

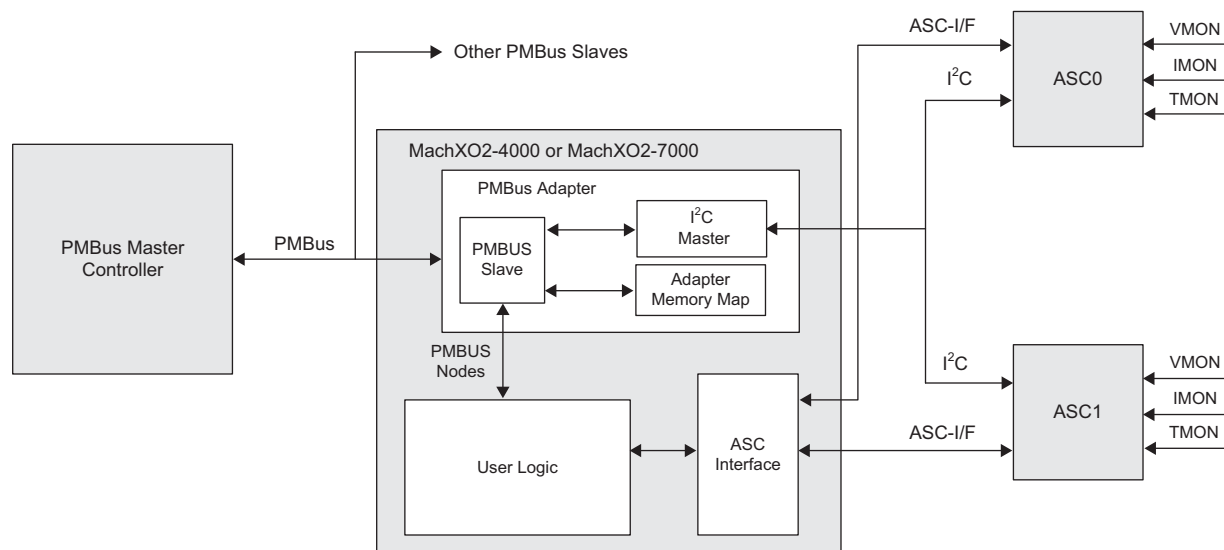
Introduction

The Platform Manager™ 2 is a fast-reacting, programmable logic based hardware management controller. Platform Manager 2 is an integrated solution combining analog sense and control elements with scalable programmable logic resources. This integrated approach allows the Platform Manager 2 to address Power Management (Power Sequencing, Voltage Monitoring, Trimming and Margining), Thermal Management (Temperature Monitoring, Fan Control, Power Control), and Control Plane Functions (System Configuration, I/O Expansion, Reset Distribution, and so forth) as a single device.

This technical note focuses on the Power Management Bus (PMBus) slave support in Platform Manager 2 using the PMBus Adapter component. The PMBus Adapter supports the connection of the MachXO2 FPGA as a PMBus slave to an external PMBus master controller. The PMBus Adapter supports a subset of commands from the PMBus specification as shown in Table 1 below. Implementing the PMBus Adapter component requires a system based on either a MachXO2-4000 or MachXO2-7000 device and at least one ASC hardware management expander.

The PMBus Adapter is configured and enabled using the Platform Designer tool, a part of Lattice Diamond software. Many of the commands supported by the PMBus Adapter require additional user logic to process. This technical note describes the steps required to properly implement a complete PMBus slave solution. An example of a system built around the PMBus Adapter is shown in Figure 1.

Figure 1. Example System Based on PMBus Adapter



PMBus Command Description

PMBus is an industry standard communication protocol for power conversion devices. It is based on the SMBus serial interface, which is built upon I²C with some extensions. In general, the PMBus Adapter can be used with any I²C master device with limited updates to firmware. More information about PMBus and SMBus can be found at:

- www.pmbus.org
- www.smbus.org

PMBus supports optional features like Packet Error Checking (PEC) and SMBAlert. The PMBus Adapter supports configuration of these optional features.

Figure 2 shows command types supported by the PMBus Adapter, as defined by the SMBus protocol. The format key for the instructions is below.

PMBus Instruction Format Key

- S – Start Condition
- Slave Address – Address[6:0] of PMBus Adapter
- W – Write Bit (Logic 0)
- A – Acknowledge Bit (Logic 0)
- A – Not Acknowledge Bit (Logic 1)
- Sr – Repeated Start Condition
- R – Read Bit (Logic 1)
- P – Stop Condition
- Command Code – Command[7:0] as defined by PMBus Command Code (See Table 1)
- Data Byte High / Data Byte Low – Data[15:8] / Data[7:0] as defined by individual command descriptions
- PEC – Packet Error Check Code as defined by PMBus specification
- Shaded Bits – Bits asserted by the PMBus Adapter

Figure 2. Supported PMBus Command Types

SEND BYTE	S	SLAVE ADDRESS	W	A	COMMAND CODE	A	P												
SEND BYTE w/ PEC	S	SLAVE ADDRESS	W	A	COMMAND CODE	A		PEC	A	P									
WRITE BYTE	S	SLAVE ADDRESS	W	A	COMMAND CODE	A		DATA BYTE	A	P									
WRITE BYTE w/ PEC	S	SLAVE ADDRESS	W	A	COMMAND CODE	A		DATA BYTE	A		PEC	A	P						
WRITE WORD	S	SLAVE ADDRESS	W	A	COMMAND CODE	A		DATA BYTE LOW	A		DATA BYTE HIGH	A	P						
WRITE WORD w/ PEC	S	SLAVE ADDRESS	W	A	COMMAND CODE	A		DATA BYTE LOW	A		DATA BYTE HIGH	A		PEC	A	P			
READ BYTE	S	SLAVE ADDRESS	W	A	COMMAND CODE	A	Sr	SLAVE ADDRESS	R	A	DATA BYTE	NA	P						
READ BYTE w/ PEC	S	SLAVE ADDRESS	W	A	COMMAND CODE	A	Sr	SLAVE ADDRESS	R	A	DATA BYTE	A		PEC	NA	P			
READ WORD	S	SLAVE ADDRESS	W	A	COMMAND CODE	A	Sr	SLAVE ADDRESS	R	A	DATA BYTE LOW	A		DATA BYTE HIGH	NA	P			
READ WORD w/ PEC	S	SLAVE ADDRESS	W	A	COMMAND CODE	A	Sr	SLAVE ADDRESS	R	A	DATA BYTE LOW	A		DATA BYTE HIGH	A		PEC	NA	P

The PMBus Adapter supports a subset of the overall PMBus commands. Table 1 summarizes the supported commands. The PMBus Adapter supports connection to up to 48 voltage monitors, 16 current sense monitors, and 32 temperature monitors. Each of these monitors is assigned to a unique page as detailed in the PAGE command section. Table 1 shows which commands are page based and which commands are global (same response regardless of page setting). Some commands are processed directly by the PMBus Adapter (such as READ_VOUT) meaning no user logic is required to support the command. Other commands require user logic for processing (such as OPERATION). This means that user logic must assign or provide the signals set or requested by the command. Some commands are processed both directly by the PMBus Adapter and user logic as well. These commands have some behavior processed by the PMBus Adapter (like CML bit in STATUS_BYTE), while others require user logic (all other bits in STATUS_BYTE). In the PMBUS specifications, commands are either read-only or read/write. This command information is captured in the TYPE column in Table 1. Each command is described in more detail later in the document.

Table 1. Commands Supported by PMBus Adapter

COMMAND CODE	COMMAND NAME	DESCRIPTION	TYPE	PAGE RANGE	PROCESSING
0x00	PAGE	Active monitor (page) for commands which support paging	R/W Byte	0x00-0x5F	PMBus Adapter
0x01	OPERATION	Manages On / Off / Margining behavior of power supplies	R/W Byte	N/A	User Logic
0x03	CLEAR_FAULTS	Clears all fault bits, deassert SMBALERT#	Send Byte	N/A	User Logic ¹
0x10	WRITE_PROTECT	Sets protection level of PMBus Adapter	R/W Byte	N/A	PMBus Adapter
0x19	CAPABILITY	Information about optional PMBus features enabled	Read Byte	N/A	PMBus Adapter
0x40	VOUT_OV_FAULT_LIMIT	Overvoltage fault limit of active page (VMONx_A threshold)	R/W Word	0x00-0x2F	PMBus Adapter
0x44	VOUT_UV_FAULT_LIMIT	Undervoltage fault limit of active page (VMONx_B threshold)	R/W Word	0x00-0x2F	PMBus Adapter
0x46	IOUT_OC_FAULT_LIMIT	Overcurrent fault limit of active page (IMONx_A threshold)	R/W Word	0x30-0x3F	PMBus Adapter
0x4B	IOUT_UC_FAULT_LIMIT	Undercurrent fault limit of active page (IMONx_B threshold)	R/W Word	0x30-0x3F	PMBus Adapter
0x4F	OT_FAULT_LIMIT	Overtemperature fault limit of active page (TMONx_A threshold)	R/W Word	0x40-0x5F	PMBus Adapter
0x53	UT_FAULT_LIMIT	Overtemperature fault limit of active page (TMONx_B threshold)	R/W Word	0x40-0x5F	PMBus Adapter
0x78	STATUS_BYTE	One byte fault status summary	Read Byte	N/A	User Logic ¹
0x79	STATUS_WORD	Two byte fault status summary	Read Word	N/A	User Logic ¹
0x7A	STATUS_VOUT	Output voltage fault and warning status	Read Byte	N/A	User Logic
0x7B	STATUS_IOUT	Output current fault and warning status	Read Byte	N/A	User Logic
0x7C	STATUS_INPUT	Input voltage, current, power fault and warning status	Read Byte	N/A	User Logic
0x7D	STATUS_TEMPERATURE	Temperature fault and warning status	Read Byte	N/A	User Logic
0x7E	STATUS_CML	Communication fault status	Read Byte	N/A	PMBus Adapter
0x7F	STATUS_OTHER	Fuse and ORing fault and warning status	Read Byte	N/A	User Logic
0x80	STATUS_MFR_SPECIFIC	User defined status information	Read Byte	N/A	User Logic
0x81	STATUS_FANs_1_2	Fan 1 and 2 fault and warning status	Read Byte	N/A	User Logic
0x82	STATUS_FANs_3_4	Fan 3 and 4 fault and warning status	Read Byte	N/A	User Logic
0x8B	READ_VOUT	Voltage measurement of active page	Read Word	0x00-0x2F	PMBus Adapter
0x8C	READ_IOUT	Current sense measurement of active page	Read Word	0x30-0x3F	PMBus Adapter

COMMAND CODE	COMMAND NAME	DESCRIPTION	TYPE	PAGE RANGE	PROCESSING
0x8D	READ_TEMPERATURE	Temperature measurement of active page	Read Word	0x40-0x5F	PMBus Adapter
0x98	PMBUS_REVISION	PMBus revision supported by PMBus Adapter (Rev 1.1)	Read Byte	N/A	PMBus Adapter
0xD0	MFR_INTERLEAVE_OFF	Clear power supply interleave request signal	Send Byte	N/A	User Logic
0xD1	MFR_INTERLEAVE_ON	Set power supply interleave request signal	Send Byte	N/A	User Logic
0xD3	MFR_COEFFICIENT	Read m coefficient for active current sense pages only	Read Word	0x30-0x3F	PMBus Adapter

1. The STATUS_CML bit is processed directly by the PMBus Adapter. All other bits require user logic for processing.

PAGE

The PAGE command is a R/W Byte command. The PAGE command is used to set the active page to switch between the voltage, current sense, and temperature monitors of the ASC devices in the system. The PMBus Adapter supports up to 96 pages, including 48 pages reserved for voltage monitors, 16 pages reserved for current sense monitors, and 32 pages reserved for temperature monitors. The page space is partitioned by monitor type as shown in Table 2 shown below:

Table 2. PAGE Map Per Monitor Type

Page Number	Monitor Type
0x00-0x2F	Voltage
0x30-0x3F	Current Sense
0x40-0x5F	Temperature
Others	Reserved

The exact monitor channel and ASC device in the system which corresponds to a given page is assigned by the user in the Platform Designer software (described in the Platform Designer Settings section). The commands listed in Table 1 identify the page range for each command. Each page based command only supports either voltage, current sense, or temperature pages. The PMBus Adapter also uses the PAGE command to trigger a voltage, current sense or temperature measurement to queue the data for the READ_VOUT, READ_IOUT or READ_TEMPERATURE commands. See the individual commands for more details. The active page is also provided as an 8-bit output signal to the user logic from the PMBus Adapter. The page information can be decoded by the user logic in order to provide page specific status bytes, see the [PAGE_BYTE](#) section for more details. The PAGE value is also available for readback by the PMBus master.

OPERATION

The OPERATION command is a R/W Byte command. The OPERATION command is used by the PMBus master to request the PMBus slave turn supplies on and off and to margin the supplies. The PMBus Adapter responds to the data byte encoding shown in Table 3. The PMBus Adapter will assert the related operation signal when an OPERATION command is received, while all other operation signals are deasserted. The OPERATION processing must be implemented in the user logic (see the [Operational Commands](#) section for details). This allows the user a wide range of flexibility to implement their required system behavior in response to the OPERATION commands. Table 3 shows the OPERATION data byte as defined by the PMBus specification, with an additional column showing the signal which is asserted to the user logic.

Table 3. OPERATION Data Byte Signal Behavior

OPERATION Data Byte				PMBus Response Definition		User Logic Signal Asserted
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	PMBUS_IMMED_OFF
01	XX	XX	XX	Soft Off (With Sequencing)	N/A	PMBUS_SOFT_OFF
10	00	XX	XX	On	Off	PMBUS_ON_MARGIN_OFF
10	01	01	XX	On	Margin Low (Ignore Fault)	PMBUS_MARGIN_LOW_IF
10	01	10	XX	On	Margin Low (Act on Fault)	PMBUS_MARGIN_LOW_AF
10	10	01	XX	On	Margin High (Ignore Fault)	PMBUS_MARGIN_HI_IF
10	10	10	XX	On	Margin High (Act on Fault)	PMBUS_MARGIN_HI_AF

The OPERATION signals are global signals, they are not asserted per individual page. The USER logic can optionally handle these signals as page based by decoding the active page information and responding accordingly. See the [PAGE_BYTE](#) section for more details.

Any data byte which is provided with the OPERATION command which is not shown in Table 3 is considered invalid. The PMBus Adapter will automatically flag an Invalid Data bit in the CML STATUS byte, and assert (pull low) the SMBALERT# signal if present.

CLEAR_FAULTS

The CLEAR_FAULTS command is a Send Byte command. The CLEAR_FAULTS command is used in PMBus systems to clear any STATUS bits that have been set in the PMBus slave. It is also used to clear and release the SMBALERT# signal output if the slave is asserting the SMBALERT# signal. If the fault is still present when the status bit is cleared, the corresponding STATUS bit should be immediately set again.

The PMBus Adapter provides the momentary signal PMBUS_CLEAR_FAULTS to the user logic. The user logic is responsible for managing the majority of the STATUS bits in the device (only the STATUS_CML bits are an exception). The user logic must implement processing of the CLEAR_FAULTS signal according to the desired system behavior.

Since the PMBUS_CLEAR_FAULTS signal is momentary, it should be used as a reset or other edge sensitive signal to the STATUS bit registers. See the [Status Bytes and Clear Faults Command](#) section under [Connecting the PMBus Adapter to the Design](#) for an example implementation.

The PMBus Adapter will automatically negate the SMBALERT# signal upon receipt of a CLEAR_FAULTS command. See the [SMBALERT# Signal](#) section for more details.

WRITE_PROTECT

The WRITE_PROTECT command is a R/W Byte command. The WRITE_PROTECT command is used in PMBus systems to control writing to PMBus slaves.

The PMBus Adapter handles the WRITE_PROTECT support directly with no requirement for user logic. The WRITE_PROTECT command supports the data bytes and settings shown in Table 4.

Table 4. WRITE_PROTECT Behavior

Data Byte	WRITE_PROTECT Setting
1000 0000	Disable all writes except to the WRITE_PROTECT command
0100 0000	Disable all writes except to the WRITE_PROTECT, OPERATION and PAGE commands
0000 0000	Enables writes to all commands
All others	Reserved (Writing these values will cause Invalid Data fault in STATUS_CML Byte)

The PMBus Adapter will only update the user logic signals or fault limits when allowed by the write protect setting. Sending a disabled command will cause the PMBus Adapter to set the Invalid Command bit in the STATUS_CML byte, and assert the SMBALERT# signal if enabled. The PMBus Adapter directly handles the write protection mechanism. Enabling the hardware write protection mechanism in the ASC device is not required to support this feature.

CAPABILITY

The CAPABILITY command is a Read Byte command. This command provides a way for a PMBus Master to determine some key capabilities of the PMBus slave. The data byte format is shown in Table 5. This command is processed directly by the PMBus Adapter. The values of the different features in the CAPABILITY data byte are programmed automatically dependent on the user-defined Platform Designer settings (described in the Platform Designer Settings section of this document).

Table 5. CAPABILITY Bit Definition

Bits	Description	Value	Meaning
7	Packet Error Checking	0	Packet Error Checking not supported
		1	Packet Error Checking is supported
6:5	Maximum Bus Speed	00	Maximum supported bus speed is 100 kHz
		01	Maximum supported bus speed is 400 kHz
		10	Reserved
		11	Reserved
4	SMBALERT#	0	The device does not have a SMBALERT# pin and does not support the SMBus Alert Response protocol
		1	The device does have a SMBALERT# pin and does support the SMBus Alert Response protocol
3:0	Reserved	X	Reserved

VOUT_OV_FAULT_LIMIT, VOUT_UV_FAULT_LIMIT

The VOUT_OV_FAULT_LIMIT and VOUT_UV_FAULT_LIMIT commands are R/W Word commands. These commands operate on the active page only and are only valid for the page range reserved for voltage monitors (0x00 to 0x2F). These commands are used to specify the voltage levels for the selected monitor that correspond to an over-voltage fault or undervoltage fault.

The PMBus Adapter supports the DIRECT data format for these transactions as defined in the PMBus protocol. There are two equations defined in the specification, one for sending data to a PMBus slave (Equation 1), and one for reading data from a PMBus slave (Equation 4). The equations are repeated here for clarity and use the variables defined below.

Equation 1. Sending Data to a PMBus Slave

$$Y = (mX + b) \cdot 10^R$$

- **X** is the real world value (in Volts);
- **Y** is a two byte two's complement integer to be sent to the PMBus slave
- **m** is the slope, a two byte two's complement integer.
- **b** is the offset, a two byte two's complement integer
- **R** is the exponent, a one byte two's complement integer

The PMBus Adapter uses a fixed set of coefficients for the voltage fault limits, with **R** and **b** both set to 0. The voltage fault limit commands use **m**=500. (This corresponds to a voltage fault limit resolution of 2 mV.) The data format can be simplified to Equation 2, where **X** is the value in Volts and **Y** is the value written in the PMBus command.

Equation 2. Sending Voltage Fault Limits to the PMBus Adapter

$$Y = 500 * X$$

Equation 3 shows an example calculation for sending a voltage fault limit.

Equation 3. Example: Sending a Voltage Fault Limit of X = 2.5 Volts

$$Y = 500 * 2.5 \text{ Volts} = 1250$$

The PMBus Adapter uses the same set of coefficients for reading back the voltage fault limits as it does for sending, as shown in Equation 4.

Equation 4. Reading Data from a PMBus Slave

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

This can be simplified to Equation 5, where **X** is the value in Volts and **Y** is the value reported in the PMBus command.

Equation 5. Reading Voltage Data from the PMBus Adapter

$$X = \frac{Y}{500}$$

Equation 6 shows an example calculation for interpreting a voltage fault limit reading.

Equation 6. Example: Interpreting a Voltage Fault Limit Reading of Y = 600

$$X = \frac{600}{500} = 1.2 \text{ Volts}$$

The PMBus Adapter supports the VOUT_OV_FAULT_LIMIT and VOUT_UV_FAULT_LIMIT directly and does not require any user logic to process these commands. The adapter assigns the VOUT_OV_FAULT_LIMIT to the VMONx_A threshold and the VOUT_UV_FAULT_LIMIT to the VMONx_B threshold. The PMBus Adapter will convert the requested voltage fault limit to the closest available voltage monitor trip point (see the data sheet section on ASC Configuration Registers, subsection on Voltage Monitors). The adapter selects either the overvoltage or undervoltage trip point table depending on the received command code. The PMBus Adapter will then write the new trip point into the ASC shadow configuration registers over the I²C link between the MachXO2 and the ASC. The PMBus Adapter determines the ASC I²C address and configuration register address based on the page map-

ping configuration set by the user in Platform Designer. The voltage fault limit commands can then be used to read-back the fault limit levels. See the [Connecting the PMBus Adapter to the Design](#) section for more details on how the fault limit commands work.

Sending a VOUT_OV_FAULT_LIMIT or VOUT_UV_FAULT_LIMIT command to the PMBus Adapter when the active page is outside the voltage page range (0x00-0x2F) will cause an Invalid Command fault in the STATUS_CML byte. Sending a value outside the fault limit range (see DS1042, [L-ASC10 Data Sheet](#) for valid voltage threshold ranges) will cause an Invalid Data fault in the STATUS_CML byte. The PMBus Adapter will also assert (pull low) the SMBALERT# signal (if the SMBALERT# feature is enabled) when either CML fault bit is set. See the [STATUS_CML](#) section command description for more details.

IOUT_OC_FAULT_LIMIT, IOUT_UC_FAULT_LIMIT

The IOUT_OC_FAULT_LIMIT and IOUT_UC_FAULT_LIMIT commands are R/W Word commands. These commands operate on the active page only and are only valid for the page range reserved for current sense monitors (0x30 to 0x3F). These commands are used to specify the current sense levels for a given monitor that will correspond to an overcurrent fault or undercurrent fault.

The PMBus Adapter supports the DIRECT data format for these transactions as defined in the PMBus protocol and described in the [VOUT_OV_FAULT_LIMIT, VOUT_UV_FAULT_LIMIT](#) section above.

The PMBus Adapter has fixed coefficients $R=0$ and $b=0$. The current sense fault limit commands uses a programmable m coefficient. The value of m is dependent on the sense resistor connected to the current sense monitor inputs of the ASC. Platform Designer will automatically calculate the value of m and store it in the adapter memory map of the FPGA. The m value can be read out using the [MFR_IOUT_COEFFICIENT](#) command. The data format for writing current limits can be simplified to Equation 7, where X is the value in Amps and Y is the value written in the PMBus command.

Equation 7. Sending Current Sense Fault Limits to the PMBus Adapter

$$Y = m * X$$

Equation 8 shows an example calculation for sending a current sense fault limit

Equation 8. Example: Sending a Current Sense Fault Limit $X = 2$ Amps with $m = 40$

$$Y = 40 * 2 \text{ Amps} = 80 = 0x0050$$

The PMBus Adapter uses the same set of coefficients for reading back the current sense fault limits. The data format for reading current sense fault limits can then be simplified to Equation 9, where X is the value in Amps and Y is the value reported in the PMBus command.

Equation 9. Reading Current Sense Fault Limits from the PMBus Adapter

$$X = \frac{Y}{m}$$

Equation 10 shows an example calculation for interpreting a current sense fault limit reading.

Equation 10. Example: Interpreting a Current Sense Fault Limit Reading of $Y = 226$ with $m = 25$

$$X = \frac{226}{25} = 9.04 \text{ Amps}$$

The PMBus Adapter supports the IOUT_OC_FAULT_LIMIT and IOUT_UC_FAULT_LIMIT directly and does not require any user logic to process these commands. The PMBus Adapter assigns the IOUT_OC_FAULT_LIMIT to the IMON1_A/HIMON_A threshold and the IOUT_UC_FAULT_LIMIT to the IMON1_B/HIMON_B threshold. The PMBus Adapter will compare the requested current sense fault limit to available trip points. The available trip point **Y** codes are shown in Table 6. There are 16 discrete trip points available in the hardware corresponding to differential voltages. The PMBus Adapter will then write the requested fault limit levels into the ASC shadow configuration registers over the I²C link between the MachXO2 and the ASC (or flag an error if the requested limit is invalid, see below). The PMBus Adapter determines the ASC I²C address and configuration register address based on the page mapping configuration set by the user in Platform Designer. The current sense fault limit commands can also be used to readback the fault limit levels. See the [Connecting the PMBus Adapter to the Design](#) for more details on how the fault limit commands work.

Table 6. Allowed IOUT_FAULT_LIMIT Data Codes

IOUT_FAULT_LIMIT Data Code (Y)	Differential Voltage
0x0020	8 mV
0x002A	10.5 mV
0x003A	14.5 mV
0x003E	15.5 mV
0x0050	20 mV
0x0052	20.5 mV
0x0072	28.5 mV
0x007A	30.5 mV
0x0094	39 mV
0x00A2	40.5 mV
0x00E2	56.5 mV
0x012C	75 mV
0x0134	77 mV
0x0190	100 mV
0x0230	140 mV
0x02F8	190 mV

Sending an IOUT_OC_FAULT_LIMIT or IOUT_UC_FAULT_LIMIT command to the PMBus Adapter when the active page is outside the current sense page range (0x30-0x3F) will cause an Invalid Command fault in the STATUS_CML byte. Sending a **Y** value not listed in Table 6 will cause an Invalid Data fault in the STATUS_CML byte. The PMBus Adapter will also assert (pull low) the SMBALERT# (if the SMBALERT# feature is enabled) when either CML fault bit is set. See the [STATUS_CML](#) section command description for more details.

OT_FAULT_LIMIT, UT_FAULT_LIMIT

The OT_FAULT_LIMIT and UT_FAULT_LIMIT commands are R/W Word commands. These commands operate on the active page only and are only valid for the page range reserved for temperature monitors (0x40 to 0x5F). These commands are used to specify the temperature thresholds for a given monitor that will correspond to an overtemperature fault or undertemperature fault.

The PMBus Adapter supports the DIRECT data format for these transactions as defined in the PMBus protocol and described in the [VOUT_OV_FAULT_LIMIT, VOUT_UV_FAULT_LIMIT](#) section above.

The PMBus Adapter has fixed coefficients **R**=0 and **b**=0. The temperature fault limit commands use **m**=4. The data format for writing temperature limits can be simplified to Equation 11, where **X** is the value in Degrees Celsius and **Y** is the value written in the PMBus command.

Equation 11. Sending Temperature Fault Limit Data to the PMBus Adapter

$$Y = 4 * X$$

Equation 12 shows an example calculation for sending a temperature fault limit.

Equation 12. Example: Sending a Temperature Fault Limit of $X = 85^{\circ}\text{C}$

$$Y = 85^{\circ}\text{C} * 4 = 340$$

The PMBus Adapter uses the same set of coefficients for reading back the temperature fault limits. The data format for reading temperature fault limits can then be simplified to Equation 13, where X is the value in Degrees Celsius and Y is the value reported in the PMBus command.

Equation 13. Reading Temperature Fault Limit Data from the PMBus Adapter

$$X = \frac{Y}{4}$$

Equation 14 shows an example calculation for interpreting a temperature fault limit reading.

Equation 14. Example: Interpreting a Temperature Fault Limit Reading of $Y = 100$

$$X = \frac{100}{4} = 25^{\circ}\text{C}$$

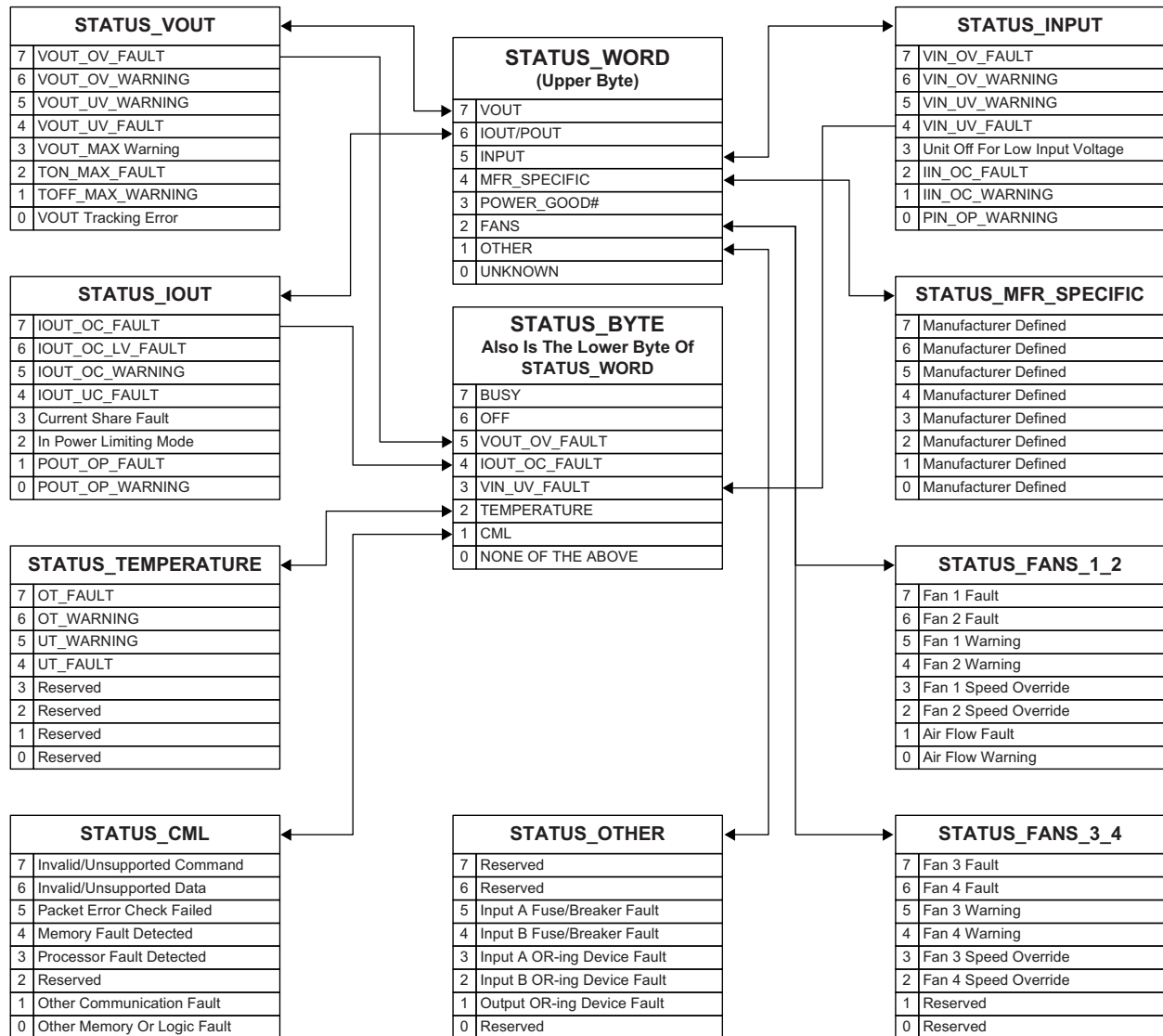
The PMBus Adapter supports the OT_FAULT_LIMIT and UT_FAULT_LIMIT directly and does not require any user logic to process these commands. The adapter assigns the OT_FAULT_LIMIT to the TMONx_A threshold and the UT_FAULT_LIMIT to the TMONx_B threshold. The PMBus Adapter will convert the requested temperature fault limit to the closest available trip point (see the data sheet section on ASC Configuration Registers, subsection on Temperature Monitors). The PMBus Adapter will then write the requested fault limit levels into the ASC shadow configuration registers over the I²C link between the MachXO2 and the ASC. The adapter decodes the ASC I²C address and configuration register address based on the page mapping configuration set by the user in Platform Designer. The temperature fault limit commands can also be used to readback the temperature fault limit levels. See the [Connecting the PMBus Adapter to the Design](#) section for more details on how the fault limit commands work.

Sending an OT_FAULT_LIMIT or UT_FAULT_LIMIT command to the adapter when the active page is outside the temperature page range (0x40-0x5F) will cause an Invalid Command fault in the STATUS_CML byte. Sending a value outside the fault limit range (see DS1042, [L-ASC10 Data Sheet](#) for valid temperature threshold ranges) will cause an Invalid Data fault in the STATUS_CML byte. The PMBus Adapter will also assert (pull low) the SMBALERT# (if the SMBALERT# feature is enabled) when either CML fault bit is set. See the [STATUS_CML](#) section command description for more details.

STATUS COMMANDS

The PMBus Adapter supports the full group of status commands shown in Figure 3. These commands are global (that is, not page specific). The PMBus specification expects the commands to be mapped together as shown in Figure 3. The STATUS_WORD (which includes the STATUS_BYTE) is a summary of all the status information. Each of the additional STATUS commands (such as STATUS_VOUT or STATUS_IOUT) are meant to correspond to one or more bits in the STATUS_WORD.

Figure 3. STATUS Bit Mapping in PMBus Systems¹



¹ Source: PMBus™ Power System Management Protocol Specification
Part II - Command Language
Page 77

The PMBus Adapter provides an individual user logic signal for each of the status bits, including the STATUS_BYTE and STATUS_WORD bits. The user logic determines the behavior and relationship between each of the bits available in the STATUS commands.

The mapping defined in Figure 3 must be completed in the user logic. There is no automatic connection between STATUS_BYTE/STATUS_WORD bits and their associated STATUS commands. For example, the user must map the connection between the STATUS_BYTE_VOUT_OV bit and the STATUS_VOUT byte bits. See the [Status Bytes and Clear Faults Command](#) section under [Connecting the PMBus Adapter to the Design](#).

The only exception is the STATUS_CML byte and the CML bit in the STATUS_WORD. The CML error handling is processed directly by the PMBus Adapter.

STATUS_BYTE

The STATUS_BYTE command is a Read Byte command. This command is used to report a summary of the status of the PMBus slave. The bits in the status byte are used to summarize the other STATUS bytes as shown in Figure 3. Each of the bits is set dependent on the state of the signal shown in Table 7.

The user logic is responsible for managing each of the signal states according to the PMBus specification and system requirement (except for the STATUS_CML bit which is handled automatically by the PMBus Adapter).

Table 7. STATUS_BYTE / STATUS_WORD Lower Byte Definition

Bit	Status	User Logic Signal
7	BUSY	PMBUS_STATUS_BYTE_BUSY
6	OFF	PMBUS_STATUS_BYTE_OFF
5	VOUT_OV	PMBUS_STATUS_BYTE_VOUT_OV
4	IOUT_OC	PMBUS_STATUS_BYTE_IOUT_OC
3	VIN_UV	PMBUS_STATUS_BYTE_VIN_UV
2	TEMPERATURE	PMBUS_STATUS_BYTE_TEMPERATURE
1	CML	N/A ¹
0	NONE OF THE ABOVE	PMBUS_STATUS_BYTE_NONE_OF_ABOVE

1. CML bit is handled automatically by the PMBus Adapter.

STATUS_WORD

The STATUS_WORD command is a Read Word command. This command is used to report an expanded summary of the status of the PMBus slave. The bits in the status word are used to summarize the other STATUS bytes as shown in Figure 3. The lower byte of the STATUS_WORD is the same as the STATUS_BYTE as shown in Table 7. The STATUS_WORD upper byte bits are set dependent on the state of the signals shown in Table 8.

The user logic is responsible for managing each of the signal states according to the PMBus specification and system requirement (except for the STATUS_CML bit which is handled automatically by the PMBus Adapter).

Table 8. STATUS_WORD Upper Byte Definition

Bit	Status	User Logic Signal
7	VOUT	PMBUS_STATUS_WORD_VOUT_OV
6	IOUT/POUT	PMBUS_STATUS_WORD_IOUT_POUT
5	INPUT	PMBUS_STATUS_WORD_INPUT
4	MFR_SPECIFIC	PMBUS_STATUS_WORD_MFR
3	POWER_GOOD#	PMBUS_STATUS_WORD_POWER_GOOD
2	FANS	PMBUS_STATUS_WORD_FANS
1	OTHER	PMBUS_STATUS_WORD_OTHER
0	UNKNOWN	PMBUS_STATUS_WORD_UNKNOWN

STATUS_VOUT

The STATUS_VOUT command is a Read Byte command. This command is used to report the voltage status of the PMBus slave. These bits are summarized with the VOUT and VOUT_OV status bits in the STATUS_WORD as shown in Figure 3. The user logic is responsible for mapping the STATUS_VOUT to the STATUS_WORD bits. Each of the STATUS_VOUT bits is set dependent on the state of the signal shown in Table 9. The user logic is responsible for managing each of the signal states according to the PMBus specification and system requirement.

Table 9. STATUS_VOUT Byte Definition

Bit	Status	User Logic Signal
7	VOUT_OV_FAULT	PMBUS_STATUS_VOUT_OV_FAULT
6	VOUT_OV_WARNING	PMBUS_STATUS_VOUT_OV_WARNING
5	VOUT_UV_WARNING	PMBUS_STATUS_VOUT_UV_WARNING
4	VOUT_UV_FAULT	PMBUS_STATUS_VOUT_UV_FAULT
3	VOUT_MAX_WARNING	PMBUS_STATUS_VOUT_MAX_WARNING
2	TON_MAX_FAULT	PMBUS_STATUS_VOUT_TON_MAX_FAULT
1	TOFF_MAX_WARNING	PMBUS_STATUS_VOUT_TOFF_MAX_WARNING
0	VOUT Tracking Error	PMBUS_STATUS_VOUT_POWER_ON_TRACKING_ERROR

STATUS_IOUT

The STATUS_IOUT command is a Read Byte command. This command is used to report the current sense status of the PMBus slave. These bits are summarized the IOUT/POUT and IOUT_OC status bits in the STATUS_WORD as shown in Figure 3. The user logic is responsible for mapping the STATUS_IOUT to the STATUS_WORD bits. Each of the STATUS_IOUT bits is set dependent on the state of the signal shown in Table 10. The user logic is responsible for managing each of the signal states according to the PMBus specification and system requirement.

Table 10. STATUS_IOUT Byte Definition

Bit	Status	User Logic Signal
7	IOUT_OC_FAULT	PMBUS_STATUS_IOUT_OC_FAULT
6	IOUT_OC_LV_FAULT	PMBUS_STATUS_IOUT_OC_LV_SHUTDOWN
5	IOUT_OC_WARNING	PMBUS_STATUS_IOUT_OC_WARNING
4	IOUT_UC_FAULT	PMBUS_STATUS_IOUT_UC_FAULT
3	Current Share Fault	PMBUS_STATUS_IOUT_CURRENT_SHARE_FAULT
2	In Power Limiting Mode	PMBUS_STATUS_IOUT_IN_POWER_LIMITING_MODE
1	POUT_OP_FAULT	PMBUS_STATUS_IOUT_POUT_OP_FAULT
0	POUT_OP_WARNING	PMBUS_STATUS_IOUT_POUT_OP_WARNING

STATUS_INPUT

The STATUS_INPUT command is a Read Byte command. This command is used to report the input supply status of the PMBus slave. These bits are summarized with the INPUT and VIN_UV_FAULT status bits in the STATUS_WORD as shown in Figure 3. The user logic is responsible for mapping the STATUS_INPUT to the STATUS_WORD bits. Each of the STATUS_INPUT bits is set dependent on the state of the signal shown in Table 11. The user logic is responsible for managing each of the signal states according to the PMBus specification and system requirement.

Table 11. STATUS_INPUT Byte Definition

Bit	Status	User Logic Signal
7	VIN_OV_FAULT	PMBUS_STATUS_INPUT_VIN_OV_FAULT
6	VIN_OV_WARNING	PMBUS_STATUS_INPUT_VIN_OV_WARNING
5	VIN_UV_WARNING	PMBUS_STATUS_INPUT_VIN_UV_WARNING
4	VIN_UV_FAULT	PMBUS_STATUS_INPUT_VIN_UV_FAULT
3	Unit Off for Low Input Voltage	PMBUS_STATUS_INPUT_UNIT_OFF_FOR_LOW_INPUT_VOLTAGE
2	IIN_OC_FAULT	PMBUS_STATUS_INPUT_IN_OC_FAULT
1	IIN_OC_WARNING	PMBUS_STATUS_INPUT_IN_OC_WARNING
0	PIN_OP_WARNING	PMBUS_STATUS_INPUT_PIN_OP_WARNING

STATUS_TEMPERATURE

The STATUS_TEMPERATURE command is a Read Byte command. This command is used to report the temperature status of the PMBus slave. These bits are summarized with the TEMPERATURE status bit in the STATUS_WORD as shown in Figure 3. The user logic is responsible for mapping the STATUS_TEMPERATURE to the STATUS_WORD bit. Each of the STATUS_TEMPERATURE bits is set dependent on the state of the signal shown in Table 12. Reserved bits will always return 0. The user logic is responsible for managing each of the signal states according to the PMBus specification and system requirement.

Table 12. STATUS_TEMPERATURE Byte Definition

Bit	Status	User Logic Signal
7	OT_FAULT	PMBUS_STATUS_TEMP_OT_FAULT
6	OT_WARNING	PMBUS_STATUS_TEMP_OT_WARNING
5	UT_WARNING	PMBUS_STATUS_TEMP_UT_WARNING
4	UT_FAULT	PMBUS_STATUS_TEMP_UT_FAULT
3	Reserved	PMBUS_STATUS_TEMP_RESERVED_3
2	Reserved	PMBUS_STATUS_TEMP_RESERVED_2
1	Reserved	PMBUS_STATUS_TEMP_RESERVED_1
0	Reserved	PMBUS_STATUS_TEMP_RESERVED_0

STATUS_CML

The STATUS_CML command is a Read Byte command. This command is used to report the CML (Communication, Memory and Logic) status of the PMBus slave. These bits are automatically mapped by the PMBus Adapter into the CML status bit in the STATUS_WORD as shown in Figure 3. Each of the bits is handled directly by the PMBus Adapter as described in Table 13. Reserved or Unused bits will always return 0.

The PMBus Adapter will automatically assert (pull low) the SMBALERT# (if the SMBALERT# feature is enabled) when any CML fault is detected. The STATUS_CML bits will be cleared upon receipt of a CLEAR_FAULTS command.

Table 13. STATUS_CML Byte Definition

Bit	Status	Error Causes
7	Invalid/Unsupported Command	1. Any command not listed in Table 1 2. Any page based command sent to an invalid page (voltage command to current sense page, and others) 3. Sending command that has been disabled by WRITE_PROTECT
6	Invalid/Unsupported Data	1. Invalid OPERATION byte (Table 3) 2. Invalid WRITE_PROTECT byte (Table 4) 3. Invalid current sense fault limit setting (Table 6) 4. Invalid/Unconfigured PAGE
5	Packet Error Check Failed	Any command PEC error
4	Memory Fault Detected	Unused
3	Processor Fault Detected	Unused
2	Reserved	Unused
1	Other Communication Fault	Unused
0	Other Memory or Logic Fault	Unused

STATUS_OTHER

The STATUS_OTHER command is a Read Byte command. This command is used to report miscellaneous status conditions of the PMBus slave. These bits are summarized with the OTHER status bit in the STATUS_WORD as shown in Figure 3. The user logic is responsible for mapping the STATUS_OTHER to the STATUS_WORD bit. Each of the STATUS_OTHER bits is set dependent on the state of the signal shown in Table 14. Reserved bits will always return 0. The user logic is responsible for managing each of the signal states according to the PMBus specification and system requirement.

Table 14. STATUS_OTHER Byte Definition

Bit	Status	User Logic Signal
7	Reserved	PMBUS_STATUS_OTHER_RESERVED_7
6	Reserved	PMBUS_STATUS_OTHER_RESERVED_6
5	Input A Fuse/Breaker Fault	PMBUS_STATUS_OTHER_INPUT_A_FUSEBREAKER_FAULT
4	Input B Fuse/Breaker Fault	PMBUS_STATUS_OTHER_INPUT_B_FUSEBREAKER_FAULT
3	Input A OR-ing Device Fault	PMBUS_STATUS_OTHER_INPUT_A_OR_ING_DEVICE_FAULT
2	Input B OR-ing Device Fault	PMBUS_STATUS_OTHER_INPUT_B_OR_ING_DEVICE_FAULT
1	Output OR-ing Device Fault	PMBUS_STATUS_OTHER_OUTPUT_OR_ING_DEVICE_FAULT
0	Reserved	PMBUS_STATUS_OTHER_RESERVED_0

STATUS_MFR_SPECIFIC

The STATUS_MFR_SPECIFIC command is a Read Byte command. This command is used to report manufacturer specific status of the PMBus slave. This byte is free for user-defined assignment according to the system design requirements. These bits are summarized with the MFR status bit in the STATUS_WORD as shown in Figure 3. The user logic is responsible for mapping the STATUS_MFR_SPECIFIC to the STATUS_WORD bit. Each of the STATUS_MFR_SPECIFIC bits is set dependent on the state of the signal shown in Table 15. The user logic is responsible for managing each of the signal states according to the system requirement. The byte could be used to encode the page of a faulty voltage rail, provide sequencing step feedback, or any other custom usage.

Table 15. STATUS_MFR_SPECIFIC Byte Definition

Bit	Status	User Logic Signal
7	Manufacturer Defined	PMBUS_STATUS_MFR_SPECIFIC_7
6	Manufacturer Defined	PMBUS_STATUS_MFR_SPECIFIC_6
5	Manufacturer Defined	PMBUS_STATUS_MFR_SPECIFIC_5
4	Manufacturer Defined	PMBUS_STATUS_MFR_SPECIFIC_4
3	Manufacturer Defined	PMBUS_STATUS_MFR_SPECIFIC_3
2	Manufacturer Defined	PMBUS_STATUS_MFR_SPECIFIC_2
1	Manufacturer Defined	PMBUS_STATUS_MFR_SPECIFIC_1
0	Manufacturer Defined	PMBUS_STATUS_MFR_SPECIFIC_0

STATUS_FANS_1_2

The STATUS_FANS_1_2 command is a Read Byte command. This command is used to report FAN1 and FAN2 status conditions from the PMBus slave. These bits are summarized with the FAN status bit in the STATUS_WORD as shown in Figure 3. The user is responsible for mapping the STATUS_FANS_1_2 to the STATUS_WORD bit. Each of the STATUS_FANS_1_2 bits is set dependent on the state of the signal shown in Table 16. The user logic is responsible for managing each of the signal states according to the PMBus specification and system requirement.

Table 16. STATUS_FANS_1_2 Byte Definition

Bit	Status	User Logic Signal
7	Fan 1 Fault	PMBUS_STATUS_FANS_1_2_FAN_1_FAULT
6	Fan 2 Fault	PMBUS_STATUS_FANS_1_2_FAN_2_FAULT
5	Fan 1 Warning	PMBUS_STATUS_FANS_1_2_FAN_1_WARNING
4	Fan 2 Warning	PMBUS_STATUS_FANS_1_2_FAN_2_WARNING
3	Fan 1 Speed Override	PMBUS_STATUS_FANS_1_2_FAN_1_SPEED_OVERRIDE
2	Fan 2 Speed Override	PMBUS_STATUS_FANS_1_2_FAN_2_SPEED_OVERRIDE
1	Air Flow Fault	PMBUS_STATUS_FANS_1_2_AIR_FLOW_FAULT
0	Air Flow Warning	PMBUS_STATUS_FANS_1_2_AIR_FLOW_WARNING

STATUS_FANS_3_4

The STATUS_FANS_3_4 command is a Read Byte command. This command is used to report FAN3 and FAN4 status conditions from the PMBus slave. These bits are summarized with the FAN status bit in the STATUS_WORD as shown in Figure 3. The user logic is responsible for mapping the STATUS_FANS_3_4 to the STATUS_WORD bit. Each of the STATUS_FANS_3_4 bits is set dependent on the state of the signal shown in Table 17. Reserved bits will always return 0. The user logic is responsible for managing each of the signal states according to the PMBus specification and system requirement.

Table 17. STATUS_FANS_3_4 Byte Definition

Bit	Status	User Logic Signal
7	Fan 3 Fault	PMBUS_STATUS_FANS_3_4_FAN_3_FAULT
6	Fan 4 Fault	PMBUS_STATUS_FANS_3_4_FAN_4_FAULT
5	Fan 3 Warning	PMBUS_STATUS_FANS_3_4_FAN_3_WARNING
4	Fan 4 Warning	PMBUS_STATUS_FANS_3_4_FAN_4_WARNING
3	Fan 3 Speed Override	PMBUS_STATUS_FANS_3_4_FAN_3_SPEED_OVERRIDE
2	Fan 4 Speed Override	PMBUS_STATUS_FANS_3_4_FAN_4_SPEED_OVERRIDE
1	Reserved	PMBUS_STATUS_FANS_3_4_RESERVED_1
0	Reserved	PMBUS_STATUS_FANS_3_4_RESERVED_0

READ_VOUT

The READ_VOUT command is a Read Word command that returns the most recent voltage measured from the page which is currently selected. This command is only supported for voltage pages (0x00 to 0x2F). Sending this command when any other page is active will cause a CML – Invalid Command fault.

The READ_VOUT voltage data is encoded in the DIRECT data Format, with $m=500$ (this corresponds to a 2 mV resolution - see Equation 4 for more details). The relationship between the READ_VOUT data and the actual voltage is shown in Equation 15, where X is the real world value in Volts and Y is the two byte two's complement value reported in the PMBus command.

Equation 15. Reading Voltage Measurement Data from the PMBus Adapter

$$X = \frac{Y}{500}$$

READ_VOUT will return the most recent measured voltage. The PMBus Adapter only initiates measurements (voltage, current sense, or temperature) when it receives a PAGE command. The PMBus Adapter will then refresh the measurement associated with the given page. The procedure for reading the voltage at a particular page is to follow the steps below:

1. PAGE Write Command with the selected page data (pages 0x00 to 0x2F are voltage pages)
 - a. The PMBus Adapter will automatically initiate a voltage measurement of the mapped voltage monitor
2. READ_VOUT Read Command to return the measured voltage code

The PMBus Adapter will not refresh the voltage measurement data unless a PAGE command is received.

READ_IOUT

The READ_IOUT command is a Read Word command that returns the most recent current measured on the page which is currently selected. This command is only supported for current sense pages (0x30 to 0x3F). Sending this command when any other page is active will cause a CML – Invalid Command fault.

The READ_IOUT current sense data is encoded in the DIRECT data Format, with a programmable **m** coefficient - see [Equation 4](#) for more details. The **m** coefficient can be readout from the PMBUS Adapter using the MFR_IOUT_COEFFICIENT command. The relationship between the READ_IOUT data and the actual temperature is shown in Equation 16, where **X** is the real world value in Amps and **Y** is the two byte two's complement value reported in the PMBus command.

Equation 16. Reading Current Sense Measurement Data from the PMBus Adapter

$$X = \frac{Y}{m}$$

READ_IOUT will return the most recent measured current. The PMBus Adapter only initiates measurements (voltage, current sense, or temperature) when it receives a PAGE command. The PMBus Adapter will refresh the measurement associated with the given page. The procedure for reading the current at a particular page is to follow the steps below:

1. PAGE Write Command with the selected page data (pages 0x30 to 0x3F are current sense pages)
 - a. The PMBus Adapter will automatically initiate a current measurement of the mapped channel
2. MFR_IOUT_COEFFICIENT Read Command to readback **m** coefficient for active page
3. READ_IOUT Read Command to return the measured current code

The PMBus Adapter will not refresh the current sense measurement data unless a PAGE command is received.

READ_TEMPERATURE

The READ_TEMPERATURE command is a Read Word command that returns the most recent temperature measured on the page which is currently selected. This command is only supported for temperature pages (0x40 to 0x5F). Sending this command when any other page is active will cause a CML – Invalid Command fault.

The READ_TEMPERATURE temperature data is encoded in the DIRECT data Format, with **m=4** - see [Equation 4](#) for more details. The relationship between the READ_TEMPERATURE data and the actual temperature is shown in the Equation 17, where **X** is the real world value in Degrees Celsius and **Y** is the two byte two's complement value reported in the PMBus command

Equation 17. Reading Temperature Measurement Data from the PMBus Adapter

$$X = \frac{Y}{4}$$

READ_TEMPERATURE will return the most recent measured temperature. The PMBus Adapter only initiates measurements (voltage, current sense, or temperature) when it receives a PAGE command. The PMBus Adapter will refresh the measurement associated with the given page. The procedure for reading the temperature of a particular page is to follow the steps below:

1. PAGE Write Command with the selected page data (pages 0x40 to 0x5F are temperature pages)
 - a. The PMBus Adapter will automatically initiate a temperature measurement of the mapped channel
2. READ_TEMPERATURE Read Command to return the measured temperature code

The PMBus Adapter will not refresh the temperature measurement data unless a PAGE command is received.

PMBUS_REVISION

The PMBUS_REVISION command is a Read Byte command that returns the PMBus standard revision supported by the PMBus Adapter. The returned data value is hard coded to 0x11, which corresponds to revision 1.1. This command is handled directly by the PMBus Adapter.

MFR_INTERLEAVE_OFF

The MFR_INTERLEAVE_OFF command is a Send Byte command. This is a manufacturer specific command which is used to de-assert the PMBUS_INTERLEAVE_ON_OFF signal. This signal can be used to support SYNC control of Point of Load (POL) power supplies connected to the PMBus Adapter. The user logic can process this signal and phase shift SYNC signals of attached POLs. This command works in conjunction with MFR_INTERLEAVE_ON, as shown in Table 18.

Table 18. MFR_INTERLEAVE_OFF / MFR_INTERLEAVE_ON Command Behavior

Command	User Logic Signal Status
MFR_INTERLEAVE_OFF	PMBUS_INTERLEAVE_ON_OFF = 0
MFR_INTERLEAVE_ON	PMBUS_INTERLEAVE_ON_OFF = 1

MFR_INTERLEAVE_ON

The MFR_INTERLEAVE_ON command is a Send Byte command. This is a manufacturer specific command which is used to assert the PMBUS_INTERLEAVE_ON_OFF signal. This command works in conjunction with MFR_INTERLEAVE_OFF, as shown in Table 18.

MFR_IOUT_COEFFICIENT

The MFR_IOUT_COEFFICIENT command is a Read Word command. This is a manufacturer specific command which is used to return only the **m** coefficient for current sense pages (page range 0x30 to 0x3F). This coefficient is calculated by Platform Designer and stored inside the PMBus Adapter memory for each configured current sense page. The coefficient is scaled depending on the sense resistor which is set by the user in the Platform Designer tool.

The coefficient is used along with the READ_IOUT, IOUT_FAULT_LIMIT_OC, and IOUT_FAULT_LIMIT_UC commands by the PMBus master to define the relationship between the PMBus data and the real world current. Each command includes a description of how to calculate the command data value based on the MFR_IOUT_COEFFICIENT reading.

Sending the MFR_IOUT_COEFFICIENT command to any page outside the current sense page range of 0x30-0x3F will result in a CML Invalid Command Fault.

PMBus Adapter Optional Features

SMBALERT# Signal

PMBus supports the optional implementation of an SMBALERT# signal. This signal can be used by PMBus slave devices to notify the PMBus master that a fault condition has occurred. The SMBALERT# signal is an open-drain output from the slave devices, with each slaves SMBALERT# signal tied together on the board and connected to a single input at the PMBus master. SMBus (and by extension PMBus) defines a protocol for the master to determine which slave (or slaves) has asserted the SMBALERT# signal. This is the SMBus Alert Response protocol.

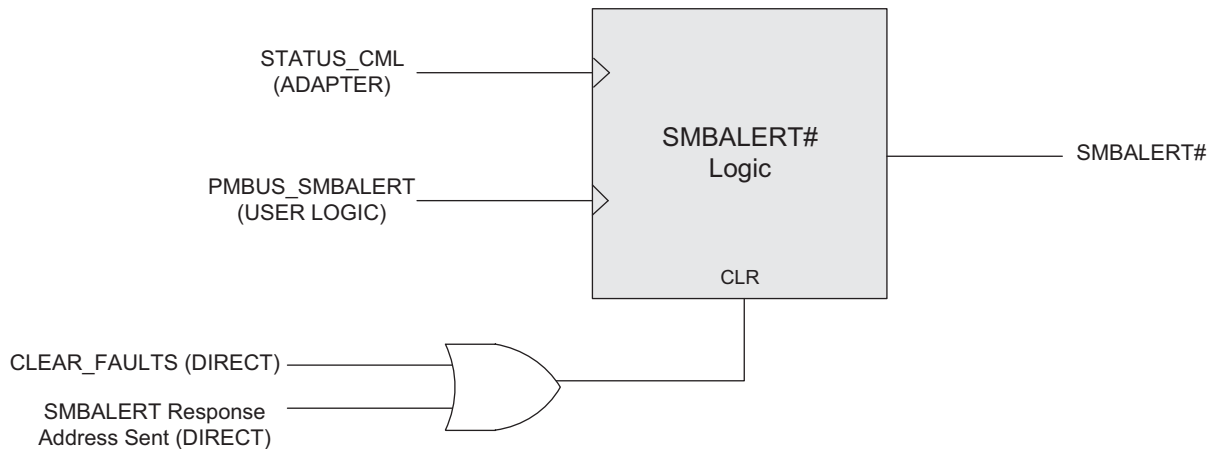
The PMBus Adapter supports this optional feature. The PMBus Adapter follows the SMBALERT# behavior defined in the PMBus specification. Figure 4 shows the implementation of the SMBALERT# signal handling in the PMBus Adapter.

The SMBALERT# signal will be asserted (pulled low) when the PMBus Adapter detects a CML fault, or the user logic drives the PMBUS_SMBALERT logic signal high. The SMBALERT# circuit is edge sensitive and it will pull SMBALERT# low when either the adapter output or the user logic signal transitions high.

The SMBALERT# signal is sticky, it will remain pulled low until it is cleared by the PMBus Adapter. The PMBus Adapter will clear the signal based on one of two conditions:

- PMBus command CLEAR_FAULTS is received by the slave
- Slave device successfully transmits its address in response to receiving the Alert Response Address from the PMBus master

Figure 4. SMBALERT# Logic



The SMBALERT# signal will be asserted (pulled low) again when either the PMBUS_SMBALERT or STATUS_CML fault signals transition high.

Packet Error Checking (PEC)

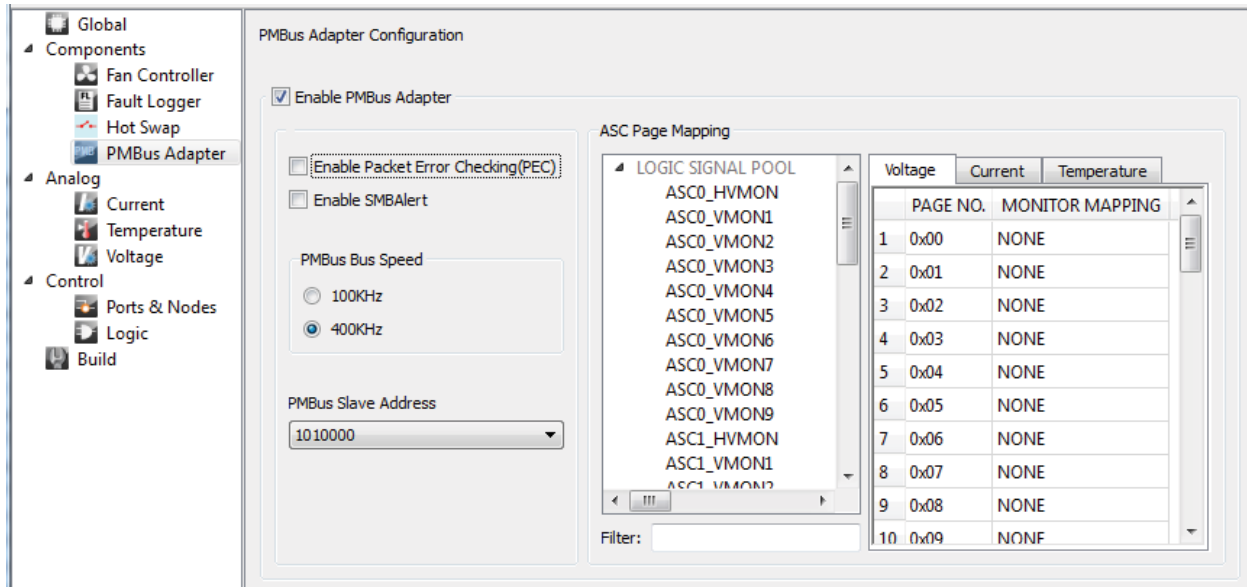
The PMBus Protocol Specification defines support for the SMBus Packet Error Checking (PEC) protocol as optional. The PMBus Adapter supports this feature as a user-selectable option. See Figure 2 for the PMBus command types including PEC. For more details on PEC see the SMBus Specification.

Platform Designer Settings

PMBus Adapter View

The PMBus Adapter is enabled and configured in the PMBus Adapter view found in the Components section of Platform Designer. The PMBus Adapter view is shown in Figure 5, with the configurable settings described in the list below.

Figure 5. PMBus Adapter View

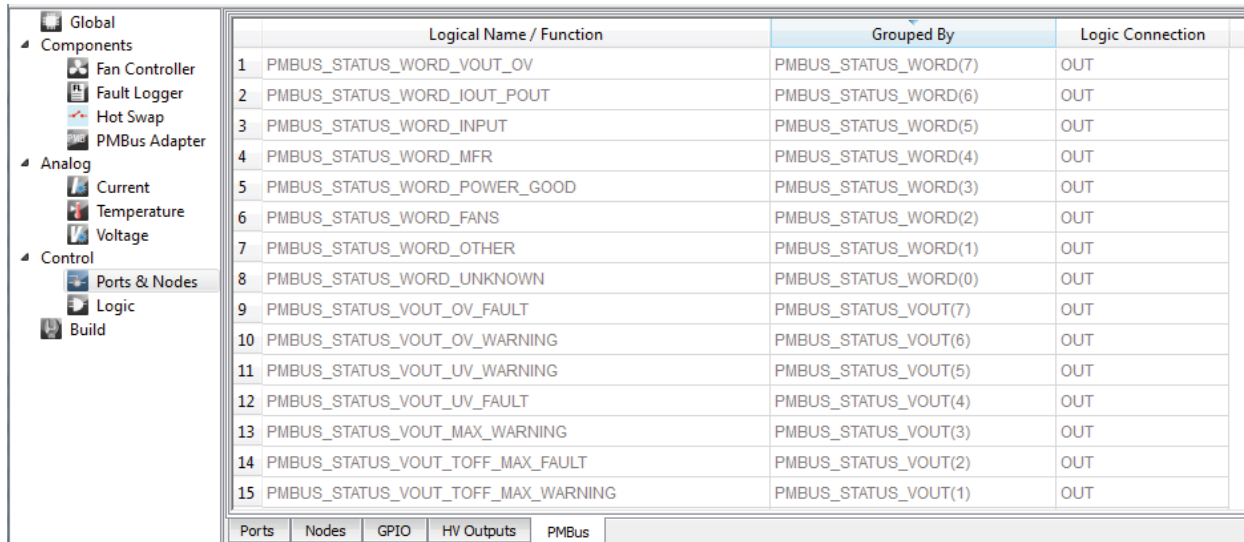


- **Enable PMBus Adapter** – This check box is used to enable the PMBus Adapter as a part of the design. Note that the PMBus Adapter is only available for projects targeted to MachXO2-4000 or MachXO2-7000 size devices. This box must be checked to proceed with configuration of the other adapter options.
- **Enable Packet Error Checking (PEC)** – This check box is used to enable the Packet Error Checking option as described in the PMBus Protocol Specification. When this box is checked, the PMBus Adapter will automatically calculate and provide the PEC during write and read transactions. PEC errors will be signalled in the STATUS_CML byte and using the SMBALERT# signal (if enabled). This setting will be reflected in the CAPABILITY command response by the PMBus Adapter.
- **Enable SMBAlert** – This check box is used to enable the SMBALERT# option described in the SMBus Specification. When this box is checked, the PMBus Adapter will include an additional output signal (SMBALERT#). The SMBALERT# signal handling is described in the SMBALERT# Signal section of this document. This setting will be reflected in the CAPABILITY command response by the PMBus Adapter.
- **PMBus Bus Speed** – This radio button is used to define the maximum allowed PMBus speed to be reported by the CAPABILITY byte. The PMBus Adapter supports up to 400 kHz operation, but the user may select 100 kHz if they prefer. The actual PMBus operational speed will be determined by the PMBus master SCL output clock.
- **PMBus Slave Address** – The 7-bit PMBus slave address is fully programmable. Addresses reserved by the SMBus are prohibited. This address has no relationship to the MachXO2 EFB I²C address or the ASC I²C addresses, as the PMBus is separate from the ASC I²C bus. See Figure 7 for details.
- **ASC Page Mapping** – This section is used to map the available monitor channels from the design into the available PMBus pages. By default none of the pages are assigned. There are three tabs of available pages (Voltage, Current, Temperature). The available monitor channels and the available page space will be automatically filtered depending on the tab selected. Channels can be mapped to pages by dragging them from the **LOGIC SIGNAL POOL** into the **MONITOR MAPPING** column.

Ports & Nodes View – PMBus Tab

After enabling the PMBus Adapter, the Ports & Nodes View will be updated to include a **PMBus** tab. This tab lists the PMBus signals which must be managed by the user logic. This is a read-only tab, as shown in Figure 6 below.

Figure 6. Ports & Nodes View - PMBus Tab



	Logical Name / Function	Grouped By	Logic Connection
1	PMBUS_STATUS_WORD_VOUT_OV	PMBUS_STATUS_WORD(7)	OUT
2	PMBUS_STATUS_WORD_IOUT_POUT	PMBUS_STATUS_WORD(6)	OUT
3	PMBUS_STATUS_WORD_INPUT	PMBUS_STATUS_WORD(5)	OUT
4	PMBUS_STATUS_WORD_MFR	PMBUS_STATUS_WORD(4)	OUT
5	PMBUS_STATUS_WORD_POWER_GOOD	PMBUS_STATUS_WORD(3)	OUT
6	PMBUS_STATUS_WORD_FANS	PMBUS_STATUS_WORD(2)	OUT
7	PMBUS_STATUS_WORD_OTHER	PMBUS_STATUS_WORD(1)	OUT
8	PMBUS_STATUS_WORD_UNKNOWN	PMBUS_STATUS_WORD(0)	OUT
9	PMBUS_STATUS_VOUT_OV_FAULT	PMBUS_STATUS_VOUT(7)	OUT
10	PMBUS_STATUS_VOUT_OV_WARNING	PMBUS_STATUS_VOUT(6)	OUT
11	PMBUS_STATUS_VOUT_UV_WARNING	PMBUS_STATUS_VOUT(5)	OUT
12	PMBUS_STATUS_VOUT_UV_FAULT	PMBUS_STATUS_VOUT(4)	OUT
13	PMBUS_STATUS_VOUT_MAX_WARNING	PMBUS_STATUS_VOUT(3)	OUT
14	PMBUS_STATUS_VOUT_TOFF_MAX_FAULT	PMBUS_STATUS_VOUT(2)	OUT
15	PMBUS_STATUS_VOUT_TOFF_MAX_WARNING	PMBUS_STATUS_VOUT(1)	OUT

There are three columns of information for each of the signals listed in the PMBus tab. These are described in the list below.

- **Logical Name / Function** – This displays the name of each signal. The signal names each correspond to status bits or operational requests as defined in the PMBus Command Description portion of this document.
- **Grouped By** – This displays the signal group, where applicable. This is used for each of the STATUS command bytes. The grouped by column displays the name of the STATUS command and the bit position in the given command.
- **Logic Connection** – This displays the direction of the signal, either IN or OUT. The connection direction is given with respect to the user logic. This means that IN signals are driven from the PMBus Adapter to the user logic. OUT signals are driven from the user logic to the PMBus Adapter.

Connecting the PMBus Adapter to the Design

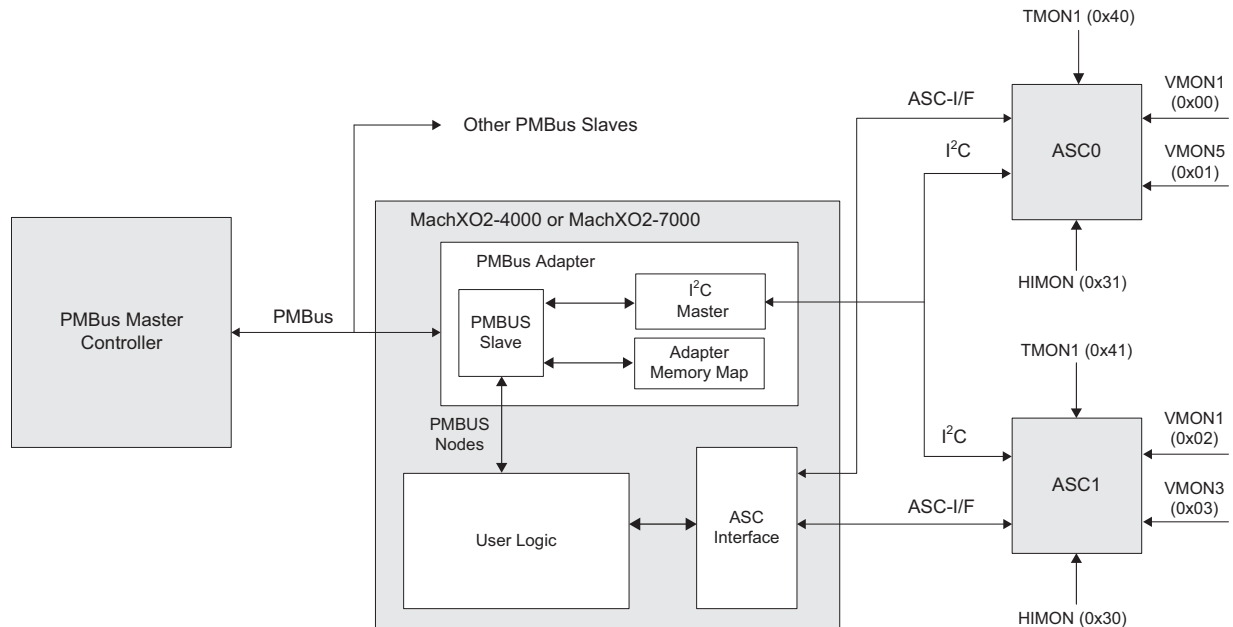
The PMBus Adapter supports the mapping of Analog Point of Load (APOL) power supplies, current sense monitor channels, and temperature sensors to a PMBus network. The PMBus Adapter interprets the PMBus protocol and bridges the command / status / measurement information to the power management system as shown in Figure 7.

Measurements and Fault Limits

Measurement and fault limit information are directly processed by the PMBus Adapter. The PMBus Adapter processes these commands by accessing the ASC I²C bus or reading values from the Adapter memory map as shown in Figure 7. The different voltage, current, and temperature monitors are configured and mapped to PMBus pages in the design phase. The example in Figure 7 shows two ASC devices, with monitors of each type mapped to different pages. The pages are shown in parentheses in the example. See the [PAGE](#) section command description for more details. The PMBus Adapter uses the stored page map to access the correct ASC device and corresponding register when a measurement or fault limit command is received. The fault limit commands will update the ASC configuration registers but not the ASC configuration EEPROM. After power-on-reset the fault limits will be restored to their values as defined in the configuration EEPROM during device programming.

The PMBus Adapter also includes a local memory map. The memory map stores the page map, initial fault limits, coefficient information for current sense monitors, and other project information. At power-on-reset, the memory map is loaded from the non-volatile FPGA configuration memory. The initial fault limits after power-on-reset for each of the channels are based on the Platform Designer project settings defined by the user. The A channel comparator thresholds are always defined as the overvoltage (VMONx_A), overcurrent (HIMON_A) or overtemperature (TMONx_A) fault limits. The B channel comparator thresholds are always defined as the undervoltage (VMONx_B), undercurrent (HIMON_B) or undertemperature (TMONx_B) fault limits. Each time the fault limits are updated over PMBus, the volatile memory map is updated as well. The fault limit commands will not update the non-volatile FPGA configuration memory. The fault limit read commands access the fault limits stored in the memory map, not the ASC configuration registers.

Figure 7. PMBus Adapter - Design Connections (Including Example Page Assignments)



Status Bytes and Operational Commands

Processing the status bytes and operational commands requires user defined logic. The PMBus Adapter presents a set of signals (the PMBus nodes as described in the Platform Designer section) to the user logic. There are three general groups of PMBus nodes as described below:

- Operational Commands
- Status Bytes and Clear Faults Command
- Page Byte

Operational Commands

The operational commands include the signals asserted by the OPERATION command and the MFR_INTERLEAVE_OFF / MFR_INTERLEAVE_ON commands. These are outputs from the PMBus Adapter and they need to be serviced by the user logic. A possible implementation is shown in the Logic view in Figure 8 below. The voltage rail sequencing waits in Step 2 for the PMBUS_ON_MARGIN_OFF signal to be asserted as a result of the OPERATION command to begin sequencing. Once all voltage rails are enabled, the sequencer waits in Step 6 for either a SOFT_OFF command or a local voltage fault. If either of these conditions is detected, the sequencer will sequence the rails off. An IMMEDIATE_OFF command from the adapter will generate an exception in the user logic, causing the rails to be shutdown immediately. The implementation of the OPERATION handling in user logic offers the benefit of mixing local fault handling with the PMBus protocol based fault handling.

Figure 8. Logic Sequence Example: Handling OPERATION ON and OFF Signals

Step	Sequence Instruction	Outputs	Interruptible	Comment
SM0 Step 0	Begin Startup Sequence		no	
SM0 Step 1	ENABLE_POWER_SUPPLY_1 = 0 ENABLE_POWER_SUPPLY_2 = 0 ENABLE_POWER_SUPPLY_3 = 0		no	Startup state with all power supplies disabled
SM0 Step 2	Wait for PMBUS_ON_MARGIN_OFF		no	Wait for ON command from PMBUS OPERATION byte
SM0 Step 3	Wait for 100ms using PowerSupplyTimer	ENABLE_POWER_SUPPLY_1 = 1	yes	Sequence On Power Supplies
SM0 Step 4	Wait for 100ms using PowerSupplyTimer	ENABLE_POWER_SUPPLY_2 = 1	yes	Sequence On Power Supplies
SM0 Step 5	Wait for 100ms using PowerSupplyTimer	ENABLE_POWER_SUPPLY_3 = 1	yes	Sequence On Power Supplies
SM0 Step 6	Wait for VOLTAGE_FAULT_SUPPLY_1_2_3 OR PMBUS_SOFT_OFF		yes	Wait for either a locally detected fault, or a SOFT_OFF command from the OPERATION command
SM0 Step 7	Wait for 100ms using PowerSupplyTimer	ENABLE_POWER_SUPPLY_1 = 1	yes	Sequence Off Power Supplies
SM0 Step 8	Wait for 100ms using PowerSupplyTimer	ENABLE_POWER_SUPPLY_2 = 1	yes	Sequence Off Power Supplies
SM0 Step 9	Wait for 100ms using PowerSupplyTimer	ENABLE_POWER_SUPPLY_3 = 1	yes	Sequence Off Power Supplies
SM0 Step 10	Halt (end-of-program)		no	

Exception ID	Boolean Expression	Outputs	Exception Handler	Comment
E0	IF PMBUS_IMMED_OFF		Start at step 1	OPERATION Immediate_Off will jump to startup state with all supplies disabled

The PMBUS_MARGIN_HI_IF, PMBUS_MARGIN_HI_AF, PMBUS_MARGIN_LOW_IF, and PMBUS_MARGIN_LOW_AF signals need to be associated to the Power Supply Margining control signals. One implementation is shown in the supervisory equations in Figure 9 below:

Figure 9. Logic Equation Example: Handling OPERATION MARGIN Signals

EQ4	ASC0_TRIM1_P0.D = PMBUS_MARGIN_LOW_IF OR PMBUS_MARGIN_LOW_AF	Node, Registered	Set to Voltage Profile 1 (Low) when PMBUS OPERATION received
EQ5	ASC0_TRIM1_P1.D = PMBUS_MARGIN_HI_IF OR PMBUS_MARGIN_HI_AF	Node, Registered	Set to Voltage Profile 2 (High) when PMBUS OPERATION received

The OPERATION bits also influence the error handling behavior of the user logic. PMBUS_MARGIN_HI_AF and PMBUS_MARGIN_LOW_AF are states which require that the user logic ignore over (HI) or under (LOW) voltage faults. An example is shown in Figure 10 below.

Figure 10. Logic Equation Example: Fault Handling based on OPERATION Ignore Fault Signals

EQ6	OVERVOLTAGE_FAULT_SUPPLY_1_2_3.D = NOT PMBUS_MARGIN_HI_AF AND (ASC0_VMON1_A_SUPPLY1_OV OR ASC0_VMON2_A_SUPPLY2_OV OR ASC0_VMON3_A_SUPPLY3_OV)	Node, Registered	Set Overvoltage Fault node dependent on PMBUS OPERATION state
-----	---	------------------	---

The PMBUS_INTERLEAVE_ON_OFF signal is designed to interface the PMBus Adapter with the SYNC inputs of connected APOLs. The PMBUS_INTERLEAVE_ON_OFF signal can be used to turn on an interleave function, which drives different SYNC inputs with phase-shifted clock signals to minimize supply current spikes. This is a simplified approach compared to the INTERLEAVE command defined in the PMBus Protocol Specification. It is up to the imported user logic to define the number of phases, connections, and frequency of the SYNC clock. The PMBUS_INTERLEAVE_ON_OFF signal can be input to this custom logic block using the Platform Designer Imported HDL feature. See TN1287, [Importing HDL Files with Platform Manager 2](#) for more details.

Status Bytes and Clear Faults Command

The Status Bytes include all the commands displayed in Figure 3. The STATUS_BYTE and STATUS_WORD provide fault summary information, while each of the additional status bytes provide more detail into the fault condition.

Each of the Status Bytes (with the exception of the STATUS_CML byte), need to be handled by the user logic. The user logic is responsible for the following:

- Setting and latching the individual status bits depending on the system conditions
- Mapping the status bytes to the summary STATUS_BYTE and STATUS_WORD bits
- Clearing the status bits when the CLEAR_FAULTS command is received
- Asserting the PMBUS_SMBALERT signal (optional)

The recommended implementation for managing these requirements is to set the PMBus status bits in supervisory logic equations. Each status bit supported by the design can be handled with a three equation set, as shown in EQ0, EQ1, and EQ2 in Figure 11 below.

Figure 11. Status Bytes - Bit Handling with Supervisory Equations

EQ0	PMBUS_STATUS_INPUT_VIN_UV_FAULT.D = PMBUS_STATUS_INPUT_VIN_UV_FAULT	Output, Registered	Latch INPUT_VIN_UV_FAULT bit
EQ1	PMBUS_STATUS_INPUT_VIN_UV_FAULT.AR = PMBUS_CLEAR_FAULTS	Output, Registered	Reset INPUT_VIN_UV_FAULT bit based on PMBUS_CLEAR_FAULTS signal
EQ2	PMBUS_STATUS_INPUT_VIN_UV_FAULT.AP = NOT ASCO_VMON1_B	Output, Registered	Set INPUT_VIN_UV_FAULT bit based on monitor alarm signal
EQ3	PMBUS_STATUS_BYTE_VIN_UV.D = PMBUS_STATUS_INPUT_VIN_UV_FAULT	Output, Registered	Map INPUT_VIN_UV_FAULT to BYTE_VIN_UV

- EQ0 latches the INPUT_VIN_FAULT bit – this is required to keep the fault bit sticky as defined by the PMBus specification.
- EQ1 resets the INPUT_VIN_FAULT bit upon receipt of the CLEAR_FAULTS command.
- EQ2 sets the INPUT_VIN_FAULT bit when the VMON1_B monitor signal is equal to zero (an undervoltage condition)

The STATUS_BYTE and STATUS_WORD summarize the fault conditions of the other status commands. EQ 3 in Figure 11 shows an example of mapping the INPUT_VIN_UV_FAULT to the BYTE_VIN_UV bit. This follows the behavior defined in Figure 3. Other STATUS_BYTE and STATUS_WORD bits may be dependent on the condition of multiple status bits, as shown in Figure 12 below. EQ 7 shows the STATUS_BYTE_TEMPERATURE bit as an example. This bit is dependent on the state of the four bits used in the STATUS_TEMPERATURE command.

Figure 12. Mapping Multiple Status Bits to STATUS_BYTE

EQ7	PMBUS_STATUS_BYTE_TEMPERATURE.D = PMBUS_STATUS_TEMP_UT_FAULT OR PMBUS_STATUS_TEMP_UT_WARNING OR PMBUS_STATUS_TEMP_OT_WARNING OR PMBUS_STATUS_TEMP_OT_FAULT	Output, Registered	Map STATUS_TEMP bits to STATUS_BYTE TEMPERATURE bit
-----	--	--------------------	--

Many designs will not use all the supported bits in the STATUS commands. Unassigned STATUS command bits will be automatically tied to 0 by the design software. Only those bits that will be used in the design need to be assigned using the equation scheme above.

Other methods for assigning the STATUS bits may be used. The most important requirements are to latch the STATUS bit and to properly respond to the PMBUS_CLEAR_FAULTS bit. As mentioned in the CLEAR_FAULTS command section, this signal is provided to the user logic as a short pulse (~100 us) when the CLEAR_FAULTS command is received.

Asserting the PMBUS_SMBALERT signal can also be managed using a supervisory logic equation. This provides a wide degree of flexibility in determining which conditions should assert the SMBALERT# signal. An example is shown in EQ 8 in Figure 13 below.

Figure 13. PMBUS_SMBALERT Handling with Supervisory Equation

EQ8	PMBUS_SMBALERT.D = PMBUS_STATUS_BYTE_VOUT_OV OR PMBUS_STATUS_BYTE_IOUT_OC OR PMBUS_STATUS_BYTE_VIN_UV OR PMBUS_STATUS_BYTE_TEMPERATURE	Output, Registered	Assert SMBALERT signal based on STATUS_BYTE faults
-----	--	--------------------	--

PAGE_BYTE

The PAGE command is used by the PMBus Adapter to specify the active voltage, current sense, or temperature monitor for the measurement and fault limit commands. The active page byte is also provided as an input to the user logic. Depending on the system requirements, the page byte can be used to report the STATUS bytes on a page specific basis rather than the default global basis. This is an optional implementation which adds complexity to the user logic in exchange for more detail in the status byte reporting.

Figure 14 shows one possible implementation for page based status bits using supervisory logic equations. In this simplified example, PAGE 0x00 is assigned to ASC0_VMON1, PAGE 0x01 is assigned to ASC0_VMON2, and PAGE 0x02 is assigned to ASC0_VMON3. EQ 9 checks the active PAGE_BYTE value, and then assigns the STATUS_VOUT_OV_FAULT based on the mapped voltage page.

Figure 14. PAGE Based Status Bits in Supervisory Equations

EQ9	PMBUS_STATUS_VOUT_OV_FAULT.D=((PMBUS_PAGE_BYTE='H00) AND ASC0_VMON1_A_SUPPLY1_OV) OR ((PMBUS_PAGE_BYTE='H01) AND ASC0_VMON2_A_SUPPLY2_OV) OR ((PMBUS_PAGE_BYTE='H02) AND ASC0_VMON3_A_SUPPLY3_OV)	Output, Registered	Assign VOUT_OV_FAULT based on SUPPLY 1 if PAGE 0x00 is active, or SUPPLY 2 if PAGE 0x01 is active, or SUPPLY 3 if PAGE 0x02 is active
-----	--	--------------------	---

This type of page based status byte handling can be implemented for any portion of the status bits, depending on the system requirement.

External Connections

Completing the design of a PMBus slave based on MachXO2 requires a set of external connections as displayed in Figure 15. There are connections required between the PMBus master controller and the MachXO2, as well as connections required between MachXO2 and the ASC hardware management expanders. The connections displayed below are for a system with two ASC devices. Note that Figure 15 does not include any passive components (such as pull-up resistors or noise filters, and the like). See TN1225, [Platform Manager 2 Hardware Checklist](#) for information on components required for these connections. The connections are described in Table 19.

Figure 15. External Connections for PMBUS Slave Based on MachXO2 and Two ASC Devices

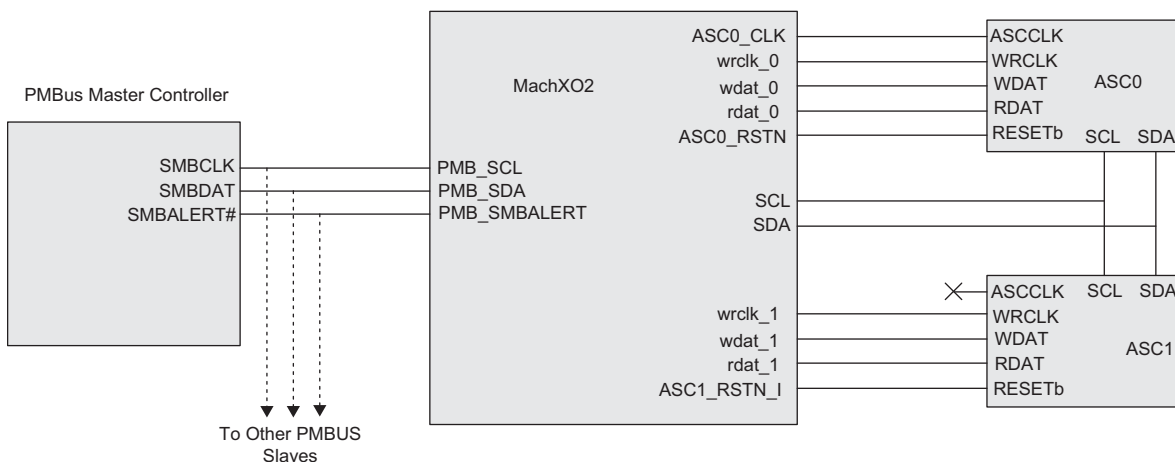


Table 19. PMBus Adapter External Connections

Pin Name	Description
PMBus Connections¹	
PMB_SCL	Connect to SMBCLK pin of PMBus Master Controller. This pin is user-assignable
PMB_SDA	Connect to SMBDAT pin of PMBus Master Controller. This pin is user-assignable
PMB_SMBALERT	Connect to SMBALERT# pin of PMBus Master Controller. This pin is user-assignable. This connection is optional.
ASC Connections²	
ASC0_CLK	Connect to ASCCLK pin of ASC0. This pin is user-assignable. This signal should be assigned to a primary clock input.
wrclk_0	Connect to WRCLK pin of ASC0. This pin is user-assignable
wdat_0	Connect to WDAT pin of ASC0. This pin is user-assignable
rdat_0	Connect to RDAT pin of ASC0. This pin is user-assignable
ASC0_RSTN	Connect to RESETb pin of ASC0. This pin is user-assignable
SCL	Connect to SCL pins of all ASCs. This pin location is locked.
SDA	Connect to SDA pins of all ASCs. This pin location is locked
wrclk_1	Connect to WRCLK pin of ASC1. This pin is user-assignable
wdat_1	Connect to WDAT pin of ASC1. This pin is user-assignable
rdat_1	Connect to RDAT pin of ASC1. This pin is user-assignable
ASC1_RSTN_I	Connect to RESETb pin of ASC1. This pin is user-assignable. This pin is for optional ASCs only.

1. For more details on the PMBus connections, see the SMBus specification at SMBUS.org

2. For more details on the ASC connections, see TN1225, [Platform Manager 2 Hardware Checklist](#).

Summary

The PMBus Adapter component for hardware management controllers based on MachXO2 supports a defined set of commands as described in this technical note. The steps to implement a full PMBus slave design by configuring the PMBus Adapter and connecting it to user defined logic have been presented.

References

- Platform Designer User Guide
- DS1042, [L-ASC10 Data Sheet](#)
- TN1225, [Platform Manager 2 Hardware Checklist](#)
- TN1287, [Importing HDL Files with Platform Manager 2](#)
- PMBus Homepage: www.pmbus.org
- SMBus Homepage: www.smbus.org

Technical Support Assistance

e-mail: techsupport@latticesemi.com

Internet: www.latticesemi.com

Revision History

Date	Version	Change Summary
August 2014	1.0	Initial release.