices supports writing to the command, regardless of whether a current WRITE_PROTECT setting allows or prevents writing to this command.
5	0	Command is not supported for read
	1	Command is supported for read
4:2	000	LINEAR11, ULINEAR16, or SLINEAR16 numeric format used (as defined by the specification of the command being queried)
	001	16 bit signed number
	010	IEEE Half Precision Floating Point Format used
	011	Direct Mode Format used

Bits	Value	Meaning
	100	8 bit unsigned number
	101	VID Mode Format used
	110	Manufacturer specific format used
	111	Command does not return numeric data. This is also used for commands that return blocks of data or the SEND BYTE protocol.
1:0	XX	Reserved for future use

If bit [7] is zero, then the rest of the bits are “don’t care”.

For any command listed as reserved, the device shall return the “Command is not supported” response (bit [7] set to 0)

11.14. PAGE_PLUS_WRITE

The PAGE_PLUS_WRITE command is used to set the page within a device, send a command, and send the data for the command in one packet.

The PAGE_PLUS_WRITE command uses the WRITE BLOCK protocol.

An example of the PAGE_PLUS command being used to send a command that has two data bytes to be written and a PEC byte is shown in Figure 23.

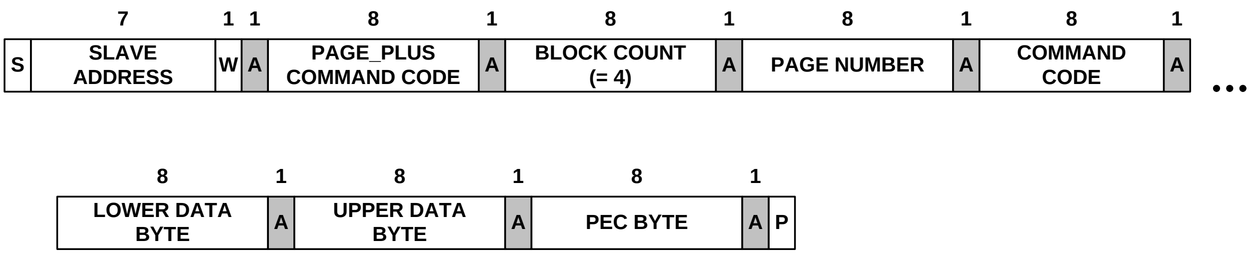


Figure 23. PAGE_PLUS_WRITE Command Example With Data Being Written And PEC

11.15. PAGE_PLUS_READ

The PAGE_PLUS_READ command is used to set the page within a device, send a command, and read the data returned by the command in one packet.

The PAGE_PLUS_READ command uses the BLOCK WRITE – BLOCK READ PROCESS CALL protocol.

An example of the PAGE_PLUS command being used to send a command that has two data bytes to be read and a PEC byte is shown in Figure 24.

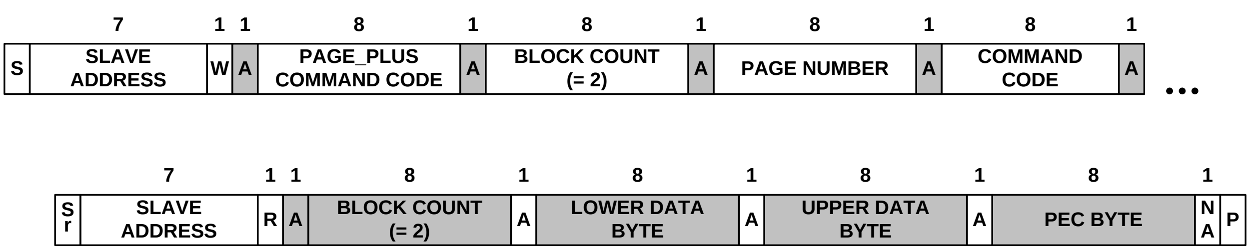


Figure 24. PAGE_PLUS_READ Command Example With Data Being Read And PEC

11.16. Zone Operation Commands

The ZONE_WRITE and ZONE_READ protocols, coupled with the ZONE_CONFIG and ZONE_ACTIVE commands, allow a master to send data to or read data from multiple PMBus devices is a single bus transaction. The ZONE_WRITE and ZONE_READ protocols are described in Part I of the PMBus specification [A01]. Examples of PMBus zone operations are given in PMBus Application Note AN001, *Using The ZONE_READ And ZONE_WRITE Protocols* [R02].

Before the ZONE_READ and ZONE_WRITE protocols can be used by a master, the zone-capable devices on a physical PMBus must be configured using the ZONE_CONFIG command (Section 11.16.1).

Once the zone capable devices are configured, the master must set the Active Write Zone and Active Read zone for the bus using the ZONE_ACTIVE command (Section 11.16.2). The Active Write Zone and Active Read Zone do not have to be the same zone number.

11.16.1. ZONE_CONFIG

The ZONE_CONFIG command is used to assign a PMBus device, which may be a discrete entity at one PMBus address, or a PAGE within an entity on the PMBus, to a specific zone number for ZONE_READ operations and to a specific zone number for ZONE_WRITE operations.

The ZONE_CONFIG is typically used as part of the system initialization process. There are no restrictions, however, on the ZONE_CONFIG being used at any time during a system's operation.

A device's write or read zone may be assigned one of 129 possible zone numbers, ranging from 00h to 7Fh plus FEh. A device's write zone and read zone do not have to be the same.

A device's read zone and/or write zone may be assigned to the "No Zone" by assigning the device's read zone and/or write zone the value FEh. A device whose assigned read zone is FEh shall ignore all ZONE_READ operations. A device whose assigned write zone is FEh shall ignore all ZONE_WRITE operations. Assigning a device to the "No Zone" is used to prevent a zone-capable device from participating in zone operations.

Zone values 80h through BFh are reserved for PMBus product manufacturer's definition. Zone values C0h to FDh are reserved for future use by this specification.

The ZONE_CONFIG command shall never configure a device's read or write zone to the value FFh. The FFh value is used with the ZONE_ACTIVE command for the "All Zone". If the master attempts to set a device's read or write zone to FFh the device shall declare an invalid data fault and respond as described in Section 10.9.3.

The ZONE_CONFIG command has two data bytes as illustrated in Figure 25. The first data byte sets the assigned write zone for the device. The second data byte sets the assigned read zone. The two bytes of data of the ZONE_CONFIG command are shown in Figure 25.

If a device supports PAGEs, then when the ZONE_CONFIG command is received that read and write zone assignments go to the current PAGE. This makes it possible for each PAGE within a device to be assigned to different zones.

The default values for a device's configured write and read zones when a device is powered up or reset shall be specified by the manufacturer in the product literature.

While it is recommended that the configured write and read zones be stored in a nonvolatile memory this is not required.

The ZONE_CONFIG command uses the Write Word and Read Word protocols.

	7	1	1	8	1	8	1	8	1	
S	SLAVE ADDRESS	W	A	ZONE_CONFIG COMMAND CODE	A	ASSIGNED WRITE ZONE	A	ASSIGNED READ ZONE	A	P

Figure 25. ZONE_CONFIG Command

11.16.2. ZONE_ACTIVE

The ZONE_ACTIVE command is used by the master to set the Active Write Zone for ZONE_WRITE operations and the Active Read Zone for ZONE_READ operations. The active zone setting is a property of the entire system attached to a given physical PMBus and is not a property of an individual device attached to that bus.

The ZONE_ACTIVE command has two data bytes. The first data byte is the active zone number for write operations (Active Write Zone). The second data byte is the active zone number for read operations (Active Read Zone).

All devices that support zone operations shall respond to the ZONE_ACTIVE command regardless of their current configuration for read and write zones. If a device has its write zone configured to the "No Zone" (FEh) it shall ignore the Active Write Zone value. If a device has its read zone configured to the "No Zone" it shall ignore the Active Read Zone value.

Values of 00h to 7Fh are used for normal active zones.

Zone values 80h through BFh are reserved for PMBus product manufacturer's definition. Zone values C0h to FDh are reserved for future use by this specification.

The active zone value of FFh is defined as the "All Zone". Every device whose write zone is not configured to be in the "No Zone" (zone number FEh) shall respond to a ZONE_WRITE when the Active Write Zone is set to FFh. Similarly, every device whose read zone is not configured to be in the "No Zone" (zone number FEh) shall respond to a ZONE_READ when the Active Read Zone is set to FFh. The use of the "All Zone" allows a master to broadcast a status request or PMBus command to all devices participating in zone operations regardless of their current zone assignment.

Master devices are prohibited from setting an Active Write Zone or Active Read Zone to the "No Zone" value of FEh. If a device receives a ZONE_ACTIVE command with an Active Write Zone or Active Read Zone value of FEh the device shall declare an invalid data fault and respond as described in Section 10.9.3.

When the ZONE_ACTIVE command is used to set the Active Write Zone and Active Read zone, it shall only be send to the ZONE_WRITE address (37h) as shown in Figure 26. If a PMBus device receives a ZONE_ACTIVE command at its own address with the R/W# bit set for writing (0b), the PMBus device shall treat this as an unsupported command code and respond as described in Section 10.9.2.

The ZONE_ACTIVE command may also be used to read from a PMBus device its currently stored values for Active Write Zone and Active Read Zone. This is done by sending the slave's address followed by the R/W# bit set for read (1b) followed by the ZONE_ACTIVE command. This is shown in Figure 27.

The ZONE_ACTIVE command uses the R/W Word protocol.

	7	1	1	8	1	8	1	8	1	
S	ZONE WRITE ADDRESS	W	A	ZONE_ACTIVE COMMAND CODE	A	ACTIVE WRITE ZONE	A	ACTIVE READ ZONE	A	P

Figure 26. ZONE_ACTIVE Command Used To Set The Active Write Zone And Active Read Zone

	7	1	1	8	1	8	1	8	1	
S	SLAVE ADDRESS	R	A	ZONE_ACTIVE COMMAND CODE	A	STORED ACTIVE WRITE ZONE VALUE	A	STORED ACTIVE READ ZONE VALUE	N	P

Figure 27. Using ZONE_ACTIVE Command To Read The Active Write Zone And Active Read Zone Stored In A Device

12. On, Off And Margin Testing Related Commands

12.1. OPERATION

The OPERATION command is used to configure the operational state of the converter, in conjunction with input from the CONTROL pin. The OPERATION command is used to:

- Turn the PMBus device output on and off with commands sent over the PMBus
- Select the margin state of the device (margin off, margin high, margin low)
- Select whether fault conditions caused by margining are ignored or acted upon
- Select whether the output voltage is set by commands over the PMBus or AVSBus
- Select whether the converter powers down immediately or follows the programmed TOFF_DELAY and TOFF_FALL commands when commanded to turn off the output

The data byte of the OPERATION command is illustrated in Figure 27.

7	6	5	4	3	2	1	0
ON/OFF State	Turn Off Behavior	Voltage Command Source		Margin Fault Response		Transition Control	Reserved

Figure 28. OPERATION Command Data Byte

12.1.1. OPERATION Command Bit [7]

Bit [7] controls whether the PMBus device output is on or off.

If bit [7] is cleared (equals 0), then the output is off.

If bit [7] is set (equals 1), then the output is on.

12.1.2. OPERATION Command Bit [6]

Bit [6] controls the power down behavior.

If bit [7] is set (equals 1), then bit [6] is ignored.

If bit [7] is cleared (equals 0), then:

- If Bit [6] is cleared (equals 0), then the output is turned off immediately and any power down sequencing commands are ignored
- Else if Bit [6] is set (equals 1), then the device powers down following the values set in the TOFF_DELAY and TOFF_DELAY command.

12.1.3. OPERATION Command Bits [5:4]

If the PMBus device output is on (bit [7] = 1), then bits [5:4] control the basic source of the output voltage command.

If bits [5:4] equal 00b, then the nominal output voltage is set by the PMBus VOUT_COMMAND data.

If bits [5:4] equal 01b, then the nominal output voltage is set by the PMBus VOUT_MARGIN_LOW data.

If bits [5:4] equal 10b, then the nominal output voltage is set by the PMBus VOUT_MARGIN_HIGH data.

If bits [5:4] equal 11b, then the nominal output voltage is set by the AVSBus (AVS_VOUT_COMMAND).

It is permissible for a device to power up with bits [5:4] set to 11b, selecting the AVSBus as the source of the nominal output voltage command.

12.1.4. OPERATION Command Bits [3:2]

If the PMBus device output is on (bit [7] = 1), then bits [3:2] select whether a fault is generated if a margin commands causes the output voltage to go beyond a limit set by the VOUT_OV_FAULT_LIMIT or VOUT_UV_FAULT_LIMIT commands.

In some cases, during system qualification testing for example, it may be desirable to program the output voltages well beyond the normal operating limits. In this case, having the PMBus device shut down its output due to a fault condition defeats the purpose of the test. This setting allows a system engineer to prevent the activation of the fault detection circuitry during margin testing.

If bits [3:2] equal 01b, then faults caused by selecting VOUT_MARGIN_HIGH or VOUT_MARGIN_LOW as the nominal output voltage source are ignored.

If bits [3:2] equal 10b, then faults caused by selecting VOUT_MARGIN_HIGH or VOUT_MARGIN_LOW as the nominal output voltage source are acted upon according to the settings of the VOUT_OV_FAULT_RESPONSE and VOUT_IV_FAULT_RESPONSE data bytes.

12.1.5. OPERATION Command Bit [1]

Bit [1] controls how the nominal output voltage command is updated, or not, when control is passed from the AVSBus to the PMBus.

If bit [1] of the OPERATION command is set, then the value of the VOUT_COMMAND data must be updated with the AVSBus Target Rail Voltage before the multiplexor switches its output to equal the VOUT_COMMAND data. Note that a side effect of this requirement is that a system host that reads the VOUT_COMMAND may get back a value that is not the same as it had previously written to VOUT_COMMAND.

If bit [1] of the OPERATION command is not set, the value of VOUT_COMMAND is not changed before the multiplexor switches its output to be equal to the VOUT_COMMAND value. If there is a difference between the AVSBus Target Rail Voltage and the VOUT_COMMAND value, the output voltage must transition at the rate set by the VOUT_TRANSITION_RATE command (if supported, see Section 13.8). If the VOUT_TRANSITION_RATE command is not supported, the output voltage transition rate will be function of the device design and output loading.

12.1.6. OPERATION Command Bit [0]

This bit is reserved for future use.

12.1.7. OPERATION Command Invalid Data

If a PMBus device receives an OPERATION command data byte that attempts to configure or operate the device in an unsupported manner then the device shall treat this as invalid data, declare a communications fault, and respond as described in Section 10.8. Examples include, but are not limited to, attempting to enable the AVSBus when the AVSBus is not supported, attempting to margin high or low when margining is not supported, or attempting to set bits [3:2] to 00b or 01b.

Table 9. OPERATION Command Data Byte Contents

Bit Number						Device State/Response			
7	**6**	**5:4**	**3:2**	**1**	**0**	**On/ Off**	**Power Off Behavior**	**Output Voltage Command Source**	**Device Response**
0	0	XX	XX	X	Reserved	Off	Immediate Off	N/A	N/A
0	1	XX	XX	X		Off	Power Down Sequencing	N/A	N/A
1	X	00	XX	X		On	N/A	VOUT_COMMAND	AVSBus Target Rail Voltage Is Updated Upon Transfer Of Control To The AVSBus Master
1	X	01	01	X		On	N/A	VOUT_MARGIN_ LOW	Ignore Faults When Margined
1	X	01	10	X		On	N/A	VOUT_MARGIN_ LOW	Act On Faults When Margined
1	X	10	01	X		On	N/A	VOUT_MARGIN_ HIGH	Ignore Faults When Margined
1	X	10	10	X		On	N/A	VOUT_MARGIN_ HIGH	Act On Faults When Margined
1	X	11	XX	0		On	N/A	AVSBus	VOUT_COMMAND Value Is Not Updated Upon Transfer Of Control To The PMBus
1	X	11	XX	1		On	N/A	AVSBus	VOUT_COMMAND Value Is Updated Upon Transfer Of Control To The PMBus

12.2. ON_OFF_CONFIG

The ON_OFF_CONFIG command configures the combination of CONTROL pin input and serial bus commands needed to turn the unit on and off. This includes how the unit responds when power is applied.

The default response for any PMBus device is specified by the device manufacturer.

The details of the ON_OFF_CONFIG data byte are shown in Table 10.

Example conditions:

- If bit [4] is cleared, then the unit powers up and operates any time bias power is available regardless of the setting of bits [3:0].
- If bit [4] is set, bit [3] is set, and bit [2] is cleared, then the unit is turned on and off only by commands received over the serial bus.
- If bit [4] is set, bit [3] is cleared, and bit [2] is set, then the unit is turned on and off only by the CONTROL pin.

- If bit [4] is set, bit [3] is set, and bit [2] is set, then the unit is turned on and off only when both the commands received over the serial bus AND the CONTROL pin are commanding the device to be on.
- If either a command from the serial bus OR the CONTROL pin commands the unit to be off, the unit turns off.

If a device receives a data byte that is not listed in Table 10, then the device shall treat this as invalid data, declare a communications fault and respond as described in Section 10.8.

Table 10. ON_OFF_CONFIG Data Byte

Bit Number	Purpose	Bit Value	Meaning
[7:5]		000	Reserved For Future Use
4	Sets the default to either operate any time power is present or for the on/off to be controlled by CONTROL pin and serial bus commands	0	Unit powers up any time power is present regardless of state of the CONTROL pin
		1	Unit does not power up until commanded by the CONTROL pin and OPERATION command (as programmed in bits [3:0]).
3	Controls how the unit responds to commands received via the serial bus	0	Unit ignores the on/off portion of the OPERATION command from serial bus
		1	To start, the unit requires that the on/off portion of the OPERATION command is instructing the unit to run. Depending on bit [2], the unit may also require the CONTROL pin to be asserted for the unit to start and energize the output.
2	Controls how the unit responds to the CONTROL pin	0	Unit ignores the CONTROL pin (on/off controlled only the OPERATION command)
		1	Unit requires the CONTROL pin to be asserted to start the unit. Depending on bit [3], the OPERATION command may also be required to instruct the device to start before the output is energized.
1	Polarity of the CONTROL pin	0	Active low (Pull pin low to start the unit)
		1	Active high (Pull high to start the unit)
0	CONTROL pin action when commanding the unit to turn	0	Use the programmed turn off delay (Section 16.5) and fall time (Section 16.6)

Bit Number	Purpose	Bit Value	Meaning
	off	1	Turn off the output and stop transferring energy to the output as fast as possible. The device's product literature shall specify whether or not the device sinks current to decrease the output voltage fall time.

13. Output Voltage Related Commands

13.1. VOUT_MODE

The operation of the VOUT_MODE command is described in Section 8.

13.2. VOUT_COMMAND

The operation of the VOUT_COMMAND command is described in Section 8.

13.3. VOUT_TRIM

The VOUT_TRIM command is used to apply a fixed offset voltage to the output voltage command value. It is most typically used by the end user to trim the output voltage at the time the PMBus device is assembled into the end user's system.

The VOUT_TRIM has two data bytes formatted as a two's complement binary integer (SLINEAR16 format). The effect of this command depends on the settings of the VOUT_MODE command (Section 8).

This command may not be used if the unit is working with the VID format for output voltage. If an attempt is made to apply this command when the unit is operating in VID format, the device must reject the command with an invalid data fault as described in Section 10.9.

The default value is 0000h.

13.4. VOUT_CAL_OFFSET

The VOUT_CAL_OFFSET command is used to apply a fixed offset voltage to the output voltage command value. It is most typically used by the PMBus device manufacturer to calibrate a device in the factory.

The VOUT_CAL_OFFSET has two data bytes formatted as a two's complement binary integer (SLINEAR16 format). The effect of this command depends on the settings of the VOUT_MODE command (Section 8).

This command may not be used if the unit is working with the VID format for output voltage. If an attempt is made to apply this command when the unit is operating in VID format, the device must reject the command as described in Section 4.1 of the PMBus specification, Part I [A01].

The default value is 0000h.

13.5. VOUT_MAX

The VOUT_ MAX command sets an upper limit on the output voltage the unit can command regardless of any other commands or combinations. The intent of this command is to provide a safeguard against a user accidentally setting the output voltage to a possibly destructive level rather than to be the primary output overvoltage protection.

If a PMBus device supports this command, it must be able to detect that an attempt has been made to program the output to a voltage in excess of the value set by the VOUT_MAX command. This will be treated as a warning condition and not a fault condition. If an attempt is made to program the output voltage higher than the limit set by this command, the device shall respond as follows:

- The commanded output voltage shall be set to VOUT_MAX,
- The NONE OF THE ABOVE bit shall be set in the STATUS_BYTE,
- The VOUT bit shall be set in the STATUS_WORD,
- The VOUT_MAX_MIN Warning bit shall be set in the STATUS_VOUT register (Section 17.3), and
- The device shall notify the host as described in Section 10.2.1.

The data bytes are two bytes formatted according the setting of the VOUT_MODE command (Section 8).

13.6. VOUT_MARGIN_HIGH

This VOUT_MARGIN_HIGH command loads the unit with the voltage to which the output is to be changed when the OPERATION command is set to “Margin High”.

The data bytes are two bytes formatted according the setting of the VOUT_MODE command (Section 8).

13.7. VOUT_MARGIN_LOW

This VOUT_MARGIN_LOW command loads the unit with the voltage to which the output is to be changed when the OPERATION command is set to “Margin Low”

The data bytes are two bytes formatted according the setting of the VOUT_MODE command (Section 8).

13.8. VOUT_TRANSITION_RATE

When a PMBus device receives either a VOUT_COMMAND or OPERATION (Margin High, Margin Low, Margin Off) that causes the output voltage to change, this command sets the rate in mV/µs at which the output should change voltage. This commanded rate of change does not apply when the unit is commanded to turn on or to turn off.

The VOUT_TRANSITION_RATE command has two data bytes encoded in any format given in Section 7. The PMBus device product literature shall clearly state which format the device uses.

The maximum possible positive value of the two data bytes indicates that the device should make the transition as quickly as possible.

13.9. VOUT_DROOP

The VOUT_DROOP sets the rate, in mV/A (mΩ) at which the output voltage decreases (or increases) with increasing (or decreasing) output current for use with Adaptive Voltage Positioning requirements and passive current sharing schemes.

Each device implements the droop calculation based on its own current with the value with which it has been programmed, regardless of whether or not any other units are operating with their outputs in parallel.

For devices that can sink output current (negative output current), the output voltage continues to increase as the output current is negative.

This command has two data bytes encoded in any format given in Section 7. The PMBus device product literature shall clearly state which format the device uses.

The default value is 0 mΩ.

13.10. VOUT_SCALE_LOOP

The operation of this command is discussed in Section 9.1.

This command has two data bytes encoded in any format given in Section 7. The PMBus device product literature shall clearly state which format the device uses.

The value is dimensionless.

The default value is 1.

13.11. VOUT_SCALE_MONITOR

The operation of this command is discussed in Section 9.1.

This command has two data bytes encoded in any format given in Section 7. The PMBus device product literature shall clearly state which format the device uses.

The value is dimensionless.

The default value is 1.

13.12. VOUT_MIN

The VOUT_ MIN command sets a lower limit on the output voltage the unit can command regardless of any other commands or combinations. The intent of this command is to provide a safeguard against a user accidentally setting the output voltage to a possibly destructive level rather than to be the primary output undervoltage protection.

If a PMBus device supports this command, it must be able to detect that an attempt has been made to program the output to a voltage less than the value set by the VOUT_MIN command. This will be treated as a warning condition and not a fault condition. If an attempt is made to program the output voltage lower than the limit set by this command, the device shall respond as follows:

- The commanded output voltage shall be set to VOUT_MIN,
- The NONE OF THE ABOVE bit shall be set in the STATUS_BYTE,
- The VOUT bit shall be set in the STATUS_WORD,
- The VOUT_MAX_MIN Warning bit shall be set in the STATUS_VOUT register (Section 17.3), and
- The device shall notify the host as described in Section 10.2.1.

The data bytes are two bytes formatted according the setting of the VOUT_MODE command (Section 8).

14. Other Commands

14.1. COEFFICIENTS

The COEFFICIENTS command is used to retrieve the *m*, *b* and *R* coefficients needed by data in the DIRECT format.

This command uses the Block Write-Block Read Process Call as described in the SMBus specification [A03].

For the write portion of the process call, the byte count is two and there are two data bytes. The first data byte is the command code from Table 31 of the command of interest. The second data byte indicates whether the host requesting the coefficients needed to encode a value to be written to the device or the coefficients needed to decode a value read from the device. A value of 01h in the second data byte indicates that the coefficients needed to decode a value read from the device are being requested. A value of 00h in the second byte indicates that the coefficients needed to encode a value for writing to the PMBus device are being requested.

For the read portion of the process call, the byte count is five and the five bytes returned are (in this order):

- Lower byte of *m*,
- Upper byte of *m*,
- Lower byte of *b*,
- Upper byte of *b*,
- Single byte of *R*.

More information on the function and application of this command is given in Section 7.4. An example of the packet construction for retrieving the coefficients from a PMBus device using PEC is shown in Figure 28.

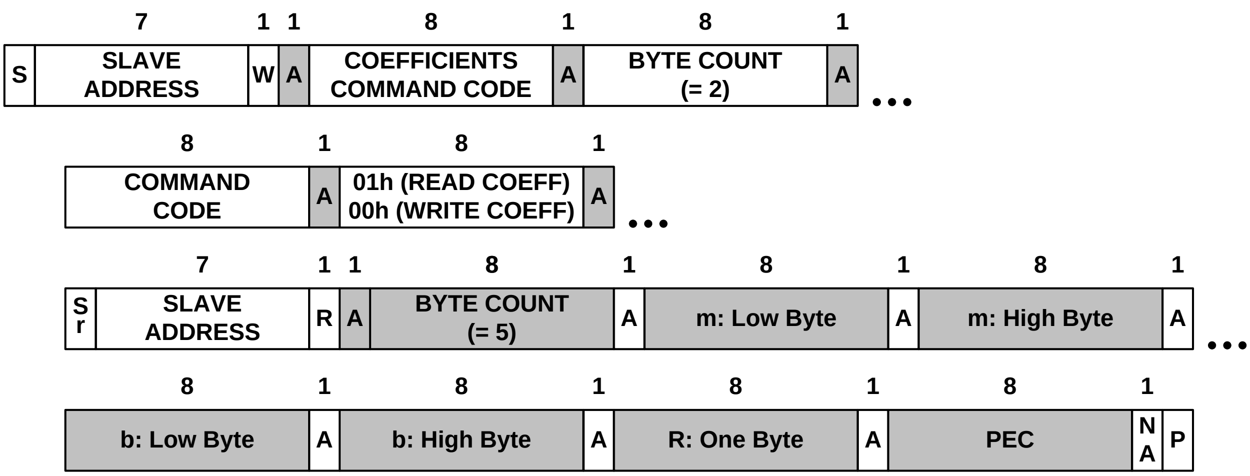


Figure 29. Retrieving Coefficients Using PEC

14.2. POUT_MAX

The POUT_MAX commands set the output power, in watts, at which the unit starts regulating in constant power mode instead of constant voltage. This command is typically used in systems that charge batteries.

The POUT_MAX command has two data bytes encoded in any format given in Section 7. The PMBus device product literature shall clearly state which format the device uses.

The default value is specified by the manufacturer.

14.3. MAX_DUTY

The MAX_DUTY command sets the maximum duty cycle, in percent, of the unit's power conversion stage.

This command has two data bytes encoded in any format given in Section 7. The PMBus device product literature shall clearly state which format the device uses.

14.4. FREQUENCY_SWITCH

The FREQUENCY_SWITCH command sets the switching frequency, in kHz, of a PMBus device.

This command has two data bytes encoded in any format given in Section 7. The PMBus device product literature shall clearly state which format the device uses.

14.5. VIN_ON

The VIN_ON command sets the value of the input voltage, in Volts, at which the unit should start power conversion.

This command has two data bytes encoded in any format given in Section 7. The PMBus device product literature shall clearly state which format the device uses.

14.6. VIN_OFF

The VIN_OFF command sets the value of the input voltage, in Volts, at which the unit, once operation has started, should stop power conversion.

This command has two data bytes encoded in any format given in Section 7. The PMBus device product literature shall clearly state which format the device uses.

14.7. INTERLEAVE

The INTERLEAVE command is used to arrange multiple units so that their switching periods can be distributed in time. This may be used to facilitate paralleling of multiple units or to reduce ac currents injected into the power bus.

To get best advantage from setting the INTERLEAVE value, the units should have their switching frequency clocks well synchronized.

The INTERLEAVE command data bytes include three pieces of information:

- A group identification number (4 bits),
- The number of units in the group (4 bits) and
- The interleave order for this particular unit (4 bits). This number ranges in value from zero to one less than the number of units in the group.

The group identification number allows for up to fifteen groups. Group Identification Number 0 is reserved to mean not a member of an interleaved group. If the group identification number is 0, then the number of units in the group and the interleave order shall also be 0.

The format of the data bytes is shown in Table 11.

Table 11. INTERLEAVE Data Bytes Format

Byte	High Byte								Low Byte							
Bit Number	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
Contents	Not Used				Group ID Number				Number In Group				Interleave Order			
Default Value	00				00				00				00			

An example of the function of the INTERLEAVE command is shown in Figure 29. In this example, there are four devices in Group Number 9. The first device, UNIT 1, is assigned Interleave Order 0; Unit 2 is assigned Interleave Order 1 and so forth. Unit 1, first in interleave order, starts its switching cycle when the Master Clock (not defined by the PMBus protocol) starts a new switching cycle. Unit 2, second in the interleave order, starts its on-time after a delay of one quarter of the Master Clock period. The one quarter cycle delay for Unit 2 is calculated as:

$$Tdelay(Unit\ 2) = \frac{Interleave\ Order\ Of\ Unit\ 2}{Number\ In\ Group} \bullet T_S = \frac{0001b}{0100b} \bullet T_S = \frac{1}{4} \bullet T_S$$

In general, for Unit N, the delay time from the triggering edge of the Master Clock to the start of Unit N's on time is:

$$Tdelay(Unit\ N) = \frac{Interleave\ Order\ Of\ Unit\ N}{Number\ In\ Group} \bullet T_S$$

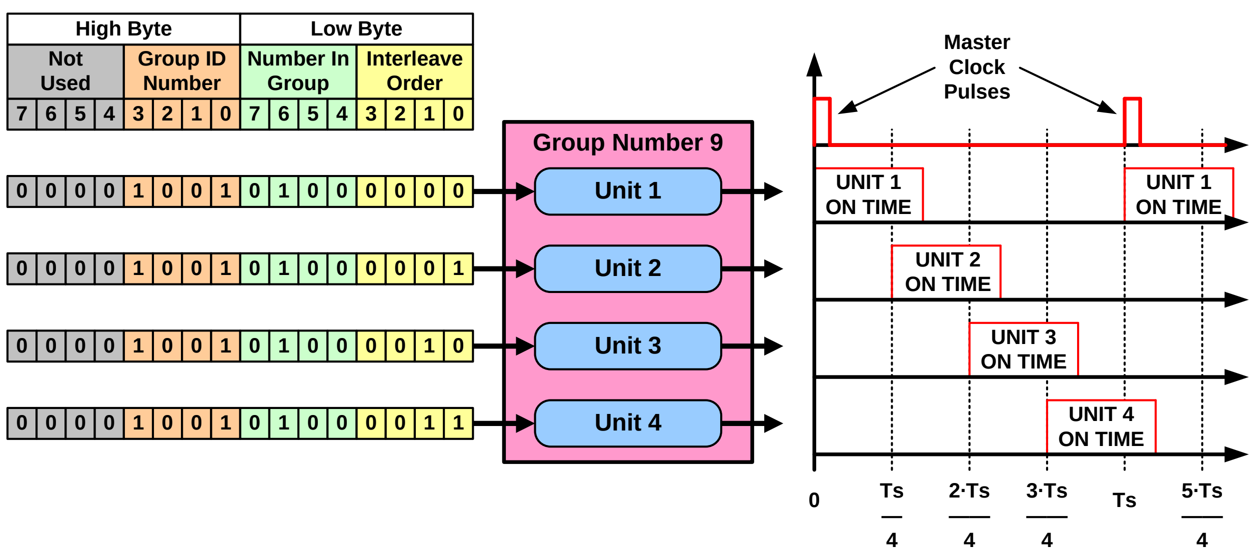


Figure 30. Illustration Of The INTERLEAVE Command Function

14.8. IOUT_CAL_GAIN

The IOUT_CAL_GAIN command is used to set the ratio of the voltage at the current sense pins to the sensed current. For devices using a fixed current sense resistor, it is typically the same value as the resistance of the resistor.

This command may also be used with the IOUT_CAL_OFFSET command (Section 14.9) to calibrate the device's current sensing circuit (Section 9.3).

The units of the IOUT_CAL_GAIN factor are milliohms.

This command has two data bytes encoded in any format given in Section 7. The PMBus device product literature shall clearly state which format the device uses.

The default value is 0 mΩ.

14.9. IOUT_CAL_OFFSET

The IOUT_CAL_OFFSET is used to null out any offsets in the output current sensing circuit. This command is most often used in conjunction with the IOUT_CAL_GAIN command (above) to minimize the error of the current sensing circuit.

This command may also be used with the IOUT_CAL_GAIN command (Section 14.8) to calibrate the device's current sensing circuit (Section 9.3).

The units of the IOUT_CAL_OFFSET are Amperes.

This command has two data bytes encoded in any format given in Section 7. The PMBus device product literature shall clearly state which format the device uses.

The default value is 0 A.

14.10. FAN_CONFIG_1_2

The FAN_CONFIG_1_2 is used to configure up to two fans associated with one PMBus device.

The first part of the configuration byte tells the PMBus device whether or not a fan associated with position 1 (or 2) is installed. Any combination of fan installation is permitted (no fans, a fan in position 1 and no fan in position 2, no fan in position 1 and a fan in position 2, fans in both positions).

The second part of the configuration tells the device whether the fan speed commands are in RPM or PWM duty cycle (in percent). Section 14.11 describes the command for setting fan speed. These settings do not have to be the same for Fan 1 and Fan 2.

The third part of the configuration data tells the PMBus device the number of tachometer pulses per revolution each fan provides. This information is needed for commanding and reporting fan speed in RPM. Two bits are provided for each fan. These settings do not have to be the same for Fan 1 and Fan 2. The binary values of these bits map to pulses per revolution as follows:

- 00b = 1 pulse per revolution,
- 01b = 2 pulses per revolution,
- 10b = 3 pulses per revolution, and
- 11b = 4 pulses per revolution.

This command has one data byte formatted as follows:

Table 12. FAN_CONFIG_1_2 Data Byte Format

Bit(s)	Value	Meaning
7	1	A Fan Is Installed In Position 1
	0	No Fan Is Installed In Position 1
6	1	Fan 1 Is Commanded In RPM
	0	Fan 1 Is Commanded In Duty Cycle

Bit(s)	Value	Meaning
5:4	00b-11b	Fan 1 Tachometer Pulses Per Revolution
3	1	A Fan Is Installed In Position 2
	0	No Fan Is Installed In Position 2
2	1	Fan 2 Is Commanded In RPM
	0	Fan 2 Is Commanded In Duty Cycle
1:0	00b-11b	Fan 2 Tachometer Pulses Per Revolution

Each fan can have its command format set individually. Not all fans must have the same command format.

The device manufacturer's product literature shall give the default values.

14.11. FAN_CONFIG_3_4

The FAN_CONFIG_3_4 is used to configure up to two fans associated with one PMBus device.

The settings of this command are independent of whether or not there are fan in positions 1 and 2.

The first part of the configuration byte tells the PMBus device whether or not a fan associated with position 3 (or 4) is installed. Any combination of fan installation is permitted (no fans, a fan in position 3 and no fan in position 4, no fan in position 3 and a fan in position 4, fans in both positions).

The second part of the configuration tells the device whether the fan speed commands are in RPM or PWM duty cycle (in percent). Section 14.11 describes the command for setting fan speed. These settings do not have to be the same for Fan 3 and Fan 4.

The third part of the configuration data tells the PMBus device the number of tachometer pulses per revolution each fan provides. This information is needed for commanding and reporting fan speed in RPM. Two bits are provided for each fan. These settings do not have to be the same for Fan 3 and Fan 4. The binary values of these bits map to pulses per revolution as follows:

- 00b = 1 pulse per revolution,
- 01b = 2 pulses per revolution,
- 10b = 3 pulses per revolution, and
- 11b = 4 pulses per revolution.

This command has one data byte formatted as follows:

Table 13. FAN_CONFIG_3_4 Data Byte Format

Bit(s)	Value	Meaning
7	1	A Fan Is Installed In Position 3
	0	No Fan Is Installed In Position 3
6	1	Fan 3 Is Commanded In RPM

Bit(s)	Value	Meaning
	0	Fan 3 Is Commanded In Duty Cycle
5:4	00b-11b	Fan 3 Tachometer Pulses Per Revolution
3	1	A Fan Is Installed In Position 4
	0	No Fan Is Installed In Position 4
2	1	Fan 4 Is Commanded In RPM
	0	Fan 4 Is Commanded In Duty Cycle
1:0	00b-11b	Fan 4 Tachometer Pulses Per Revolution

14.12. FAN_COMMAND_n

The FAN_COMMAND_1, FAN_COMMAND_2, FAN_COMMAND_3 and FAN_COMMAND_4 commands are used to adjust the operation of up to four fans contained in the PMBus device or in the host system. For fans contained in the PMBus device, the host system may override the commanded values if needed to maintain proper system temperatures.

This command has two data bytes encoded in any format given in Section 7. The PMBus device product literature shall clearly state which format the device uses. The command may be in RPM or duty cycle, as set by the FAN_COMMAND_CONFIG command (Section 14.10).

The default value is specified in the device manufacturer product literature.

14.13. POWER_MODE

The POWER_MODE command is used to set or read the PMBus device power conversion mode of operation.

If a device supports PMBus, AVSBus, and POWER_MODE, the register that stores the POWER_MODE setting is used by both the PMBus and AVSBus. The requirement is that only the source controlling the output voltage, PMBus or AVSBus, can change the POWER_MODE.

When the output voltage is controlled by the PMBus commands VOUT_COMMAND, VOUT_MARGIN_HIGH, or VOUT_MARGIN_LOW (OPERATION command bits [5:4] are not equal to 11b) then:

- The system host can change the POWER_MODE and
- The AVSBus host is not permitted to change the POWER_MODE.

When the output voltage is controlled by the AVSBus (OPERATION command bits [5:4] are equal to 11b), then:

- The AVSBus bus host can change the POWER_MODE and
- The system host is not permitted change the POWER_MODE.

Table 14. POWER_MODE Command Data Byte Format

Bits [7:3]	Bits [2:0]	Power Mode
Reserved	000	Maximum Efficiency
Reserved	001	Reserved
Reserved	010	Reserved
Reserved	011	Maximum Power
Reserved	100	Manufacturer Defined
Reserved	101	Manufacturer Defined
Reserved	110	Manufacturer Defined
Reserved	111	Manufacturer Defined

Maximum efficiency means that the PMBus device is configured to operate with maximum efficiency. This can be achieved with techniques such as, but not limited to, phase shedding, reduction of switching frequency, pulse skipping, or discontinuous mode. How the device achieves maximum efficiency is left to the device manufacturer and is to be described in the device's product literature.

Maximum power means the PMBus device is configured to provide its maximum rated output power while also meeting all specifications for dynamic response. This can be achieved, for example, by operating all phases at the maximum switching frequency. How the device is configured to operate in this condition is left to the device manufacturer and is to be described in the device's product literature.

Manufacturer defined power modes, if implemented, are to be described in the device's product literature.

15. Fault Related Commands

15.1. CLEAR_FAULTS

The CLEAR_FAULTS command is used to clear any fault bits that have been set. This command clears all bits in all status registers simultaneously. At the same time, the device negates (clears, releases) its SMBALERT# signal output if the device is asserting the SMBALERT# signal.

The CLEAR_FAULTS command does not cause a unit that has latched off for a fault condition to restart. Units that have shut down for a fault condition are restarted as described in Section 10.7.

If the fault is still present when the bit is cleared, the fault bit shall immediately be set again and the host notified by the usual means.

This command is write only. There is no data byte for this command.

15.2. VOUT_OV_FAULT_LIMIT

The VOUT_OV_FAULT_LIMIT command sets the value of the output voltage measured at the sense or output pins that causes an output overvoltage fault.

The data bytes are two bytes formatted according to the setting of the VOUT_MODE command (Section 8).

The default value is specified by the device manufacturer in the product literature.

15.3. VOUT_OV_FAULT_RESPONSE

The VOUT_OV_FAULT_RESPONSE command instructs the device on what action to take in response to an output overvoltage fault. The data byte is in the format given in Section 10.5.1.

The device also:

- Sets the VOUT_OV_FAULT bit in the STATUS_BYTE,
- Sets the VOUT bit in the STATUS_WORD,
- Sets the VOUT_OV_FAULT bit in the STATUS_VOUT register, and
- Notifies the host as described in Section 10.2.2.

The delay time unit is specified by the device manufacturer in the device's product literature.

The default value is specified by the device manufacturer in the product literature.

15.4. VOUT_OV_WARN_LIMIT

The VOUT_OV_WARN_LIMIT command sets the value of the output voltage at the sense or output pins that causes an output voltage high warning. This value is typically less than the output overvoltage threshold.

The data bytes are two bytes formatted according the setting of the VOUT_MODE command (Section 8).

In response to the VOUT_OV_WARN_LIMIT being exceeded, the device:

- Sets the NONE OF THE ABOVE bit in the STATUS_BYTE,
- Sets the VOUT bit in the STATUS_WORD,
- Sets the VOUT_OV_WARNING bit in the STATUS_VOUT register, and
- Notifies the host as described in Section 10.2.1.

The default value is specified by the device manufacturer in the product literature.

15.5. VOUT_UV_WARN_LIMIT

The VOUT_UV_WARN_LIMIT command sets the value of the output voltage at the sense or output pins that causes an output voltage low warning. This value is typically greater than the output undervoltage fault threshold.

This warning is masked until the unit reaches the programmed output voltage. This warning is also masked when the unit is disabled.

The data bytes are two bytes formatted according to the setting of the VOUT_MODE command (Section 8).

In response to the VOUT_UV_WARN_LIMIT being exceeded, the device:

- Sets the NONE OF THE ABOVE bit in the STATUS_BYTE,
- Sets the VOUT bit in the STATUS_WORD,
- Sets the VOUT_UV_WARNING bit in the STATUS_VOUT register, and
- Notifies the host as described in Section 10.2.1.

The default value is specified by the device manufacturer in the product literature.

15.6. VOUT_UV_FAULT_LIMIT

The VOUT_UV_FAULT_LIMIT command sets the value of the output voltage at the sense or output pins that causes an output undervoltage fault.

This fault is masked until the unit reaches the programmed output voltage. This fault is also masked when the unit is disabled.

The data bytes are two bytes formatted according the setting of the VOUT_MODE command (Section 8).

The default value is 0000h.

15.7. VOUT_UV_FAULT_RESPONSE

The VOUT_UV_FAULT_RESPONSE command instructs the device on what action to take in response to an output undervoltage fault. The data byte is in the format given in Section 10.5.1.

The device also:

- Sets the NONE OF THE ABOVE bit in the STATUS_BYTE,
- Sets the VOUT bit in the STATUS_WORD,
- Sets the VOUT_UV_FAULT bit in the STATUS_VOUT register, and
- Notifies the host as described in Section 10.2.2.

The delay time unit is specified by the device manufacturer in the device's product literature.

The default value is specified by the device manufacturer in the product literature.

15.8. IOUT_OC_FAULT_LIMIT

The IOUT_OC_FAULT_LIMIT command sets the value of the output current, in Amperes, that causes the overcurrent detector to indicate an overcurrent fault condition.

The two data bytes are encoded in any format given in Section 7. The PMBus device product literature shall clearly state which format the device uses.

The default value is specified by the device manufacturer in the product literature.

15.9. IOUT_OC_FAULT_RESPONSE

The IOUT_OC_FAULT_RESPONSE command instructs the device on what action to take in response to an output overcurrent fault. The data byte is in the format given in Section10.5.2.

The device also:

- Sets the IOUT_OC_FAULT bit in the STATUS_BYTE,
- Sets the IOUT bit in the STATUS_WORD,
- Sets the IOUT_OC_FAULT bit in the STATUS_IOUT register, and
- Notifies the host as described in Section 10.2.2.

The delay time unit is specified by the device manufacturer in the device's product literature.

The default value is specified by the device manufacturer in the product literature.

15.10. IOUT_OC_LV_FAULT_LIMIT

In the case where the response to an overcurrent condition is to operate in a constant current mode unless the output voltage is pulled below the specified value, the IOUT_OC_LV_FAULT_LIMIT specifies that voltage threshold.

The data bytes are two bytes formatted according the setting of the VOUT_MODE command (Section 8).

The default value is 0000h.

15.11. IOUT_OC_LV_FAULT_RESPONSE

The IOUT_OC_LV_FAULT_RESPONSE command instructs the device on what action to take in response to an output overcurrent fault when the output voltage has been pulled below the specified threshold. The data byte is in the format given in Section 10.5.2.

The device also:

- Sets the IOUT_OC_FAULT bit in the STATUS_BYTE,
- Sets the IOUT bit in the STATUS_WORD,
- Sets the IOUT_OC_LV_FAULT bit in the STATUS_IOUT register, and
- Notifies the host as described in Section 10.2.2.

The delay time unit is specified by the device manufacturer in the device's product literature.

The default value is specified by the device manufacturer in the product literature.

15.12. IOUT_OC_WARN_LIMIT

The IOUT_OV_WARN_LIMIT command sets the value of the output current that causes an output overcurrent warning.

The two data bytes are encoded in any format given in Section 7. The PMBus device product literature shall clearly state which format the device uses.

In response to the IOUT_OC_WARN_LIMIT being exceeded, the device:

- Sets the NONE OF THE ABOVE bit in the STATUS_BYTE
- Sets the IOUT bit in the STATUS_WORD,
- Sets the IOUT_OC_WARNING bit in the STATUS_IOUT register, and
- Notifies the host as described in Section 10.2.1.

The default value is specified by the device manufacturer in the product literature.

15.13. IOUT_UC_FAULT_LIMIT

For units with a synchronous rectifier in the output, current can flow from the unit to the load or from the load into the output. When current is flowing from the unit to the load the unit is said to be sourcing current and the output current declared to be positive. When current is flowing into the unit from the load, the units is said to be sinking current and the current is declared to be negative.

This command sets the maximum output current, in Amperes, that is allowed before action is taken. Note that the IOUT_UC_FAULT_LIMIT value is generally negative, corresponding to a negative output current (current flowing from the load into the output of the device).

The two data bytes are encoded in any format given in Section 7. The PMBus device product literature shall clearly state which format the device uses.

The default value is 0 A.

15.14. IOUT_UC_FAULT_RESPONSE

The IOUT_UC_FAULT_RESPONSE command instructs the device on what action to take in response to an output undercurrent fault. The data byte is in the format given in Section 10.5.2.

For this fault condition, the Inhibit Operation option refers only to stopping the synchronous rectification (allowing the output current to freewheel through the freewheel device) and not to turning off the output (stopping the transfer of energy to the output).

The device also:

- Sets the NONE OF THE ABOVE bit in the STATUS_BYTE,
- Sets the IOUT bit in the STATUS_WORD,
- Sets the IOUT_UC_FAULT bit in the STATUS_IOUT register, and
- Notifies the host as described in Section 10.2.2.

The delay time unit is specified by the device manufacturer in the device's product literature.

The default value is specified by the device manufacturer in the product literature.

15.15. DELETED

Superseded by section 15.33.

15.16. DELETED

Superseded by section 15.34.

15.17. OT_FAULT_LIMIT

The OT_FAULT_LIMIT command sets the temperature of the unit, in degrees Celsius, at which it should indicate an Overtemperature Fault.

The two data bytes are encoded in any format given in Section 7. The PMBus device product literature shall clearly state which format the device uses.

The default value is specified by the device manufacturer in the product literature.

15.18. OT_FAULT_RESPONSE

The OT_FAULT_RESPONSE command instructs the device on what action to take in response to an Overtemperature Fault. The data byte is in the format given in Section 10.5.1.

The device also:

- Sets the TEMPERATURE bit in the STATUS_BYTE,
- Sets the OT_FAULT bit in the STATUS_TEMPERATURE register, and
- Notifies the host as described in Section 10.2.2.

The delay time unit is specified by the device manufacturer in the device's product literature.

The default value is specified by the device manufacturer in the product literature.

15.19. OT_WARN_LIMIT

The OT_WARN_LIMIT command sets the temperature of the unit, in degrees Celsius, at which it should indicate an Overtemperature Warning alarm.

The two data bytes are encoded in any format given in Section 7. The PMBus device product literature shall clearly state which format the device uses.

In response to the OT_WARN_LIMIT being exceeded, the device:

- Sets the TEMPERATURE bit in the STATUS_BYTE,
- Sets the OT_WARNING bit in the STATUS_TEMPERATURE register, and
- Notifies the host as described in Section 10.2.1.

The default value is specified by the device manufacturer in the product literature.

15.20. UT_WARN_LIMIT

The UT_WARN_LIMIT command sets the temperature of the unit, in degrees Celsius, at which it should indicate an Undertemperature Warning alarm.

The two data bytes are encoded in any format given in Section 7. The PMBus device product literature shall clearly state which format the device uses.

In response to the UT_WARN_LIMIT being exceeded, the device:

- Sets the TEMPERATURE bit in the STATUS_BYTE,
- Sets the UT_WARNING bit in the STATUS_TEMPERATURE register, and
- Notifies the host as described in Section 10.2.1.

The default value is specified by the device manufacturer in the product literature.

15.21. UT_FAULT_LIMIT

The UT_FAULT_LIMIT command sets the temperature of the unit, in degrees Celsius, at which it should indicate an Undertemperature Fault.

The two data bytes are encoded in any format given in Section 7. The PMBus device product literature shall clearly state which format the device uses.

The default value is specified by the device manufacturer in the product literature.

15.22. UT_FAULT_RESPONSE

The UT_FAULT_RESPONSE command instructs the device on what action to take in response to an Undertemperature Fault. The data byte is in the format given in Section 10.5.1.

The device also:

- Sets the TEMPERATURE bit in the STATUS_BYTE,
- Sets the UT_FAULT bit in the STATUS_TEMPERATURE register, and
- Notifies the host as described in Section 10.2.2.

The delay time unit is specified by the device manufacturer in the device's product literature.

The default value is specified by the device manufacturer in the product literature.

15.23. VIN_OV_FAULT_LIMIT

The VIN_OV_FAULT_LIMIT command sets the value of the input voltage that causes an Input Overvoltage Fault.

The two data bytes are encoded in any format given in Section 7. The PMBus device product literature shall clearly state which format the device uses.

The default value is specified by the device manufacturer in the product literature.

15.24. VIN_OV_FAULT_RESPONSE

The VIN_OV_FAULT_RESPONSE command instructs the device on what action to take in response to an Input Overvoltage Fault. The data byte is in the format given in Section 10.5.1.

The device also:

- Sets the NONE OF THE ABOVE bit in the STATUS_BYTE,
- Set the INPUT bit in the upper byte of the STATUS_WORD,
- Sets the VIN_OV_FAULT bit in the STATUS_INPUT register, and
- Notifies the host as described in Section 10.2.2.

The delay time unit is specified by the device manufacturer in the device's product literature.

The default value is specified by the device manufacturer in the product literature.

15.25. VIN_OV_WARN_LIMIT

The VIN_OV_WARN_LIMIT command sets the value of the input voltage that causes an input voltage high warning. This value is typically less than the Input Overvoltage Fault threshold.

The two data bytes are encoded in any format given in Section 7. The PMBus device product literature shall clearly state which format the device uses.

In response to the VIN_OV_WARN_LIMIT being exceeded, the device:

- Sets the NONE OF THE ABOVE bit in the STATUS_BYTE,
- Sets the INPUT bit is the upper byte of the STATUS_WORD,
- Sets the VIN_OV_WARNING bit in the STATUS_INPUT register, and
- Notifies the host as described in Section 10.2.1.

The default value is specified by the device manufacturer in the product literature.

15.26. VIN_UV_WARN_LIMIT

The VIN_UV_WARN_LIMIT command sets the value of the input voltage that causes an input voltage low warning. This value is typically greater than the Input Undervoltage Fault threshold, VIN_UV_FAULT_LIMIT (Section 15.27).

This alarm is masked until the input exceeds the value set by the VIN_ON command (Section 14.5) and the unit has been enabled.

The two data bytes are encoded in any format given in Section 7. The PMBus device product literature shall clearly state which format the device uses.

In response to the VIN_UV_WARN_LIMIT being exceeded, the device:

- Sets the NONE OF THE ABOVE bit in the STATUS_BYTE,
- Sets the INPUT bit is the upper byte of the STATUS_WORD,
- Sets the VIN_UV_WARNING bit in the STATUS_INPUT register, and
- Notifies the host as described in Section 10.2.1.

The default value is specified by the device manufacturer in the product literature.

15.27. VIN_UV_FAULT_LIMIT

The VIN_UV_FAULT_LIMIT command sets the value of the input voltage that causes an Input Undervoltage Fault.

This alarm is masked until the input exceeds the value set by the VIN_ON command (Section 14.5) and the unit has been enabled.

The two data bytes are encoded in any format given in Section 7. The PMBus device product literature shall clearly state which format the device uses.

The default value is specified by the device manufacturer in the product literature.

15.28. VIN_UV_FAULT_RESPONSE

The VIN_UV_FAULT_RESPONSE command instructs the device on what action to take in response to an Input Undervoltage Fault. The data byte is in the format given in Section 10.5.1.

The device also:

- Sets the VIN_UV_FAULT bit in the STATUS_BYTE,
- Sets the INPUT bit is the upper byte of the STATUS_WORD,
- Sets the VIN_UV_FAULT bit in the STATUS_INPUT register, and
- Notifies the host as described in Section 10.2.2.

The delay time unit is specified by the device manufacturer in the device's product literature.

15.29. IIN_OC_FAULT_LIMIT

The IIN_OC_FAULT_LIMIT command sets the value of the input current, in Amperes, that causes an Input Overcurrent Fault.

The two data bytes are encoded in any format given in Section 7. The PMBus device product literature shall clearly state which format the device uses.

The default value is specified by the device manufacturer in the product literature.

15.30. IIN_OC_FAULT_RESPONSE

The IIN_OC_FAULT_RESPONSE command instructs the device on what action to take in response to an Input Overcurrent Fault. The data byte is in the format given in Section 10.5.1.

The device also:

- Sets the NONE OF THE ABOVE bit in the STATUS_BYTE,
- Sets the INPUT bit is the upper byte of the STATUS_WORD,
- Sets the IIN_OC_FAULT bit in the STATUS_INPUT register, and
- Notifies the host as described in Section 10.2.2.

The delay time unit is specified by the device manufacturer in the device's product literature.

The default value is specified by the device manufacturer in the product literature.

15.31. IIN_OC_WARN_LIMIT

The IIN_OC_WARN_LIMIT command sets the value of the input current, in Amperes, that causes an Input Overcurrent Warning.

The two data bytes are encoded in any format given in Section 7. The PMBus device product literature shall clearly state which format the device uses.

In response to the IIN_OC_WARN_LIMIT being exceeded, the device:

- Sets the NONE OF THE ABOVE bit in the STATUS_BYTE,
- Sets the INPUT bit is the upper byte of the STATUS_WORD,
- Sets the IIN_OC_WARNING bit in the STATUS_INPUT register, and
- Notifies the host as described in Section 10.2.1.

The default value is specified by the device manufacturer in the product literature.

15.32. POWER_GOOD Signal Limits

For PMBus devices that offer a POWER_GOOD signal, these commands are used for setting the output voltage at which a power good signal should be asserted and negated.

Power Good signals will be device and manufacturer specific. Many factors other than output voltage may be used to determine whether or not the POWER_GOOD signal is to be asserted. PMBus device users are instructed to consult the device manufacturer's product literature for the specifics of the device they are using.

15.32.1. POWER_GOOD_ON

The POWER_GOOD_ON command sets the output voltage at which an optional POWER_GOOD signal should be asserted, indicating that the output voltage is valid. Note that depending on the choice of the device manufacturer that a device may drive a POWER_GOOD signal high or low to indicate that the signal is asserted.

The data bytes are two bytes formatted according the setting of the VOUT_MODE command (Section 8).

The default value is specified by the device manufacturer in the product literature.

15.32.2. POWER_GOOD_OFF

The POWER_GOOD_OFF command sets the output voltage at which an optional POWER_GOOD signal should be negated, indicating that the output voltage is not valid. Note that depending on the choice of the device manufacturer that a device may drive a POWER_GOOD signal high or low to indicate that the signal is negated.

The data bytes are two bytes formatted according the setting of the VOUT_MODE command (Section 8).

The default value is specified by the device manufacturer in the product literature.

15.33. POUT_OP_FAULT_LIMIT

The POUT_OP_FAULT_LIMIT command sets the value of the output power, in watts, that causes an output overpower fault.

The two data bytes are encoded in any format given in Section 7. The PMBus device product literature shall clearly state which format the device uses.

The default value is specified by the device manufacturer in the product literature.

15.34. POUT_OP_FAULT_RESPONSE

The POUT_OP_FAULT_RESPONSE command instructs the device on what action to take in response to an output overpower fault. The data byte is in the format given in Section 10.5.1.

The device also:

- Sets the IOUT_OC bit in the STATUS_BYTE,
- Sets the IOUT/POUT bit is the upper byte of the STATUS_WORD,
- Sets the POUT_OP_FAULT bit in the STATUS_IOUT register, and
- Notifies the host as described in Section 10.2.2.

The delay time unit is specified by the device manufacturer in the device's product literature.

The default value is specified by the device manufacturer in the product literature.

15.35. POUT_OP_WARN_LIMIT

The POUT_OP_WARN_LIMIT command sets the value of the output power, in watts, that causes a warning that the output power is high.

The two data bytes are encoded in any format given in Section 7. The PMBus device product literature shall clearly state which format the device uses.

In response to the POUT_OP_WARN_LIMIT being exceeded, the device:

- Sets the IOUT_OC bit in the STATUS_BYTE,
- Sets the IOUT/POUT bit is the upper byte of the STATUS_WORD,
- Sets the POUT_OP_WARNING bit in the STATUS_IOUT register, and
- Notifies the host as described in Section 10.2.1.

The default value is specified by the device manufacturer in the product literature.

15.36. PIN_OP_WARN_LIMIT

The PIN_OP_WARN_LIMIT command sets the value of the input power, in watts, that causes a warning that the input power is high.

The two data bytes are encoded in any format given in Section 7. The PMBus device product literature shall clearly state which format the device uses.

In response to the PIN_OP_WARN_LIMIT being exceeded, the device:

- Sets the INPUT bit is the upper byte of the STATUS_WORD,
- Sets the PIN_OP_WARNING bit in the STATUS_INPUT register, and

- Notifies the host as described in Section 10.2.1.

The default value is specified by the device manufacturer in the product literature.

15.37. Other Fault And Warning Responses

There are several status bits listed in Section 17 that do not have commands to define thresholds and fault responses. The exact implementation and programmability of these is left to the device manufacturers.

These bits must behave the same way as the fault and warning bits defined in this specification. For example, when a bit is set, the device must notify the host as described in Section 10.2.1. Also, once a bit is set, it must remain set until cleared by the host, as described in Section 10.2.3.

15.38. SMBALERT_MASK Command

The SMBALERT_MASK command may be used to prevent a warning or fault condition from asserting the SMBALERT# signal.

The command format used to block a status bit or bits from causing the SMBALERT# signal to be asserted is shown in Figure 30. The bits in the mask byte align with the bits in the corresponding status register. For example if the STATUS_TEMPERATURE command code were sent with the mask byte 01000000b, then an Overtemperature Warning condition would be blocked from asserting SMBALERT#.

	7	1	1	8	1	8	1	8	1	
S	SLAVE ADDRESS	W	A	SMBALERT_MASK COMMAND CODE	A	STATUS_x COMMAND CODE	A	MASK BYTE	A	P

Figure 31. SMBALERT_MASK Command Packet Format

The command format used by the host to determine the SMBALERT_MASK setting for a given status register is shown in Figure 31.

	7	1	1	8	1	8	1	8	1	
S	SLAVE ADDRESS	W	A	SMBALERT_MASK COMMAND CODE	A	BLOCK COUNT (= 1)	A	STATUS_x COMMAND CODE	A	...

	7	1	1	8	1	8	1	
Sr	SLAVE ADDRESS	R	A	BLOCK COUNT (= 1)	A	MASK BYTE	NA	P

Figure 32. Retrieving The SMBALERT_MASK Setting For A Given Status Register

16. Output Voltage Sequencing Commands

16.1. TON_DELAY

The TON_DELAY command sets the time, in milliseconds, from when a start condition is received (as programmed by the ON_OFF_CONFIG command) until the output voltage starts to rise.

The two data bytes are encoded in any format given in Section 7. The PMBus device product literature shall clearly state which format the device uses.

The default value is 0 ms.

16.2. TON_RISE

The TON_RISE command sets the time, in milliseconds, from when the output starts to rise until the voltage has entered the regulation band.

A value of 0 milliseconds instructs the unit to bring its output voltage to the programmed regulation value as quickly as possible.

The two data bytes are encoded in any format given in Section 7. The PMBus device product literature shall clearly state which format the device uses.

The default value is 0 ms.

16.3. TON_MAX_FAULT_LIMIT

The TON_MAX_FAULT_LIMIT command sets an upper limit, in milliseconds, on how long the unit can attempt to power up the output without reaching the output undervoltage fault limit (Section15.6).

A value of 0 milliseconds means that there is no limit and that the unit can attempt to bring up the output voltage indefinitely.

The two data bytes are encoded in any format given in Section 7. The PMBus device product literature shall clearly state which format the device uses.

The default value of the data bytes is 0 ms.

16.4. TON_MAX_FAULT_RESPONSE

The TON_MAX_FAULT_RESPONSE command instructs the device on what action to take in response to a TON_MAX fault. The data byte is in the format given in Section 10.5.1.

The device also:

- Sets the NONE OF THE ABOVE bit in the STATUS_BYTE,
- Sets the VOUT bit in the STATUS_WORD,
- Sets the TON_MAX_FAULT bit in the STATUS_VOUT register, and
- Notifies the host as described in Section 10.2.2.

The delay time unit is specified by the device manufacturer in the device's product literature.

The default value is specified by the device manufacturer in the product literature.

16.5. TOFF_DELAY

The TOFF_DELAY command sets the time, in milliseconds, from when a stop condition is received (as programmed by the ON_OFF_CONFIG command) until the unit stops transferring energy to the output.

The two data bytes are encoded in any format given in Section 7. The PMBus device product literature shall clearly state which format the device uses.

The default value is 0 ms.

16.6. TOFF_FALL

The TOFF_FALL command sets the time, in milliseconds, from the end of the turn-off delay time (Section 16.5) until the voltage is commanded to zero. Note that this command can only be used with a device whose output can sink enough current to cause the output voltage to decrease at a controlled rate.

A value of 0 milliseconds means that the device should ramp the output voltage down as fast as it can without exceeding the IOUT_UC_FAULT_LIMIT current (Section 15.13).

The two data bytes are encoded in any format given in Section 7. The PMBus device product literature shall clearly state which format the device uses.

The default value is 0 ms.

16.7. TOFF_MAX_WARN_LIMIT

The TON_MAX_WARN_LIMIT command sets an upper limit, in milliseconds, on how long the unit can attempt to power down the output without reaching 12.5% of the output voltage programmed at the time the unit is turned off (Section 16.6).

A value of 0 milliseconds means that there is no limit and that the unit waits indefinitely for the output voltage to decay.

The two data bytes are encoded in any format given in Section 7. The PMBus device product literature shall clearly state which format the device uses.

In response to the TOFF_MAX_WARN_LIMIT being exceeded, the device:

- Sets the NONE OF THE ABOVE bit in the STATUS_BYTE,
- Sets the VOUT bit is the upper byte of the STATUS_WORD,
- Sets the TOFF_MAX Warning bit in the STATUS_VOUT register, and
- Notifies the host as described in Section 10.2.1.
- The default value of the data bytes is 0 milliseconds.

17. Unit Status Commands

This section describes commands to retrieve status information from PMBus units. Status information is binary.

A value of 1 indicates a fault or warning event has occurred and a value of 0 indicates that a fault or warning event has not occurred.

Bits for unsupported features shall be reported as zero.

Figure 32 shows a summary of the status command registers and the mapping from the STATUS_BYTE and STATUS _WORD registers to the other status registers.

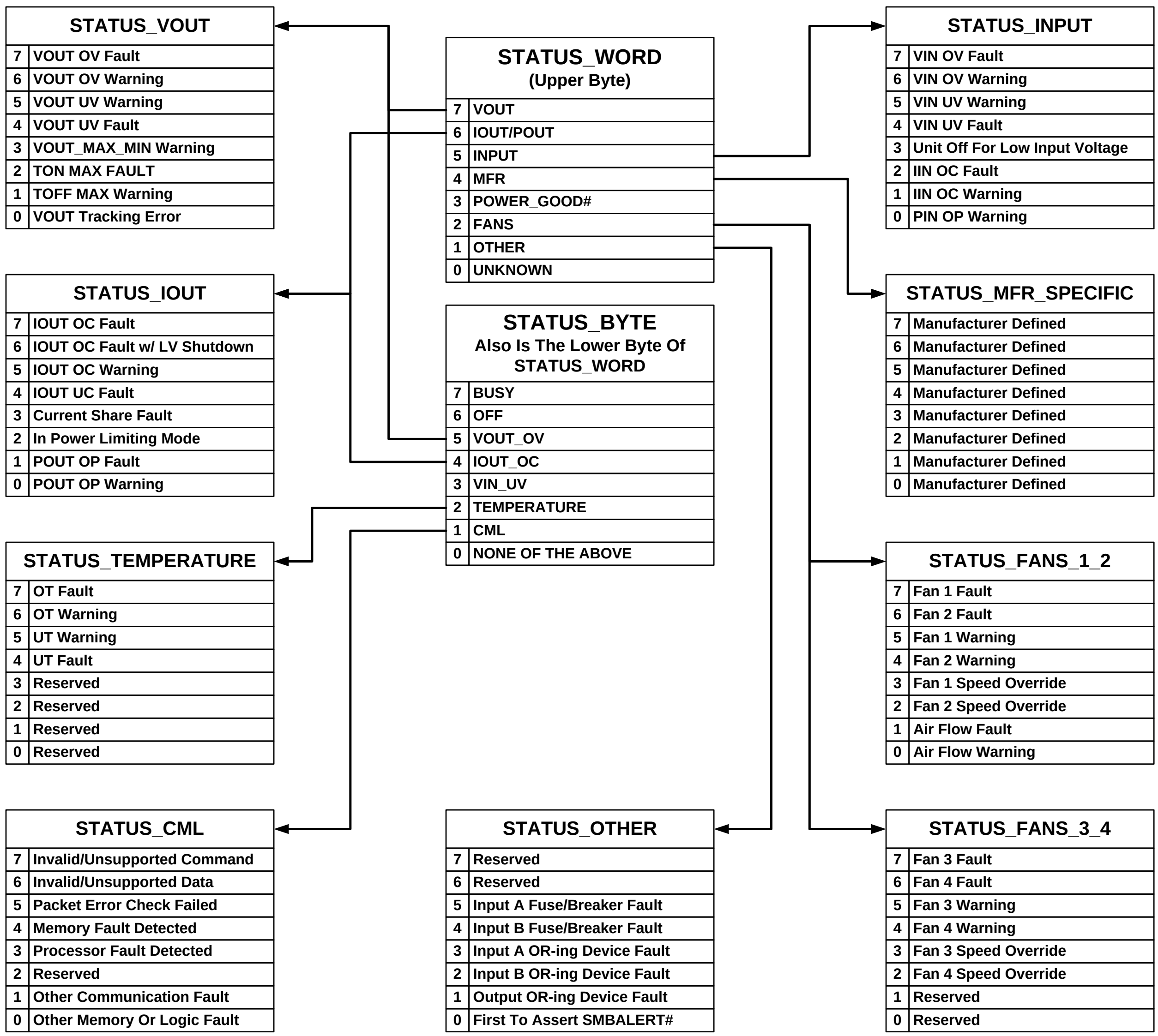


Figure 33. Summary Of The Status Registers

17.1. STATUS_BYTE

The STATUS_BYTE command returns one byte of information with a summary of the most critical faults.

The STATUS_BYTE message content is described in Table 15.

Table 15. STATUS_BYTE Message Contents

Bit	Status Bit Name	Meaning
7	BUSY	A fault was declared because the device was busy and unable to respond.
6	OFF	This bit is asserted if the unit is not providing power to the output, regardless of the reason, including simply not being enabled.
5	VOUT_OV_FAULT	An output overvoltage fault has occurred.

Bit	Status Bit Name	Meaning
4	IOUT_OC_FAULT	An output overcurrent fault has occurred.
3	VIN_UV_FAULT	An input undervoltage fault has occurred.
2	TEMPERATURE	A temperature fault or warning has occurred.
1	CML	A communications, memory or logic fault has occurred.
0	NONE_OF_THE_ ABOVE	A fault or warning not listed in bits [7:1] has occurred.

17.2. STATUS_WORD

The STATUS_WORD command returns two bytes of information with a summary of the unit's fault condition. Based on the information in these bytes, the host can get more information by reading the appropriate status registers.

The low byte of the STATUS_WORD is the same register as the STATUS_BYTE command.

The STATUS_WORD message content is described in Table 16.

Table 16. STATUS_WORD Message Contents

Byte	Bit	Status Bit Name	Meaning
Low	7	BUSY	A fault was declared because the device was busy and unable to respond.
	6	OFF	This bit is asserted if the unit is not providing power to the output, regardless of the reason, including simply not being enabled.
	5	VOUT_OV_FAULT	An output overvoltage fault has occurred.
	4	IOUT_OC_FAULT	An output overcurrent fault has occurred.
	3	VIN_UV_FAULT	An input undervoltage fault has occurred.
	2	TEMPERATURE	A temperature fault or warning has occurred.
	1	CML	A communications, memory or logic fault has occurred.
	0	NONE_OF_THE_ ABOVE	A fault or warning not listed in bits [7:1] of this byte has occurred.
High	7	VOUT	An output voltage fault or warning has occurred.
	6	IOUT/POUT	An output current or output power fault or warning has occurred.
	5	INPUT	An input voltage, input current, or input power fault or warning has occurred.
	4	MFRSPECIFIC	A manufacturer specific fault or warning has occurred.
	3	PG_STATUS#	The POWER_GOOD signal, if present, is negated[1].
	2	FANS	A fan or airflow fault or warning has occurred.
	1	OTHER	A bit in STATUS_OTHER is set
	0	UNKNOWN	A fault type not given in bits [15:1] of the SATUS_WORD has been detected

Note 1: POWER_GOOD is a signal that may be available from the PMBus device that indicates if output voltage of the device is valid (POWER_GOOD asserted) or not

(POWER_GOOD negated). Note that the assertion level of the POWER_GOOD signal could be high or low depending on the choice of the manufacturer. The PG_STATUS# bit reflects the status of the POWER_GOOD signal.

If the POWER_GOOD signal is present and is negated (output voltage is not valid), then the PG_STATUS# bit is set. PG_STATUS# set is interpreted as "the status of the POWER_GOOD signal is negated".

If the POWER_GOOD signal is present and is asserted (output voltage is valid) then the PG_STATUS# bit is cleared. This is interpreted as "the status of the POWER_GOOD signal is 'not negated'", or in other words, the status of the POWER_GOOD signal is indicating that the output voltage is valid.

If the PG_STATUS# bit is set, this indicates that the POWER_GOOD signal, if present, is signaling that the output power is not good.

17.3. STATUS_VOUT

The STATUS_VOUT command returns one data byte with contents as follows:

Table 17. STATUS_VOUT Data Byte

Bit	Meaning
7	VOUT_OV_FAULT (Output Overvoltage Fault)
6	VOUT_OV_WARNING (Output Overvoltage Warning)
5	VOUT_UV_WARNING (Output Undervoltage Warning)
4	VOUT_UV_FAULT (Output Undervoltage Fault)
3	VOUT_MAX_MIN Warning (An attempt has been made to set the output voltage to a value higher than allowed by the VOUT_MAX command (Section 13.5) or lower than the limited allowed by the VOUT_MIN command (Section 13.12).
2	TON_MAX_FAULT
1	TOFF_MAX_WARNING
0	VOUT Tracking Error [1]

[1] The conditions that cause the VOUT Tracking Error bit to be set are defined by each device manufacturer. This status bit is intended to allow the device to notify the host that there was error in output voltage tracking during the most recent power or power down event.

17.4. STATUS_IOUT

The STATUS_IOUT command returns one data byte with contents as follows:

Table 18. STATUS_IOUT Data Byte

Bit	Meaning
7	IOUT_OC_FAULT (Output Overcurrent Fault)
6	IOUT_OC_LV_FAULT (Output Overcurrent And Low Voltage Fault)
5	IOUT_OC_WARNING (Output Overcurrent Warning)
4	IOUT_UC_FAULT (Output Undercurrent Fault)
3	Current Share Fault [1]

Bit	Meaning
2	In Power Limiting Mode [2]
1	POUT_OP_FAULT (Output Overpower Fault)
0	POUT_OP_WARNING (Output Overpower Warning)

[1] The conditions that cause the Current Share Fault bit to be set are defined by each device manufacturer.

[2] This bit is to be asserted when the unit is operating with the output in constant power mode at the power set by the POUT_MAX command (Section 14.2).

17.5. STATUS_INPUT

The STATUS_INPUT command returns one data byte with contents as follows:

Table 19. STATUS_INPUT Data Byte

Bit	Meaning
7	VIN_OV_FAULT (Input Overvoltage Fault)
6	VIN_OV_WARNING (Input Overvoltage Warning)
5	VIN_UV_WARNING (Input Undervoltage Warning)
4	VIN_UV_FAULT (Input Undervoltage Fault)
3	Unit Off For Insufficient Input Voltage [1]
2	IIN_OC_FAULT (Input Overcurrent Fault)
1	IIN_OC_WARNING (Input Overcurrent Warning)
0	PIN_OP_WARNING (Input Overpower Warning)

[1] Either the input voltage has never exceeded the input turn-on threshold (Section14.5) or if the unit did start, the input voltage decreased below the turn-off threshold (Section14.6).

17.6. STATUS_TEMPERATURE

The STATUS_TEMPERATURE command returns one data byte with contents as follows:

Table 20. STATUS_TEMPERATURE Data Byte

Bit	Meaning
7	OT_FAULT (Overtemperature Fault)
6	OT_WARNING (Overtemperature Warning)
5	UT_WARNING (Undertemperature Warning)
4	UT_FAULT (Undertemperature Fault)
3	Reserved
2	Reserved
1	Reserved
0	Reserved

17.7. STATUS_CML (Communications, Logic, And Memory)

The STATUS_CML command returns one data byte with contents as follows:

Table 21. STATUS_CML Data Byte

Bit	Meaning
7	Invalid Or Unsupported Command Received
6	Invalid Or Unsupported Data Received
5	Packet Error Check Failed
4	Memory Fault Detected [1]
3	Processor Fault Detected [2]
2	Reserved
1	A communication fault other than the ones listed in this table has occurred
0	Other Memory Or Logic Fault has occurred. [3]

[1] The conditions that cause the Memory Fault Detected bit to be set, and the response to this condition, are defined by each device manufacturer. One example of an error that would cause this bit to be set is a CRC of the memory that does not match the initial CRC value.

[2] The conditions that cause the Processor Fault Detected bit to be set, and the response to this condition, are defined by each device manufacturer.

[3] The conditions that cause the Other Memory Or Logic Fault Detected bit to be set, and the response to this condition, are defined by each device manufacturer.

17.8. STATUS_OTHER

The STATUS_OTHER command returns one data byte with contents as follows:

Table 22. STATUS_OTHER Data Byte

Bit	Meaning
7	Reserved (Replaced by STATUS_FANS)
6	Reserved (Replaced By STATUS_FANS)
5	Input A Fuse Or Circuit Breaker Fault [1]
4	Input B Fuse Or Circuit Breaker Fault [1]
3	Input A OR-ing Device Fault [2]
2	Input B OR-ing Device Fault [2]
1	Output OR-ing Device Fault [3]
0	First To Assert SMBALERT#

[1] The conditions that cause either of the Input Fuse Or Circuit Breaker Fault bits to be set, and the response to this condition, are defined by each device manufacturer.

[2] The conditions that cause either of the Input OR-ing Device Fault bits to be set, and the response to this condition, are defined by each device manufacturer.

[3] The conditions that cause the Output OR-ing Device Fault bit to be set, and the response to this condition, are defined by each device manufacturer.

17.9. STATUS_MFR_SPECIFIC

The STATUS_MFR_SPECIFIC command returns one data byte with contents as follows:

Table 23. STATUS_MFR_SPECIFIC Data Byte

Bit	Meaning
7	Manufacturer Defined
6	Manufacturer Defined
5	Manufacturer Defined
4	Manufacturer Defined
3	Manufacturer Defined
2	Manufacturer Defined
1	Manufacturer Defined
0	Manufacturer Defined

17.10. STATUS_FANS_1_2

The STATUS_FANS_1_2 command reports on the status of any fans installed in position 1 or position 2.

This command returns one data byte with contents as follows:

Table 24. STATUS_FANS_1_2 Data Byte

Bit	Meaning
7	Fan 1 Fault [1]
6	Fan 2 Fault [1]
5	Fan 1 Warning [2]
4	Fan 2 Warning [2]
3	Fan 1 Speed Overridden [3]
2	Fan 2 Speed Overridden [3]
1	Airflow Fault [4]
0	Airflow Warning [4]

[1] The conditions that cause either of the Fan Fault bits to be set, and the response to this condition, are defined by each device manufacturer. Typically, these bits are set if the fan has failed completely or is simply not able to provide the minimum RPM needed to cool the device or system in which it is embedded. Whether or not these conditions are programmable by the user, and the means to provide such programming, are left to the device manufacturers.

[2] The conditions that cause either of the Fan Warning bits to be set, and the response to this condition, are defined by each device manufacturer. Typically these bits are set if the excitation to the fan to maintain a given RPM has increased over time enough to indicate that the fan should be replaced. Whether or not these conditions are programmable by the user, and the means to provide such programming, are left to the device manufacturers.

[3] These bits are set when an agent or controller sets the fan speed to a higher value than that commanded by the PMBus device. This typically occurs when the PMBus unit is embedded into a larger system and the fans that cool the PMBus unit also cool the system being powered.

[4] The conditions that cause the Airflow Fault or Airflow Warning bits to be set, and the response to this condition, are defined by each device manufacturer. Whether or not these conditions are programmable by the user, and the means to provide such programming, are left to the device manufacturers.

17.11. STATUS_FANS_3_4

The STATUS_FANS_3_4 command reports on the status of any fans installed in position 3 or position 4.

This command returns one data byte with contents as follows:

Table 25. STATUS_FANS_3_4 Data Byte

Bit	Meaning
7	Fan 3 Fault [1]
6	Fan 4 Fault [1]
5	Fan 3 Warning [2]
4	Fan 4 Warning [2]
3	Fan 3 Speed Overridden [3]
2	Fan 4 Speed Overridden [3]
1	Reserved
0	Reserved

[1] The conditions that cause either of the Fan Fault bits to be set, and the response to this condition, are defined by each device manufacturer. Typically, these bits are set if the fan has failed completely or is simply not able to provide the minimum RPM needed to cool the device or system in which it is embedded. Whether or not these conditions are programmable by the user, and the means to provide such programming, are left to the device manufacturers.

[2] The conditions that cause either of the Fan Warning bits to be set, and the response to this condition, are defined by each device manufacturer. Typically these bits are set if the excitation to the fan to maintain a given RPM has increased over time enough to indicate that the fan should be replaced. Whether or not these conditions are programmable by the user, and the means to provide such programming, are left to the device manufacturers.

[3] These bits are set when an agent or controller sets the fan speed to a higher value than that commanded by the PMBus device. This typically occurs when the PMBus unit is embedded into a larger system and the fans that cool the PMBus unit also cool the system being powered.

18. Reading Parametric Information

The READ commands allow the host to read various parameters of the PMBus device. These commands are read only.

18.1. READ_VIN

The READ_VIN command returns the input voltage in Volts.

The two data bytes are encoded in any format given in Section 7. The PMBus device product literature shall clearly state which format the device uses.

18.2. READ_IIN

The READ_IIN command returns the input current in Amperes.

The two data bytes are encoded in any format given in Section 7. The PMBus device product literature shall clearly state which format the device uses.

18.3. READ_VCAP

The READ_VCAP command returns voltage on the energy storage (hold-up or ride-through) capacitor in Volts.

The two data bytes are encoded in any format given in Section 7. The PMBus device product literature shall clearly state which format the device uses.

18.4. READ_VOUT

The READ_VOUT command returns the actual, measured (not commanded) output voltage in the same format as set by the VOUT_MODE command. See Section 9.1 for how the VOUT_SCALE command (Section 18.4) applies to the value returned by this command.

If the VOUT_MODE is set for ULINEAR16, IEEE-754 or Direct format, the returned value is in Volts. If the VOUT_MODE is set to VID format, then the returned value is the VID code corresponding to the voltage closest to the measured voltage.

18.5. READ_IOUT

The READ_IOUT command returns the measured output current in Amperes. See Sections 9.3 and 9.5 for information on how the IOUT_CAL_GAIN (Section 14.8) and IOUT_CAL_OFFSET (Section 14.9) apply to this command.

The two data bytes are encoded in any format given in Section 7. The PMBus device product literature shall clearly state which format the device uses.

18.6. READ_TEMPERATURE_n

Up to three temperature readings can be returned for each device. The device’s product literature shall describe the temperature being measured. For example, an ac-dc power supply might return the temperature of a critical heatsink and the temperature of the inlet cooling air.

The three commands for reading temperature are:

- READ_TEMPERATURE_1,
- READ_TEMPERATURE_2, and
- READ_TEMPERATURE_3.

Each returns the temperature in degrees Celsius. The two data bytes are encoded in any format given in Section 7. The PMBus device product literature shall clearly state which format the device uses.

18.7. READ_FAN_SPEED_n

Up to four fan speed readings can be returned for each device. The four commands for reading fan speed are:

- READ_FAN_SPEED_1,
- READ_FAN_SPEED_2,
- READ_FAN_SPEED_3, and
- READ_FAN_SPEED_4

The value returned is in RPM.

The two data bytes are encoded in any format given in Section 7. The PMBus device product literature shall clearly state which format the device uses.

18.8. Deleted

18.9. READ_DUTY_CYCLE

The READ_DUTY_CYCLE command returns the duty of the PMBus device’s main power converter in percent.

The two data bytes are encoded in any format given in Section 7. The PMBus device product literature shall clearly state which format the device uses.

18.10. READ_FREQUENCY

The READ_FREQUENCY command returns the switching frequency of the PMBus device’s main power converter in kilohertz. This command returns the actual switching frequency and not the commanded switching frequency.

The two data bytes are encoded in any format given in Section 7. The PMBus device product literature shall clearly state which format the device uses.

18.11. READ_POUT

The READ_POUT command returns the output power, in watts, of the PMBus device.

The two data bytes are encoded in any format given in Section 7. The PMBus device product literature shall clearly state which format the device uses.

18.12. READ_PIN

The READ_PIN command returns the input power, in watts, of the PMBus device.

The two data bytes are encoded in any format given in Section 7. The PMBus device product literature shall clearly state which format the device uses.

18.13. READ_EIN And READ_EOUT

The READ_EIN and READ_EOUT commands are used to return information the host can use to calculate the input power consumption of a PMBus device. The information provided by this command is independent of any device specific averaging period, sampling frequency, or calculation algorithm.

Each command returns six data byes. The first two bytes are the output of an accumulator that continuously sums samples of the instantaneous input power (the product of the samples of the input voltage and input current). The accumulator value is scaled so that the units are "watt-samples". These two data bytes are encoded in any format given in Section 7. The PMBus device product literature shall clearly state which format the device uses.

The next data byte is a ROLLOVER_COUNT for the accumulator. This byte is an unsigned integer. The ROLLOVER_COUNT will periodically roll over from its maximum positive value to zero. It is up to the host to keep track of the state of the ROLLOVER_COUNT and account for the rollovers.

The other three data bytes are a 24 bit unsigned integer that counts the number of samples of the instantaneous input power. This value will also roll over periodically from its maximum positive value to zero. It is up to the host to keep track of the sample count and account for the rollovers.

The host uses the accumulator value and rollover count to calculate the current "energy count". If the format of the accumulator is returned in LINEAR11 Format, the calculation of the energy count is as follows:

$$Energy\_Count = Rollover\_Count * Maximum\_Linear\_Format\_Value \\ + Accumulator\_Value$$

where

$$Maximum\_Linear\_Format\_Value = Y_{MAX}\, 2^{N_{MAX}} = \left(2^{10}-1\right)\left(2^{15}\right) = 33{,}521{,}664$$

If the format of the accumulator is in Direct Format, the calculation of the energy count is as follows:

$$Energy\_Count = Rollover\_Count * Maximum\_Direct\_Format\_Value(m,b,R) \\ + Accumulator\_Value$$

where the maximum Direct Format value is a function of the current values of *m*, *b*, *R*:

$$Maximum\_Direct\_Format\_Value(m,b,R) = \frac{1}{m}\cdot\left(Y_{MAX}\cdot 10^{-R} - b\right)$$

and

$$Y_{MAX} = 2^{15} - 1 = 32{,}767.$$

The host calculates the average power since the last reading using the formula:

$$Average\_Power = \frac{Current\_Energy\_Count - Last\_Energy\_Count}{Current\_Sample\_Count - Last\_Sample\_Count}.$$

Figure 33 shows an example of the READ_EIN command packet format when using Packet Error Checking (PEC)

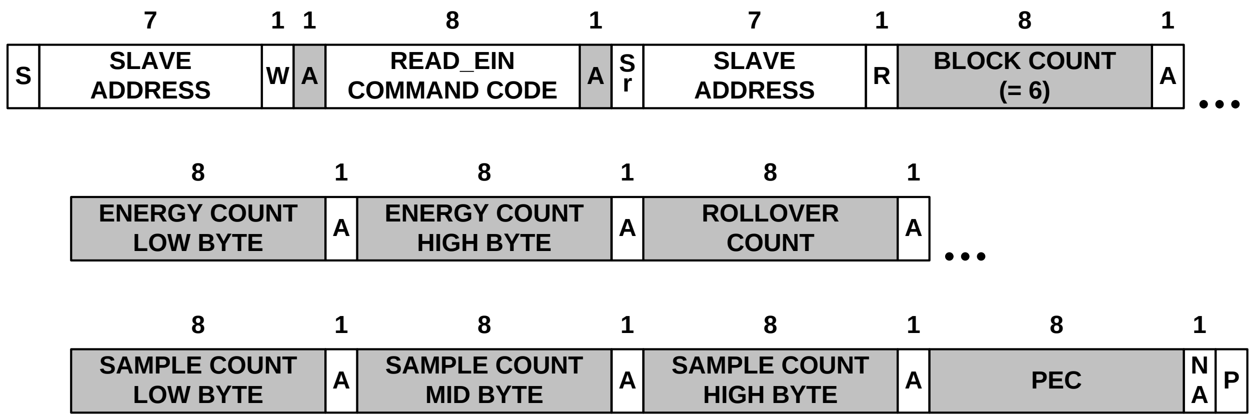


Figure 34. READ_EIN Command Packet Format

Figure 34 shows an example of the READ_EOUT command packet format when using Packet Error Checking (PEC).

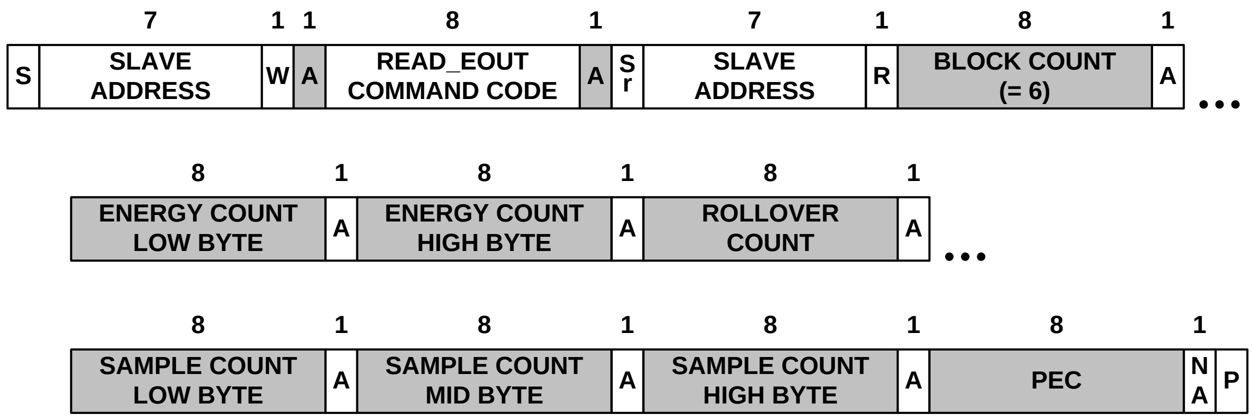


Figure 35. READ_EOUT Command Packet Format

18.14. READ_KWH_IN/READ_KWH_OUT

The READ_KWH_IN and READ_KWH_OUT commands allow the system host to read, respectively, the cumulative energy input or output of a PMBus device directly in units of energy.

A PMBus device may support the READ_KWH_IN command, the READ_KWH_OUT command, or both. In the description of these commands, READ_KWH_IN/OUT should be read as “READ_KWH_IN or READ_KWH_OUT”.

Depending on the capabilities of the PMBus device as reported by the READ_KWH_CONFIG command, the data may be returned

- In units of kW-hours, W-hours, or mW-hours and
- Either as a 32 bit unsigned integer (if floating point is not supported) or as a single precision IEEE-754 floating point number (if floating point is supported). These are the only two numeric formats supported by this command.

Reading this command uses the READ 32 protocol (SMBus specification [A03]). The READ_KWH_IN/OUT commands are not writable.

The format of the command packet format for a reading of the energy usage, with packet error checking, is shown in Figure 35. The 32 bit data is returned starting with the most significant byte and ending with the least significant byte.

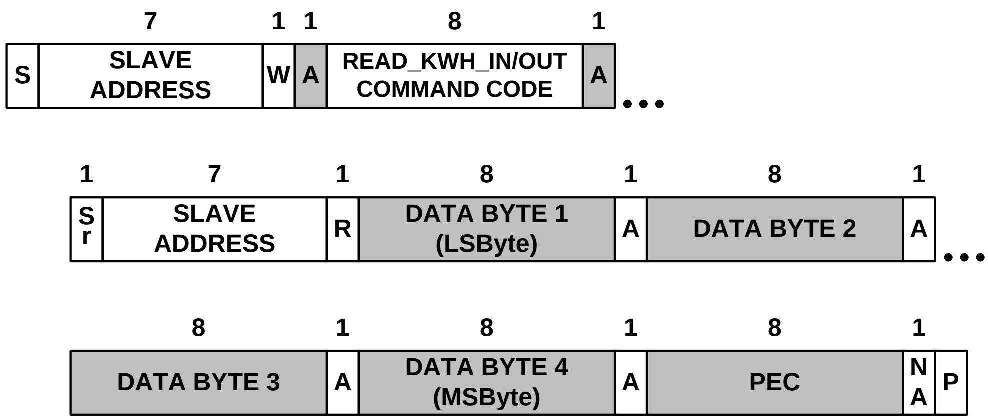


Figure 36. READ_KWH_IN/OUT Command Packet Format With Packet Error Checking

18.15. READ_KWH_CONFIG

The system host uses the READ_KWH_CONFIG command to determine the format and status of the energy accumulators used for the READ_KWH_IN and READ_KWH_OUT commands.

The contents of the READ_KWH_CONFIG data bytes are given in Table 26.

Table 26. READ_KWH_CONFIG Data Byte Format

Byte	Bits	Value	Meaning
High Byte: Input Energy Accumulator Status/Config	7	1	READ_KWH_IN is supported/an input energy accumulator is present
		0	READ_KWH_IN is not supported/an input energy accumulator is not present
	6:5	11	Reserved
		10	Input energy usage reported units are kW-hours
		01	Input energy usage reported units are W-hours
		00	Input energy usage reported units are mW-hours
	4	1	The data is returned in IEEE Single Precision format
		0	The data is returned as an unsigned 32 bit integer
	3	1	The accumulated value is volatile (reset when the input power is removed)
		0	The reported value is non-volatile (not reset when the input power is removed)
	2	1	The device supports resetting or clearing the accumulated energy value by the system host
		0	The device does not support resetting or clearing the accumulated energy value by the system host

Byte	Bits	Value	Meaning
	1	1	The reported value has rolled over since last reset of the accumulated value
		0	The reported value has not rolled over since last reset of the accumulated value
	0	1	The reported value has been reset by the system host since the last power on or rollover of the accumulated value
		0	The reported value has not been reset by the system host since the last power on or rollover of the accumulated value
Low Byte: Input Energy Accumulator Status/Config	7	1	READ_KWH_OUT is supported/an output energy accumulator is present
		0	READ_KWH_IN is not supported/an output energy accumulator is not present
	6:5	11	Reserved
		10	Output energy usage reported units are kW-hours
		01	Output energy usage reported units are W-hours
		00	Output energy usage reported units are mW-hours
	4	1	The data is returned in IEEE Single Precision format
		0	The data is returned as an unsigned 32 bit integer
	3	1	The accumulated value is volatile (reset when the input power is removed)
		0	The reported value is non-volatile (not reset when the input power is removed)
	2	1	The device supports resetting or clearing the accumulated energy value by the system host
		0	The device does not support resetting or clearing the accumulated energy value by the system host
	1	1	The reported value has rolled over since last reset of the accumulated value
		0	The reported value has not rolled over since last reset of the accumulated value
	0	1	The reported value has been reset by the system host since the last power on or rollover of the accumulated value
		0	The reported value has not been reset by the system host since the last power on or rollover of the accumulated value

If bit [7] of the high or low byte is not set (an energy accumulator is not present), the PMBus device shall return zeroes for bits [6:0] of that byte.

The values of bits [7:2] of each byte are static and set by the device manufacturer. Bits [1:0] of each byte are dynamic.

Bit [1] of each byte is not writable by the system host. The device will set bit [1] when the respective accumulated energy value overflows the maximum possible value. Note that when using the IEEE Single Precision floating point format, an overflow is unlikely. Bit [1] is cleared if:

- The energy accumulator is volatile and the input power is removed or
- The device supports a reset of the accumulator value by the system host and the system does reset the accumulator.

If the device supports resetting an energy accumulator (bit [2] = 1), then the energy accumulator is reset by writing a 1 to the bit [0] of the appropriate byte.

- If the device supports the READ_KWH_IN command and resetting the input energy accumulator, the accumulator is reset by writing a 1 to bit [0] of the high byte (data value 0100h).
- If the device supports the READ_KWH_OUT command and resetting the output energy accumulator, the accumulator is reset by writing a 1 to bit [0] of the low byte (data value 0001h).
- If the device supports the READ_KWH_IN and READ_KWH_OUT commands and resetting the both energy accumulators, the both accumulators are simultaneously reset by writing a 1 to bit [0] of each byte (data value 0101h).

If the system host attempts to reset a non-existent accumulator or writes any value other than 0100h, 0001h, or 0101h to the READ_KWH_CONFIG command shall result in an Invalid Data error as described in Section 10.9.3.

An example of the command packet to reset simultaneously the input and output energy accumulator is shown in Figure 36.

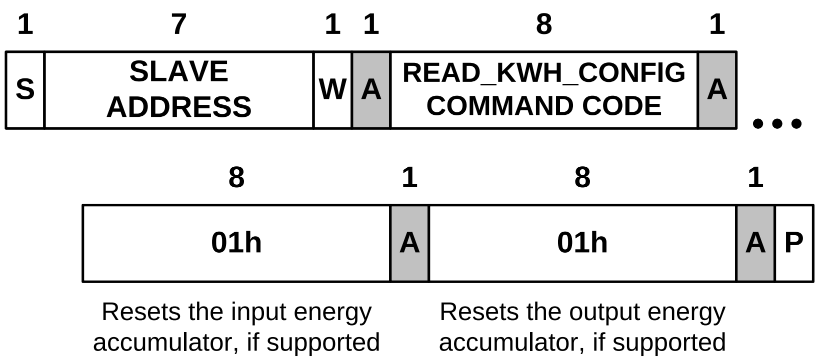


Figure 37. READ_KWH_IN/OUT Command Packet Format To Reset The Accumulator

19. Reserved

This section number is reserved for future use.

20. Reserved

This section number is reserved for future use.

21. Reserved

This section number is reserved for future use.

22. Manufacturer's Information

22.1. PMBUS_REVISION

PMBUS_REVISION command stores or reads the revision of the PMBus to which the device is compliant.

The command has one data byte. Bits [7:4] indicate the revision of PMBus specification Part I to which the device is compliant. Bits [3:0] indicate the revision of PMBus specification Part II to which the device is compliant. The permissible values are shown in Table 27.

Devices may support this as a read only command.

Table 27. PMBus Revision Data Byte Contents

Bits [7:4]	Part I Revision		Bits [3:0]	Part II Revision
0000b	1.0		0000b	1.0
0001b	1.1		0001b	1.1
0010b	1.2		0010b	1.2
0011b	1.3		0011b	1.3

22.2. Inventory Information

The PMBus protocol provides commands for the storage and retrieval of the device manufacturer's inventory information. This is more typically the manufacturer of an assembled power supply or dc-dc converter than an IC manufacturer.

The length of data for type of inventory information varies from manufacturer to manufacturer so the length of the data for each type is not specified. Instead, if a PMBus device supports manufacturer's inventory information, the device's product literature will state the total space available, in bytes, for all inventory information.

SMBus block write and block read commands (SMBus specification [A03]) are used to write and retrieve inventory information. The block write and read commands require that the first data byte be the number of bytes to follow (Byte Count). The bytes used for the byte count take up space in the available memory. For example, suppose the MFR_ID is six bytes. The manufacturer sends the number 6 (the Byte Count) plus six bytes of data, for a total of seven bytes. If the available memory was 128 bytes before the MFR_ID is loaded, then 121 bytes are available after.

Manufacturer's inventory information is always loaded using one byte text (ISO/IEC 8859-1 [A05]) characters.

The preferred practice is for characters to be delivered to the system host in the same order as if they were being read from a printed page. For example, if the MFR_ID was "SMIF", then data byte 1 of the 4 byte block transfer would be 53h (ISO/IEC 8859-1 code for "S") and data byte 4 of the block transfer would be 46h (ISO-IEC 8859-1 code for "F").

The preferred practice is for device manufacturer to clearly identify the total amount of memory available for storing inventory information.

22.2.1. MFR_ID

The MFR_ID command is used to either set or read the manufacturer's ID (name, abbreviation or symbol that identifies the unit's manufacturer). Each manufacturer chooses their identifier. MFR_ID is typically only set once, at the time of manufacture.

22.2.2. MFR_MODEL

The MFR_MODEL command is used to either set or read the manufacturer's model number. MFR_MODEL is typically set once, at the time of manufacture.

22.2.3. MFR_REVISION

The MFR_REVISION command is used to either set or read the manufacturer's revision number. Each manufacturer uses the format of their choice for the revision number. MFR_REVISION is typically set at the time of manufacture or if the device is updated to a later revision.

22.2.4. MFR_LOCATION

The MFR_REVISION command is used to either set or read the manufacturing location of the device. Each manufacturer uses the format of their choice for the location information. MFR_REVISION is typically only set once, at the time of manufacture.

22.2.5. MFR_DATE

The MFR_DATE command is used to either set or read the date the device was manufactured. While each manufacturer uses the format of their choice for the revision number, the recommended MFR_DATE format is YYMMDD where Y, M and D are integer values from 0 to 9, inclusive. MFR_DATE is typically only set once, at the time of manufacture.

22.2.6. MFR_SERIAL

The MFR_SERIAL command is used to either set or read the manufacturer's serial number of the device. Each manufacturer uses the format of their choice for the serial number. MFR_SERIAL is typically only set once, at the time of manufacture.

22.2.7. IC_DEVICE_ID

The IC_DEVICE_ID command is used to either set or read the type or part number of an IC embedded within a PMBus that is used for the PMBus interface. Each manufacturer uses the format of their choice for the IC device identification. IC_DEVICE_ID is typically only set once, at the time of manufacture.

22.2.8. IC_DEVICE_REV

The IC_DEVICE_REV command is used to either set or read the revision of the IC whose type or part number is set or read with the IC_DEVICE_ID command. Each manufacturer uses the format of their choice for the IC device revision. IC_DEVICE_REV is typically only set once, at the time of manufacture.

22.3. Manufacturer Ratings

The following commands provide the ability for manufacturers to provide summary information about the unit's ratings. This information serves as an electronic nameplate for the user's convenience.

PMBus devices are not required to report violations of any of these ratings. For any supported Manufacturer Ratings command, the product literature shall describe if and how the device responds to violations of the ratings.

Unless otherwise specified, each of the Manufacturer's Ratings commands has two data bytes in any format given in Section 7. The PMBus device's product literature shall clearly state which format the device supports.

22.3.1. MFR_VIN_MIN

The MFR_VIN_MIN command sets or retrieves the minimum rated value, in Volts, of the input voltage.

22.3.2. MFR_VIN_MAX

The MFR_VIN_MAX command sets or retrieves the maximum rated value, in Volts, of the input voltage.

22.3.3. MFR_IIN_MAX

The MFR_IIN_MIN command sets or retrieves the maximum rated value, in Amperes, of the input current.

22.3.4. MFR_PIN_MAX

The MFR_PIN_MIN command sets or retrieves the maximum rated value, in watts, of the input power.

22.3.5. MFR_VOUT_MIN

The MFR_VOUT_MIN command sets or retrieves the minimum rated value, in Volts, to which the output voltage may be set.

22.3.6. MFR_VOUT_MAX

The MFR_VOUT_MAX command sets or retrieves the maximum rated value, in Volts, to which the output voltage may be set.

22.3.7. MFR_IOUT_MAX

The MFR_IOUT_MAX command sets or retrieves the maximum rated value, in Amperes, to which the output may be loaded.

22.3.8. MFR_POUT_MAX

The MFR_POUT_MAX command sets or retrieves the maximum rated output power, in watts, that the unit is rated to supply.

22.3.9. MFR_TAMBIENT_MAX

The MFR_TAMBIENT_MAX command sets or retrieves the maximum rated ambient temperature, in degrees Celsius, in which the unit may be operated.

22.3.10. MFR_TAMBIENT_MIN

The MFR_TAMBIENT MIN command sets or retrieves the minimum rated ambient temperature, in degrees Celsius, in which the unit may be operated.

22.3.11. MFR_EFFICIENCY_LL

The MFR_EFFICIENCY_LL command sets or retrieves information about the efficiency of the device while operating at a low line condition. Not including the PEC byte, if used, and the byte count byte, there are fourteen data bytes as described below.

The efficiency is specified at one input voltage and three data points consisting of output power and the efficiency at that output power. The three power ratings are typically referred as low, medium and high output power and are transmitted in that order. For example, the low, medium and high output power might correspond to 30%, 50% and 90% of the rated output power. The exact values at which the power is specified is left to the PMBus device manufacturer.

Each value (voltage, power or efficiency) is transmitted as two bytes in the LINEAR11 format.

Table 28. Data Format Of The MFR_EFFICIENCY_LL Command

Byte Number	Byte Order.	Description
0	Low Byte	The input voltage, in Volts, at which the low line efficiency data is applicable. Note that byte 0 is the first data byte transmitted as part of the block transfer.
1	High Byte	
2	Low Byte	Power, in watts, at which the low power efficiency is specified
3	High Byte	
4	Low Byte	The efficiency, in percent, at the specified low power.
5	High Byte	
6	Low Byte	Power, in watts, at which the medium power efficiency is specified
7	High Byte	
8	Low Byte	The efficiency, in percent, at the specified medium power.
9	High Byte	
10	Low Byte	Power, in watts, at which the high power efficiency is specified
11	High Byte	
12	Low Byte	The efficiency, in percent, at the specified high power. Note that byte 13 is the last data byte transmitted as part of the block transfer.
13	High Byte	

22.3.12. MFR_EFFICIENCY_HL

The MFR_EFFICIENCY_HL command sets or retrieves information about the efficiency of the device while operating at a high line condition. Not including the PEC byte, if used, and the byte count byte, there are fourteen data bytes as described below.

The efficiency is specified at one input voltage and three data points consisting of output power and the efficiency at that output power. The three power ratings are typically referred as low, medium and high output power and are transmitted in that order. For example, the low, medium and high output power might correspond to 30%, 50% and 90% of the rated output power. The exact values at which the output power is specified is left to the PMBus device manufacturer.

Each value (voltage, power or efficiency) is transmitted as two bytes in the LINEAR11 format.

Table 29. Data Format Of The MFR_EFFICIENCY_HL Command

Byte Number	Byte Order.	Description
0	Low Byte	The input voltage, in Volts, at which the high line efficiency data is applicable. Note that byte 0 is the first data byte transmitted as part of the block transfer.
1	High Byte	
2	Low Byte	Power, in watts, at which the low power efficiency is specified
3	High Byte	
4	Low Byte	The efficiency, in percent, at the specified low power.
5	High Byte	
6	Low Byte	Power, in watts, at which the medium power efficiency is specified
7	High Byte	
8	Low Byte	The efficiency, in percent, at the specified medium power.
9	High Byte	
10	Low Byte	Power, in watts, at which the high power efficiency is specified
11	High Byte	
12	Low Byte	The efficiency, in percent, at the specified high power. Note that byte 13 is the last data byte transmitted as part of the block transfer.
13	High Byte	

22.3.13. MFR_PIN_ACCURACY

The MFR_PIN_ACCURACY command returns the accuracy, in percent, of the value returned by the READ_PIN command.

There is one data byte. The value is 0.1% per bit which gives a range of ±0.0% to ±25.5%.

The range of input voltage, output loading and operating temperature over which this accuracy applies is to be specified in the product literature.

22.3.14. APP_PROFILE_SUPPORT

The APP_PROFILE_SUPPORT command provides a mean for a host to determine which PMBus Applications Profiles, and the revision of those profiles, that the device supports.

The Block Read/Write protocol is used with this command.

Each profile is identified by two data bytes. The first data byte transmitted indicates a supported profile and the second data byte indicates the revision.

Any value not listed in Table 30 is reserved for future use.

The second data byte, indicating revision shall be formatted as two four bit nibbles. Bits [7:4] shall indicate the major revision and bits [3:0] shall indicate the minor revision. The value 00h shall be used only when the first byte is also 00h, indicating that the device does not support any application profiles. For example, revision 1.2 would be reported as 0102h.

Table 30. APP_PROFILE_SUPPORT First Data Byte Contents

First Byte	Application Profile
00h	No Application Profiles Are Supported
01h	Server AC-DC Power Supply [A06]
02h	DC-DC Converters For Microprocessor Power And Other Computer Applications [A07]
03h	DC-DC Converters For General Purpose Use [A08]

If a device supports multiple Application Profiles, the device may report these in any order.

An example of the packet created when a host issues an APP_PROFILE_SUPPORT command to a device that supports two Applications Profile and Packet Error Checking is shown in Figure 37.

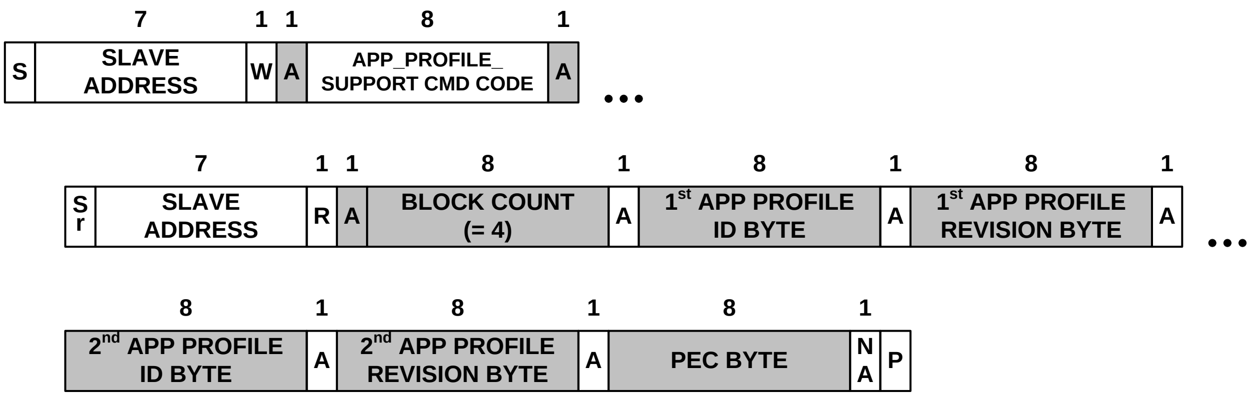


Figure 38. APP_PROFILE_SUPPORT Packet Example

22.3.15. MFR_MAX_TEMP_1, _2, _3

The MFR_MAX_TEMP_1, MFR_MAX_TEMP_2, and MFR_MAX_TEMP_3 commands set and retrieve the manufacturer's maximum rated temperature, in degrees Celsius, associated with the READ_TEMPERATURE_n commands.

The format of the returned values shall be the same as the format used for the MFR_TAMBIENT_MAX command (if that command is supported).

23. User Data And Configuration

The PMBus protocol reserves 16 commands for PMBus device manufacturers to provide memory for their customers to store information. These commands, for example, could be used to store end user specific inventory information or configuration information such as digital control loop coefficients.

The names of the commands are USER_DATA_00 through USER_DATA_15.

Each of these commands may use the block write and block read to store and retrieve up to 255 bytes of data for each command for a maximum possible User Data storage of 4,080 byes.

The PMBus device's product literature shall describe the manufacturer's implementation of these commands.

24. Manufacturer Specific Commands

The PMBus protocol reserves 46 command codes for manufacturer specific commands. These commands will be unique to a particular device or manufacturer and allow for unique or proprietary extensions to the PMBus protocol.

The names of the commands are MFR_SPECIFIC_00 through MFR_SPECIFIC_45.

The PMBus device's product literature shall describe the manufacturer's implementation of these commands.

25. Command Extensions

25.1. MFR_SPECIFIC_COMMAND_EXT

The MFR_SPECIFIC_COMMAND_EXT is used to allow PMBus device manufacturers to extend the command set beyond the available 256 command codes.

This command uses the Extended Command: Read/Write Byte or Extended Command: Read/Write Word protocols described in the PMBus specification, Part I [A01].

25.2. PMBUS_COMMAND_EXT

The PMBUS_COMMAND_EXT is reserved for future use to extend the PMBus command set beyond the available 256 command codes.

This command uses the Extended Command: Read/Write Byte or Extended Command: Read/Write Word protocols described in the PMBus specification, Part I [A01].

APPENDIX I. **Command Summary**

Any command codes not used in Table 31 are reserved for future use.

Table 31. Command Summary

Note: The Number Of Data Bytes does not include PEC bytes, if used, nor does it include the byte count byte of block transactions.

Command Code	Command Name	SMBus Transaction Type: Writing Data	SMBus Transaction Type: Reading Data	Number Of Data Bytes
00h	PAGE	Write Byte	Read Byte	1
01h	OPERATION	Write Byte	Read Byte	1
02h	ON_OFF_CONFIG	Write Byte	Read Byte	1
03h	CLEAR_FAULTS	Send Byte	N/A	0
04h	PHASE	Write Byte	Read Byte	1
05h	PAGE_PLUS_WRITE	Block Write	N/A	Variable
06h	PAGE_PLUS_READ	N/A	Block Write – Block Read Process Call	Variable
07h	ZONE_CONFIG	Write Word	Read Word	2
08h	ZONE_ACTIVE	Write Word	Read Word	2
09h	Reserved			
0Ah	Reserved			
0Bh	Reserved			
0Ch	Reserved			
0Dh	Reserved			
0Eh	Reserved			
0Fh	Reserved			
10h	WRITE_PROTECT	Write Byte	Read Byte	1
11h	STORE_DEFAULT_ALL	Send Byte	N/A	0
12h	RESTORE_DEFAULT_ALL	Send Byte	N/A	0
13h	STORE_DEFAULT_CODE	Write Byte	N/A	1
14h	RESTORE_DEFAULT_CODE	Write Byte	N/A	1
15h	STORE_USER_ALL	Send Byte	N/A	0
16h	RESTORE_USER_ALL	Send Byte	N/A	0
17h	STORE_USER_CODE	Write Byte	N/A	1
18h	RESTORE_USER_CODE	Write Byte	N/A	1
19h	CAPABILITY	N/A	Read Byte	1

Command Code	Command Name	SMBus Transaction Type: Writing Data	SMBus Transaction Type: Reading Data	Number Of Data Bytes
1Ah	QUERY	N/A	Block Write-Block Read Process Call	1
1Bh	SMBALERT_MASK	Write Word	Block Write-Block Read Process Call	2
1Ch	Reserved			
1Dh	Reserved			
1Eh	Reserved			
1Fh	Reserved			
20h	VOUT_MODE	Write Byte	Read Byte	1
21h	VOUT_COMMAND	Write Word	Read Word	2
22h	VOUT_TRIM	Write Word	Read Word	2
23h	VOUT_CAL_OFFSET	Write Word	Read Word	2
24h	VOUT_MAX	Write Word	Read Word	2
25h	VOUT_MARGIN_HIGH	Write Word	Read Word	2
26h	VOUT_MARGIN_LOW	Write Word	Read Word	2
27h	VOUT_TRANSITION_RATE	Write Word	Read Word	2
28h	VOUT_DROOP	Write Word	Read Word	2
29h	VOUT_SCALE_LOOP	Write Word	Read Word	2
2Ah	VOUT_SCALE_MONITOR	Write Word	Read Word	2
2Bh	VOUT_MIN	Write Word	Read Word	2
2Ch	Reserved			
2Dh	Reserved			
2Eh	Reserved			
2Fh	Reserved			
30h	COEFFICIENTS	N/A	Block Write-Block Read Process Call	5
31h	POUT_MAX	Write Word	Read Word	2
32h	MAX_DUTY	Write Word	Read Word	2
33h	FREQUENCY_SWITCH	Write Word	Read Word	2
34h	POWER_MODE	Write Byte	Read Byte	1
35h	VIN_ON	Write Word	Read Word	2

Command Code	Command Name	SMBus Transaction Type: Writing Data	SMBus Transaction Type: Reading Data	Number Of Data Bytes
36h	VIN_OFF	Write Word	Read Word	2
37h	INTERLEAVE	Write Word	Read Word	2
38h	IOUT_CAL_GAIN	Write Word	Read Word	2
39h	IOUT_CAL_OFFSET	Write Word	Read Word	2
3Ah	FAN_CONFIG_1_2	Write Byte	Read Byte	1
3Bh	FAN_COMMAND_1	Write Word	Read Word	2
3Ch	FAN_COMMAND_2	Write Word	Read Word	2
3Dh	FAN_CONFIG_3_4	Write Byte	Read Byte	1
3Eh	FAN_COMMAND_3	Write Word	Read Word	2
3Fh	FAN_COMMAND_4	Write Word	Read Word	2
40h	VOUT_OV_FAULT_LIMIT	Write Word	Read Word	2
41h	VOUT_OV_FAULT_RESPONSE	Write Byte	Read Byte	1
42h	VOUT_OV_WARN_LIMIT	Write Word	Read Word	2
43h	VOUT_UV_WARN_LIMIT	Write Word	Read Word	2
44h	VOUT_UV_FAULT_LIMIT	Write Word	Read Word	2
45h	VOUT_UV_FAULT_RESPONSE	Write Byte	Read Byte	1
46h	IOUT_OC_FAULT_LIMIT	Write Word	Read Word	2
47h	IOUT_OC_FAULT_RESPONSE	Write Byte	Read Byte	1
48h	IOUT_OC_LV_FAULT_LIMIT	Write Word	Read Word	2
49h	IOUT_OC_LV_FAULT_RESPONSE	Write Byte	Read Byte	1
4Ah	IOUT_OC_WARN_LIMIT	Write Word	Read Word	2
4Bh	IOUT_UC_FAULT_LIMIT	Write Word	Read Word	2
4Ch	IOUT_UC_FAULT_RESPONSE	Write Byte	Read Byte	1
4Dh	Reserved			
4Eh	Reserved			
4Fh	OT_FAULT_LIMIT	Write Word	Read Word	2
50h	OT_FAULT_RESPONSE	Write Byte	Read Byte	1
51h	OT_WARN_LIMIT	Write Word	Read Word	2
52h	UT_WARN_LIMIT	Write Word	Read Word	2
53h	UT_FAULT_LIMIT	Write Word	Read Word	2
54h	UT_FAULT_RESPONSE	Write Byte	Read Byte	1
55h	VIN_OV_FAULT_LIMIT	Write Word	Read Word	2
56h	VIN_OV_FAULT_RESPONSE	Write Byte	Read Byte	1

Command Code	Command Name	SMBus Transaction Type: Writing Data	SMBus Transaction Type: Reading Data	Number Of Data Bytes
57h	VIN_OV_WARN_LIMIT	Write Word	Read Word	2
58h	VIN_UV_WARN_LIMIT	Write Word	Read Word	2
59h	VIN_UV_FAULT_LIMIT	Write Word	Read Word	2
5Ah	VIN_UV_FAULT_RESPONSE	Write Byte	Read Byte	1
5Bh	IIN_OC_FAULT_LIMIT	Write Word	Read Word	2
5Ch	IIN_OC_FAULT_RESPONSE	Write Byte	Read Byte	1
5Dh	IIN_OC_WARN_LIMIT	Write Word	Read Word	2
5Eh	POWER_GOOD_ON	Write Word	Read Word	2
5Fh	POWER_GOOD_OFF	Write Word	Read Word	2
60h	TON_DELAY	Write Word	Read Word	2
61h	TON_RISE	Write Word	Read Word	2
62h	TON_MAX_FAULT_LIMIT	Write Word	Read Word	2
63h	TON_MAX_FAULT_RESPONSE	Write Byte	Read Byte	1
64h	TOFF_DELAY	Write Word	Read Word	2
65h	TOFF_FALL	Write Word	Read Word	2
66h	TOFF_MAX_WARN_LIMIT	Write Word	Read Word	2
67h	Reserved (Was Used In Revision 1.0)			
68h	POUT_OP_FAULT_LIMIT	Write Word	Read Word	2
69h	POUT_OP_FAULT_RESPONSE	Write Byte	Read Byte	1
6Ah	POUT_OP_WARN_LIMIT	Write Word	Read Word	2
6Bh	PIN_OP_WARN_LIMIT	Write Word	Read Word	2
6Ch	Reserved			
6Dh	Reserved			
6Eh	Reserved			
6Fh	Reserved			
70h	Reserved (Test Input Fuse A)			
71h	Reserved (Test Input Fuse B)			
72h	Reserved (Test Input OR-ing A)			
73h	Reserved (Test Input OR-ing B)			
74h	Reserved (Test Output OR-ing)			
75h	Reserved			
76h	Reserved			
77h	Reserved			

Command Code	Command Name	SMBus Transaction Type: Writing Data	SMBus Transaction Type: Reading Data	Number Of Data Bytes
78h	STATUS_BYTE	Write Byte	Read Byte	1
79h	STATUS_WORD	Write Word	Read Word	2
7Ah	STATUS_VOUT	Write Byte	Read Byte	1
7Bh	STATUS_IOUT	Write Byte	Read Byte	1
7Ch	STATUS_INPUT	Write Byte	Read Byte	1
7Dh	STATUS_TEMPERATURE	Write Byte	Read Byte	1
7Eh	STATUS_CML	Write Byte	Read Byte	1
7Fh	STATUS_OTHER	Write Byte	Read Byte	1
80h	STATUS_MFR_SPECIFIC	Write Byte	Read Byte	1
81h	STATUS_FANS_1_2	Write Byte	Read Byte	1
82h	STATUS_FANS_3_4	Write Byte	Read Byte	1
83h	READ_KWH_IN	N/A	Read 32	4
84h	READ_KWH_OUT	N/A	Read 32	4
85h	READ_KWH_CONFIG	Write Word	Read Word	2
86h	READ_EIN	N/A	Block Read	5
87h	READ_EOUT	N/A	Block Read	5
88h	READ_VIN	N/A	Read Word	2
89h	READ_IIN	N/A	Read Word	2
8Ah	READ_VCAP	N/A	Read Word	2
8Bh	READ_VOUT	N/A	Read Word	2
8Ch	READ_IOUT	N/A	Read Word	2
8Dh	READ_TEMPERATURE_1	N/A	Read Word	2
8Eh	READ_TEMPERATURE_2	N/A	Read Word	2
8Fh	READ_TEMPERATURE_3	N/A	Read Word	2
90h	READ_FAN_SPEED_1	N/A	Read Word	2
91h	READ_FAN_SPEED_2	N/A	Read Word	2
92h	READ_FAN_SPEED_3	N/A	Read Word	2
93h	READ_FAN_SPEED_4	N/A	Read Word	2
94h	READ_DUTY_CYCLE	N/A	Read Word	2
95h	READ_FREQUENCY	N/A	Read Word	2
96h	READ_POUT	N/A	Read Word	2
97h	READ_PIN	N/A	Read Word	2
98h	PMBUS_REVISION	N/A	Read Byte	1

Command Code	Command Name	SMBus Transaction Type: Writing Data	SMBus Transaction Type: Reading Data	Number Of Data Bytes
99h	MFR_ID	Block Write	Block Read	Variable
9Ah	MFR_MODEL	Block Write	Block Read	Variable
9Bh	MFR_REVISION	Block Write	Block Read	Variable
9Ch	MFR_LOCATION	Block Write	Block Read	Variable
9Dh	MFR_DATE	Block Write	Block Read	Variable
9Eh	MFR_SERIAL	Block Write	Block Read	Variable
9Fh	APP_PROFILE_SUPPORT	N/A	Block Read	Variable
A0h	MFR_VIN_MIN	N/A	Read Word	2
A1h	MFR_VIN_MAX	N/A	Read Word	2
A2h	MFR_IIN_MAX	N/A	Read Word	2
A3h	MFR_PIN_MAX	N/A	Read Word	2
A4h	MFR_VOUT_MIN	N/A	Read Word	2
A5h	MFR_VOUT_MAX	N/A	Read Word	2
A6h	MFR_IOUT_MAX	N/A	Read Word	2
A7h	MFR_POUT_MAX	N/A	Read Word	2
A8h	MFR_TAMBIENT_MAX	N/A	Read Word	2
A9h	MFR_TAMBIENT_MIN	N/A	Read Word	2
AAh	MFR_EFFICIENCY_LL	N/A	Block Read	14
ABh	MFR_EFFICIENCY_HL	N/A	Block Read	14
ACh	MFR_PIN_ACCURACY	N/A	Read Byte	1
ADh	IC_DEVICE_ID	N/A	Block Read	Variable
AEh	IC_DEVICE_REV	N/A	Block Read	Variable
AFh	Reserved			
B0h	USER_DATA_00	Block Write	Block Read	Variable
B1h	USER_DATA_01	Block Write	Block Read	Variable
B2h	USER_DATA_02	Block Write	Block Read	Variable
B3h	USER_DATA_03	Block Write	Block Read	Variable
B4h	USER_DATA_04	Block Write	Block Read	Variable
B5h	USER_DATA_05	Block Write	Block Read	Variable
B6h	USER_DATA_06	Block Write	Block Read	Variable
B7h	USER_DATA_07	Block Write	Block Read	Variable
B8h	USER_DATA_08	Block Write	Block Read	Variable
B9h	USER_DATA_09	Block Write	Block Read	Variable

Command Code	Command Name	SMBus Transaction Type: Writing Data	SMBus Transaction Type: Reading Data	Number Of Data Bytes
BAh	USER_DATA_10	Block Write	Block Read	Variable
BBh	USER_DATA_11	Block Write	Block Read	Variable
BCh	USER_DATA_12	Block Write	Block Read	Variable
BDh	USER_DATA_13	Block Write	Block Read	Variable
BEh	USER_DATA_14	Block Write	Block Read	Variable
BFh	USER_DATA_15	Block Write	Block Read	Variable
C0h	MFR_MAX_TEMP_1	Write Word	Read Word	2
C1h	MFR_MAX_TEMP_2	Write Word	Read Word	2
C2h	MFR_MAX_TEMP_3	Write Word	Read Word	2
C3h	Reserved			
C4h	MFR_SPECIFIC_C4	Mfr. Defined	Mfr. Defined	Mfr. Defined
C5h	MFR_SPECIFIC_C5	Mfr. Defined	Mfr. Defined	Mfr. Defined
C6h	MFR_SPECIFIC_C6	Mfr. Defined	Mfr. Defined	Mfr. Defined
C7h	MFR_SPECIFIC_C7	Mfr. Defined	Mfr. Defined	Mfr. Defined
C8h	MFR_SPECIFIC_C8	Mfr. Defined	Mfr. Defined	Mfr. Defined
C9h	MFR_SPECIFIC_C9	Mfr. Defined	Mfr. Defined	Mfr. Defined
CAh	MFR_SPECIFIC_CA	Mfr. Defined	Mfr. Defined	Mfr. Defined
CBh	MFR_SPECIFIC_CB	Mfr. Defined	Mfr. Defined	Mfr. Defined
CCh	MFR_SPECIFIC_CC	Mfr. Defined	Mfr. Defined	Mfr. Defined
CDh	MFR_SPECIFIC_CD	Mfr. Defined	Mfr. Defined	Mfr. Defined
CEh	MFR_SPECIFIC_CE	Mfr. Defined	Mfr. Defined	Mfr. Defined
CFh	MFR_SPECIFIC_CF	Mfr. Defined	Mfr. Defined	Mfr. Defined
D0h	MFR_SPECIFIC_D0	Mfr. Defined	Mfr. Defined	Mfr. Defined
D1h	MFR_SPECIFIC_D1	Mfr. Defined	Mfr. Defined	Mfr. Defined
D2h	MFR_SPECIFIC_D2	Mfr. Defined	Mfr. Defined	Mfr. Defined
D3h	MFR_SPECIFIC_D3	Mfr. Defined	Mfr. Defined	Mfr. Defined
D4h	MFR_SPECIFIC_D4	Mfr. Defined	Mfr. Defined	Mfr. Defined
D5h	MFR_SPECIFIC_D5	Mfr. Defined	Mfr. Defined	Mfr. Defined
D6h	MFR_SPECIFIC_D6	Mfr. Defined	Mfr. Defined	Mfr. Defined
D7h	MFR_SPECIFIC_D7	Mfr. Defined	Mfr. Defined	Mfr. Defined
D8h	MFR_SPECIFIC_D8	Mfr. Defined	Mfr. Defined	Mfr. Defined
D9h	MFR_SPECIFIC_D9	Mfr. Defined	Mfr. Defined	Mfr. Defined
DAh	MFR_SPECIFIC_DA	Mfr. Defined	Mfr. Defined	Mfr. Defined

Command Code	Command Name	SMBus Transaction Type: Writing Data	SMBus Transaction Type: Reading Data	Number Of Data Bytes
DBh	MFR_SPECIFIC_DB	Mfr. Defined	Mfr. Defined	Mfr. Defined
DCh	MFR_SPECIFIC_DC	Mfr. Defined	Mfr. Defined	Mfr. Defined
DDh	MFR_SPECIFIC_DD	Mfr. Defined	Mfr. Defined	Mfr. Defined
DEh	MFR_SPECIFIC_DE	Mfr. Defined	Mfr. Defined	Mfr. Defined
DFh	MFR_SPECIFIC_DF	Mfr. Defined	Mfr. Defined	Mfr. Defined
E0h	MFR_SPECIFIC_E0	Mfr. Defined	Mfr. Defined	Mfr. Defined
E1h	MFR_SPECIFIC_E1	Mfr. Defined	Mfr. Defined	Mfr. Defined
E2h	MFR_SPECIFIC_E2	Mfr. Defined	Mfr. Defined	Mfr. Defined
E3h	MFR_SPECIFIC_E3	Mfr. Defined	Mfr. Defined	Mfr. Defined
E4h	MFR_SPECIFIC_E4	Mfr. Defined	Mfr. Defined	Mfr. Defined
E5h	MFR_SPECIFIC_E5	Mfr. Defined	Mfr. Defined	Mfr. Defined
E6h	MFR_SPECIFIC_E6	Mfr. Defined	Mfr. Defined	Mfr. Defined
E7h	MFR_SPECIFIC_E7	Mfr. Defined	Mfr. Defined	Mfr. Defined
E8h	MFR_SPECIFIC_E8	Mfr. Defined	Mfr. Defined	Mfr. Defined
E9h	MFR_SPECIFIC_E9	Mfr. Defined	Mfr. Defined	Mfr. Defined
EAh	MFR_SPECIFIC_EA	Mfr. Defined	Mfr. Defined	Mfr. Defined
EBh	MFR_SPECIFIC_EB	Mfr. Defined	Mfr. Defined	Mfr. Defined
ECh	MFR_SPECIFIC_EC	Mfr. Defined	Mfr. Defined	Mfr. Defined
EDh	MFR_SPECIFIC_ED	Mfr. Defined	Mfr. Defined	Mfr. Defined
EEh	MFR_SPECIFIC_EE	Mfr. Defined	Mfr. Defined	Mfr. Defined
EFh	MFR_SPECIFIC_EF	Mfr. Defined	Mfr. Defined	Mfr. Defined
F0h	MFR_SPECIFIC_F0	Mfr. Defined	Mfr. Defined	Mfr. Defined
F1h	MFR_SPECIFIC_F1	Mfr. Defined	Mfr. Defined	Mfr. Defined
F2h	MFR_SPECIFIC_F2	Mfr. Defined	Mfr. Defined	Mfr. Defined
F3h	MFR_SPECIFIC_F3	Mfr. Defined	Mfr. Defined	Mfr. Defined
F4h	MFR_SPECIFIC_F4	Mfr. Defined	Mfr. Defined	Mfr. Defined
F5h	MFR_SPECIFIC_F5	Mfr. Defined	Mfr. Defined	Mfr. Defined
F6h	MFR_SPECIFIC_F6	Mfr. Defined	Mfr. Defined	Mfr. Defined
F7h	MFR_SPECIFIC_F7	Mfr. Defined	Mfr. Defined	Mfr. Defined
F8h	MFR_SPECIFIC_F8	Mfr. Defined	Mfr. Defined	Mfr. Defined
F9h	MFR_SPECIFIC_F9	Mfr. Defined	Mfr. Defined	Mfr. Defined
FAh	MFR_SPECIFIC_FA	Mfr. Defined	Mfr. Defined	Mfr. Defined
FBh	MFR_SPECIFIC_FB	Mfr. Defined	Mfr. Defined	Mfr. Defined

Command Code	Command Name	SMBus Transaction Type: Writing Data	SMBus Transaction Type: Reading Data	Number Of Data Bytes
FCh	MFR_SPECIFIC_FC	Mfr. Defined	Mfr. Defined	Mfr. Defined
FDh	MFR_SPECIFIC_FD	Mfr. Defined	Mfr. Defined	Mfr. Defined
FEh	MFR_SPECIFIC_COMMAND EXT	Extended Command	Extended Command	Mfr Defined
FFh	PMBUS_COMMAND_EXT	Extended Command	Extended Command	Mfr Defined

APPENDIX II. **Summary Of Changes**

DISCLAIMER: The section is provided for reference only and for the convenience of the reader. No suggestion, statement or guarantee is made that the description of the changes listed below is sufficient to design a device compliant with this document.

A summary of the changes made in Part II of the PMBus specification from Revision 1.3X to this revision, 1.3.1, is given below. This is not an exact list of every change made between the two documents; rather, it is a summary of the changes deemed significant by the editor.

- The 11 bit and 16 bit linear data formats are now specifically identified as LINEAR11, ULINEAR16, and SLINEAR16 throughout the document. This was done to highlight the differences in the three formats and when each is used.
- The ZONE Operation Commands section (Section 11.16) has been significantly rewritten.
- Minor clarifications were made to the description of the CAPABILITY command (Section 11.12).
- Minor clarifications were made to the description of the QUERY command (Section 11.13).
- Previously reserved command codes C4h though CFh are now assigned as Manufacturer Specific Command codes.
- The names of Manufacturer Specific Command codes was changed from numbering based (MFR_SPECIFIC_00, MFR_SPECIFIC_01, …) to an command code based naming (MFR_SPECIFIC_C4, MFR_SPECIFIC_C5, … , MFR_SPECIFIC_FD).