ISSN: 1863-5598

ZKZ 64717
08-16

Bodo´s Power Systems®

Electronics in Motion and Conversion

August 2016

Introduction to the AVSBus™

In all forms of today's electronics, the goal is maximum performance while using the least amount of energy. For a smart phone, this is all about customer satisfaction by providing a responsive device with a battery that lasts all day. For servers, routers, and storage in data centers customers need high availability and quick response and the operator needs to minimize both capital and operating expense. For today's computing equipment, the energy bill is a large part of the operating cost and everyone is interested in minimizing energy consumption without sacrificing performance.

By Bob White, President and Chief Engineer, Embedded Power Labs LLC

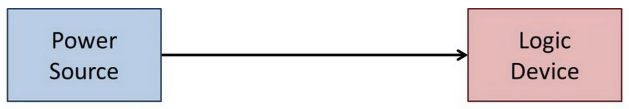
For years power consumption could be minimized by reducing the CMOS feature size and operating at a lower voltage. However, as feature size was reduced the savings in reduced power due to switching the capacitance of the transistors became matched by the off-state leakage current. Now to minimize power consumption a processor or other larger digital device (such as a large ASIC or FPGA) has to quickly shift between a low power idle or sleep mode and a high performance mode. In low power mode, the device minimizes its operating voltage to to reduce leakage losses and minimizes the clock frequency to minimize switching losses. When performance is needed, the supply voltage is ramped as fast as possible to the higher voltage needed to support the higher clock speed and then the clock frequency is increased. This approach to optimizing power consumption is called dynamic voltage and frequency scaling (DVFS) and is illustrated in Figure 1.

Years ago Intel implemented this in their high performance processors. The processor communicated with its power supplies via a proprietary high speed serial bus ("Serial VID"). Other manufacturers were left to devise their own means of implementing DVFS.

In 2012 Juan Arango and Travis Summerlin of Texas Instruments (TI) were developing a protocol for DVFS they called AVSBus (Adaptive Voltage Bus). In January 2013 TI offered this AVSBus protocol to the System Management Interface Forum (SMIF), who owns and manages the SMBus and PMBus™ standards, for inclusion as an extension of the PMBus. The SMIF accepted and after further development in the PMBus Specification Working Group, the AVSBus was released in March 2014 as part of Revision 1.3 of the PMBus standards. The AVSBus is now made available to any PMBus adopter a non-proprietary high speed bus for controlling power converters and implementing DVFS in any digital IC.

The Need For Speed

For DVFS to be effective, the power converter must change its output voltage very quickly as seen by the digital IC load. For a digital device operating with a clock frequency measured in GHz, a power converter that takes hundreds, or even tens, of milliseconds to respond will be seen as glacially slow. When transitioning to a low power mode, the digital device must lower its clock frequency immediately as it does not know exactly when the supply voltage will be lowered. If the transition time for the supply voltage is long (meaning even hundreds of microseconds) then power is being wasted. When the digital device needs to return to high power, high performance mode, it must wait until the supply voltage is at the required higher level before increasing the clock frequency. This delay reduces the performance. When power management schemes are switching between power saving and performance modes on a millisecond time scale, power conversion delays must be minimized to get the maximum combination of power savings and performance.



$$P \propto K_{SWITCH} \cdot V_{DD}^{\;2} \cdot F_{SW} + K_{STATIC} \cdot V_{DD}$$

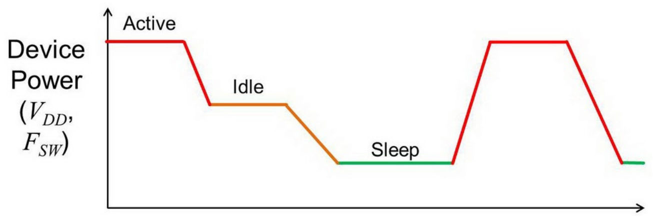


Figure 1: Saving Energy With DVFS

There are three elements to the delay from when a digital device issues the command to change voltage until the voltage has reached the new desired value:

- The communication delay from the digital device to the power converter,
- The time it takes the power converter to process the command and change the control signal (e.g. duty cycle) to the power stage, and
- The time it takes the power stage to respond

As for the time it takes the power stage to respond, that is being addressed by efforts to drive the switching ever higher. This reduces the size of the output filter inductor and allows the output current slew faster for a given input and output voltage.

The other two factors, communications delay and command processing time are addressed by the AVSBus.

While it would have been good to use the existing PMBus protocols for communicating DVFS commands from a digital device to its power converters, it is just too slow. At the typical PMBus speed of 100 kHz, it would take about 360 μs just to send the command. The PMBus protocol, which was designed to be applicable for a wide range of devices, was also not optimized for processing commands in nanoseconds. Even if the bus speed were pushed to the 1 MHZ, maximum allowed in the PMBus 1.3 standard, the 36 µs to transmit the voltage command is very slow by the digital device standards.

AVSBus addresses the communications delay in two ways: a maximum bus speed of 50 MHz and a fixed, 32-bit frame. Operating at the maximum bus speed, the command to change voltage can now be sent from a digital device to its power converter in 640 ns. The cost of the interface is kept low by using a standard SPI-like physical layer (PHY). For now this does limit the AVSBus to point-to-point communication between a digital device and its power converter but as discussed below, this is not a significant limitation.

The AVSBus addresses the processing delay by using a simple, efficient command set that is optimized for this application. This means that minimal logic is needed to decode the received command and generate the control signal that the power stage requires. With a good design and moderate clock speeds in the power controller IC, the command processing can be completed in tens of nanoseconds.

AVSBus Signals And Systems
As mentioned above, the PHY of the AVSBus is based on the standard SPI I/O ports that are readily available in the industry. The minimal implementation of the AVSBus requires two signals, AVS_Clock that is generated by the AVSBus master and AVS_MData that is used to send data from the AVSBus master (digital device) to the AVSBus slave device (power converter). If the application requires that the AVSBus master be able to receive status information from the AVSBus slave, then an optional third signal, AVS_SData, is added. The connections are illustrated in Figure 2.

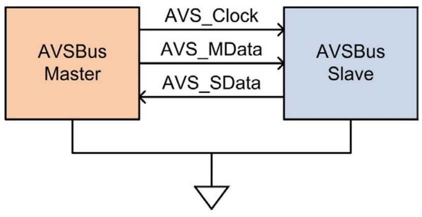


Figure 2: AVSBus Connections

For digital devices that need to control more than one rail, there are two options. The first is simply to have as many AVSBus ports as rails that need to be controlled. We think the preferred option is going to be a single AVSBus communicating with a power management IC (PMIC) that has multiple power controllers, as shown in Figure 3. The AVSBus specification provides for addressing up to fifteen separate power rails over one AVSBus.

In multiple rail systems, the AVSBus specification provides flexibility on scheduling execution of commands. An AVSBus command can be sent to a rail controlled by the PMIC with the instruction to execute immediately. Or, the AVSBus master could send a command to rail 1 with the instruction to hold execution, followed by a command to rail 2 with a command to hold execution, followed by a command to rail 3 with the instruction to execute all pending commands. This provides a way for an AVSBus master to cause multiple rails to change simultaneously, which is often required to observe power sequencing requirements.

It is also possible to have a system with both PMBus and AVSBus, as shown in Figure 4. In this case, at power up the rail voltages will be set by the default PMBus output voltage settings. Once operating, the PMBus master can transfer control to the AVSBus master. The PMBus master, if need be, can also take back control from the AVSBus master. How these control transitions are managed to avoid glitching the rail voltages is described in the AVSBus and PMBus specifications.

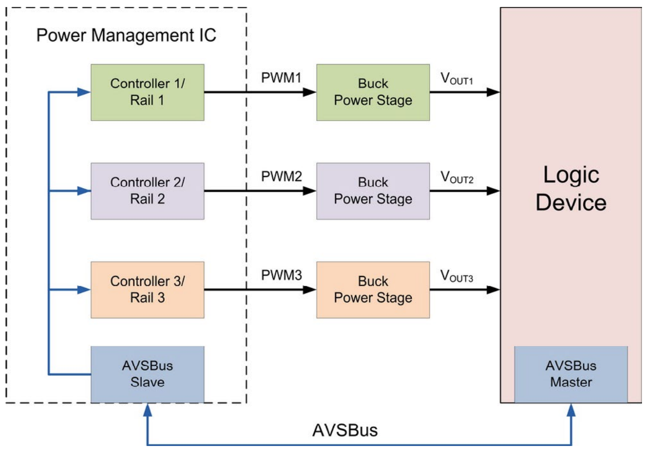


Figure 3: AVSBus System With Multiple Power Rails

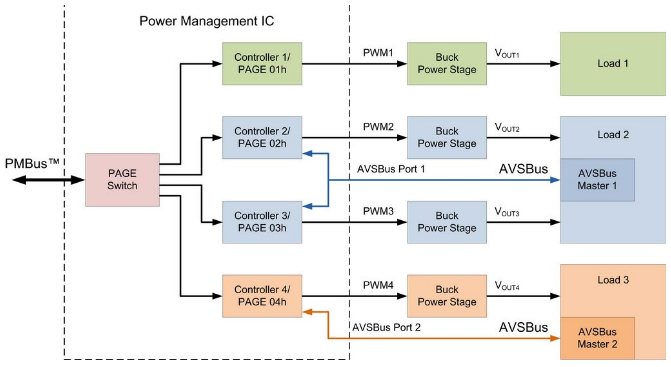


Figure 4: System With Both PMBus And AVSBus

AVSBus Data Frames and Transmission
AVSBus data is always transmitted in 32 bit fames. When the AVSBus is idle, all data lines are high. A data transmission starts when the master data signal, AVS_MData, transitions from high to low. It stays low for one bit and then goes high for one bit to complete the start of the transmission. The device sending data, such as a master transmitting a frame to a slave, launches the data on the rising edge of the clock. The receiving device captures the data on the falling edge of the clock. The last three bits of every frame is a cyclic redundancy check (CRC) that provides good assurance that the fame has been received uncorrupted.

When the master is sending a frame to the slave, there are two possible formats. A Write Frame is used to send a command to a slave device. In systems where the master can read data from an AVSBus slave a Read Frame is used to tell the slave device that the master is requesting data, such as a rail voltage. The formats of these two frame types are given in Tables 1 and 2.

If the system is using the full three wire AVSBus with the AVS_SData line, then every time a master sends a frame to the slave the slave must respond with a frame that includes a two bit ACK field and a five bit status field. If the master had sent a Write Frame and no data was requested from the slave, then most of the bits in the frame sent by the slave are set to 1. If the master had requested data, then the slave returns that data. The format of a slave-to-master frame that includes returned data is shown in Table 3.

The two bit Acknowledge field from the slave to the master provides a significant advantage over the one bit ACK/NACK of I²C/SMBus/PMBus systems. With just one ACK/NACK bit, if a slave does not acknowledge ("NACKs") the master cannot know why. It might have been because the received data was corrupt, the received command or data was not valid, the device might have had an error processing the received data, or the slave might have been too busy to process the incoming transmission. The two bit Acknowledge of the AVSBus provides some additional information at to whether a frame was properly received and processed or not, and if not, why not. The two AVSBus Acknowledge bits indicate:

10: Bad CRC, No action taken;
11: Good CRC, Invalid selector, No action taken;
01: Good CRC but no action taken because resource was unavailable; or
00: Good CRC, All OK and action taken.

Bits	Field	Description
31-30	Start Code	Always 01b
29-28	Cmd	Command code that indicates the action the master requires, such as Write And Hold
27	CmdGroup	Tells the slave whether the coming command is an AVSBus standard or a manufacturer specific command
26-23	CmdDataType	Tells the slave what data is coming, such as a value for the target rail voltage
22-19	Select	Generally selects the voltage rail to which the command is directed. The value 1111b means "all rails".
18-03	CmdData	The data for the command, such as a new rail voltage
02-00	CRC	CRC check bits

Table 1: AVSBus Write Frame Bit Fields

Bits	Field	Description
31-30	Start Code	Always 01b
29-28	Cmd	Always 11b for a Read Frame
27	CmdGroup	Tells the slave whether the requested data is in AVSBus standard format or a manufacturer specified format
26-23	CmdDataType	Tells the slave the source of the data, such as a value for the target rail voltage
22-19	Select	Tells the slave from which rail the data is requested.
18-03	Reserved	Since no data is being transmitted, these bits are reserved and are be all 1's (FFFFh)
02-00	CRC	CRC check bits

Table 2: AVSBus Read Frame Bit Fields

Bits	Field	Description
31-30	SlaveAck	A two bit code providing basic information on whether a command was executed or not, and if not, why
29	0	This bit is always 0b
28-24	StatusResp	Five bits that provide information about the slave such as "The last commanded output voltage change has been completed or not" or the slave has status information that requires the master to send a Read Status command.
23-08	CmdData	The data for the command, such as the rail voltage
07-03	Reserved	Reserved bits, the returned value is always 11111b
02-00	CRC	CRC check bits

Table 3: AVSBus Status Response Frame Bit Fields When Data Is Being Returned To The Master

AVSBus Commands

The goal of AVSBus is to be very fast. This means that the command set, unlike the PMBus which is intended to cover a wide range of power converters, is narrowly focused. There are only nine standard commands available plus provisions for custom manufacturer-specific commands. The main command is used to set or read the rail voltage. To further simplify the implementation, the resolution of the command is fixed at 1 mV/bit. This gives a range of possible output voltages from 0 V for a command value of 0000h to 65.535 V for a command value of FFFFh.

Other commands include setting or reading the output voltage transition rate (1 mV/µs per bit), reading the output current (1 mA/bit), and reading the temperature (0.1 °C/bit). There is also an output voltage reset command that sets the output voltage to the default value. Another command instructs the power converter to operate in either a "maximum power" or "maximum efficiency" mode.
There are also commands to read the slave status and the version of AVSBus the slave device supports.

Where To Get The AVSBus Specification and Support

The AVSBus specification is available as Part III of the PMBus specification at the PMBus website, www.pmbus.org. The PMBus specifications are available at no cost although one must register with the website to access the download.
For questions about the AVSBus specification, as well as the PMBus and SMBus specifications, send your inquiry to techquestions@smiforum.org.

Summary

Large, fast digital devices such as processors, ASICs, and FPGAs use dynamic voltage and frequency scaling to minimize energy consumption. The AVSBus is now offered by the System Management Interface Forum as a non-proprietary protocol for these devices to manage the converters supplying them with power. The AVSBus has been designed specifically for this task offering high speed transmission, a task focused protocol and command language, and low cost. PMICs supporting AVSBus are already in the market and it is expected that many more will be announced in the coming months.

About PMBus

The Power Management Bus (PMBus) is an open-standard digital power management protocol: simple, standard, flexible, extensible, and easy to program. The PMBus command language enables communication between components of a power system: CPUs, power supplies, power converters, and more. For more information, please go to the PMBus.org website and download an Introduction to PMBus.

About SMIF

The System Management Interface Forum is an industry Special Interest Group (SIG) composed of more than 35 member companies and adopters who work together to develop, implement and promote standardized communications protocols. The PMBus and AVSBus name and logo are trademarks of SMIF, Inc. Commercial use of the PMBus and AVSBus name and logo is restricted to PMBus adopters. Refer to the PMBus.org website for additional details.

About The Author

Bob White has more than thirty years of experience in power electronics. He is well known for his work in digital power, power system architecture, and the application of wide bandgap devices. His work at Artesyn Technologies became the starting point for the development of the PMBus standard. He led the PMBus Specification Working Group for many years and remains the editor of the PMBus and SMBus specifications. He has his own consulting company, Embedded Power Labs. He holds a BSEE from MIT, a MSEE from WPI, and is a Fellow of the IEEE.
Robert V. White, President and Chief Engineer,
Embedded Power Labs LLC, Highlands Ranch, Colorado, USA

www.embeddedpowerlabs.com