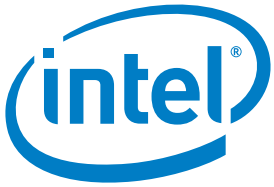


Intel® Enpirion® Power Solutions

EM21xx Family Supported PMBus™ Commands

Application Note

Contents

List of Figures

List of Tables

1. Introduction

The EM21xx family supports the PMBus™ protocol to enable the use of configuration, monitoring and fault management during run-time. The PMBus host controller is connected to EM21xx via the PMBus pins.

An EM21xx device contains a true digital controller, and supports many operations including configuring the output voltage sensing and extensive fault monitoring and handling options that can be communicated back to the user through the PMBus interface.

The family is designed to comply with the PMBus specification revision 1.2 (September 2010), and supports both 100 kHz and 400 kHz bus operation. The devices support packet error correction (PEC) and also operate by stretching clock pulses when required. The EM21xx family of modules only supports the linear data format.

This application note contains a detailed description of all PMBus supported commands by the EM21xx family.

For greater detail, please refer to “PMBus™ Specification Parts I & II Revision 1.2”, available at www.pmbus.org, and “SMBus Version 2.0” specifications, available at www.smbus.org. Please also refer to the individual EM21xx device datasheet, available at www.altera.com/enpirion.

2. Timing and Bus Specification

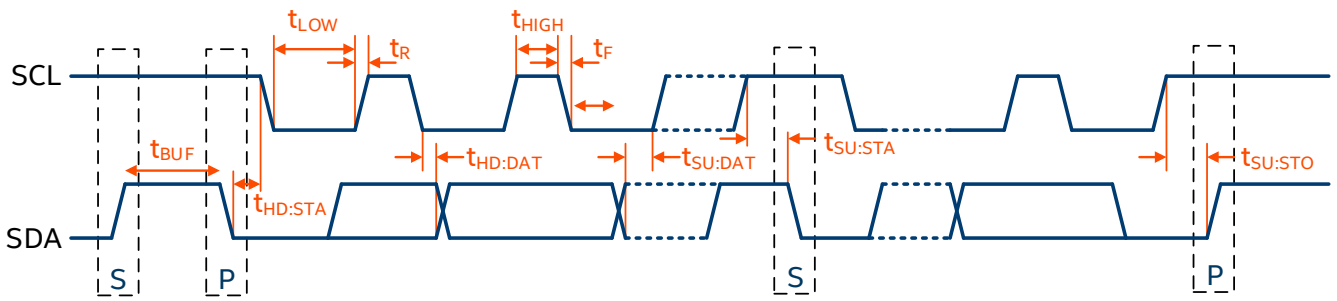


Figure 1: PMBus Timing Diagram

Table 1: PMBus Timing Specification

Parameter	Symbol	Min	Typ	Max	Units
SMBus operation frequency	f_{SMB}	10	100	400	kHz
Bus free time between start and stop	t_{BUF}	1.3			µs
Hold time after start condition	$t_{HD:STA}$	0.6			µs
Repeat start condition setup time	$t_{SU:STA}$	0.6			µs
Stop condition setup time	$t_{SU:STO}$	0.6			µs
Data hold time	$t_{HD:DAT}$	300			ns
Data setup time	$t_{SU:DAT}$	150 *(Note 1)*			ns
Clock low time-out	$t_{TIMEOUT}$		25	35	ms
Clock low period	t_{LOW}	1.3			µs
Clock high period	t_{HIGH}	0.6			µs
Cumulative clock low extend time	$t_{LOW:SEXT}$			25	ms
Clock or data fall time	t_F			300	ns
Clock or data rise time	t_R			300	ns

NOTE: *(1) The EM21xx family fully complies with PMBus 1.2 specifications for operation up to 100 kHz on SCL. They may be operated at frequencies up to at least 400 kHz on SCL if $t_{SU:DAT}$ is maintained greater than 150 ns.*

2.1 The Linear Data Format

The EM21xx family uses the linear data format for communicating data. The format is two bytes in length, in 2's complement binary. The linear format consists of two bytes, where the Least Significant Byte and the the lower three bits of the Most Significant Byte representing the Mantissa, and the upper five bits of the Most Significant byte representing the Exponent (scaling factor). The two bytes consist of two parts as outlined in Figure 2 below.

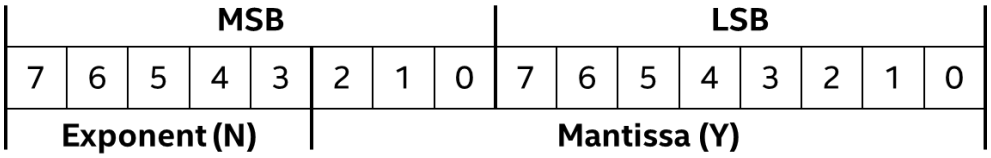


Figure 2: Linear Data Format

The following equation translates the linear data format to the real world value:

$$X = Y * 2^N$$

Where:

- X is the "real world" value
- Y is an 11 bit, two's complement integer
- N is a 5 bit, two's complement integer

3. List of Supported Commands

The EM21xx family supports two different sets of configuration parameters. The first set of parameters can only be configured during the configuration phase of the module. These values are written into the one-time programmable (OTP) memory and cannot be changed using PMBus commands during run-time. A second set of parameters can be configured during run-time using the appropriate PMBus commands.

Table 2: List of Supported PMBus Commands

Command Code	PMBus Parameter	Description
01h	OPERATION	On/Off command
02h	ON_OFF_CONFIG	On/off configuration
03h	CLEAR_FAULTS	Clear status information
10h	WRITE_PROTECT	Protect against changes
11h	STORE_DEFAULT_ALL	Copy entire memory into OTP
12h	RESTORE_DEFAULT_ALL	Copy entire memory from OTP
13h	STORE_DEFAULT_CODE	Copy single parameter into OTP
14h	RESTORE_DEFAULT_CODE	Copy single parameter from OTP
20h	VOUT_MODE *(Note 2)*	Exponent of the VOUT_COMMAND value
21h	VOUT_COMMAND	Set output voltage
22h	VOUT_TRIM	Apply a fixed offset voltage
23h	VOUT_CAL_OFFSET	Apply a fixed offset voltage
25h	VOUT_MARGIN_HIGH	Sets Max Value
26h	VOUT_MARGIN_LOW	Sets Min Value
29h	VOUT_SCALE_LOOP	Scalar for output voltage divider
2Ah	VOUT_SCALE_MONITOR	Scalar for output voltage divider
35h	VIN_ON	Input voltage turn on threshold
36h	VIN_OFF	Input voltage turn off threshold
40h	VOUT_OV_FAULT_LIMIT	Over-voltage fault limit
41h	VOUT_OV_FAULT_RESPONSE	Over-voltage fault response
42h	VOUT_OV_WARN_LIMIT	Over-voltage warning level
43h	VOUT_UV_WARN_LIMIT	Under-voltage warning level

Command Code	PMBus Parameter	Description
44h	VOUT_UV_FAULT_LIMIT	Under-voltage fault level
45h	VOUT_UV_FAULT_RESPONSE	Under-voltage fault response
55h	VIN_OV_FAULT_LIMIT	Over-voltage fault limit
56h	VIN_OV_FAULT_RESPONSE	Over-voltage fault response
57h	VIN_OV_WARN_LIMIT	Over-voltage warning level
58h	VIN_UV_WARN_LIMIT	Under-voltage warning level
59h	VIN_UV_FAULT_LIMIT	Under-voltage fault level
5Ah	VIN_UV_FAULT_RESPONSE	Under-voltage fault response
5Eh	POWER_GOOD_ON	Power good on threshold
5Fh	POWER_GOOD_OFF	Power good off threshold
60h	TON_DELAY	Turn-on delay
61h	TON_RISE	Turn-on rise time
62h	TON_MAX_FAULT_LIMIT	Turn-on maximum fault time
64h	TOFF_DELAY	Turn-off delay
65h	TOFF_FALL	Turn-off fall time
66h	TOFF_MAX_WARN_LIMIT	Turn-off maximum warning time
78h	STATUS_BYTE	Unit status byte
79h	STATUS_WORD	Unit status word
7Ah	STATUS_VOUT	Output voltage status
7Bh	STATUS_IOUT	Output current status
7Ch	STATUS_INPUT	Input status
7Eh	STATUS_CML	Communication and memory status
80h	STATUS_MFR_SPECIFIC	Manufacturer specific status
88h	READ_VIN	Input voltage read back
8Bh	READ_VOUT	Output voltage read back
8Ch	READ_IOUT	Output current read back
8Eh	READ_TEMPERATURE	Temperature read back
94h	READ_DUTY_CYCLE	Current Duty Cycle read back
95h	READ_FREQUENCY	Switching frequency read back
98h	PMBUS_REVISION	PMBus revision

Command Code	PMBus Parameter	Description
99h	MFR_ID	Manufacturer ID
9Ah	MFR_MODEL	Manufacturer model identifier
9Bh	MFR_REVISION	Manufacturer product revision
9Eh	MFR_SERIAL	Serial number
A0h	MFR_VIN_MIN	Minimum input voltage
A4h	MFR_VOUT_MIN	Minimum Output voltage
D0h	MFR_SPECIFIC_00	Write word (once) / Read word – 2 bytes
D1h	MFR_SPECIFIC_01	Write word / read word – 2 bytes
D2h	MFR_READ_VCC	Read-back value for VCC
D3h	MFR_RESYNC	Reacquire the SYNC pin input signal
DAh	MFR_RTUNE_CONFIG	Configure RTUNE compensator Index and scaling factor
DDh	MFR_RTUNE_INDEX	Returns Index derived from Resistor detected on RTUNE pin
DEh	MFR_RVSET_INDEX	Returns Index derived from Resistor detected on RVSET pin
E0h	MFR_VOUT_OFF	Sets the target turn off Voltage
E2h	MFR_OT_FAULT_LIMIT	Over Temperature Fault level
E3h	MFR_OT_WARN_LIMIT	Over Temperature Warning level
E5h	MFR_OT_FAULT_RESPONSE	Over Temperature Fault Response
E6h	MFR_TEMP_ON	Over Temperature On level
E7h	MFR_PIN_CONFIG	Enable/Disable operation VTRACK, RTUNE, RVSET, SYNC
E9h	MFR_STORE_CONFIG_ADDR_READ	Read back a configuration Value
EAh	MFR_STORE_PARAMS_REMAINING	# of STORE_DEFAULT_ALL commands remaining
EBh	MFR_STORE_CONFIGS_REMAINING	# of full configurations remaining
ECh	MFR_STORE_CONFIG_BEGIN	Commence programming of OTP
EDh	MFR_STORE_CONFIG_ADDR_DATA	Program a configuration Value
EEh	MFR_STORE_CONFIG_END	Complete programming of OTP

NOTE: *(2) VOUT_MODE is read only for the EM21xx family.*

4. Detailed Description: Supported PMBus Commands

Power conversion is not possible before the initial configuration is written to the digital controller integrated within the EM21xx device. Power conversion can only begin once a a configuration is written to the device, and provided all circuit conditions are clear (for example: no faults detected, etc.)

For the purpose of this document, all appropriate default value settings mentioned within are set via the default configuration taken from the Intel® Enpirion® Digital Power Configurator, which is a graphical user interface (GUI) designed to be used with the EM21xx family of devices. If the user changes the initial default settings values from the GUI (either through the GUI or directly via PMBus), then the default settings values defined in this document are no longer appropriate. All default settings affected in this way will be highlighted in this document with *underlining and italics*; example commands that are impacted include VIN_ON and VOUT-_OV_FAULT_LIMIT.

4.1 OPERATION – 01h

The OPERATION command is used to turn the unit on and off in conjunction with the input from the CONTROL pin. The unit stays in the commanded operating mode until a subsequent OPERATION command or change in the state of the CONTROL pin instructs the device to change to another mode. The supported operation modes are listed in

Table 3.

Table 3: Supported PMBus OPERATION Modes

OPERATION (Read/Write)					
Bits [7:6]	**Bits [5:4]**	**Bits [3:2]**	**Bits [1:0]**	**Unit On or Off**	**Margin State**
00	XX	XX	XX	Immediate Off (No Sequencing)	N/A
01	XX	XX	XX	Soft Off (With Sequencing)	N/A
10	00	XX	XX	On	Off
10	01	01	XX	On	Margin Low (Ignore Fault)
10	01	10	XX	On	Margin Low (Act on Fault)
10	10	01	XX	On	Margin High (Ignore Fault)
10	10	10	XX	On	Margin Low (Act on Fault)

NOTE: *Bits containing "XX" are don't care and, when written to, the contents are ignored.*

The EM21xx device will treat any values outside those defined in

Table 3 as an undefined and invalid command, and they will be ignored and a communications fault will be declared. It will set the CML [1] bit of the STATUS_BYTE register, set the INVALID_COMMAND [7] of the STATUS_CML register, and notify the host through asserting the SMBALERT# signal (output goes low).

When the device is configured for Margin High (Act on Fault) and an output overvoltage warning or fault occurs when the output is margined high, then the controller responds to this warning or fault as programmed. Conversely, if configured for Margin High (Ignore Fault) and an output overvoltage warning of fault occurs when the output is margined high, then the controller ignores this warning or fault. When the device is programmed for Margin Low, the same operation occurs in the event of an under-voltage warning or fault being detected.

4.2 ON_OFF_CONFIG – 02h

The ON_OFF_CONFIG command gives the user a number of configuration options as to how the unit may be turned on and off, including when power is applied. It is one byte in size, but only bits [4:0] may be written to. All bits may be read from. Table 4 outlines the ON_OFF_CONFIG Data Byte Structure. If a data byte not shown in Table 4 is received, the device will treat this as invalid data and a communication fault will be declared. The device will set the CML bit [1] of the STATUS_BYTE register, set the INVALID_COMMAND [7] of the STATUS_CML register, set the Packet Error Check Failed bit [5] in the STATUS_CML, and notify the host through asserting the SMBALERT# signal (output goes low).

Table 4: ON_OFF_CONFIG Data Byte Structure

Bits	Function	Format	Value	Description
[7:5]	Reserved per PMBus specification	Read	000	
[4]	Defines PWM begin operation	Read/Write	0	Device PWM begins once powered *(Note 3)*
			1	Device PWM begins dependent on bits [3:2] *(Note 3)*
[3]	Defines Serial Bus On/Off control OPERATION	Read/Write	0	On/Off portion of OPERATION command is ignored and device PWM begins regardless *(Note 3)*
			1	Device PWM may begin on receiving On/Off portion of OPERATION command (dependent on Control pin operation [b2]) *(Note 3)*
[2]	Defines Control pin operation	Read/Write	0	Status of Control is ignored and device PWM begins regardless *(Note 3)*

			1	Device PWM may begin when Control pin status correct (dependent on Serial Bus Control operation [b3]) *(Note 3)*
[1]	Defines polarity of Control pin	Read/Write	0	Active low (pull low to start the PWM)
			1	Active high (pull high to start the PWM)
[0]	Defines Control pin action	Read/Write	0	Turn off the output as per programmed turn-off delay time and fall time
			1	Turn off the output and stop transferring energy to the output as fast as possible

NOTE: *(3) Provided no fault conditions are present*

Bit 4 defines whether the device starts immediately upon power being applied and no fault condition present (e.g. V_{IN} UVLO, etc.). Bits 2 and 3 configure how the device will respond to the status of the Control pin and the On/Off portion of OPERATION command. Bit 1 allows the user to define the Control pin polarity. Bit 0 allows configuration of the control pin response action.

4.3 CLEAR_FAULTS – 03h

The CLEAR_FAULTS command is used to clear any fault bits that have been set in any of the status registers. Additionally, the SMBALERT signal is cleared if it was previously asserted. If a fault/warning is still present, the respective bit is immediately set again.

Upon receiving a CLEAR_FAULTS command, the device does not resume operation automatically. It requires an OPERATION command, toggling of the CONTROL pin, or the combined action of both, in order to turn off and then back on. The device is dependent on the setting of the ON_OFF_CONFIG. If a fault is still present, the device will respond as programmed, and the affected status bits will be reasserted automatically. The CLEAR_FAULTS command is write only and has no data byte.

4.4 WRITE_PROTECT – 10h

The WRITE_PROTECT command is used to control writing to the device, and is intended to provide protection against accidental changes. The WRITE_PROTECT command is one byte in size. Only values in Table 5 may be written to, but all bits may be read from.

Table 5: WRITE_PROTECT Data Byte Structure

Value	Description
0000_0000	Enable writes to all commands
0010_0000	Disable all writes except WRITE_PROTECT, OPERATION, ON_OFF_CONFIG, and VOUT_COMMAND commands
0100_0000	Disable all writes except WRITE_PROTECT and OPERATION commands
1000_0000	Disable all writes except WRITE_PROTECT command

Any values other than those defined in Table 5 will be treated as invalid, ignored, and a communications fault will be declared. The device will set the CML [1] bit of the STATUS_BYTE register, set the INVALID_COMMAND [7] of the STATUS_CML register, and notify the host through asserting the SMBALERT# signal (output goes low).

4.5 STORE_DEFAULT_ALL – 11h

The STORE_DEFAULT_ALL command is used to store the entire PMBus operating memory into OTP. To prevent unpredicted behavior during operation, the output is disabled prior to the device implementing this command. The STORE_DEFAULT_ALL command is write only and has no data bytes.

4.6 RESTORE_DEFAULT_ALL – 12h

The RESTORE_DEFAULT_ALL command is used to restore the entire PMBus operating memory from OTP. To prevent unpredicted behavior during operation, the output is disabled prior to the device implementing this command. The RESTORE_DEFAULT_ALL command is write only and has no data bytes.

4.7 STORE_DEFAULT_CODE – 13h

The STORE_DEFAULT_CODE command is used to store a single PMBus parameter into OTP. To prevent unpredicted behavior during operation, the output is disabled prior to the device implementing this command. The STORE_DEFAULT_CODE command is write only and has one data byte.

4.8 RESTORE_DEFAULT_CODE – 14h

The RESTORE_DEFAULT_CODE command is used to restore a single PMBus parameter from OTP. To prevent unpredicted behavior during operation, the output is disabled prior to the device implementing this command. The RESTORE_DEFAULT_CODE command is write only and has one data bytes.

4.9 VOUT_MODE – 20h

The VOUT_MODE command is used to retrieve information about the data format for all output voltage related commands and is one byte in length. The EM21xx device will only ever operate in linear mode for output voltage related commands, so this is a read-only command.

Table 6: VOUT_MODE Data Byte Structure

Bits	Value	Name	Description
[4:0]	10011	PARAMETER	2's complement of the exponent for the mantissa
[7:5]	000	MODE	000: Linear data format

If the user attempts to send a VOUT_MODE command, the command will be treated as invalid, ignored, and a communications fault will be declared. The device will set the CML [1] bit of the STATUS_BYTE register, set the INVALID_COMMAND [7] of the STATUS_CML register, and notify the host through asserting the SMBALERT# signal (output goes low).

4.10 VOUT_COMMAND – 21h

The VOUT_COMMAND is used to set the output voltage during run-time. It can be read and written to, and consists of two bytes. The VOUT_COMMAND data along with bits [4:0] of the VOUT_MODE command are used to calculate the output voltage value that the device regulates to.

Figure 3 shows the data byte structure for the VOUT_MODE and VOUT_COMMAND commands.

VOUT_MODE

MODE			Exponent				
7	6	5	4	3	2	1	0
0	0	0	1	0	0	1	1

VOUT_COMMAND

Data Byte High								Data Byte Low							
7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
Mantissa (V)															

Figure 3: Data Byte Structure, VOUT_MODE and VOUT_COMMAND

The output voltage in volts is calculated form the following equation:

$$Voltage = V * 2^N$$

Where:

- V is the 16-bit unsigned binary integer (VOUT_COMMAND)
- N is a 5-bit two's complement binary Integer (bits 4:0 of VOUT_MODE = -13)

An example VOUT_COMMAND value and conversion to the real world value is:

$$2666h = 9830d * 2^{-13} = 1.2V$$

4.11 VOUT_TRIM – 22h

The VOUT_TRIM command is used to apply a fixed offset voltage to the output voltage command value. The VOUT_TRIM command allows the end user to trim the output voltage using the PMBus during End of Line testing. The VOUT_TRIM command may be read and written to and consists of two bytes formatted as a two's complement binary integer. Its default value is 0000h.

4.12 VOUT_CAL_OFFSET – 23h

The VOUT_CAL_OFFSET command is used to apply a fixed offset voltage to the output voltage command value. The VOUT_CAL_OFFSET command may be read and written to and consists of two bytes formatted as a two's complement binary integer. Its default value is 0000h.

4.13 VOUT_MARGIN_HIGH – 25h

The VOUT_MARGIN_HIGH command contains the output voltage value which the output will change to when the OPERATION command is changed to "Margin High". The VOUT_MARGIN_HIGH command may be read and written to and consists of two bytes in linear mode as when using VOUT_COMMAND.

An example VOUT_MARGIN_HIGH value and conversion to the real world value is:

$$2A3Dh = 10813d * 2^{-13} = 1.32V\ (+10\%)$$

4.14 VOUT_MARGIN_LOW – 26h

The VOUT_MARGIN_LOW command contains the output voltage value which the output will change to when the OPERATION command is changed to "Margin LOW". The VOUT_MARGIN_LOW command may be read and written to and consists of two bytes in linear mode as when using VOUT_COMMAND.

An example VOUT_MARGIN_LOW value and conversion to the real world value is:

$$228Fh = 8847d * 2^{-13} = 1.08V\ (-10\%)$$

4.15 VOUT_SCALE_LOOP – 29h

For V_{OUT} > 1.325V, the EM21xx device output voltage will be sensed using a resistor divider connected to the VSENP input. When a resistor divider is used, the PMBus-commanded voltage is always the desired output voltage but the voltage that the EM21xx device senses, V_{SENSED}, may therefore be a different, scaled voltage. The VOUT_SCALE_LOOP command is used to map the PMBus-commanded output voltage and the output voltage that the EM21xx device senses, and is equal to the feedback resistor ratio used to set the EM21xx device output voltage. Figure 4 provides an overview of how a resistor divider can be used with an EM21xx device.

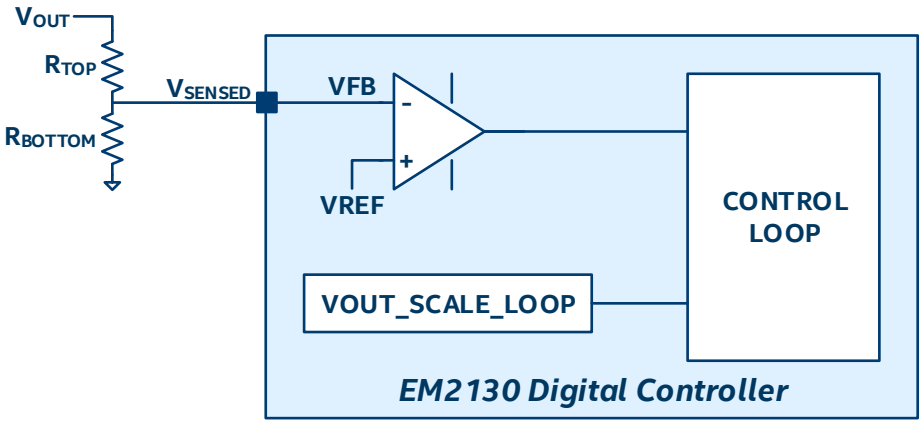


Figure 4: V_{OUT} Scaling Diagram

The VOUT_SCALE_LOOP command may be read and written to and consists of two bytes formatted in Linear Data format (See

The Linear Data Format).

An example VOUT_SCALE_LOOP value when a voltage divider is not used is:

$$BA00h = 1$$

An example scenario where a resistor divider is used with an EM21xx device is for V_{OUT} = 2.4V. In this scenario, the resistor divider is set with a top resistor value of R_{TOP} = 2000Ω and a bottom resistor value of R_{BOTTOM} = 2000Ω, which creates a default input to the controller of 1.2V. This resistor divider ratio is calculated as:

$$Ratio = \frac{R_{TOP}}{R_{TOP} + R_{BOTTOM}} = \frac{2000\Omega}{2000\Omega + 2000\Omega} = 0.5$$

In this example, then, the VOUT_SCALE_LOOP value is calculated as:

$$Ratio * 2^{10} = 512d = B200h$$

4.16 VOUT_SCALE_MONITOR – 2Ah

The VOUT_SCALE_MONITOR command is very similar to the VOUT_SCALE_LOOP command with the exception of being independent of the VOUT_COMMAND. It to the used to map the protection values relating to the output voltage and also the read back values of the output voltage. It is a function of the feedback resistor ratio.

The VOUT_SCALE_MONITOR command also compensates for the gain reduction caused by inclusion of the voltage divider. PMBus commands scaled by the VOUT_SCALE_MONITOR command are shown in Table 7.

An example VOUT_SCALE_MONITOR value and conversion to the real world value when a resistor divider is not used is:

$$BA00h = 512d * 2^{-9} = 1$$

Table 7: PMBus Commands Scaled by VOUT_SCALE_MONITOR

PMBus Command
VOUT_OV_FAULT_LIMIT
VOUT_OV_WARN_LIMIT
VOUT_UV_WARN_LIMIT
VOUT_UV_FAULT_LIMIT
POWER_GOOD_ON
POWER_GOOD_OFF

PMBus Command
READ_VOUT
MFR_VOUT_MARGIN_HIGH
MFR_VOUT_MARGIN_HIGH

4.17 VIN_ON – 35h

The VIN_ON command is used to set the input voltage, in volts, at which the EM21xx device will enable power conversion. The VIN_ON command can be read and written to and consists of two bytes formatted in Linear Data Format (see

The Linear Data Format).

An example VOUT_MARGIN_HIGH value and conversion to the real world value is:

$$CA34h = 564d * 2^{-7} = 4.40625V$$

4.18 VIN_OFF – 36h

The VIN_OFF command is used to set the input voltage, in volts, at which the EM21xx device will disable the power conversion. It can be read and written to and consists of two bytes formatted in Linear Data Format (see

The Linear Data Format).

An example VIN_OFF value and conversion to the real world value is:

$$CA1Ah = 538d * 2^{-7} = 4.20312V$$

4.19 VOUT_OV_FAULT_LIMIT – 40h

The VOUT_OV_FAULT_LIMIT command is used to set the V_{OUT} overvoltage fault limit. When the output voltage exceeds this limit, an overvoltage fault is deemed to have occurred. It can be read and written to and consists of two bytes formatted in Linear Data Format (see

The Linear Data Format).

An example VOUT_OV_FAULT_LIMIT value and conversion to the real world value is:

$$219Ah = 8602d * 2^{-13} = 1.05005V$$

4.20 VOUT_OV_FAULT_RESPONSE – 41h

The VOUT_OV_FAULT_RESPONSE command sets how the device will respond if a V_{OUT} overvoltage fault occurs. It can be read and written to and consists of a single byte consisting as shown in

Table 8.

When a V_{OUT} overvoltage event happens, the following also occurs:

- The HOST is notified through asserting the SMBALERT# signal (output goes low).
- The VOUT bit b[15] in the STATUS_WORD is set.
- The VOUT_OV_FAULT bit b[7] in the STATUS_VOUT register is set.
- The VOUT_OV_FAULT bit b[5] in the STATUS_BYTE is set.

The delay time unit is 0 and the default value is 80h. The output is disabled and there is no response.

Table 8: Voltage, Temperature, and TON_MAX FAULTS Response Data Byte Structure

Bits	Function	Format	Value	Description
[7:6]	Response The Host is notified of the fault through asserting the SMBALERT. The appropriate fault bit in the status registers is set. *(Note 4)*	Read/Write	00	The device continues operating, ignoring the fault.
			01	In the event of a fault, the device continues operating until the delay time specified by bits [2:0] is reached. Upon exceeding this time, the device responds as programmed by the Retry bits [5:3].
			10	The device disables the PWM outputs and responds according to the retry bits [5:3].
			11	The device disables the PWM outputs while the fault is present. When the fault is no longer present, the output is enabled.
[5:3]	Retry Setting	Read/Write	000	The device disables the PWM output/s and does not attempt to restart.
			001-110	The device disables the PWM output/s and attempts to restart a number of times as specified here (minimum of 1 retry, maximum of 6 retries). If the fault is removed during this time, the PWM output/s will remain operating. The time between each restart is defined by bits [2:0]. If the fault condition is still present after the device reaches the programmed number of retries, then the device will no longer attempt to restart.
			111	The device will attempt to restart the PWM outputs indefinitely until it receives a command OFF signal (via the OPERATION command, control pin, or both), device power is removed, or another fault condition disables the PWM outputs.
[2:0]	Delay Time	Read/Write	XXX	Contains the number of delay units (1.5 ms). This value changes both the delay to fault time and delay to retry time.

NOTE: *(4) Once set, the fault notification is only cleared if the bit is individually cleared, a CLEAR_FAULTS command is received, or the output is toggled from off and on (depending on configuration - OPERATION, CONTRO,L or both). It will not be cleared in the event of the fault condition being removed.*

4.21 VOUT_OV_WARN_LIMIT – 42h

The VOUT_OV_WARN_LIMIT command is used to set the Vout overvoltage warning threshold. When the output voltage exceeds this limit an overvoltage warning is deemed to have occurred. It can be read and written to and consists of two bytes, formatted in Linear Data Format (see

The Linear Data Format).

An example VOUT_OV_WARN_LIMIT value and conversion to the real world value is:

$$1ED3h = 7891d * 2^{-13} = 0.963257V$$

When this threshold is exceeded and the a VOUT_OV_WARN_LIMIT is deemed to have occurred the following occurs:

- The HOST is notified through asserting the SMBALERT# signal (output goes low).
- The VOUT bit b[15] in the STATUS_WORD is set.
- The VOUT_OV_WARNING bit b[6] in the STATUS_VOUT register is set.
- The NONE OF THE ABOVE bit b[0] in the STATUS_BYTE is set.

4.22 VOUT_UV_WARN_LIMIT – 43h

The VOUT_UV_WARN_LIMIT command is used to set the V_{OUT} under-voltage warning threshold. When the output voltage falls below this limit an under-voltage warning is deemed to have occurred. It can be read and written to and consists of two bytes, formatted in Linear Data Format (see

The Linear Data Format).

An example VOUT_OV_FAULT_LIMIT value and conversion to the real world value is:

$$1AC7h = 6855d * 2^{-13} = 0.836792V$$

This warning functionality is masked when the PWM output is disabled and also during start-up until V_{OUT} reaches its programmed value.

When this threshold is exceeded and the VOUT_UV_WARN_LIMIT is deemed to have occurred, the following also occurs:

- The HOST is notified as through asserting the SMBALERT# signal (output goes low).
- The VOUT bit [15] in the STATUS_WORD is set.
- The VOUT_UV_WARNING bit b[5] in the STATUS_VOUT register is set.
- The NONE OF THE ABOVE bit b[0] in the STATUS_BYTE is set.

4.23 VOUT_UV_FAULT_LIMIT – 44h

The VOUT_UV_FAULT_LIMIT command is used to set the V_{OUT} under-voltage fault threshold. When the output voltage falls below this limit, an under-voltage fault is deemed to have occurred. The VOUT_UV_FAULT_LIMIT command can be read and written to and consists of two bytes, formatted in Linear Data Format (see

The Linear Data Format).

An example VOUT_UV_FAULT_LIMIT value and conversion to the real world value is:

$$187Bh = 6267d * 2^{-13} = 0.765015V$$

This fault functionality is masked when the PWM output is disabled and also during start-up until V_{OUT} reaches its programmed value.

4.24 VOUT_UV_FAULT_RESPONSE – 45h

The VOUT_UV_FAULT_RESPONSE command sets how the controller will respond in the event of a V_{OUT} under-voltage fault occurring. The VOUT_UV_FAULT_RESPONSE command can be read and written to and consists of a single byte as described in

Table 8.

When the VOUT_UV_FAULT_LIMIT threshold is exceeded and a V_{OUT} under-voltage event is deemed to have occurred, the following also occurs:

- The HOST is modified through asserting the SMBALERT# signal (output goes low).
- The VOUT bit b[15] in the STATUS_WORD is set.
- The VOUT_UV_FAULT bit b[4] in the STATUS_VOUT register is set.
- The NONE OF THE ABOVE bit b[0] in the STATUS_BYTE is set.

The delay time unit is 0 and default value is 80h - the output is disabled and there is no response.

4.25 VIN_OV_FAULT_LIMIT – 55h

The VIN_OV_WARN_LIMIT command is used to set the V_{IN} overvoltage fault threshold. When the input voltage exceeds this limit an input overvoltage fault is deemed to have occurred. The VIN_OV_FAULT_LIMIT command can be read and written to and consists of two bytes, formatted in Linear Data Format (see

The Linear Data Format).

The default value is DA0Eh, which converts to a real world value as:

$$DA0Eh = 576d * 2^{-5} = 16.4375V$$

4.26 VIN_OV_FAULT_RESPONSE – 56h

The VIN_OV_FAULT_RESPONSE command sets how the device will respond in the event an input overvoltage fault occurs. It consists of a single data byte with the structure outlined in

Table 8.

When an input overvoltage fault occurs, the device also performs the following tasks:

- The HOST is notified through asserting the SMBALERT# signal (output goes low).
- The VIN_OV_FAULT bit [7] in the STATUS_INPUT register is set.
- The INPUT bit [5] in the STATUS_WORD is set.
- The NONE OF THE ABOVE bit [0] in the STATUS_BYTE is set.

The delay time unit is 0 and default value is 80h - the output is disabled and there is no response.

4.27 VIN_OV_WARN_LIMIT – 57h

The VIN_OV_WARN_LIMIT command is used to set the V_{IN} overvoltage warning threshold. When the input voltage exceeds this limit, an input overvoltage warning is deemed to have occurred. The VIN_OV_WARN_LIMIT command can be read and written to and consists of two bytes, formatted in Linear Data Format (see

The Linear Data Format).

When this threshold is exceeded and a VIN_OV_WARN_LIMIT is deemed to have occurred, the following also occurs:

- The HOST is modified through asserting the SMBALERT# signal (output goes low).
- The INPUT bit [6] in the STATUS_WORD is set.
- The VIN_OV_WARNING bit [6] in the STATUS_INPUT register is set.
- The NONE OF THE ABOVE bit [0] in the STATUS_BYTE is set.

The default value is D3FFh, which converts to a real world value by:

$$D3FFh = 1023d * 2^{-6} = 15.9844V$$

4.28 VIN_UV_WARN_LIMIT – 58h

The VIN_UV_WARN_LIMIT command is used to set the V_{IN} under voltage warning threshold. When the input voltage exceeds this limit, an input under voltage warning event is deemed to have occurred. The VIN_UV_WARN_LIMIT command can be read and written to and consists of two bytes, formatted in Linear Data Format (see

The Linear Data Format).

This alarm is masked until the input has exceeded the VIN_ON limit and the unit has been enabled.

When the input has fallen below this limit and a VIN_OV_WARN_LIMIT is deemed to have occurred, the following also occurs:

- The HOST is modified through asserting the SMBALERT# signal (output goes low).
- The INPUT bit [6] in the STATUS_WORD is set.
- The VIN_UV_WARNING bit [5] in the STATUS_INPUT register is set.
- The NONE OF THE ABOVE bit [0] in the STATUS_BYTE is set.

The default value is CA29h, which converts to a real world value by:

$$CA29h = 553d * 2^{-7} = 4.32031V$$

4.29 VIN_UV_FAULT_LIMIT – 59h

The VIN_UV_FAULT_LIMIT command is used to set the V_{IN} under-voltage fault threshold. When the input voltage exceeds this limit, an input under-voltage fault is deemed to have occurred. The VIN_UV_FAULT_LIMIT command can be read and written to and consists of two bytes, formatted in Linear Data Format (see

The Linear Data Format).

This alarm is masked until the input has exceeded the VIN_ON limit and the unit has been enabled.

The default value is C3F5h, which converts to a real world value by:

$$C3F5h = 1013d * 2^{-8} = 3.957V$$

4.30 VIN_UV_FAULT_RESPONSE – 5Ah

The VIN_OV_FAULT_RESPONSE command sets how the device will respond in the event an input under voltage fault occurs. It consists of a single data byte of which the structure was outlined previously in

Table 8.

In the event of an input under-voltage fault occurring, the device also performs the following tasks:

- The HOST is notified through asserting the SMBALERT# signal (output goes low).
- The VIN_UV_FAULT bit [4] in the STATUS_INPUT register is set.
- The VIN_UV_FAULT bit [3] in the STATUS_BYTE is set.
- The INPUT bit [5] in the STATUS_WORD is set.

The delay time unit is 0 and default value is 80h - the output is disabled and there is no response.

4.31 POWER_GOOD_ON – 5Eh

The POWER_GOOD_ON command is used to set the output voltage threshold at which the POWER_GOOD signal should be asserted. It can be read and written to and consists of two bytes, formatted in Linear Data Format (see

The Linear Data Format).

An example POWER_COOD_ON value and conversion to the real world value is:

$$1B28h = 6952d * 2^{-13} = 0.848633V$$

4.32 POWER_GOOD_OFF – 5Fh

The POWER_GOOD_OFF command is used to set the output voltage threshold at which the POWER_GOOD signal should be de-asserted. It can be read and written to and consists of two bytes, formatted in Linear Data Format (see

The Linear Data Format).

An example POWER_GOOD_OFF value and conversion to the real world value is:

$$19ECh = 6636d * 2^{-13} = 0.81V$$

4.33 TON_DELAY – 60h

The TON_DELAY command is used to set the time from when the start condition is received until output voltage starts to rise. It can be read and written to and consists of two bytes, formatted in Linear Data Format (see

The Linear Data Format). The start condition is defined by the ON-OFF_CONFIG command. The unit of measure is in milliseconds.

The default value is F800h, which converts to a real world value by:

$$F800h = 0d = 0\ ms$$

4.34 TON_RISE – 61h

The TON_RISE command is used to set the time from when the output voltage starts to rise until the output voltage enters the regulation band. It can be read and written to and consists of two bytes, formatted in Linear Data Format (see

The Linear Data Format). The unit of measure is in milliseconds.

An example TON_RISE value and conversion to the real world value is:

$$CA80h = 640d * 2^{-7} = 5\ ms$$

4.35 TON_MAX_FAULT_LIMIT – 62h

The TON_MAX_FAULT_LIMIT command is used to set the upper limit on the amount of time the device can attempt to power up the output before an under-voltage fault is deemed to have occurred. It can be read and written to and consists of two bytes, formatted in Linear Data Format (see

The Linear Data Format).

A value of 0 milliseconds means that there is no limit and the device can attempt to power up the output indefinitely. The unit of measure is in milliseconds.

An example TON_MAX_FAULT_LIMIT value and conversion to the real world value is:

$$DA80h = 640d * 2^{-5} = 20\ ms$$

4.36 TOFF_DELAY – 64h

The TOFF_DELAY command is used to set the time from when a stop condition is received until the device stops transferring energy to the output. It can be read and written to and consists of two bytes, formatted in Linear Data Format (see

The Linear Data Format). The stop condition is defined by the ON-OFF_CONFIG command. The unit of measure is in milliseconds.

The default value is F800h, which converts to a real world value by:

$$F800h = 0d = 0\ ms$$

4.37 TOFF_FALL – 65h

The TOFF_FALL command is used to set the time from the end of the turn-off delay time until the output voltage is commanded to zero. 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. It can be read and written to and consists of two bytes, formatted in Linear Data Format (see

The Linear Data Format). The unit of measure is in milliseconds.

An example TOFF_FALL value and conversion to the real world value is:

$$DA80h = 640d * 2^{-7} = 5\ ms$$

4.38 TOFF_MAX_WARN_LIMIT – 66h

The TOFF_MAX_WARN_LIMIT command is used to set the upper limit on the amount of time the device can attempt to power down the output without reaching 12.5% of the programmed output voltage at the time the unit is turned off. It can be read and written to and consists of two bytes, formatted in Linear Data Format (see

The Linear Data Format). The unit of measure is in milliseconds.

If the TOFF_MAX_WARN_LIMIT is exceeded, the device performs the following tasks:

- The HOST is notified through asserting the SMBALERT# signal (output goes low).
- The VOUT bit [7] in the STATUS_WORD register is set.
- The TOFF_MAX Warning bit [1] in the STATUS_VOUT register is set.

- The NONE OF THE ABOVE bit [0] in the STATUS_BYTE is set.

A value of 0 ms means there is no limit and the unit waits can wait for the output voltage to decay indefinitely. The unit of measure is in milliseconds.

An example TOFF_MAX_WARN_LIMIT value and conversion to the real world value is:

$$E320h = 800d * 2^{-4} = 50\,ms$$

4.39 STATUS_BYTE – 78h

The STATUS_BYTE command returns a summary of the most critical faults in one byte and is read only.

Table 9: STATUS_BYTE Data Byte Structure

STATUS_BYTE (Read Only)		
Bits	**Name**	**Description**
[0]	NONE OF THE ABOVE	A fault or warning not listed in bits [7:1] has occurred.
[1]	CML	A communication fault has occurred.
[2]	TEMPERATURE	A temperature fault or warning has occurred.
[3]	VIN_UV_FAULT	An input under-voltage fault has occurred.
[4]	IOUT_OC_FAULT	An output over-current fault has occurred.
[5]	VOUT_OV_FAULT	An output over-voltage fault has occurred.
[6]	OFF	This bit is asserted if the device is not providing power to the output, regardless of the reason, including if the device is simply not enabled.
[7]	NOT SUPPORTED	Not supported.

For each individual bit, a value of 1 indicates a warning or fault event has occurred, while a 0 indicates that no warning or fault event has occurred.

4.40 STATUS_WORD – 79h

The STATUS_WORD command returns a summary of the device status information in two data bytes and is read only. The lower byte of the STATUS_WORD is the same as the STATUS_BYTE above.

Table 10: STATUS_WORD Data Byte Structure

STATUS_WORD (Read Only)		
Bits	**Name**	**Description**
[7:0]	STATUS_BYTE	See STATUS_BYTE (Table 9).
[8]	NOT SUPPORTED	Not supported.
[9]	NOT SUPPORTED	Not supported.
[10]	NOT SUPPORTED	No supported.
[11]	POWER_GOOD#	The POWER_GOOD signal is not valid.
[12]	MFR_SPECIFIC	A manufacturer specific fault or warning has occurred.
[13]	INPUT	An input voltage warning or fault has occurred. (I_{IN}, P_{IN} not supported)
[14]	IOUT/POUT	An output current warning or fault has occurred. (P_{OUT} not supported)
[15]	VOUT	An output voltage related warning or fault has occurred.

For each individual bit, a value of 1 indicates a warning or fault event has occurred, while a 0 indicates that no warning or fault event has occurred.

4.41 STATUS_VOUT – 7Ah

The STATUS_VOUT command returns a summary of the V_{OUT} status information in one data byte and is read only.

Table 11: STATUS_VOUT Data Byte Structure

STATUS_VOUT (Read Only)		
Bits	**Name**	**Description**
[0]	NOT SUPPORTED	Not supported.
[1]	NOT SUPPORTED	Not supported.
[2]	NOT SUPPORTED	Not supported.
[3]	NOT SUPPORTED	Not supported.
[4]	VOUT_UV_FAULT	An output voltage under-voltage fault has occurred.
[5]	VOUT_UV_WARN	An output voltage under-voltage warning has occurred.
[6]	VOUT_OV_WARN	An output voltage over-voltage warning has occurred.
[7]	VOUT_OV_FAULT	An output voltage over-voltage fault has occurred.

For each individual bit, a value of 1 indicates a warning or fault event has occurred, while a 0 indicates that no warning or fault event has occurred.

4.42 STATUS_IOUT – 7Bh

The STATUS_IOUT command returns a summary of the I_{OUT} status information in one data byte and is read only.

Table 12: STATUS_IOUT Data Byte Structure

STATUS_IOUT (Read Only)		
Bits	**Name**	**Description**
[0]	NOT SUPPORTED	Not supported.
[1]	NOT SUPPORTED	Not supported.
[2]	NOT SUPPORTED	Not supported.
[3]	NOT SUPPORTED	Not supported.
[4]	IOUT_UC_FAULT	An under-current warning has occurred.
[5]	IOUT_OC_WARNING	An over-current warning has occurred.
[6]	ICOUT_OC_LV_FAULT	An over-current low-voltage shutdown fault has occurred.
[7]	IOUT_OC_FAULT	An over-current fault has occurred.

4.43 STATUS_INPUT – 7Ch

The STATUS_INPUT command returns a summary of the voltage input status information in one data byte and is read only.

Table 13: STATUS_INPUT Data Byte Structure

STATUS_INPUT (Read Only)		
Bits	**Name**	**Description**
[0]	NOT SUPPORTED	Not supported.
[1]	NOT SUPPORTED	Not supported.
[2]	NOT SUPPORTED	Not supported.
[3]	DEVICE OFF FOR INSUFFICIENT INPUT VOLTAGE	The input voltage has not risen above the input turn-on threshold or, in if the device has started, the input voltage has fallen below the turn-off threshold.
[4]	VIN_UV_FAULT	An input voltage under-voltage fault has occurred.
[5]	VIN_UV_WARNING	An input voltage under-voltage warning has occurred.
[6]	VIN_OV_WARNING	An input voltage over-voltage warning has occurred.
[7]	VIN_OV_FAULT	An input voltage over-voltage fault has occurred.

For each individual bit, a value of 1 indicates a warning or fault event has occurred, while a 0 indicates that no warning or fault event has occurred. For bits that are not supported, a value of 0 will be returned when read.

4.44 STATUS_CML – 7Eh

The STATUS_CML command returns a summary of the communications status information in one data byte and is read only.

Table 14: STATUS_CML Data Byte Structure

STATUS_CML (Read Only)		
Bits	**Name**	**Description**
[0]	NOT SUPPORTED	Not supported.
[1]	SMBUS_FAULT	SMBus timeout or a format error have occurred.
[2]	RESERVED	PMBus reserved.
[3]	NOT SUPPORTED	Not supported.
[4]	NOT SUPPORTED	Not supported.
[5]	PEC_FAULT	A packet error check fault has occurred.
[6]	NOT SUPPORTED	Not supported.
[7]	CMD_FAULT	An invalid or an unsupported command has been received.

4.45 STATUS_MFR_SPECIFIC – 80h

The STATUS_MFR_SPECIFIC command returns a summary of the temperature status information in one data byte and is read only.

Table 15: STATUS_MFR_SPECIFIC Data Byte Structure

STATUS_MFR_SPECIFIC (Read Only)		
Bits	**Name**	**Description**
[0]	UNUSED	Not supported.
[1]	UNUSED	Not supported.
[2]	UNUSED	Not supported.
[3]	UNUSED	Not supported.
[4]	UNUSED	Not supported.
[5]	UNUSED	Not supported.
[6]	TEMP_OV_WARN	An over-temperature warning has occurred.
[7]	TEMP_OV_FAULT	An over-temperature fault has occurred.

4.46 READ_VIN – 88h

The READ_VIN command is used to read the input voltage in volts. It is read only and consists of two bytes formatted in Linear Data Format (see

The Linear Data Format).

Table 16: READ_VIN Data Byte Structure

READ_VIN (Read Only)		
Bits	**Name**	**Description**
[15:0]	VIN	Input voltage in V (linear data format).

An example READ_VIN read-back value and conversion to the real world value is:

$$0xD2FCh = 768d * 2^{-6} = 12V$$

4.47 READ_VOUT – 8Bh

The READ_VOUT command is used to read the output voltage in volts. It is read only and consists of two bytes formatted in Linear Data Format (see

The Linear Data Format).

Table 17: READ_VOUT Data Byte Structure

READ_VOUT (Read Only)		
Bits	**Name**	**Description**
[15:0]	VOUT	Output voltage in V (linear data format). Note that this command is mantissa only.

An example READ_VOUT read-back value and conversion to the real world value is:

$$0x2666h = 9830d * 2^{-13} = 1.2V$$

4.48 READ_IOUT – 8Ch

The READ_IOUT command is used to read the output current in amperes. It is read only and consists of two bytes formatted in Linear Data Format (see

The Linear Data Format).

Table 18: READ_IOUT Data Byte Structure

READ_IOUT (Read Only)		
Bits	**Name**	**Description**
[15:0]	IOUT	Output current in A (linear data format).

4.49 READ_TEMPERATURE – 8Eh

The READ_TEMPERATURE command is used to return the temperature read back from the device. The temperature is returned in degrees Celsius, is read only, and consists of two bytes formatted in Linear Data Format (see

The Linear Data Format).

Table 19: READ_TEMPERATURE Data Byte Structure

READ_TEMPERATURE (Read Only)		
Bits	**Name**	**Description**
[15:0]	TEMP	Internal temperature in °C (linear data format).

4.50 READ_DUTY_CYCLE – 94h

The READ_DUTY_CYCLE command is used to return the current duty cycle value. The duty cycle is returned in percent, is read only and consists of two bytes formatted in Linear Data Format (see

The Linear Data Format).

Table 20: READ_DUTY_CYCLE Data Byte Structure

READ_DUTY_CYCLE (Read Only)		
Bits	**Name**	**Description**
[15:0]	DUTY CYCLE	Duty cycle of the device, in percent (linear data format).

4.51 READ_FREQUENCY – 95h

The READ_FREQUENCY command is used to return the switching frequency of the device. The frequency returned is in kilohertz, is read only, and consists of two bytes formatted in Linear Data Format (see

The Linear Data Format).

Table 21: READ_FREQUENCY Data Byte Structure

READ_FREQUENCY (Read Only)		
Bits	**Name**	**Description**
[15:0]	FREQUENCY	Switching frequency in kHz (linear data format).

4.52 PMBUS_REVISION – 98h

The PMBUS_REVISION command is used to return the version of PMBus that the device supports. This command is read only and consists of one byte. The EM21xx family supports PMBus version 1.2.

Table 22: PMBUS_REVISION Data Byte Structure

PMBUS_REVISION (Read Only)		
Bits	**Name**	**Description**
[7:4]	0010	PMBUS PART I REVISION 1.2
[3:0]	0010	PMBUS PART II REVISION 1.2

4.53 MFR_ID – 99h

The MFR_ID command returns the text (ISO/IEC 8859-1[A05]) characters that contain the manufacturer's ID "INTL". This command is read only and consists of four bytes.

Table 23: MFR_ID Data Byte Structure

MFR_ID (Read Only)	
Bits	**Description**
[7:0]	0x49 = "I"
[15:8]	0x4E = "N"

[23:16]	0x54 = “T”
[31:24]	0x4C = “L”

4.54 MFR_MODEL – 9Ah

The MFR_MODEL command returns the text (ISO/IEC 8859-1[A05]) characters that contain the manufacturer’s model number. This command is read only and consists of four bytes. The example shown in

Table 24 is for the manufacturer model number “2130L.”

Table 24: MFR_MODEL Data Byte Structure

MFR_MODEL (Read Only)	
Bits	**Description**
[7:0]	0x32 = “2”
[15:8]	0x31 = “1”
[23:16]	0x33 = “3”
[31:24]	0x30 = “0”
[39:32]	0x4C = “L”
[47:40]	0x00 = “”

4.55 MFR_REVISION – 9Bh

The MFR_REVISION command returns the text (ISO/IEC 8859-1[A05]) characters that contain the manufacturer’s revision number. This command is read only and consists of four bytes. The example shown in Table 25 is for the F/W revision 00.94.26.

Table 25: MFR_REVISION Data Byte Structure

MFR_REVISION (Read Only)	
Bits	**Description**
[7:0]	“0” = 0x30
[15:8]	“0” = 0x30
[23:16]	“.” = 0x2E
[31:24]	“9” = 0x39
[39:32]	“4” = 0x34
[47:40]	“.” = 0x2E
[55:48]	“2” = 0x32
[63:56]	“6” = 0x36

4.56 MFR_SERIAL – 9Eh

The MFR_SERIAL command returns the text (ISO/IEC 8859-1[A05]) characters that uniquely identifies the device. This command is read only and consists of twelve bytes, containing the unique serial number of the device.

Table 26: MFR_SERIAL Data Byte Structure

MFR_SERIAL (Read Only)	
Bits	**Description**
[31:0]	LOT 1 Example: 0x31,0x32,0x33,0x34 = “1234”
[47:32]	LOT 2 Example: 0x39,0x41 = “9A”
[63:48]	WAFER-NUMBER Example: 0x42,0x43 = “BC”
[79:64]	X-POSITION Example: 0x37,0x44 = “7D”
[95:80]	Y-POSITION Example: 0x46,0x35 = “F5”

4.57 MFR_VIN_MIN – A0h

The MFR_VIN_MIN command reads back the minimum rated input voltage in volts. This command consists of two bytes, in linear format.

Table 27: MFR_VIN_MIN Data Byte Structure

MFR_VIN_MIN (Read Only)		
Bits	**Name**	**Description**
[15:0]	MFR_VIN_MIN	Input voltage in volts (linear data format).

4.58 MFR_VOUT_MIN – A4h

The MFR_VOUT_MIN command reads back the minimum rated output voltage in volts. This command consists of two bytes, in linear format.

Table 28: MFR_VIN_MIN Data Byte Structure

MFR_VIN_MIN (Read Only)		
Bits	**Name**	**Description**
[15:0]	MFR_VIN_MIN	Output voltage in volts (linear data format).

4.59 MFR_SPECIFIC_00 – D0h

The MFR_SPECIFIC_00 command is available for use as required. This command consists of two bytes and can be read back **but can only be written to once**.

Table 29: MFR_SPECIFIC_00 Data Byte Structure

MFR_SPECIFIC_00 (Read/Write)		
Bits	**Name**	**Description**
[15:0]	MFR_SPECIFIC_00	MFR defined

4.60 MFR_SPECIFIC_01 – D1h

The MFR_SPECIFIC_01 command is available for use as required. This command consists of six bytes and can be read and written to.

The default value is 0x000000.

Table 30: MFR_SPECIFIC_01 Data Byte Structure

MFR_SPECIFIC_01 (Read/Write)		
Bits	**Name**	**Description**
[47:0]	MFR_SPECIFIC_01	MFR defined

4.61 MFR_READ_VCC – D2h

The MFR_READ_VCC PMBus command returns the device V_{CC} voltage, in volts. This command consists of two bytes in linear format and is read only.

Table 31: MFR_READ_VCC Data Byte Structure

MFR_READ_VCC (Read Only)		
Bits	**Name**	**Description**
[15:0]	MFR_READ_VCC	VCC voltage in volts (linear data format).

4.62 MFR_RESYNC – D3h

The MFR_RESYNC PMBus command causes the device to activate the SYNC pin for 1 ms to 1.5 ms in order to re-acquire the SYNC pin input signal.

Table 32: MFR_RESYNC Data Byte Structure

MFR_RESYNC (Write Only)		
Bits	**Name**	**Description**
No data.		

4.63 MFR_RTUNE_CONFIG – DAh

The MFR_RTUNE_CONFIG PMBus command gets/sets the RTUNE compensator index and scaling factor.

Table 33: MFR_RTUNE_CONFIG Data Byte Structure

MFR_RTUNE_CONFIG (Read/Write)		
Bits	**Name**	**Description**
[3:0]	RTUNE_INDEX	RTUNE compensator index (0-3).
[15:4]	RTUNE_SCALE	RTUNE compensator scaling factor. Format: unsigned Q4.8

4.64 MFR_RTUNE_INDEX – DDh

The MFR_RTUNE_INDEX PMBus command returns the index derived from the resistor strapped to the RTUNE pin of the device. Note that the index will depend upon whether the E12 or E96 resistor series has been configured for RTUNE pin-strapping. This command consists of one byte and is read only.

Table 34: MFR_RTUNE_INDEX Data Byte Structure

MFR_RTUNE_INDEX (Read Only)		
Bits	**Name**	**Description**
[7:0]	MFR_RTUNE_INDEX	RTUNE index

4.65 MFR_RVSET_INDEX – DEh

The MFR_RVSET_INDEX PMBus command returns the index derived from the resistor strapped to the RVSET pin of the device. Note that the index will depend upon whether the E12 or E96 resistor series has been configured for RVSET pin-strapping. This command consists of one byte and is read only.

Table 35: MFR_RVSET_INDEX Data Byte Structure

MFR_RVSET_INDEX (Read Only)		
Bits	**Name**	**Description**
[7:0]	MFR_RVSET_INDEX	RVSET index

4.66 MFR_VOUT_OFF – E0h

The MFR_VOUT_OFF command gets/sets the target output voltage when switching off the device, in volts. Setting a non-zero value here will enable shut-down into a pre-bias condition. This command consists of two bytes, in linear format.

Table 36: MFR_VOUT_OFF Data Byte Structure

MFR_VOUT_OFF (Read/Write)		
Bits	**Name**	**Description**
[15:0]	MFR_VOUT_OFF	Target output voltage when switching off the device in volts (linear data format).

4.67 MFR_OT_FAULT_LIMIT – E2h

The MFR_OT_FAULT_LIMIT command is used to set the over temperature fault threshold. When the temperature exceeds this limit, an over temperature fault is deemed to have occurred. The MFR_OT_FAULT_LIMIT command can be read and written to and consists of two bytes, formatted in Linear Data Format (see

The Linear Data Format). The temperature read-back value is rounded to the nearest 0.125°C value.

The default value is EBBFh, which converts to a real world value by:

$$EBBFh = 959d * 2^{-3} = 119.875℃$$

4.68 MFR_OT_WARN_LIMIT – E3h

The MFR_OT_WARN_LIMIT command is used to set the over temperature warn threshold. When the temperature exceeds this limit, an over temperature warning is deemed to have occurred. The MFR_OT_WARN_LIMIT command can be read and written to and consists of two bytes, formatted in Linear Data Format (see

The Linear Data Format).

The default value is EB6Fh, which converts to a real world value by:

$$EB6Fh = 879d * 2^{-3} = 109.875℃$$

4.69 MFR_OT_FAULT_RESPONSE – E5h

The MFR_OT_FAULT_RESPONSE command sets how the device will respond in the event of an over-temperature fault occurring. It consists of a single data byte, the structure of which is outlined in

Table 8. In addition to detecting that an OT_FAULT_LIMIT FAULT has occurred, the device also performs the following tasks:

- The HOST is notified through asserting the SMBALERT# signal (output goes low).
- The OT_FAULT bit [7] in the STATUS_TEMPERATURE register is set.
- The TEMPERATURE bit [2] in the STATUS_BYTE is set.

The delay time unit is 0 and default value is B8h. The output is disabled and attempts to restart continuously once temperature drops below MFR_TEMP_ON level.

4.70 MFR_TEMP_ON – E6h

The MFR_TEMP_ON command is used to set the over temperature turn on threshold. This is effectively the lower hysteresis value for an over temperature fault. When the temperature falls below this limit after a temperature fault has occurred, the device will restart automatically. The MFR_TEMP_ON command can be read and written to and consists of two bytes, formatted in Linear Data Format (see

The Linear Data Format).

The default value is EB2Fh, which converts to a real world value by:

$$EB2Fh = 815d * 2^{-3} = 101.875℃$$

4.71 MFR_PIN_CONFIG – E7h

The MFR_PIN_CONFIG PMBus command configures the behavior of various pins of the device. The behavior of the ADDR0, ADDR1, RTUNE, RVSET, VTRACK, and SYNC pins can be configured via this command. This command can be used to read and write to the part and consists of two bytes.

Table 37: MFR_PIN_CONFIG Data Byte Structure

MFR_PIN_CONFIG (Read/Write)		
Bits	**Name**	**Description**
[0:3]	Reserved	Reserved
[4]	COMP_STRAP_EN	0 = Disabled 1 = Enabled
[5]	VOUT_STRAP_EN	0 = Disabled 1 = Enabled
[6]	COMP_SELECT_EN	0 = Disabled 1 = Enabled
[7]	Reserved	Reserved
[8]	VTRACK Enable	0 = Disabled 1 = Enabled
[9]	SYNC Enable	0 = Disabled 1 = Enabled
[10]	SYNC IN/OUT	0 = Sync signal input 1 = Sync signal output
[11]	SYNC Edge	0 = Rising edge 1 = Falling edge
[15:12]	Reserved	Reserved

Any values other than those defined in

Table 37 will be treated as invalid, ignored, and a communications fault will be declared. The device will set the CML [1] bit of the STATUS_BYTE register, set the INVALID_COMMAND [7] of the STATUS_CML register, and notify the host through asserting the SMBALERT# signal (output goes low).

The default value is 003Ch. By default, pin strapping via via RVSET and RTUNE is enabled.

4.72 MFR_STORE_CONFIG_ADDR_READ – E9h

The MFR_STORE_CONFIG_ADDR_READ PMBus command can be used to verify the configuration value programmed via the MFR_STORE_CONFIG_ADDR_WRITE PMBus command.

CAUTION: This command only functions within the programming routine (i.e. between the Start (ECh) and end (EEh) commands).

The command is implemented as a PMBus process call, taking the address as an input (host word write) and providing the value as an output (host word read).

This command must be used in conjunction with the MFR_STORE_CONFIG_ADDR_WRITE command. It must be issued after the MFR_STORE_CONFIG_ADDR_WRITE command that programmed the configuration value to be verified.

Table 38: MFR_STORE_CONFIG_ADDR_READ Data Byte Structure

MFR_STORE_CONFIG_ADDR_READ (Write)		
Bits	**Name**	**Description**
[15:0]	ADDRESS	Configuration address
MFR_STORE_CONFIG_ADDR_READ (Read)		
Bits	**Name**	**Description**
[15:0]	DATA	Configuration value

As an example, to read the data value 0x647A from OTP address 0x100A, the necessary I2C byte sequence would be:

Write: {0xED 0x0A 0x10} + Read: {0x7A 0x64}

CAUTION: If the value read does not match the value programmed, then an unrecoverable programming error has occurred and the reconfiguration has failed.

4.73 MFR_STORE_PARAMS_REMAINING – EAh

The MFR_STORE_PARAMS_REMAINING PMBus command returns the number of remaining times that a STORE_DEFAULT_ALL command can be issued to store all PMBus parameters. Each time the STORE_DEFAULT_ALL command is issued, a portion of the available space in OTP is permanently used up.

Table 39: MFR_STORE_PARAMS_REMAINING Data Byte Structure

MFR_STORE_PARAMS_REMAINING (Read Only)		
Bits	**Name**	**Description**
[15:0]	DATA	The number of remaining times the STORE_DEFAULT_ALL commands can be successfully issued, in linear format.

4.74 MFR_STORE_CONFIGS_REMAINING – EBh

The MFR_STORE_CONFIGS_REMAIN PMBus command returns the number of remaining times that a full configuration can be stored via the MFR_STORE_CONFIG_BEGIN, MFR_STORE_CONFIG_ADDR_WRITE, and MFR_STORE_CONFIG_END commands. Each time a full configuration is stored, a portion of the available space in OTP is permanently used up.

In this case, the magnitude of the return value indicates the number of times a configuration could be subsequently programmed once the device is configured.

Table 40: MFR_STORE_CONFIGS_REMAINING Data Byte Structure

MFR_STORE_CONFIGS_REMAINING (Read Only)		
Bits	**Name**	**Description**
[15:0]	CONFIGS_COUNT	The number of remaining times a full configuration can be successfully stored, in linear format. A negative result indicates that the device has not yet been configured OTP.

4.75 MFR_STORE_CONFIG_BEGIN – ECh

The MFR_STORE_CONFIG_BEGIN PMBus command is used to commence programming of configuration to OTP. This command must be used in conjunction with the MFR_STORE_CONFIG_ADDR_DATA and MFR_STORE_CONFIG_END commands. It **MUST** be issued prior to sending any MFR_STORE_CONFIG_ADDR_DATA commands.

Table 41: MFR_STORE_CONFIG_BEGIN Command Structure

MFR_STORE_CONFIG_BEGIN (Write Only)		
Bits	**Name**	**Description**
No data.		

4.76 MFR_STORE_CONFIG_ADDR_WRITE – EDh

This command must be used in conjunction with the MFR_STORE_CONFIG_BEGIN and MFR_STORE_CONFIG_END commands. It must be sent only after issuing a MFR_STORE_CONFIG_BEGIN command and prior to issuing a MFR_STORE_CONFIG_END command.

Table 42: MFR_STORE_CONFIG_ADDR_WRITE Command Structure

MFR_STORE_CONFIG_ADDR_WRITE (Write Only)		
Bits	**Name**	**Description**
[15:0]	ADDRESS	Configuration address
[31:16]	DATA	Configuration value

As an example, to read the data value 0x647A from OTP address 0x100A, the necessary I2C byte sequence would be:

$$Write: \{0xED\ 0x0A\ 0x10\ 0x7A\ 0x64\}$$

The following is an example program reconfiguration sequence flow when using an EM21xx device:

MFR_STORE_CONFIG_BEGIN

MFR_STORE_CONFIG_ADDR_WRITE (ADDRESS=0x0C00, DATA=0x....)

MFR_STORE_CONFIG_ADDR_WRITE (ADDRESS=0x0C01, DATA=0x....)

…

MFR_STORE_CONFIG_ADDR_WRITE (ADDRESS=0x0C33, DATA=0x....)

MFR_STORE_CONFIG_ADDR_WRITE (ADDRESS=0x0C34, DATA=0x....)

MFR_STORE_CONFIG_ADDR_WRITE (ADDRESS=0x11B8, DATA=0x....)

MFR_STORE_CONFIG_ADDR_WRITE (ADDRESS=0x11B9, DATA=0x....)

…

MFR_STORE_CONFIG_ADDR_WRITE (ADDRESS=0x11F2, DATA=0x....)

MFR_STORE_CONFIG_ADDR_WRITE (ADDRESS=0x11F3, DATA=0x....)

MFR_STORE_CONFIG_END

The MFR_STORE_CONFIG_ADDR_WRITE PMBus command is used to program a configuration value (identified by address and data) to OTP. A sequence of these commands can be used to efficiently program an entire configuration to OTP.

The address/data pair corresponds directly to one line in a configuration ROM file. Addresses must be programmed in strict ascending order, as they appear in a configuration ROM file.

4.77 MFR_STORE_CONFIG_END – EEh

The MFR_STORE_CONFIG_END PMBus command is used to complete programming of configuration to OTP. This command must be used in conjunction with the MFR_STORE_CONFIG_BEGIN and MFR_STORE_CONFIG_ADDR_DATA commands. It MUST be issued after sending all required MFR_STORE_CONFIG_ADDR_DATA commands.

Table 43: MFR_STORE_CONFIG_END Command Structure

MFR_STORE_CONFIG_END (Write Only)		
Bits	**Name**	**Description**
No data.		

5. Additional Error Checking On PMBus Commands Sent To EM21xx Device

The EM21xx firmware has some error checking logic. If the user attempts to write an inappropriate command value, the device will ignore the new value and report a CML error.

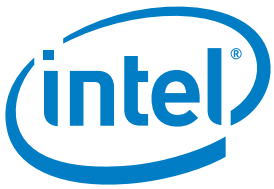
For example, consider a case where the current values for VIN_ON and VIN_OFF are:

- VIN_ON = 4.4V (CA33h)
- VIN_OFF = 4.15V (CA13h)

If a user attempts to program VIN_OFF to a value greater than VIN_ON, such as attempting to write VIN_OFF = 4.5V (CA40h), then the device will ignore the new value and assert the CML.

Other commands where this logic also applies to ensure that:

- VIN_ON ≥ VIN_OFF
- VOUT_OV_FAULT_LIMIT ≥ VOUT_OV_WARN_LIMIT
- VOUT_UV_WARN_LIMIT ≥ VOUT_UV_FAULT_LIMIT
- IOUT_OC_FAULT_LIMIT ≥ IOUT_OC_WARN_LIMIT
- OT_FAULT_LIMIT ≥ OT_WARN_LIMIT
- OT_FAULT_LIMIT ≥ MFR_TEMP_ON
- VIN_OV_FAULT_LIMIT ≥ VIN_OV_WARN_LIMIT
- VIN_UV_WARN_LIMIT ≥ VIN_UV_FAULT_LIMIT
- POWER_GOOD_ON ≥ POWER_GOOD_OFF
- MFR_OT_FAULT_LIMIT ≥ MFR_OT_WARN_LIMIT
- VIN_OV_FAULT_LIMIT ≥ VIN_UV_FAULT_LIMIT
- VOUT_OV_FAULT_LIMIT ≥ VOUT_UV_FAULT_LIMIT



6. Revision History

Revision Number	Description	Revision Date
001	Initial release.	March 2017