Aemulus

AM400e

Programming Manual

Version 1.6, 02-2014

Table of Contents

AemDCPwr_InitChannels

Synopsis

ViStatus AemDCPwr_InitChannels (resourceName, channelName, init_options, optionString, vi)

Arguments

ViRsrc resourceName (in)

Specifies the resourceName assigned by PXI. For example, "PXI36::0::INSTR" is the resourceName of an instrument. resourceName can also be a logical IVI name.

ViConstString channelName (in)

Specifies the channel or channels to be acted upon. For example, "0" specifies channel 0 only. This can also be a channel list or a channel range.

A channel list is a comma (,) separated sequence of channel names (for example, "0,2" specifies channels 0 and 2).

A channel range is a lower bound channel followed by a hyphen (-) or colon (:) followed by an upper bound channel (for example, "0-2" specifies channels 0, 1, and 2).

ViInt32 init_options (in)

bit[0] = Specifies whether to reset the instrument during the initialization procedure (enabled) or maintain current configuration in the hardware (disabled).

bit[1] = Specifies whether to reset the *command FIFO* during initialization.

bit[2] = Specifies whether to reset the *result FIFO* during initialization.

bit[3] = Specifies whether to reset the lock status to "unlocked" during initialization.

ViConstString optionString (in)

Specifies available options (case insensitive):

a. "Simulate = 0" OR "Simulate = 1". This option string can place the instrument in simulation mode.
b. "DriverSetup=Model:<model number>"

Example: "Simulate = 0, DriverSetup=Model:AM430e"

ViSession* vi (out)

Returns an instrument handle that you can use to identify the instrument in all subsequent function calls.

Descriptions

AemDCPwr_InitChannels creates a new VISA session to the instrument specified in the resourceName, which provides methods to control and interact with the instrument, and returns a session handle you use to identify the session in all subsequent VI function calls.

This VI also allows you to configure the instrument into known states upon initialization via the init_options:

1. Set bit[0] to "1" to reset the instrument during instrument initialization.

2. Bits[1-2] are meant to clear the command FIFO[1] and result FIFO during initialization. Command FIFO is used to store any data written from the PXI/PXIe bus from the backplane, keeping the instructions in proper order for execution. Result FIFO is used to keep the data to be written to the PXI/PXIe bus back to the host computer. The status of Result FIFO will be read before the data is retrieved back to the host.

[1] FIFO is an acronym for **First In, First Out,** which is an abstraction related to ways of organizing and manipulation of data relative to time and prioritization. This expression describes the principle of a queue processing technique or servicing conflicting demands by ordering process by first-come, first-served (FCFS) behavior: where the persons leave the queue in the order they arrive, or waiting one's turn at a traffic control signal. FIFO blocks are designed in the processor of the instrument to manage data write and read to avoid traffic jam.

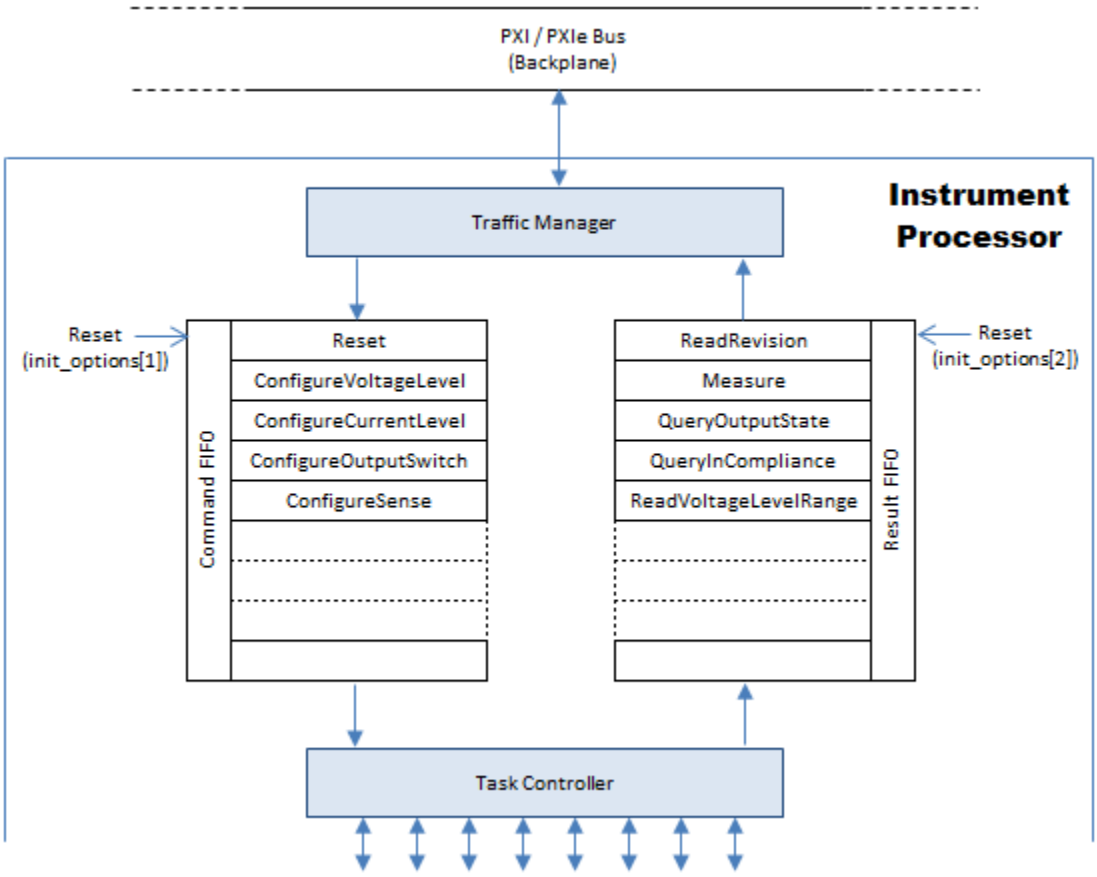

Figure 1: Command and Result FIFO

For example, if you call AemDCPwr_ReadVoltageLevelRange, the instrument driver will write this instruction into the Command FIFO for further processing. Then the instrument driver will check the status of the Result FIFO for its readiness, before retrieving the data back to the host computer.

Consider the following scenario:

a. When you call AemDCPwr_ReadVoltageLevelRange, the instrument driver will write this instruction into the Command FIFO for further processing. However, if the program is suddenly terminated (for whatever reasons), this instruction still remains in the Command FIFO. After the termination happens, if you reinitialize the instrument and try to do some other operations, you will always get the wrong result because AemDCPwr_ReadVoltageLevelRange instruction still remains in the FIFO queue. If you try to read some data, you will get VoltageLevelRange. In this case, when you AemDCPwr_InitChannels, both bits[1-2] need reset.

b. An instrument can be configured to single-site or multi-site mode via AemDCPwr_ConfigureMultiSiteMode. By setting instrument to multi-site mode, you can actually split the available channels in the same instrument to serve multiple site testing. Example, you can use channels 0-1 of AM430e to serve Site-1 and its channels 2-3 to serve Site-2, and both test sites can run either synchronously or asynchronously, for very flexible multi-site configuration.

   However, if you configure the instrument to single-site mode, but you actually run it in multi-site mode, the following may happen:

   Let's imagine Site-1 is running production and are continuously writing instructions to the Command FIFO. Then another test program tries to initialize channels on Site-2 and it clears the Command FIFOs. This interruption may render Site-1 to appear "hang" as the Command FIFO is now empty.

   If the instrument is configured as multi-site mode, then it will "lock" the FIFO until the current instruction is completed before the next instruction write to the FIFO is possible.

3. Bit[3] is used to reset the lock status if it has been unlocked previously.

This function returns zero if successful and non-zero if otherwise.

AemDCPwr_Close

Synopsis

ViStatus AemDCPwr_Close (vi)

Arguments

ViSession vi (in)

Specifies the instrument handle.

Descriptions

AemDCPwr_Close closes the session specified in instrument handle. The instrument will maintain its last running state. This means if power output is enabled when you call this VI, the channel will remain in its current state and continue providing power.

This function returns zero if successful and non-zero if otherwise.

AemDCPwr_ReadRevision

Synopsis

ViStatus AemDCPwr_ReadRevision (vi, instrumentDriverRevision, firmwareRevision)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViChar* instrumentDriverRevision (out)

Space allocated by the caller, and must be at least 32 bytes. It returns the DLL version of AemDCPwr.dll and instruments DLL (eg. AM430.dll, AM471.dll, etc). The DLL version is separated by a hyphen.

ViChar* firmwareRevision (out)

Space allocated by the caller, and must be at least 32 bytes. It returns firmware revision information for the instrument. This argument returns both firmware version of DC and MC separated by a hyphen.

Descriptions

AemDCPwr_ReadRevision returns the revision of the instrument driver as well as the firmware of the instrument specified in instrument handle.

This function returns zero if successful and non-zero if otherwise.

AemDCPwr_ReadCurrentTemperature

Synopsis

ViStatus AemDCPwr_ReadCurrentTemperature (vi, temperature)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViReal64* temperature (out)

Returns the current on-board temperature, in degrees Celsius, of the instrument. This temperature sensor is placed between master and daughter cards.

Descriptions

AemDCPwr_ReadCurrentTemperature returns the temperature measured by a sensor placed between master and daughter card.

This function returns zero if successful and non-zero if otherwise.

AemDCPwr_ReadAmbientTemperature

Synopsis

ViStatus AemDCPwr_ReadAmbientTemperature (vi, temperature)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViReal64* temperature (out)

Returns the current ambient temperature, in degrees Celsius. This temperature sensor is placed on bottom plane of master card to allow it to sense the air flow of the chassis effectively.

Descriptions

AemDCPwr_ReadAmbientTemperature returns the temperature measured by a sensor placed at the bottom plane of master card.

This function returns zero if successful and non-zero if otherwise.

AemDCPwr_ReadSerialNumber

Synopsis

ViStatus AemDCPwr_ReadSerialNumber (vi, sn)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViChar* sn (out)

Space allocated by the caller, and should be at least 64 bytes. It returns both master and daughter card serial numbers separated by a hyphen.

Descriptions

AemDCPwr_ReadSerialNumber returns the serial number of both master and daughter cards.

This function returns zero if successful and non-zero if otherwise.

AemDCPwr_Reset

Synopsis

ViStatus AemDCPwr_Reset (vi)

Arguments

ViSession vi (in)

Specifies the instrument handle.

Descriptions

AemDCPwr_Reset resets the instrument specified by the instrument handle, including all session attributes to default value on the hardware, and then turns off all channels.

This function returns zero if successful and non-zero if otherwise.

AemDCPwr_ResetChannel

Synopsis

ViStatus AemDCPwr_ResetChannel (vi, channelName)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViConstString channelName (in)

Specifies the channel or channels to be acted upon. For example, "0" specifies channel 0 only. This can also be a channel list or a channel range.

A channel list is a comma (,) separated sequence of channel names (for example, "0,2" specifies channels 0 and 2).

A channel range is a lower bound channel followed by a hyphen (-) or colon (:) followed by an upper bound channel (for example, "0-2" specifies channels 0, 1, and 2).

Descriptions

AemDCPwr_ResetChannel resets the selected channels, including all session attributes to default value on the hardware, and then turns off the channels.

This function returns zero if successful and non-zero if otherwise.

AemDCPwr_GetError

Synopsis

ViStatus AemDCPwr_GetError (vi, code, bufferSize, description)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViStatus* code (out)

Returns the error code for the session or execution thread.

ViInt32 bufferSize (in)

Specifies the number of bytes in the ViChar array you specify for description.

If the error description, including the terminating NULL byte, contains more bytes than you indicate in this attribute, the API copies (buffer size - 1) bytes into the buffer, places an ASCII NULL byte at the end of the buffer, and returns the buffer size you must pass to get the entire value.

For example, if the value is ABCDEF and the bufferSize is 4, the API places ABC into the buffer and returns 7.

If you pass 0 for this attribute, you can pass VI_NULL for description.

ViChar* description (out)

Returns the error description for the error code. If there is no description, the API returns an empty string.

Descriptions

AemDCPwr_GetError retrieves the last error code and its description if bufferSize is more than 0. If bufferSize is 0, this API does not clear the error information.

By passing 0 for the bufferSize, you can ascertain the bufferSize required to get the entire error description string and then call the API again with a sufficiently large bufferSize.

If bufferSize is > 0, this API retrieves and then clears the error information for the session. The error information describes the last error that occurred since you last called AemDCPwr_GetError or AemDCPwr_ClearError.

This function returns zero if successful and non-zero if otherwise.

AemDCPwr_GetErrorMessage

Synopsis

ViStatus AemDCPwr_GetErrorMessage (vi, errorCode, errorMessage)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViStatus errorCode (in)

Specifies the error code.

ViChar* errorMessage (out)

Returns the error description for the specified errorCode. If there is no description, the API returns an empty string.

Descriptions

AemDCPwr_GetErrorMessage returns the description of the specified errorCode.

This function returns zero if successful and non-zero if otherwise.

AemDCPwr_ClearError

Synopsis

ViStatus AemDCPwr_ClearError (vi)

Arguments

ViSession vi (in)

Specifies the instrument handle.

Descriptions

AemDCPwr_ClearError clears the last error code for the specified instrument handle.

This function returns zero if successful and non-zero if otherwise.

AemDCPwr_ConfigureMultiSiteMode

Synopsis

ViStatus AemDCPwr_ConfigureMultiSiteMode (vi, mode)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViInt32 mode (in)

Specifies the operation mode:

mode	Operation
0	Single-site (Default)
1	Multi-site

Table 1: Operation Mode

Descriptions

AemDCPwr_ConfigureMultiSiteMode allows you configure the specified instrument to either in single-site or multi-site mode.

At single-site mode, lock/unlock operation is not performed hence yields better test time performance. Use bit[3] of init_options of AemDCPwr_InitChannels to clear the lock status upon initialization of instrument.

This function returns zero if successful and non-zero if otherwise.

AemDCPwr_ConfigureOutputResistance

Synopsis

ViStatus AemDCPwr_ConfigureOutputResistance (vi, channelName, resistance, Irange)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViConstString channelName (in)

Specifies the channel or channels to be acted upon. For example, "0" specifies channel 0 only. This can also be a channel list or a channel range.

A channel list is a comma (,) separated sequence of channel names (for example, "0,2" specifies channels 0 and 2).

A channel range is a lower bound channel followed by a hyphen (-) or colon (:) followed by an upper bound channel (for example, "0-2" specifies channels 0, 1, and 2).

ViReal64 resistance (in)

Configures the value of the virtual resistance in series with FH.

ViReal64 Irange (in)

Specifies the targeted current range.

Descriptions

AemDCPwr_ConfigureOutputResistance configures the virtual resistance in series with FH as below settings:

Current Range	Programmable Resistance Range
1uA	±1MΩ
10uA	±100kΩ
100uA	±10kΩ
1mA	±1kΩ
10mA	±100Ω
100mA	±10Ω

Table 2: Current Range

This function returns zero if successful and non-zero if otherwise.

QueryModuleType

Synopsis

ViStatus QueryModuleType (vi, module_type)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViUInt32* module_type (out)

Returns the module type of the specified instrument handle.

Descriptions

QueryModuleType returns the module type of the specified instrument handle.

module_type	Module
0xA10A430e	AM430e
0xA10A471e	AM471e

Table 3: Module Type

This function returns zero if successful and non-zero if otherwise.

ReadSlotAddress

Synopsis

ViStatus ReadSlotAddress (vi, slotAddress)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViInt32* slotAddress (out)

Returns the slot number of the instrument where it is slotted in.

Descriptions

ReadSlotAddress returns the PXI/PXIe slot number of the instrument where it is slotted in.

This function returns zero if successful and non-zero if otherwise.

AemDCPwr_ConfigureVoltageLevel

Synopsis

ViStatus AemDCPwr_ConfigureVoltageLevel (vi, channelName, level)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViConstString channelName (in)

Specifies the channel or channels to be acted upon. For example, "0" specifies channel 0 only. This can also be a channel list or a channel range.

A channel list is a comma (,) separated sequence of channel names (for example, "0,2" specifies channels 0 and 2).

A channel range is a lower bound channel followed by a hyphen (-) or colon (:) followed by an upper bound channel (for example, "0-2" specifies channels 0, 1, and 2).

ViReal64 level (in)

Specifies the voltage level, in volts.

Descriptions

AemDCPwr_ConfigureVoltageLevel sets the voltage level of the specified channel. The voltage level range will be updated automatically based on the voltage level.

In order for the specified level to take effect, the channel must be enabled via AemDCPwr_ConfigureOutputSwitch. The voltage setting is only applicable when the operation of the channel is DVCI (Drive-Voltage Clamp-Current) as configured via AemDCPwr_ConfigureOutputFunction.

This function returns zero if successful and non-zero if otherwise.

AemDCPwr_ConfigureVoltageLevelAndRange

Synopsis

ViStatus AemDCPwr_ConfigureVoltageLevelAndRange (vi, channelName, level, range)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViConstString channelName (in)

Specifies the channel or channels to be acted upon. For example, "0" specifies channel 0 only. This can also be a channel list or a channel range.

A channel list is a comma (,) separated sequence of channel names (for example, "0,2" specifies channels 0 and 2).

A channel range is a lower bound channel followed by a hyphen (-) or colon (:) followed by an upper bound channel (for example, "0-2" specifies channels 0, 1, and 2).

ViReal64 level (in)

Specifies the voltage level, in volts.

ViReal64 range (in)

Specifies the voltage level range, in volts.

Descriptions

AemDCPwr_ConfigureVoltageLevelAndRange sets the voltage level as well as its range of the specified channel.

In order for the specified level to take effect, the channel must be enabled via AemDCPwr_ConfigureOutputSwitch. The voltage setting is only applicable when the operation of the channel is DVCI (Drive-Voltage Clamp-Current) as configured via AemDCPwr_ConfigureOutputFunction.

This function returns zero if successful and non-zero if otherwise.

AemDCPwr_ConfigureCurrentLimit

Synopsis

ViStatus AemDCPwr_ConfigureCurrentLimit (vi, channelName, behavior, limit)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViConstString channelName (in)

Specifies the channel or channels to be acted upon. For example, "0" specifies channel 0 only. This can also be a channel list or a channel range.

A channel list is a comma (,) separated sequence of channel names (for example, "0,2" specifies channels 0 and 2).

A channel range is a lower bound channel followed by a hyphen (-) or colon (:) followed by an upper bound channel (for example, "0-2" specifies channels 0, 1, and 2).

ViInt32 behavior (in)

This is a reserved argument. Set it to "0".

ViReal64 limit (in)

Specifies the current limit, in amps.

Descriptions

AemDCPwr_ConfigureCurrentLimit sets the current limit as well as its range of the specified channel.

In order for the specified level to take effect, the channel must be enabled via AemDCPwr_ConfigureOutputSwitch. The current limit setting is only applicable when the operation of the channel is DVCI (Drive-Voltage Clamp-Current) as configured via AemDCPwr_ConfigureOutputFunction.

This function returns zero if successful and non-zero if otherwise.

AemDCPwr_ConfigureCurrentLimitAndRange

Synopsis

ViStatus AemDCPwr_ConfigureCurrentLimitAndRange (vi, channelName, behavior, limit, range)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViConstString channelName (in)

Specifies the channel or channels to be acted upon. For example, "0" specifies channel 0 only. This can also be a channel list or a channel range.

A channel list is a comma (,) separated sequence of channel names (for example, "0,2" specifies channels 0 and 2).

A channel range is a lower bound channel followed by a hyphen (-) or colon (:) followed by an upper bound channel (for example, "0-2" specifies channels 0, 1, and 2).

ViInt32 behavior (in)

This is a reserved argument. Set it to "0".

ViReal64 limit (in)

Specifies the current limit, in amps.

ViReal64 range (in)

Specifies the current limit range, in amps.

Descriptions

AemDCPwr_ConfigureCurrentLimitAndRange sets the current limit as well as its range of the specified channel.

In order for the specified level to take effect, the channel must be enabled via AemDCPwr_ConfigureOutputSwitch. The current limit setting is only applicable when the

operation of the channel is DVCI (Drive-Voltage Clamp-Current) as configured via AemDCPwr_ConfigureOutputFunction.

This function returns zero if successful and non-zero if otherwise.

AemDCPwr_ConfigureCurrentLevel

Synopsis

ViStatus AemDCPwr_ConfigureCurrentLevel (vi, channelName, level)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViConstString channelName (in)

Specifies the channel or channels to be acted upon. For example, "0" specifies channel 0 only. This can also be a channel list or a channel range.

A channel list is a comma (,) separated sequence of channel names (for example, "0,2" specifies channels 0 and 2).

A channel range is a lower bound channel followed by a hyphen (-) or colon (:) followed by an upper bound channel (for example, "0-2" specifies channels 0, 1, and 2).

ViReal64 level (in)

Specifies the current level, in amps.

Descriptions

AemDCPwr_ConfigureCurrentLevel sets the current level of the specified channel. The current level range will be updated automatically based on the current level.

In order for the specified level to take effect, the channel must be enabled via AemDCPwr_ConfigureOutputSwitch. The voltage setting is only applicable when the operation of the channel is DICV (Drive-Current Clamp-Voltage) as configured via AemDCPwr_ConfigureOutputFunction.

This function returns zero if successful and non-zero if otherwise.

AemDCPwr_ConfigureCurrentLevelAndRange

Synopsis

ViStatus AemDCPwr_ConfigureCurrentLevelAndRange (vi, channelName, level, range)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViConstString channelName (in)

Specifies the channel or channels to be acted upon. For example, "0" specifies channel 0 only. This can also be a channel list or a channel range.

A channel list is a comma (,) separated sequence of channel names (for example, "0,2" specifies channels 0 and 2).

A channel range is a lower bound channel followed by a hyphen (-) or colon (:) followed by an upper bound channel (for example, "0-2" specifies channels 0, 1, and 2).

ViReal64 level (in)

Specifies the current level, in amps.

ViReal64 range (in)

Specifies the current level range, in amps.

Descriptions

AemDCPwr_ConfigureCurrentLevelAndRange sets the current level as well as its range of the specified channel.

In order for the specified level to take effect, the channel must be enabled via AemDCPwr_ConfigureOutputSwitch. The current setting is only applicable when the operation of the channel is DICV (Drive-Current Clamp-Voltage) as configured via AemDCPwr_ConfigureOutputFunction.

This function returns zero if successful and non-zero if otherwise.

AemDCPwr_ConfigureVoltageLimit

Synopsis

ViStatus AemDCPwr_ConfigureVoltageLimit (vi, channelName, behavior, limit)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViConstString channelName (in)

Specifies the channel or channels to be acted upon. For example, "0" specifies channel 0 only. This can also be a channel list or a channel range.

A channel list is a comma (,) separated sequence of channel names (for example, "0,2" specifies channels 0 and 2).

A channel range is a lower bound channel followed by a hyphen (-) or colon (:) followed by an upper bound channel (for example, "0-2" specifies channels 0, 1, and 2).

ViInt32 behavior (in)

This is a reserved argument. Set it to "0".

ViReal64 limit (in)

Specifies the voltage limit, in volts.

Descriptions

AemDCPwr_ConfigureVoltageLimit sets the voltage limit as well as its range of the specified channel.

In order for the specified level to take effect, the channel must be enabled via AemDCPwr_ConfigureOutputSwitch. The voltage limit setting is only applicable when the operation of the channel is DICV (Drive-Current Clamp-Voltage) as configured via AemDCPwr_ConfigureOutputFunction.

This function returns zero if successful and non-zero if otherwise.

AemDCPwr_ConfigureVoltageLimitAndRange

Synopsis

ViStatus AemDCPwr_ConfigureVoltageLimitAndRange (vi, channelName, limit, range)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViConstString channelName (in)

Specifies the channel or channels to be acted upon. For example, "0" specifies channel 0 only. This can also be a channel list or a channel range.

A channel list is a comma (,) separated sequence of channel names (for example, "0,2" specifies channels 0 and 2).

A channel range is a lower bound channel followed by a hyphen (-) or colon (:) followed by an upper bound channel (for example, "0-2" specifies channels 0, 1, and 2).

ViReal64 limit (in)

Specifies the voltage limit, in volts.

ViReal64 range (in)

Specifies the voltage limit range, in volts.

Descriptions

AemDCPwr_ConfigureVoltageLimitAndRange sets the voltage limit as well as its range of the specified channel.

In order for the specified level to take effect, the channel must be enabled via AemDCPwr_ConfigureOutputSwitch. The voltage limit setting is only applicable when the operation of the channel is DICV (Drive-Current Clamp-Voltage) as configured via AemDCPwr_ConfigureOutputFunction.

This function returns zero if successful and non-zero if otherwise.

AemDCPwr_ConfigureOutputFunction

Synopsis

ViStatus AemDCPwr_ConfigureOutputFunction (vi, channelName, function)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViConstString channelName (in)

Specifies the channel or channels to be acted upon. For example, "0" specifies channel 0 only. This can also be a channel list or a channel range.

A channel list is a comma (,) separated sequence of channel names (for example, "0,2" specifies channels 0 and 2).

A channel range is a lower bound channel followed by a hyphen (-) or colon (:) followed by an upper bound channel (for example, "0-2" specifies channels 0, 1, and 2).

ViInt32 function (in)

Select DVCI or DICV mode.

Descriptions

AemDCPwr_ConfigureOutputFunction configures the channel to operate either in DVCI (Drive-Voltage Clamp Current) or DICV (Drive-Current Clamp Voltage) mode:

function	Setting
DVCI	0 (Default)
DICV	1

Table 4: Output Function

This function returns zero if successful and non-zero if otherwise.

AemDCPwr_ConfigureOutputSwitch

Synopsis

ViStatus AemDCPwr_ConfigureOutputSwitch (vi, channelName, switch)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViConstString channelName (in)

Specifies the channel or channels to be acted upon. For example, "0" specifies channel 0 only. This can also be a channel list or a channel range.

A channel list is a comma (,) separated sequence of channel names (for example, "0,2" specifies channels 0 and 2).

A channel range is a lower bound channel followed by a hyphen (-) or colon (:) followed by an upper bound channel (for example, "0-2" specifies channels 0, 1, and 2).

ViInt32 switch (in)

Configure the output state of the selected channels.

Descriptions

AemDCPwr_ConfigureOutputSwitch configures the output switch of the channel to one of the operations below:

switch	Description
0	Switch off the channel
1	Switch on the channel
2	Switch on the channel, and insert a 100K resistor at Force-High output

Table 5: Output Switch

The 100K series resistor at the Force-High output, together with local sense operation (refer to AemDCPwr_ConfigureSense), allow the following possibilities:

1. Allow fast charging of capacitor load initially (when series resistor disabled) and low current measurement (by enabling the resistor again).
2. Prevent fully charged up capacitor from discharging when the channel is changing from high current range to low current range for leakage measurement.
3. Prevent small level of noise voltage from translating to noise current due to low impedance of capacitor at noise frequency. This allows less averaging to get the DC average current (less test time).
4. Stabilize the channel control loop at high capacitance load. This is critical, because at low current range the output current sense resistor is large, and will cause significant RC phase shift when loaded with large capacitor.

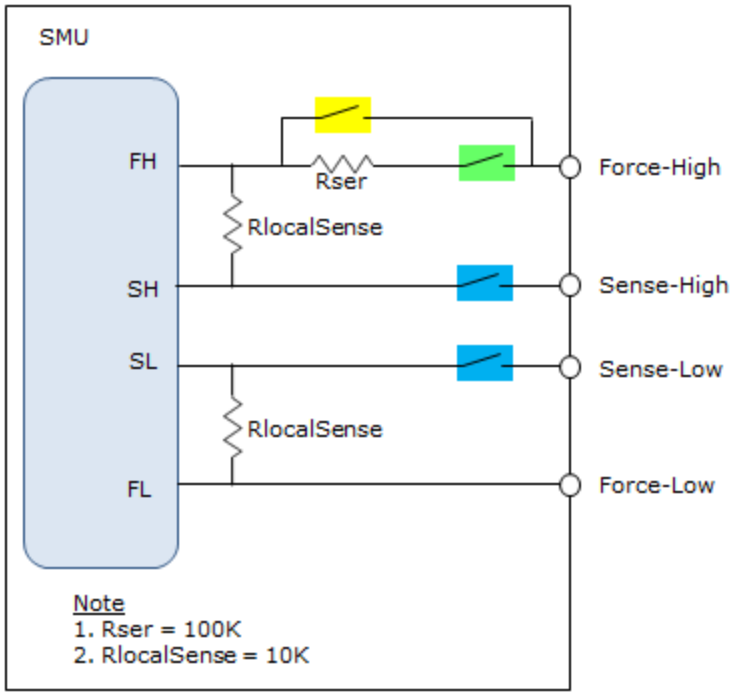

Figure 2: Output Switch Configuration

In the presence of huge capacitor when leakage current measurement is required, the following steps are recommended:

1. Configure channel to higher current limit. This is to allow fast charging of capacitor load.
2. When capacitor is fully charged up, which can be estimated by dt=C(dV)/I, turn on the series resistor immediately.
3. Configure channel to lower current limit and range to perform leakage current accurately.

4. Once measurement is completed, configure the channel back to original settings, and then turn off the resistor.

This function returns zero if successful and non-zero if otherwise.

AemDCPwr_ ConfigureOutputEnabled

Synopsis

ViStatus AemDCPwr_ ConfigureOutputEnabled (vi, channelName, enabled)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViConstString channelName (in)

Specifies the channel or channels to be acted upon. For example, "0" specifies channel 0 only. This can also be a channel list or a channel range.

A channel list is a comma (,) separated sequence of channel names (for example, "0,2" specifies channels 0 and 2).

A channel range is a lower bound channel followed by a hyphen (-) or colon (:) followed by an upper bound channel (for example, "0-2" specifies channels 0, 1, and 2).

ViBoolean enabled (in)

Specifies to enabled (true) or disable (false) the SMU feedback control loop.

Descriptions

Specifies to enabled (true) or disable (false) the SMU feedback control loop.

This function returns zero if successful and non-zero if otherwise.

AemDCPwr_ConfigureOutputTransient

Synopsis

ViStatus AemDCPwr_ConfigureOutputTransient (vi, channelName, transient)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViConstString channelName (in)

Specifies the channel or channels to be acted upon. For example, "0" specifies channel 0 only. This can also be a channel list or a channel range.

A channel list is a comma (,) separated sequence of channel names (for example, "0,2" specifies channels 0 and 2).

A channel range is a lower bound channel followed by a hyphen (-) or colon (:) followed by an upper bound channel (for example, "0-2" specifies channels 0, 1, and 2).

ViInt32 transient (in)

Specifies the transient (bandwidth) setting of the specified channel.

Descriptions

AemDCPwr_ConfigureOutputTransient configures the transient (bandwidth) response of the specified channel:

transient	Description
0	Slow
1	Normal (Default)
2	Fast
3	Custom bandwidth store location 0
4	Custom bandwidth store location 1
5	Custom bandwidth store location 2
6	Custom bandwidth store location 3
7	Custom bandwidth store location 4

Table 6: Output Transient

Fast transient response shortens the rise time and settling time of the signal, but at the risk of having unwanted overshoots. On the other hand, slow transient response allows stable and smooth signal but at a slower pace. In addition, up to 5 custom bandwidth settings are supported by using AemDCPwr_ComputeAIBandwidth. The custom bandwidth setting must be pre-calculated using AemDCPwr_ComputeAIBandwidth and be placed in proper store_location. For example, transient=3 refers to custom bandwidth setting stored in location 0. Refer to AemDCPwr_ComputeAIBandwidth for more information.

Anyway, you need to observe the waveform using oscilloscope to get the best response for your application.

This function returns zero if successful and non-zero if otherwise.

AemDCPwr_ComputeAIBandwidth

Synopsis

ViStatus AemDCPwr_ComputeAIBandwidth (vi, channelName, mode, vRange, iRange, drive_settling_time, clamp_settling_time, resistance, capacitance, reserved, store_location)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViConstString channelName (in)

Specifies the channel or channels to be acted upon. For example, "0" specifies channel 0 only. This can also be a channel list or a channel range.

A channel list is a comma (,) separated sequence of channel names (for example, "0,2" specifies channels 0 and 2).

A channel range is a lower bound channel followed by a hyphen (-) or colon (:) followed by an upper bound channel (for example, "0-2" specifies channels 0, 1, and 2).

ViInt32 mode (in)

Specifies whether the load is resistive or capacitive.

mode	Load
0	Resistor
1	Capacitor

Table 7: mode

ViReal64 vRange (in)

Specifies the voltage level range, in volts.

ViReal64 iRange (in)

Specifies the current level range, in amps.

ViReal64 drive_settling_time (in)

Specifies the desired source settling time, in seconds.

ViReal64 clamp_settling_time (in)

Specifies the desired settling time for compliance, in seconds.

ViReal64 resistance (in)

Specifies the resistance of the resistor connected to the channel. This value will be ignored if the specified load is capacitor (mode = 1).

ViReal64 capacitance (in)

Specifies the capacitance of the capacitor connected to the channel. This value will be ignored if the specified load is resistor (mode = 0).

ViReal64 reserved (in)

This is a reserved argument. Set it to "0".

ViInt32 store_location (in)

Specifies the location of where the calculated bandwidth setting is placed in AemDCPwr_ConfigureOutputTransient. Available store_location are 0 to 4.

Descriptions

AemDCPwr_ComputeAIBandwidth calculates the bandwidth setting based on voltage and current ranges (vRange, iRange), the desired settling time (drive_settling_time, clamp_settling_time), as well as the load (resistance, capacitance).

Up to 5 custom bandwidth setting are supported. You can calculate a bandwidth setting based on the combinations of inputs, and set it to store_location "0". When you want to activate this setting, call AemDCPwr_ConfigureOutputTransient, by specifying transient to "3", which corresponds to store_location "0".

This function returns zero if successful and non-zero if otherwise. An error will be returned if the desired settling time cannot be achieved.

AemDCPwr_ConfigurePulse

Synopsis

ViStatus AemDCPwr_ConfigurePulse (vi, channelName, function, range, base, pulse, pulse_s, hold_s, reserved, measurePercentage, cycles)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViConstString channelName (in)

Specifies the channel or channels to be acted upon. For example, "0" specifies channel 0 only. This can also be a channel list or a channel range.

A channel list is a comma (,) separated sequence of channel names (for example, "0,2" specifies channels 0 and 2).

A channel range is a lower bound channel followed by a hyphen (-) or colon (:) followed by an upper bound channel (for example, "0-2" specifies channels 0, 1, and 2).

ViInt32 function (in)

Select DVCI or DICV mode.

function	Description
0	The channel maintains a constant voltage by adjusting the current (DVCI)
1	The channel maintains a constant current by adjusting the voltage (DICV)

Table 8: function selection

ViReal64 range (in)

Range is vRange if DVCI, iRange if DICV.

ViReal64 base (in)

Specifies the base value of the pulse.

ViReal64 pulse (in)

Specifies the pulse value of the pulse.

ViReal64 pulse_s (in)

Specifies the pulse width in second

0<= pulse_s<=20ms

ViReal64 hold_s (in)

Specifies the period of the base value in second.

100us <= hold_s <= 500ms

ViInt32 reserved (in)

This is a reserved argument. Set it to "0".

ViReal64 measurePercentage (in)

Specifies the time for measurement during drive pulse.

ViInt32 cycles (in)

Specifies the number of pulses targeted.

1<= cycles <= 64

Descriptions

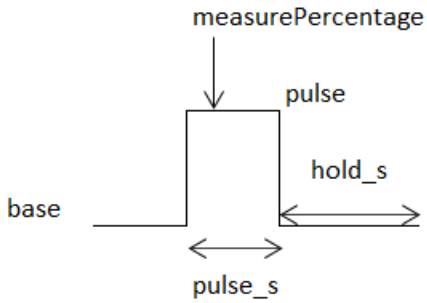

Figure 3 : Generated pulse

From the figure 3, it shows the parameters of base, pulse, pulse_s, hold_s and measurePercentage. SMU will make measurement according to the user input measurePercentage. For example, measurePercentage is 40, and pulse_s is 10ms, then, the pulse measurement starts when t = 4ms from the pulse rising edge.

AemDCPwr_ConfigurePulse generates a pulse with the desired settings and measures the voltage and current when driving pulse. The measurements can be retrieved with AemDCPwr_MeasurePulseVI.

This function returns zero if successful and non-zero if otherwise.

AemDCPwr_ConfigureExtenderModule

Synopsis

ViStatus AemDCPwr_ConfigureExtenderModule (vi, moduleStatus)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViInt32* moduleStatus (out)

Returns the status of the extender module.

moduleStatus	Description
0	Extender card not detected and power supply generation is disabled.
1	Extender card is detected and power supply generation is enabled.

Table 9 : Module Status

Descriptions

AemDCPwr_ConfigureExtenderModule queries the status of the extender module.

User needs to connect the extender card and do not directly connect SMU to any point when call this API. This API need to be called before sourcing pulse.

This function returns zero if successful and non-zero if otherwise.

AemDCPwr_ConfigureContinuousPulse

Synopsis

ViStatus AemDCPwr_ConfigureContinuousPulse (vi, channelName, function,base, pulse, range, pulse_s,hold_s, cycle, infiniteCycle)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViConstString channelName (in)

Specifies the channel or channels to be acted upon. For example, "0" specifies channel 0 only. This can also be a channel list or a channel range.

A channel list is a comma (,) separated sequence of channel names (for example, "0,2" specifies channels 0 and 2).

A channel range is a lower bound channel followed by a hyphen (-) or colon (:) followed by an upper bound channel (for example, "0-2" specifies channels 0, 1, and 2).

ViInt32 function (in)

Select DVCI or DICV mode.

function	Description
0	The channel maintains a constant voltage by adjusting the current (DVCI)
1	The channel maintains a constant current by adjusting the voltage (DICV)

Table 10: Output Function

ViReal64 base (in)

Specifies the base value of the pulse.

ViReal64 pulse (in)

Specifies the pulse value of the pulse.

ViReal64 range (in)

Range is vRange if DVCI, iRange if DICV.

ViReal64 pulse_s (in)

Specifies the pulse width in second

0<= pulse_s<=20ms

ViReal64 hold_s (in)

Specifies the period of the base value in second.

100us <= hold_s <= 500ms

ViInt32 reserved (in)

This is a reserved argument. Set it to "0".

ViReal64 measurePercentage (in)

Specifies the time for measurement during drive pulse.

ViInt32 cycles (in)

Specifies the number of pulses targeted.

1<= cycles <= 64

ViInt32 infiniteCycle (in)

Specifies the number of pulses cycle.

infiniteCycle	Description
0	The number of pulse is referred to argument cycle.
1	Infinite generation of pulse.

Table 11: Infinite Cycle Selection

Descriptions

AemDCPwr_ConfigureContinuousPulse generates a number of cycles of pulse or infinite pulse with the desired current or voltage.

This function returns zero if successful and non-zero if otherwise.

AemDCPwr_StartContinuousPulse

Synopsis

ViStatus AemDCPwr_StartContinuousPulse (vi, channelName)

Arguments

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViConstString channelName (in)

Specifies the channel or channels to be acted upon. For example, "0" specifies channel 0 only. This can also be a channel list or a channel range.

A channel list is a comma (,) separated sequence of channel names (for example, "0,2" specifies channels 0 and 2).

A channel range is a lower bound channel followed by a hyphen (-) or colon (:) followed by an upper bound channel (for example, "0-2" specifies channels 0, 1, and 2).

Descriptions

The continuous pulse generation is only started when AemDCPwr_StartContinuousPulse is called.

This function returns zero if successful and non-zero if otherwise.

AemDCPwr_StopContinuousPulse

Synopsis

ViStatus AemDCPwr_StopContinuousPulse (vi, channelName)

Arguments

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViConstString channelName (in)

Specifies the channel or channels to be acted upon. For example, "0" specifies channel 0 only. This can also be a channel list or a channel range.

A channel list is a comma (,) separated sequence of channel names (for example, "0,2" specifies channels 0 and 2).

A channel range is a lower bound channel followed by a hyphen (-) or colon (:) followed by an upper bound channel (for example, "0-2" specifies channels 0, 1, and 2).

Descriptions

This API configures the bandwidth setting for acquire array measurement.

This function returns zero if successful and non-zero if otherwise.

AemDCPwr_ConfigureAcquireArrayBWLimit

Synopsis

ViStatus AemDCPwr_ConfigureAcquireArrayBWLimit (vi, channelName, setting)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViConstString channelName (in)

Specifies the channel or channels to be acted upon. For example, "0" specifies channel 0 only. This can also be a channel list or a channel range.

A channel list is a comma (,) separated sequence of channel names (for example, "0,2" specifies channels 0 and 2).

A channel range is a lower bound channel followed by a hyphen (-) or colon (:) followed by an upper bound channel (for example, "0-2" specifies channels 0, 1, and 2).

ViInt32 setting (in)

Configures the bandwidth setting for acquire array measurement.

Descriptions

AemDCPwr_ConfigureAcquireArrayBWLimit allows user to configure the bandwidth setting during voltage or current array measurement. This setting does not affect normal measurements.

This function returns zero if successful and non-zero if otherwise.

AemDCPwr_ConfigureSamplingTime

Synopsis

ViStatus AemDCPwr_ConfigureSamplingTime (vi, channelName, samplingTime, units)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViConstString channelName (in)

Specifies the channel or channels to be acted upon. For example, "0" specifies channel 0 only. This can also be a channel list or a channel range.

A channel list is a comma (,) separated sequence of channel names (for example, "0,2" specifies channels 0 and 2).

A channel range is a lower bound channel followed by a hyphen (-) or colon (:) followed by an upper bound channel (for example, "0-2" specifies channels 0, 1, and 2).

ViReal64 samplingTime (in)

Specifies the sampling time.

ViInt32 units (in)

Specifies the unit of the sampling time:

units	Description
0	Seconds
1	PLC

Table 12: Sampling Time Unit

Descriptions

AemDCPwr_ConfigureSamplingTime configures the sampling time or NPLC (number of power line cycle) on the specified channel(s). The supported values depend on the units.

The measurement result is the average value of all samples taken for nth number of power line cycle (PLC). For example, if the line frequency is set to 50Hz (period is 20ms), then 1 PLC equals to 20ms. The measured data is a resultant of averaged of raw samples captured by ADC for 20ms.

The NPLC setting of an instrument allows adjustment of the tradeoff between speed and accuracy. The greater the number of power line cycles, the greater noise rejection and better resolution the signal value will be.

The following table lists some examples of sampling times for the instrument, in PLC format:

PLC	60Hz Power Line Frequency	50Hz Power Line Frequency
8	133.3 ms	160 ms
4	66.66 ms	80 ms
2	33.33 ms	40 ms
1	16.66 ms	20 ms
1/2	8.33 ms	10 ms
1/4	4.16 ms	5 ms
1/8	2.08 ms	2.5 ms
1/16	1.04 ms	1.25 ms
1/32	520 us	625 us
1/64	260 us	312 us

Table 13: PLC and Corresponding Frequencies

This function returns zero if successful and non-zero if otherwise.

AemDCPwr_ConfigurePLF

Synopsis

ViStatus AemDCPwr_ConfigurePLF (vi, plf)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViReal64 plf (in)

Specifies the power line frequency.

Descriptions

AemDCPwr_ConfigurePLF specifies the frequency of the AC power line:

plf	Description
50.0	50Hz power line frequency (Default)
60.0	60Hz power line frequency

Table 14: Power Line Frequency

This function returns zero if successful and non-zero if otherwise.

AemDCPwr_ConfigureSense

Synopsis

ViStatus AemDCPwr_ConfigureSense (vi, channelName, sense)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViConstString channelName (in)

Specifies the channel or channels to be acted upon. For example, "0" specifies channel 0 only. This can also be a channel list or a channel range.

A channel list is a comma (,) separated sequence of channel names (for example, "0,2" specifies channels 0 and 2).

A channel range is a lower bound channel followed by a hyphen (-) or colon (:) followed by an upper bound channel (for example, "0-2" specifies channels 0, 1, and 2).

ViInt32 sense (in)

Specifies local or remote sense operation.

Descriptions

AemDCPwr_ConfigureSense selects either local or remote sense operation:

sense	Description
0	Local sense (Default)
1	Remote sense

Table 15: Local/Remote Sense Operation

This function returns zero if successful and non-zero if otherwise.

AemDCPwr_Measure

Synopsis

ViStatus AemDCPwr_Measure (vi, channelName, measurementType, measurement)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViConstString channelName (in)

Specifies the channel or channels to be acted upon. For example, "0" specifies channel 0 only. This can also be a channel list or a channel range.

A channel list is a comma (,) separated sequence of channel names (for example, "0,2" specifies channels 0 and 2).

A channel range is a lower bound channel followed by a hyphen (-) or colon (:) followed by an upper bound channel (for example, "0-2" specifies channels 0, 1, and 2).

ViInt32 measurementType (in)

Specifies voltage or current measurement.

ViReal64* measurement (out)

Returns the measured results.

Descriptions

AemDCPwr_Measure returns the measured data based on measurementType:

measurementType	Description
0	Measure Current
1	Measure Voltage

Table 16: Voltage/Current Measurement Selection

AemDCPwr_Measure allows you to perform measurement on multiple channels. You need to allocate sufficient memory location (in array form) and pass it to measurement. The measurements in the array are returned in the same order as the channel(s) specified.

This function returns zero if successful and non-zero if otherwise.

AemDCPwr_MeasureArray

Synopsis

ViStatus AemDCPwr_MeasureArray (vi, channelName, printToTxt, measurementType, measurement)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViConstString channelName (in)

Specifies the channel to be acted upon. Only one channel is allowed at one time.

ViBoolean printToTxt (in)

Specifies whether to print out the raw data in a text document.

ViInt32 measurementType (in)

Specifies the type of measurement.

ViReal64* measurement (out)

Returns an array of measured results.

Descriptions

AemDCPwr_MeasureArray returns the measured data based on measurementType:

measurementType	Description
0	Measure Current
1	Measure Voltage
2	Measure both current and voltage

Table 17: Measurement Type

AemDCPwr_MeasureArray can only work on one channel at one time. You must ensure that enough memory is allocated for output measurement. For example, you need to allocate 20000 for output measurement if:

1. NPLC is 1
2. Power line frequency is 50Hz
3. measurementType is 0 or 1

If measurementType is 2, you must allocate double the size of output measurement. The measurement output starts with all voltage values followed by current values.

Maximum NPLC supported by this function is 3 due to limited on-board memory space.

This function returns zero if successful and non-zero if otherwise.

AemDCPwr_MeasureMultiple

Synopsis

ViStatus AemDCPwr_MeasureMultiple (vi, channelName, voltageMeasurements, currentMeasurements)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViConstString channelName (in)

Specifies the channel or channels to be acted upon. For example, "0" specifies channel 0 only. This can also be a channel list or a channel range.

A channel list is a comma (,) separated sequence of channel names (for example, "0,2" specifies channels 0 and 2).

A channel range is a lower bound channel followed by a hyphen (-) or colon (:) followed by an upper bound channel (for example, "0-2" specifies channels 0, 1, and 2).

ViReal64* voltageMeasurements (out)

Space allocated by the caller. Returns an array of voltage measurements. The measurements in the array are returned in the same order as the channels specified in channelName argument.

ViReal64* currentMeasurements (out)

Space allocated by the caller. Returns an array of current measurements. The measurements in the array are returned in the same order as the channels specified in channelName argument.

Descriptions

AemDCPwr_MeasureMultiple returns the arrays of the measured voltage and current values on the specified output channel(s). Each call to this function blocks other function calls until the measurements are returned from the device. The order of the measurements returned in the array corresponds to the order on the specified output channel(s).

This function returns zero if successful and non-zero if otherwise.

AemDCPwr_ConfigureMeasureMode

Synopsis

ViStatus AemDCPwr_ConfigureMeasureMode (vi, channelName, mode)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViConstString channelName (in)

Specifies the channel or channels to be acted upon. For example, "0" specifies channel 0 only. This can also be a channel list or a channel range.

A channel list is a comma (,) separated sequence of channel names (for example, "0,2" specifies channels 0 and 2).

A channel range is a lower bound channel followed by a hyphen (-) or colon (:) followed by an upper bound channel (for example, "0-2" specifies channels 0, 1, and 2).

ViInt32 mode (in)

Specifies measurement operation mode.

Descriptions

AemDCPwr_ConfigureMeasureMode defines how AemDCPwr_AcquireMultiple or AemDCPwr_AcquireArray function start measurement data acquisition.

mode	Description
0	AemDCPwr_AcquireMultiple starts immediately after AemDCPwr_ConfigureVoltageLevel, AemDCPwr_ConfigureVoltageLevelAndRange, AemDCPwr_ConfigureCurrentLevel or AemDCPwr_ConfigureCurrentLevelAndRange functions. It will not wait until the channel output reach the desired voltage or current level.
1	AemDCPwr_AcquireMultiple starts after receive the trigger signal. The selection of trigger signal is based on input setting of AemDCPwr_ConfigureInputTriggerSelect function.
2	AemDCPwr_AcquireMultiple starts upon being called. Use this for

3	AemDCPwr_AcquireArray starts after receiving the trigger signal. The selection of trigger signal is based on input setting of AemDCPwr_ConfigureInputTriggerSelect function.

Table 18: Measurement Mode

This function returns zero if successful and non-zero if otherwise.

AemDCPwr_AcquireMultiple

Synopsis

ViStatus AemDCPwr_AcquireMultiple (vi, channelName, timeout, count, voltageMeasurements, currentMeasurements, inCompliance, actualCount)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViConstString channelName (in)

Specifies the channel to be acted upon. Only one channel is allowed at one time.

ViReal64 timeout (in)

Specifies the maximum time allowed for this function to complete, in seconds. If the function does not complete within this time interval, an error will be returned.

ViInt32 count (in)

Specifies the number of measurements to fetch.

ViReal64* voltageMeasurements (out)

Space allocated by the caller. Returns an array of voltage measurement.

ViReal64* currentMeasurements (out)

Space allocated by the caller. Returns an array of current measurement.

ViBoolean* inCompliance (out)

Space allocated by the caller. Returns an array of boolean values indicating whether the output was in compliance at the time the measurement was taken.

ViInt32* actualCount (out)

Indicates the number of measured values actually retrieved from the instrument.

Descriptions

AemDCPwr_AcquireMultiple retrieves an array of voltage measurements, an array of current measurements, and an array of compliance measurements that are previously taken and are stored in the buffer. AemDCPwr_AcquireMultiple does not make any measurements. It only retrieves a set of measurements taken previously. Refer to AemDCPwr_ConfigureMeasureMode on how to make us of AemDCPwr_AcquireMultiple.

This function returns zero if successful and non-zero if otherwise.

AemDCPwr_AcquireArray

Synopsis

ViStatus AemDCPwr_AcquireArray (vi, channelName, timeout, voltageMeasurements, currentMeasurements, actualCount)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViConstString channelName (in)

Specifies the channel to be acted upon. Only one channel is allowed at one time.

ViReal64 timeout (in)

Specifies the maximum time allowed for this function to complete, in seconds. If the function does not complete within this time interval, an error will be returned.

ViReal64* voltageMeasurements (out)

Space allocated by the caller. Returns an array of voltage measurement.

ViReal64* currentMeasurements (out)

Space allocated by the caller. Returns an array of current measurement.

ViInt32* actualCount (out)

Indicates the number of measured values actually retrieved from the instrument.

Descriptions

AemDCPwr_AcquireArray retrieves an array of voltage measurements and an array of current measurements that are previously taken and are stored in the buffer. AemDCPwr_AcquireArray does not make any measurements. It only retrieves a set of measurements taken previously. Refer to AemDCPwr_ConfigureMeasureMode on how to make use of AemDCPwr_AcquireArray.

The actualCount returned depends on the NPLC setting. If NPLC is 1, then, actualCount should be 20000. Memory must be allocated by user for voltageMeasurements and currentMeasurements.

Note that you must not change NPLC after calling AemDCPwr_ConfigureMeasureMode (mode=3) until AemDCPwr_AcquireArray is called.

This function returns zero if successful and non-zero if otherwise.

AemDCPwr_AcquireCurrentArray

Synopsis

ViStatus AemDCPwr_AcquireCurrentArray (vi, channelName, timeout, currentMeasurements, actualCount)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViConstString channelName (in)

Specifies the channel to be acted upon. Only one channel is allowed at one time.

ViReal64 timeout (in)

Specifies the maximum time allowed for this function to complete, in seconds. If the function does not complete within this time interval, an error will be returned.

ViReal64* currentMeasurements (out)

Space allocated by the caller. Returns an array of current measurement.

ViInt32* actualCount (out)

Indicates the number of measured values actually retrieved from the instrument.

Descriptions

AemDCPwr_AcquireCurrentArray retrieves an array of current measurements that are previously taken and are stored in the buffer. AemDCPwr_AcquireCurrentArray does not make any measurements. It only retrieves a set of measurements taken previously. Refer to AemDCPwr_ConfigureMeasureMode on how to make use of AemDCPwr_AcquireCurrentArray.

The actualCount returned depends on the NPLC setting. If NPLC is 1, then, actualCount should be 20000. Memory must be allocated by user for voltageMeasurements and currentMeasurements.

Note that you must not change NPLC after calling AemDCPwr_ConfigureMeasureMode (mode=3) until AemDCPwr_AcquireCurrentArray is called.

This function returns zero if successful and non-zero if otherwise.

AemDCPwr_AcquireVoltageArray

Synopsis

ViStatus AemDCPwr_AcquireVoltageArray (vi, channelName, timeout, voltageMeasurements, actualCount)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViConstString channelName (in)

Specifies the channel to be acted upon. Only one channel is allowed at one time.

ViReal64 timeout (in)

Specifies the maximum time allowed for this function to complete, in seconds. If the function does not complete within this time interval, an error will be returned.

ViReal64* voltageMeasurements (out)

Space allocated by the caller. Returns an array of voltage measurement.

ViInt32* actualCount (out)

Indicates the number of measured values actually retrieved from the instrument.

Descriptions

AemDCPwr_AcquireVoltageArray retrieves an array of voltage measurements that are previously taken and are stored in the buffer. AemDCPwr_AcquireVoltageArray does not make any measurements. It only retrieves a set of measurements taken previously. Refer to AemDCPwr_ConfigureMeasureMode on how to make use of AemDCPwr_AcquireArray.

The actualCount returned depends on the NPLC setting. If NPLC is 1, then, actualCount should be 20000. Memory must be allocated by user for voltageMeasurements and currentMeasurements.

Note that you must not change NPLC after calling AemDCPwr_ConfigureMeasureMode (mode=3) until AemDCPwr_AcquireVoltageArray is called.

This function returns zero if successful and non-zero if otherwise.

AemDCPwr_ContactCheck

Synopsis

ViStatus AemDCPwr_ContactCheck (vi, channelName, shi_h_slo_l, threshold_mohm, delay_s, pass_h_fail_l, mohm)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViConstString channelName (in)

Specifies the channel to be acted upon. Only one channel is allowed at one time.

ViInt32 shi_h_slo_l (in)

Specifies whether to perform contact check on Force-High/Sense-High pair or Force-Low/Sense-Low pair.
Set to "1" for Force-High/Sense-High pair.
Set to "0" for Force-Low/Sense-Low pair.

ViInt32 threshold_mohm (in)

Specifies the threshold to determine pass or fail, in ohms.

ViReal64 delay_s (in)

Specifies measurement delay (the delay after drive current and before read voltage).

ViInt32* pass_h_fail_l (out)

Returns test result based on threshold_mohm. "1" means pass, "0" means fail.

ViInt32* mohm (out)

Returns the measured resistance, in ohms.

Descriptions

AemDCPwr_ContactCheck performs Kelvin contact check on either Force-High/Sense-High or Force-Low/Sense-Low pair.

This function returns zero if successful and non-zero if otherwise.

AemDCPwr_ConfigureAcquireRecordLength

Synopsis

ViStatus AemDCPwr_ConfigureAcquireRecordLength (vi, channelName, length)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViConstString channelName (in)

Specifies the channel or channels to be acted upon. For example, "0" specifies channel 0 only. This can also be a channel list or a channel range.

A channel list is a comma (,) separated sequence of channel names (for example, "0,2" specifies channels 0 and 2).

A channel range is a lower bound channel followed by a hyphen (-) or colon (:) followed by an upper bound channel (for example, "0-2" specifies channels 0, 1, and 2).

ViInt32 length (in)

Specifies the number of samples to be acquired by AemDCPwr_AcquireMultiple.

Descriptions

AemDCPwr_ConfigureAcquireRecordLength configures the number of samples to be acquired by AemDCPwr_AcquireMultiple.

This function returns zero if successful and non-zero if otherwise.

AemDCPwr_ClearAcquireRecordLength

Synopsis

ViStatus AemDCPwr_ClearAcquireRecordLength (vi, channelName)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViConstString channelName (in)

Specifies the channel or channels to be acted upon. For example, "0" specifies channel 0 only. This can also be a channel list or a channel range.

A channel list is a comma (,) separated sequence of channel names (for example, "0,2" specifies channels 0 and 2).

A channel range is a lower bound channel followed by a hyphen (-) or colon (:) followed by an upper bound channel (for example, "0-2" specifies channels 0, 1, and 2).

Descriptions

AemDCPwr_ClearAcquireRecordLength resets the acquired record length to zero. After calling this function, if you call AemDCPwr_QueryAcquireRecordLength, you will read length output zero.

This function returns zero if successful and non-zero if otherwise.

AemDCPwr_MeasurePulseVI

Synopsis

ViStatus AemDCPwr_MeasurePulseVI (vi, channelName, cycles, vranges, iranges, voltages, currents)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViConstString channelName (in)

Specifies the channel or channels to be acted upon. For example, "0" specifies channel 0 only. This can also be a channel list or a channel range.

A channel list is a comma (,) separated sequence of channel names (for example, "0,2" specifies channels 0 and 2).

A channel range is a lower bound channel followed by a hyphen (-) or colon (:) followed by an upper bound channel (for example, "0-2" specifies channels 0, 1, and 2).

ViInt32* cycles (out)

Returns the number of targeted cycle when sourcing pulse.

ViReal64* vranges (out)

Returns the voltage range when DVCI.

ViReal64* iranges (out)

Returns the current range when DICV.

ViReal64* voltages (out)

Return an array of the voltage measurements during AemDcPwr_ConfigurePulse.

ViReal64* currents (out)

Return an array of the current measurements during AemDcPwr_ConfigurePulse.

Descriptions

AemDCPwr_MeasurePulseVI is to be used together with AemDCPwr_ConfigurePulse. Everytime AemDCPwr_ConfigurePulse is called, module will store the measurement value on chip ram and these values can only be retrieved via AemDCPwr_MeasurePulseVI. The number of measurement is same as the number of cycles.

This function returns zero if successful and non-zero if otherwise.

AemDCPwr_ConfigureAcquireArrayInterval

Synopsis

ViStatus AemDCPwr_ConfigureAcquireArrayInterval (vi, channelName, interval_s)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViConstString channelName (in)

Specifies the channel or channels to be acted upon. For example, "0" specifies channel 0 only. This can also be a channel list or a channel range.

A channel list is a comma (,) separated sequence of channel names (for example, "0,2" specifies channels 0 and 2).

A channel range is a lower bound channel followed by a hyphen (-) or colon (:) followed by an upper bound channel (for example, "0-2" specifies channels 0, 1, and 2).

ViReal64 interval_s (in)

By default, upon power-up interval_s is 1us. Resolution of interval_s is 1us. Unit of interval_s is in seconds and therefore automatically rounded to the closest us. Minimum interval_s is 1us and maximum interval_s is 1ms.

Descriptions

AemDCPwr_ConfigureAcquireArrayInterval is used together with AemDcPwr_AcquireCurrentArray or AemDcPwr_AcquireVoltageArray. AemDcPwr_ConfigureAcquireArrayInterval configures the collected sample to sample interval delay.

This function returns zero if successful and non-zero if otherwise.

AemDCPwr_QueryInCompliance

Synopsis

ViStatus AemDCPwr_QueryInCompliance (vi, channelName, inCompliance)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViConstString channelName (in)

Specifies the channel or channels to be acted upon. For example, "0" specifies channel 0 only. This can also be a channel list or a channel range.

A channel list is a comma (,) separated sequence of channel names (for example, "0,2" specifies channels 0 and 2).

A channel range is a lower bound channel followed by a hyphen (-) or colon (:) followed by an upper bound channel (for example, "0-2" specifies channels 0, 1, and 2).

ViBoolean* inCompliance (out)

Returns the compliance state of the specified channels.

Descriptions

AemDCPwr_QueryInCompliance queries the specified output instrument to determine if it is operating at the compliance limit.

You need to allocate sufficient memory space (in array form) if you query the compliance states for multiple channels.

This function returns zero if successful and non-zero if otherwise.

AemDCPwr_QueryOutputState

Synopsis

ViStatus AemDCPwr_QueryOutputState (vi, channelName, outputState, inState)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViConstString channelName (in)

Specifies the channel or channels to be acted upon. For example, "0" specifies channel 0 only. This can also be a channel list or a channel range.

A channel list is a comma (,) separated sequence of channel names (for example, "0,2" specifies channels 0 and 2).

A channel range is a lower bound channel followed by a hyphen (-) or colon (:) followed by an upper bound channel (for example, "0-2" specifies channels 0, 1, and 2).

ViInt32 outputState (in)

Specifies the state of the specified channels:

outputState	Description
0	The channel maintains a constant voltage by adjusting the current (DVCI)
1	The channel maintains a constant current by adjusting the voltage (DICV)

Table 19: Output State

ViBoolean* inState (out)

Returns whether the device output channel is in the specified state.

Descriptions

AemDCPwr_QueryInCompliance queries the specified output channel to determine if the output channel is currently in the state specified by outputState.

You need to allocate sufficient memory space (in array form) if you query the output states for multiple channels.

This function returns zero if successful and non-zero if otherwise.

AemDCPwr_QueryMaxVoltageLevel

Synopsis

ViStatus AemDCPwr_QueryMaxVoltageLevel (vi, channelName, currentLimit, maxVoltageLevel)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViConstString channelName (in)

Specifies the channel or channels to be acted upon. For example, "0" specifies channel 0 only. This can also be a channel list or a channel range.

A channel list is a comma (,) separated sequence of channel names (for example, "0,2" specifies channels 0 and 2).

A channel range is a lower bound channel followed by a hyphen (-) or colon (:) followed by an upper bound channel (for example, "0-2" specifies channels 0, 1, and 2).

ViReal64 currentLimit (in)

Specifies the current limit to use when calculating the maxVoltageLevel.

ViReal64* maxVoltageLevel (out)

Returns the maximum voltage level that can be achieved given the currentLimit.

Descriptions

AemDCPwr_QueryMaxVoltageLevel queries the maximum voltage level on an output channel if the output channel is set to the specified currentLimit.

You need to allocate sufficient memory space (in array form) if you query the maxVoltageLevel for multiple channels.

This function returns zero if successful and non-zero if otherwise.

AemDCPwr_QueryMaxCurrentLimit

Synopsis

ViStatus AemDCPwr_QueryMaxCurrentLimit (vi, channelName, voltageLimit, maxCurrentLimit)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViConstString channelName (in)

Specifies the channel or channels to be acted upon. For example, "0" specifies channel 0 only. This can also be a channel list or a channel range.

A channel list is a comma (,) separated sequence of channel names (for example, "0,2" specifies channels 0 and 2).

A channel range is a lower bound channel followed by a hyphen (-) or colon (:) followed by an upper bound channel (for example, "0-2" specifies channels 0, 1, and 2).

ViReal64 voltageLevel (in)

Specifies the voltage limit to use when calculating the maxCurrentLimit.

ViReal64* maxCurrentLimit (out)

Returns the maximum current level that can be achieved given the voltageLevel.

Descriptions

AemDCPwr_QueryMaxCurrentLimit queries the maximum current limit on an output channel if the output channel is set to the specified voltageLevel.

You need to allocate sufficient memory space (in array form) if you query the maxCurrentLimit for multiple channels.

This function returns zero if successful and non-zero if otherwise.

AemDCPwr_QueryMinCurrentLimit

Synopsis

ViStatus AemDCPwr_QueryMinCurrentLimit (vi, channelName, voltageLevel, minCurrentLimit)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViConstString channelName (in)

Specifies the channel or channels to be acted upon. For example, "0" specifies channel 0 only. This can also be a channel list or a channel range.

A channel list is a comma (,) separated sequence of channel names (for example, "0,2" specifies channels 0 and 2).

A channel range is a lower bound channel followed by a hyphen (-) or colon (:) followed by an upper bound channel (for example, "0-2" specifies channels 0, 1, and 2).

ViReal64 voltageLevel (in)

Specifies the voltage limit to use when calculating the minCurrentLimit.

ViReal64* minCurrentLimit (out)

Returns the minimum current level that can be achieved given the voltageLevel.

Descriptions

AemDCPwr_QueryMinCurrentLevel queries the minimum current limit on an output channel if the output channel is set to the specified voltageLevel.

You need to allocate sufficient memory space (in array form) if you query the minCurrentLimit for multiple channels.

This function returns zero if successful and non-zero if otherwise.

AemDCPwr_QueryAcquireRecordLength

Synopsis

ViStatus AemDCPwr_QueryAcquireRecordLength (vi, channelName, length)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViConstString channelName (in)

Specifies the channel or channels to be acted upon. For example, "0" specifies channel 0 only. This can also be a channel list or a channel range.

A channel list is a comma (,) separated sequence of channel names (for example, "0,2" specifies channels 0 and 2).

A channel range is a lower bound channel followed by a hyphen (-) or colon (:) followed by an upper bound channel (for example, "0-2" specifies channels 0, 1, and 2).

ViInt32* length (out)

Returns the acquired record length by AemDCPwr_AquireMultiple.

Descriptions

AemDCPwr_QueryAcquireRecordLength queries the acquired record length by AemDCPwr_AquireMultiple.

You need to allocate sufficient memory space (in array form) if you query the length for multiple channels.

This function returns zero if successful and non-zero if otherwise.

AemDCPwr_ReadVoltageLevelRange

Synopsis

ViStatus AemDCPwr_ReadVoltageLevelRange (vi, channelName, range)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViConstString channelName (in)

Specifies the channel or channels to be acted upon. For example, "0" specifies channel 0 only. This can also be a channel list or a channel range.

A channel list is a comma (,) separated sequence of channel names (for example, "0,2" specifies channels 0 and 2).

A channel range is a lower bound channel followed by a hyphen (-) or colon (:) followed by an upper bound channel (for example, "0-2" specifies channels 0, 1, and 2).

ViReal64* range (out)

Returns the voltage level range of the specified channel, in volts.

Descriptions

AemDCPwr_ReadVoltageLevelRange queries voltage level range of the specified channel.

You need to allocate sufficient memory space (in array form) if you query the range for multiple channels.

This function returns zero if successful and non-zero if otherwise.

AemDCPwr_ReadVoltageLimitRange

Synopsis

ViStatus AemDCPwr_ReadVoltageLimitRange (vi, channelName, range)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViConstString channelName (in)

Specifies the channel or channels to be acted upon. For example, "0" specifies channel 0 only. This can also be a channel list or a channel range.

A channel list is a comma (,) separated sequence of channel names (for example, "0,2" specifies channels 0 and 2).

A channel range is a lower bound channel followed by a hyphen (-) or colon (:) followed by an upper bound channel (for example, "0-2" specifies channels 0, 1, and 2).

ViReal64* range (out)

Returns the voltage limit range of the specified channel, in volts.

Descriptions

AemDCPwr_ReadVoltageLimitRange queries voltage limit range of the specified channel.

You need to allocate sufficient memory space (in array form) if you query the range for multiple channels.

This function returns zero if successful and non-zero if otherwise.

AemDCPwr_ReadCurrentLevelRange

Synopsis

ViStatus AemDCPwr_ReadCurrentLevelRange (vi, channelName, range)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViConstString channelName (in)

Specifies the channel or channels to be acted upon. For example, "0" specifies channel 0 only. This can also be a channel list or a channel range.

A channel list is a comma (,) separated sequence of channel names (for example, "0,2" specifies channels 0 and 2).

A channel range is a lower bound channel followed by a hyphen (-) or colon (:) followed by an upper bound channel (for example, "0-2" specifies channels 0, 1, and 2).

ViReal64* range (out)

Returns the current level range of the specified channel, in amps.

Descriptions

AemDCPwr_ReadCurrentLevelRange queries current level range of the specified channel.

You need to allocate sufficient memory space (in array form) if you query the range for multiple channels.

This function returns zero if successful and non-zero if otherwise.

AemDCPwr_ReadCurrentLimitRange

Synopsis

ViStatus AemDCPwr_ReadCurrentLimitRange (vi, channelName, range)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViConstString channelName (in)

Specifies the channel or channels to be acted upon. For example, "0" specifies channel 0 only. This can also be a channel list or a channel range.

A channel list is a comma (,) separated sequence of channel names (for example, "0,2" specifies channels 0 and 2).

A channel range is a lower bound channel followed by a hyphen (-) or colon (:) followed by an upper bound channel (for example, "0-2" specifies channels 0, 1, and 2).

ViReal64* range (out)

Returns the current limit range of the specified channel, in amps.

Descriptions

AemDCPwr_ReadCurrentLimitRange queries current limit range of the specified channel.

You need to allocate sufficient memory space (in array form) if you query the range for multiple channels.

This function returns zero if successful and non-zero if otherwise.

AemDCPwr_ConfigureTriggerEdgeLevel

Synopsis

ViStatus AemDCPwr_ConfigureTriggerEdgeLevel (vi, triggerEnum, option)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViInt32 triggerEnum (in)

Specifies the triggering source.

ViInt32 option (in)

Specifies the triggering mode.

Descriptions

AemDCPwr_ConfigureVoltageLevel configures how SMU channel(s) can be triggered.

triggerEnum	Description
0	Tri-state the trigger source
1	PXI backplane trigger line 0
2	PXI backplane trigger line 1
3	PXI backplane trigger line 2
4	PXI backplane trigger line 3
5	PXI backplane trigger line 4
6	PXI backplane trigger line 5
7	PXI backplane trigger line 6
8	PXI backplane trigger line 7
9	PXI backplane PXI_LBL6 signal
10	PXI backplane PXI_LBR6 signal
11	PXI backplane star trigger line
12	PXI backplane differential start trigger PXIE_DSTARA
13	PXI backplane differential start trigger PXIE_DSTARB
14	PXI backplane differential start trigger PXIE_DSTARC

19	Software trigger signal 0
20	Software trigger signal 1
21	Software trigger signal 2
22	Software trigger signal 3
23	External trigger port

Table 20: Trigger Source

option	Description
0	Channel will be triggered when rising edge is detected. This is the default mode
1	Channel will be triggered when falling edge is detected
2	Channel will be triggered when trigger signal is below a TTL logic level
3	Channel will be triggered when trigger signal exceeds logic level high

Table 21: Trigger Mode

Note that AemDCPwr_ConfigureVoltageLevel has no effects on software trigger lines.

This function returns zero if successful and non-zero if otherwise.

AemDCPwr_ConfigureTriggerEdgeLevelExtra

Synopsis

ViStatus AemDCPwr_ConfigureTriggerEdgeLevelExtra (vi, triggerEnum, option, ignore_trigger_count)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViInt32 triggerEnum (in)

Specifies the triggering source.

ViInt32 option (in)

Specifies the triggering mode.

ViInt32 ignore_trigger_count (in)

Specifies the number of triggers to be ignored, before performing measurement.

Descriptions

AemDCPwr_ConfigureVoltageLevelExtra configures how SMU channel(s) can be triggered.

triggerEnum	Description
0	Tri-state the trigger source
1	PXI backplane trigger line 0
2	PXI backplane trigger line 1
3	PXI backplane trigger line 2
4	PXI backplane trigger line 3
5	PXI backplane trigger line 4
6	PXI backplane trigger line 5
7	PXI backplane trigger line 6
8	PXI backplane trigger line 7
9	PXI backplane PXI_LBL6 signal
10	PXI backplane PXI_LBR6 signal
11	PXI backplane star trigger line

12	PXI backplane differential start trigger PXIE_DSTARA
13	PXI backplane differential start trigger PXIE_DSTARB
14	PXI backplane differential start trigger PXIE_DSTARC
19	Software trigger signal 0
20	Software trigger signal 1
21	Software trigger signal 2
22	Software trigger signal 3
23	External trigger port

Table 22: Trigger Source

option	Description
0	Channel will be triggered when rising edge is detected. This is the default mode
1	Channel will be triggered when falling edge is detected
2	Channel will be triggered when trigger signal is below a TTL logic level
3	Channel will be triggered when trigger signal exceeds logic level high

Table 23: Trigger Mode

Note that AemDCPwr_ConfigureVoltageLevelExtra has no effects on software trigger lines.

AemDCPwr_ConfigureVoltageLevelExtra allows users to specify the number of triggers to be ignored, before performing actual measurement. For example, if ignore_trigger_count = 0, first trigger is a valid trigger. If ignore_trigger_count = 2, module will ignore the first two triggers and only begin measuring after third trigger.

This function returns zero if successful and non-zero if otherwise.

AemDCPwr_MapTriggerInToTriggerOut

Synopsis

ViStatus AemDCPwr_MapTriggerInToTriggerOut (vi, inputTerminal, outputTerminal)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViInt32 inputTerminal (in)

Specifies the triggering line to be connected to outputTerminal.

ViInt32 outputTerminal (in)

Specifies the triggering line to be connected to inputTerminal.

Descriptions

AemDCPwr_MapTriggerInToTriggerOut connect 2 trigger lines together.

inputTerminal / outputTerminal	Description
0	Tri-state the trigger source
1	PXI backplane trigger line 0
2	PXI backplane trigger line 1
3	PXI backplane trigger line 2
4	PXI backplane trigger line 3
5	PXI backplane trigger line 4
6	PXI backplane trigger line 5
7	PXI backplane trigger line 6
8	PXI backplane trigger line 7
9	PXI backplane PXI_LBL6 signal
10	PXI backplane PXI_LBR6 signal
11	PXI backplane star trigger line
12	PXI backplane differential start trigger PXIE_DSTARA
13	PXI backplane differential start trigger PXIE_DSTARB
14	PXI backplane differential start trigger PXIE_DSTARC
19	Software trigger signal 0

20	Software trigger signal 1
21	Software trigger signal 2
22	Software trigger signal 3
23	External trigger port

Table 24: Available Trigger Lines

Please note that external trigger is a dual direction trigger. It follows the last state of the switch, therefore for driving an input to external trigger, it is advisable to map it to TRISTATE before driving the trigger to it.

This function returns zero if successful and non-zero if otherwise.

AemDCPwr_DriveSoftwareTrigger

Synopsis

ViStatus AemDCPwr_DriveSoftwareTrigger (vi, select, pulseWidth)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViInt32 select (in)

Specifies the software trigger line.

ViReal64 pulseWidth (in)

Specifies the pulse width of the trigger signal, in seconds.

Descriptions

AemDCPwr_DriveSoftwareTrigger drives software trigger immediately. Software trigger lines are digital signals coming from the on-board processor.

select	Description
0	Software trigger signal 0
1	Software trigger signal 1
2	Software trigger signal 2
3	Software trigger signal 3

Table 25: Available Software Trigger Lines

This function returns zero if successful and non-zero if otherwise.

AemDCPwr_ConfigureInputTriggerSelect

Synopsis

ViStatus AemDCPwr_ConfigureInputTriggerSelect (vi, channelName, triggerInput, delay_s)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViConstString channelName (in)

Specifies the channel or channels to be acted upon. For example, "0" specifies channel 0 only. This can also be a channel list or a channel range.

A channel list is a comma (,) separated sequence of channel names (for example, "0,2" specifies channels 0 and 2).

A channel range is a lower bound channel followed by a hyphen (-) or colon (:) followed by an upper bound channel (for example, "0-2" specifies channels 0, 1, and 2).

ViInt32 triggerInput (in)

Specifies the trigger source.

ViReal64 delay_s (in)

Specifies the delay inserted after being triggered before next operation such as AemDCPwr_AcquireMultiple, is executed.

Descriptions

AemDCPwr_ConfigureInputTriggerSelect selects the trigger source to be used for operations such as AemDCPwr_AcquireMultiple and AemDCPwr_AcquireArray. Upon power-up nothing is selected for triggerInput.

triggerInput	Description
0	Tri-state the trigger source (Default)
1	PXI backplane trigger line 0
2	PXI backplane trigger line 1
3	PXI backplane trigger line 2
4	PXI backplane trigger line 3

5	PXI backplane trigger line 4
6	PXI backplane trigger line 5
7	PXI backplane trigger line 6
8	PXI backplane trigger line 7
9	PXI backplane PXI_LBL6 signal
10	PXI backplane PXI_LBR6 signal
11	PXI backplane star trigger line
12	PXI backplane differential start trigger PXIE_DSTARA
13	PXI backplane differential start trigger PXIE_DSTARB
14	PXI backplane differential start trigger PXIE_DSTARC
19	Software trigger signal 0
20	Software trigger signal 1
21	Software trigger signal 2
22	Software trigger signal 3
23	External trigger port

Table 26: Available Trigger Source

The example below shows how to set up the external trigger as input to perform AemDCPwr_AcquireArray upon being triggered by a rising edge signal:

1. AemDCPwr_ConfigureInputTriggerSelect
    a. triggerInput = 23 (External trigger)
    b. delay_s = 20e-6 (20us delay inserted after being triggered, before starts measurement)
2. AemDCPwr_ConfigureTriggerEdgeLevel
    a. triggerEnum = 23 (External trigger)
    b. option = 0 (rising edge)
3. AemDCPwr_ConfigureMeasureMode
    a. mode = 3
4. AemDCPwr_AcquireArray

After operation completes, do the following:

1. AemDCPwr_ConfigureInputTriggerSelect
    a. triggerInput = 0 (tri-state the trigger)

NOTE

Please note that external trigger is a dual direction trigger. It follows the last state of the trigger (input or output). To configure the external trigger port to become an input, it is required to map it to TRISTATE via AemDCPwr_MapTriggerInToTriggerOut.

This function returns zero if successful and non-zero if otherwise.

AemDCPwr_ConfigureSMUOutputTriggerMode

Synopsis

ViStatus AemDCPwr_ConfigureSMUOutputTriggerMode (vi, channelName, mode)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViConstString channelName (in)

Specifies the channel or channels to be acted upon. For example, "0" specifies channel 0 only. This can also be a channel list or a channel range.

A channel list is a comma (,) separated sequence of channel names (for example, "0,2" specifies channels 0 and 2).

A channel range is a lower bound channel followed by a hyphen (-) or colon (:) followed by an upper bound channel (for example, "0-2" specifies channels 0, 1, and 2).

ViInt32 mode (in)

Specifies the trigger mode.

Descriptions

AemDCPwr_ConfigureSMUOutputTriggerMode configures when a trigger output signal is generated.

mode	Description
0	No trigger action
1	Generate trigger output when source action is completed
2	Generate trigger output when measure action is completed
3	Generate trigger output when compliance hit
4	Generate trigger output when compliance hit exits
5	Generate trigger output when during source action

Table 27: Trigger mode

When trigger output is no longer required, set mode back to 0.

This function returns zero if successful and non-zero if otherwise.

AemDCPwr_ConfigureSMUOutputTriggerPulseWidth

Synopsis

ViStatus AemDCPwr_ConfigureSMUOutputTriggerPulseWidth (vi, channelName, pulseWidth)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViConstString channelName (in)

Specifies the channel or channels to be acted upon. For example, "0" specifies channel 0 only. This can also be a channel list or a channel range.

A channel list is a comma (,) separated sequence of channel names (for example, "0,2" specifies channels 0 and 2).

A channel range is a lower bound channel followed by a hyphen (-) or colon (:) followed by an upper bound channel (for example, "0-2" specifies channels 0, 1, and 2).

ViReal64 pulseWidth (in)

Specifies the width of the trigger output pulse, in seconds.

Descriptions

AemDCPwr_ConfigureSMUOutputTriggerPulseWidth configures the pulse width of trigger output signal.

This function returns zero if successful and non-zero if otherwise.

AemDCPwr_ConfigureSMUOutputTriggerDuringSource

Synopsis

ViStatus AemDCPwr_ConfigureSMUOutputTriggerDuringSource (vi, channelName, level, range, mode, edgeSetting)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViConstString channelName (in)

Specifies the channel or channels to be acted upon. For example, "0" specifies channel 0 only. This can also be a channel list or a channel range.

A channel list is a comma (,) separated sequence of channel names (for example, "0,2" specifies channels 0 and 2).

A channel range is a lower bound channel followed by a hyphen (-) or colon (:) followed by an upper bound channel (for example, "0-2" specifies channels 0, 1, and 2).

ViReal64 level (in)

Specifies the threshold level of the channel where a trigger output is generated when this threshold level is hit, depending on edgeSetting.

ViReal64 range (in)

Specifies the range of the threshold level.

ViInt32 mode (in)

Specifies whether the threshold is voltage or current value.

Set to "0" for voltage.

Set to "1" for current.

ViInt32 edgeSetting (in)

Specifies the condition on how to trigger output will be generated by the specified channel.

For multi-channel operation, a trigger signal will be generated if any one of channels meets the trigger condition.

Descriptions

AemDCPwr_ConfigureSMUOutputTriggerDuringSource configures when trigger signal is generated when mode=5 is selected for AemDCPwr_ConfigureSMUOutputTriggerMode.

edgeSetting	Description
0	A trigger signal will be generated when existing level is below the threshold level
1	A trigger signal will be generated when existing level is beyond the threshold level
2	A trigger signal will be generated when falling edge is detected via the threshold level
3	A trigger signal will be generated when rising edge is detected via the threshold level

Table 28: Available edgeSetting

This function returns zero if successful and non-zero if otherwise.

AemDCPwr_ConfigureOutputTriggerSelect

Synopsis

ViStatus AemDCPwr_ConfigureOutputTriggerSelect (vi, channelName, triggerOutput, delay_s)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViConstString channelName (in)

Specifies the channel or channels to be acted upon. For example, "0" specifies channel 0 only. This can also be a channel list or a channel range.

A channel list is a comma (,) separated sequence of channel names (for example, "0,2" specifies channels 0 and 2).

A channel range is a lower bound channel followed by a hyphen (-) or colon (:) followed by an upper bound channel (for example, "0-2" specifies channels 0, 1, and 2).

ViInt32 triggerOutput (in)

Specifies the trigger output signal.

ViReal64 delay_s (in)

Specifies the delay inserted before generating the trigger output, after operation such as AemDCPwr_ConfigureSMUOutputTriggerMode, is executed.

Descriptions

AemDCPwr_ConfigureOutputTriggerSelect selects the trigger output to be used for operations such as AemDCPwr_ConfigureSMUOutputTriggerMode.

triggerOutput	Description
0	Tri-state the trigger source
1	PXI backplane trigger line 0
2	PXI backplane trigger line 1
3	PXI backplane trigger line 2
4	PXI backplane trigger line 3
5	PXI backplane trigger line 4

6	PXI backplane trigger line 5
7	PXI backplane trigger line 6
8	PXI backplane trigger line 7
9	PXI backplane PXI_LBL6 signal
10	PXI backplane PXI_LBR6 signal
11	PXI backplane star trigger line
12	PXI backplane differential start trigger PXIE_DSTARA
13	PXI backplane differential start trigger PXIE_DSTARB
14	PXI backplane differential start trigger PXIE_DSTARC
19	Software trigger signal 0
20	Software trigger signal 1
21	Software trigger signal 2
22	Software trigger signal 3
23	External trigger port

Table 29: Available Trigger Output

The example below shows how to set up a channel to generate a 10ms pulse at external trigger output, whenever the voltage level hits 5V when the channel drives from 0V to 10V:

1. AemDCPwr_ConfigureOutputTriggerSelect
    a. triggerInput = 23 (External trigger)
    b. delay_s = 0 (No delay inserted after being triggered, before starts generating the trigger output pulse)
2. AemDCPwr_ConfigureSMUOutputTriggerMode
    a. mode = 5 (Generate trigger output when during source action)
3. AemDCPwr_ConfigureSMUOutputTriggerDuringSource
    a. Level = 5
    b. Range = 10
    c. mode = 0 (Compare voltage)
    d. edgeSetting = 3 (A trigger signal will be generated when rising edge is detected via the threshold level)
4. AemDCPwr_ConfigureSMUOutputTriggerPulseWidth
    a. pulseWidth = 10e-3 (10ms pulse width)
5. AemDCPwr_ConfigureOutputFunction
    a. Function = 0 (DVCI)
6. AemDCPwr_ConfigureVoltageLevel
    a. Level = 10

After operation completes, do the following:

1. AemDCPwr_ConfigureSMUOutputTriggerMode
    a. mode = 0 (No trigger action)

This function returns zero if successful and non-zero if otherwise.

Section 6: Revision History

1.0	DEC 2012	INITIAL RELEASE
1.1	MAR 2013	ADDED AEMDCPWR_CONFIGURETRIGGEREDGELEVELEXTRA
1.2	JUNE 2013	EDITED AEMDCPWR_QUERYMAXCURRENTLIMIT & AEMDCPWR_QUERYMINCURRENTLIMIT
1.3	JULY 2013	EDITED AEMDCPWR_CONFIGURESMUOUTPUTTRIGGERDURINGSOURCE REVISED AEMDCPWR_DRIVESOFTWARETRIGGER ADDED AEMDCPWR_CONFIGUREPULSE ADDED AEMDCPWR_CONFIGUREEXTENDERMODULE ADDED AEMDCPWR_STARTCONTINOUSPULSE ADDED AEMDCPWR_STOPCONTINOUSPULSE ADDED AEMDCPWR_MEASUREPULSEVI
1.4	OCT 2013	ADDED AEMDCPWR_CONFIGUREOUTPUTENABLED
1.5	DEC 2013	ADDED AEMDCPWR_CONFIGUREACQUIREARRAYINTERVAL ADDED DESCRIPTION FOR EXTERNAL TRIGGER
1.6	FEB 2014	ADDED AEMDCPWR_CONFIGUREACQUIREARRAYBWLIMIT

Section 7: Contact Us

To obtain service, warranty or technical assistance, please contact Aemulus.

Aemulus Corporation Sdn. Bhd.
Krystal Point, B-2-04, B-2-05, B-2-06 & B-2-07
303, Jalan Sultan Azlan Shah,
11900 Penang, Malaysia
Tel: +604 6446399
Fax: +604 6466799

Web: www.aemulus.com
Email: enquiry@aemulus.com

Product specifications and descriptions in this document are subject to change without prior notice.