Aemulus

DM483e

Programming Manual

Version 1.0, 08-2014

Table of Contents

DM483e_DPINOpen

Synopsis

ViStatus DM483e_DPINOpen (resourceName, dpingroup_sel, 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.

ViInt32 dpingroup_sel (in)

Specifies the pin group to be turned on or activated.
1 = DPin group 0 (PIN0 to PIN5) and trigout0
2 = DPin group 1 (PIN6 to PIN11) and trigout1
3 = DPin group 0 and 1(PIN0 to PIN11), trigout0 and trigout1

ViInt32 init_options (in)

To open a session and leave the device in its existing configuration, set this argument to 0.

To place the device in a known start-up state when creating a new session set the appropriate bit value:

bit[0] = Specifies whether to reset the device during the initialization procedure (enabled) or maintain current configuration in the hardware (disabled).
bit[1] = Specifies whether to reset the cmd fifo during the initialization procedure.
bit[2] = Specifies whether to reset the result fifo during the initialization procedure.
bit[3] = Specifies whether to reset the lock status to unlock during the initialization procedure.
bit[5] = To bypass calibration data loading process.

ViConstString optionString (in)

You can optionally set the initial state of the following session attributes, separated by semicolons. Otherwise pass in an empty string ("").

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

Example: "Simulate = 0, DriverSetup=Model:DM428e"

ViSession* vi (out)

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

Descriptions

DPINOpen 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 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.

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

[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.

DM483e_DPINClose

Synopsis

ViStatus DM483e_DPINClose (vi)

Arguments

ViSession vi (in)

Specifies the instrument handle.

Descriptions

DM483e_DPINClose closes the session specified in instrument handle. The instrument will maintain its last running state.

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

DM483e_DPINForce

Synopsis

ViStatus DM483e_DPINForce (vi, pinNo, state)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViInt32 pinNo (in)

Specify the pin number.
If dpingroup_sel = 1 during DM483e_DPINOpen, pinNo=0-5 (PIN0 to PIN5).
If dpingroup_sel = 2 during DM483e_DPINOpen, pinNo=6-11 (PIN6 to PIN11).
If dpingroup_sel = 3 during DM483e_DPINOpen, pinNo=0-11 (PIN0 to PIN11).

ViInt32 state (in)

Specifies the operation mode:

State	Description
0	CONST_FORCE_STATE_VECTOR
1	CONST_FORCE_STATE_PMU
2	CONST_FORCE_STATE_DIO
5	CONST_FORCE_STATE_CLOCK
6	CONST_FORCE_STATE_INVERTED_CLOCK

Table 1: Mode of Operation

Descriptions

DM483e_DPINForce specifies the operation mode. Definition of each mode is explained below.

Mode	Description
CONST_FORCE_STATE_VECTOR	Run a series of digital patterns
CONST_FORCE_STATE_PMU	Perform parametric measurement unit functions

CONST_FORCE_STATE_DIO	Perform general digital I/O functions such as Drive Pin, Driver Port, Read Pin and Read Port.
CONST_FORCE_STATE_CLOCK	Configure the selected pin to drive continuous clock signal
CONST_FORCE_STATE_INVERTED_CLOCK	Configure the selected pin to drive continuous clock signal, whose state is inversed of the pin configured in CONST_FORCE_STATE_CLOCK mode.

Table 2: Description of Mode

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

DM483e_ReadRevision

Synopsis

ViStatus DM483e_ReadRevision (vi, instrumentDriverRevision, firmwareRevision)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViChar* instrumentDriverRevision (out)

Returns the driver revision information of DM483e.dll. Space allocated by the caller, must be at least 32 bytes.

ViChar* firmwareRevision (out)

Space allocated by the caller, must be at least 32 bytes, returns firmware revision information for the device you are using. This argument returns both firmware version of master and daughter card separated by a hyphen.

Descriptions

DM483e_ReadRevision returns the driver and firmware revision of the DM483e.

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

DM483e_ReadChannelTemperature

Synopsis

ViStatus DM483e_ReadChannelTemperature (vi, pinNo, temperature)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViInt32 pinNo (in)

Specify the pin number.
If dpingroup_sel = 1 during DM483e_DPINOpen, pinNo=0-5.
If dpingroup_sel = 2 during DM483e_DPINOpen, pinNo=6-11.
If dpingroup_sel = 3 during DM483e_DPINOpen, pinNo=0-11.

ViReal64* temperature (out)

Read the temperature of the on-board Pin Electronics IC. Temperature is in degree Celsius.

Descriptions

DM483e_ReadChannelTemperature reads the channel temperature of the on-board Pin Electronics IC.

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

DM483e_ReadAmbientTemperature

Synopsis

ViStatus DM483e_ReadAmbientTemperature (vi, temperature)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViReal64* temperature (out)

Return the current temperature on the board, in degrees Celsius. This temperature sensor is placed on bottom plane of master card.

Descriptions

DM483e_ReadAmbientTemperature returns the current temperature of the board. Refer to DM483e_ReadChannelTemperature for more precise temperature of each pin.

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

DM483e_ReadSerialNumber

Synopsis

ViStatus DM483e_ReadSerialNumber (vi, sn)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViChar* sn (out)

Serial number length should at least 64 bytes. It returns both master and daughter card serial numbers separated by a hyphen.

Descriptions

DM483e_ReadSerialNumber returns the serial number of the device. Space allocated by the caller, and must be at least 64 characters.

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

DM483e_Reset

Synopsis

ViStatus DM483e_Reset (vi)

Arguments

ViSession vi (in)

Specifies the instrument handle.

Descriptions

DM483e_Reset resets all session attributes to default value on hardware and turn off all pins.

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

DM483e_ResetGroup

Synopsis

ViStatus DM483e_ResetGroup (vi, group)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViInt32 group (in)

Specifies the pin group to be reset.

dpingroup_sel	group	Description
1	0	Pin 0 to Pin 5
2	1	Pin 6 to Pin 11

Table 3: Group definition for Reset

Descriptions

DM483e_ResetGroup resets only selected pin group to default value on hardware and turn off all pins in the group.

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

DM483e_ConfigureMultiSiteMode

Synopsis

ViStatus DM483e_ConfigureMultiSiteMode (vi, mode)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViInt32 mode (in)

Mode = 0 for single site operation. Lock/unlock operation is not performed. This is the default mode.

Mode = 1 for multi-site operation. Lock/unlock operation is performed.

Descriptions

DM483e_ConfigureMultiSiteMode configures DM483e to operate in multi-site or single-site mode.

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.

Vint32* slotAddress (out)

Return the slot address of the DM483e.

Descriptions

ReadSlotAddress returns the slot number of the module where it is slotted.

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

DM483e_DrivePort

Synopsis

ViStatus DM483e_DrivePort (vi, value)

Arguments

ViSession vi (in)

Specifies the instrument handle.

Vint32 value (in)

Specifies the port value to be driven.
If dpingroup_sel = 1 during DM483e_DPINOpen, port covers PIN0-5 only. LSB refers to pin 0.
If dpingroup_sel = 1 during DM483e_DPINOpen, port covers PIN6-11 only. LSB refers to pin 6.
If dpingroup_sel = 3 during DM483e_DPINOpen, port covers PIN0-11. LSB refers to pin 0.

Descriptions

DM483e_DrivePort drives port value to the specified pin group.

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

DM483e_DrivePin

Synopsis

ViStatus DM483e_DrivePin (vi, pinNo, value)

Arguments

ViSession vi (in)

Specifies the instrument handle.

Vint32 pinNo (in)

If dpingroup_sel = 1 during DM483e_DPINOpen, pinNo refers to pin 0 to pin 5.
If dpingroup_sel = 2 during DM483e_DPINOpen, pinNo refers to pin 6 to pin 11.
If dpingroup_sel = 3 during DM483e_DPINOpen, pinNo refers to pin 0 to pin 11.

Vint32 value (in)

Specifies the drive value of the selected pin.

Descriptions

DM483e_DrivePin drives logic 0 or 1 to the specified pin number.

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

DM483e_SetPortDirection

Synopsis

ViStatus DM483e_SetPortDirection (vi, value)

Arguments

ViSession vi (in)

Specifies the instrument handle.

Vint32 value (in)

Specifies the direction of the targeted port where each bit value represents one port.
If dpingroup_sel = 1 during DM483e_DPINOpen, port covers pinNo 0-5 only. LSB refers to pin 0 value.
If dpingroup_sel = 2 during DM483e_DPINOpen, port covers pinNo 6-11 only. LSB refers to pin 6 value.
If dpingroup_sel = 3 during DM483e_DPINOpen, port covers pinNo 0-11. LSB refers to pin 0 value.
Bit 0 means input, bit 1 means output.

Descriptions

DM483e_SetPortDirection sets the direction of the selected pin group.

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

DM483e_SetPinDirection

Synopsis

ViStatus DM483e_SetPinDirection (vi, pinNo, value)

Arguments

ViSession vi (in)

Specifies the instrument handle.

Vint32 pinNo (in)

Specifies the pin to be acted upon.
If dpingroup_sel = 1 during DM483e_DPINOpen, pinNo is 0-5 only.
If dpingroup_sel = 2 during DM483e_DPINOpen, pinNo is 6-11 only.
If dpingroup_sel = 3 during DM483e_DPINOpen, pinNo is 0-11.

Vint32 value (in)

Specifies the direction of the targeted pin.

Descriptions

DM483e_SetPinDirection sets the direction of the selected pin.

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

DM483e_ReadPort

Synopsis

ViStatus DM483e_ReadPort (vi, value)

Arguments

ViSession vi (in)

Specifies the instrument handle.

Vint32* value (out)

Returns the read back port value.

If dpingroup_sel = 1 during DM483e_DPINOpen, port covers pinNo is 0-5 only. LSB refers to pin 0 value.

If dpingroup_sel = 2 during DM483e_DPINOpen, port covers pinNo is 6-11 only. LSB refers to pin 6 value.

If dpingroup_sel = 3 during DM483e_DPINOpen, port covers pinNo is 0-11. LSB refers to pin 0 value.

Descriptions

DM483e_ReadPort reads back the value present at the specified port. 2 binary bits are returned for each pin, which represents:

Binary Bits	Description
00	Logic low (L)
01	Logic high (H)
10	Undefined state (Z)

Table 4: Binary Representation of Logic State

For example, if dpingroup_sel = 3, and the returned value is 0x5505, the binary representation is "10 10 01 01 01 01 00 00 01 01". Every 2 binary bits represent the logic state of a pin. In this example, the state of each pin is "LLZZHHHHLLHH".

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

DM483e_ReadPin

Synopsis

ViStatus DM483e_ReadPin (vi, pinNo, value)

Arguments

ViSession vi (in)

Specifies the instrument handle.

Vint32 pinNo (in)

Specifies the pin to be acted upon.
If dpingroup_sel = 1 during DM483e_DPINOpen, pinNo is 0-5 only.
If dpingroup_sel = 2 during DM483e_DPINOpen, pinNo is 6-11 only.
If dpingroup_sel = 3 during DM483e_DPINOpen, pinNo is 0-11.

Vint32* value (out)

Returns the read back pin value.

Descriptions

DM483e_ReadPin reads back the value present at the specified pin. 2 binary bits are returned, which represents:

Binary Bits	Description
00	Logic low (L)
01	Logic high (H)
10	Undefined state (Z)

Table 5: Binary Representation of Logic State

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

DM483e_DPINVectorResourceAllocation

Synopsis

ViStatus DM483e_DPINVectorResourceAllocation (vi, vecSetCount, resourceArray)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViInt32 vecSetCount (in)

The amount of vector set to be used.

ViInt32 * resourceArray (in)

Specifies the required memory depth of each vector set. This has to be multiple of 512k. For example, if resourceArray[0]=6, resourceArray[1]=2, resourceArray[2]=3, it means that vector set 0 requires 6*512k of memory, vector set 1 requires 2*512k of memory, and vector set 2 requires 3*512k of memory.

By default, vecSetCount is 32 and resourceArray is 1 for each vector set.

Descriptions

DM483e_DPINVectorResourceAllocation specifies the resource required to be allocated in the hardware. This function has to be called before vectors are loaded. DM483e_Reset will not reset vecSetCount and resourceArray on the hardware.

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

DM483e_DPINLevel

Synopsis

ViStatus DM483e_DPINLevel (vi, pinNo, VIH, VIL, VOH, VOL, IOH, IOL, VCH, VCL, VTERM)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViInt32 pinNo (in)

If dpingroup_sel = 1 during DM483e_DPINOpen, pinNo refers to pin 0 to pin 5.
If dpingroup_sel = 2 during DM483e_DPINOpen, pinNo refers to pin 6 to pin 11.
If dpingroup_sel = 3 during DM483e_DPINOpen, pinNo refers to pin 0 to pin 11.

ViReal64 VIH (in)

Specifies the output driver high voltage level from the pin.

ViReal64 VIL (in)

Specifies the output driver low voltage level from the pin.

ViReal64 VOH (in)

Specifies the input comparator high voltage level into the pin.

ViReal64 VOL (in)

Specifies the input comparator low voltage level into the pin.

ViReal64 IOH (in)

Specifies the sink current when active load is enabled.

ViReal64 IOL (in)

Specifies the source current when active load is enabled.

ViReal64 VCH (in)

Specifies the reflection voltage clamp level for high range.

ViReal64 VCL (in)

Specifies the reflection voltage clamp level for low range.

ViReal64 VTERM (in)

Specifies the termination voltage when input termination is enabled. If input termination is disabled, driver is in high impedance.

Descriptions

DM483e_DPINLevel programs the level setting for the specified pin.

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

DM483e_DPINVecLoad

Synopsis

ViStatus DM483e_DPINVecLoad (vi, option, vecSetNo, vecFileName)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViInt32 option (in)

Put 'o'.

This argument is provided to support test program compatibility when you port from DM482e (Option 1 for dedicated IO) to DM483e.

ViInt32 vecSetNo (in)

Specifies the vector set to be loaded.

char* vecFileName (in)

Specifies file name of the vector file to be loaded.

Descriptions

DM483e_DPINVecLoad loads the vectors to the device specified in the vector file.

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

DM483e_DPINVecLoadArray

ViStatus DM483e_DPINVecLoadArray (vi, option, vecSetNo, timingSet, channelState, array_dw_count, groupo_vectors, group1_vectors)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViInt32 option (in)

Put '0'.

This argument is provided to support test program compatibility when you port from DM482e (Option 1 for dedicated IO) to DM483e.

ViInt32 vecSetNo (in)

Specifies the vector set to be loaded.

ViInt32 timingSet (in)

Specifies the timing set to be used.

ViInt32 channelState (in)

Put '0'.

ViInt32 array_dw_count (in)

Specifies the length of groupo_vectors and/or group1_vectors.

If dpingroup_sel = 1, groupo_vectors refer to vectors for PIN0 to PIN5, group1_vectors is ignored.

If dpingroup_sel = 2, groupo_vectors refer to vectors for PIN6 to PIN11, group1_vectors is ignored.

If dpingroup_sel = 3, groupo_vectors refer to vectors for physical PIN0 to PIN5, and group1_vectors refer to vectors for PIN6 to PIN11.

ViInt32* groupo_vectors (in)

Specifies vectors or patterns for groupo_vectors.

ViInt32* group1_vectors (in)

Specifies vectors or patterns for group1_vectors.

Descriptions

DM483e_ DPINVecLoadArray loads vectors in array form.

Supported vector formats are:

0 : Logic low
1 : Logic high
2 : Compare low
3 : Compare high
4 : Tristate and Don't-compare

The pin format of group0_vectors and group1_vectors [18:0] are:

[18:16] channel5_format
[15:13] channel4_format
[12:10] channel3_format
[9:7] channel2_format
[6:4] channel1_format
[3:1] channel0_format
[0] trigger-out

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

DM483e_DPINPeriod

Synopsis

ViStatus DM483e_DPINPeriod (vi, timingSetNo, period_s)

Arguments

ViSession vi (in)
: Specifies the instrument handle.

ViInt32 timingSetNo
: Specifies the timing set to be used.

ViReal64 period_s (in)
: Specifies the period of each bit of data as pointed by the specified timing set.

Descriptions

DM483e_DPINPeriod specifies the timing set to be used.

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

DM483e_DPINOn

Synopsis

ViStatus DM483e_DPINOn (vi, pinNo)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViInt32 pinNo (in)

Specifies the pin to be acted upon.
If dpingroup_sel = 1 during DM483e_DPINOpen, pinNo is 0-5 only.
If dpingroup_sel = 2 during DM483e_DPINOpen, pinNo is 6-11 only.
If dpingroup_sel = 3 during DM483e_DPINOpen, pinNo is 0-11.

Descriptions

DM483e_DPINOn enables the specified pin. This function needs to be called before mode of operation (DM483e_DPINForce) can be effective.

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

DM483e_DPINOff

Synopsis

ViStatus DM483e_DPINOff (vi, pinNo)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViInt32 pinNo (in)

Specifies the pin to be acted upon.
If dpingroup_sel = 1 during DM483e_DPINOpen, pinNo is 0-5 only.
If dpingroup_sel = 2 during DM483e_DPINOpen, pinNo is 6-11 only.
If dpingroup_sel = 3 during DM483e_DPINOpen, pinNo is 0-11.

Descriptions

DM483e_DPINOff disables the specified pin.

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

DM483e_DPINHVOn

Synopsis

ViStatus DM483e_DPINHVOn (vi, pinNo)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViInt32 pinNo (in)

Specifies the pin to be acted upon.
If dpingroup_sel = 1 during DM483e_DPINOpen, pinNo is 0-5 only.
If dpingroup_sel = 2 during DM483e_DPINOpen, pinNo is 6-11 only.
If dpingroup_sel = 3 during DM483e_DPINOpen, pinNo is 0-11.

Descriptions

DM483e_DPINHVOn enables the high voltage (HV) output relay of the specified pin. Only pins 0, 2, 4, 6, 8, and 10 contain high voltage capabilities.

Mapping between high voltage drivers and pinNo is tabulated below. Refer to user manual for pin configuration of high voltage driver at the output connector.

pinNo	HV Driver
0	HV0
2	HV1
4	HV2
6	HV3
8	HV4
10	HV5

Table 6: HV Driver mapping to pinNo

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

DM483e_DPINHVOff

Synopsis

ViStatus DM483e_DPINHVOff (vi, pinNo)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViInt32 pinNo (in)

Specifies the pin to be acted upon.
If dpingroup_sel = 1 during DM483e_DPINOpen, pinNo is 0-5 only.
If dpingroup_sel = 2 during DM483e_DPINOpen, pinNo is 6-11 only.
If dpingroup_sel = 3 during DM483e_DPINOpen, pinNo is 0-11.

Descriptions

DM483e_DPINHVOff disables the high voltage (HV) output relay of the specified pin. Only pins 0, 2, 4, 6, 8, and 10 contain high voltage capabilities.

Mapping between high voltage drivers and pinNo is tabulated below. Refer to user manual for pin configuration of high voltage driver at the output connector.

pinNo	HV Driver
0	HV0
2	HV1
4	HV2
6	HV3
8	HV4
10	HV5

Table 7: HV Driver mapping to pinNo

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

DM483e_RunVector

Synopsis

ViStatus DM483e_RunVector (vi, vecSetNo)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViInt32 vecSetNo (in)

Specifies the vector set to be executed.

Descriptions

DM483e_RunVector execute the specified vector set.

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

DM483e_StopVector

Synopsis

ViStatus DM483e_StopVector (vi)

Arguments

ViSession vi (in)

Specifies the instrument handle.

Descriptions

DM483e_StopVector immediately stops the vector execution of the dpingroup_sel as defined in DM483e_DPINOpen, if vectors are running. When this happens, the vector output will remain at the last state.

If the vector engine is armed but has not been triggered yet by the input trigger set via DM483e_ConfigureInputTrigger, calling this function will disarm the vector engine.

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

DM483e_AcquireVecEngineStatus

Synopsis

ViStatus DM483e_AcquireVecEngineStatus (vi, status)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViInt32* status (out)

Return the status of the vector execution.

0: Completed.

1: Busy.

Descriptions

DM483e_AcquireVecEngineStatus return the status of the vector execution.

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

DM483e_ReadHistoryRam

Synopsis

ViStatus DM483e_ReadHistoryRam (vi, vectorCount, startVectorLocation, vecSetNo, history_ram_data)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViInt32 vectorCount (in)

Specifies the number of vectors to be returned from the device.

ViInt32 startVectorLocation (in)

Specifies the starting location of the vectors to be returned from the device.

ViInt32 ViInt32 vecSetNo (in)

Specifies the vector set of interest.

ViUInt32* history_ram_data(out)

Returns an array of measurements. User has to allocate enough memory for history_ram_data.

Descriptions

DM483e_ReadHistoryRam reads back every stored bit in the device. One memory space of history_ram_data contains information of one vector for all the pins. The format of the returned history_ram_data is as follows:

Binary Bits	Description
00	Logic low (L)
01	Logic high (H)
10	Undefined state (Z)

Table 8: Binary Representation of Logic State

For example, if dpingroup_sel = 3, and history_ram_data returns 0x5505, the binary representation is "10 10 01 01 01 01 00 00 01 01". Every 2 binary bits represent the logic state of a pin. In this example, the state of each pin is "LLZZHHHHLLHH".

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

DM483e_ConfigurePEAttribute

Synopsis

ViStatus DM483e_ConfigurePEAttribute (vi, pinNo, inputTerminationEnable, HVEnable, activeLoadEnable, differentialComparatorEnable)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViInt32 pinNo (in)

Specifies the pin to be acted upon.
If dpingroup_sel = 1 during DM483e_DPINOpen, pinNo is 0-5 only.
If dpingroup_sel = 2 during DM483e_DPINOpen, pinNo is 6-11 only.
If dpingroup_sel = 3 during DM483e_DPINOpen, pinNo is 0-11.

ViBoolean inputTerminationEnable (in)

Specifies whether to enable input termination.
If input termination is disabled, the specified pin is in high-Z.
If input termination is enabled, the specified pin drives VT voltage level.

ViBoolean HVEnable (in)

Specifies whether to enable high voltage capability.
When HVEnable is enabled, VIH is twice of VT level while VIL maintains.

ViBoolean activeLoadEnable (in)

Specifies whether to enable active load capability.

ViBoolean differentialComparatorEnable (in)

Specifies whether to enable differential comparator capability.
When differential comparator is enabled, odd pins (1, 3, 5, 7, 9 and 11) will be disabled.
The voltage difference between pin 0 and pin 1 will be feed to the comparator of pin 0 only. Same goes to other pair of differential channels.

Descriptions

DM483e_ConfigurePEAttribute programs the attribute of pin electronics of the specified pin.

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

DM483e_ConfigureVectorEngineAttribute

Synopsis

ViStatus DM483e_ConfigureVectorEngineAttribute (vi, triggerEnable, continuousEnable)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViBoolean triggerEnable (in)

Vector Engine only starts to drive vector when an external trigger is received. External trigger can be configured using triggers API.

ViBoolean continuousEnable(in)

Vector Engine continuously repeats the vector until reset is called or argument continuousEnable is disabled. Vector file has to contain at least 5k bits of vector to avoid discontinuity. In addition, the vector file has to be in the multiple of 16 bits to avoid discontinuity.

Descriptions

DM483e_ConfigureVectorEngineAttribute configures the attribute of vector engine.

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

DM483e_ConfigureClockFrequency

Synopsis

ViStatus DM483e_ConfigureClockFrequency (vi, frequency)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViReal64 frequency (in)

Specify the frequency, in Hz.

Descriptions

DM483e_ConfigureClockFrequency programs the clock frequency for states DM483E_CONST_FORCE_STATE_CLOCK and DM483E_CONST_FORCE_STATE_INVERTED_CLOCK as specified in DM483e_DPINForce function.

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

DM483e_AcquireVectorFailCount

Synopsis

ViStatus DM483e_AcquireVectorFailCount (vi, failCount)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViInt32* failCount (out)

Return the total fail count.

Descriptions

DM483e_AcquireVectorFailCount returns the total fail count during the vector execution.

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

DM483e_AcquireChannelVectorFailCount

Synopsis

ViStatus DM483e_AcquireChannelVectorFailCount (vi, pinNo, failCount)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViInt32 pinNo(in)

Specifies the pin to be acted upon.
If dpingroup_sel = 1 during DM483e_DPINOpen, pinNo is 0-5 only.
If dpingroup_sel = 2 during DM483e_DPINOpen, pinNo is 6-11 only.
If dpingroup_sel = 3 during DM483e_DPINOpen, pinNo is 0-11.

ViInt32* failCount (out)

Return the fail count.

Descriptions

DM483e_AcquireChannelVectorFailCount returns the fail count of the specified pin during the vector execution.

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

DM483e_AcquireChannelFirstFailVectorCount

Synopsis

ViStatus DM483e_AcquireChannelFirstFailVectorCount (vi, pinNo, failCount)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViInt32 pinNo(in)

Specifies the pin to be acted upon.
If dpingroup_sel = 1 during DM483e_DPINOpen, pinNo is 0-5 only.
If dpingroup_sel = 2 during DM483e_DPINOpen, pinNo is 6-11 only.
If dpingroup_sel = 3 during DM483e_DPINOpen, pinNo is 0-11.

ViInt32* failCount (out)

Return the first-failed vector location count.

Descriptions

DM483e_AcquireChannelFirstFailVectorCount returns the first location where the vector starts to fail. This function will return -1 if no vector fails.

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

DM483e_ConfigureInputChannelDelay

Synopsis

ViStatus DM483e_ConfigureInputChannelDelay (vi, pinNo, delay)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViInt32 pinNo(in)

Specifies the pin to be acted upon.
If dpingroup_sel = 1 during DM483e_DPINOpen, pinNo is 0-5 only.
If dpingroup_sel = 2 during DM483e_DPINOpen, pinNo is 6-11 only.
If dpingroup_sel = 3 during DM483e_DPINOpen, pinNo is 0-11.

ViReal64 delay (in)

Specifies the delay, in seconds. Delay range is from -10 of clock cycle to 100 of clock cycles.
The delay resolution is 11 ps for delay from 0 ns to 5.621 ns.
For delay more than 5.621 ns, the delay resolution is half of the clock cycle.

Descriptions

DM483e_ConfigureInputChannelDelay configure the delay to be inserted to the specified pin during DM483e_RunVector for pin-to-pin skew adjustment purpose.

Below is an example of how to use DM483e_ConfigureInputChannelDelay to adjust for the input skew, when sending "0010111" to the receiver (pin 0).

1. Reset delay to 0 via DM483e_ConfigureInputChannelDelay.
2. Transmit data "0010111" to receiver.
3. Read back data via DM483e_ReadHistoryRam, and compare with expected results.
4. Acquire the pin fail count via DM483e_AcquireChannelVectorFailCount.
5. For every fail bit, tune the delay via DM483e_ConfigureInputChannelDelay, and repeat steps 3-4, until all results are passing.

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

DM483e_ConfigurePMUVoltageLevel

Synopsis

ViStatus DM483e_ConfigurePMUVoltageLevel (vi, pinNo, level)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViInt32 pinNo(in)

Specifies the pin to be acted upon.
If dpingroup_sel = 1 during DM483e_DPINOpen, pinNo is 0-5 only.
If dpingroup_sel = 2 during DM483e_DPINOpen, pinNo is 6-11 only.
If dpingroup_sel = 3 during DM483e_DPINOpen, pinNo is 0-11.

ViReal64 level (in)

Specifies the voltage level, in volts.

Descriptions

DM483e_ConfigurePMUVoltageLevel configures the voltage level for the specified pin.

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

DM483e_ConfigurePMUVoltageLimit

Synopsis

ViStatus DM483e_ConfigurePMUVoltageLimit (vi, pinNo, high_limit, low_limit)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViInt32 pinNo(in)

Specifies the pin to be acted upon.
If dpingroup_sel = 1 during DM483e_DPINOpen, pinNo is 0-5 only.
If dpingroup_sel = 2 during DM483e_DPINOpen, pinNo is 6-11 only.
If dpingroup_sel = 3 during DM483e_DPINOpen, pinNo is 0-11.

ViReal64 high_limit (in)

Specifies the voltage clamp level for High Clamp Range (VCH), in volts.

ViReal64 low_limit (in)

Specifies the voltage clamp level for Low Clamp Range (VCL), in volts.

Descriptions

DM483e_ConfigurePMUVoltageLimit configures the voltage compliance/clamp limits for the specified pin.

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

DM483e_ConfigurePMUCurrentLevel

Synopsis

ViStatus DM483e_ConfigurePMUCurrentLevel (vi, pinNo, level)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViInt32 pinNo(in)

Specifies the pin to be acted upon.
If dpingroup_sel = 1 during DM483e_DPINOpen, pinNo is 0-5 only.
If dpingroup_sel = 2 during DM483e_DPINOpen, pinNo is 6-11 only.
If dpingroup_sel = 3 during DM483e_DPINOpen, pinNo is 0-11.

ViReal64 level (in)

Specifies the current level, in amps.

Descriptions

DM483e_ConfigurePMUCurrentLevel configures the current level for the specified pin.

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

DM483e_ConfigurePMUCurrentLevelAndRange

Synopsis

ViStatus DM483e_ConfigurePMUCurrentLevelAndRange (vi, pinNo, level, range)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViInt32 pinNo(in)

Specifies the pin to be acted upon.
If dpingroup_sel = 1 during DM483e_DPINOpen, pinNo is 0-5 only.
If dpingroup_sel = 2 during DM483e_DPINOpen, pinNo is 6-11 only.
If dpingroup_sel = 3 during DM483e_DPINOpen, pinNo is 0-11.

ViReal64 level (in)

Specifies the current level, in amps.

ViReal64 range (in)

Specifies the current range, in amps.

Descriptions

DM483e_ConfigurePMUCurrentLevelAndRange configures the current level and its range for the specified pin.

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

DM483e_ConfigurePMUCurrentLimitRange

Synopsis

ViStatus DM483e_ConfigurePMUCurrentLimitRange (vi, pinNo, range)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViInt32 pinNo(in)

Specifies the pin to be acted upon.
If dpingroup_sel = 1 during DM483e_DPINOpen, pinNo is 0-5 only.
If dpingroup_sel = 2 during DM483e_DPINOpen, pinNo is 6-11 only.
If dpingroup_sel = 3 during DM483e_DPINOpen, pinNo is 0-11.

ViReal64 range (in)

Specifies the current limit range, in amps.

Descriptions

DM483e_ConfigurePMUCurrentLimitRange configures the current limit (for source/sink operations) range for the specified pin.

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

DM483e_GetPMUVoltageLevelRange

Synopsis

ViStatus DM483e_GetPMUVoltageLevelRange (vi, pinNo, range)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViInt32 pinNo(in)

Specifies the pin to be acted upon.
If dpingroup_sel = 1 during DM483e_DPINOpen, pinNo is 0-5 only.
If dpingroup_sel = 2 during DM483e_DPINOpen, pinNo is 6-11 only.
If dpingroup_sel = 3 during DM483e_DPINOpen, pinNo is 0-11.

ViReal64* range (out)

Returns the range of the voltage level, in volts.

Descriptions

DM483e_GetPMUVoltageLevelRange returns the range of the voltage level, in volts, of the specified pin.

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

DM483e_GetPMUVoltageLimitRange

Synopsis

ViStatus DM483e_GetPMUVoltageLimitRange (vi, pinNo, range)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViInt32 pinNo(in)

Specifies the pin to be acted upon.
If dpingroup_sel = 1 during DM483e_DPINOpen, pinNo is 0-5 only.
If dpingroup_sel = 2 during DM483e_DPINOpen, pinNo is 6-11 only.
If dpingroup_sel = 3 during DM483e_DPINOpen, pinNo is 0-11.

ViReal64* range (out)

Returns the range of the voltage limit, in volts.

Descriptions

DM483e_GetPMUVoltageLimitRange returns the range of the voltage limit, in volts, of the specified pin.

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

DM483e_GetPMUCurrentLevelRange

Synopsis

ViStatus DM483e_GetPMUCurrentLevelRange (vi, pinNo, range)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViInt32 pinNo(in)

Specifies the pin to be acted upon.
If dpingroup_sel = 1 during DM483e_DPINOpen, pinNo is 0-5 only.
If dpingroup_sel = 2 during DM483e_DPINOpen, pinNo is 6-11 only.
If dpingroup_sel = 3 during DM483e_DPINOpen, pinNo is 0-11.

ViReal64* range (out)

Returns the range of the current level, in amps.

Descriptions

DM483e_GetPMUCurrentLevelRange returns the range of the current level, in amps, of the specified pin.

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

DM483e_GetPMUCurrentLimitRange

Synopsis

ViStatus DM483e_GetPMUCurrentLimitRange (vi, pinNo, range)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViInt32 pinNo(in)

Specifies the pin to be acted upon.
If dpingroup_sel = 1 during DM483e_DPINOpen, pinNo is 0-5 only.
If dpingroup_sel = 2 during DM483e_DPINOpen, pinNo is 6-11 only.
If dpingroup_sel = 3 during DM483e_DPINOpen, pinNo is 0-11.

ViReal64* range (out)

Returns the range of the current limit, in amps.

Descriptions

DM483e_GetPMUCurrentLimitRange returns the range of the current limit, in amps, of the specified pin.

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

DM483e_ConfigurePMUOutputFunction

Synopsis

ViStatus DM483e_ConfigurePMUOutputFunction (vi, pinNo, function)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViInt32 pinNo(in)

Specifies the pin to be acted upon.
If dpingroup_sel = 1 during DM483e_DPINOpen, pinNo is 0-5 only.
If dpingroup_sel = 2 during DM483e_DPINOpen, pinNo is 6-11 only.
If dpingroup_sel = 3 during DM483e_DPINOpen, pinNo is 0-11.

ViInt32 function (out)

Select DVCI or DICV mode.

Descriptions

DM483e_ConfigurePMUOutputFunction configures the pin to operate either in DVCI (Drive-Voltage Clamp Current) or DICV (Drive-Current Clamp Voltage) mode:

function	Setting
DVCI	0 (Default)
DICV	1

Table 9: Output Function

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

DM483e_ConfigurePMUSamplingTime

Synopsis

ViStatus DM483e_ConfigurePMUSamplingTime (vi, pinNo, samplingTime, units)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViInt32 pinNo(in)

Specifies the pin to be acted upon.
If dpingroup_sel = 1 during DM483e_DPINOpen, pinNo is 0-5 only.
If dpingroup_sel = 2 during DM483e_DPINOpen, pinNo is 6-11 only.
If dpingroup_sel = 3 during DM483e_DPINOpen, pinNo is 0-11.

ViReal64 samplingTime (in)

Specifies the sampling time.

ViInt32 units (in)

Specifies the unit of the sampling time:

units	Description
0	Seconds
1	PLC

Table 10: Sampling Time Unit

Descriptions

DM483e_ConfigurePMUSamplingTime 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 11: PLC Setting

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

DM483e_ConfigurePowerLineFrequency

Synopsis

ViStatus DM483e_ConfigurePowerLineFrequency (vi, powerLineFrequency)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViReal64 powerLineFrequency (in)

Specifies the power line frequency.

Descriptions

DM483e_ConfigurePowerLineFrequency specifies the frequency of the AC power line:

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

Table 12: Power Line Frequency

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

DM483e_ConfigurePMUSense

Synopsis

ViStatus DM483e_ConfigurePMUSense (vi, pinNo, sense)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViInt32 pinNo(in)

Specifies the pin to be acted upon.
If dpingroup_sel = 1 during DM483e_DPINOpen, pinNo is 0-5 only.
If dpingroup_sel = 2 during DM483e_DPINOpen, pinNo is 6-11 only.
If dpingroup_sel = 3 during DM483e_DPINOpen, pinNo is 0-11.

ViInt32 sense (in)

Specifies local or remote sense operation.

Descriptions

DM483e_ConfigurePMUSense selects either local or remote sense operation:

sense	Description
0	Local sense (Default)
1	Remote sense

Table 13: Local/Remote Sense Operation

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

DM483e_GetPMUSense

Synopsis

ViStatus DM483e_GetPMUSense (vi, pinNo, sense)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViInt32 pinNo(in)

Specifies the pin to be acted upon.
If dpingroup_sel = 1 during DM483e_DPINOpen, pinNo is 0-5 only.
If dpingroup_sel = 2 during DM483e_DPINOpen, pinNo is 6-11 only.
If dpingroup_sel = 3 during DM483e_DPINOpen, pinNo is 0-11.

ViInt32* sense (out)

Returns the sense (local/remote) operation.

Descriptions

DM483e_GetPMUSense returns the current sense status of the specified pin.

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

DM483e_PMUMeasure

Synopsis

ViStatus DM483e_PMUMeasure (vi, pinNo, measurementType, measurement)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViInt32 pinNo(in)

Specifies the pin to be acted upon.
If dpingroup_sel = 1 during DM483e_DPINOpen, pinNo is 0-5 only.
If dpingroup_sel = 2 during DM483e_DPINOpen, pinNo is 6-11 only.
If dpingroup_sel = 3 during DM483e_DPINOpen, pinNo is 0-11.

ViInt32 measurementType (in)

Specifies voltage or current measurement.

ViReal64* measurement (out)

Returns the measured result.

Descriptions

DM483e_PMUMeasure returns the measured data based on measurementType:

measurementType	Description
0	Measure Current
1	Measure Voltage

Table 14: Voltage/Current Measurement Selection

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

DM483e_ConfigureTriggerEdgeLevel

Synopsis

ViStatus DM483e_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

DM483e_ConfigureTriggerEdgeLevel configures how device can be triggered.

triggerEnum	Description
0	No Connect
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
11	PXI star
12	PXIE DStar A
13	PXIE DStar B
19	Software trigger signal

20	Software trigger signal 2
23	External trigger in 0
24	External trigger in 1
25	External trigger out 0
26	External trigger out 1
27	MIPI trigger out 0
28	MIPI trigger out 1

Table 15: Trigger Source

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

Table 16: Trigger Mode

Note that DM483e_ConfigureTriggerEdgeLevel has no effects on software trigger lines.

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

DM483e_ConfigureTriggerEdgeLevelExtra

ViStatus DM483e_ConfigureTriggerEdgeLevelExtra (vi, triggerEnum, option, ignore_trigger_count)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViInt32 triggerEnum (in)

Specifies the triggering source. Refers to DM483e_ConfigureTriggerEdgeLevel.

ViInt32 option (in)

Specifies the triggering mode. Refers to DM483e_ConfigureTriggerEdgeLevel.

ViInt32 ignore_trigger_count (in)

Specifies the number of trigger count to be ignored before next one is recognized as valid trigger for other operations.
If ignore_trigger_count is 0, the first trigger is a valid trigger.
If ignore_trigger_count is 2, for instance, module will ignore the first two triggers and only begin operation after detecting third in-coming trigger.

Descriptions

DM483e_ConfigureTriggerEdgeLevelExtra configures how device can be triggered.

Note that DM483e_ConfigureTriggerEdgeLevelExtra has no effects on software trigger lines.

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

DM483e_MapTriggerInToTriggerOut

Synopsis

ViStatus DM483e_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

DM483e_MapTriggerInToTriggerOut connects 2 trigger lines together.

inputTerminal / outputTerminal	Description
0	No Connect
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
11	PXI star
12	PXIE DStar A
13	PXIE DStar B
19	Software trigger signal
20	Software trigger signal 2

23	External trigger in 0
24	External trigger in 1
25	External trigger out 0
26	External trigger out 1
27	MIPI trigger out 0
28	MIPI trigger out 1

Table 17: Available Trigger Lines

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

DM483e_ConfigureInputTriggerSelect

Synopsis

ViStatus DM483e_ConfigureInputTriggerSelect (vi, triggerInput)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViInt32 triggerInput (in)

Specifies the triggering source.

Descriptions

DM483e_ConfigureInputTriggerSelect selects the trigger input to be used.

triggerEnum	Description
0	No Connect
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
11	PXI star
12	PXIE DStar A
13	PXIE DStar B
19	Software trigger signal
20	Software trigger signal 2
23	External trigger in 0
24	External trigger in 1
25	External trigger out 0
26	External trigger out 1

27	MIPI trigger out 0
28	MIPI trigger out 1

Table 18: Available Trigger Inputs

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

DM483e_ConfigureInputTriggerSelect

Synopsis

ViStatus DM483e_ConfigureInputTriggerSelect (vi, triggerInput, delay_s)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViInt32 triggerInput (in)

Specifies the triggering source.

ViReal64 delay_s (in)

Specifies the delay after input trigger is detected and before this trigger activates other operations.

Descriptions

DM483e_ConfigureInputTriggerSelect selects the trigger input to be used for operation DM483e_RunVector. Upon power-up nothing is selected for the triggerInput and delay_s is zero.

triggerEnum	Description
0	No Connect
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
11	PXI star
12	PXIE DStar A
13	PXIE DStar B

19	Software trigger signal
20	Software trigger signal 2
23	External trigger in 0
24	External trigger in 1
25	External trigger out 0
26	External trigger out 1
27	MIPI trigger out 0
28	MIPI trigger out 1

Table 19: Available Trigger Inputs

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

DM483e_ConfigureOutputTriggerSelect

Synopsis

ViStatus DM483e_ConfigureOutputTriggerSelect (vi, triggerOutputo, triggerOutput1)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViInt32 triggerOutputo (in)

Specifies the triggering output signal.

ViInt32 triggerOutput1 (in)

Specifies the triggering output signal.

Descriptions

DM483e_ConfigureOutputTriggerSelect selects the trigger output to be used.

triggerEnum	Description
0	No Connect
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
11	PXI star
12	PXIE DStar A
13	PXIE DStar B
19	Software trigger signal
20	Software trigger signal 2
23	External trigger in 0

24	External trigger in 1
25	External trigger out 0
26	External trigger out 1
27	MIPI trigger out 0
28	MIPI trigger out 1

Table 20: Available Trigger Outputs

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

DM483e_DriveSoftwareTrigger

Synopsis

ViStatus DM483e_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

DM483e_DriveSoftwareTrigger drives software trigger immediately. Software trigger lines are digital signals coming from the on-board processor.

select	Description
16	Software trigger signal 0
19	Software trigger signal 1

Table 21: Available Software Trigger Lines

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

DM483e_MIPI_Configure_Clock

Synopsis

ViStatus DM483e_MIPI_Configure_Clock (vi, mipi_pair, freq_Hz)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViInt32 mipi_pair (in)

Specifies the MIPI controller set number.

mipi_pair	Remark	Pin Number	Function
0	Available for dpingroup 1 & 3 only	0	SCLK0
		1	SDATA0
		2	VIO0
1	Available for dpingroup 1 & 3 only	3	VIO1
		4	SCLK1
		5	SDATA1
2	Available for dpingroup 2 & 3 only	6	SCLK2
		7	SDATA2
		8	VIO2
3	Available for dpingroup 2 & 3 only	9	VIO3
		10	SCLK3
		11	SDATA3

Table 22: MIPI Pair

ViInt32 freq_Hz (in)

Specifies the frequency of the full-speed clock.

Descriptions

DM483e_MIPI_Configure_Clock configures the full-clock frequency for MIPI operations.

Refer to Specifications and User Manual for the available number sets of MIPI controller per DM483e, as well as the supported full-clock frequency.

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

DM483e_MIPI_Connect

Synopsis

ViStatus DM483e_MIPI_Connect (vi, mipi_pair, setting)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViInt32 mipi_pair (in)

Specifies the MIPI controller set number.

mipi_pair	Remark	Pin Number	Function
0	Available for dpingroup 1 & 3 only	0	SCLK0
		1	SDATA0
		2	VIO0
1	Available for dpingroup 1 & 3 only	3	VIO1
		4	SCLK1
		5	SDATA1
2	Available for dpingroup 2 & 3 only	6	SCLK2
		7	SDATA2
		8	VIO2
3	Available for dpingroup 2 & 3 only	9	VIO3
		10	SCLK3
		11	SDATA3

Table 23: MIPI Pair

ViInt32 setting (in)

Enable or disable MIPI operations.

Descriptions

DM483e_MIPI_Connect enables or disables MIPI operations for the selected mipi_pair. Once MIPI is enabled, all other operation modes (VECTOR, PMU, DIO, CLOCK, and INVERTED_CLOCK) will be disabled automatically.

setting	Description
0	Disable MIPI
1	Enable MIPI

Table 24: MIPI setting

Refer to Specifications and User Manual for the available number sets of MIPI controller per DM483e, as well as the supported full-clock frequency.

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

DM483e_MIPI_RFFE_WR

Synopsis

ViStatus DM483e_MIPI_RFFE_WR (vi, mipi_pair, Command, Data)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViInt32 mipi_pair (in)

Specifies the MIPI controller set number.

mipi_pair	Remark	Pin Number	Function
0	Available for dpingroup 1 & 3 only	0	SCLK0
		1	SDATA0
		2	VIO0
1	Available for dpingroup 1 & 3 only	3	VIO1
		4	SCLK1
		5	SDATA1
2	Available for dpingroup 2 & 3 only	6	SCLK2
		7	SDATA2
		8	VIO2
3	Available for dpingroup 2 & 3 only	9	VIO3
		10	SCLK3
		11	SDATA3

Table 25: MIPI Pair

ViInt32 Command (in)

Specifies the command for the operation.

Command_frame [11:0] = { Slave_Address[2], Command_frame_lower }		
Register 0 Write	Command_frame_lower[7:0]	8'b{ 1, Data[6:0] }
Register Write	Command_frame_lower[7:0]	8'b{ 0,1,0, Address[4:0] }
Extended Register Write	Command_frame_lower[7:0]	8'b{ 0,0,0,0, BC[3:0] }
Extended Register Write	Command_frame_lower[7:0]	8'b{ 0,0,1,1,0, BC[2:0] }

Long		

Table 26: Write Command Frame

1. Command frame excludes parity bit, which is handled by firmware.
2. Slave address is 4 bits and common for all operations.
3. BC => byte_count for data frame, ie. If BC is zero, then M = 1, refer to data frame table

ViInt32 * Data (in)

Specifies a pointer to an array corresponding to each 8 bit data frame that will be written to the pin.

Data_frame[7:0]		
Register 0 Write	-	-
Register Write	One Data_frame	-
Extended Register Write	One Data_frame for Address	Data_frame[0]= address [7:0], followed by M^2 number of Data Frames containing up to 16 bytes of data. Example: If M = 3, Data_frame[1] = data0, Data_frame[2] = data1, Data_frame[3] = data2.
Extended Register Write Long	Two Data_frame for Address	Data_frame[0]= address [15:8] and Data_frame[1] = address [7:0], followed by M^2 number of Data Frames containing up to 8 bytes of data. Example: If M = 3, Data_frame[2] = data0, Data_frame[3] = data1, Data_frame[4] = data2

Table 27: Write Data Frame

1. Data frame excludes parity bit, which is handled by firmware.
2. BC => byte_count for data frame. If BC is zero, then M = 1, refer to data frame table.

Descriptions

DM483e_MIPI_RFFE_WR writes the data into the pin according to the operation selected.

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

Example Code

This example performs the operation of Extended Register Write Long.

```
Command = (0xF<<8)|(0x6<<3)|(0x2); //Extended register write long, 3 data bytes
Data[0] = 0x1; //Address [15:8]
Data[1] = 0x23; //Address [7:0]
Data[2] = 0x31; //Byte 1 data
Data[3] = 0x31; //Byte 2 data
Data[4] = 0x31; //Byte 3 data
DM483e_MIPI_RFFE_WR(vi, 0, Command, Data); //Select MIPI controller set 0
```

For the Command, (0xF<<8)is the slave address, (0x6<<3)is the command frame and (0x2) is data_frame with the number of byte to be written into the channel.

Note: Refer to appendix for more details.

DM483e_MIPI_RFFE_RD

Synopsis

ViStatus DM483e_MIPI_RFFE_RD (vi, mipi_pair, speed, Command, Data)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViInt32 mipi_pair (in)

Specifies the MIPI controller set number.

mipi_pair	Remark	Pin Number	Function
0	Available for dpingroup 1 & 3 only	0	SCLK0
		1	SDATA0
		2	VIO0
1	Available for dpingroup 1 & 3 only	3	VIO1
		4	SCLK1
		5	SDATA1
2	Available for dpingroup 2 & 3 only	6	SCLK2
		7	SDATA2
		8	VIO2
3	Available for dpingroup 2 & 3 only	9	VIO3
		10	SCLK3
		11	SDATA3

Table 28: MIPI Pair

ViInt32 speed (in)

Specifies the speed of reading.

1 = full speed, 0 = half speed.

ViInt32 Command (in)

Specifies the command of read operation.

Command_frame [11:0] = { Slave_Address[2], Command_frame_lower }		
Register Read	Command_frame_lower[7:0]	8'b{ 0,1,1, Address[4:0] }
Extended Register Read	Command_frame_lower[7:0]	8'b{ 0,0,1,0, BC[3:0] }
Extended Register Read Long	Command_frame_lower[7:0]	8'b{ 0,0,1,1,1, BC[2:0] }

Table 29: Read Command Frame

1. Command frame excludes parity bit, which is handled by firmware.
2. Slave address is 4 bits and common for all operations.

ViInt32* Data (in)

The array of addresses that will be reading from the channel.

Data_frame[7:0]		
Register Read	-	-
Extended Register Read	One Data frame for Address	Data_frame[0]= address [7:0]
Extended Register Read Long	Two Data frame for Address	Data_frame[0]= address [15:8]; Data_frame[1]= address [7:0]

Table 30: Read Data Frame

1. Data frame excludes parity bit, which is handled by firmware.

Descriptions

DM483e_MIPI_RFFE_RD reads the data from the corresponding addresses.

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

Example Code

This example performs Extended Register Read operation.

```
Command = (0xF<<8)|(0x2<<4)|(0x2);//Specifies command for Register Read
DataRead[0] = 0x1; //Specifies the address to read from
DM483e_MIPI_RFFE_RD(vi,0,1,Command,DataRead)//Select MIPI controller set 0. Use full-speed clock.
```

For the Command, (oxF<<8)is the slave address, (ox2<<4)is the command frame and (ox2) is the number of byte to be read from the channel.

Note: Refer to appendix for more details.

DM483e_MIPI_RFFE_RETRIEVE

Synopsis

ViStatus DM483e_MIPI_RFFE_RETRIEVE (vi, mipi_pair, rd_byte_data_count, rd_data, parity_check)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViInt32 mipi_pair (in)

Specifies the MIPI controller set number.

mipi_pair	Remark	Pin Number	Function
0	Available for dpingroup 1 & 3 only	0	SCLK0
		1	SDATA0
		2	VIO0
1	Available for dpingroup 1 & 3 only	3	VIO1
		4	SCLK1
		5	SDATA1
2	Available for dpingroup 2 & 3 only	6	SCLK2
		7	SDATA2
		8	VIO2
3	Available for dpingroup 2 & 3 only	9	VIO3
		10	SCLK3
		11	SDATA3

Table 31: MIPI Pair

ViInt32* rd_byte_data_count (out)

Specifies the number of bytes of data retrieved from the pin.

ViInt32* rd_data (out)

Returns the array of data that retrieved from pin.

rd_data format is "ABCDEFGHIJKLMNOP" in binary, where
"OP" refers to LSB, "MN" refers to bit 1, "KL" refers to bit 2, "AB" refers to MSB, and so on. 2 binary bit represent the logic state, as shown below.

Binary Bits	Description
00	Logic low (L)
01	Logic high (H)
10	Undefined state (Z)

Table 32: Binary Representation of Logic State

ViInt32* parity_check (out)

Returns the array of parity_check corresponding to the array of rd_data that retrieved from the pin.

Value	Description
0	Parity check passes
1	Parity check fails, or undefined read back data found

Table 33: Binary Representation of Logic State

Descriptions

DM483e_MIPI_RFFE_RETRIEVE retrieves and returns an array of data specified in operation DM483e_MIPI_RFFE_RD.

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

DM483e_MIPI_ConfigureInputDelay

Synopsis

ViStatus DM483e_MIPI_ConfigureInputDelay (vi, mipi_pair, value)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViInt32 mipi_pair (in)

Specifies the MIPI controller set number.

mipi_pair	Remark	Pin Number	Function
0	Available for dpingroup 1 & 3 only	0	SCLK0
		1	SDATA0
		2	VIO0
1	Available for dpingroup 1 & 3 only	3	VIO1
		4	SCLK1
		5	SDATA1
2	Available for dpingroup 2 & 3 only	6	SCLK2
		7	SDATA2
		8	VIO2
3	Available for dpingroup 2 & 3 only	9	VIO3
		10	SCLK3
		11	SDATA3

Table 34: MIPI Pair

ViInt32 value (in)

Specifies the delay inserted before data sampling during read operation starts.

$0 \leqslant$ value $\leqslant 7$.

Value is 0.5*MIPI full speed clock period.

If value is 1, then, input delay is 0.5*MIPI full speed clock period.

If value is 2, then, input delay is 1*MIPI full speed clock period.

Descriptions

DM483e_MIPI_ConfigureInputDelay is needed when the cable distance between DM483e and the DUT is too long. This API tells the MIPI IP to sample the incoming SDATA slower than usual due to long cable delay.

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

DM483e_MIPI_ConfigureOutputTrigger

Synopsis

ViStatus DM483e_MIPI_ConfigureOutputTrigger (vi, mipi_pair, on_h, pulse_width_s)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViInt32 mipi_pair (in)

Specifies the MIPI controller set number.

mipi_pair	Remark	Pin Number	Function
0	Available for dpingroup 1 & 3 only	0	SCLK0
		1	SDATA0
		2	VIO0
1	Available for dpingroup 1 & 3 only	3	VIO1
		4	SCLK1
		5	SDATA1
2	Available for dpingroup 2 & 3 only	6	SCLK2
		7	SDATA2
		8	VIO2
3	Available for dpingroup 2 & 3 only	9	VIO3
		10	SCLK3
		11	SDATA3

Table 35: MIPI Pair

ViInt32 on_h (in)

Configures output trigger after MIPI operation.

0: turn off output trigger after MIPI operation.

1: turn on output trigger after MIPI operation.

ViReal64 pulse_width_s (in)

Configures output trigger pulse width in s.

128e-9 $\leqslant$ pulse_width_s $\leqslant$ 0.001.

Descriptions

DM483e_MIPI_ConfigureOutputTrigger turns on or off the output trigger after the each MIPI operation like DM483e_MIPI_RFFE_WR and DM483e_MIPI_RFFE_RD.

Please note that this function will follow its last state, until power off.

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

DM483e_SPI_CHSEL

Synopsis

ViStatus DM483e_SPI_CHSEL (vi, spi_group, clk_chsel, ncs_chsel, mosi_chsel, miso_chsel)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViInt32 spi_group (in)

Specifies the SPI group number.

dpingroup_sel	spi_group	Pin
1	0	Pin 0 to Pin 5
2	0	Pin 6 to Pin 11
3	0	Pino to Pin 11
	1	

Table 36: SPI Group

ViInt32 clk_chsel (in)

Specifies the pin number of the SPI clock.

ViInt32 ncs_chsel (in)

Specifies the pin number of the SPI NCS (chip select).

ViInt32 mosi_chsel (in)

Specifies the pin number of the SPI MOSI (master-output slave-input) signal.

ViInt32 miso_chsel (in)

Specifies the pin number of the SPI MISO (master-input slave-output) signal.

Descriptions

DM483e_SPI_CHSEL specifies the channel for all SPI signals, i.e. serial clock, chip select, master-output-slave-input (MOSI), master-input-slave-output (MISO).

dpingroup_sel	spi_group	Valid Pin Number	Comment
1	0	-1 to 5	-1 means no pin is selected
2	0	-1 to 5	
3	0	-1 to 5	
	1	-1, 6 to 11	

Table 37: Valid Pin Number

After calling DM483e_SPI_CHSEL, pin direction is automatically configured as follows:

Signal	Pin Direction
CLK	Output
NCS	Output
MOSI	Output
MISO	Input

Table 38: SPI Pin Direction

If MOSI and MISO refers to the same pin number, it means the SPI data pin is bidirectional. The direction of the pin during SPI operation is set by 25th-bit of spi_wr_rd_data of DM483e_SPI_START() API.

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

DM483e_SPI_CONFIGURE

Synopsis

ViStatus DM483e_SPI_CONFIGURE (vi, spi_group, spi_clk_divider, CPOL, CPHA)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViInt32 spi_group (in)

Specifies the SPI group number.

dpingroup_sel	spi_group	Pin
1	0	Pin 0 to Pin 5
2	0	Pin 6 to Pin 11
3	0	Pino to Pin 11
	1	

Table 39: SPI Group

ViInt32 spi_clk_divider (in)

Specifies the divider ratio to derive the SPI clock frequency.

ViInt32 CPOL (in)

Specifies the clock polarity of the SPI interface. Valid values are 0 to 1.

ViInt32 CPHA (in)

Specifies the clock phase of the SPI interface. Valid values are 0 to 1.

Descriptions

DM483e_SPI_CONFIGURE configures the clock properties of the SPI interface.

At CPOL=0 the base value of the clock is zero:

- For CPHA=0, data is captured on the clock's rising edge (low to high transition) and data is propagated on a falling edge (high to low clock transition).

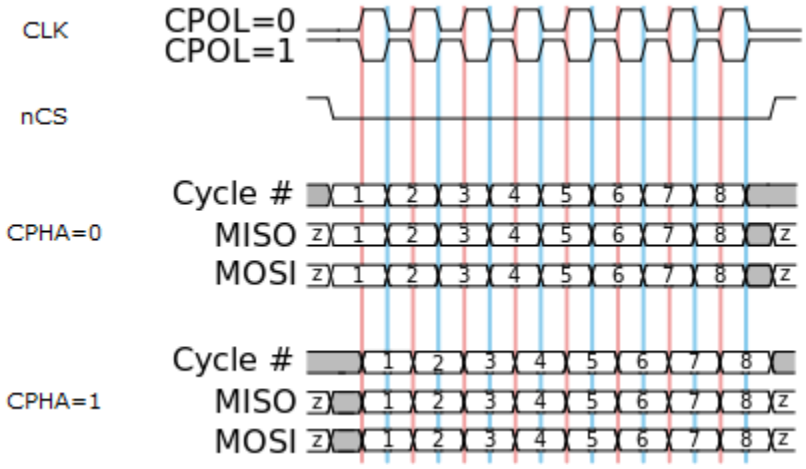
- For CPHA=1, data is captured on the clock's falling edge and data is propagated on a rising edge.

At CPOL=1 the base value of the clock is one (inversion of CPOL=0)

- For CPHA=0, data is captured on clock's falling edge and data is propagated on a rising edge.
- For CPHA=1, data is captured on clock's rising edge and data is propagated on a falling edge.

That is, CPHA=0 means sample on the leading (first) clock edge, while CPHA=1 means sample on the trailing (second) clock edge, regardless of whether that clock edge is rising or falling. Note that with CPHA=0, the data must be stable for a half cycle before the first clock cycle. For all CPOL and CPHA modes, the initial clock value must be stable before the chip select line goes active.

A timing diagram is shown below.



Refer to Specifications for details.

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

DM483e_SPI_START

Synopsis

ViStatus DM483e_SPI_START (vi, spi_group, spi_wr_rd_data, spi_byte_length, delay_s_after_byte, bit_width)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViInt32 spi_group (in)

Specifies the SPI group number.

dpingroup_sel	spi_group	Pin
1	0	Pin 0 to Pin 5
2	0	Pin 6 to Pin 11
3	0	Pino to Pin 11
	1	

Table 40: SPI Group

ViInt32* spi_wr_rd_data (in)

Specifies an array of data to be written to the SPI device.

ViInt32 spi_byte_length (in)

Specifies the array size of the data to be written to the SPI device.

ViReal64* delay_s_after_byte (in)

Specifies an array of delay inserted between each data in the spi_wr_rd_data array.

ViInt32 bit_width (in)

Specifies the width of the data.

Descriptions

DM483e_SPI_START executes the SPI commands. An array can be sent to the SPI device consecutively. Each data length is specified by bit_width. In addition, a small delay can be inserted between each data transfer to the SPI device.

If spi_wr_rd_data is written with a 1 on the 25th bit, this indicates the data is to be read back. If x number of data packets are transmitted with 1 in the 25th bit, then x number of data will be read back into rd_spi_data when DM483e_SPI_RETRIEVE() is called.

For example:

data[0] = 0xaa
data[1] = 0x10000ff
data[2] = 0x10000ae
data[3] = 0x11
data[4]=0x10000de

Then data[1], data[2], data[4] will be retrieved and stored in rd_spi_data where rd_byte_length is equal to 3.

Refer to Specifications for valid inputs.

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

DM483e_SPI_STATUS

Synopsis

ViStatus DM483e_SPI_STATUS (vi, spi_group, spistatus)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViInt32 spi_group (in)

Specifies the SPI group number.

dpingroup_sel	spi_group	Pin
1	0	Pin 0 to Pin 5
2	0	Pin 6 to Pin 11
3	0	Pino to Pin 11
	1	

Table 41: SPI Group

ViInt32* spistatus (out)

Returns the SPI operation status.

Descriptions

DM483e_SPI_STATUS returns the SPI operation status.

spistatus	Description
0	Done
1	Busy
2	Idle

Table 42: SPI Status

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

DM483e_SPI_RETRIEVE

Synopsis

ViStatus DM483e_SPI_RETRIEVE (vi, spi_group, rd_byte_length, rd_spi_data, rd_spi_biterror)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViInt32 spi_group (in)

Specifies the SPI group number.

dpingroup_sel	spi_group	Pin
1	0	Pin 0 to Pin 5
2	0	Pin 6 to Pin 11
3	0	Pino to Pin 11
	1	

Table 43: SPI Group

ViInt32* rd_byte_length (out)

Returns the size of the read back data array.

ViInt32* rd_spi_data (out)

Returns an array of read back data from SPI device. Sufficient memory space must be allocated.

ViInt32* rd_spi_biterror (out)

If any bit of MISO sampled is above VOL but below VOH, then, the respective bit for rd_spi_biterror output will be high.

Descriptions

DM483e_SPI_RETRIEVE is used to retrieve the read back data from the SPI device. During DM483e_SPI_START, if m number of data packets are transmitted with 1 in the 25th bit, then m number of data will be read back into rd_spi_data when DM483e_SPI_RETRIEVE is called.

For example:

data[0] = 0xaa
data[1] = 0x10000ff
data[2] = 0x10000ae
data[3] = 0x11
data[4]=0x10000de

Then data[1], data[2], data[4] will be retrieved and stored in rd_spi_data where rd_byte_length is equal to 3.
Likewise, if there is no data with 9th bit set, then the expected rd_byte_length output is 0.

User need to allocate enough memory for rd_spi_data and rd_spi_biterror output.

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

DM483e_SPI_NCS_ACTIVE_STATE

Synopsis

ViStatus DM483e_SPI_NCS_ACTIVE_STATE (vi, spi_group, ncs_active_state)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViInt32 spi_group (in)

Specifies the SPI group number.

dpingroup_sel	spi_group	Pin
1	0	Pin 0 to Pin 5
2	0	Pin 6 to Pin 11
3	0	Pino to Pin 11
	1	

Table 44: SPI Group

ViInt32 ncs_active_state (in)

Specifies the active logic state of NCS signal.

Descriptions

By default, upon power-up, default NCS active state is low, as indicated by its name "NCS". For some devices, the SPI chip-select signal is active high. To configure chip-select to active high, set ncs_active_state to 1. If ncs_active_state is 0, SPI NCS is active low.

If the DM483e remains powered-up, ncs_active_state remains as it is in hardware until it is changed by calling this API. It is not affected by any other APIs.

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

DM483e_I2C_CHSEL

Synopsis

ViStatus DM483e_I2C_CHSEL (vi, i2c_group, scl_chsel, sda_chsel)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViInt32 i2c_group (in)

Specifies the I2C group number.

dpingroup_sel	i2c_group	Pin
1	0	Pin 0 to Pin 5
2	0	Pin 6 to Pin 11
3	0	Pino to Pin 11
	1	

Table 45: I2C Group

ViInt32 scl_chsel (in)

Specifies the channel number for serial clock signal.

ViInt32 sda_chsel (in)

Specifies the channel number for serial data signal.

Descriptions

DM483e_I2C_CHSEL specifies the channel for all I2C signals, i.e. serial clock and serial data.

dpingroup_sel	I2c_group	Valid Pin Number	Comment
1	0	-1 to 5	-1 means no pin is selected
2	0	-1 to 5	
3	0	-1 to 5	
	1	-1, 6 to 11	

Table 46: Valid Pin Number

It is recommended for the user to configure the SCL and SDA pins as input as I2C signals are pulled-up during IDLE state.

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

DM483e_I2C_CONFIGURE

Synopsis

ViStatus DM483E_I2C_CONFIGURE (vi, i2c_group, scl_freq_Hz)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViInt32 i2c_group (in)

Specifies the I2C group number.

dpingroup_sel	i2c_group	Pin
1	0	Pin 0 to Pin 5
2	0	Pin 6 to Pin 11
3	0	Pino to Pin 11
	1	

Table 47: I2C Group

ViInt32 scl_freq_Hz (in)

Specifies the frequency of serial clock, in Hertz.

Descriptions

DM483E_I2C_CONFIGURE specifies the frequency of serial clock signal. Refer to Specifications for supported frequencies.

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

DM483E_I2C_START

Synopsis

ViStatus DM483E_I2C_START (vi, i2c_group, command_count, i2c_command, rddata_count, i2c_rddata, i2c_rddata_biterror, i2c_timeout_s)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViInt32 i2c_group (in)

Specifies the I2C group number.

dpingroup_sel	i2c_group	Pin
1	0	Pin 0 to Pin 5
2	0	Pin 6 to Pin 11
3	0	Pino to Pin 11
	1	

Table 48: I2C Group

ViInt32 command_count (in)

Specifies the number of I2C commands.

ViInt32* i2c_command (in)

Specifies an array of I2C commands.

ViInt32* rddata_count (out)

Returns the number of read back data from the I2C device.

ViInt32* i2c_rddata (out)

Returns an array of read back data from I2C device. Sufficient memory space must be allocated.

ViInt32* i2c_rddata_biterror (out)

If any bit of SDA sampled is above VOL but below VOH, then, the respective bit for i2c_rddata_biterror output will be high. Sufficient memory space must be allocated.

ViReal64 i2c_timeout_s (in)

Specifies the timeout setting of the I2C operation, in seconds.

Descriptions

DM483E_I2C_START executes I2C write/read commands.

i2c_command also includes the first command which is always the (start_address + r/w bit).

For the i2c_command array:

bito-7 is the i2c data.
bit 8 is rsbit. If set to 1, then "restart" is perform after driving bito-7.
bit 9 is stopbit. If set to 1, then "stop" is perform after driving bito-7.
bit 10 is writebit. If set to 1, then sda bito-7 is output during i2c operation.
bit 11 is readbit. If set to 1, then, sda bito-7 is input during i2c operation.
bit 12 is ackbit. If ackbit is 1 when writebit is 1, then ACK is ignored, and no "restart" if there is no ACK from the slave I2C device.

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

DM483e_InstructionMode_StartRecording

Synopsis

ViStatus DM483e_InstructionMode_StartRecording (vi, instruction_core_no)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViInt32 instruction_core_no (in)

Specifies the core number for instruction mode.

dpingroup_sel	instruction_core_no	Pin
1	0	Pin 0 to Pin 5
2	0	Pin 6 to Pin 11
3	0	Pino to Pin 11
	1	

Table 49: Instruction Core Number

Descriptions

Upon calling DM483e_InstructionMode_StartRecording, subsequent called APIs will be recorded in the instruction RAM. This is just for recording purpose, and no actual operations are executed.

The following APIs are supported by instruction mode:

1. DM483e_DPINOn
2. DM483e_DPINOff
3. DM483e_DrivePin
4. DM483e_ReadPin
5. DM483e_MIPI_Configure_Clock

6. DM483e_MIPI_Connect
7. DM483e_MIPI_RFFE_WR
8. DM483e_MIPI_RFFE_RD
9. DM483e_MIPI_RFFE_Retrieve
10. DM483e_MIPI_ConfigureInputDelay

The number of APIs that can be recorded depends on the complexity of the APIs called. When the instruction RAM is overflow, the subsequent called APIs will return error. The recording process will be terminated at the same time.

You must call DM483e_InstructionMode_FinishRecording to terminate the recording process. On the other hand, calling DM483e_Reset or DM483e_ResetGroup will also terminate the recording process.

You do not need to repeat the recording process after each DM483e_InstructionMode_Run API, unless you want to change the content in the Instruction Mode.

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

DM483e_InstructionMode_FinishRecording

ViStatus DM483e_InstructionMode_FinishRecording (vi, instruction_core_no)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViInt32 instruction_core_no (in)

Specifies the core number for instruction mode.
Refers to DM483e_InstructionMode_StartRecording for details.

Descriptions

DM483e_InstructionMode_FinishRecording stops the recording for the instruction mode operations.

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

DM483e_InstructionMode_Reset

ViStatus DM483e_InstructionMode_Reset (vi, instruction_core_no)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViInt32 instruction_core_no (in)

Specifies the core number for instruction mode.
Refers to DM483e_InstructionMode_StartRecording for details.

Descriptions

DM483e_InstructionMode_Reset resets the instruction mode core.

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

DM483e_InstructionMode_Delay

ViStatus DM483e_InstructionMode_Delay (vi, instruction_core_no, delay_s)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViInt32 instruction_core_no (in)

Specifies the core number for instruction mode.
Refers to DM483e_InstructionMode_StartRecording for details.

ViReal64 delay_s (in)

Specifies delay in seconds
Resolution of delay_s is 1us. Minimum delay_s is 0s. Maximum delay_s is 1s.

Descriptions

DM483e_InstructionMode_Delay is only used for instruction mode to insert delay between any operations when instruction mode is running.

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

DM483e_InstructionMode_Run

ViStatus DM483e_InstructionMode_Run (vi, instruction_core_no, wait_trigger_before_start, wait_trigger_before_loop, loop_count, timeout_s)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViInt32 instruction_core_no (in)

Specifies the core number for instruction mode.

Refers to DM483e_InstructionMode_StartRecording for details.

ViInt32 wait_trigger_before_start (in)

Specifies whether to wait for trigger to run the instruction mode.

wait_trigger_before_start	Description
0	Execute the instructions immediately without waiting for trigger
1	Only execute the instructions after receive the input trigger as set by DM483e_InstructionMode_InputTrigger

Table 50: wait_trigger_before_start Setting

ViInt32 wait_trigger_before_loop (in)

Specifies whether to wait for trigger to run next loop of instructions.

wait_trigger_before_loop	Description
0	Do not wait for trigger during looping
1	Wait for trigger during looping. If input trigger is not received before timeout, then, the looping stops.

Table 51: wait_trigger_before_loop Setting

Vint32 loop_count (in)

Specifies the number of loops to run the instructions.

If loop_count is 0, it means looping indefinitely. Valid values are 0 to 64000.

ViReal64 timeout_s (in)

Specifies the time-out for input trigger.

If timeout_s is 0, it means no time-out.

Resolution of timeout_s is 100us. Maximum timeout_s is 6.4s. If timeout more than 6.4s is required, put '0' as no timeout

Descriptions

DM483e_InstructionMode_Run executes the instructions recorded.

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

DM483e_InstructionMode_Stop

ViStatus DM483e_InstructionMode_Stop (vi, instruction_core_no)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViInt32 instruction_core_no (in)

Specifies the core number for instruction mode.
Refers to DM483e_InstructionMode_StartRecording for details.

Descriptions

DM483e_InstructionMode_Stop terminates the instruction mode immediately.

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

DM483e_InstructionMode_Status

ViStatus DM483e_InstructionMode_Status (vi, instruction_core_no, status, current_loop_count)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViInt32 instruction_core_no (in)

Specifies the core number for instruction mode.

Refers to DM483e_InstructionMode_StartRecording for details.

ViInt32* status (in)

Returns the current status of the instruction mode.

status	Description
0	Idle.
1	Armed, but input trigger is not detected yet.
2	Busy
3	Completed all loops of instruction sets.
4	Timeout happens due to missing input trigger

Table 52: Status

ViInt32* current_loop_count (in)

Returns the current loop count of instruction mode.

The count starts from 1. Once the instruction mode starts, this value is incremented after each loop.

Descriptions

DM483e_InstructionMode_Status returns the status of Instruction Mode.

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

DM483e_InstructionMode_RetrieveResults

ViStatus DM483e_InstructionMode_RetrieveResults (vi, instruction_core_no, results_count, results)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViInt32 instruction_core_no (in)

Specifies the core number for instruction mode.
Refers to DM483e_InstructionMode_StartRecording for details.

ViInt32* results_count (out)

Returns the number of results returned.

ViReal64* results (out)

Returns the values of results captured.

Descriptions

DM483e_InstructionMode_RetrieveResults returns all results captured during instruction mode. This API returns all the results in double format. You must allocate enough memory for all the results.

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

DM483e_InstructionMode_RetrieveResultsInt

ViStatus DM483e_InstructionMode_RetrieveResultsInt (vi, instruction_core_no, results_count, results)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViInt32 instruction_core_no (in)

Specifies the core number for instruction mode.
Refers to DM483e_InstructionMode_StartRecording for details.

ViInt32* results_count (out)

Returns the number of results returned.

ViReal64* results (out)

Returns the values of results captured.

Descriptions

DM483e_InstructionMode_RetrieveResultsInt returns all results captured during instruction mode. This API returns all the results in integer format. You must allocate enough memory for all the results.

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

DM483e_InstructionMode_ConfigureInputTrigger

ViStatus DM483e_InstructionMode_ConfigureInputTrigger (vi, instruction_trigin_no, ViInt32 triggerInput, delay_s)

Arguments

ViSession vi (in)

Specifies the instrument handle.

ViInt32 instruction_trigin_no (in)

Specifies the trigger input number for instruction mode.

dpingroup_sel	instruction_trigin_no	Pin
1	0	Pin 0 to Pin 5
2	0	Pin 6 to Pin 11
3	0	Pino to Pin 11
	1	

Table 53: Instruction Trigger Input Number

ViInt32 triggerInput (in)

Specifies the triggering source. Refers to DM483e_ConfigureInputTriggerSelect.

ViReal64 delay_s (in)

Specifies the delay after input trigger is detected and before this trigger activates other operations.

Descriptions

DM483e_InstructionMode_ConfigureInputTrigger is used to select which trigger line is connected to instruction_trigino (for core 0) or instruction_trigin1 (for core 1). For an example, if triggerInput is 2 and dpingroupsel is 2, then PXI_TRIG1 is connected to instruction_trigin1.

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

Section 10: Appendix – RFFE Supported Command Sequences

Mandatory Commands M	Mandatory Commands S	Hex	Description	SSC	Command Frame	Data Frame
Y	O	00 - 0F	Extended Register Write	1 0	SA[3:0] 0 0 0 0 BC[3:0] (0000 – 1111) P	Address[7:0] P Up to 16 Bytes of Data with Parity Master must support up to 4 bytes BP
		10	(Reserved)		0 0 0 1 0 0 0 0 P	Undefined
		11	(Reserved)		0 0 0 1 0 0 0 1 P	Undefined
		12	(Reserved)		0 0 0 1 0 0 1 0 P	Undefined
		13	(Reserved)		0 0 0 1 0 0 1 1 P	Undefined
		14	(Reserved)		0 0 0 1 0 1 0 0 P	Undefined
		15	(Reserved)		0 0 0 1 0 1 0 1 P	Undefined
		16	(Reserved)		0 0 0 1 0 1 1 0 P	Undefined
		17	(Reserved)		0 0 0 1 0 1 1 1 P	Undefined
		18	(Reserved)		0 0 0 1 1 0 0 0 P	Undefined
		19	(Reserved)		0 0 0 1 1 0 0 1 P	Undefined
		1A	(Reserved)		0 0 0 1 1 0 1 0 P	Undefined
		1B	(Reserved)		0 0 0 1 1 0 1 1 P	Undefined
		1C	(Reserved)		0 0 0 1 1 1 0 0 P	Undefined
		1D	(Reserved)		0 0 0 1 1 1 0 1 P	Undefined
		1E	(Reserved)		0 0 0 1 1 1 1 0 P	Undefined
		1F	(Reserved)		0 0 0 1 1 1 1 1 P	Undefined
Y	O	20 - 2F	Extended Register Read		SA[3:0] 0 0 1 0 BC[3:0] (0000 – 1111) P	Address[7:0] P BP Up to 16 Bytes of Data with Parity Master must support up to 4 bytes BP
O	O	30 - 37	Extended Register Write Long		SA[3:0] 0 0 1 1 0 BC[2:0] (000 – 111) P	Address[15:8] P Address[7:0] P Up to 8 Bytes of Data with Parity BP
O	O	38 - 3F	Extended Register Read Long		SA[3:0] 0 0 1 1 1 BC[2:0] (000 – 111) P	Address[15:8] P Address[7:0] P BP Up to 8 Bytes of Data with Parity BP
Y	O	40 - 5F	Register Write		SA[3:0] 0 1 0 Address[4:0] (00000 – 11111) P	Data[7:0] P BP
Y	O	60 - 7F	Register Read		SA[3:0] 0 1 1 Address[4:0] (00000 – 11111) P	BP Data[7:0] P BP
Y	O	80 - FF	Register 0 Write		SA[3:0] 1 Data[6:0] (0000000 – 1111111) P	BP

Notes
All non-implemented Command Sequences and register accesses are ignored, resulting in a null response.

Nomenclature
BC = Byte Count; M = Master; P = Parity; Y = Mandatory
BP = Bus Park; O = Optional; SA = Slave Address; S = Slave; SSC = Sequence Start Condition

Figure 1: RFFE Supported Command Sequences

Section 11: Error Message

Error Code (Hex)	Description
0xAE050006	AEMHW_E_WRONG_MSG_SIZE
0xAE050007	AEMHW_E_API_NOT_IMPLEMENTED
0xAE050009	AEMHW_E_FIRMWARE_FILE_CHECKSUM_ERROR
0xAE05000A	AEMHW_E_FIRMWARE_FILESIZE_ERROR
0xAE05000B	AEMHW_E_WAIT4UNLOCK_TIMEOUT
0xAE090001	AEMPXIE_E_OPEN_FAIL
0xAE090002	AEMPXIE_E_DLL_NOT_FOUND
0xAE090003	AEMPXIE_E_API_NOT_FOUND
0xAE090004	AEMPXIE_E_INVALID_MULTI_SITE_MODE
0xAE040002	AEMVECTOR_SYNTAX_ERROR
0xAE040003	AEMVECTOR_FILE_CORRUPTED
0xAE040008	AEMVECTOR_NOT_FOUND
0xAE04000C	AEMVECTOR_INVALID_ARGUMENT
0xAE04000D	AEMVECTOR_INVALID_CH
0xAE04000E	AEMVECTOR_UNDEFINED_PIN
0xAE0B0001	AEMDM_E_INVALID_ARGUMENTS
0xAE0B0002	AEMDM_E_INVALID_CALIBRATION_ARGUMENTS
0xAE0B0003	AEMDM_E_UNDEFINED_VECTOR_SET
0xAE0B0004	AEMDM_E_CHANNEL_NOT_IN_PMU_STATE
0xAE0B0005	AEMDM_E_ADATE_TIMEOUT
0xAE0B0006	AEMDM_E_ADC_TIMEOUT
0xAE0B0007	AEMDM_E_DDS_TIMEOUT
0xAE0B0008	AEMDM_E_PATTERN_MEMORY_TIMEOUT
0xAE0B0009	AEMDM_E_HISTORY_RAM_TIMEOUT
0xAE0B000A	AEMDM_E_D2R_FAIL_CALIBRATION_STAGE1
0xAE0B000B	AEMDM_E_CAL_HEADER_NOT_FOUND
0xAE0B000C	AEMDM_E_CAL_SIZE_NOT_MATCH
0xAE0B000D	AEMDM_E_INVALID_CAL_ACCUM_COUNT
0xAE0B000E	AEMDM_E_INVALID_CAL_DATE
0xAE0B000F	AEMDM_E_VECTOR_ENGINE_BUSY
0xAE0B0010	AEMDM_E_EXCEED_VECTOR_MEMORY_LIMIT
0xAE0B0011	AEMDM_E_EXCEED_TOTAL_VECTOR_MEMORY_LIMIT
0xAE0B0012	AEMDM_E_DM483E_NOT_FOUND
0xAE0B0013	AEMDM_E_D2R_FAIL_CALIBRATION_STAGE2

0xAE0B0014	AEMDM_E_INPUT_DELAY_OUT_OF_RANGE
0xAE0B0015	AEMDM_E_EXCEED_PERIOD_LIMIT
0xAE0B0016	AEMDM_E_EXCEED_READ_VECTOR_LIMIT
0xAE0B0017	AEMDM_E_CHANNEL_NOT_IN_DIO_STATE
0xAE0B0018	AEMDM_E_INVALID_DIO_VALUE
0xAE0B0041	AEMDM_E_INVALID_CLK_FREQ
0xAE0B0042	AEMDM_E_INVALID_CH
0xAE0B0043	AEMDM_E_INVALID_MIPI_COMMAND
0xAE0B0044	AEMDM_E_INVALID_MIPI_DATA
0xAE0B0045	AEMDM_E_INVALID_MIPI_CONNECT
0xAE0B0046	AEMDM_E_INVALID_MIPI_SPEED
0xAE0B0047	AEMDM_E_MIPI_STATUS_TIMEOUT
0xAE0B0048	AEMDM_E_INVALID_MIPI_RDDATA_COUNT
0xAE0B0049	AEMDM_E_INVALID_MIPI_RDDATA

Section 12: Revision History

1.0	AUG 2014	INITIAL RELEASE

Section 13: Contact Us

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

Aemulus Corporation Berhad
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.