PM BUS communication setup

A guideline with examples for correct hardware setup of PM Bus.

WP-2005

Contents

Scope

This application note should help design engineers to setup the proper electrical interface between the power supply and the entire system using the Power Management Bus ("PMBus"). It is a quick guide to prepare the required connections at the system level. The examples listed below are based on the GU300 family power supplies from SL Power Electronics, which were designed to meet standardized requirements for a PMBus interface. Detailed full information can be found in "PMBus™ Power System Management Protocol Specification Part I – General Requirements, Transport and Electrical Interface" and Part II – Command Language (www.pmbus.org)

Quick Facts to PMBus

- PMBus interface derives from SMBus which comes from I²C physical layer. PMBus is electrically compatible with I²C. Users who are familiar with I²C protocol will find differences in the data content transferred from device to device and additional prescribed commands used for power management units.
- The GU300, as most power supplies with PMBus capabilities on the market can be used as "slave" only. The entire system must have a "master" device which will send/receive data to/from all slaves on the shared communication bus.
- Each slave unit has a 7-Bit address to be accessed at, two or more devices with same address are not permitted on same bus.
- Not all possible PMBus commands are supported by all power supplies. The variety of existing devices on the market shows limited set of available commands for the selected power supply.
- The entire list of all existing commands is listed in "PMBus™ Power System Management Protocol Specification Part II – Command Language"
- The GU300 Unit is configured to be controlled by serial communication.

Color codes used in this app note:

COLOR CODE	DEFINITION
🟩	START/RESTART
🟦	SLAVE ADDRESS (7-BIT)
🟦	READ (1) / WRITE (0) BIT
🟦	ACKNOWLEDGE (ACK) OR NOT-ACKNOWLEDGE (NACK)
🟧	COMMAND
⬛	I²C CLOCK PULLED LOW
🟨	DATA SEND OR RECEIVED BY MASTER
🟥	STOP

Hardware setup

- Use general safety precautions while preparing the setup
- PMBus uses the same 3-wire connection as with I²C; make the power supply (I²C Slave) connections to the I²C master devise as per Table 1

Description	Connector on PSU	Connector on Customer Side
I2C Clock / SCL	J300-12	I2C Master Clock Line
I2C Data / SDA	J300-10	I2C Master Data Line
Return	J300-11	I2C Master Return

Table 1: PMBus/I²C Connections

- The length of the PMBus connection cable should be below 1 meter (~40 Inch), the recommended length is less than 50 cm (~20 Inch) to ensure stable communication.
- DC output return and PMBus connections share the same output return. For series connection of the power supply units or for negative output voltage setup, additional isolation is required as the PMBus hardware is referenced to the output return. Contact your local application engineering team for support.
- Ensure correct pull ups to Vdd on master – slave communication lines. The power supply (slave) has internal 20kOhm pull up resistors. The recommended values of the pull up resistors on the system side (master) are 10kOhm as shown on figure 1. Be sure not to override the maximum allowed current of 4mA if pull up resistors with other values are used.
- Pins J300-3 and J300-4 must be shorten to activate the power supply (logic LOW)

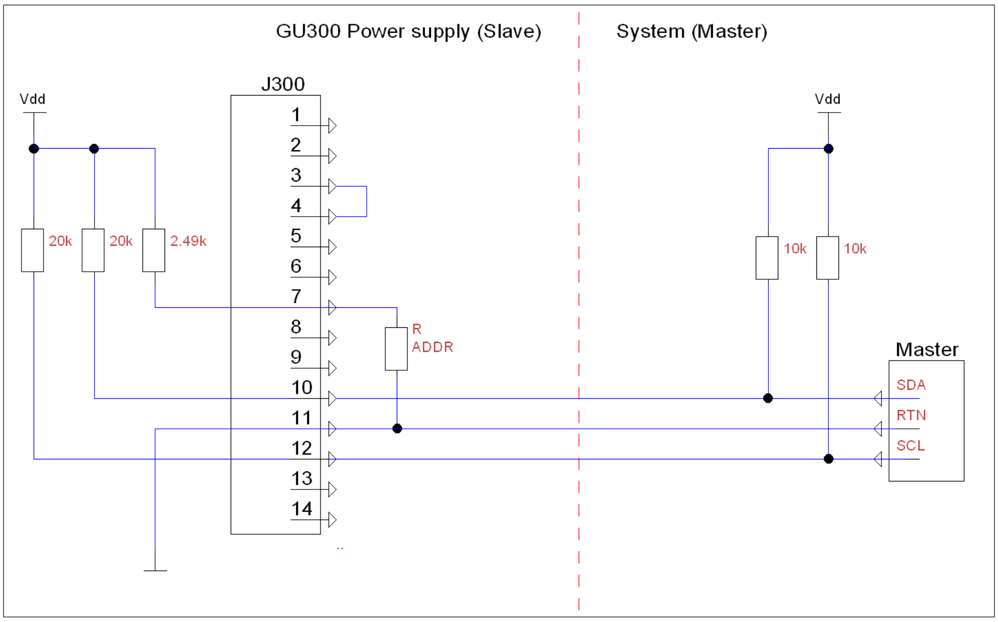


Figure 1: PMBus Master - Slave connection setup

- The recommended pull-up voltage for I²C lines is 3.3 V (V_{DD}).
- Capacitive impedance on communication line may create delayed logical state change. If the pull up resistors are too high, the logical “high” may not reach the signal threshold and so not be recognized correctly.
- The recommended communication bitrate is 100kHz
- The communication time window is limited, the data transfer including the stop bit must be completed within 25ms time frame.

- The GU300 power supply unit has a default 7-bit slave address of 0x58 if there is no connection on pin J300-7. To change the address, select the recommended R_{ADDR} from Table 2 and connect it between ADDR _ MODE (pin J300-7) and Return (pin J300-11).

  Note: The address assignment takes place only during the start-up. The only way to change the address is to disconnect the PSU from AC source and let it discharge completely (The LED's on digital card will stop glowing as indication for power loss).

ADDR VOLTAGE	R_{ADDR} (Ω)	SLAVE ADDRESS (7-Bit)	Write Slave Address	Read Slave Address
0.0V – 0.4V	<350	58	B0	B1
0.4V-0.8V	549	59	B2	B3
0.8V-1.2V	1100	5A	B4	B5
1.2V-1.6V	1820	5B	B6	B7
1.6V-2.0V	3010	5C	B8	B9
>2.0V	Open	58	B0	B1

Table 2: PMBus (I^2C) slave address selection table

PMBus Communication Protocol - examples

PMBus: WRITE to slave

Following steps are required to write any data from master device to the slave unit via PMBus protocol:

a. Send out start condition
b. Send out the 8-bit write slave address, or 7-bit slave address followed by 0.
c. The slave corresponding to that address will send out an acknowledgement (ACK)
d. After receiving the acknowledgement, send out the required 8-bit I^2C Command.
e. If the command type has data associated with it which needs to be send out to slave, send the data in the packets of 8-bits. Each 8-bits packet is followed by acknowledgement from slave.
f. After sending the data, send out the stop command.

The visual description of the PMBus Write command package is shown in Figure 2:

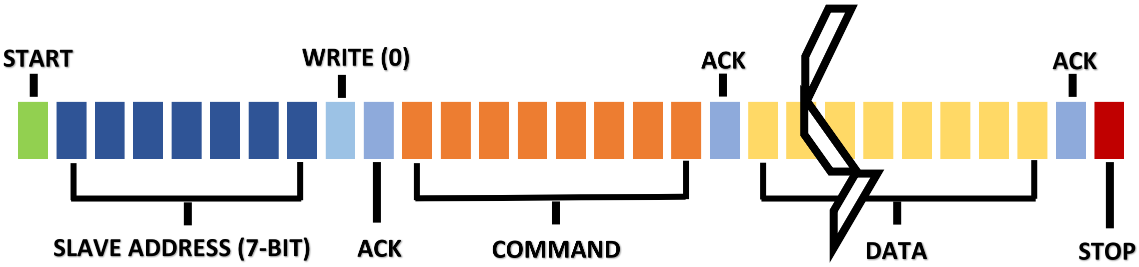


Figure 2: PMBus data package for "WRITE"

Example 1: Turn Power supply off

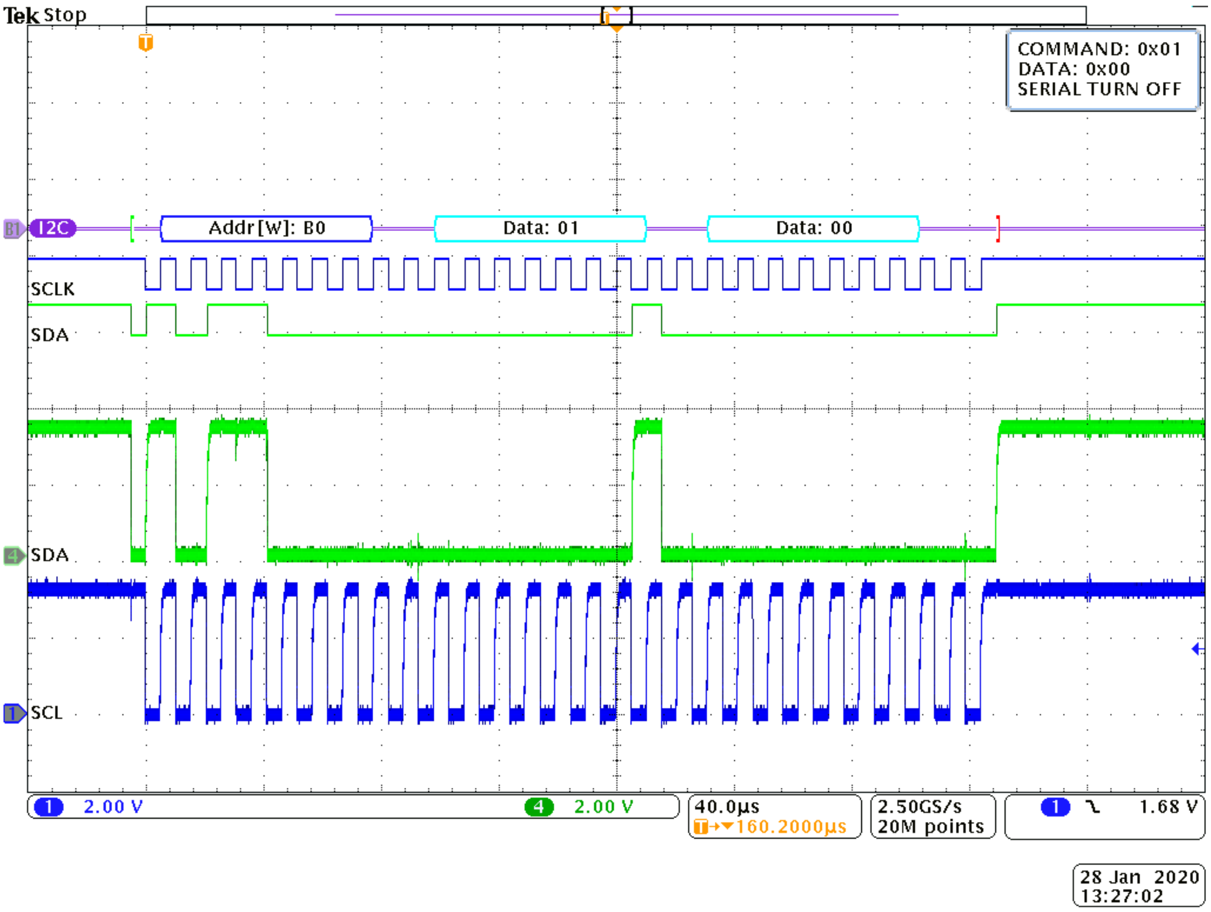
To turn off the power supply via I²C PMBus, a command 0x01 needs to be communicated to the dedicated slave module followed by data packet 0x00. Assuming the GU300 unit is configured to the default address of 0x58 the slave write address will be according to Table 2: 0xB0. The entire command line is illustrated below on Figure 3. Figure 4 shows also the scope screenshot with the entire command.

Slave Write Address: 0xB0
Command: 0x01
Data: 0x00

M	M	M	S	M	S	M	S	M
START	0x58	0	ACK	0x01	ACK	0x00	ACK	STOP

M	M							M	S	M								S	M								S	M
S	1	0	1	1	0	0	0	0	A	0	0	0	0	0	0	0	1	A	0	0	0	0	0	0	0	0	A	P

Figure 3: Example for I²C Command 0x01 "Turn Off"



PMBUS COMMAND: 0x01 Turn Off

Figure 4: Scope screenshot I²C Command 0x01 "Turn Off"

Example 2: Turn Power supply on

To turn on the power supply via I²C PMBus, a command 0x01 need to be communicated to the dedicated slave module, but this time followed by data packet 0x80. Assuming the GU300 unit is configured to the default address of 0x58 the slave write address will be according to Table 2: 0xB0. The entire command line is illustrated below on Figure 5. Figure 6 shows also the scope screenshot with the entire command.

Slave Write Address: 0xB0
Command: 0x01
Data: 0x80

M	M	M	S	M	S	M	S	M
START	0x58	0	ACK	0x01	ACK	0x80	ACK	STOP

M	M							M	S	M								S	M								S	M
S	1	0	1	1	0	0	0	0	A	0	0	0	0	0	0	0	1	A	1	0	0	0	0	0	0	0	A	S

Figure 5:Example for I²C Command 0x01 "Turn On"

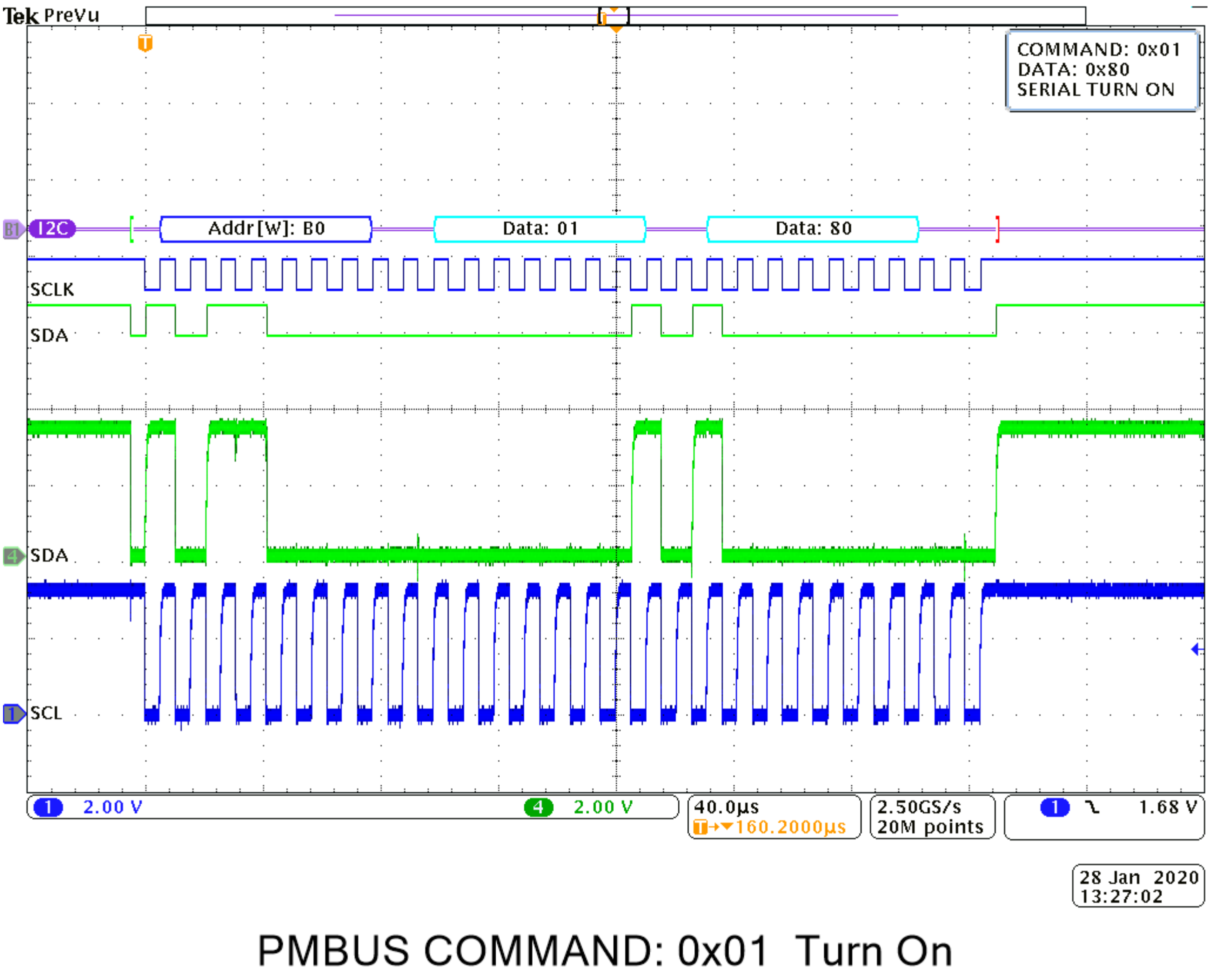


Figure 6: Scope screenshot I²C Command 0x01 "Turn On"

Example 3: Clear All Faults

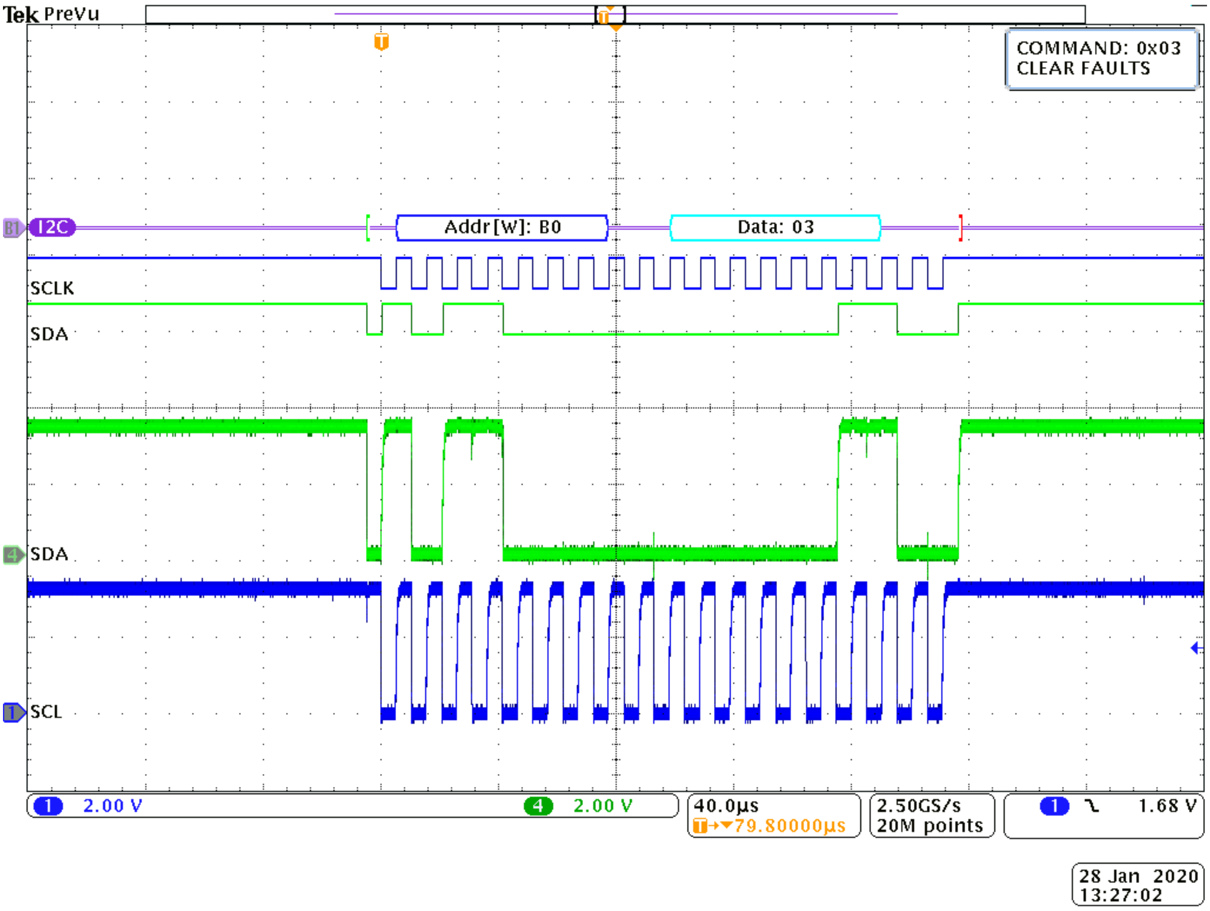
To clear all faults of the power supply's SMBAlert signal (Pin J300-6) via I²C PMBus, a command 0x03 need to be sent to the dedicated slave module. No data is required for this command. Assuming the GU300 unit is configured to the default address of 0x58 the slave write address will be according to Table 2: 0xB0. The entire command line is illustrated below on Figure 7. Figure 8 shows also the scope screenshot with the entire command.

Slave Write Address: 0xB0
Command: 0x03
Data: --

M	M	M	S	M	S	M
START	0x58	0	ACK	0x03	ACK	STOP

M	M							M	S	M								S	M
S	1	0	1	1	0	0	0	0	A	0	0	0	0	0	0	1	1	S	S

Figure 7: Example for I²C Command 0x03 "Clear all Faults"



PMBUS COMMAND: 0x03

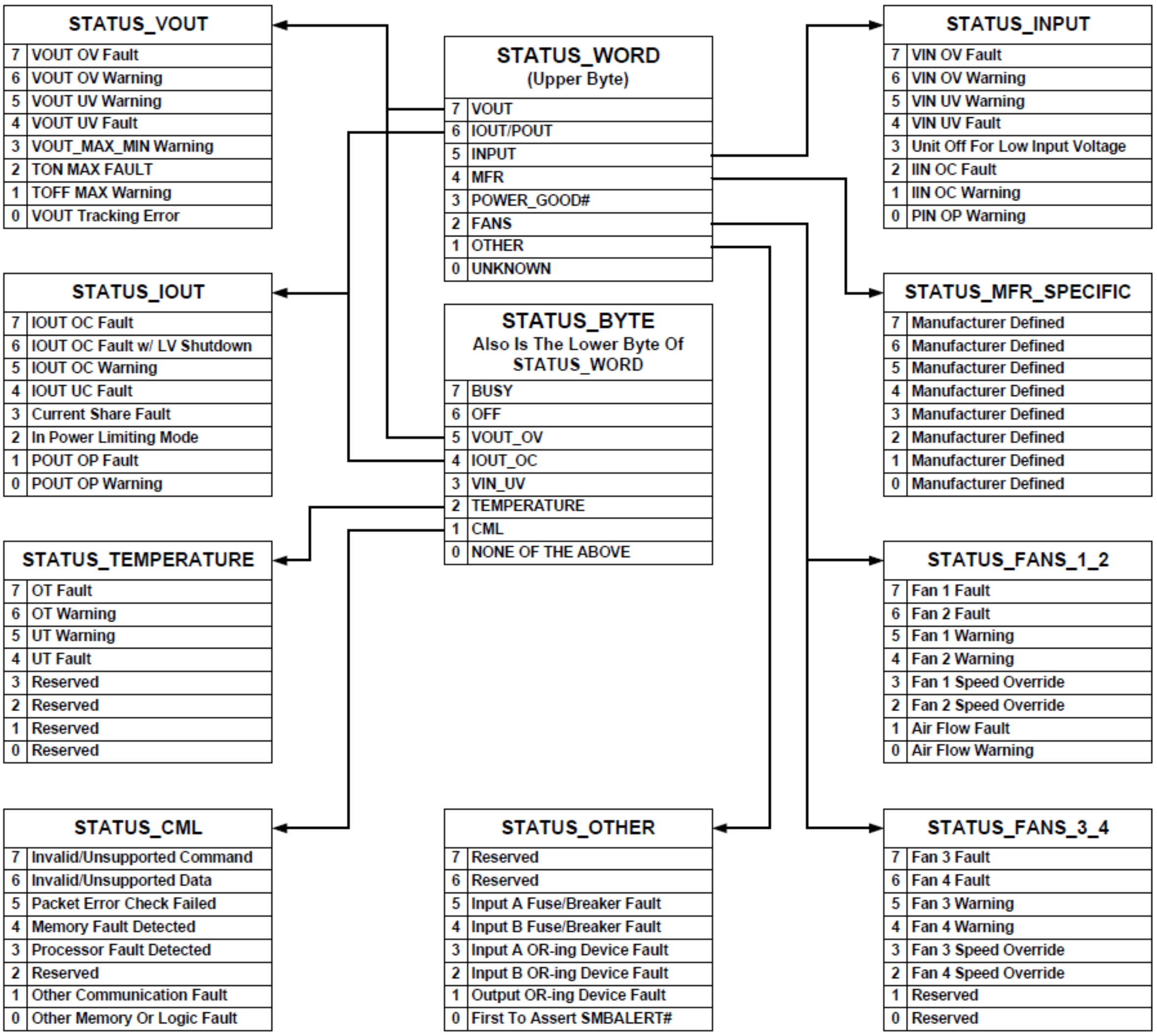
Figure 8: Scope screenshot I²C Command 0x03 "Clear All Faults"

Faults table:

Table 3 shows all possible fault flags stored in STATUS_BYTE. All green marked flags can be cleared at once by sending the command 0x03 to the slave unit. Figure 9 shows the connection between these flags and other status registers of the slave unit.

STATUS	BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
LO_STAT	BUSY	OFF	VOUT_OV_FLT	IOUT_OC_FLT	VIN_UV_FLT	TEMPERATURE	CML	NONE OF ABOVE
HI_STAT	VOUT	IOUT/POUT	INPUT	MFR_SPECIFIC	POWER_GOOD#	FANS	OTHER	UNKNOWN
VO_STAT	VOUT_OV_FAULT	VOUT_OV_WARNING	VOUT_UV_WARNING	VOUT_UV_FAULT	VOUT_MAX_WARNING	TON_MAX_FAULT	TOFF_MAX_Warning	VOUT Tracking Error
IO_STAT	IOUT_OC_FAULT	IOUT_OC_LV_FAULT	IOUT_OC_WARNING	IOUT_UC_FAULT	Current Share Fault	In Power Limit Mode	POUT_OP_FAULT	POUT_OP_WARNING
TEMP_STAT	OT_FAULT	OT_WARNING	UT_WARNING	UT_FAULT	-	-	-	-
CML_STAT	Invalid Command	Invalid Data	Packet Error chk Fail	Memory Fault	Processor Fault	-	Other Comm Fault	Other memory fault
FAN12_STAT	Fan 1 Fault	-	Fan1 Warning	-	Fan 1 Speed Override	-	Air flow fault	Air flow warning
iError	'0' => 0x30							
SMBAlert	Low on SMB Alert pin							

Table 3: Faults stored in STATUS BYTE (lower Byte of STATUS_WORD)



STATUS_VOUT	
7	VOUT OV Fault
6	VOUT OV Warning
5	VOUT UV Warning
4	VOUT UV Fault
3	VOUT_MAX_MIN Warning
2	TON MAX FAULT
1	TOFF MAX Warning
0	VOUT Tracking Error

STATUS_WORD (Upper Byte)	
7	VOUT
6	IOUT/POUT
5	INPUT
4	MFR
3	POWER_GOOD#
2	FANS
1	OTHER
0	UNKNOWN

STATUS_INPUT	
7	VIN OV Fault
6	VIN OV Warning
5	VIN UV Warning
4	VIN UV Fault
3	Unit Off For Low Input Voltage
2	IIN OC Fault
1	IIN OC Warning
0	PIN OP Warning

STATUS_IOUT	
7	IOUT OC Fault
6	IOUT OC Fault w/ LV Shutdown
5	IOUT OC Warning
4	IOUT UC Fault
3	Current Share Fault
2	In Power Limiting Mode
1	POUT OP Fault
0	POUT OP Warning

STATUS_BYTE Also Is The Lower Byte Of STATUS_WORD	
7	BUSY
6	OFF
5	VOUT_OV
4	IOUT_OC
3	VIN_UV
2	TEMPERATURE
1	CML
0	NONE OF THE ABOVE

STATUS_MFR_SPECIFIC	
7	Manufacturer Defined
6	Manufacturer Defined
5	Manufacturer Defined
4	Manufacturer Defined
3	Manufacturer Defined
2	Manufacturer Defined
1	Manufacturer Defined
0	Manufacturer Defined

STATUS_TEMPERATURE	
7	OT Fault
6	OT Warning
5	UT Warning
4	UT Fault
3	Reserved
2	Reserved
1	Reserved
0	Reserved

STATUS_FANS_1_2	
7	Fan 1 Fault
6	Fan 2 Fault
5	Fan 1 Warning
4	Fan 2 Warning
3	Fan 1 Speed Override
2	Fan 2 Speed Override
1	Air Flow Fault
0	Air Flow Warning

STATUS_CML	
7	Invalid/Unsupported Command
6	Invalid/Unsupported Data
5	Packet Error Check Failed
4	Memory Fault Detected
3	Processor Fault Detected
2	Reserved
1	Other Communication Fault
0	Other Memory Or Logic Fault

STATUS_OTHER	
7	Reserved
6	Reserved
5	Input A Fuse/Breaker Fault
4	Input B Fuse/Breaker Fault
3	Input A OR-ing Device Fault
2	Input B OR-ing Device Fault
1	Output OR-ing Device Fault
0	First To Assert SMBALERT#

STATUS_FANS_3_4	
7	Fan 3 Fault
6	Fan 4 Fault
5	Fan 3 Warning
4	Fan 4 Warning
3	Fan 3 Speed Override
2	Fan 4 Speed Override
1	Reserved
0	Reserved

PMBus: READ from slave

Following steps are required to read any data from master device to the slave unit via PMBus protocol:

a. Send out start condition
b. Send out the 8-bit write slave address, or 7-bit slave address followed by 0.
c. The slave corresponding to that address will send out an acknowledgement (ACK)
d. After receiving the acknowledgement, send out the required 8-bit I2C Command.
e. The slave will again send out an acknowledgment showing it has accepted the command.
f. Send out a restart condition.
g. Send out the 8-bit read slave address, or 7-bit slave address followed by 1.
h. The slave corresponding to that address will send out an acknowledgement.
i. The Clock will go low for a while as slave is processing data.
j. Read the incoming 8-bit data packets from slave.
k. After receiving data packet, send out an acknowledgement if more packets to be received.
l. After receiving the desired number of 8-bit data packets, send out a non-acknowledgement (NACK) followed by a stop condition.

The visual description of the PMBus Write command package is shown in Figure 10:

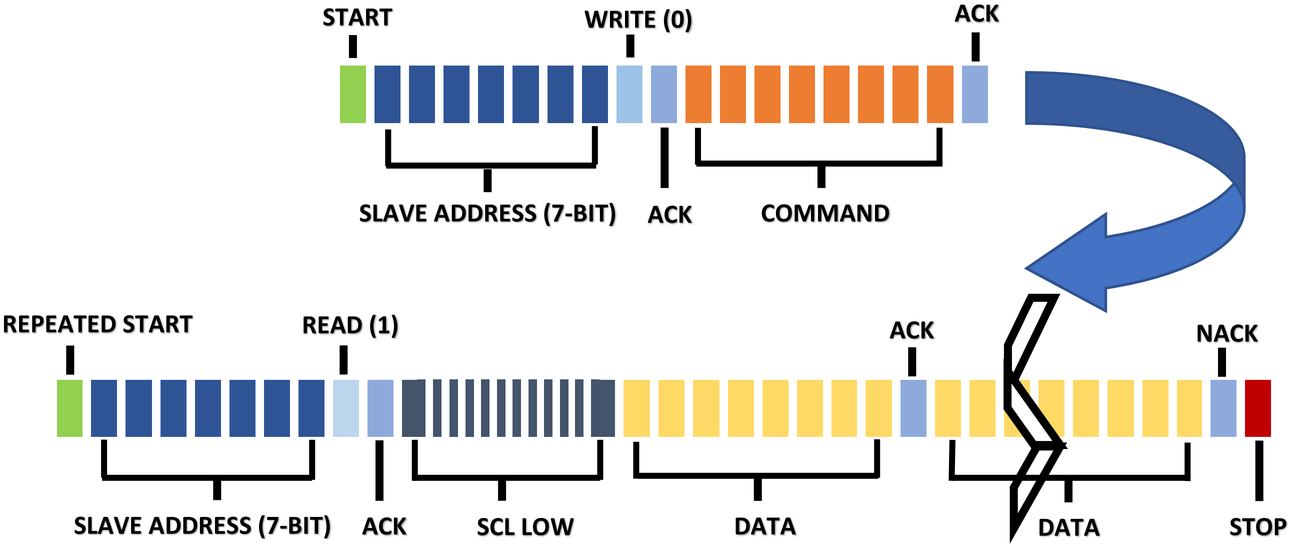
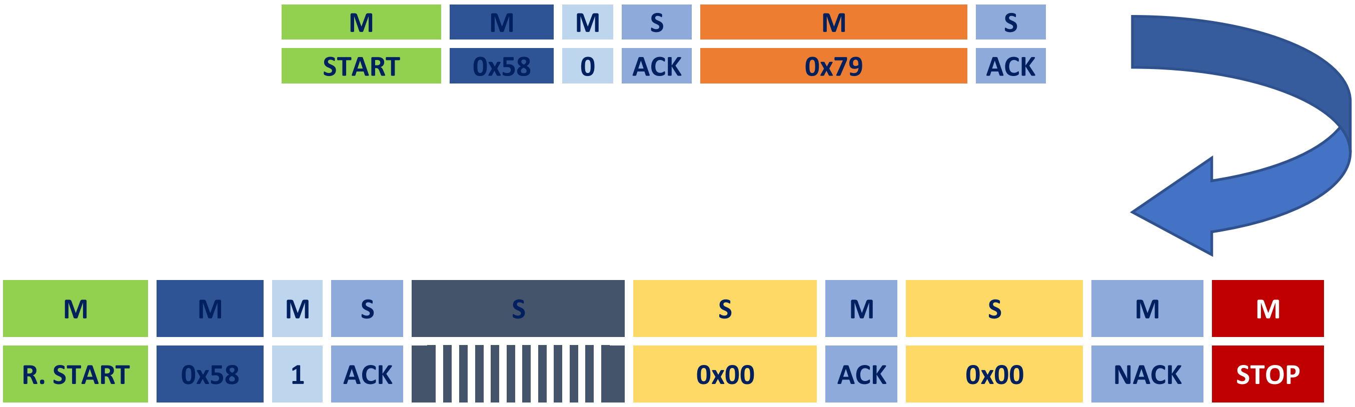


Figure 10:PMBus data package for "READ"

Example 4: Read STATUS_WORD – Slave unit is ON, no fault flags.

Figure 11 shows visual description for STATUS_WORD bytes request and response from a slave unit which is ON and has no fault flags.

Slave Write Address: 0xB0
Command: 0x79
Slave Read Address: 0xB1
Data (Lo Byte): 0x00
Data (Hi Byte): 0x00



M	M	M	S	M	S
START	0x58	0	ACK	0x79	ACK

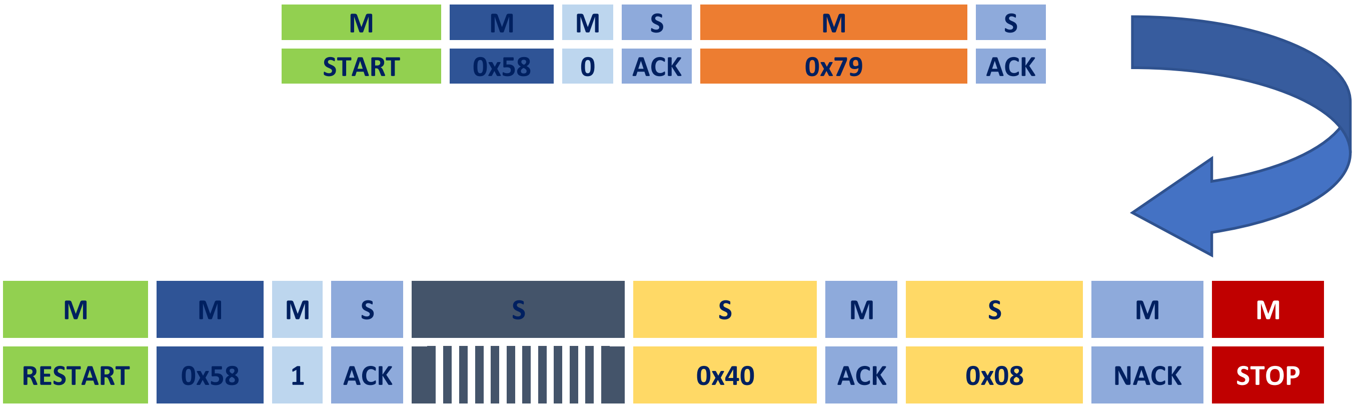
M	M	M	S	S	S	M	S	M	M
R. START	0x58	1	ACK		0x00	ACK	0x00	NACK	STOP

Figure 11: Example for I²C Command 0x79 "Read Status" - Slave=ON

Example 5: Read STATUS_WORD – Slave unit is OFF.

Figure 12 shows visual description for STATUS_WORD bytes request and response from a slave unit which is OFF.

Slave Write Address: 0xB0
Command: 0x79
Slave Read Address: 0xB1
Data (Lo Byte): 0x40
Data (Hi Byte): 0x08



M	M	M	S	M	S
START	0x58	0	ACK	0x79	ACK

M	M	M	S	S	S	M	S	M	M
RESTART	0x58	1	ACK		0x40	ACK	0x08	NACK	STOP

Figure 12: Example for I²C Command 0x79 "Read STATUS_WORD" - Slave=OFF

Example 6: Read STATUS_WORD – Slave has a fault

Figure 13 shows visual description for STATUS_WORD bytes request and response from a slave unit which has a fault.

Slave Write Address: 0xB0
Command: 0x79
Slave Read Address: 0xB1
Data (Lo Byte): 0x51
Data (Hi Byte): 0x48

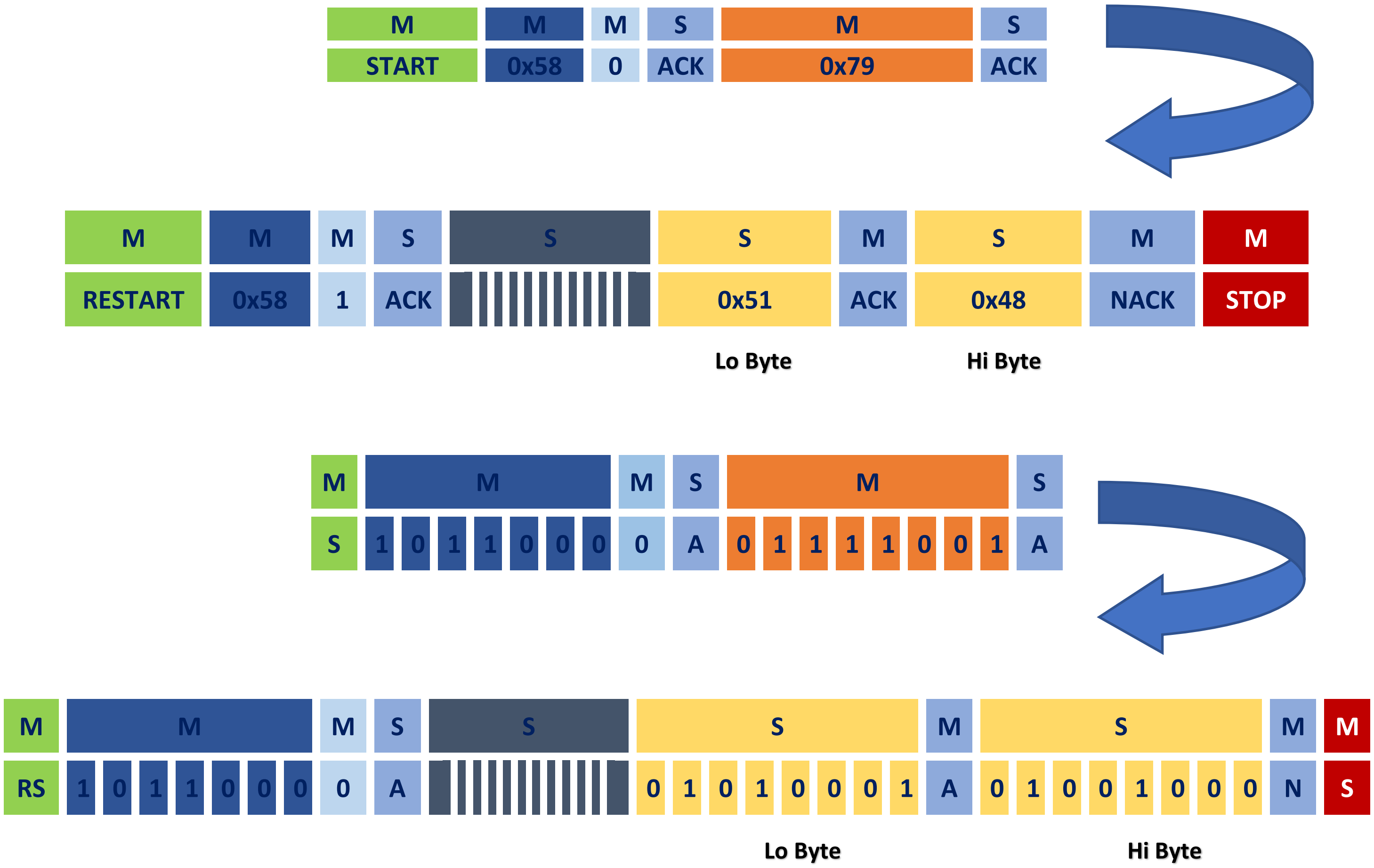


Figure 13: Example for I²C Command 0x79 "Read STATUS_WORD" - Slave=ON, with fault

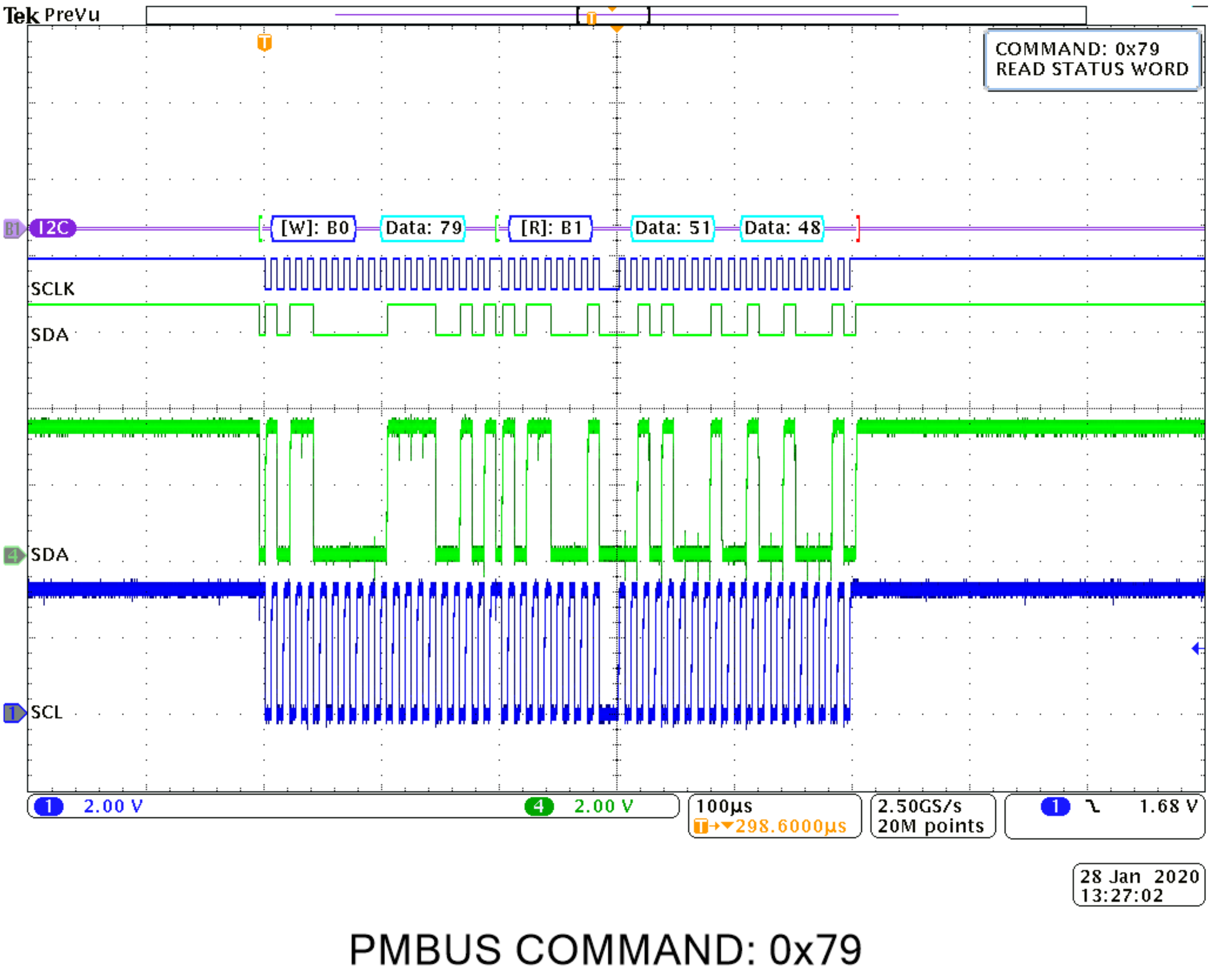


Figure 14:Scope screenshot I²C Command 0x79 "Read STATUS_WORD" - Slave=OFF, with fault 0x51 and 0x48

Table 4 shows the bitmap of upper and lower Bytes of STATUS_WORD to decode the received response 0x51 and 0x48:

STATUS	BIT 15	BIT 14	BIT 13	BIT 12	BIT 11	BIT 10	BIT 9	BIT 8
LO_STAT	BUSY	OFF	VOUT_OV_FLT	IOUT_OC_FLT	VIN_UV_FLT	TEMPERATURE	CML	NONE OF ABOVE
VALUE	0	1	0	1	0	0	0	1
STATUS	**BIT 7**	**BIT 6**	**BIT 5**	**BIT 4**	**BIT 3**	**BIT 2**	**BIT 1**	**BIT 0**
HI_STAT	VOUT	IOUT/POUT	INPUT	MFR_SPECIFIC	POWER_GOOD#	FANS	OTHER	UNKNOWN
VALUE	0	1	0	0	1	0	0	0

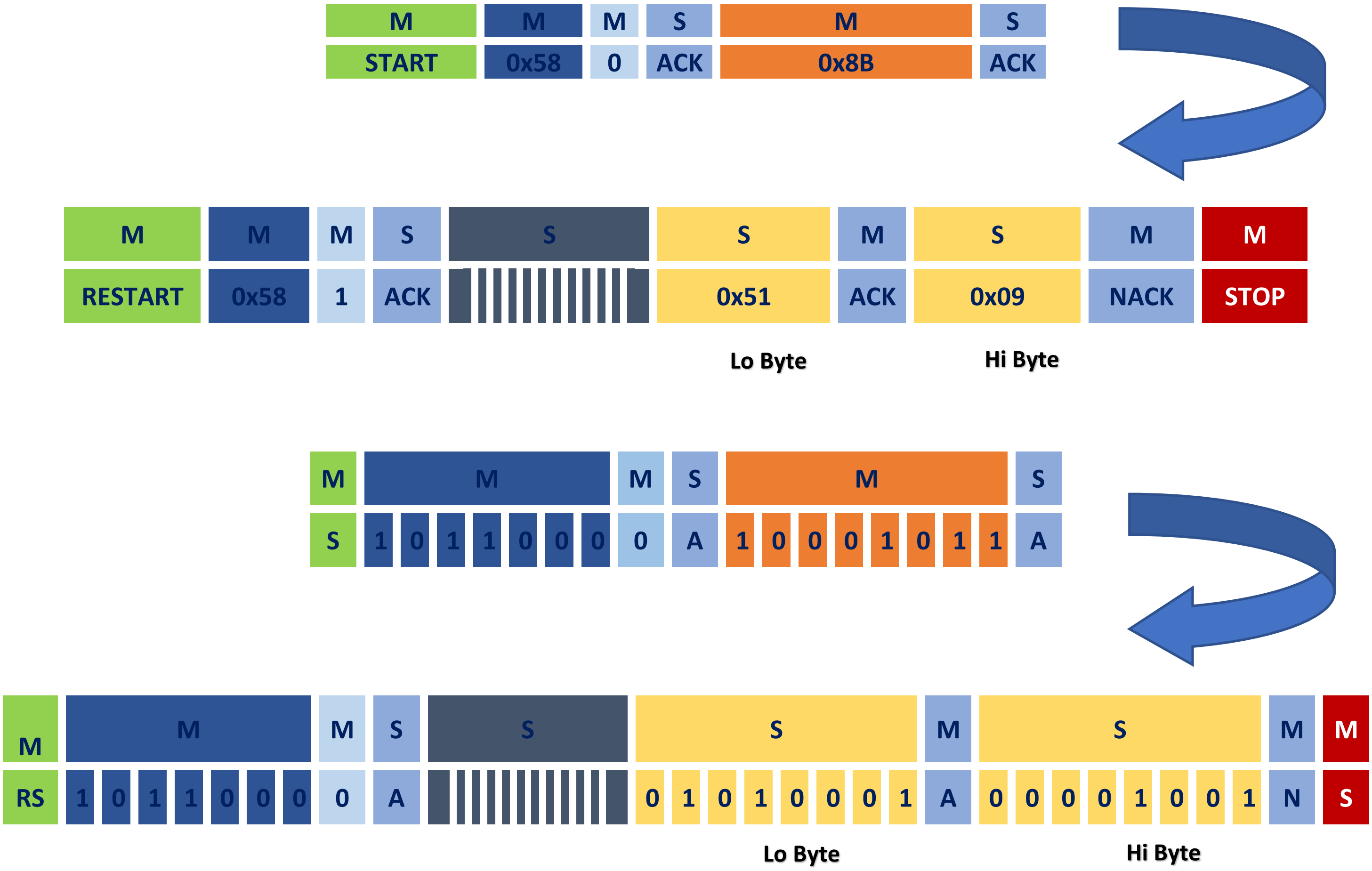
Table 4: STATUS_WORD response bitmap for existing faults 0x51 and 0x48

Interpretation of this is that unit is off, power is not good, OC fault has been detected, none of the above flags shows the warning has also been detected (maybe over-current warning)

Example 7: READ Output Voltage

Figure 15 shows visual description for output voltage status request and response from a slave unit.

Slave Write Address: 0xB0
Command: 0x8B
Slave Read Address: 0xB1
Data (Lo Byte): 0x51
Data (Hi Byte): 0x09



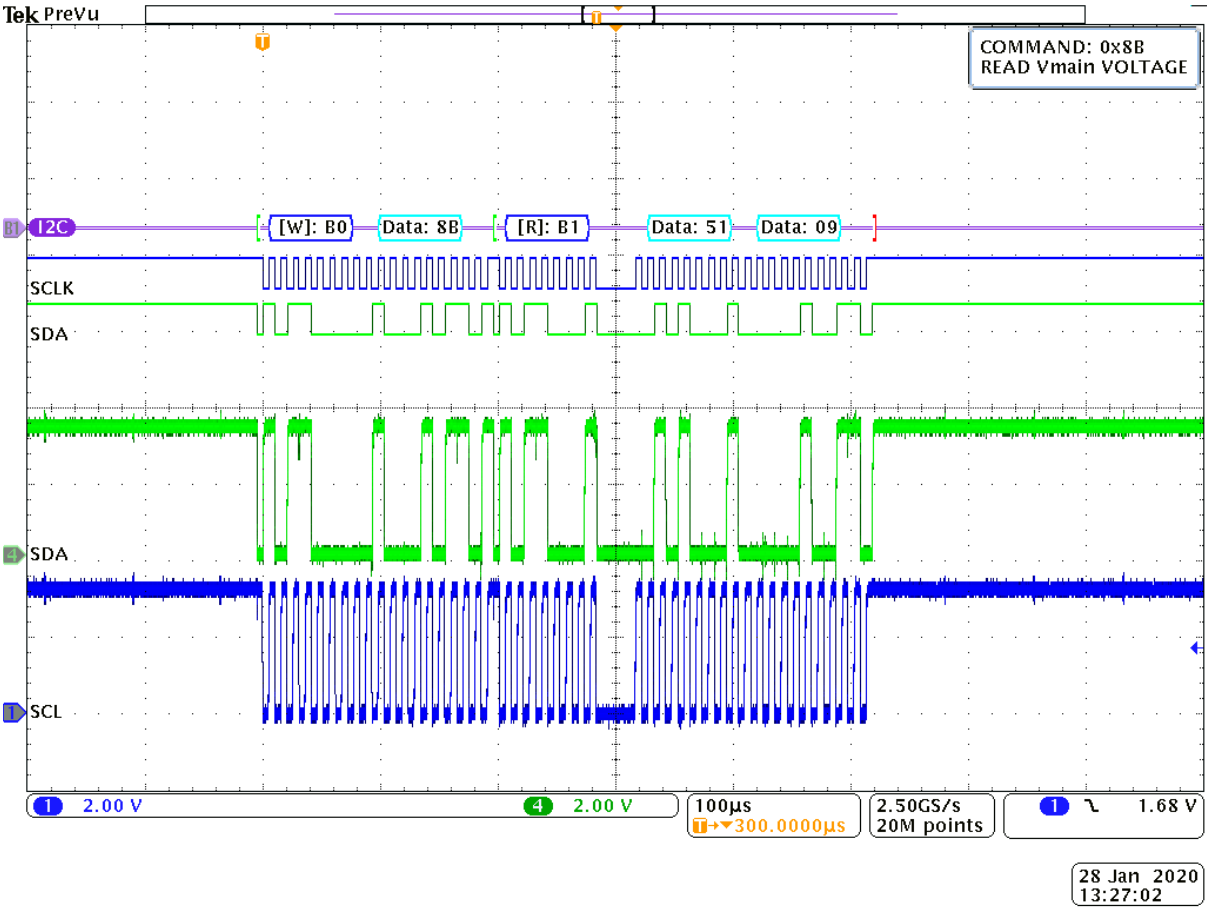
M	M	M	S	M	S
START	0x58	0	ACK	0x8B	ACK

M	M	M	S	S	S	M	S	M	M
RESTART	0x58	1	ACK		0x51	ACK	0x09	NACK	STOP
					Lo Byte		Hi Byte		

M	M							M	S	M								S
S	1	0	1	1	0	0	0	0	A	1	0	0	0	1	0	1	1	A

M	M							M	S	S	S								M	S								M	M
RS	1	0	1	1	0	0	0	0	A		0	1	0	1	0	0	0	1	A	0	0	0	0	1	0	0	1	N	S
											Lo Byte									Hi Byte									

Figure 15: Example for I²C Command 0x8B "Read Output Voltage" – Response 0x51 and 0x09



PMBUS COMMAND: 0x8B

Figure 16: Scope Screenshot I²C Command 0x8B "Read Output Voltage" – Response 0x51 and 0x09

The output voltage is calculated as per equation 1:

$$Output\ Voltage = \left(\frac{Hex\ to\ decimal(\ High\ Byte \times 100\ + Low\ Byte)}{100}\right)$$

Equation 1: Output Voltage Calculation

Hence for the response 0x51 and 0x09 the output voltage will be 23.85 V as shown in equation 2:

$$Output\ Voltage = \left(\frac{Hex\ to\ decimal(\ 9 \times 100\ + 51)}{100}\right) = \frac{Hex\ to\ dec\ (951)}{100} = \frac{2385}{100} = 23.85\ V$$

Equation 2: Example for Output Voltage to response 0x51 and 0x09

General operation commands – additional options

PMBus: COMMAND: 0x01 (Turn on/off)

MODE: READ/ WRITE

DESCRIPTION:

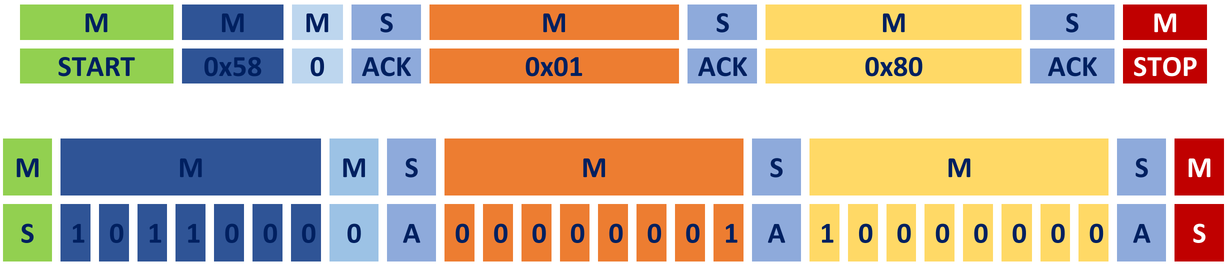
Some of the PMBus commands accept dual use, hence can be used in read and write modes. Examples 8 and 9 show options with command 0x01 which accepts secondary commands. This command supports beside write mode also the read mode where the last received secondary command can be read. In write mode it is used to turn on/ turn off power supply only if the serial control is enabled while in the read mode it sends out the last serial control send out by the user.

Example 8: Write Mode (0x01)

The byte following the command 0x01 contains the data for changing the state of power supply. Following Table 5 describes the different control settings if serial control is enabled followed by Figure 17and Figure 18 with examples.

Sec Command	Description
0x00	Instant shutdown of power supply with no delay.
0x40	Graceful shutdown with a delay of 224 ms if there is no fault or the fault level is less than 4.
0x80	It will turn on the power supply if conditions are met as defined in the Appendix 1: Table A1 - Command 0x02
None of above	CML flag of Invalid data goes high, no change in power supply operating state.

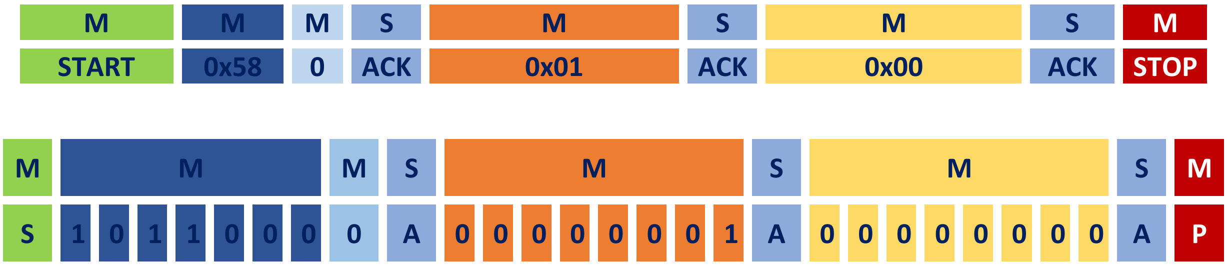
Table 5: Command 0x01 settings



M	M	M	S	M	S	M	S	M
START	0x58	0	ACK	0x01	ACK	0x80	ACK	STOP

M	M							M	S	M								S	M								S	M
S	1	0	1	1	0	0	0	0	A	0	0	0	0	0	0	0	1	A	1	0	0	0	0	0	0	0	A	S

Figure 17: Example with secondary command 0x80



M	M	M	S	M	S	M	S	M
START	0x58	0	ACK	0x01	ACK	0x00	ACK	STOP

M	M							M	S	M								S	M								S	M
S	1	0	1	1	0	0	0	0	A	0	0	0	0	0	0	0	1	A	0	0	0	0	0	0	0	0	A	P

Figure 18: Example with secondary command 0x00

Example 9: READ Mode (0x01)

This command sends out the last serial control command sent out by the user. Figure 19 shows example with data packets to be sent from master to slave device.

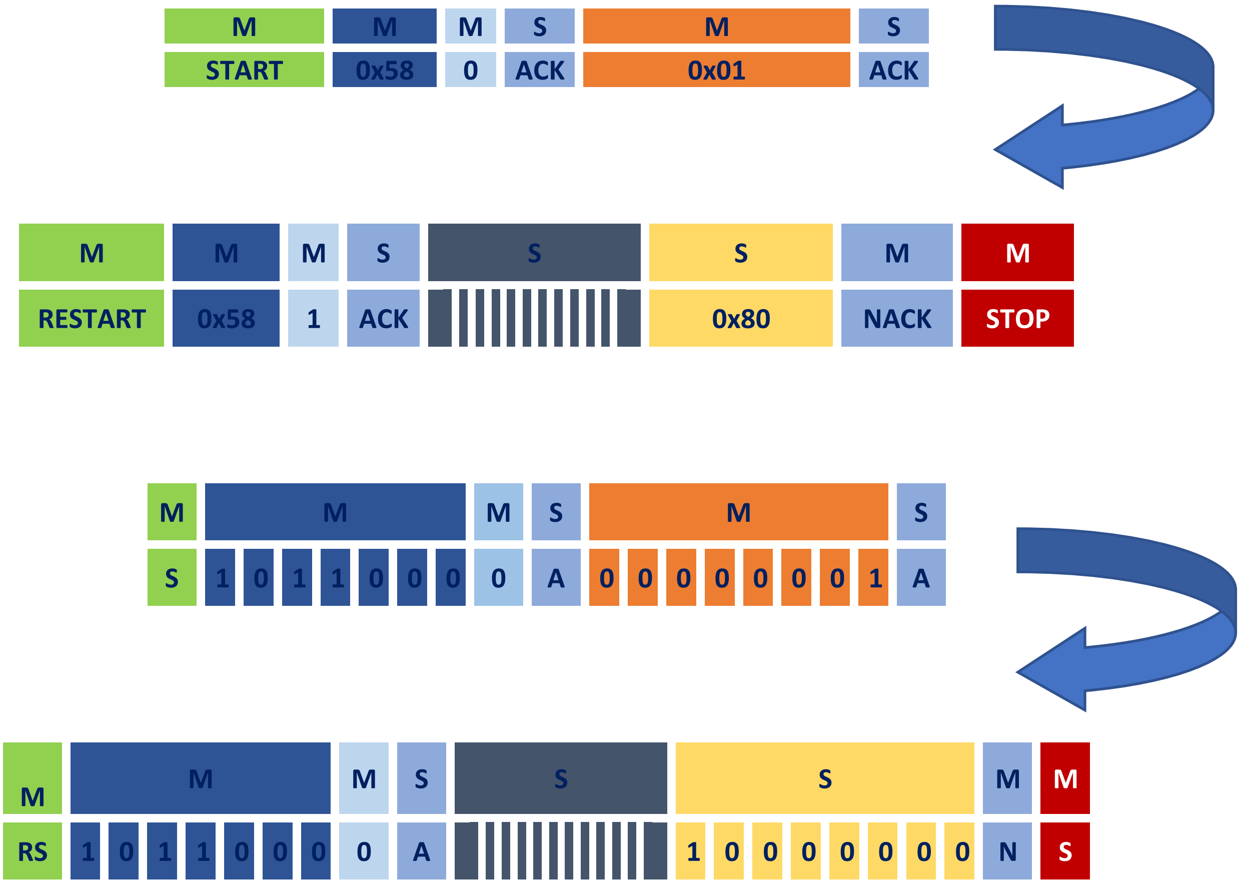
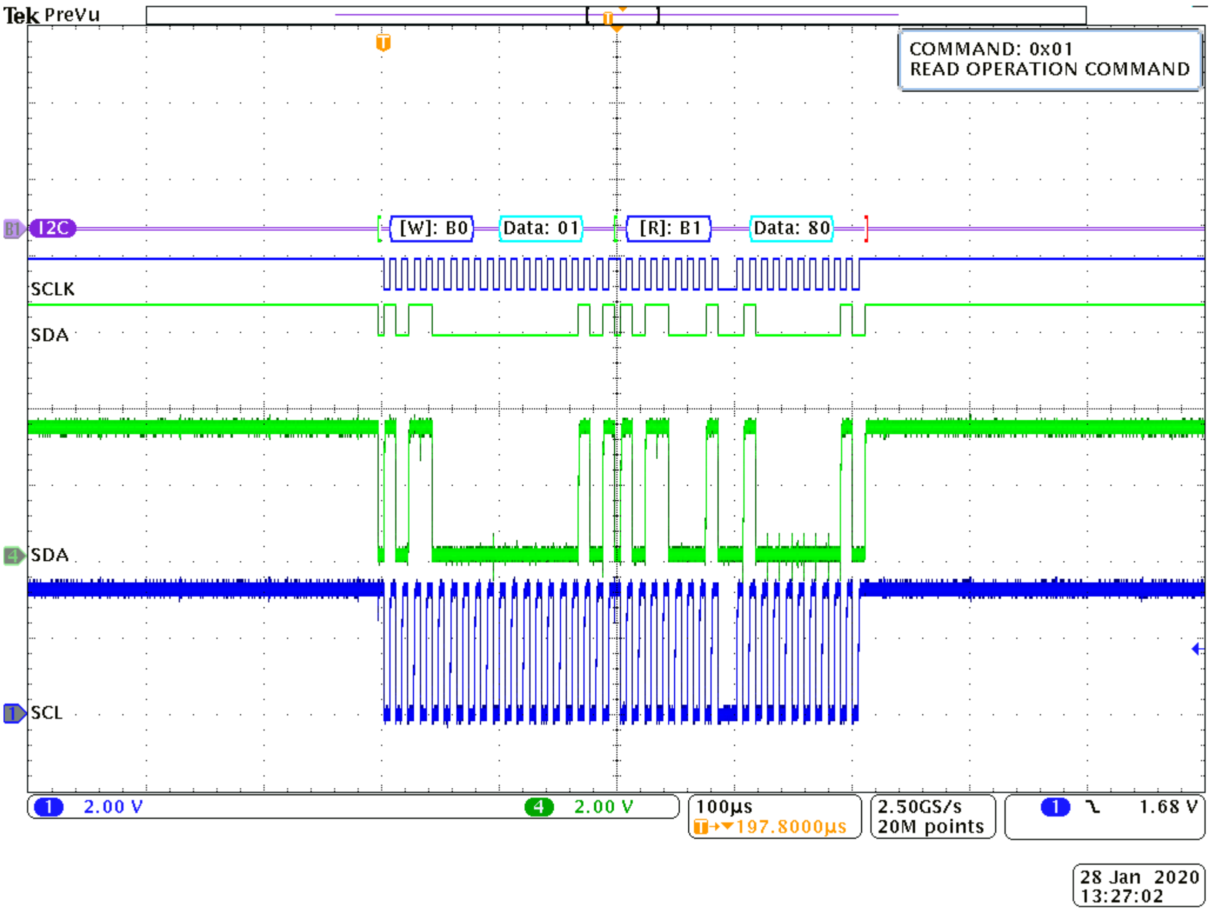

M M M S M S
START 0x58 0 ACK 0x01 ACK
M M M S S S M M
RESTART 0x58 1 ACK 0x80 NACK STOP
M M M S M S
S 1 0 1 1 0 0 0 0 A 0 0 0 0 0 0 0 1 A
M M M S S S M M
RS 1 0 1 1 0 0 0 0 A 1 0 0 0 0 0 0 0 N S

Figure 19: Example to read last control command



PMBUS COMMAND: 0x01 READ

Figure20: Scope Screenshot I²C Command 0x01 "Read Last Operation Cmd" – Response 0x80

Appendix 1:

Table A1 - Command 0x02

BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
Controllable	Serial Enable	Control Switch Enable	Polarity of Control Switch	No Ctrl Delay

Configuration	Controllable	Serial Enable	Control SW Enable	Polarity of Control Switch	No ctrl Delay	Serial Input	Control Switch Input	Power Supply at bootup	Note
0x00-0x0F	0	x	x	x	x	x	x	Always On	Un-controllable. Power supply will turn on irrespective of the state of switch or serial enable.
0x10-0x13	1	0	0	x	x	x	x	No change	Invalid / Unsupported Command
0x14	1	0	1	0	0	x	Low	On	PS on/off operation via serial communication is disabled. PS can only be turned off with a delay of **SD1** by giving high/ floating signal to control switch. After fault, PS can be turned on by switch reset as Low -> High/floating-> Low or AC recycle.
							High/ Floating	Off	
0x15	1	0	1	0	1	x	Low	On	PS on/off operation via serial communication is disabled. PS can only be turned off with a delay of **SD0** by giving high/ floating signal to control switch. After fault, PS can be turned on by switch reset as Low -> High/ floating-> Low or AC recycle.
							High/ Floating	Off	
0x16	1	0	1	1	0	x	Low/ Floating	Off	PS on/off operation via serial communication is disabled. PS can only be turned off with a delay of **SD1** by giving low/ floating signal to control switch. After fault, PS can be

							High	On	turned on by switch reset as High -> Low/ floating-> High or AC recycle.
0x17	1	0	1	1	1	x	Low/ Floating	Off	PS on/off operation via serial communication is disabled. PS can only be turned off with a delay of **SD0** by giving low/ floating signal to control switch. After fault, PS can be turned on by switch reset as High -> Low/floating-> High or AC recycle.
							High	On	
0x18 – 0x1B	1	1	0	x	x	Serial Control Commands	X	On	Power supply can be turned on/off only by giving turn on/off command via serial communication as per Serial control Commands. The control of power supply via external switch is disabled.
0x1C	1	1	1	0	0	Serial Control Commands	Low	On	**Default Setting** Power supply can only be turned on if switch is Low and turn on command is given (**turn on command present by default**) via serial communication. Power can be turned off either by sending off signal via serial command or by giving high/ floating signal to control switch with a shutdown delay of **SD1**. If switch is high/ floating and an off signal is given out via serial, then to turn on PS, switch must go low and turn on signal from serial communication. If shutdown via fault, then turn on can be done either by sending turn on signal via serial comm or reset external switch as Low -> High/ Floating-> Low
							High/ Floating	Off	
0x1D	1	1	1	0	1	Serial Control Commands	Low	Off	Same as above. Only difference is that the shutdown via external switch or serial command will be delayed by **SD0**.

							High/ Floating	Off	
0x1E	1	1	1	1	0	Serial Control Commands	Low/ Floating	Off	Power supply can only be turned on if switch is High and turn on command is given via serial communication **turn on command present by default**). Power can be turned off either by sending off signal via serial command or by giving low/ floating signal to control switch with a shutdown delay of **SD1**. If switch is low/floating and an off signal is given out via serial, then to turn on PS, switch must go high and turn on signal from serial communication. If shutdown via fault, then turn on can be done either by sending turn on signal via serial comm or reset external switch as High -> Low/Floating-> High
							High	Off	
0x1F	1	1	1	1	0	Serial Control Commands	Low/ Floating	Off	Same as above. Only difference is that the shutdown via external switch or serial command will be delayed by **SD0**.
							High	Off	

SD0 : Instant shutdown of PSU.

SD1 : Shutdown with a delay of ~224ms if there is no fault or the fault is of lower severity.