DM8168 AVS PMBus Driver User Guide

- **This AVS driver is applicable only for Characterized silicon samples**

1 Introduction

SmartReflex-AVS is a technology that uses adaptive power supply to achieve the goal of reducing active power consumption. DM816x device have Class 2B implementation of smart reflex and this allows dynamic AVS using software.

PMBus specific driver support is required for PMIC's such as TP40400. PMBus is an open standard protocol that defines a means of communicating with power conversion and other devices. It is a communications protocol based on I2C. Hence it is just a specification or a wrapper over I2C.

TPS40400 is a buck controller and allows programming and monitoring via the PMBus interface. The TPS40400 supports a subset of the commands in the PMBus 1.1 specification.

2 Acronyms & definitions

Acronym	Definition
AVS	Adaptive Voltage Scaling
SR	SmartReflex
HVT	High Voltage Threshold sensor
SVT	Standard Voltage Threshold sensor
GPIO	General Purpose Input Output
PMIC	Power Management Integrated Circuit
VR	Voltage regulator
PMBus	Power Management Bus

3 Driver Configuration

This section describes about the kernel configurations for PMBus driver & its dependencies

3.1 Voltage Regulator Driver configuration

The default kernel configuration enables support for TPS40400 PMBus voltage regulator Driver (built into the kernel).

To enable or disable TPS40400 PMBus based voltage regulator driver kernel build, follow these steps:

```
$ make CROSS_COMPILE=arm-none-linux-gnueabi- ARCH=arm menuconfig
```

- Select Device Drivers from the main menu.

```
   Power management options  --->
[*] Networking support  --->
   Device Drivers  --->
   File systems  --->
...
...
```

- Select Voltage and Current Regulator Support from the menu.

```
   Sonics Silicon Backplane  --->
-*- Multifunction device drivers  --->
-*- Voltage and Current Regulator Support  --->
<*> Multimedia support  --->
...
...
```

- Select TI TPS40400 PMBus PMIC from the menu

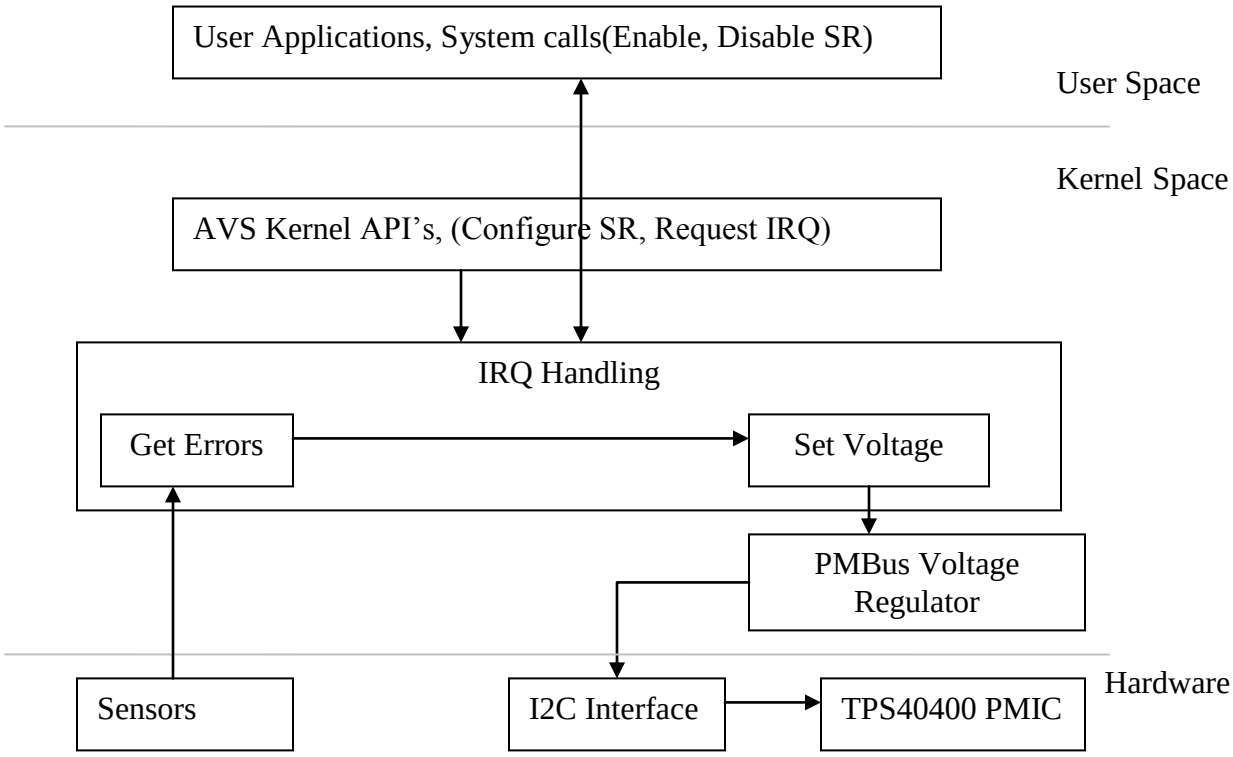
```
< > National Semiconductors LP3972 PMIC regulator driver
<*> TI TPS40400 PMBus PMIC
<*> GPIO voltage regulator
< > TI TPS6507X Power regulators
< > Intersil ISL6271A Power regulator
...
...
```

- After doing driver selection, exit and save the kernel configuration when prompted.

4 Kernel Building

- Once the configuration done according to your requirement then build the kernel by referring *Compiling Linux Kernel* part of PSP User Guide

5 SmartReflex-PMBus Driver Architecture



6 Features

The AVS driver supports following features

- Supports the PMIC TPS40400
- Supports both HVT and SVT sensors
- Voltage control over PMBus-I2C lines

7 Implemented PMBus Commands

The voltage regulator driver which has been developed for AVS supports the following PMBus commands only required for the this purpose:

- **PMBUS_OPERATION**

  The OPERATION command is used to turn the device output on or off. It is also used to set the output voltage to the upper or lower MARGIN voltages.

- **PMBUS_CLEAR_FAULTS**

  The CLEAR_FAULTS command is used to clear any fault bits that have been set.

- **PMBUS_VOUT_MODE**

  It is a byte that consists of a 3-bit Mode and 5-bit exponent parameter, as shown below. The 3-bit Mode sets whether the device uses the Linear or Direct modes for output voltage related commands. The 5-bit parameter sets the exponent value for the linear data mode. It is a read only based command.

- **PMBUS_VOUT_TRIM**

  The VOUT_TRIM command is used to apply a fixed offset voltage to the output voltage command value.

- **PMBUS_VOUT_CAL_OFFSET**

  This command applies an offset to the READ_VOUT command results to calibrate out offset errors in the on board measurement system.

- **PMBUS_VOUT_CAL_GAIN**

  This command applies a gain correction to the READ_VOUT command results to calibrate out gain errors in the on board measurement system.

- **PMBUS_IOUT_CAL_GAIN**

  The IOUT_CAL_GAIN is the ratio of the voltage at the current sense element to the sensed current

- **PMBUS_FREQUENCY_SWITCH**

  The FREQUENCY_SWITCH command sets the switching frequency.

- **PMBUS_VOUT_SCALE_LOOP**

VOUT_SCALE_LOOP is equal to the feedback resistor ratio. The nominal output voltage is set by a resistor divider and the internal 600mV reference voltage.

- **PMBUS_READ_VOUT**

The READ_VOUT commands returns data that represents the output voltage of the controller.

- **PMBUS_REVISION**

The PMBUS_REVISION command returns data that indicates that the TPS40400 is compliant with the 1.1 revision of the PMBus specification.

8 Using this Driver

Type the below commands to enable/disable the driver:

$ mount -t debugfs debugfs /sys/kernel/debug

Enable: $ echo 1 > /sys/kernel/debug/smartreflex/autocomp

Note: This is enabled by default in the driver, hence the above steps may not be required

Disable: $ echo 0 > /sys/kernel/debug/smartreflex/autocomp

9 Additions made to the Kernel to Support AVS with a PMBus based PMIC

Note: This section is only meant for controlling the vdd_avs voltage from SmartReflex driver

1. Implement voltage regulator for chosen PMBus based PMIC and provide the generic calls to the SmartReflex driver

Like, regulator_get(), regulator_get_voltage(), regulator_set_voltage(). If needed provide the enable and disable hook ups.

For reference go through voltage regulator driver at "**drivers/regulator/tps40400-regulator.c**". Current implementation of SmartReflex driver uses this PMBus based voltage regulator

2. Add Voltage Regulator platform specific data to the board file at "**arch/arm/mach-omap2/board-ti8168evm.c**".

Change the pmbus_pmic_init_data for the max and min voltage defaults specific to the PMIC.

Change the PMBus-I2C address which is specific to the PMIC. Here

"I2C_BOARD_INFO("pmbus", 0x38)"

3. Add SR platform specific data based on the chosen PMIC to the devices file at "**arch/arm/mach-omap2/devices.c**", so that SR driver request the same

These fields must change according to the PMIC

a. voltage domain name, which is registered as a supply name in voltage regulator driver
.vd_name = "vdd_avs",
b. Step size of the PMIC, which is used to calculate the voltage delta
.vstep_size = 15000, Ex: 15mV, which is a default one for voltage regulator.

4. Based on voltage step size, modify err_minlimit & e2v_gain fields in ti816x_sr_sdata structure.

Table based on step size

5. Based on the platform change the nominal voltage value & the exponent for PMIC here in this file "**drivers/regulator/tps40400-regulator.c**"

```
pmic->exponent = -10; /*Set this as per default value for PMIC(TPS40400)*/

pmic->nominal_uV = 1000000; /* For DM8168 - nominal voltage is 1V)*/
```

10 TPS40400 Configurations

10.1 AVS Control System

The AVS control system in context with the TPS40400 has been described in much details in the below sections

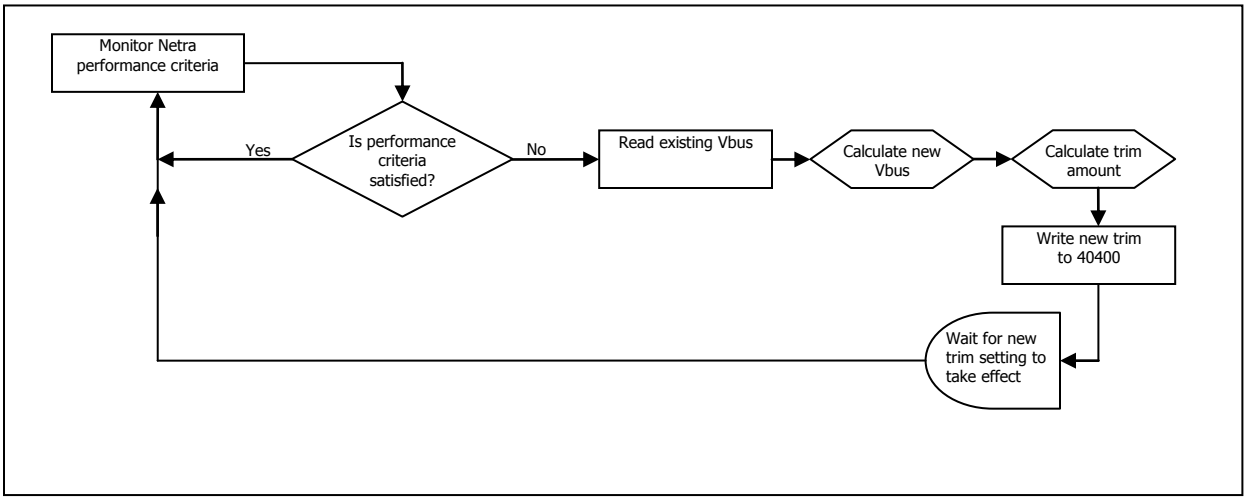
10.1.1 Description

The purpose of this section is to provide details behind the necessary steps required to successfully implement an AVS control algorithm for the TPS40400.

In an AVS control system, the output voltage bus that is produced and controlled by the TPS40400 (hereafter called "**Vbus**") is monitored by another device (external to the TPS40400) and can be adjusted by that external device in order to meet some specified dynamic performance requirements. A typical application would be the Vbus for a device like the Netra TMS320DM8168 Digital Media Processor (hereafter called Netra). This device requires that the Vbus be adjusted during live operation so that performance speed and power consumption are maintained at optimal levels. This document will describe an AVS control system for such an application.

The TPS40400 is an analog controller (analog control loop, analog PWM) but it has an integrated PMBus interface. This interface can be used to retrieve real-time operating conditions such as output voltage and current, and it can also be used to trim the Vbus during operation.

For a typical AVS control system, the host, in this case the Netra, monitors its own performance parameters and based on those parameters it can determine if a new/different Vbus setting is required to maintain optimal performance. In simple terms, Netra might apply the following:



This simplified flow chart highlights a possible process that Netra could use to adjust the Vbus based on Netra's own performance needs.

As stated, the 40400 is an analog controller which is capable of controlling its duty cycle to any required value, subject to the limitations of the 40400 controller specifications.

10.1.2 Output voltage trim logic

However, the PMBus interface is digital in nature and as such it imposes limitations on the trim settings as well as the voltage readback resolution. The manner in which trim is achieved in the 40400 is by changing the internal reference by a fixed number of steps over a fixed range. The nominal reference voltage is 600mV, and the adjustment range for the reference is +/- 25%, or from 450mV to 750mV. The trim resolution is based on a 7-bit architecture, which implies a total of (2^7), or 128 discrete steps. Each step is then defined by Equation 1:

Equation 1:

$$\text{Vref trim step size} = \frac{(750\text{m} - 450\text{m})}{2^7} = 2.34375\text{mV}$$

The trim setting is controlled by this 7-bit word mantissa. This mantissa can be set anywhere from 0000000 to 1111111, or in hex from 0x00 to 0x7F.

This result in the following available reference voltages (not all steps are shown):

	Mantissa Bits								
Decimal	MSB **6**	**5**	**4**	**3**	**2**	**1**	LSB **0**	**Available Vref Steps**	**Comment**
0	0	0	0	0	0	0	0	450.000E-3	Min value
1	0	0	0	0	0	0	1	452.344E-3	Min + 1 step
2	0	0	0	0	0	1	0	454.688E-3	Min + 2 steps
:	:	:	:	:	:	:	:	:	:
63	0	1	1	1	1	1	1	597.656E-3	No trim -1 step
64	1	0	0	0	0	0	0	**600.000E-3**	**No trim**
65	1	0	0	0	0	0	1	602.344E-3	No trim +1 step
:	:	:	:	:	:	:	:	:	:
125	1	1	1	1	1	0	1	742.969E-3	Max - 2 steps
126	1	1	1	1	1	1	0	745.312E-3	Max - 1 step
127	1	1	1	1	1	1	1	747.656E-3	Max value

The table above shows the available voltage reference settings. The resulting available Vbus settings depend on the resistor divider values in the feedback circuit of the 40400.

As a representative example which abpplies to the DVR-RDK Netra board, if the nominal output voltage is designed to be 1.0V, this defines the ratio of the divider resistors. The upper resistor would be 0.4X, and the lower divider resistor would be 0.6X (X is the same for both values, and would be chosen to yield appropriate real resistor values). If we combine these divider values with the available Vref steps, then these reference voltage steps would then map to the following Vbus steps:

Decimal	Available Vref Steps	Available Vbus Steps
0	450.000E-3	750.000E-3
1	452.344E-3	753.906E-3
2	454.688E-3	757.813E-3
:	:	:
63	597.656E-3	996.094E-3
64	**600.000E-3**	**1.000E+0**
65	602.344E-3	1.004E+0
:	:	:
125	742.969E-3	1.238E+0
126	745.312E-3	1.242E+0
127	747.656E-3	1.246E+0

This shows that while the step size of the reference is defined by the equation above, which yields a step size of 2.34375mV, the resulting Vbus steps depend on the voltage divider values and are given by Equation 2:

Equation 2**:**

$$\text{Vbus trim step size} = \text{Vref Step Size} * \frac{(R_{top} + R_{bottom})}{(R_{bottom})} = (2.34375\text{mV}) * \frac{(0.4 + 0.6)}{0}.6 = 3.90625\text{mV}$$

This implies that for a Vbus with a designed nominal voltage of 1.0V, this Vbus can be trimmed only in integer multiples of 3.90625mV, either up or down, from nominal. So for example, if the Netra desired to set the new Vbus to 1.002V, this Vbus setting would not be possible. The closest achievable Vbus setting would have to be either 1.0000V (no trim) or 1.0039V (1 LSB above no trim).

The preceding is a description of available Vref trim steps. There is another set of constraints to consider when using the 40400 in an AVS control system.

The 40400 can report its actual output voltage (which is Vbus), and this voltage is determined based on a 10-bit architecture. This implies a total of (2^10), or 1024 discrete steps. The range of Vbus measurement is fixed and is 0V to 16V. Each Vbus read step is then defined by Equation 3:

Equation 3:

$$\text{Vbus read step size} = \frac{(16 - 0)}{2^{10}} = 15.625\text{mV}$$

This Vbus read step resolution is fixed, and it is not dependent on the Vref trim setting or the feedback resistor divider values. The Vbus read word is reported by a 10-bit word mantissa. This word can be anywhere from 0000000000 to 1111111111, or in a hex word from 0x000 to 0x3FF. In this case, the 10-bit mantissa is embedded in a 16-bit word, and the other 6 bits are not discussed here, but note that the 10-bit mantissa for Read_Vout is mapped to bits 13 through 4.

This results in the following available Vbus read voltages (not all steps are shown):

	Mantissa Bit											
Dec	MSB **13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	LSB **4**	**Available Read_Vout Steps**	**Comment**
0	0	0	0	0	0	0	0	0	0	0	0.000000	Min value
1	0	0	0	0	0	0	0	0	0	1	0.015625	Min + 1 step
2	0	0	0	0	0	0	0	0	1	0	0.031250	Min + 2 steps
:	:	:	:	:	:	:	:	:	:	:	:	:
63	0	0	0	0	1	1	1	1	1	1	0.984375	Nominal -1 step
64	0	0	0	1	0	0	0	0	0	0	1.000000	Nominal Vbus
65	0	0	0	1	0	0	0	0	0	1	1.015625	Nominal +1 step
:	:	:	:	:	:	:	:	:	:	:	:	:
1021	1	1	1	1	1	1	1	1	0	1	15.953125	Max - 2 steps
1022	1	1	1	1	1	1	1	1	1	0	15.968750	Max - 1 step
1023	1	1	1	1	1	1	1	1	1	1	15.984375	Max value

It is important to note that while the available Vbus trim settings have a step size of 3.91mV, the available Read_Vout steps have a step size of 15.625mV (when nominal is 1.0V). This means that there are 4 Vbus trim steps for every Read_Vout step, so it is possible to trim the output voltage by a few steps and not get an appreciable change in the Read_Vout data, even though the actual output voltage will have changed by that many steps. This will be discussed further in the measured data discussion.

As can be seen from the available Read_Vout steps, even though it is possible that the actual output voltage exist between available steps, the 40400 will only report in the available steps. So during an AVS exercise, it would not serve any purpose to expect a Read_Vout result at any voltage other than the available steps. If the Netra wanted a Vbus of 0.996V, the 40400 could only report a Vbus of either 0.984375 or 1.000000V, even though the desired Vbus of 0.996V is achievable from the available Vbus trim steps.

10.1.3 Averaging the read vout values and its necessity

Note: Before you read this section, READ_VOUT implementation will no longer be used in the AVS-TPS40400 Driver. Instead whatever is set by the SR driver last will be sent back to its read requests.

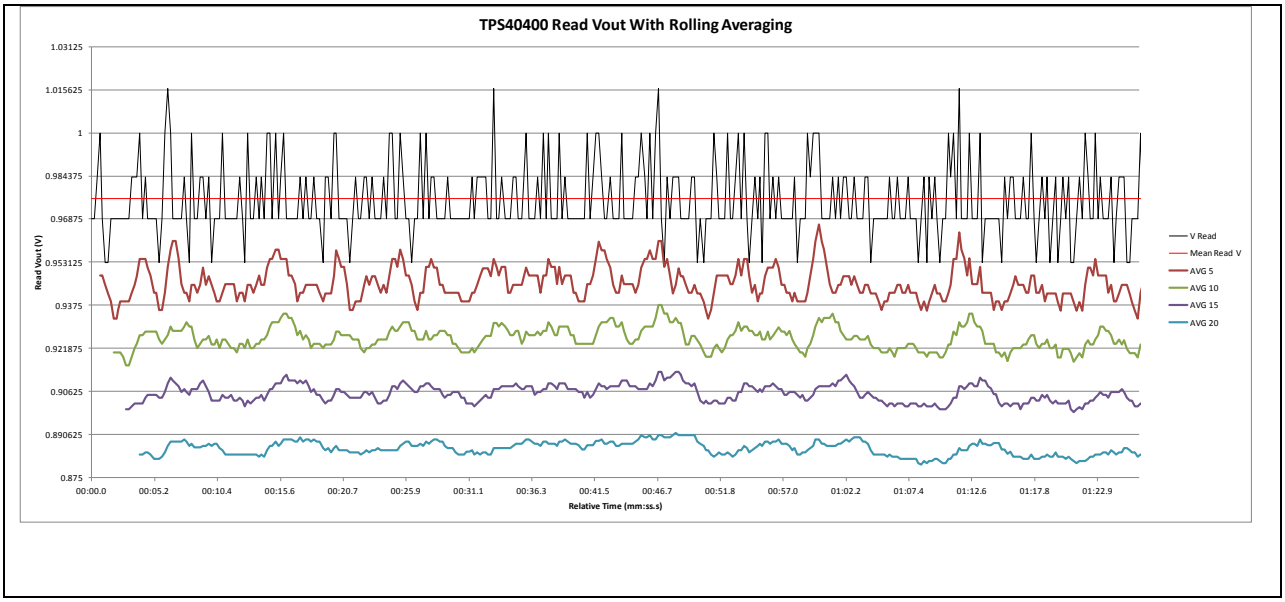
At this point it is necessary to introduce the variability in the Read_Vout data. While the 40400 is an analog controller and as such it produces an output voltage that is free of limit-cycle oscillations, like any switcher the Vbus will contain switching and system

noise. So even during steady state, when the Vbus analog average voltage is actually not changing, the 40400 will report Vbus within a range of a few LSBs about the analog average. As an example, with a Vbus setting of 1.0V, and even an actual analog Vbus also of 1.0V, several successive Read_Vout data points reported by the 40400 would report some Vbus readings of 1.0V, some at 0.984375V (-1 LSB), some at 1.015625V (+1 LSB), and perhaps some at more than 1 LSB away from the actual analog Vbus of 1.0V. The deviation from the nominal in the reported data will depend largely on the actual, real Vbus noise amplitude. If a long term average is taken of the Read_Vout data, the average would be within tolerance of 1.0V (note that the average can be any voltage and is not restricted to any of the available trim or read steps). However, if no, or insufficient averaging is performed on the Read_Vout data, the data will rarely yield the desired Vbus. This is important to note because it implies that without averaging, the Vbus will rarely satisfy the AVS algorithm, and so the AVS controller would never cease trimming Vbus.

The picture below shows actual measured data from the 40400 EVM, as retrieved using the Fusion Digital Power Designer GUI. This EVM is not related to the Netra, it is the EVM for the 40400 controller, so the amount of noise may not be representative of any other particular design.

In this picture, the Black trace is the actual Read_Vout data, the straight Red trace is the long term average (3441 successive data points), the Brown trace is a 5-point rolling average of the Read_Vout data, the Green trace is a 10-point rolling average of the Read_Vout data, the Purple trace is a 15-point rolling average of the Read_Vout data, and the Blue trace is a 20-point rolling average of the Read_Vout data. **Note**: the rolling average traces all have slightly different artificial voltage offsets added to them so that the traces can all be seen clearly. In reality, all traces have the same long term average and would be overlaid, but they would then be difficult to see.

Black Trace	**Red Trace**	**Brown Trace**	**Green Trace**
Actual Read_Vout data	Long term average of Read_Vout	5-point rolling average of Read_Vout	10-point rolling average of Read_Vout
Purple Trace	**Blue Trace**	**Analog Vbus Average**	**Read_Vout Spread**
15-point rolling average of Read_Vout	20-point rolling average of Read_Vout	0.976V	4 LSB



This picture is instrumental in understanding the various steps required to achieve a stable AVS algorithm.

EXAMPLE: As an example, let's assume the Netra determined that it needed Vbus to be trimmed higher by 15.625mV (so, 4 Vbus trim LSB steps or 1 Read_Vout step), and let's assume there was no averaging performed on the Read_Vout data. The existing analog Vbus voltage (before the new trim setting is applied) is the red trace, or 0.976V. Since there is no way of knowing when a Read_Vout command will be performed, let's assume the system had the misfortune of capturing a single data point (only one since there is no averaging) at one of the 4 +Ve peaks of the black trace (so 1.015625V). This would result in:

Existing Vbus = 0.976V

New Vbus Requirement = 0.976 + 15.625mV = 0.9917V

Single Read_Vout data point = 1.015625V

So, even though the Netra had determined that it needed Vbus to be trimmed up by 15.625mV from its existing 0.976V to 0.9917V, when it reads the Vbus, it finds that Vbus is already higher than it wanted. Depending on the AVS algorithm, this might actually result in a downward trim, or in other words, further from desired than the existing Vbus. It may trim in the wrong direction. This is a clear argument for the need for averaging the Read_Vout data. The more data points in the average, the higher the likelihood of trimming in the desired direction. As shown on the EVM, even a 20-point rolling average has a spread of 16.3mV (slightly more than 1 Read_Vout LSB).

The next key point to be derived from the picture is the fact that even a multi-point rolling average of Read_Vout data points yields averaged data that would rarely, if ever, satisfy a precise Vbus target. While the 20-point rolling average has the smallest spread of data in the traces shown, that average data never remains at a fixed voltage level, even if the average analog Vbus voltage is constant. So, for example, if the Netra determined that Vbus needed to be an exact value of 1.0V, even the 20-point rolling average would

rarely work out to be exactly 1.0V. So the Vbus reading, even with multi-point averaging, would never satisfy the AVS algorithm, and the Netra would never stop trimming Vbus. This brings about the requirement to apply a "dead band" to the Read_Vout data (even the averaged Read_Vout data). A dead band in this context is a range of acceptable Vbus readings, rather than one precise voltage level. In this example, the desired Vbus is 1.0V, but the AVS algorithm must include a spread of Vbus levels that are deemed acceptable (an example would be: desired Vbus is 1.0V, but anywhere between 0.95V and 1.05V is acceptable or "close enough", such that no further trim is necessary). The size of the dead band depends on the actual spread of Read_Vout data (after averaging) in the system in question. In the picture above, if a 20-point rolling average was used on the Read_Vout data, then the dead band would need to be at least 16.3mV wide, or "desired" +/-8.15mV.

10.1.4 Wait time after a set voltage

The next point to be discussed is particular to the 40400. The 40400 has a configurable soft-start time, and that time can be set to any of several fixed numbers (see 40400 datasheet for details). But this soft start time sets the slew rate for any operation that changes the output voltage, including trim and margin. If for example the nominal Vbus is 1.0V and the soft start time of 2.7mSec was selected in configuration, that means that any trim event would slew at that same rate (in this case 1V in 2.7mSec). So after a trim command is written to the 40400, the host must allow the 40400 enough time to slew the output voltage to the new setting, given this fixed slew rate. This amounts to applying a wait time as shown in Equation 4.

Equation 4:

$$\text{Wait time after Trim} \geq \frac{(\text{trim } \delta V) * (\text{TON}_{\text{RISE}}\text{time})}{(\text{Vout Nominal})}\text{, in Sec, where:}$$

- Wait time is the amount of time between the Trim Vout write and the next Read Vout
- trim δV is the requested change in output voltage from where it is now (the delta V)
- TON_RISE time is the configured soft-start time for the 40400
- Vout Nominal is the un-trimmed nominal Vbus voltage

The reason for this required wait time is because if a Read_Vout is performed before the required wait time, the 40400 will not have completed the transition from old to new Vbus settings. So even if all else was correctly applied (averaging, dead band), this would result in an unstable AVS algorithm because the Netra would potentially apply a second trim before the first one was completed. The second trim would in all likelihood apply an over-trim, and then a trim would be required in the other direction. Repeat. This would result in oscillations on the Vbus.

10.1.5 Summary

So in summary, all three of these requirements should be included in the AVS algorithm:

- Averaging several Vout readings. The more data points in the average, the smaller the required dead band, but the slower the AVS functionality. From the data in the picture above, a 20-point rolling average would not be sufficient for many stringent AVS systems.
- Implement a dead band so that the AVS does not try to "regulate out" the noisy Read_Vout data.
- Wait a pre-determined amount of time after a Trim Vout command. The wait time must be at least (Trim delta V)/(Soft Start slew rate)

Note: The drawback of using a dead band is that the bus voltage is less accurate because any voltage within the dead band is acceptable.

Note: The drawback of using averaging is that it slows the AVS functionality down.

10.2 Voltage Control Logic and Calculations

10.2.1 Setting the VOUT_SCALE_LOOP

This setting is necessary for the correct calculation of the VOUT_TRIM, VOUT_MARGIN_HIGH and VOUT_MARGIN_LOW.

i.) The feedback resistors in the schematic of the DVR RDK are 15k and 10k. The nominal reference (VFB) is 600 mV. The expected output voltage is 10k*0.6/15k = 1.0V.

ii.) VOUT_SCALE_LOOP is defined as VFB/VOUT(nom) = 0.6

iii.) Converting 0.6 to the linear format (Mantissa * 2^ Exponent)

- the exponent is fixed by design at -9 decimal or 10111 binary.
- the mantissa is calculated as 0.6/(2^-9) = 0.6 *2^9 = 0.6 * 512 = 307.2 or 00100110011 as an unsigned binary integer

iv.) The resultant conjugation is 1011 1001 0011 0011 b or B933 h.

10.2.2 Understanding the IOUT_CAL_GAIN

The DCR of the output inductor must be entered into this register. The calculation is as follows:

DCR = (decimal entry)*2^{-15}

The range of programmable DCR is 30.5uΩ to 15.6mΩ, so the range of hex entry is 1h to 200h, and the corresponding range in decimal is 1 to 512.

So for example,

1. If the DCR was 6.348mΩ, you would enter a value of 6.384 *10^{-3}*2^{15}= 208 = D0h

2. If the DCR was 2mΩ, you would enter a hex value of 42h, which would give 2.01mΩ

Setting this is a Must. Otherwise reported current, OC Warn and OC Fault will be incorrect.

10.2.3 Checking and setting the offset for VOUT_CAL_OFFSET

READ_VOUT calibration example: On initial system startup, the output is precisely trimmed to be 1.0 volts. Issuing a PMBus read on the READ_VOUT command returns an average value of say the output voltage reading 916 mV so We set vout_cal_offset to be the difference of nominal voltage and the above value as offset required.
Therefore, This command applies an offset to the READ_VOUT command results to calibrate out offset errors in the on board measurement system

10.2.4 Understanding the READ_VOUT Command

The accuracy of this output is greatly influenced by the quality of the PCB layout and the precision of the calibration

i) For example, if issuing a READ_VOUT command returns 480h, this would indicate that the output voltage is 480h * 2^-10 = 1152/1024 = 1.125 volts

ii) Detailed understanding of mechanism to read voltages is mentioned in this section above 10.1.3.

10.2.5 Understanding the VOUT_TRIM Command

This command allows the PMBus host to adjust or trim the output voltage. If you use this command to correct for an offset within your system for absolute accuracy AND use this command to implement DVS, the host will need to account for the fixed offset.
The trimming of output voltage happens in certain steps as descibed in section above 10.1.2

i.) If Vout(nominal) is 1 volt, the range of acceptable values for VOUT_TRIM would be -250 mV to 250 mV.

ii.) Suppose that you choose to increase the output voltage by 100 mV:

1.) First you would calculate the difference between the nominal and the required absolute voltage(current+100mV).

2.) To get to the desired voltage, you would write the above difference to VOUT_TRIM.

10.2.6 Understanding PMBUS_IOUT_OC_FAULT_LIMIT/ PMBUS_IOUT_OC_WARN_LIMIT

The PMBUS_IOUT_OC_FAULT_LIMIT command sets the value of the output current, in amperes, that causes the over-current detector to indicate an over-current fault condition.

The PMBUS_IOUT_OC_WARN_LIMIT command sets the value of the output current, in amperes, that causes the over-current detector to indicate an over-current warning.