



# PMBus™

## Power System Management Protocol Specification

### Part I – General Requirements, Transport And Electrical Interface

Revision 1.3.1

13 March 2015

[www.powerSIG.org](http://www.powerSIG.org)

© 2015 System Management Interface Forum, Inc. – All Rights Reserved

**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**

<b>REV</b>	<b>DATE</b>	<b>DESCRIPTION</b>	<b>EDITED BY</b>
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 .....	5
1.1. Specification Scope .....	5
1.1.1. Specification Structure .....	5
1.1.2. What Is Included .....	5
1.1.3. What Is Not Included In The PMBus Specification .....	5
1.2. Specification Changes Since The Last Revision .....	5
1.3. Where To Send Feedback And Comments .....	5
2. Related Documents .....	5
2.1. Scope .....	5
2.2. Applicable Documents .....	5
2.3. Reference Documents .....	6
3. Reference Information .....	6
3.1. Signal and Parameter Names .....	6
3.2. Numerical Formats .....	6
3.2.1. Decimal Numbers .....	6
3.2.2. Floating Point Numbers .....	6
3.2.3. Binary Numbers .....	6
3.2.4. Hexadecimal Numbers .....	6
3.2.5. Examples .....	7
3.3. Byte And Bit Order .....	7
3.4. Bit And Byte Illustrations .....	7
3.5. Abbreviations, Acronyms And Definitions .....	9
4. General Requirements .....	11
4.1. Compliance .....	11
4.2. Unassisted Start Up And Operation .....	12
4.3. High Impedance When Unpowered .....	12
4.4. PMBus And AVSBus Supply Voltages .....	12
5. Transport .....	12
5.1. SMBus, Version 3.0 .....	12
5.2. Bus Speed .....	12
5.3. Electrical Drive Levels .....	12
5.4. SMBus Protocols .....	12
5.5. SMBus Features, Functions, And Protocols That Are Optional In PMBus .....	12
5.5.1. Packet Error Checking .....	12
5.5.2. SMBus Address Resolution Protocol (ARP) .....	13
5.6. Extensions To The SMBus Specification .....	13
5.6.1. Group Command Protocol .....	13
5.6.2. Extended Command Protocol .....	14
5.6.3. ZONE_READ And ZONE_WRITE Protocols .....	16
5.7. Exceptions To The SMBus Specification .....	24
6. Addressing .....	24
7. Communication From PMBus Devices To The Host .....	25
7.1. Communicating Over The Bus .....	25

7.2.	Communicating With An Interrupt Signal .....	25
8.	Hardwired Signals.....	25
8.1.	Electrical Interface .....	25
8.2.	Timing .....	25
8.3.	Control Signal (CONTROL) .....	25
8.4.	Write Protect (WP).....	25
8.5.	Other Pins.....	26
9.	Accuracy .....	26
10.	Firmware Updates .....	26
APPENDIX I.	Summary Of Changes .....	27

### **Table Of Figures**

Figure 1.	Bit Order Within A Byte .....	7
Figure 2.	Group Command Protocol Without PEC .....	14
Figure 3.	Group Command Protocol With PEC .....	14
Figure 4.	Extended Command Read Byte Protocol .....	15
Figure 5.	Extended Command Read Byte With PEC .....	15
Figure 6.	Extended Command Write Byte Protocol .....	15
Figure 7.	Extended Command Write Byte Protocol With PEC .....	15
Figure 8.	Extended Command Read Word Protocol .....	16
Figure 9.	Extended Command Read Word Protocol With PEC.....	16
Figure 10.	Extended Command Write Word Protocol.....	16
Figure 11.	Extended Command Write Word Protocol With PEC .....	16
Figure 12.	ZONE_READ With STATUS DATA Response .....	20
Figure 13.	ZONE_READ With PMBus Command .....	20
Figure 14.	ZONE_READ Returning Status Information with PEC .....	20
Figure 15.	ZONE_READ With PMBus Command Returning Data And PEC .....	21
Figure 16.	Command Control Code .....	21
Figure 17.	ZONE_WRITE Operation With PMBus Command And Two Data Bytes.....	24

### **Table Of Tables**

Table 1.	Bit And Byte Symbols Used In This Specification.....	7
Table 2.	Summary Of The ST (Status), DI (Data Inversion), And DS (Data Significance) Bit Meanings ...	22

## 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](http://www.powerSIG.org).

### 1.1. Specification Scope

#### 1.1.1. Specification Structure

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

Part II defines the command language used with the PMBus.

Part III provides the specification for the AVSBus interface.

#### 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 I.

### 1.3. Where To Send Feedback And Comments

Please send all comments by email to: [techquestions@smiforum.org](mailto: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 device 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 II, Command Language*, System Management Interface Forum, Revision 1.3.1, March 2015
- [A02] *PMBus Power System Management Protocol, Part III, AVSBus*, System Management Interface Forum, Revision 1.3.1, March 2015
- [A03] *System Management Bus (SMBus) Specification*, System Management Interface Forum, Version 3.0, 21 December 2014
- [A04] *I<sup>2</sup>C-bus specification and user manual*, Revision 6, NXP Semiconductors, 4 April 2014
- [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.

### 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] 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, commands 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. Byte And Bit Order



As specified in the SMBus specification, Version 3.0 [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.

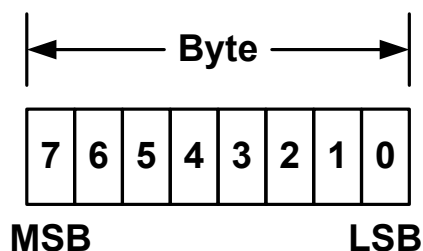




Figure 1. Bit Order Within A Byte

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
<div style="display: flex; justify-content: space-around; align-items: center;"> <div style="text-align: center;">1 </div> <div style="text-align: center;">1 </div> </div>	A unshaded vertical rectangle indicates a single bit sent from the host (bus master) to a slave
<div style="display: flex; justify-content: space-around; align-items: center;"> <div style="text-align: center;">1 </div> <div style="text-align: center;">1 </div> </div>	A shaded vertical rectangle with a shaded interior indicates a bit sent from a slave device to the bus master.
<div style="text-align: center;">8 </div> <div style="text-align: center;">8 <div style="border: 1px solid black; padding: 2px; display: inline-block;">DATA NAME DATA VALUE</div></div>	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 value is written below the data or bit field name.

Symbol	Meaning
<div style="text-align: center;">8    8  </div>	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.
<div style="border: 1px solid black; padding: 2px; width: 30px; margin: 0 auto;">S</div>	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.
<div style="border: 1px solid black; padding: 2px; width: 30px; margin: 0 auto;">S r</div>	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.
<div style="text-align: center;">1  <div style="border: 1px solid black; padding: 2px; width: 30px; margin: 0 auto;">A</div> </div>	An Acknowledge (ACK) condition send from the host
<div style="text-align: center;">1  <div style="border: 1px solid black; padding: 2px; width: 30px; margin: 0 auto;">N A</div> </div>	A Not Acknowledge (NACK) condition sent from the host
<div style="text-align: center;">1  <div style="border: 1px solid black; padding: 2px; width: 30px; margin: 0 auto;">A</div> </div>	An Acknowledge (ACK) condition sent from a slave device
<div style="text-align: center;">1  <div style="border: 1px solid black; padding: 2px; width: 30px; margin: 0 auto;">N A</div> </div>	A Not Acknowledge (NACK) condition sent from a slave device
<div style="border: 1px solid black; padding: 2px; width: 30px; margin: 0 auto;">P</div>	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.
<div style="text-align: center;">7  <div style="border: 1px solid black; padding: 2px; width: 100px; margin: 0 auto;">SLAVE ADDRESS</div> </div>	The first seven bits of the address byte, generally corresponding to the physical address of the device.
<div style="text-align: center;">1  <div style="border: 1px solid black; padding: 2px; width: 30px; margin: 0 auto;">R</div> </div>	The bit [0] of the address byte with a value of 1, indicating the device is being addressed with a read.



Symbol	Meaning
<div style="text-align: center;">1  <div style="border: 1px solid black; padding: 2px; display: inline-block;">W</div> </div>	The bit [0] of the address byte with a value of 0, indicating the device is being addressed with a write.
<div style="text-align: center;">7  <div style="border: 1px solid black; padding: 2px; display: inline-block;">BROADCAST ADDRESS</div> </div>	The SMBus broadcast address to which all devices must respond. The value is 0000000b. This is always used only with the bit [0] equal to 0 (write).

### 3.5. Abbreviations, Acronyms And Definitions

Term	Definition
ACK	ACKnowledge. 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.
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.

<b>Term</b>	<b>Definition</b>
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.

<b>Term</b>	<b>Definition</b>
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. General Requirements**

### **4.1. Compliance**

The PMBus protocol is intended to cover a wide range of power system architectures and converters.

Not all PMBus devices must support all of the available features, functions and commands.

To be compliant to the PMBus specification, a device must:

- Meet all of the requirements of Part I of the PMBus specification (this document),
- Support at least one of the non-manufacturer specific commands given in Part II of the PMBus specification [A01],
- If a device accepts a PMBus command code, it must execute that function as described in Part II of the PMBus specification [A01],
- If a device does not accept a given PMBus command code, it must respond as described in the Fault Management And Reporting section of Part II of the PMBus specification [A01].

### **4.2. Unassisted Start Up And Operation**

PMBus devices, upon application of power, must start up and begin operation in a controlled manner, as programmed internally or externally, without requiring communication from the serial bus.

### **4.3. High Impedance When Unpowered**

As described in of the SMBus specification, [A03], the electrical signals of a PMBus device must present a high impedance to the bus when the device is unpowered, during startup until fully powered, and during shutdown once the device can no longer assure the proper signal levels.

### **4.4. PMBus And AVSBus Supply Voltages**

The VDD supplies for the PMBus interface and AVSBus interface are independent of each other. It is the user's responsibility to assure that all system and device power sequencing requirements to assure proper operation are met.

## **5. Transport**

### **5.1. SMBus, Version 3.0**

PMBus devices must use the System Management Bus (SMBus), Version 3.0 [A03], for transport with the extensions and exceptions listed below.

### **5.2. Bus Speed**

All PMBus devices must support operation at 100 kHz as described in the SMBus specification [A03]. Support for operation at maximum bus speeds of 400 kHz and 1 MHz, as described in the SMBus specification [A03] is optional for PMBus devices.

### **5.3. Electrical Drive Levels**

A PMBus device shall comply with the corresponding electrical drive levels as given in the SMBus specification [A03] for the maximum bus speed supported by the device.

### **5.4. SMBus Protocols**

PMBus devices must use the SMBus transaction protocol associated with each PMBus command as described in Part II of the PMBus specification [A01].

### **5.5. SMBus Features, Functions, And Protocols That Are Optional In PMBus**

#### **5.5.1. Packet Error Checking**

Support for the SMBus Packet Error Checking (PEC) protocol is optional.

### 5.5.2. SMBus Address Resolution Protocol (ARP)

Support for the SMBus Address Resolution Protocol (ARP) is optional for PMBus devices.

## 5.6. Extensions To The SMBus Specification

### 5.6.1. Group Command Protocol

PMBus devices must support the Group Command Protocol. The Group Command Protocol is used to send commands to more than one PMBus device. The commands are sent in one continuous transmission. When the devices detect the STOP condition that ends the sending of commands, they all begin executing the command they received.

It is not a requirement that all devices receive the same command.

No more than one command can be sent to any one device in one Group Command packet.

The Group Command Protocol must not be used with commands that require the receiving device to respond with data, such as the STATUS\_BYTE command (PMBus specification, Part II [A01]).

Example: The Group Command Protocol could be used to signal all of the devices on the PMBus to change their margin state at one time. All of the devices on the bus might be instructed to set their output voltage to the VOUT\_MARGIN\_HIGH with which they have been programmed. Equally valid would be a Group Command Protocol transmission that instructed one device on the bus to set the output voltage to its programmed VOUT\_MARGIN\_LOW while all of the other devices were instructed to set their output voltage to the programmed VOUT\_MARGIN\_HIGH values.

As shown below in Figure 2 and Figure 3, the Group Command Protocol uses REPEATED START conditions to separate commands for each device. The Group Command Protocol begins with the START condition, followed by the seven bit address of the first device to receive a command and then by the write bit (0). The slave device ACKs and the master or host sends a command with the associated data byte or bytes.

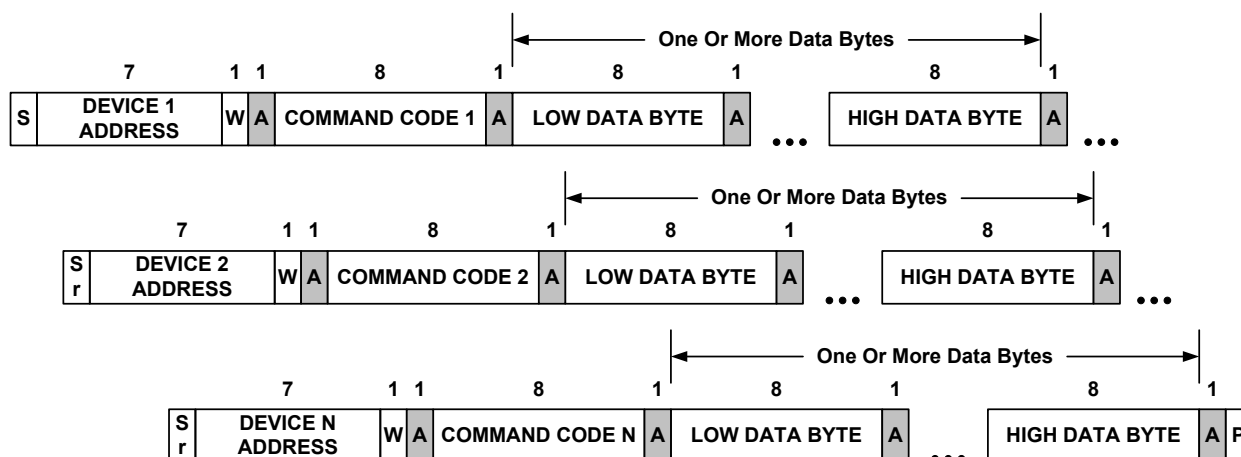
After the last data byte is sent to the first device, the host or master does NOT send a STOP condition. Instead, it sends a REPEATED START condition, followed by the seven bit address of the second device to receive a command, a write bit and the command code and the associated data bytes.

If, and only if, this is the last device to receive a command, the host or master sends a STOP condition. Otherwise, the host or master sends a REPEATED START condition and starts transmitting the address of the third device to receive a command.

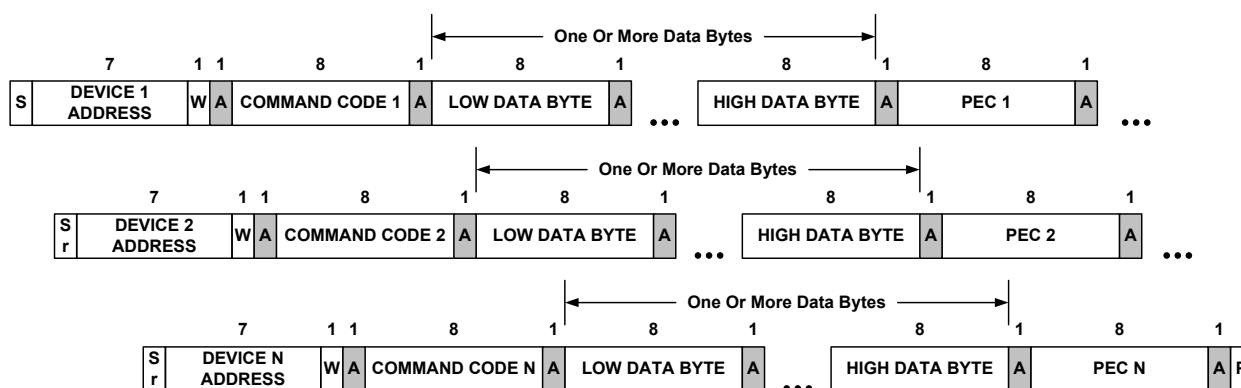
This process continues until all devices have received their command codes, data bytes, and if used and supported, PEC byte. Then when all devices have received their information, the host or master sends a STOP condition.

If PEC is used, then each device's sub-packet has its own PEC byte, computed only for that device's sub-packet, including that device's address.

When the devices who have received a command through this protocol detect the STOP condition, they are to immediately begin execution of the received command.



**Figure 2. Group Command Protocol Without PEC**



**Figure 3. Group Command Protocol With PEC**

When using Packet Error Checking with the Group Command Protocol, the PEC byte is calculated using only the address, command and data bytes for each device. For example, in Figure 3, PEC 1 is calculated using Device Address 1 including the Write bit, Command Code 1, and the data associated with Command Code 1. PEC 1 need only be calculated by the device at Device Address 1.

Similarly, PEC Byte 2 is calculated using Device Address 2 including the Write bit, Command Code 2, and the data associated with Command Code 2. Device 1 must not continue calculating PEC 1 after it sees the Repeated Start.

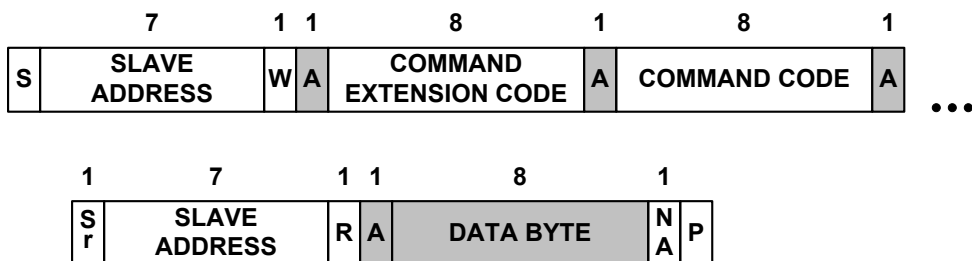
### 5.6.2. Extended Command Protocol

The Extended Command protocol allows for an extra 256 command codes. This is done by making the command code two bytes. The first byte (the low data byte) is a reserved value indicating that the extended command format is being used. The second byte (the high order byte) is the PMBus command to be executed.

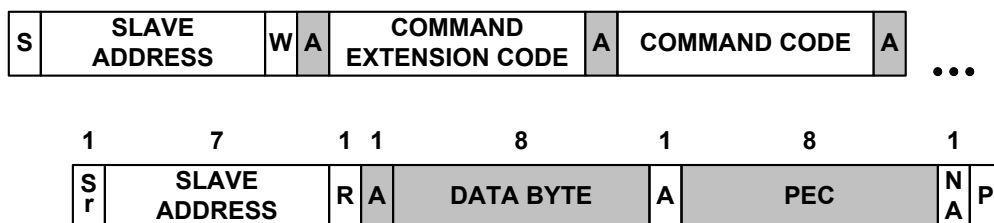
Shown below in Figure 4 through Figure 9 are examples of the Extended Command Protocol with commands that use the READ BYTE, WRITE BYTE, READ WORD, and WRITE WORD protocols, but the Extended Command Protocol can be used with any PMBus command and its associated protocol, including Manufacturer Specific Commands and any custom protocols they may use.

5.6.2.1. Example: Extended Command Protocol For READ BYTE and WRITE BYTE

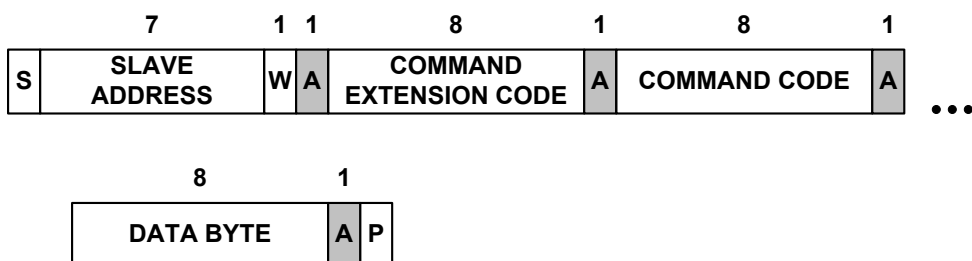
The use of the Extended Command Protocol with the READ BYTE and WRITE BYTE protocols are illustrated below.



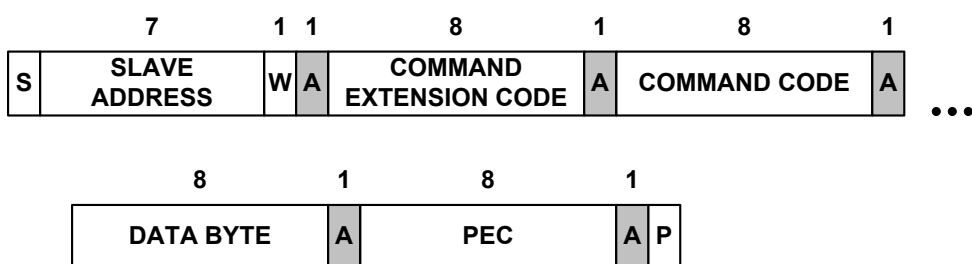
**Figure 4. Extended Command Read Byte Protocol**



**Figure 5. Extended Command Read Byte With PEC**



**Figure 6. Extended Command Write Byte Protocol**



**Figure 7. Extended Command Write Byte Protocol With PEC**

5.6.2.2. Example: Extended Command Protocol For READ WORD and WRITE WORD

The use of the Extended Command Protocol with the READ WORD and WRITE WORD protocols are illustrated below.

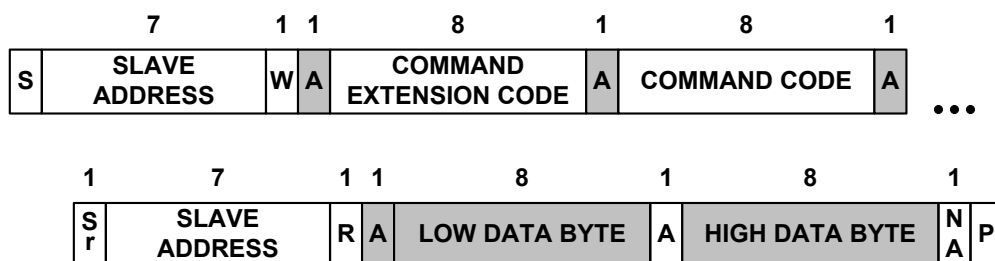


Figure 8. Extended Command Read Word Protocol

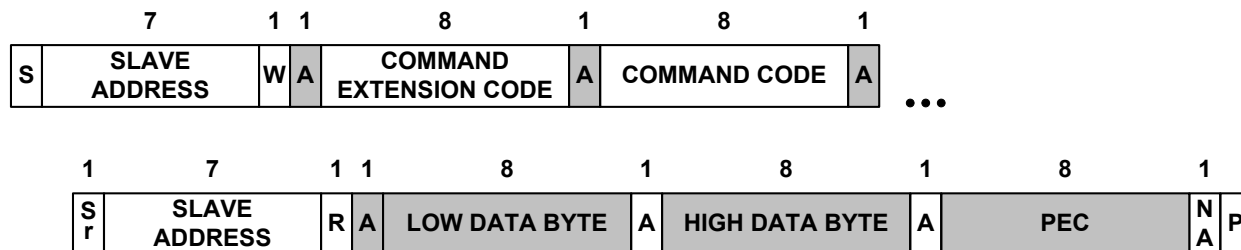


Figure 9. Extended Command Read Word Protocol With PEC

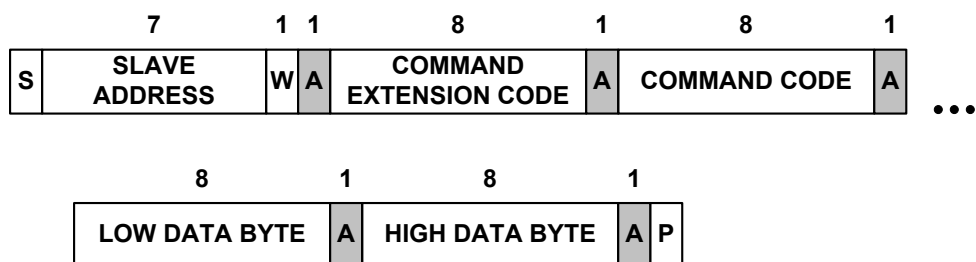


Figure 10. Extended Command Write Word Protocol

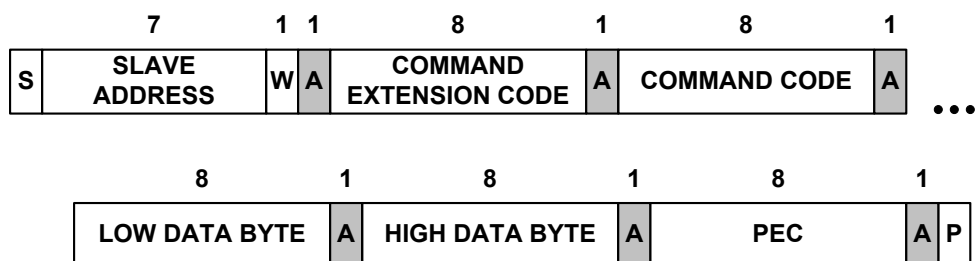


Figure 11. Extended Command Write Word Protocol With PEC

### 5.6.3. ZONE\_READ And ZONE\_WRITE Protocols

#### 5.6.3.1. Overview

The ZONE\_READ (Section 5.6.3.2) and ZONE\_WRITE (Section 5.6.3.8) protocols are used to simultaneously read from or write to some or all of the devices on a bus. For example, a system might have four major subsystems, each with its own set of power sources. Each power source for a given subsystem could be assigned to one read zone and one write zone. The master could then, for example, turn all of the power sources for one subsystem off simultaneously with one ZONE\_WRITE command. The master could also use the ZONE\_READ protocol to request the status of all power sources for that subsystem in one bus transaction.



Support for ZONE\_READ is optional. To support ZONE\_READ, a PMBus device must support clock stretching and repeated START conditions as defined in the SMBus specification.

For the purposes of zone operations, a “device” may be an IC, power converter, or other entity that responds to commands sent to a particular PMBus address. A “device” may also refer to an entity that responds to commands sent to a PAGE within that entity. For example, a power management IC may contain three complete PWM controllers. Each of those controllers may be accessed through the power management IC’s PMBus address and a PAGE number. One PWM controller may be assigned to PAGE 01h, another may be assigned to PAGE 02h, and the third may be assigned to PAGE 03h. In this case, each of the PWM controllers could be a zone capable device the can each be assigned to the same or different write and read zone numbers.

Zone operations start with the master sending the ZONE\_CONFIG command to each device to assign it to a zone number. Devices may be assigned to one zone for ZONE\_WRITE operations and assigned to the same or different zone for ZONE\_READ operations. The ZONE\_CONFIG command is sent individually to each device to be assigned to a zone. This means that if there are 12 devices that need to be configured for zone operations the master will have to send 12 different ZONE\_CONFIG commands.

It is also possible to configure a device that supports zone operations to respond to “No Zone” so that it does not respond any ZONE\_WRITE or ZONE\_READ operations.

Once the master has completed configuring devices that will be accessed through zone operations, the master uses the ZONE\_ACTIVE command to set the Active Write Zone and the Active Read Zone. The master can also set the Active Write Zone or Active Read Zone to the “All Zone” to instruct all devices participating in zone operations to respond.

When a device receives a ZONE\_ACTIVE command it compares the now active write and read zones with the write and read and zones to which it had been previously assigned. If the new Active Write Zone matches the write zone to which the device had previously been assigned, or if the Active Write Zone is set to the “All Zone”, it responds to subsequent ZONE\_WRITE operations. If there is no match, or if the device write zone had been set to “No Zone”, the device ignores subsequent ZONE\_WRITE operations.

Similarly for read zone, if the new Active Read Zone matches the read zone to which the device had previously been assigned, or if the Active Read Zone is set to the “All Zone”, it responds to subsequent ZONE\_READ operations. If there is no match, or if the device had been set to “No Zone”, the device ignores subsequent ZONE\_READ operations.

See Part II of the PMBus specification [A01] for descriptions of the ZONE\_CONFIG and ZONE\_ACTIVE commands.

PMBus Application Note AN001, *Using The ZONE\_READ And ZONE\_WRITE Protocols* [R01], provides several examples how of the ZONE\_WRITE and ZONE\_READ protocols can be used.

### 5.6.3.2. ZONE\_READ Protocol

As described above, the ZONE\_READ protocol is used to retrieve information from all devices in a specific read zone in one transaction.

A ZONE\_READ is initiated by the master by putting on the bus the ZONE\_READ address, 28h, followed with the R/W# bit set to write (0b). This notifies all of the devices configured to participate in ZONE\_READ operations (their assigned read zone matches the Active Read Zone or the Active Read Zone is set to the “All Zone”(FFh)) to prepare to receive a CONTROL COMMAND CODE. The CONTROL COMMAND CODE (Section 5.6.3.5) controls which data is to be returned, the format of that data, and flow of the returned data.

One of the options in the CONTROL COMMAND CODE is for the slaves to send the master basic status information. If this is the case, the CONTROL COMMAND CODE is followed by a STATUS MASK (Section 5.6.3.7). The STATUS MASK allows the master to select only the status information it wants and to filter out information it does not want.

Another option in the CONTROL COMMAND CODE is for the slaves to respond by sending data in response to the reading of a specific PMBus command. In this case, the CONTROL COMMAND CODE is followed by the PMBus command code.

Furthermore, another option in the COMMAND CONTROL CODE controls whether the master only wants information from the first device to respond and not lose the bit-wise arbitration of the data being returned by all responding devices, or whether the master will continue the READ ZONE operation until all devices have had a chance to complete their response.

After sending the COMMAND CONTROL CODE and either a STATUS MASK or PMBus Command Code to the ZONE\_READ address, the master then sends a REPEATED\_START, the ZONE\_READ address (28h) followed by the R/W# bit set to read (1h).

At this point all devices configured to respond to a READ ZONE operation start transmitting the requested data. As multiple devices typically will be responding simultaneously, the natural SMBus bit-wise arbitration will determine the order in which the devices respond. Once a device loses the bit-wise arbitration, it stops sending data.

The slaves return:

- The data requested by the master,
- A byte consisting of its seven bit address plus a PAGE STATUS bit, and
- The SLAVE PAGE byte.

If the PAGE STATUS is 0b, the SLAVE PAGE byte is always FFh and the returned information is either from:

- A device that does not support pages, or
- From a device that supports pages but the information is “global” to that device.

Global data is information that applies across all pages of the device. If the device were a power management IC, a possible example of global data is its temperature.

If the PAGE STATUS bit is 1b, then the device supports PAGES and the data being returned is from the PAGE whose number is returned in the SLAVE PAGE byte. A

device that supports PAGES must respond with data from every PAGE that is configured to participate in ZONE\_READ operations (the PAGE's read zone is equal to the Active Read Zone or is equal to the "All Zone").

If a device supports PAGES and global data and/or status information, and a request for global data or status information is received by the device, then the device shall return the global information and use the current PAGE setting as the value for the SLAVE PAGE byte.

Once a device successfully completes transmitting its data without losing arbitration, it stops responding to the ZONE\_READ address and NACKs the ZONE\_READ address and R/W# bit until a STOP condition is detected. This means that a device that does not support PAGES shall only successfully respond once during a given ZONE\_READ operation.

If the master has set the option in the COMMAND CONTROL CODE that it only wants the data from the first device to respond without losing the bit-wise arbitration, the master will terminate the ZONE\_READ with a STOP condition. The STOP condition is used by the master to signal the end of the ZONE\_READ operation. With the end of the ZONE\_READ operation, all devices reset themselves to be ready to accept a PMBus command or participate in another ZONE operation.

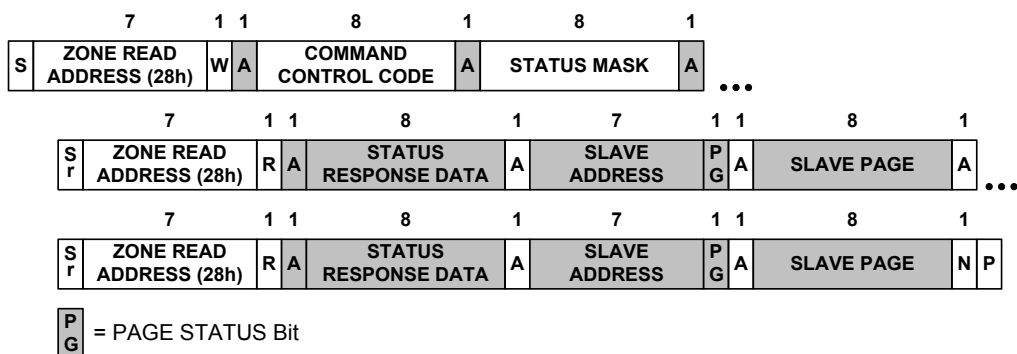
If the master has set the option in the COMMAND CONTROL CODE that it wants responses from all devices participating in the READ ZONE operation, after it has received the data from the first device that sends its data without losing arbitration, the master sends another REPEATED START, the ZONE\_READ address (28h) followed by the R/W# bit set to read (1h). All devices that have not sent their data without losing arbitration ACK and start sending their data. Again, when a device loses a bit-wise arbitration, it stops sending data until the next REPEATED START, ZONE\_READ Address, and R/W# set to read (1h) is received.

When the master has received the data from the second device that sends its data without losing arbitration, it sends another REPEATED START, the ZONE\_READ address (28h) followed by the R/W# bit set to read (1h). All devices that have not sent their data without losing arbitration ACK start sending their data. Again, when a device loses a bit-wise arbitration, it stops sending data until the next REPEATED START, ZONE\_READ Address, and R/W# set to read (1h) is received.

The master continues sending REPEATED START conditions and reads of the ZONE\_READ address until all devices have responded. This may be determined by the master checking the devices that have responded against a list of devices it knows to be in the Active Read Zone or when the master receives a NACK in response to the ZONE\_READ address and R/W#. Once the ZONE\_READ is complete the master signals to all devices in the Active Read Zone that the transaction is complete by sending a STOP condition. The STOP condition signals the devices participating in the ZONE\_READ to reset themselves and be ready to accept a new PMBus transaction or participate in another ZONE operation.

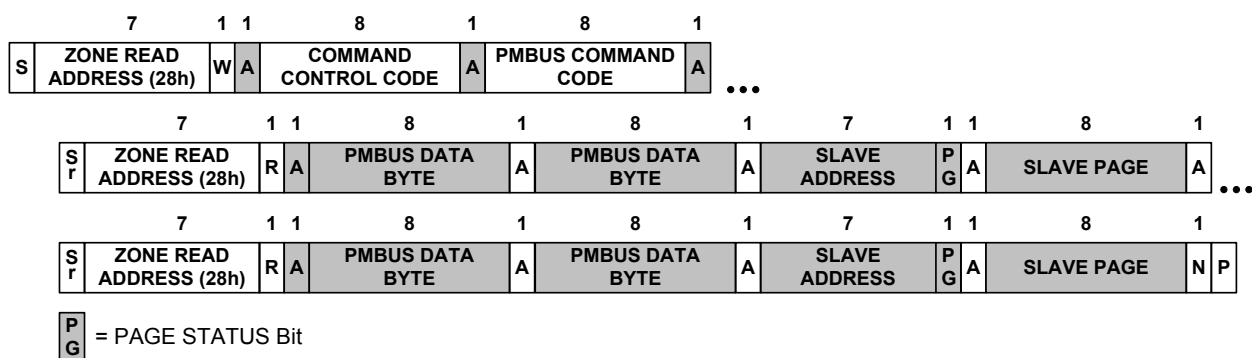
The master may terminate a ZONE\_READ before all devices have responded by issuing a STOP condition instead of a REPEATED START. The slave devices shall not treat this early termination of the ZONE\_READ as an error or fault condition. When the STOP condition is received any slaves that were participating in the ZONE\_READ shall reset themselves and be ready to accept a new PMBus transaction including another ZONE\_READ.

Figure 12 illustrates a ZONE\_READ operation in which the master is requesting status information and two slaves are responding.



**Figure 12. ZONE\_READ With STATUS DATA Response**

Figure 13 illustrates a ZONE\_READ operation in which the master sends a PMBus command that returns two bytes data. Two slaves are shown to be responding.

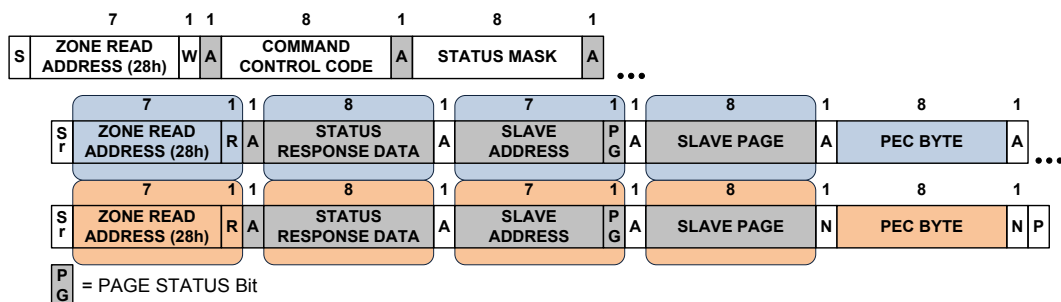


**Figure 13. ZONE\_READ With PMBus Command**

## 5.6.3.3. ZONE\_READ with PEC

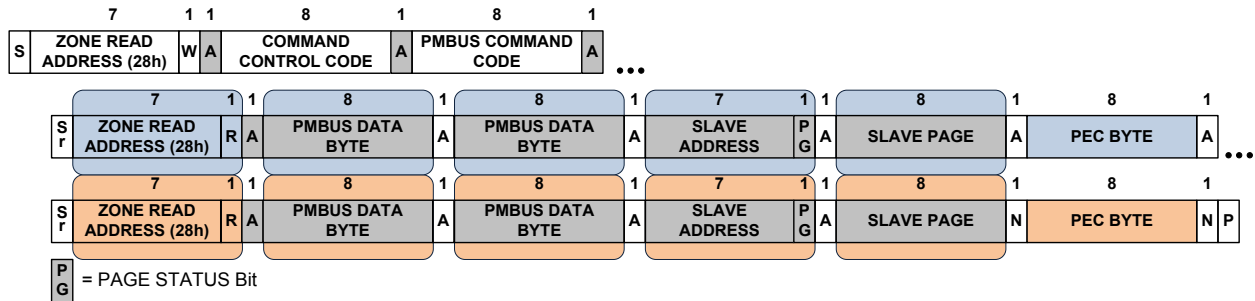
Packet Error Checking (PEC) may be used with ZONE\_READ operations.

Figure 14 illustrates a ZONE\_READ operation in which the master requests status information. The slaves return their status information, address, SLAVE PAGE number, and a PEC byte. The PEC byte is calculated by each slave on its individual sub-transaction including the byte including the ZONE\_READ address and R/W# bit. The colored boxes in Figure 14 highlight the data used to calculate the same colored PEC byte.



**Figure 14. ZONE\_READ Returning Status Information with PEC**

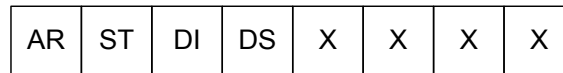
Figure 15 illustrates a ZONE\_READ operation in which the master send a PMBus command that returns two bytes of data. The slaves return the data, address, SLAVE PAGE number, and a PEC byte. The PEC byte is calculated by each slave on its individual sub-transaction including the byte including the ZONE\_READ address and R/W# bit. The colored boxes in Figure 15 highlight the data used to calculate the same colored PEC byte.



**Figure 15. ZONE\_READ With PMBus Command Returning Data And PEC**

## 5.6.3.4. COMMAND CONTROL CODE

The COMMAND CONTROL CODE sets the information to be returned, how that data is formatted, and the flow of that data. The COMMAND CONTROL CODE byte is a bit field with four active bits and four reserved bits as shown in Figure 16.



**Figure 16. Command Control Code**

### 5.6.3.4.1 COMMAND CONTROL CODE bit [7]: AR (All Respond)

The AR bit defines the All Respond mode.

When AR = 0, all devices respond once with their data and address, but only one will win the bit-wise arbitration. Thus arbitration success and prioritization are set by the value of the data that is sent by the responding PMBus devices.

When AR = 1, all devices respond with their data and address to every read to the ZONE\_READ address (28h) until they are successful in sending information to the system host. The responses become appended in succession until all devices have successfully arbitrated their response. When a device loses arbitration, it shall try again after the last successfully arbitrating device is finished. Each device that is successful in transmitting its response must ignore subsequent responses until a STOP condition is detected.

### 5.6.3.4.2 COMMAND CONTROL CODE bits [6:4]: ST, DI, and DS

COMMAND CONTROL CODE bits [6:4] are:

- ST: Status, governing whether status information or response to a PMBus command is being requested
- DI: Data Inversion, governing whether the bits in the returned data are bit-wise inverted or not
- DS: Data Swap/byte order, governing whether data bytes are returned in the SMBus standard least significant byte first or with the most significant byte first.

Table 2 provides a concise summary of how these bits control the returned data.

**Table 2. Summary Of The ST (Status), DI (Data Inversion),  
And DS (Data Significance) Bit Meanings**

ST	DI	DS	Action	Data Inversion	Data Swap/Byte Order
0	0	0	Device returns PMBus command data	Data is returned with no inversion of the data bits	SMBus standard
0	0	1			Most significant byte first
0	1	0		Each bit in the returned data is inverted	SMBus standard
0	1	1			Most significant byte first
1	0	0	Device returns the status byte requested by the DS bit ANDed with the bit-wise inversion of the STATUS MASK byte	Data is returned with no inversion of the data bits	Upper byte of STATUS_WORD returned
1	0	1			STATUS_BYTE (lower byte of STATUS_WORD) returned
1	1	0		Each bit in the returned data is inverted	Upper byte of STATUS_WORD returned
1	1	1			STATUS_BYTE (lower byte of STATUS_WORD) returned

#### 5.6.3.4.3 COMMAND CONTROL CODE bits [3:0]: Reserved

COMMAND CONTROL CODE bits [3:0] are reserved for future use and should always be 0000b.

#### 5.6.3.5. STATUS MASK Byte

The STATUS MASK byte is used to control the status information returned to the system host.

As described in Section 5.6.3.4.2, when the ST bit is set, the slave device returns one byte of status information. Which byte, either the STATUS\_BYTE or the upper byte of STATUS\_WORD, is controlled by the setting of the DS bit.

When the slave is returning a byte of status information, which we call here just STATUS, the master can instruct the slave to only return bits of interest through the use of the STATUS MASK byte. A bit set to 1b in the STATUS MASK will cause the corresponding bit in STATUS to be set to be masked, that is, set to 0b. The data returned to the master is calculated as:

$$\text{RETURNED DATA} = \text{STATUS} \& \text{INV}(\text{STATUS MASK}).$$

Note that if the STATUS MASK is set to FFh, the RETURNED DATA will be 00h, which provides no useful information to the master. With this in mind, the STATUS MASK value of FFh is reserved for a special use. If STATUS MASK is set to FFh, then the slave devices responding to the initiation of a ZONE\_READ shall reset any

status bits or flags that had been set causing the device to ignore reads to the ZONE\_READ address, such as when a device won the bit-wide arbitration during a ZONE\_READ operation. Ordinarily such a bit or flag would be reset when the STOP condition terminating the ZONE\_READ operation is detected. However, this provides a means to force this reset if for some reason the STOP condition was not properly issued, was corrupted by noise, or otherwise not properly received by a slave device.

### 5.6.3.6. Device Address Discovery

Slave address discovery can be done by setting AR=1 and ST=1. In this mode each compatible device returns its status and address through arbitration. The PMBus Command Code can be set to any value except FFh for this function.

### 5.6.3.7. ZONE\_READ Restrictions

#### 5.6.3.7.1 Prohibited SMBus Transaction Protocols

When a ZONE\_READ is being initiated, PMBus commands that use the following SMBus transaction protocols shall not be used:

- SEND BYTE
- PROCESS CALL
- BLOCK WRITE-BLOCK READ PROCESS CALL

Examples of prohibited PMBus commands are CLEAR\_FAULTS (SEND BYTE protocol) and QUERY (BLOCK WRITE-BLOCK READ PROCESS CALL protocol),

Devices that support ZONE\_READ that receive a PMBus command that uses a prohibited protocol when a ZONE\_READ is being initiated shall either:

- NACK the byte containing the PMBus command code or
- Declare a communication fault and respond as described in Section 10 of Part II of the PMBus specifications [A01].

#### 5.6.3.7.2 Prohibited PMBus Commands

The following commands, which are not prohibited due to SMBus transaction type (Section 5.6.3.7.1) shall not be used when initiating a ZONE\_READ:

- None in this revision

Devices that support ZONE\_READ that receive a prohibited PMBus command when a ZONE\_READ is being initiated shall either:

- NACK the byte containing the PMBus command code or
- Declare a communication fault and respond as described in Section 10 of Part II of the PMBus specifications [A01].

### 5.6.3.8. Zone Write Protocol

The ZONE\_WRITE protocol is used to send a PMBus command and the associated data to all devices in the Active Write Zone in one transaction.

A ZONE\_WRITE operation is initiated when the master puts the ZONE\_WRITE address, 37h, on the bus followed by the R/W# bit set to write (0b). All devices configured to respond to a ZONE\_WRITE (their write zone matches the Active Write Zone or the Active Write Zone equals the “All Zone” (FFh)) receive the PMBus command and data and process them as any other PMBus command.

Figure 17 illustrates a ZONE\_WRITE that sends a PMBus command and two data bytes.



**Figure 17. ZONE\_WRITE Operation With PMBus Command And Two Data Bytes**

#### 5.6.3.9. ZONE\_WRITE Restrictions

##### 5.6.3.9.1 Prohibited SMBus Transaction Protocols

A ZONE\_WRITE operation cannot be used with any PMBus command that although written, returns data. This includes any PMBus command that uses the following SMBus protocols:

- PROCESS CALL
- BLOCK WRITE-BLOCK READ PROCESS CALL

##### 5.6.3.9.2 Prohibited PMBus Commands

The following PMBus commands shall not be sent as part of a ZONE\_WRITE operation:

- PAGE
- PAGE\_PLUS\_READ
- ZONE\_CONFIG

Devices that support ZONE\_WRITE that receive a prohibited PMBus command when a ZONE\_WRITE is being initiated shall either:

- NACK the byte containing the PMBus command code or

Declare a communication fault and respond as described in Section 10 of Part II of the PMBus specifications [A01].

#### 5.7. Exceptions To The SMBus Specification

None in this revision of the PMBus specification.

### 6. Addressing

PMBus devices use seven bit addresses as described in the SMBus specification [A03].

Addresses described as reserved in the SMBus specification ([A03]) or the I<sup>2</sup>C specification [A04] may not be used.

Physical addresses are programmed through pins. PMBus devices are not required to be able to support the entire seven bit address space through pin programming. The addresses available through pin programming are left to the PMBus device manufacturer. How a device's address is set shall be described in the device's product literature.



## **7. Communication From PMBus Devices To The Host**

### **7.1. Communicating Over The Bus**

As an option, PMBus devices may temporarily become bus masters and communicate with the host through the SMBus Host Notify Protocol defined in SMBus specification [A03].

The contents of the two data bytes shall be the same as the contents of the data bytes for the PMBus STATUS\_WORD command (PMBus specification, Part II [A01]).

### **7.2. Communicating With An Interrupt Signal**

As an option, PMBus devices may notify the host that they want to communicate with the host by asserting the SMBALERT# signal, as described in Appendix A of the SMBus specification, [A03].

## **8. Hardwired Signals**

### **8.1. Electrical Interface**

The electrical interface for hardwired signals shall comply with the High Power DC Electrical Interface for 100 kHz devices of the SMBus specification [A03].

The only exception to this are the pins used to set the physical address. If the electrical interface to address pins is not in compliance with the High Power DC Electrical Interface for 100 kHz devices of the SMBus specification, [A03], the electrical interface shall be described in the PMBus device's product literature.

### **8.2. Timing**

There are no specific requirements on when a PMBus device must respond to a change in state of a hardwired signal.

### **8.3. Control Signal (CONTROL)**

The CONTROL signal is an input signal on a PMBus device. It is used to turn the device on and off in conjunction with commands received via the serial bus. For more information, see Part II of the PMBus specification [A01].

It can be configured as an active high or active low signal through the ON\_OFF\_CONFIG command (PMBus specification, Part II [A01]).

The electrical interface for the CONTROL signal shall comply with the High Power DC Electrical Interface for 100 kHz devices of the SMBus specification [A03].

This signal is optional but recommended.

### **8.4. Write Protect (WP)**

The PMBus protocol supports the use of optional Write Protect (WP) signal inputs.

There may be more than one Write Protect input signal with each signal protecting a different type or different portions of memory.

If the WP input is present, then no updates to any internal memory is allowed when the WP input is high or open. Updates are permitted only when the WP signal is low.

The electrical interface for WP signals shall comply with the High Power DC Electrical Interface for 100 kHz devices of the SMBus specification [A03].

### **8.5. Other Pins**

PMBus devices may use pins for programming or configuration. The function and electrical interface of any such pins shall be described in the device's product literature.

Examples of such pins are a RESET pin or pins that are used to set the output voltage to the high or low margin voltages.

Pins that provide a binary input (high or low) shall have an electrical interface that meets requirements of the High Power DC Electrical Interface for 100 kHz devices of the SMBus specification [A03].

For pin functions with an equivalent command (PMBus specification, Part II [A01]), the command received from the bus will override the pin programming. See the description in Part II of the PMBus specification [A01] on how a PMBus device configures its operating parameters for more information.

## **9. Accuracy**

Each PMBus device will specify in its product literature the accuracy with which the output voltage and other parameters can be set and reported.

## **10. Firmware Updates**

PMBus devices can, as an option, support upgrading its firmware via the SMBus interface. The methods of such updates are left to the discretion of the device manufacturers.

## **APPENDIX I. 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 I of the PMBus specification from Revision 1.3 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.

- Section 5.6.3, describing the ZONE\_READ and ZONE\_WRITE, protocols has been completely re-written to clarify and better explain the zone protocols. Figures in this section have been updated and some figures have been added.