



816x VXI*plug&play* Instrument Driver Help Contents



Programming Information

[Using the Instrument Driver](#)

[Introduction to Programming](#)

[Applications Functions](#)

[Equally Spaced Datapoints](#)

[Lambda Scan Application](#)

[Multi Frame Lambda Scan Application](#)

Example Programs

[Example Programs](#)

Reference Information

[Function Reference \(Alphabetical\)](#)

[Function Reference \(Hierarchical\)](#)

Where to Find

[Online and User Manual Information](#)

[In Case of Trouble](#)

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



Using the Instrument Driver

The Keysight Technologies 816x VXIplug&play Instrument Driver revision 4.5 complies with the following:

WIN 95/98 and WIN NT and WIN XP System Frameworks

VISA revision F.01.02.00 and newer

This version is also tested and used with Windows 7, including 64-bit systems, and with Keysight VISA 80.3.17281.1, installed with IO Library Suites 16.3.17218

8163A Lightwave Multimeter Firmware revision 5.0

8164A Lightwave Measurement System Firmware revision 5.0

8166A Lightwave Multichannel System Firmware revision 5.0

8163B Lightwave Multimeter Firmware revision 5.0

8164B Lightwave Measurement System Firmware revision 5.0

8166B Lightwave Multichannel System Firmware revision 5.0

N773xA Optical Switch Firmware revision 1.2

N7744A and N7745A Optical Power Meter Firmware revision 1.18

N7747A and N7748A Optical Power Meter Firmware revision 1.30

N775xA-series Optical Attenuator and Power Meter Firmware revision 1.00

N776xA-series Optical Attenuator Firmware revision 1.00

The following information is common to all programs that use the Keysight Technologies 816x VXIplug&play instrument driver:

[VISA, VXIplug&play, and the Keysight Technologies 816x VXIplug&play instrument driver](#)

[Directory Structure](#)

[Opening an Instrument Session](#)

[Closing an Instrument Session](#)

[VISA Data Types and Selected Constant Definitions](#)

[Error Handling](#)

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



VISA, VXI*plug&play*, and the Keysight Technologies 816x Instrument Driver

This Keysight Technologies 816x VXI*plug&play* instrument driver conforms to all portions of the VXI*plug&play* driver standard which are applicable to conventional GP-IB and other non-VXI instruments (that is, rack and stack instruments).

The formal VXI*plug&play* standard only covers VXI instruments, and two elements of the standard do not apply to the Keysight Technologies 816x VXI*plug&play* instrument driver since it is not a VXI instrument. Firstly, there is no soft front panel, as the Keysight Technologies 816x VXI*plug&play* instrument driver can be controlled from the instrument's front panel. Secondly, there is no knowledge base file, which is primarily a physical description of a VXI board.

Aside from these exceptions, you'll find the same features in this driver as in Keysight's other VXI*plug&play* drivers.

1. Conformance with the VXI*plug&play* standard. The only exceptions are that it does not have a soft front panel or a knowledge base file.
2. It is built on top of, and uses the services provided by VISA. VISA supports GP-IB and VXI protocols. The driver can be used with any GP-IB card for which the manufacturer has provided a VISA DLL. Instruments on LAN or USB connections with VISA addresses can also be controlled.
3. It includes a "Function Panel" (.fp) file which allows it to be used with visual programming environments such as Keysight VEE, LabWindows, and LabVIEW.
4. It includes a comprehensive on-line help file which complements the instrument manual. The help file presents application programming examples, a cross-reference between instrument commands and drive functions, and detailed documentation of each function, with examples.
5. It includes a Visual Basic include file (.bas) which contains the function calls in Visual Basic syntax, so that driver functions can be called from Visual Basic. If you use Visual Basic with this driver, you should be familiar with C/C++ function declarations. In particular, care must be taken when working with C/C++ pointers.

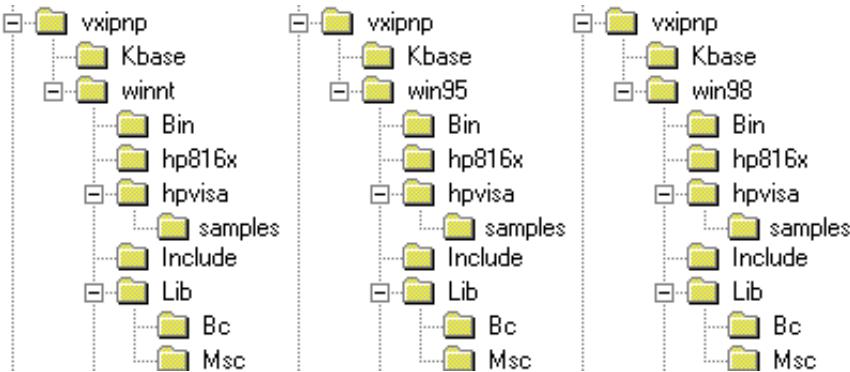
© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



Directory Structure

The setup program which installs the Keysight Technologies 816x VXIplug&play instrument driver creates the VXIPNP directory if it does not already exist. The structures for the Windows NT, Windows 95, and Windows 98 VXIPNP subdirectory tree are:



In the directory example, `hp816x` is a directory containing the instrument driver. There would be a directory for each instrument driver. In more recent Windows XP installations, the `hp816x` directory is usually located within the IVI Foundation directory, under VISA.

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



Opening an Instrument Session

Introduction

To control an instrument from a program, a communication path between the computer/controller and the instrument must be opened. This path is known as an instrument session and is opened with the [hp816x_init](#) function:

```
ViStatus hp816x\_init(ViRsrc resourceName, ViBoolean IDQuery, ViBoolean reset, ViPSSession ihandle);
```

Instruments are assigned a handle when the instrument session is opened. The handle, which is a pointer to the instrument, is the first parameter passed in all subsequent calls to driver functions.

The parameters of function [hp816x_init](#) include:

ViRsrc resourceName - the [address of the instrument](#).

ViBoolean IDQuery - a Boolean flag which indicates if in-system verification should be performed. Passing VI_TRUE (1) will perform an in-system verification. Passing VI_FALSE (0) will not. If you set id_query to false it is possible to use the generic functions of the instrument driver with other instruments.

ViBoolean reset - a Boolean flag which indicates if the instrument should be reset when it is opened. Passing VI_TRUE (1) will perform a reset when the session is opened. Passing VI_FALSE (0) will not perform a reset.

ViPSSession instrumentHandle - a pointer to an instrument session. **instrumentHandle** is the handle which addresses the instrument and is the first parameter passed in all driver functions.

Successful completion of this function returns VI_SUCCESS.

For more information see:

[Opening an Instrument Session - Visual C Example](#)

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



[Alphabetical Reference](#)

[Hierarchical Reference](#)

Closing an Instrument Session

Sessions (instrumentHandle) opened with the Initialize ([hp816x_init](#)) function are closed with the function:

```
hp816x_close (ViSession instrumentHandle);
```

When no further communication with an instrument is required, the session must be explicitly closed using hp816x_close() function. VISA does not remove sessions unless they are explicitly closed. Closing the instrument session frees all data structures and system resources allocated to that session.

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



VISA Data Types and Selected Constant Definitions

The driver functions use VISA data types. VISA data types are identified by the ‘Vi’ prefix in the data type name (e.g. ViInt16, ViUInt16, ViChar). The *visatype.h* file contains a complete listing of the VISA data types, function call casts and some of the common constants. The following is a partial list of the type definitions and constant definitions in *visatype.h*.

Completion and Error Codes

VI_SUCCESS A long equal to zero

VI_ERROR A long equal to hex 0x80000000

Other VISA Definitions

The size of the following three symbols is operating system dependent and is equal to the default integer size.

VI_NULL An integer equal to 0

VI_TRUE An integer equal to 1

VI_FALSE An integer equal to 0

Variable types

ViUInt32 A 32-bit unsigned integer

ViPUInt32 A pointer to a 32-bit unsigned integer

ViAUInt32 An array of 32-bit unsigned integers

ViInt32 A 32-bit signed integer

ViPInt32 A pointer to a 32-bit signed integer

ViAInt32 An array of 32-bit signed integers

ViUInt16 A 16-bit unsigned integer

ViPUInt16 A pointer to a 16-bit unsigned integer

ViAUInt16 An array of 16-bit unsigned integers

ViInt16 A 16-bit signed integer

ViPInt16 A pointer to a 16-bit signed integer

ViAInt16 An array of 16-bit signed integers

ViUInt8 A unsigned byte

ViPUInt8 A pointer to an unsigned byte

ViAUInt8 An arrays of unsigned bytes

ViInt8 A 8-bit signed integer

ViPInt8 A pointer to a 8-bit signed integer

ViAInt8 An array of 8-bit signed

ViChar A character

ViPChar A pointer to a character

ViAChar An array of characters

ViByte A byte

ViPByte A pointer to a byte

ViAByte An array of bytes

ViAddr A Void pointer

ViPAddr A pointer to a void pointer

ViAAddr An array of void pointers

ViReal32 A 32-bit floating point

ViPReal32 A pointer to a 32-bit floating point

ViAReal32 An array of 32-bit floating points

ViReal64 A 64-bit floating point

ViPReal64 A pointer a 64-bit floating point

ViAReal64 A pointer to an array of 64-bit floating points

ViBuf A pointer to an array of bytes

ViPBuf A pointer to an array of bytes

ViABuf An array of byte pointers

ViString An array of characters

ViPString A pointer to an array of characters

ViAString An array of ViStrings

ViRsrc A ViString

ViPRsrc A pointer to a ViString

ViARsrc An array of ViStrings

ViBoolean A 16-bit unsigned boolean value

ViPBoolean A pointer to a 16-bit unsigned boolean

ViABoolean An array of 16-bit unsigned booleans

ViStatus A 32-bit integer status value

ViPStatus A pointer to a 32-bit integer status value

ViAStatus An array of 32-bit integer status values

ViVersion A 32-bit unsigned integer version number

ViPVersion A pointer to a 32-bit unsigned integer version number

ViAVersion An array of 32-bit unsigned integer version numbers

ViObject A 32-bit unsigned integer object

ViPObject A pointer to a 32-bit unsigned integer object

ViAOBJECT An array of 32-bit unsigned objects

ViSession A 32-bit unsigned integer object

ViPSession A pointer to a 32-bit unsigned integer object

ViASession An array of 32-bit unsigned objects

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



Error Handling

Events and errors within instrument control program can be detected by polling the instrument. Most of the [example programs](#) poll (query) the instrument after each function to determine if an error or other event has occurred. Polling is used in application development environments (ADEs) that do not support asynchronous activities where callbacks can be used. The example programs set up and use polling as shown below.

1. Declare a variable to contain the function completion code.

```
ViStatus errStatus;
```

Every driver function returns the completion code ViStatus. If the function executes with no I/O errors, driver errors, or instrument errors, ViStatus is 0 (VI_SUCCESS). If an error occurs, ViStatus is a negative error code. Warnings are positive error codes, and indicate the operation succeeded but special conditions exist.

2. Enable automatic instrument error checking following each function call.

```
hp816x_errorQueryDetect(instrumentHandle, VI_TRUE);
```

When enabled, the driver queries the instrument for an error condition before returning from the function.

If an error occurred, errStatus (Step 1) will contain a code indicating that an error was detected (hp816x_INSTR_ERROR_DETECTED).

3. Check for an error (or event) after each function.

```
errStatus = hp816x_cmd(instrumentHandle, "MEAS:FREQ");
check(instrumentHandle, errStatus);
```

After the function executes, errStatus contains the completion code. The completion code and instrument ID are passed to an error checking routine. In the above statement, the routine is called 'check'.

4. Create a routine to respond to the error or event. The following routine is used by the example programs to read errors.

```
void check (ViSession instrumentHandle, ViStatus errStatus)
{
    /* variables for error code and message */
```

```

ViInt32 inst_err;
ViChar err_message[256];

/* VI_SUCCESS is 0 and is defined in VISATYPE.h */
if(VI_SUCCESS > errStatus)
{
/* send a device clear - to ensure communication with the instrument */
hp816x_dcl(instrumentHandle);

/* hp816x_INSTR_ERROR_DETECTED defined in hp816x.h */
if(hp816x_INSTR_ERROR_DETECTED == errStatus)

{
/* query the instrument for the error */
hp816x_error_query(instrumentHandle, &inst_err, err_message);

/* display the error */
printf("Instrument Error : %ld, %s\n", inst_err, err_message);
}
else /* driver error */
{
/* get the driver error message */
hp816x_error_message(instrumentHandle, errStatus, err_message);

/* display the error */
printf("Driver Error : %ld, %s\n", errStatus, err_message);

}
/* optionally reset the instrument, close the instrument handle */
hp816x_reset(instrumentHandle);
hp816x_close(instrumentHandle);
exit(1);
}
return;
}

```

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



[Alphabetical Reference](#)

[Hierarchical Reference](#)

Introduction to Programming

To identify the functions you will need to write your instrument control program, select:

[Selecting Functions](#)

To install some typical programs, select:

[Example Programs](#)

For information on how to create programs and use this driver in a number of common programming environments, select:

[Creating, Compiling and Linking 32-bit Programs Using Microsoft Visual C++ Version 4.2](#)

[Creating and Running Visual BASIC 4.0 Applications](#)

These sections explain how to begin using the Keysight Technologies 816x VXIplug&play instrument driver with these products.

[Getting Started with Keysight VEE](#)

[Getting Started with LabView](#)

[Getting Started with LabWindows](#)

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



Selecting Functions

The help file's function reference lists the Keysight Technologies 816x VXIplug&play instrument driver functions alphabetically and hierarchically. There are nine categories of functions based on their intended use. The categories are:

- [Utility Functions](#)
- [Mainframe Specific Functions](#)
- [Attenuator Specific Functions](#)
- [Power Sensor Specific Functions](#)
- [Fixed Laser Source Specific Functions](#)
- [Switch Specific Functions](#)
- [Return Loss Module Specific Functions](#)
- [Tunable Laser Source Specific Functions](#)
- [Applications](#)

For details of which functions are in which category, see the Function Reference (Hierarchical).

The example programs in this helpfile demonstrate most of the functions listed.

Utility Functions

Utility functions perform a variety of standard tasks. For example, Some of the Keysight Technologies 816x VXIplug&play instrument driver's utility functions assist error handling, managing instrument ids, and unit conversion.

Mainframe and Module Specific Functions

Typically, these functions combine multiple SCPI commands into a single, functional operation. They are designed to allow quick and easy access to common instrument command sequences. The functionality they encapsulate should be applicable to a wide variety of functional tasks. These functions are mainframe specific.

Note that this version of the 816x VXIplug&play instrument driver has been extended to support the stand-alone N774xA optical power meter instruments, which do not require a mainframe. These instruments are supported by both relevant mainframe specific functions and power sensor specific functions, as well as the Multi Frame Lambda Scan application function. The relevant mainframe-specific functions and attenuator functions also support the stand-alone N775xA and N776xA series of attenuator instruments. The N775xA-series attenuators also include power meters that are supported by the basic sensor functions of the PnP driver. These latter power meters do not have the functionality for logging or stability measurement and so are also not used by the Multi Frame Lambda Scan function. The relevant mainframe and switch functions also support the stand-alone N773xA series of switch instruments.



Information for N77 Series Instruments

This version of the 816x VXIplug&play instrument driver has been extended to support the N773xA, N774xA optical power meter and the N775xA and N776xA Optical Attenuator instruments. These are stand-alone instruments that do not require installation in an 816x mainframe. The SCPI command set has been designed for almost complete programming compatibility with the modular instruments of the 816x family. Please be aware of the following details and limitations, when using this driver.

- The N77-series instruments can be controlled via USB, LAN or GPIB interfaces. Especially for transferring large data volumes, the USB and usually LAN interfaces provide faster performance.
- The identification of the individual ports of these multiport instruments in the SCPI and hp816x functions corresponds to the slot number. So for example the ports of the N7745A can be addressed with the slot numbers 1 to 8. The channel number in the commands and functions is not used and can generally be omitted or set to Channel 1 (index 0).
- The N771x-series and 81950A tunable lasers are not supported by the 816x VXIplug&play driver.
- The N773x-series optical switches are supported by the 816x VXIplug&play functions for switches and relevant functions for the mainframes.
- The N774x-series power meters are supported by the 816x VXIplug&play functions for power meters, relevant functions for the mainframes like for standard trigger configuration, and the Multi Frame Lambda Scan application functions.
- The N776x-series optical attenuators are supported by the 816x VXIplug&play functions for attenuators and the relevant functions for the mainframes.
- The N775x-series optical attenuators and power meters are supported by the 816x VXIplug&play functions for attenuators and the relevant functions for the mainframes and the basic functions for the power meters, excluding the advanced functions using data power logging. The choices of averaging time is also more limited for these power meters.
- The averaging time parameter for the power meters is limited in the 816x VXIplug&play functions to a limited set of discrete values, similar to the modular 816x power meters. This set includes: 1, 2, 5, 10, 25, 100, 200, and 500 μ s, 1, 2, 5, 10, 20, 50, 100, 200, 500 ms, and 1, 2, 5, and 10s.
- The maximum number of logging data points that can be acquired is limited by the 816x VXIplug&play data acquisition functions to 100,000 per port.
- The shortest averaging time used by the Multi Frame Lambda Scan application is 25 μ s.
- For the Multi Frame Lambda Scan application, the N774xA instruments can be included simply by registering the instrument as a mainframe, after the first

mainframe with the tunable laser has been registered.

Further details to available functions are found under the categories below:

[Utility Functions](#)

[Mainframe Specific Functions](#)

[Power Sensor Specific Functions](#)

[Applications](#)

[Alphabetical Reference](#)

[Hierarchical Reference](#)

Instrument Addressing

The 8163A/B Lightwave Multimeter, the 8164A/B Lightwave Measurement System and the 8166A/B Lightwave Multichannel System use a GP-IB interface. Later models of the 8163B and 8164B also have a LAN interface. The N773xA Optical Switch, N774xA Optical Power Meter and N775x and N776x Optical Attenuator instruments have USB, LAN and GPIB interfaces. When opening an instrument session using the [`hp816x_init\(\)`](#) function, an instrument address must be specified for the function's *ViRsrc resourceName* parameter.

When addressing the instrument over the GP-IB interface the instrument address is based on the logical address and on the communication interface between the computer/controller and the instrument. The default GP-IB address of your instrument is 20.

For more information see:

[Opening an Instrument Session - Visual C Example](#)

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



[Alphabetical Reference](#)

[Hierarchical Reference](#)

Example Programs

To install example programs when you install the Keysight Technologies 816x VXI*plug&play* Instrument Driver:

1. Choose Custom setup from the Setup Type menu.
2. Select Example Files from the Select Components menu.
3. Select one or more of the following programming environments and complete the installation:

Visual Basic

VISA

VEE

LabView

4. If you have chosen the default directory, the examples will be installed in the following directories respectively:

```
<Hard Drive>:\vxipnp\winnt\hp816x\Examples\Vb  
<Hard Drive>:\vxipnp\winnt\hp816x\Visa\VC  
<Hard Drive>:\vxipnp\winnt\hp816x\Examples\VEE  
<Hard Drive>:\vxipnp\winnt\hp816x\Examples\LabView
```

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



Creating, Compiling, and Linking 32-bit Programs Using Microsoft Visual C++ Version 4.2

The example that follows is a step-by-step list of instructions on how to create a 32-bit console application project using Microsoft Visual C++ Version 4.2 or higher. Any project type such as a MFC application or windows application could be created if desired. The name of the program is *SimpleExample*. The example program opens a GP-IB connection to the instrument, querys the instruments ID string, prints the string, closes the GP-IB connection and terminates. It is assumed that the VISA I/O configuration has already been successfully completed.

To create a new project:

Select *File | New | Project Workspace*

Click *OK*

Select the project type, for this example we will build a simple console applications, but any other project type could be used:

Select Console Application as the Type

Enter a name for the project, for example *SimpleExample*

Change the location the project will be created in if desired.

Click *Create*

Create the program source file:

Select *File | New | Text File*

Enter the following program into the text file window:

```
#include <stdio.h>
#include "hp816x.h"
main()
{
    ViStatus vistatReturn;
    ViSession instrumentHandle;
```

```

ViChar viszResponse[256];

// open session, no verification, no reset
if ((vistatReturn = hp816x\_init("GPIB::20::INSTR",
    VI_FALSE, VI_FALSE, &instrumentHandle)) != VI_SUCCESS)
{
    // connection failed
    printf("Connection Failed!\n");
    return vistatReturn;
}
// query instrument ID string
vistatReturn = hp816x_cmdString_Q(instrumentHandle,"*IDN?",
    256,viszResponse);
printf("The instrument is: %s",viszResponse);
// close session
vistatReturn = hp816x_close(instrumentHandle);
printf("Program complete.\n");
return 0;
}

```

Save the file:

Select *File | Save*

Enter *SimpleExample.cpp* for the file name.

Click *Save*

Insert the source file into the project:

Select *Insert | Files into Project...*

Click on *SimpleExample.cpp*

Click *Add*

Set the include file search path:

Select *Tools | Options...*

Select the *Directory* tab

Select *Show Directories for Include Files*

Scroll through the *Directory* list looking for the directory that the *hp816x.h* file is in

If the *hp816x.h* path isn't already in the list add it to the bottom of the list

Click *OK*

Insert the hp816x library into the project link libraries:

Select *Build | Settings | Link | General*
Insert *hp816x.lib* in object/libraries modules edit box.
Click *OK*

Set the library search path:

Select *Tools | Options...*
Select the *Directory* tab
Select *Show Directories for Library Files*
Scroll through the *Directory* list looking for the directory that the *hp816x.lib* file is in
If the *hp816x.lib* path isn't already in the list add it to the bottom of the list
Click *OK*

Compile the project:

Click *Build | Build SimpleExample.exe*

Run the program

Select *Build | Debug | Go*

You may want to put a break point on the last line of the program to pause the program while the output window is still visible. Alternatively open a DOS command window, switch to the directory that contains the executable file (SimpleExample/Debug) and run the program from the command line.

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



Opening an Instrument Session - Visual C Example

```
/* The following example demonstrates how to establish communication using
VXIplug&play driver calls. */
#include <stdio.h>
#include <conio.h>

/*header of the vxipnp driver*/
#include <hp816x.h>

/*GPIB address 20 on local controller 0*/
#define INSTR_ADDRESS "GPIB0::20::INSTR"

/*function prototypes*/
void check_err (ViSession vi, ViStatus errStatus);

int main (void )

{
    ViStatus err_status;
    ViSession iHandle;
    ViChar driver_rev[256];
    ViChar instr_rev[256];
    ViBoolean do_reset = VI_TRUE;

    /* Initialize the instrument. Note that this function will always verify */
    /* that the instrument specified is an Keysight Technologies 816x Series instrument and will send a reset*/
    /* to the instrument if do_reset=VI_TRUE. */

    err_status = hp816x\_init(INSTR_ADDRESS , VI_FALSE, do_reset, &iHandle);
    if(err_status < VI_SUCCESS )
    {
        check_err(iHandle, err_status);
        return -1;
    }

    err_status = hp816x\_revision\_Q(iHandle, driver_rev, instr_rev);
    if(err_status < VI_SUCCESS )
```

```

{
check_err(iHandle, err_status);
return -1;
}
else printf ("Driver Revision:%s\nFirmware Revision:%s\n",driver_rev, instr_rev);
/* wait till a key is pressed */
while(!kbhit());
hp816x_close(iHandle);
return 0;
}

/*****************************************/
void check_err (ViSession vi, ViStatus errStatus)
/*****************************************/

{
ViInt32 inst_err;
ViChar err_message[256];

if(VI_SUCCESS > errStatus)
{
/*check if an instrument error occured*/
if(hp816x_INSTR_ERROR_DETECTED == errStatus)
{
/*get a readable message*/
hp816x_error_query(vi, &inst_err, err_message);
printf("Instrument Error : %lx, %s\n",inst_err, err_message);
}
else /*query the driver to get a readable message*/
{
hp816x_error_message(vi, errStatus, err_message);
printf("Driver Error :%lx, %s\n", errStatus, err_message);
}
}

return;
}

```

© Copyright 1999-2015 Keysight Technologies. All rights reserved.
E0615, Jun 2015



Creating and Running Visual BASIC Applications

Typically Visual Basic (VB) programs are created as event-driven applications. The code representing the main body of the program is stored in a .FRM file and executes when a VB form receives an event (i.e. a button click). The example that follows is a step-by-step list of instructions on how to create a program using Microsoft Visual Basic Version 5.0. The name of the program is Project1. When the user clicks the Run button the program opens a GP-IB connection to the instrument, queries the instruments ID string, prints the string, closes the GP-IB connection and terminates. It is assumed that the VISA I/O configuration has already been successfully completed.

See Also:

[Visual BASIC Data Types](#)

[Visual BASIC Important Notes for Programmers](#)

To create a new project:

Select *File | New Project Workspace*

Click *OK*

Add the [*hp816x.bas*](#) file to the project::

Select *File | Add File*

Browse to and select the *hp816x.bas* file

Click *Open*

Change the focus to Form1 and insert a button on the form:

Click on *Form1*

Click on the command button on the toolbox

With the mouse point over Form1 hold down the right mouse button a drag to create a rectangle where the button will be placed.

Release the right mouse button and a command button will be placed on Form1

Change the name on the button:

Click on the command button just created.

Change the focus to *Properties-Form1*, with *Command1 CommandButton* visible in the drop down list box

Edit the name of the caption to be *Run*

Press *<Enter>* to enter the change

Open the event handler for *Command1*

Change the focus to the project window

Highlight *Form1*

Click *View Code*

Select *Command1* as the Object

Insert the following code between the Private Sub *Command1_Click()* and End Sub statements:

```
Dim lReturn As Long
Dim iHandle As Long
Dim szResponse As String *256
Dim lLength As Long

' open session, no verification, no reset
lReturn = hp816x_init("GPIB::20::INSTR", VI_FALSE, VI_FALSE, iHandle)
If (lReturn <> VI_SUCCESS) Then
    ' connection failed
    Print "Connection Failed!"
    Return
End If
' query instrument ID string
lLength = 256
lReturn = hp816x_cmdString_Q(iHandle, "*IDN?", lLength, szResponse)
Print "The instrument is: ", szResponse

' close session
lReturn = hp816x_close(iHandle)
Print "Program complete."
```

Save the project

Select *File | Save Project*

Navigate to the desire directory to save the project in.

Click **Save** to save the form.

Click **Save** to save the project.

Run the program:

Select *Run | Start*

Execute the command handler:

Click *Run*

The results will be printed on Form1.

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



Visual BASIC Data Types

The Visual BASIC data types for VXIplug&play functions are contained in the driver's Visual BASIC header (.BAS) file (e.g. hp816x.bas). In a standard this file is located in the directory:

```
c:\vxipnp\win95\include for Win95  
c:\vxipnp\win98\include for Win98  
c:\vxipnp\winnt\include for WinNT  
c:\Program Files\VXIplnp\WinNT\include for Win2000  
c:\Program Files\IVI Foundation\VISA\WinNT\include for WinXP
```

The VISA types and their Visual BASIC equivalents are shown below. You can also determine a function's data types by selecting the Visual BASIC header file from the Project window and then selecting View Code. The data types by function are located in the 'Function Declarations' section of the header file.

VISA Type Visual BASIC Type Declaration

```
ViBoolean  
ViPBoolean  
ViBoolean[ ] Dim As Integer  
ViInt16  
ViPInt16  
ViInt16[ ]
```

```
ViInt32  
ViPInt32  
ViInt32[ ] Dim ... As Long  
ViSession  
ViPSession  
ViStatus
```

```
ViReal64  
ViPReal64 Dim ... As Double  
ViReal64[ ]
```

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



Visual BASIC Important Notes for Programmers

When programming in Visual BASIC note the following:

6. When passing strings to Keysight Technologies 816x VXI*plug&play* instrument driver functions the strings must be allocated with a fixed length. For example to allocate a 256 character string use the statement:

```
Dim szMyString As String *256
```

7. When passing arrays to Keysight Technologies 816x VXI*plug&play* instrument driver functions pass the array by reference by specifying the first element in the array. For example:

```
IMyArray(1)
```

8. Visual BASIC defaults to passing parameters by reference. The `ByVal` keyword is used with numeric types to specify that the parameter is to be passed by value. The `ByVal` keyword is also used with Visual BASIC String parameters to indicate that the string should be converted to a C string that is then passed by reference.

. Visual BASIC uses signed integers only.

10. For boolean operations that affect program control, use the VISA `VI_TRUE` and `VI_FALSE` conditions rather than the True/False states of the Visual BASIC boolean data type. `VI_TRUE` and `VI_FALSE` are available with the `VISATYPE.BAS` header file.

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



Getting Started with Keysight VEE

Keysight Visual Engineering Environment (Keysight VEE) is a visual programming language optimized for instrument control applications. To develop programs in Keysight VEE, you connect graphical "objects" instead of writing lines of code. These programs resemble easy-to-understand block diagrams with lines.

Keysight VEE allows you to leverage your investment in textual languages by integrating with languages such as C, C++, Visual Basic, FORTRAN, Pascal and BASIC. Keysight VEE is used in a wide variety of applications such as test and measurement, data acquisition and monitoring and control. Keysight VEE controls GP-IB, VXI, USB, Serial, GPIO, PC Plug-in and LAN instruments directly over the interfaces or by using instrument drivers.

VEE version 3.2 through VEE 6.0 support drivers that comply with the Windows 95/98, or Windows NT VXIplug&play specification version 3.0 or later. VEE version 7.0 does not support HP-UX or Windows 95. VEE version 8.0 supports Windows XP or Windows 2000, but not Windows 98 or Windows NT. VEE version 8.5 supports Windows Vista, Windows 2000 and Windows XP. In addition, VEE versions 3.2 and greater support the graphical Function Panel interface in VXIplug&play drivers, which provides a function tree of the hierarchy of the driver. In addition VEE automatically calls the "initialize" and "close" functions and performs automatic error checking. Keysight VEE gives you an easy-to-use high-level interface to VXIplug&play drivers while also allowing you low-level control. Our customers say Keysight VEE is the easiest way to call VXIplug&play drivers.

The 32-bit Keysight Technologies 816x VXIplug&play instrument driver can be used with VEE 3.2 and above.

To access the functions of the Keysight Technologies 816x VXIplug&play instrument driver from within VEE, consult the manual Using VXIplug&play Drivers with VEE, which comes with VEE 3.2 or the VEE on-line help.

VEE also supports RS-232 port for interfacing with the instrument. The following needs to be performed before one can use the RS-232 port from the VEE 4.0

In IO menu you have Instrument Manager and within that you have Add Button to help you add a new instrument. Double clicking the add button presents you with a Device Configuration screen wherein you are requested the following

Name of instrument - you can key in any name which you like,

Interface - Select GP-IB even when you wish to use the serial port,

Address of the instrument - you can key in any number as you will only be using one of the serial ports,

Gateway - should read This host.

The Advanced I/O Config button in this screen will allow you to select the Keysight Technologies 816x VXIplug&play instrument driver from a drop down list provided the driver is installed properly. You have to specify the address of the instrument as ASRLx if you are planning to use the COMx port in the machine. You can also select whether Reset and Instrument Name Check should be performed whenever VEE opens the instrument for interaction, from this screen.

You need to save this configuration, after getting back to the Instrument Manager Screen by clicking OK buttons, for VEE to remember the configuration.

For more specific and up-to-date instructions using newer versions of VEE, please consult the VEE on-line help.

E0615, Jun 2015



Getting Started with LabView

The 32-bit Keysight Technologies 816x VXIplug&play instrument driver can be used with LabVIEW 4.0 and above. LabVIEW 4.0 is a 32-bit version of LabVIEW which runs on Windows 95/98 and Windows NT.

To access the functions of the Keysight Technologies 816x VXIplug&play instrument driver from within LabVIEW 4.0, select File from the main menu, then select the *Convert CVI FP File* submenu item. In the file selection dialog box which appears, select *hp816x.fp* and click on the OK button. LabVIEW will create a series of VI's, one per driver function. It will create a file called *hp816x.llb* which contains these VI's. This library of VI's can then be accessed like any other VI library in LabVIEW.

With newer versions of LabView, the wrapper for the driver can be generated with the **Instrument Driver Import Wizard** available from National Instruments.

LabWindows is a trademark of National Instruments Corporation

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



Getting Started with LabWindows

You must use the 32-bit version of the Keysight Technologies 816x VXIplug&play instrument driver with LabVIEW 4.0.

The 32-bit Keysight Technologies 816x VXIplug&play instrument driver can be used with LabWindows 4.0 and above. LabWindows 4.0 is a 32-bit version of LabWindows which runs on Windows 95/98 and Windows NT.

To access the functions of the Keysight Technologies 816x VXIplug&play instrument driver from within LabWindows, select *Instrument* from the main menu, and then select the *LOAD...* submenu item. In the file selection dialog box which appears, select *hp816x.tp* and click on the *OK* button. LabWindows will load the function panel and instrument driver. The driver now appears as a selection on the *Instrument* menu, and can be treated like any LabWindows driver.

LabView is a trademark of National Instruments Corporation

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



Application Functions

These functions combine multiple SCPI commands into a single, functional operation. They are designed to allow quick and easy access to common instrument command sequences for coordinating two or more instruments, especially a tunable laser with one or more power meters for spectral measurements.

These application functions allow you to perform one of the following applications:

- A [Lambda Scan](#) - a spectrum analysis operation where a single 8164A/B, 8163A/B or 8166A/B mainframe, with a back-loadable or 819x0-series compact Tunable Laser module and one or more Power Meters modules installed, performs a wavelength sweep and the Tunable Laser module and Power Meters are coordinated with each other. If more than one tunable laser is installed in the mainframe, only the laser with the lowest slot number (e.g. Slot 0 for the back-loadable laser) can be used.
- A [Multi Frame Lambda Scan](#) - a spectrum analysis operation where an back-loadable tunable laser or 819x0-series compact tunable laser module in one 8164A/B, 8163A/B or 8166A/B mainframe performs a wavelength sweep and is coordinated with Power Meters that are installed in the same or other mainframes. The stand-alone N7744A and N7745A Optical Power Meter instruments are also supported in this version. These additional mainframes or instruments must be connected to the GPIB bus or other interface and have their Input Trigger Connector connected to the Output Trigger Connector of the mainframe with the tunable laser. If more than one tunable laser is installed in the mainframe, only the laser with the lowest slot number (e.g. Slot 0 for the back-loadable laser) can be used. If two lasers are present in separate mainframes, the mainframe to be used for the laser can be selected by the program.

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

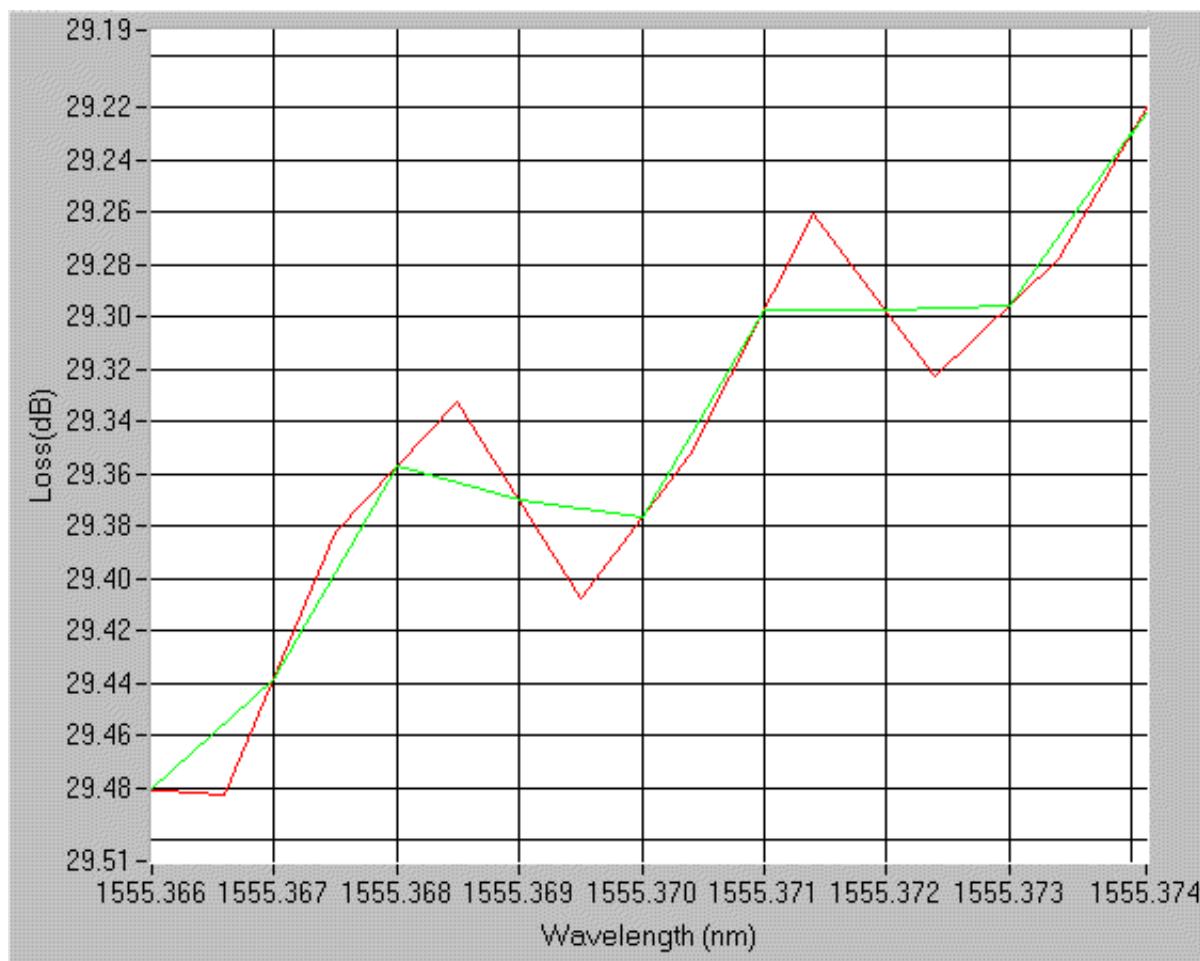
E0615, Jun 2015



Equally Spaced Datapoints

A linear interpolation is performed on all wavelength and power data for the Lambda Scan Application and is optional for the Multi Frame Lambda Scan Application.

The advantage of spacing all measurements equally is that presenting results through use of a spreadsheet is greatly simplified. The operation returns one wavelength array and a power array for each power meter channel.

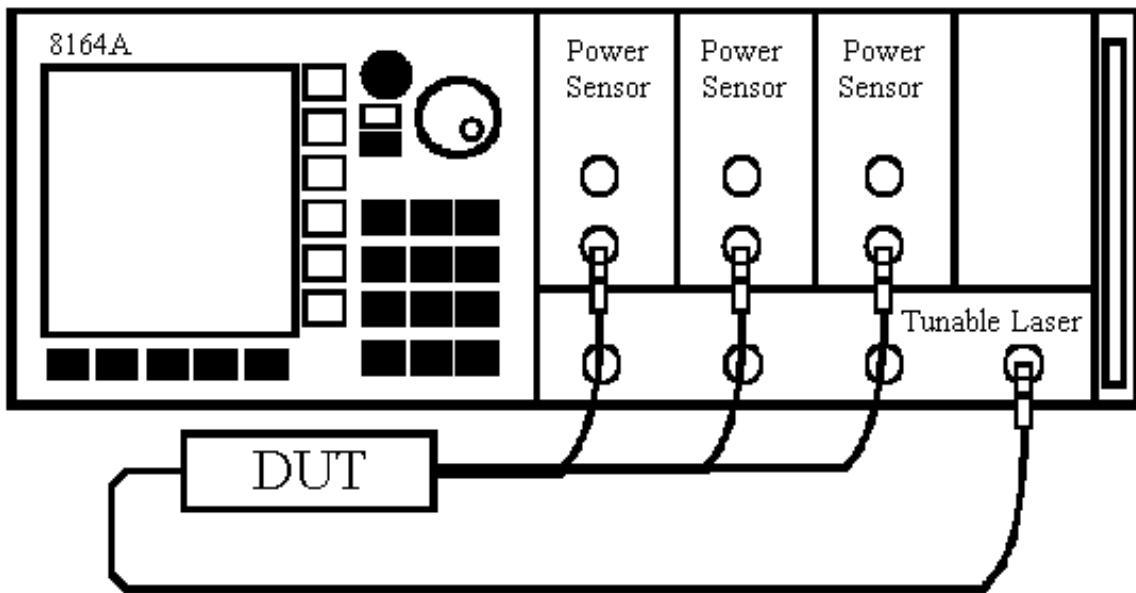


The disadvantage of using equally spaced datapoints is that the linear interpolation is analogous to the use of a low pass filter. The figure above shows two curves: the original curve, in red, as measured directly by a Power Meter and the interpolated curve, in green, which is the result of the linear interpolation. Interpolation will always tend to produce a smoother curve by rounding off any peaks in the curve.

Equally Spaced Datapoints is enabled by default for the Multi Frame Lambda Scan Application but you may choose not to use it, see the Equally Spaced Datapoints ([hp816x_returnEquidistantData](#)) function.

▲

Lambda Scan



A Lambda Scan Operation performs a wavelength sweep where the Tunable Laser module and Power Sensors installed in the 8164A/B Lightwave Measurement System are coordinated with each other. When an 819x0-series compact tunable laser is used, the modules may also be installed in an 8163A/B or 8166A/B mainframe.

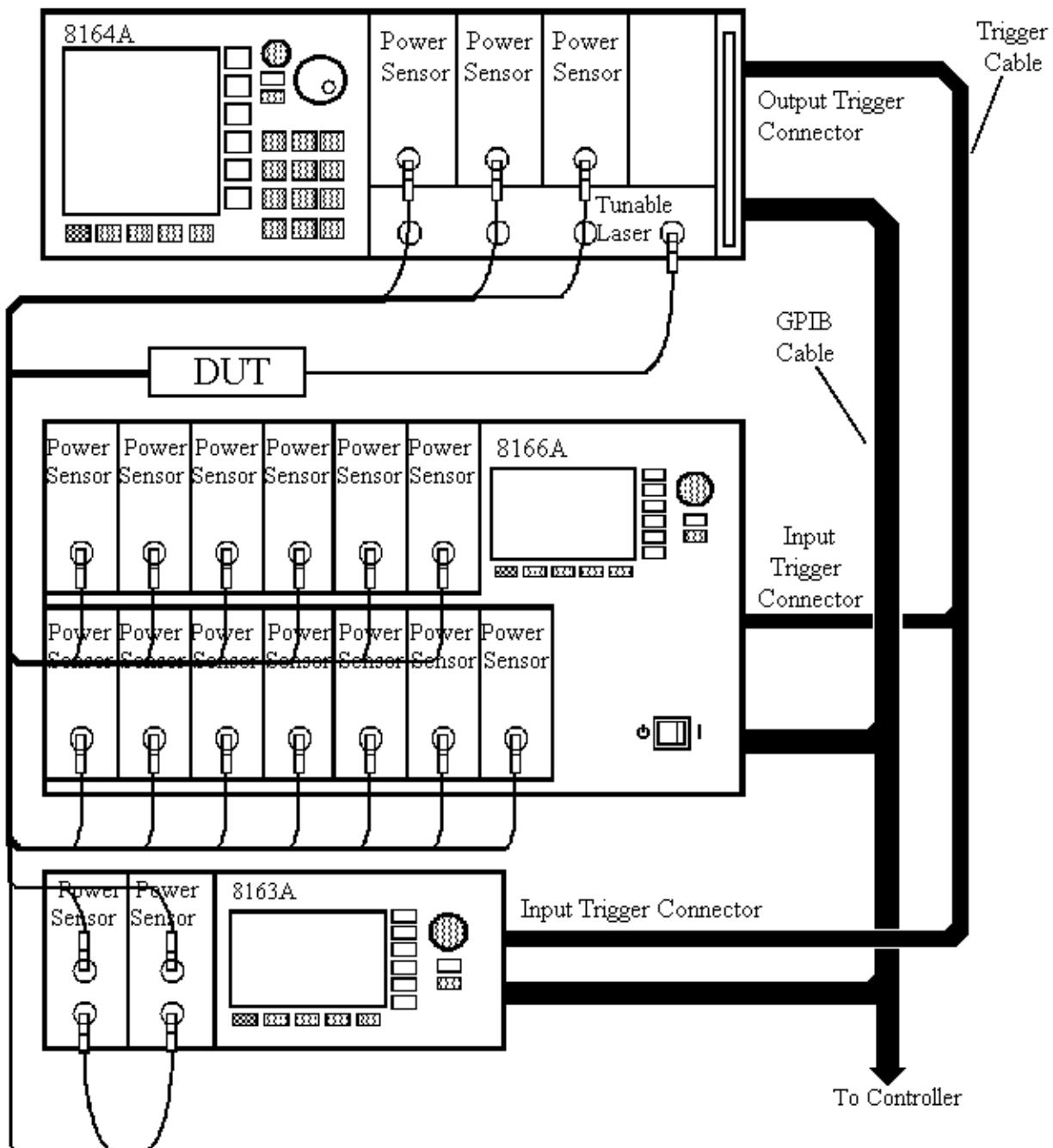
[Equally Spaced Datapoints](#) is enabled as part of the Lambda Scan operation.

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



Multi Frame Lambda Scan



A Multi Frame Lambda Scan Operation coordinates a Lambda Scan for multiple mainframe instruments (up to 150 mainframes) including the stand-alone N774xA optical power meter instruments. All mainframes must be connected to the GPIB, LAN, or USB interface and have their Input Trigger Connector connected to the Output Trigger Connector of the

8164A/B Lightwave Measurement System mainframe containing the tunable laser. If an 819x0-series tunable laser is used, then it may also be installed in an 8163A/B or 8166A/B mainframe.

[Equally Spaced Datapoints](#) is enabled by default for the Multi Frame Lambda Scan Application but you may choose not to use it, see the Equally Spaced Datapoints ([hp816x_returnEquidistantData](#)) function.

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



[Alphabetical Reference](#)

[Hierarchical Reference](#)

Example Programs

To install example programs when you install the Keysight Technologies 816x VXI*plug&play* Instrument Driver:

1. Choose Custom setup from the Setup Type menu.
2. Select Example Files from the Select Components menu.
3. Select one or more of the following programming environments and complete the installation:

Visual Basic

VISA

VEE

LabView

4. If you have chosen the default directory, the examples will be installed in the following directories respectively:

<Hard Drive>:\vxipnp\winnt\hp816x\Examples\Vb

<Hard Drive>:\vxipnp\winnt\hp816x\Visa\VC

<Hard Drive>:\vxipnp\winnt\hp816x\Examples\VEE

<Hard Drive>:\vxipnp\winnt\hp816x\Examples\LabView

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_init

Syntax:

```
ViStatus _VI_FUNC hp816x_init(ViRsrc resourceName, ViBoolean /DQuery,
ViBoolean reset, ViPSession ihandle);
```

The Initialize function (hp816x_init) connects the instrument to the VISA driver and initializes VISA driver resources.

The function identifies the instrument and sets up an internal parameter structure that reads data from the instrument, for example, the current configuration.

Changing the instrument configuration without calling the Initialize function (hp816x_init) again will lead to unpredictable results. For example, inserting a different module changes the instrument configuration, so the driver may not recognise the new module if you do not call the Initialize function again.

<u>Parameter</u>	<u>Description</u>
<i>resourceName</i>	The Resource Name represents the connection type and address (ASRL or GPIB). For example, "GPIBO::20::INSTR", where GPIB0 identifies the GPIB board 0 (installed in your computer), 20 is the mainframe's GPIB address, and INSTR is a suffix needed by the VISA driver. If you use serial interface, make sure the parameters of the instrument match the COM ports parameter. Use the control panel (icon PORTS) to adjust the serial interface parameters.
	Data Type: ViRsrc Input/Output: IN
<i>IDQuery</i>	Use the ID Query control, a boolean input, to perform an identification check, by choosing On (VI_TRUE). This instrument series ignores this parameter. The identification check is always executed.
	Data Type: ViBoolean Input/Output: IN

Values:

Name	Value	Description
VI_TRUE	(1)	On
VI_FALSE	(0)	Off

reset

Use the Reset control, a boolean input, to reset the instrument, by choosing On (VI_TRUE).

Data Type: ViBoolean

Input/Output: IN

Values:

Name	Value	Description
VI_TRUE	(1)	On
VI_FALSE	(0)	Off

ihandle

This output, *ihandle*, returns the session handle. A new session handle is created every time the mainframe is initialized.

Data Type: ViPSession

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)"

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_getHandle

Syntax:

```
ViStatus _Status hp816x_getHandle(ViSession IHandle, ViSession *  
Instrument_Handle );
```

The Get Session Handle function (hp816x_getHandle) returns the handle for the hp816x driver session.

This function allows, for example, Keysight VEE users to pass the session handle to other functions, such as those provided by the Keysight Photonic Foundation Library (pfl).

<u>Parameter</u>	<u>Description</u>
<i>IHandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized.

For LabView Programmers:

This input terminal must be connected to the session handle, *iHandle*, that is returned from the Initialize Mainframe function (hp816x_init).

Data Type: ViSession

Input/Output: IN

Instrument_Handle

Returns the session handle. Use this handle as an input parameter in subsequent calls.

Passed by reference

Data Type: ViSession *

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine

["hp816x_error_message"](#)

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_revision_Q

Syntax:

```
ViStatus _VI_FUNC hp816x_revision_Q(ViSession ihandle,  
ViChar* driverRevision[ ], ViChar* firmwareRevision[ ]);
```

The Revision Query function (hp816x_revision_Q) returns the revision number of the instrument driver and of the instrument firmware.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized. For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).
 <i>driverRevision</i>	Data Type: ViSession Input/Output: IN
 <i>firmwareRevision</i>	Driver Revision returns the revision number of the instrument driver. Data Type: ViChar[] Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)"

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_self_test

Syntax:

```
ViStatus _VI_FUNC hp816x_self_test(ViSession ihandle, ViPInt16 self-testResult, ViChar self-testMessage[ ]);
```

The Self Test function (hp816x_self_test) makes the instrument perform a self-test on the mainframe and all slots. The final result is returned as Self-Test Message.

If the self-test fails, the instrument returns an error code as the Self-Test Result. Look up the *TST? command in your instrument's Programming Guide for an explanation of the Self-Test Result.

No further commands are allowed while the test is running. After the self-test the instrument is returned to the setting that was active at the time the self-test query was processed.

Remark: This function is equivalent to the Mainframe Self-Test function ([hp816x_mainframeSelftest](#)).

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized.

For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function ([hp816x_init](#)).

Data Type: ViSession
Input/Output: IN

self-testResult

Self-Test Result returns the error code, this is an integer. A value of zero indicates no errors.

Look up the *TST? command in your instrument's Programming Guide for an explanation of the Self-Test Result.

Data Type: ViPInt16
Input/Output: OUT

self-testMessage

Self-Test Message returns the self-test status message string. The maximum message length is 256 characters.

Data Type: ViChar[]

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)"

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_close

Syntax:

ViStatus _VI_FUNC hp816x_close(ViSession *ihandle*);

The Close (hp816x_close) function terminates the software connection to the instrument and deallocates system resources.

It is advisable to use this function when the program has finished using the instrument.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized.
	For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).
	Data Type: ViSession Input/Output: IN

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)"

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_timeOut

Syntax:

```
ViStatus _VI_FUNC hp816x_timeOut(ViSession ihandle, ViInt32 timeout);
```

The Timeout function (hp816x_timeOut) sets the driver timeout value in milliseconds (ms). This value specifies the time a VXI/PnP driver function will wait till a timeout error occurs.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized.
<i>timeout</i>	For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).
	Data Type: ViSession Input/Output: IN
<i>timeout</i>	This input specifies the time in milliseconds a function will wait before a timeout error is flagged.
	Data Type: ViInt32 Input/Output: IN

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "[hp816x_error_message](#)"



hp816x_timeOut_Q

Syntax:

```
ViStatus _VI_FUNC hp816x_timeOut_Q(ViSession ihandle, ViPInt32 timeout);
```

The Query Timeout function (hp816x_timeOut_Q) returns the timeout value for driver I/O transactions in milliseconds. Use the Timeout function to set the timeout value. The default value is 20 seconds (20000 milliseconds).

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized.
<i>timeout</i>	For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).
	Data Type: ViSession Input/Output: IN
<i>timeout</i>	Timeout returns the minimum timeout period that the driver is set to in milliseconds (ms).
	Data Type: ViPInt32 Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "[hp816x_error_message](#)"



hp816x_getInstrumentId_Q

Syntax:

```
ViStatus _VI_FUNC hp816x_getInstrumentId_Q(ViString busAddress,  
ViChar/DNString[ ]);
```

The Get Instrument ID function (hp816x_getInstrumentId_Q) returns the IDN string of an instrument.

For example, the IDN string might include the instrument's manufacturer, model, serial number, and firmware revision.

<u>Parameter</u>	<u>Description</u>
<i>busAddress</i>	Use Bus Address to enter the bus address of the instrument. Data Type: ViString Input/Output: IN
<i>IDNString</i>	IDN String returns the IDN string of the intrument. Data Type: ViChar[] Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)"

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_listVisa_Q

Syntax:

```
ViStatus _VI_FUNC hp816x_listVisa_Q(ViInt32 interface, ViBoolean select,
ViPInt32 numberOfDevices, ViChar listofAddresses[ ]);
```

The List Visa Addresses (hp816x_listVisa_Q) function returns a comma separated list of VISA interface addresses.

You can use these addresses to initialize the hp816x VXIplug&play Instrument Driver.

<u>Parameter</u>	<u>Description</u>
<i>interface</i>	Use the Interface control to select one of following: GPIB (hp816x_INTF_GPIB), the function returns a list of the GPIB addresses of all instruments that are connected to the controller. These addresses are returned in the form "GBIBxx::YY::INSTR", where xx is the board number and YY is the GPIB address. Serial (hp816x_INTF_ASRL), the function returns a list of the Serial addresses. These addresses are returned in the form "ASRLxx::INSTR", where xx is the serial port number. All (hp816x_INTF_ALL), the function returns a list of all interface types available for visa, that is, VXI, GPIB and Serial addresses.

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_INTF_GPIB	0	GPIB
hp816x_INTF_ASRL	1	Serial
hp816x_INTF_ALL	2	All

select

Use the Select control to choose to return the following addresses:

816x (hp816x_SEL_816X), only 816x series instruments or

All (hp816x_SEL_ALL), all instruments connected to the controller.

Remark:

If you select 816x, an instrument ID query (*IDN) is sent to each instrument. This ensures that the instrument is attached and powered on.

Data Type: ViBoolean

Input/Output: IN

Values:

Name	Value	Description
hp816x_SEL_816X	1	816x
hp816x_SEL_ALL	0	All

numberOfDevices

Number of Devices returns the number of VISA addresses found.

Data Type: ViPInt32

Input/Output: OUT

listofAddresses

List of Addresses returns a list of VISA addresses separated by commas.

Data Type: ViChar[]

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine

["hp816x_error_message"](#)

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_setBaudrate

Syntax:

```
ViStatus _VI_FUNC hp816x_setBaudrate(ViRsrc interfaceIdentifier, ViUInt32 baudrate);
```

The Set Baudrate function (hp816x_setBaudrate) sets the baudrate of the instrument serial port.

Remark: This function does not needs an instrument handle because getting a handle implies an init call, which requires communication to be running. This function cannot be used with VEE.

Parameter

interfaceIdentifier

Description

Use the Interface Identifier control to specify the communication port for which the baudrate should be set. The string you pass must be of the form ASRLx::INSTR where x is the port number (1,...).

Data Type: ViRsrc

Input/Output: IN

baudrate

Use the Baudrate control to choose a baudrate.

Data Type: ViUInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_BAUD_9600	0	9600
hp816x_BAUD_19200	1	19200
hp816x_BAUD_38400	2	38400

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)"

E0615, Jun 2015



hp816x_dcl

Syntax:

ViStatus _VI_FUNC hp816x_dcl(ViSession *ihandle*);

The Device Clear function (hp816x_dcl) clears the following:

1. The Instrument Error Queue
2. The Standard Event Status Register
3. The Status Byte

See the your instrument's Programming Guide for further information. This function performs the same purpose as the *CLS GPIB command.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized. For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).
	Data Type: ViSession Input/Output: IN

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)"



hp816x_error_query

Syntax:

```
ViStatus _VI_FUNC hp816x_error_query(ViSession ihandle, ViPInt32  
instrumentErrorCode, ViCharerrorMessage[ ]);
```

The Instrument Error Query function (hp816x_error_query) returns the error number and corresponding error message from the error queue of the instrument. This type of error is caused by the instrument or one of the installed modules.

See your instrument's Programming Guide for a listing of the instrument error numbers and messages.

Instrument errors can occur because you have placed the instrument in a bad state. For example, if you send an invalid sequence of coupled commands.

Automatic error detection can be accomplished by using the Automatic Error Detection function ([\(hp816x_errorQueryDetect\)](#)).

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function ((hp816x_init)). A new session handle is created every time the mainframe is initialized. For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function ((hp816x_init)).

Data Type: ViSession

Input/Output: IN

instrumentErrorCode

Error Code returns the instrument error code.

See your instrument's Programming Guide for a listing of the instrument error numbers and messages.

Data Type: ViPInt32

Input/Output: OUT

errorMessage

Error Message returns an error message string from the instrument.

Data Type: ViChar[]

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)"

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_error_message

Syntax:

```
ViStatus _VI_FUNC hp816x_error_message(ViSession ihandle, ViStatus  
errorCode, ViString errorMessage);
```

The Error Message function (hp816x_error_message) transforms the error code returned from any instrument driver to a user-readable string.

Connect the Status output of any function to the Error Code input of this function to get information about any instrument or driver errors.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized. For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).
 Data Type: ViSession Input/Output: IN	
 <i>errorCode</i>	Use Error Code to input an error code. If Status for any function returns an error code other than 0 (VI_SUCCESS), input this value as the Error Code.
 Data Type: ViStatus Input/Output: IN	
 <i>errorMessage</i>	Error Message returns the error message string.

Data Type: ViString
Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "hp816x_error_message"

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_errorQueryDetect

Syntax:

```
ViStatus _VI_FUNC hp816x_errorQueryDetect(ViSession ihandle, ViBoolean automaticErrorDetection);
```

The Automatic Error Detection function (hp816x_errorQueryDetect) checks for instrument errors.

After each command the Error Status Register is read and checked for errors.

If you want to increase execution speed, it is recommended to turn Automatic Error Detection OFF. After an application is tested, instrument errors are less likely.

Parameter

ihandle

Description

IHandle is the session handle returned from the Initialize Mainframe function ([hp816x_init](#)). A new session handle is created every time the mainframe is initialized.

For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function ([hp816x_init](#)).

Data Type: ViSession

Input/Output: IN

automaticErrorDetection

Switch Automatic Error Detection to On (VI_TRUE) to automatically check for instrument error.

Switch Automatic Error Detection to Off (VI_FALSE) to disable this function.

Data Type: ViBoolean

Input/Output: IN

Values:

Name

Value

Description

VI_TRUE	(1)	On
VI_FALSE	(0)	Off

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)"

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_dbmToWatt

Syntax:

```
ViStatus _VI_FUNC hp816x_dbmToWatt(ViSession ihandle, ViReal64 dbm,  
ViPReal64 watt);
```

The dBm to Watt function (hp816x_dBmToWatt) converts a power value in dBm into Watts.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized. For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).
	Data Type: ViSession Input/Output: IN
<i>dbm</i>	Use dBm to input a power value in dBm. Data Type: ViReal64 Input/Output: IN
<i>watt</i>	Watt returns a power value in Watts (W) that has been converted from dBm. Data Type: ViPReal64 Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)"

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_wattToDBm

Syntax:

```
ViStatus _VI_FUNC hp816x_WattToDBm(ViSession ihandle, ViPReal64 watt,  
ViReal64 dbm);
```

The Watt to dBm function (hp816x_WattToDBm) converts a power value in Watt into dBm.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized. For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).
	Data Type: ViSession Input/Output: IN
<i>watt</i>	Use Watt to input a power value in Watts (W) Data Type: ViPReal64 Input/Output: IN
<i>dbm</i>	dBm returns a power value in dBm that has been converted from Watts. Data Type: ViReal64 Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)"

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_setDate

Syntax:

```
ViStatus _VI_FUNC hp816x_setDate(ViSession ihandle, ViInt32 year, ViInt32 month, ViInt32 day);
```

The Set Date function (hp816x_setDate) sets the current date of the instrument.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized. For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).
	Data Type: ViSession Input/Output: IN
<i>year</i>	Use Year to set the current year of the instrument. Data Type: ViInt32 Input/Output: IN
<i>month</i>	Use Month to set the current month of the instrument. Data Type: ViInt32 Input/Output: IN
<i>day</i>	

Use Day to set the current day of the instrument.

Data Type: ViInt32

Input/Output: IN

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_getDate

Syntax:

```
ViStatus _VI_FUNC hp816x_getDate(ViSession ihandle, ViPInt32 year,  
ViPInt32 month, ViPInt32 day);
```

The Get Date function (hp816x_getDate) returns the current date of the instrument.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized. For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).
	Data Type: ViSession Input/Output: IN
<i>year</i>	Year returns the current year from the instrument. Data Type: ViPInt32 Input/Output: OUT
<i>month</i>	Month returns the current month from the instrument. Data Type: ViPInt32 Input/Output: OUT
<i>day</i>	

Day returns the current day from the instrument.

Data Type: ViPInt32

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_setTime

Syntax:

```
ViStatus _VI_FUNC hp816x_setTime(ViSession ihandle, ViInt32 hour, ViInt32  
minute, ViInt32 second);
```

The Set Time function (hp816x_setTime) sets the current time of the instrument.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized. For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).
	Data Type: ViSession Input/Output: IN
<i>hour</i>	Use Hour to set the current hour of the instrument. Data Type: ViInt32 Input/Output: IN
<i>minute</i>	Use Minute to set the current minute of the instrument. Data Type: ViInt32 Input/Output: IN
<i>second</i>	

Use Second to set the current second of the instrument.

Data Type: ViInt32

Input/Output: IN

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)"

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_getTime

Syntax:

```
ViStatus _VI_FUNC hp816x_getTime(ViSession ihandle, ViPInt32 hour,  
ViPInt32 minute, ViPInt32 second);
```

The Get Time function (hp816x_getTime) returns the current time of the instrument.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized. For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).
	Data Type: ViSession Input/Output: IN
<i>hour</i>	Hour returns the current hour of the instrument. Data Type: ViPInt32 Input/Output: OUT
<i>minute</i>	Minute returns the current minute of the instrument. Data Type: ViPInt32 Input/Output: OUT
<i>second</i>	

Second returns the current second of the instrument.

Data Type: ViPInt32

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)"

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_cls

Syntax:

ViStatus _VI_FUNC hp816x_cls(ViSession ihandle);

The Clear Error Queue function (hp816x_cls) clears the instrument's error queue. This function performs the same function as the GPIB command, "*CLS".

See your instrument's Programming Guide for more information.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized.
	For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).
	Data Type: ViSession Input/Output: IN

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "[hp816x_error_message](#)"

hp816x_forceTransaction

Syntax:

```
ViStatus _VI_FUNC hp816x_forceTransaction(ViSession ihandle, ViBoolean forceTransaction);
```

The Force Transaction function (hp816x_forceTransaction) enables you to increase execution speed.

If Force Transaction is turned Off, a command is only transmitted to the instrument, if a difference between an actual parameter and a previous setting is noticed.

If Force Transaction is turned On, each parameter setting is transmitted to the instrument, independent of previous settings.

Turning the Force Transaction function Off causes the driver to run faster.

REMARKS The default value is ON. If Force Transactions is turned Off, manual adjustments are not noticed by the driver and can falsify results.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized. For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).
	Data Type: ViSession Input/Output: IN

forceTransaction

If Force Transaction is turned Off (VI_FALSE), a command is only transmitted to the instrument, if a difference between an actual parameter and a previous setting is noticed.

If Force Transaction is turned On (VI_TRUE), each setting of parameters is transmitted to the instrument, independent of previous setting.

REMARKS The default value is On. If Force Transaction is turned Off, manual adjustments are not noticed by the driver and can falsify results.

Data Type: ViBoolean

Input/Output: IN

Values:

Name	Value	Description
VI_TRUE	(1)	On
VI_FALSE	(0)	Off

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_driverLogg

Syntax:

```
ViStatus _VI_FUNC hp816x_driverLogg(ViSession ihandle, ViString filename,
ViBoolean logging, ViBoolean includeReplies);
```

The Enable Driver Logging function (hp816x_driverLogg) logs all commands sent to the instrument to a file in text format, with the exception of data aquisition commands.

If the file exists, the output will be appended.

This log file can become very large, especially if the instrument is polled for some reason.

Parameter

Description

ihandle

IHandle is the session handle returned from the Initialize Mainframe function ([hp816x_init](#)). A new session handle is created every time the mainframe is initialized.

For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function ([hp816x_init](#)).

Data Type: ViSession

Input/Output: IN

filename

Use Filename to enter the name of the file to which all of the commands will be logged.

Data Type: ViString

Input/Output: IN

logging

Switch Logging to Enable (VI_TRUE) to enable driver logging.

Switch Logging to Disable (VI_FALSE) to disable

driver logging.

Data Type: ViBoolean

Input/Output: IN

Values:

Name	Value	Description
VI_TRUE	(1)	Enable
VI_FALSE	(0)	Disable

includeReplies

Use the Include Replies control to decide if the response from the instrument should be also logged.

Switch the Include Replies control to Yes (VI_TRUE) if you want to log the response to commands.

Switch the Include Replies control to No (VI_FALSE) if you do not want to log the response to commands.

Data Type: ViBoolean

Input/Output: IN

Values:

Name	Value	Description
VI_TRUE	(1)	Yes
VI_FALSE	(0)	No

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_cmd

Syntax:

```
ViStatus _VI_FUNC hp816x_cmd(ViSession ihandle,  
ViCharcommandString[ ]);
```

The Send String Command function (hp816x_cmd) sends a GPIB command (conforming to the SCPI standard) to the instrument. It does not look for a response.

See your instrument's Programming Guide for information on GPIB commands.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized. For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).
	Data Type: ViSession Input/Output: IN
<i>commandString</i>	Use Command String to enter a string containing a GPIB command. See your instrument's Programming Guide for information on GPIB commands.
	Data Type: ViChar[] Input/Output: IN

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "[hp816x_error_message](#)"

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_cmdString_Q

Syntax:

```
ViStatus _VI_FUNC hp816x_cmdString_Q(ViSession ihandle,
ViChar*inputQuery[ ], Integer stringSize, ViChar*result[ ]);
```

The Send String Query function (hp816x_cmdString_Q) sends a GPIB query string to the instrument and waits for a response which must be a string (character data).

A minimum of one byte will be read from the instrument and the string will be NULL terminated (so its length can be found with strlen, the String Length C function).

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized.

For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function ([hp816x_init](#)).

Data Type: ViSession
Input/Output: IN

inputQuery

Use Input Query to enter a query string containing a GPIB query. This GPIB query should return a character string. All GPIB queries have a question mark, ?, as their last character.

See your instrument's Programming Guide for information on GPIB commands.

Data Type: ViChar[]
Input/Output: IN

stringSize

String Size sets the size of the character array

that the function uses to store the Result.

Data Type: Integer

Input/Output: IN

Values:

String Size_MIN 2

String Size_MAX 32767

result

Result returns a string from the instrument, that is, the reply to your Input Query.

See your instrument's Programming Guide for information on GPIB commands.

Data Type: ViChar[]

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_cmdInt

Syntax:

```
ViStatus _VI_FUNC hp816x_cmdInt(ViSession ihandle,  
ViChar*integerCommand[ ], ViInt32 integerValue);
```

The Send Integer Command function (hp816x_cmdInt) sends an instrument command which requires one integer parameter.

See your instrument's Programming Guide for information on GPIB commands.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized. For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).
	Data Type: ViSession Input/Output: IN
<i>integerCommand</i>	Use Integer Command to enter a command string containing a GPIB command. The command string should require a single integer parameter as the Integer Value. For example, the GPIB command to set the display brightness to 65 is: disp:brig 65 In this case, enter disp:brig as Integer Command. See your instrument's Programming Guide for information on GPIB commands.
	Data Type: ViChar[]

Input/Output: IN

integerValue

Use Integer Value to enter the integer value you wish to send to the instrument.

For example, the GPIB command to set the display brightness to 65 is: disp brig 65

In this case, enter 65 as Integer Value.

See your instrument's Programming Guide for information on GPIB commands.

Data Type: ViInt32

Input/Output: IN

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_CmdInt32_Q

Syntax:

```
ViStatus _VI_FUNC hp816x_CmdInt32_Q(ViSession ihandle,  
ViChar*integerQuery[], ViPInt32 integerResult);
```

The Send Integer Query function (hp816x_CmdInt32_Q) sends a GPIB query string to the instrument and waits for a response that must be representable as a Vilnt32. A non-numeric instrument response returns zero as the result.

See your instrument's Programming Guide for information on GPIB commands.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized. For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).

Data Type: ViSession

Input/Output: IN

integerQuery

Use Integer Query to enter a query string containing a GPIB query. This GPIB query should return a Vilnt32. All GPIB queries have a question mark, ?, as their last character.

See your instrument's Programming Guide for information on GPIB commands.

Data Type: ViChar[]

Input/Output: IN

integerResult

Integer Result returns a Vilnt32 value from the

instrument, that is, the reply to your Integer Query.

A non-numeric instrument response returns zero as the result.

See your instrument's Programming Guide for information on GPIB commands.

Data Type: ViPInt32

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_cmdReal

Syntax:

```
ViStatus _VI_FUNC hp816x_cmdReal(ViSession ihandle, ViString  
realCommand, ViReal64 realValue);
```

The Send Real Command function (hp816x_cmdReal) sends an instrument command which requires one real parameter.

See your instrument's Programming Guide for information on GPIB commands.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized. For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).
	Data Type: ViSession Input/Output: IN
<i>realCommand</i>	Use Real Command to enter a string containing a GPIB command. The command string should require a single real parameter. For example, the GPIB command to set the averaging time is: sens1:pow:atime 1.234s In this case, enter sens1:pow:atime as Real Command. This command sets the averaging time of the Power Sensor in slot 1. See your instrument's Programming Guide for information on GPIB commands.
	Data Type: ViString Input/Output: IN

realValue

Use Real Value to enter the real value you want to send to the instrument.

For example, the GPIB command to set the averaging time is: sens1:pow:atime 1.234s

In this case, enter 1.234 as Real Value. This sets the averaging time of the Power Sensor in slot 1 to 1.234 seconds.

You cannot use any units, default units are presumed by the function.

See your instrument's Programming Guide for information on GPIB commands.

Data Type: ViReal64

Input/Output: IN

Values:

Real Value_MIN -1.000000E+300

Real Value_MAX 1.000000E+300

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "[hp816x_error_message](#)"

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_cmdReal64_Q

Syntax:

```
ViStatus _VI_FUNC hp816x_cmdReal64_Q(ViSession ihandle, ViString  
realQuery, ViPReal64 realResult);
```

The Send Real Query function (hp816x_cmdReal64_Q) sends a GPIB query string to the instrument and waits for a response that must be representable as a ViReal64.

See your instrument's Programming Guide for information on GPIB commands.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized. For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).
	Data Type: ViSession Input/Output: IN
<i>realQuery</i>	Use Real Query to enter a query string containing a GPIB query. This GPIB query should return a ViReal64 value. The command string cannot exceed 256 bytes in length. All GPIB queries have a question mark, ?, as their last character. See your instrument's Programming Guide for information on GPIB commands.
	Data Type: ViString Input/Output: IN
<i>realResult</i>	Real Result returns a ViReal64 value from the

instrument, that is, the reply to your Real Query.

A non-numeric instrument response returns zero as the result.

See your instrument's Programming Guide for information on GPIB commands.

Data Type: ViPReal64

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



Information for N77 Series Instruments

This version of the 816x VXIplug&play instrument driver has been extended to support the N773xA, N774xA optical power meter and the N775xA and N776xA Optical Attenuator instruments. These are stand-alone instruments that do not require installation in an 816x mainframe. The SCPI command set has been designed for almost complete programming compatibility with the modular instruments of the 816x family. Please be aware of the following details and limitations, when using this driver.

- The N77-series instruments can be controlled via USB, LAN or GPIB interfaces. Especially for transferring large data volumes, the USB and usually LAN interfaces provide faster performance.
- The identification of the individual ports of these multiport instruments in the SCPI and hp816x functions corresponds to the slot number. So for example the ports of the N7745A can be addressed with the slot numbers 1 to 8. The channel number in the commands and functions is not used and can generally be omitted or set to Channel 1 (index 0).
- The N771x-series and 81950A tunable lasers are not supported by the 816x VXIplug&play driver.
- The N773x-series optical switches are supported by the 816x VXIplug&play functions for switches and relevant functions for the mainframes.
- The N774x-series power meters are supported by the 816x VXIplug&play functions for power meters, relevant functions for the mainframes like for standard trigger configuration, and the Multi Frame Lambda Scan application functions.
- The N776x-series optical attenuators are supported by the 816x VXIplug&play functions for attenuators and the relevant functions for the mainframes.
- The N775x-series optical attenuators and power meters are supported by the 816x VXIplug&play functions for attenuators and the relevant functions for the mainframes and the basic functions for the power meters, excluding the advanced functions using data power logging. The choices of averaging time is also more limited for these power meters.
- The averaging time parameter for the power meters is limited in the 816x VXIplug&play functions to a limited set of discrete values, similar to the modular 816x power meters. This set includes: 1, 2, 5, 10, 25, 100, 200, and 500 μ s, 1, 2, 5, 10, 20, 50, 100, 200, 500 ms, and 1, 2, 5, and 10s.
- The maximum number of logging data points that can be acquired is limited by the 816x VXIplug&play data acquisition functions to 100,000 per port.
- The shortest averaging time used by the Multi Frame Lambda Scan application is 25 μ s.
- For the Multi Frame Lambda Scan application, the N774xA instruments can be included simply by registering the instrument as a mainframe, after the first

mainframe with the tunable laser has been registered.

Further details to available functions are found under the categories below:

[Utility Functions](#)

[Mainframe Specific Functions](#)

[Power Sensor Specific Functions](#)

[Applications](#)

hp816x_reset

Syntax:

```
ViStatus _VI_FUNC hp816x_reset(ViSession ihandle);
```

The Reset function (hp816x_reset) resets the instrument and the GPIB bus to their initial state. The parameters of the instrument and its modules are set to their default values.

This function is identical to the "*RST" GPIB command. See your instrument's Programming Guide for information on GPIB commands.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized.
	For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).
	Data Type: ViSession Input/Output: IN

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)"

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_preset

Syntax:

```
ViStatus _VI_FUNC hp816x_preset(ViSession ihandle);
```

The Preset function (hp816x_preset) resets the instrument to its initial state. The parameters of the instrument and its modules are set to their default values.

This function is the same as pressing the preset hardkey on the instrument's front panel.

This function is also identical to the ":syst:preset" GPIB command. See your instrument's Programming Guide for information on GPIB commands.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized.
	For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).
	Data Type: ViSession Input/Output: IN

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "[hp816x_error_message](#)"



hp816x_mainframeSelftest

Syntax:

```
ViStatus _VI_FUNC hp816x_mainframeSelftest(ViSession ihandle, ViPInt16 self-testResult, ViChar self-testMessage[ ]);
```

The Mainframe Self-Test function (hp816x_mainframeSelftest) makes the instrument perform a self-test on the mainframe and all slots. The final result is returned as Self-Test Message.

If the self-test fails, the instrument returns an error code as the Self-Test Result. Look up the *TST? command in your instrument's Programming Guide for an explanation of the Self-Test Result.

No further commands are allowed while the test is running. After the self-test the instrument is returned to the setting that was active at the time the self-test query was processed.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized. For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).

Data Type: ViSession

Input/Output: IN

self-testResult

Self-Test Result returns the error code, this is an integer. A value of zero indicates no errors.

Look up the *TST? command in your instrument's Programming Guide for an explanation of the Self-Test Result.

Data Type: ViPInt16

Input/Output: OUT

self-testMessage

Self-Test Message returns the self-test status message string. The maximum message length is 256 characters.

Data Type: ViChar[]

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)"

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_moduleSelftest

Syntax:

```
ViStatus _VI_FUNC hp816x_moduleSelftest(ViSession ihandle, ViInt32
slotToTest, ViPInt16 result, ViChar* self-testMessage[ ]);
```

The Module Self-Test (hp816x_moduleSelftest) returns the latest selftest results for a module inserted in a given slot.

NOTE: This command does not perform a selftest. Use the Mainframe Self-Test function ([hp816x_mainframeSelftest](#)) to make the instrument perform a self-test on the mainframe and all slots.

Parameter

ihandle

Description

IHandle is the session handle returned from the Initialize Mainframe function ([hp816x_init](#)). A new session handle is created every time the mainframe is initialized.

For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function ([hp816x_init](#)).

Data Type: ViSession

Input/Output: IN

slotToTest

Use the Slot to Test control to choose an installed module.

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_0	0	Slot 0
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3

hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

result

Result returns the error code, this is an integer. A value of zero indicates no errors.

See your instrument's Programming Guide for a listing of the instrument error numbers and messages.

Data Type: ViPInt16

Input/Output: OUT

self-testMessage

Self-Test Message returns the self-test status message string. The maximum message length is 256 characters.

Data Type: ViChar[]

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)"

E0615, Jun 2015



hp816x_revision_query

Syntax:

```
ViStatus _VI_FUNC hp816x_revision_query(ViSession ihandle,
ViChar*driverRevision[ ], ViChar*firmwareRevision[ ]);
```

The revision query function (hp816x_revision_query) returns the revision number of the instrument driver and of the instrument firmware.

Remark: This function is equivalent to the Revision Query function ([hp816x_revision_Q](#)).

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized.

For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function ([hp816x_init](#)).

Data Type: ViSession
Input/Output: IN

driverRevision

Driver Revision returns the revision number of the instrument driver.

Data Type: ViChar[]
Input/Output: OUT

firmwareRevision

Firmware Revision returns the revision number of the instrument firmware.

Data Type: ViChar[]
Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)"

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_opc_Q

Syntax:

```
ViStatus _VI_FUNC hp816x_opc_Q(ViSession iHandle, ViPBoolean  
operationComplete);
```

The Operation Complete function (hp816x_opc_Q) sends the "*OPC?" GPIB command to the instrument and returns 1 when all pending operations are complete.

See your instrument's Programming Guide for more information.

<u>Parameter</u>	<u>Description</u>
<i>iHandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized. For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).
	Data Type: ViSession Input/Output: IN
<i>operationComplete</i>	Operation Complete returns VIStatus when all pending operations are complete. Data Type: ViPBoolean Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "[hp816x_error_message](#)"

■

hp816x_lockUnlockInstrument

Syntax:

```
ViStatus _VI_FUNC hp816x_lockUnlockInstrument(ViSession ihandle,
ViBoolean softlock, ViString password);
```

The Lock-Unlock Instrument function (hp816x_lockUnlockInstrument) locks or unlocks the software lock for high power source modules inserted in the mainframe.

Parameter

ihandle

Description

IHandle is the session handle returned from the Initialize Mainframe function ([hp816x_init](#)). A new session handle is created every time the mainframe is initialized.

For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function ([hp816x_init](#)).

Data Type: ViSession

Input/Output: IN

softlock

If you choose Lock (VI_TRUE), high power source modules will be locked.

If you choose Unlock (VI_FALSE), high power source modules will be unlocked.

Data Type: ViBoolean

Input/Output: IN

Values:

Name	Value	Description
VI_TRUE	(1)	Lock
VI_FALSE	(0)	Unlock

password

Use Password to enter the instrument password which is used to lock or unlock high power source modules.

1234 is your instrument's default password.

Data Type: ViString

Input/Output: IN

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)"

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_getLockState

Syntax:

```
ViStatus _VI_FUNC hp816x_getLockState(ViSession Ihandle, ViBoolean *  
Soft_Lock, ViBoolean * Remote_Interlock);
```

The Get Lock State function (hp816x_getLockState) returns the status of the mainframe's softlock and its remote interlock. If either the softlock or the remote interlock is locked, or both are locked, high-power laser sources cannot be turned on.

<u>Parameter</u>	<u>Description</u>
<i>Ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized.

For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function ([hp816x_init](#)).

Data Type: ViSession
Input/Output: IN

Soft_Lock

Returns the status of the softlock.

If 0 (VI_FALSE) is returned, the softlock is unlocked.

If 1 (VI_TRUE) is returned, the softlock is locked.

Set the status of the softlock using the Lock-Unlock Instrument function ([hp816x_lockUnlockInstrument](#)).

Data Type: ViBoolean
Input/Output: OUT

Remote_Interlock

Returns the status of the remote interlock.

If 0 (VI_FALSE) is returned, the remote interlock is unlocked.

If 1 (VI_TRUE) is returned, the remote interlock is locked.

Data Type: ViBoolean

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_enableDisableDisplay

Syntax:

```
ViStatus _VI_FUNC hp816x_enableDisableDisplay(ViSession ihandle,  
ViBoolean display);
```

The Enable-Disable Display function (hp816x_enableDisableDisplay) turns the display of the instrument On or Off.

Parameter

ihandle

Description

IHandle is the session handle returned from the Initialize Mainframe function ([hp816x_init](#)). A new session handle is created every time the mainframe is initialized.

For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function ([hp816x_init](#)).

Data Type: ViSession

Input/Output: IN

display

If you select On (VI_TRUE), the Enable Disable Display function turns the display of the instrument On.

If you select Off (VI_FALSE), the Enable Disable Display function turns the display of the instrument Off.

Data Type: ViBoolean

Input/Output: IN

Values:

Name	Value	Description
VI_TRUE	(1)	On
VI_FALSE	(0)	Off

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)"

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_getSlotInformation_Q

Syntax:

```
ViStatus _VI_FUNC hp816x_getSlotInformation_Q(ViSession ihandle, ViInt32
arraySize, ViInt32slotInformation[ ]);
```

The Get Slot Information function (hp816x_getSlotInformation_Q) returns information about installed modules. The Slot Information can be used to write applications that are independent of the physical location of the inserted modules.

REMARKS You must assure that the Array Size is properly allocated.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized.
	For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).
	Data Type: ViSession Input/Output: IN
<i>arraySize</i>	Use Array Size to set the size of the memory buffer needed for the function. Depending on your mainframe, you should set Array Size to the following integer value: 8163A/B: 3 8164A/B: 5 8166A/B: 18
	Data Type: ViInt32 Input/Output: IN

slotInformation

Slot Information is both an input and a output.

As an input, allocate Slot Information as a one-dimensional integer array of the following dimensions:

8163A/B: 1 x 3

8164A/B: 1 x 5

8166A/B: 1 x 18

As a return, Slot Information returns a one dimensional array. Each component consists of the array component number and the module type number. The array component number corresponds to the slot number, starting at 0 and ending at the highest slot number. The following numbers define the module type number:

0 The slot is empty hp816x_UNDEF

1 A single-channel Power Sensor
hp816x_SINGLE_SENSOR

2 A dual-channel Power Sensor
hp816x_DUAL_SENSOR

3 A single Laser Source
hp816x_FIXED_SINGLE_SOURCE

4 A dual-wavelength Laser Source
hp816x_FIXED_DUAL_SOURCE

5 A Tunable Laser module
hp816x_TUNABLE_SOURCE

6 A Return Loss Module hp816x_RETURN_LOSS

7 A Return Loss Combo Module
hp816x_RETURN_LOSS_COMBO

For example, if the result below is returned for an 8163A/B, it means that a single-channel Power Sensor Module is installed in slot 1 and a single-

channel Laser Source Module is installed in slot 2
(slot 0 does not exist for the 8163A/B):

0: 0

1: 1

2: 3

Data Type: ViInt32[]

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)"

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_getModuleStatus_Q

Syntax:

```
ViStatus _VI_FUNC hp816x_getModuleStatus_Q(ViSession ihandle,
ViPBoolean statusSummary, ViInt32moduleStatusArray[ ], ViPInt32
maxMessageLength);
```

The Get Module Status function (hp816x_getModuleStatus_Q) checks the questionable status condition register of all modules inserted into the instrument. The questionable status condition is the current status of all modules. See your instrument's Programming Guide for information on the instrument's Status System.

Questionable status information is information about the following module errors:

If excessive power is set by the user, if zeroing failed, if temperature is out of range, if laser protection is switched on, if the module has not settled, if the module is out of specifications, if ARA is recommended, or if the duty cycle is out of range.

REMARKS You must assure that the Array Size is properly allocated.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized. For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).
	Data Type: ViSession Input/Output: IN
<i>statusSummary</i>	If Status Summary equals 0, there are no current questionable events for all inserted modules. If Status Summary equals 1, there is at least one current questionable event for one inserted module. Data Type: ViPBoolean

Input/Output: OUT

moduleStatusArray

Module Status Array is both an input and an output.

As an input, Module Status Array accepts a one-dimensional integer array that determines the size of the Module Status Array output. Allocate a one-dimensional integer array of the following dimensions for your mainframe:

8163A/B: 1 x 3

8164A/B: 1 x 5

8166A/B: 1 x 18

As an output, Module Status Array returns a one dimensional array. Each array component consists of the questionable status condition value for the corresponding slot. The array includes a result for Slot 0 even if Slot 0 does not exist, that is, for the 8163A/B and 8166A/B.

The questionable status condition value is the current questionable status of the slot. The value is continuously updated. It is not changed by being read.

A questionable status of 0 means that are no current questionable events for the slot.

For any other value, set the Questionable Status Input of the Convert Questionable Status function ([hp816x_convertQuestionableStatus_Q](#)) to the displayed value to get a message about the questionable status.

Data Type: ViInt32[]

Input/Output: OUT

maxMessageLength

Max Message Length returns the maximum number of bytes needed to translate the status information; use this value or a bigger one to

allocate a message buffer for the Message Status function (hp816x_messageStatus_Q).

Data Type: ViPInt32

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)"

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_convertQuestionableStatus_Q

Syntax:

```
ViStatus _VI_FUNC hp816x_convertQuestionableStatus_Q(ViSession ihandle,
Vlnt32 questionableStatusInput, ViCharmessage[ ]);
```

The Convert Questionable Status function (hp816x_convertQuestionableStatus_Q) converts a questionable Status word to a message string describing the questionable event.

The questionable status word for all slots is retuned by the Module Status Array output of the Get Module Status function ([hp816x_getModuleStatus_Q](#)).

You must allocate a message buffer to assure that the array Message is connected to the Max Message Length output returned by the Get Module Status function ([hp816x_getModuleStatus_Q](#)). This allocates a size to the message buffer for the Message output.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized. For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).

Data Type: ViSession
Input/Output: IN

questionableStatusInput

The Questionable Status Input should be connected to the questionable status value for a single slot.

The questionable status condition values for all slots are returned by the Module Status Array output of the Get Module Status function ([hp816x_getModuleStatus_Q](#)).

Questionable Status Input only reads integer values.

Data Type: ViInt32
Input/Output: IN

message

Status Message is both an input and an output.

The Status Message input is a message buffer. Allocate a message buffer with the same size as the Max Message Length output of the Get Module Status function ([hp816x_getModuleStatus_Q](#)).

As an output, Status Message returns the translated status message of the questionable Status Input value. Because each bit set in the status word marks a condition, the explanations for each condition are separated with a comma.

Data Type: ViChar[]
Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "[hp816x_error_message](#)"

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_standardTriggerConfiguration

Syntax:

```
ViStatus _VI_FUNC hp816x_standardTriggerConfiguration(ViSession ihandle,
ViInt32 triggerConfiguration, ViUInt32 nodeAInputConfig, ViUInt32
nodeBInputConfig, ViUInt32 outputMatrixConfiguration);
```

The Standard Trigger Configuration (hp816x_standardTriggerConfiguration) function configures the mainframe trigger system.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized.
<i>triggerConfiguration</i>	For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).
	Data Type: ViSession Input/Output: IN

triggerConfiguration

The Trigger Configuration control allows you to choose the trigger configuration. The following describe the triggering modes:

Disabled (hp816x_TRIG_DISABLED), which you should choose if you do not want to use triggering.

Default (hp816x_TRIG_DEFAULT), which you should choose if you want to enable the trigger connectors.

Passthrough (hp816x_TRIG_PASSTHROUGH), which you should choose if you want an input trigger to automatically generate an output trigger. This allows you to trigger another instrument almost simultaneously.

Loopback (hp816x_TRIG_LOOPBACK), which you should choose if you want an output trigger to automatically generate an input trigger. For example, using this mode, you could trigger each step of a wavelength sweep with just one externally generated input trigger.

Custom (hp816x_TRIG_CUSTOM), which you should choose if you want to configure the trigger function yourself.

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_TRIG_DISABLED	0	Disabled
hp816x_TRIG_DEFAULT	1	Default
hp816x_TRIG_PASSTHROUGH	2	Passthrough
hp816x_TRIG_LOOPBACK	3	Loopback
hp816x_TRIG_CUSTOM	4	Custom

nodeAInputConfig

If you choose Custom as the Trigger Configuration, Node A Input Config must contain a 32-bit value that configures Node A of the trigger system.

Use your instrument's Programming Guide to obtain information about the bits to set.

See the Trigger Input Configuration function ([hp816x_nodeInputConfiguration](#)) for information on setting this value.

Data Type: ViUInt32

Input/Output: IN

nodeBInputConfig

If you choose Custom as the Trigger Configuration, Node B Input Config must contain a 32-bit value that configures Node B of the trigger system.

Use your instrument's Programming Guide to

obtain information about the bits to set.

See the Trigger Input Configuration function ([hp816x_nodeInputConfiguration](#)) for information on setting this value.

Data Type: ViUInt32

Input/Output: IN

outputMatrixConfiguration

If you choose Custom as the Trigger Configuration, Output Matrix Configuration must contain a 32-bit value that sets the node interconnection.

Use your instrument's Programming Guide to obtain information about the bits to set.

See the Trigger Output Configuration function ([hp816x_trigOutConfiguration](#)) for information on setting this value.

Data Type: ViUInt32

Input/Output: IN

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "[hp816x_error_message](#)"

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_standardTriggerConfiguration_Q

Syntax:

```
ViStatus _VI_FUNC hp816x_standardTriggerConfiguration_Q(ViSession
ihandle, ViInt32 triggerConfiguration, ViUInt32 nodeAInputConfig, ViUInt32
nodeBInputConfig, ViUInt32 outputMatrixConfiguration);
```

The Standard Trigger Configuration Query (hp816x_standardTriggerConfiguration_Q) function queries the mainframe trigger system configuration.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized.
<i>triggerConfiguration</i>	For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).
	Data Type: ViSession Input/Output: IN

triggerConfiguration

The Trigger Configuration returns the trigger configuration. The following describe the triggering modes:

Disabled (hp816x_TRIG_DISABLED), if the trigger input and output are disabled.

Default (hp816x_TRIG_DEFAULT), when input triggers are accepted and output triggers are sent over the trigger ports.

Passthrough (hp816x_TRIG_PASSTHROUGH), when an input trigger automatically generates an output trigger. This allows you to trigger another instrument almost simultaneously.

Loopback (hp816x_TRIG_LOOPBACK), when an output trigger automatically generates an input

trigger. For example, using this mode, you could trigger each step of a wavelength sweep with just one externally generated input trigger. Loopback is not supported on the N774xA, N775xA and N776xA and acts the same as Passthrough.

Custom (hp816x_TRIG_CUSTOM), when a custom configuration of the trigger function is set.

Data Type: ViInt32

Input/Output: OUT

Values:

Name	Value	Description
hp816x_TRIG_DISABLED	0	Disabled
hp816x_TRIG_DEFAULT	1	Default
hp816x_TRIG_PASSTHROUGH	2	Passthrough
hp816x_TRIG_LOOPBACK	3	Loopback
hp816x_TRIG_CUSTOM	4	Custom

nodeAInputConfig

If the Trigger Configuration is Custom, the Node A Input Config returns a 32-bit value that configures Node A of the trigger system.

Use your instrument's Programming Guide to obtain information about the bits to set.

See the Trigger Input Configuration function ([hp816x_nodeAInputConfiguration](#)) for information on setting this value.

Data Type: ViUInt32

Input/Output: OUT

nodeBInputConfig

If the Trigger Configuration is Custom, the Node B Input Config returns a 32-bit value that configures Node B of the trigger system.

Use your instrument's Programming Guide to obtain information about the bits to set.

See the Trigger Input Configuration function

[\(hp816x_nodeInputConfiguration\)](#) for information on setting this value.

Data Type: ViUInt32

Input/Output: OUT

outputMatrixConfiguration

If the Trigger Configuration is Custom, the Output Matrix Configuration returns a 32-bit value that sets the node interconnection.

Use your instrument's Programming Guide to obtain information about the bits to set.

See the Trigger Output Configuration function [\(hp816x_trigOutConfiguration\)](#) for information on setting this value.

Data Type: ViUInt32

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "[hp816x_error_message](#)"

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_generateTrigger

Syntax:

```
ViStatus _VI_FUNC hp816x_generateTrigger(ViSession ihandle, ViBoolean triggerAt);
```

The Generate Trigger function (hp816x_generateTrigger) triggers the selected trigger node.

Parameter

ihandle

Description

IHandle is the session handle returned from the Initialize Mainframe function ([hp816x_init](#)). A new session handle is created every time the mainframe is initialized.

For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function ([hp816x_init](#)).

Data Type: ViSession

Input/Output: IN

triggerAt

Use the Trigger At control to trigger Node A (hp816x_NODE_A) or Node B (hp816x_NODE_B).

Data Type: ViBoolean

Input/Output: IN

Values:

Name	Value	Description
hp816x_NODE_A	0	Node A
hp816x_NODE_B	1	Node B

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "[hp816x_error_message](#)"

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_nodeInputConfiguration

Syntax:

```
ViStatus _VI_FUNC hp816x_nodeInputConfiguration(ViBoolean
connectionFunctionNodeA, ViBoolean BNCTriggerConnector, ViBoolean
nodeBTriggerOutput, ViBoolean slot0, ViBoolean slot1, ViBoolean slot2,
ViBoolean slot3, ViBoolean slot4, ViBoolean connectionFunctionNodeB,
ViBoolean BNCTriggerConnector, ViBoolean nodeATriggerOutput, ViBoolean
slot0, ViBoolean slot1, ViBoolean slot2, ViBoolean slot3, ViBoolean slot4,
ViUInt32 nodeAInputConfiguration, ViUInt32 nodeBInputConfiguration);
```

The Trigger Input Configuration function (hp816x_nodeInputConfiguration) helps you use the "Custom" Trigger Configuration of the Standard Trigger Configuration function ([hp816x_standardTriggerConfiguration](#)).

The "Custom" configuration requires you to enter three 32-bit values as input parameters. This function calculates two of these 32-values, Node A Input Config and Node B Input Config.

You should use the Outputs of this function as Inputs for the Standard Trigger Configuration function ([hp816x_standardTriggerConfiguration](#)).

<u>Parameter</u>	<u>Description</u>
<i>connectionFunctionNodeA</i>	Connection Function Node A determines if the trigger inputs to Node A use OR logic or AND logic. If you choose OR logic, any active trigger input can trigger Node A.
<i>BNCTriggerConnector</i>	If you choose AND logic, all active trigger inputs must be fulfilled before Node A is triggered.
Data Type: ViBoolean Input/Output: IN	

Values:

Name	Value	Description
VI_TRUE	(1)	AND
VI_FALSE	(0)	OR

BNCTriggerConnector

BNC Trigger Connector determines if the Input Trigger Connector can trigger Node A.

Data Type: ViBoolean

Input/Output: IN

Values:

Name	Value	Description
VI_TRUE	(1)	Active
VI_FALSE	(0)	Inactive

nodeBTriggerOutput

Node B Trigger Output determines if a trigger at Node B can trigger Node A.

Data Type: ViBoolean

Input/Output: IN

Values:

Name	Value	Description
VI_TRUE	(1)	Active
VI_FALSE	(0)	Inactive

slot0

Slot 0 determines if a trigger event at slot 0 can trigger Node A.

Data Type: ViBoolean

Input/Output: IN

Values:

Name	Value	Description
VI_TRUE	(1)	Active
VI_FALSE	(0)	Inactive

slot1

Slot 1 determines if a trigger event at slot 1 can trigger Node A.

Data Type: ViBoolean

Input/Output: IN

Values:

Name	Value	Description
VI_TRUE	(1)	Active
VI_FALSE	(0)	Inactive

slot2

Slot 2 determines if a trigger event at slot 2 can trigger Node A.

Data Type: ViBoolean

Input/Output: IN

Values:

Name	Value	Description
VI_TRUE	(1)	Active
VI_FALSE	(0)	Inactive

slot3

Slot 3 determines if a trigger event at slot 3 can trigger Node A.

Data Type: ViBoolean

Input/Output: IN

Values:

Name	Value	Description
VI_TRUE	(1)	Active
VI_FALSE	(0)	Inactive

slot4

Slot 4 determines if a trigger event at slot 4 can trigger Node A.

Data Type: ViBoolean

Input/Output: IN

Values:

<u>Name</u>	<u>Value</u>	<u>Description</u>
VI_TRUE	(1)	Active
VI_FALSE	(0)	Inactive

connectionFunctionNodeB

Connection Function Node B determines if the trigger inputs to Node B use OR logic or AND logic.

If you choose OR logic, any active trigger input can trigger Node B.

If you choose AND logic, all active trigger inputs must be fulfilled before Node B is triggered.

Data Type: ViBoolean

Input/Output: IN

Values:

<u>Name</u>	<u>Value</u>	<u>Description</u>
VI_TRUE	(1)	AND
VI_FALSE	(0)	OR

BNCTriggerConnector

BNC Trigger Connector determines if the Input Trigger Connector can trigger Node B.

Data Type: ViBoolean

Input/Output: IN

Values:

<u>Name</u>	<u>Value</u>	<u>Description</u>
VI_TRUE	(1)	Active
VI_FALSE	(0)	Inactive

nodeATriggerOutput

Node A Trigger Output determines if a trigger at Node B can trigger Node B.

Data Type: ViBoolean

Input/Output: IN

Values:

Name	Value	Description
VI_TRUE	(1)	Active
VI_FALSE	(0)	Inactive

slot0

Slot 0 determines if a trigger event at slot 0 can trigger Node B.

Data Type: ViBoolean

Input/Output: IN

Values:

Name	Value	Description
VI_TRUE	(1)	Active
VI_FALSE	(0)	Inactive

slot1

Slot 1 determines if a trigger event at slot 1 can trigger Node B.

Data Type: ViBoolean

Input/Output: IN

Values:

Name	Value	Description
VI_TRUE	(1)	Active
VI_FALSE	(0)	Inactive

slot2

Slot 2 determines if a trigger event at slot 2 can trigger Node B.

Data Type: ViBoolean

Input/Output: IN

Values:

Name	Value	Description
VI_TRUE	(1)	Active

VI_FALSE	(0)	Inactive
-----------------	-----	----------

slot3

Slot 3 determines if a trigger event at slot 3 can trigger Node B.

Data Type: ViBoolean

Input/Output: IN

Values:

Name	Value	Description
VI_TRUE	(1)	Active
VI_FALSE	(0)	Inactive

slot4

Slot 4 determines if a trigger event at slot 4 can trigger Node B.

Data Type: ViBoolean

Input/Output: IN

Values:

Name	Value	Description
VI_TRUE	(1)	Active
VI_FALSE	(0)	Inactive

nodeAInputConfiguration

Node A Input Configuration returns the trigger configuration of Node A as a 32-bit value.

Pass this value as a parameter to the function Standard Trigger Configuration function ([hp816x_standardTriggerConfiguration](#)).

Data Type: ViUInt32

Input/Output: OUT

nodeBInputConfiguration

Node B Input Configuration returns the trigger configuration of Node B as a 32-bit value.

Pass this value as a parameter to the function
Standard Trigger Configuration function
([hp816x_standardTriggerConfiguration](#)).

Data Type: ViUInt32

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine
["hp816x_error_message"](#)

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_trigOutConfiguration

Syntax:

```
ViStatus _VI_FUNC hp816x_trigOutConfiguration(ViBoolean
nodeswitchedtoBNCOutput, ViBoolean slot0, ViBoolean slot1, ViBoolean slot2,
ViBoolean slot3, ViBoolean slot4, ViUInt32 outputMatrixConfiguration);
```

The Trigger Output Configuration function (hp816x_nodeOutputConfiguration) helps you use the "Custom" Trigger Configuration of the Standard Trigger Configuration function ([hp816x_standardTriggerConfiguration](#)).

The "Custom" configuration requires you to enter three 32-bit values as input parameters. This function calculates one of these 32-values, Output Matrix Configuration.

You should use the Output of this function as an Input for the Standard Trigger Configuration function ([\(hp816x_standardTriggerConfiguration\)](#)).

<u>Parameter</u>	<u>Description</u>
<i>nodeswitchedtoBNCOutput</i>	<p>Node switched to BNC Output determines whether Node A or Node B can generate a trigger at the Output Trigger Connector.</p> <p>Data Type: ViBoolean Input/Output: IN</p>

Values:

Name	Value	Description
VI_TRUE	(1)	Node B
VI_FALSE	(0)	Node A

slot0

Slot 0 determines which node can trigger slot 0.

Data Type: ViBoolean
Input/Output: IN

Values:

Name	Value	Description
VI_TRUE	(1)	Node B
VI_FALSE	(0)	Node A

slot1

Slot 1 determines which node can trigger slot 1.

Data Type: ViBoolean

Input/Output: IN

Values:

<u>Name</u>	<u>Value</u>	<u>Description</u>
VI_TRUE	(1)	Node B
VI_FALSE	(0)	Node A

slot2

Slot 2 determines which node can trigger slot 2.

Data Type: ViBoolean

Input/Output: IN

Values:

<u>Name</u>	<u>Value</u>	<u>Description</u>
VI_TRUE	(1)	Node B
VI_FALSE	(0)	Node A

slot3

Slot 3 determines which node can trigger slot 3.

Data Type: ViBoolean

Input/Output: IN

Values:

<u>Name</u>	<u>Value</u>	<u>Description</u>
VI_TRUE	(1)	Node B
VI_FALSE	(0)	Node A

slot4

Slot 4 determines which node can trigger slot 4.

Data Type: ViBoolean

Input/Output: IN

Values:

Name	Value	Description
VI_TRUE	(1)	Node B
VI_FALSE	(0)	Node A

outputMatrixConfiguration

Output Matrix Configuration returns the output matrix configuration as a 32-bit value.

Pass this value as a parameter to the function Standard Trigger Configuration function ([hp816x_standardTriggerConfiguration](#)).

Data Type: ViUInt32

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "[hp816x_error_message](#)"

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



Information for N77 Series Instruments

This version of the 816x VXIplug&play instrument driver has been extended to support the N773xA, N774xA optical power meter and the N775xA and N776xA Optical Attenuator instruments. These are stand-alone instruments that do not require installation in an 816x mainframe. The SCPI command set has been designed for almost complete programming compatibility with the modular instruments of the 816x family. Please be aware of the following details and limitations, when using this driver.

- The N77-series instruments can be controlled via USB, LAN or GPIB interfaces. Especially for transferring large data volumes, the USB and usually LAN interfaces provide faster performance.
- The identification of the individual ports of these multiport instruments in the SCPI and hp816x functions corresponds to the slot number. So for example the ports of the N7745A can be addressed with the slot numbers 1 to 8. The channel number in the commands and functions is not used and can generally be omitted or set to Channel 1 (index 0).
- The N771x-series and 81950A tunable lasers are not supported by the 816x VXIplug&play driver.
- The N773x-series optical switches are supported by the 816x VXIplug&play functions for switches and relevant functions for the mainframes.
- The N774x-series power meters are supported by the 816x VXIplug&play functions for power meters, relevant functions for the mainframes like for standard trigger configuration, and the Multi Frame Lambda Scan application functions.
- The N776x-series optical attenuators are supported by the 816x VXIplug&play functions for attenuators and the relevant functions for the mainframes.
- The N775x-series optical attenuators and power meters are supported by the 816x VXIplug&play functions for attenuators and the relevant functions for the mainframes and the basic functions for the power meters, excluding the advanced functions using data power logging. The choices of averaging time is also more limited for these power meters.
- The averaging time parameter for the power meters is limited in the 816x VXIplug&play functions to a limited set of discrete values, similar to the modular 816x power meters. This set includes: 1, 2, 5, 10, 25, 100, 200, and 500 μ s, 1, 2, 5, 10, 20, 50, 100, 200, 500 ms, and 1, 2, 5, and 10s.
- The maximum number of logging data points that can be acquired is limited by the 816x VXIplug&play data acquisition functions to 100,000 per port.
- The shortest averaging time used by the Multi Frame Lambda Scan application is 25 μ s.
- For the Multi Frame Lambda Scan application, the N774xA instruments can be included simply by registering the instrument as a mainframe, after the first

mainframe with the tunable laser has been registered.

Further details to available functions are found under the categories below:

[Utility Functions](#)

[Mainframe Specific Functions](#)

[Power Sensor Specific Functions](#)

[Applications](#)

hp816x_set_ATT_attenuation

Syntax:

```
ViStatus _VI_FUNC hp816x_set_ATT_attenuation(ViSession ihandle, ViInt32
ATTSlot, ViInt32 selection, ViReal64 attenuation, ViBoolean
waitForCompletion);
```

Use the Set Attenuation function (hp816x_set_ATT_attenuation) to set the attenuation factor.

Adjusting attenuation also adjusts the filter attenuation:

$a_{\text{Filter(new)}} = a_{\text{(new)}} - a_{\text{Offset}}$

and power CALCULATED using:

$P[\text{dBm}] = \text{Pref}[\text{dBm}] - a_{\text{Filter}}$

While the filter is moving to its new position, the status line displays "Busy" (8164) or "B" (8163, 8166).

<u>Parameter</u>	<u>Description</u>
<i>iHandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized.

For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function ([hp816x_init](#)).

Data Type: ViSession

Input/Output: IN

ATTSlot

Use the ATT Slot control to specify the slot used to host the Attenuator module.

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
------	-------	-------------

hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

selection

Sets attenuation factor to device limits or device default. If Manual Entry is activated the entry of Attenuation is used.

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_SELECT_MIN	0	Minimum
hp816x_SELECT_MAX	1	Maximum
hp816x_SELECT_DEF	2	Default
hp816x_SELECT_MANUAL	3	Manual Entry

attenuation

The attenuation factor in dB

Data Type: ViReal64

Input/Output: IN

waitForCompletion

Use this control to turn "Wait for Completion"

On(Off). If "Wait for Completion" is set to Yes(1) the function will not return until the attenuation is set. If "Wait for Completion" is set to "No"(0) the function returns immediately. In this case it is not guaranteed that the attenuation adjustment has been completed. You may use [hp816x_opc_Q](#) to ask for the command completion.

Data Type: ViBoolean

Input/Output: IN

Values:

Name	Value	Description
VI_TRUE	(1)	Yes
VI_FALSE	(0)	No

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "hp816xerror_message"

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_get_ATT_attenuation_Q

Syntax:

```
ViStatus _VI_FUNC hp816x_get_ATT_attenuation_Q(ViSession ihandle,
ViInt32 ATTSlot, ViPReal64 minimum, ViPReal64 maximum, ViPReal64 default,
ViPReal64 current);
```

Use the Get Attenuation function (hp816x_get_ATT_attenuation_Q) to query, for a specified Attenuator module, the current user-set attenuation factor and its limits.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized.
<i>ATTSlot</i>	For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).
	Data Type: ViSession Input/Output: IN

ATTSlot

Use the ATT Slot control to specify the slot used to host the Attenuator module.

Data Type: ViInt32
Input/Output: IN

Values:

<u>Name</u>	<u>Value</u>	<u>Description</u>
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7

hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

minimum

Returns the minimum attenuation.

Data Type: ViPReal64

Input/Output: OUT

maximum

Returns the maximum attenuation.

Data Type: ViPReal64

Input/Output: OUT

default

Returns the default attenuation.

Data Type: ViPReal64

Input/Output: OUT

current

Returns the current user-set attenuation factor.

Data Type: ViPReal64

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine

["hp816xerror_message"](#)

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_set_ATT_attenuationOffset

Syntax:

```
ViStatus _VI_FUNC hp816x_set_ATT_attenuationOffset(ViSession ihandle,
ViInt32 selection, ViReal64 offset, ViInt32 ATTSlot);
```

Use the Set Attenuation Offset function (hp816x_set_ATT_attenuationOffset) to set the attenuation offset factor.

Remark: This function changes the range of the attenuator and recalculates its limits.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized. For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).

Data Type: ViSession

Input/Output: IN

selection

Sets the attenuation offset factor to device limits or device default. If Manual Entry is activated the Offset entry is used.

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_SELECT_MIN	0	Minimum
hp816x_SELECT_MAX	1	Maximum
hp816x_SELECT_DEF	2	Default
hp816x_SELECT_MANUAL	3	Manual Entry

offset

The offset factor in dB.

Data Type: ViReal64

Input/Output: IN

ATTSlot

Use the ATT Slot control to specify the slot used to host the Attenuator module.

Data Type: ViInt32

Input/Output: IN

Values:

<u>Name</u>	<u>Value</u>	<u>Description</u>
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "hp816xerror_message"

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_get_ATT_attenuationOffset_Q

Syntax:

```
ViStatus _VI_FUNC hp816x_get_ATT_attenuationOffset_Q(ViSession ihandle,
ViInt32 ATTSlot, ViPReal64 minimum, ViPReal64 maximum, ViPReal64 default,
ViPReal64 current);
```

Use the Get Attenuation Offset function (hp816x_get_ATT_attenuationOffset_Q) to query, for a specified Attenuator module, the current user-set attenuation offset factor and its limits.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized.
<i>ATTSlot</i>	For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).
	Data Type: ViSession Input/Output: IN

ATTSlot

Use the ATT Slot control to specify the slot used to host the Attenuator module.

Data Type: ViInt32
Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7

hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

minimum

Returns the minimum offset.

Data Type: ViPReal64

Input/Output: OUT

maximum

Returns the maximum offset.

Data Type: ViPReal64

Input/Output: OUT

default

Returns the default offset.

Data Type: ViPReal64

Input/Output: OUT

current

Returns the current offset.

Data Type: ViPReal64

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine

["hp816xerror_message"](#)

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_set_ATT_attenuatorSpeed

Syntax:

```
ViStatus _VI_FUNC hp816x_set_ATT_attenuatorSpeed(ViSession ihandle,
Vilnt32 ATTSlot, Vilnt32 selection, ViReal64 speed);
```

Use the Set Attenuation Speed function (hp816x_set_ATT_attenuatorSpeed) to set the filter transition speed. That is, the speed at which the filter moves from one attenuation setting to another, in dB/s.

Parameter

ihandle

Description

IHandle is the session handle returned from the Initialize Mainframe function ([hp816x_init](#)). A new session handle is created every time the mainframe is initialized.

For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function ([hp816x_init](#)).

Data Type: ViSession

Input/Output: IN

ATTSlot

Use the ATT Slot control to specify the slot used to host the Attenuator module.

Data Type: Vilnt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8

hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

selection

Set filter speed to device limits or use the manual Speed entry.

Data Type: ViInt32

Input/Output: IN

Values:

<u>Name</u>	<u>Value</u>	<u>Description</u>
hp816x_SELECT_MIN	0	Minimum
hp816x_SELECT_MAX	1	Maximum
hp816x_SELECT_DEF	2	Default
hp816x_SELECT_MANUAL	3	Manual Entry

speed

New transition speed.

Data Type: ViReal64

Input/Output: IN

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "hp816xerror_message"

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_get_ATT_attenuatorSpeed_Q

Syntax:

```
ViStatus _VI_FUNC hp816x_get_ATT_attenuatorSpeed_Q(ViSession ihandle,
ViInt32 ATTSlot, ViPReal64 minimum, ViPReal64 maximum, ViPReal64 default,
ViPReal64 current);
```

Use the Get Attenuator Speed function (hp816x_get_ATT_attenuatorSpeed_Q) to query the attenuation filter's transition speed and its limits.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized.
<i>ATTSlot</i>	For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).
	Data Type: ViSession Input/Output: IN

ATTSlot

Use the ATT Slot control to specify the slot used to host the Attenuator module.

Data Type: ViInt32
Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7

hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

minimum

Minimum transition speed.

Data Type: ViPReal64

Input/Output: OUT

maximum

Maximum transition speed.

Data Type: ViPReal64

Input/Output: OUT

default

Default transition speed.

Data Type: ViPReal64

Input/Output: OUT

current

Current transition speed.

Data Type: ViPReal64

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine

["hp816xerror_message"](#)

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_ATT_displayToOffset

Syntax:

```
ViStatus _VI_FUNC hp816x_ATT_displayToOffset(ViSession ihandle, ViInt32
ATTSlot);
```

The Display To OffSet function (hp816x_ATT_displayToOffset) reads the currently displayed attenuation then sets this, as a negative value, as the attenuation offset for the specified Attenuator.

Remark: The range limits of the Attenuator are displaced by this offset value.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized.

For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function ([hp816x_init](#)).

Data Type: ViSession
Input/Output: IN

ATTslot

Use the ATT Slot control to specify the slot used to host the Attenuator module.

Data Type: ViInt32
Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6

hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816xerror_message"](#)"

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_set_ATT_wavelength

Syntax:

```
ViStatus _VI_FUNC hp816x_set_ATT_wavelength(ViSession ihandle, ViInt32
ATTSlot, ViInt32 selection, ViReal64 wavelength);
```

Use the Set Wavelength function (hp816x_set_ATT_wavelength) to set the wavelength used by the Attenuator module. The Attenuator module is factory calibrated for wavelength dependencies.

If a powermeter is integrated into the Attenuator module, this wavelength setting applies to both the Attenuator filter calibration and the powermeter.

If the wavelength:offset table is activated, this wavelength setting also applies the corresponding offset.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized. For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).
	Data Type: ViSession Input/Output: IN

ATTSlot

Use the ATT Slot control to specify the slot used to host the Attenuator module.

Data Type: ViInt32
Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4

hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

selection

Set the wavelength to device limits or uses the Wavelength entry.

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_SELECT_MIN	0	Minimum
hp816x_SELECT_MAX	1	Maximum
hp816x_SELECT_DEF	2	Default
hp816x_SELECT_MANUAL	3	Manual Entry

wavelength

Wavelength value

Data Type: ViReal64

Input/Output: IN

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)"

■

hp816x_get_ATT_wavelength_Q

Syntax:

```
ViStatus _VI_FUNC hp816x_get_ATT_wavelength_Q(ViSession ihandle,
ViInt32 ATTSlot, ViPReal64 minimum, ViPReal64 maximum, ViPReal64 default,
ViPReal64 current);
```

Use the Get Wavelength function (hp816x_get_ATT_wavelength_Q) to query the wavelength calibration for the specified Attenuator module, including its device limits.

If a powermeter is integrated into the Attenuator module, this wavelength setting applies to both the Attenuator filter calibration and the powermeter.

If the wavelength:offset table is activated, this wavelength setting also applies the corresponding offset.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized.
<i>ATTSlot</i>	For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).
	Data Type: ViSession Input/Output: IN

ATTSlot

Use the ATT Slot control to specify the slot used to host the Attenuator module.

Data Type: ViInt32
Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3

hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

minimum

Returns the minimum wavelength calibration.

Data Type: ViPReal64

Input/Output: OUT

maximum

Returns the maximum wavelength calibration.

Data Type: ViPReal64

Input/Output: OUT

default

Returns the default wavelength calibration.

Data Type: ViPReal64

Input/Output: OUT

current

Returns the current wavelength calibration.

Data Type: ViPReal64

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "[hp816xerror_message](#)"

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_set_ATT_powerUnit

Syntax:

```
ViStatus _VI_FUNC hp816x_set_ATT_powerUnit(ViSession ihandle, ViInt32
ATTSlot, ViBoolean unit);
```

Use the Set Power Unit function (hp816x_set_ATT_powerUnit) to set the power unit used by the Attenuator module.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized. For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).

Data Type: ViSession

Input/Output: IN

ATTSlot

Use the ATT Slot control to specify the slot used to host the Attenuator module.

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8

hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

unit

Use this control to select either Watt (1) or dBm (0) as power unit.

Data Type: ViBoolean

Input/Output: IN

Values:

Name	Value	Description
VI_TRUE	(1)	Watt
VI_FALSE	(0)	Dbm

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "hp816xerror_message"

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_get_ATT_powerUnit_Q

Syntax:

```
ViStatus _VI_FUNC hp816x_get_ATT_powerUnit_Q(ViSession ihandle,
ViPBoolean unit, ViInt32 ATTSlot);
```

Use the Get Power Unit function (hp816x_get_ATT_powerUnit_Q) to query the power unit set for the specified Attenuator module.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized. For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).

Data Type: ViSession
Input/Output: IN

<i>unit</i>	This controls returns the current power unit set for the Attenuator module hosted by <i>ATTSlot</i>
-------------	---

Data Type: ViPBoolean
Input/Output: OUT

<i>ATTSlot</i>	Use the ATT Slot control to specify the slot used to host the Attenuator module.
	Data Type: ViInt32 Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_1	1	Slot 1

hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816xerror_message"](#)"

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_set_ATT_absPowerMode

Syntax:

```
ViStatus _VI_FUNC hp816x_set_ATT_absPowerMode(ViSession ihandle,
ViInt32 ATTSlot, ViBoolean absolutePowerMode);
```

Use the Set Absolute Power Mode function (hp816x_set_ATT_absPowerMode) to set the operation mode of the specified Attenuator module. This command exists for compatibility with the 8156A Attenuator.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized.
<i>ATTSlot</i>	For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).
	Data Type: ViSession Input/Output: IN

ATTSlot

Use the ATT Slot control to specify the slot used to host the Attenuator module.

Data Type: ViInt32
Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8

hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

absolutePowerMode

OFF: The Attenuator module attempts to restore attenuation to the last set attenuation after bootup.

ON: The Attenuator module attempts to restore power to the last set power after bootup.

Data Type: ViBoolean

Input/Output: IN

Values:

Name	Value	Description
VI_TRUE	(1)	On
VI_FALSE	(0)	Off

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816xerror_message"](#)"

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_get_ATT_absPowerMode_Q

Syntax:

```
ViStatus _VI_FUNC hp816x_get_ATT_absPowerMode_Q(ViSession ihandle,
ViInt32 ATTSlot, ViPBoolean absolutePowerMode);
```

Use the Get Absolute Power Mode function (hp816x_get_ATT_absPowerMode_Q) to query the current operational mode of a specified Attenuator module.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized.
<i>ATTSlot</i>	For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).
	Data Type: ViSession Input/Output: IN

ATTSlot

Use the ATT Slot control to specify the slot used to host the Attenuator module.

Data Type: ViInt32
Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8

hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

absolutePowerMode

Returns the current operational mode of the specified Attenuator module.

If 0 (VI_FALSE) is returned, the Attenuator module attempts to restore attenuation to the last set attenuation after bootup.

If 1 (VI_TRUE) is returned, the Attenuator module attempts to restore power to the last set power after bootup.

Data Type: ViPBoolean

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816xerror_message"](#)"

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_set_ATT_power

Syntax:

```
ViStatus _VI_FUNC hp816x_set_ATT_power(ViSession ihandle, ViInt32
ATTSlot, ViBoolean powerControl, ViBoolean powerUnit, ViInt32 selection,
ViReal64 power);
```

Use the Set Power (hp816x_set_ATT_power) to set a new power.

If you have an Attenuator module with power control, you can activate the power control mode so that the Attenuator module automatically corrects for changes to input power, so maintaining the desired output power.

If you have an Attenuator module without power control, a reference calibration must be set using [hp816x_set_ATT_powerReference](#).

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized.
<i>ATTslot</i>	For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).

Data Type: ViSession
Input/Output: IN

ATTslot

Use the ATT Slot control to specify the slot used to host the Attenuator module.

Data Type: ViInt32
Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3

hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

powerControl

OFF: The filter position is not automatically corrected.

ON : Internal control loop activated. The filter position is automatically corrected to compensate for changes to input power, so maintaining the desired output power.

Remark: This control is ignored by Attenuator modules with no integrated powermeter.

Data Type: ViBoolean

Input/Output: IN

Values:

Name	Value	Description
VI_TRUE	(1)	On
VI_FALSE	(0)	Off

powerUnit

Sets the power unit.

Data Type: ViBoolean

Input/Output: IN

Values:

Name	Value	Description

VI_TRUE	(1)	Watt
VI_FALSE	(0)	Dbm

selection

Sets the through power to device limits or, if Manual Entry is selected, the value entered in the power control is used.

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_SELECT_MIN	0	Minimum
hp816x_SELECT_MAX	1	Maximum
hp816x_SELECT_DEF	2	Default
hp816x_SELECT_MANUAL	3	Manual Entry

power

If selection equals hp816x_SELECT_MANUAL, sets the the desired output power.

Data Type: ViReal64

Input/Output: IN

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "hp816xerror_message"

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_get_ATT_power_Q

Syntax:

```
ViStatus _VI_FUNC hp816x_get_ATT_power_Q(ViSession ihandle, ViInt32
ATTSlot, ViPBoolean powerControl, ViPBoolean powerUnit, ViPReal64 minimum,
ViPReal64 maximum, ViPReal64 default, ViPReal64 current);
```

Use the Get Power function (hp816x_get_ATT_power_Q) to query the power values and device limits.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized.
<i>ATTSlot</i>	For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).
	Data Type: ViSession Input/Output: IN

ATTSlot

Use the ATT Slot control to specify the slot used to host the Attenuator module.

Data Type: ViInt32
Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7

hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

powerControl

Returns the power control mode

Data Type: ViPBoolean

Input/Output: OUT

powerUnit

Returns the power unit.

If VI_FALSE (0) the power unit is dBm,

If VI_TRUE (1) the power unit is Watt.

Data Type: ViPBoolean

Input/Output: OUT

minimum

Returns the minimum power.

Data Type: ViPReal64

Input/Output: OUT

maximum

Returns the maximum power.

Data Type: ViPReal64

Input/Output: OUT

default

Returns the default power.

Data Type: ViPReal64

Input/Output: OUT

current

Returns the current power.

Data Type: ViPReal64

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816xerror_message"](#)"

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_cp_ATT_refFromExtPM

Syntax:

```
ViStatus _VI_FUNC hp816x_cp_ATT_refFromExtPM(ViSession ihandle, ViInt32  
ATTSlot, ViInt32 PWMSlot, ViInt32 channel);
```

The Copy Ref. from Powermeter function (hp816x_cp_ATT_refFromExtPM) reads the power value from the reference powermeter and copies this value to the reference power of the specified Attenuator.

Remark: This function is only applies to Attenuator modules without power control.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized.

For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function ([hp816x_init](#)).

Data Type: ViSession
Input/Output: IN

ATTslot

Use the ATT Slot control to specify the slot used to host the Attenuator module.

The Attenuator module specified by ATTslot must be hosted by the same mainframe as the Power Meter specified by PWM Slot.

Data Type: ViInt32
Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3

hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

PWMSlot

Use the PWM Slot control to specify the slot hosting the Power Meter (a Power Sensor or Optical Head Interface Module) used to provide the external reference measurement for the Attenuator module.

The Power Meter specified by PWM Slot must be hosted by the same mainframe as the Attenuator module specified by ATTSslot.

Data Type: Vlnt32

Input/Output: IN

Values:

<u>Name</u>	<u>Value</u>	<u>Description</u>
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11

hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

channel

If your reference Power Meter is a dual-channel Power Sensor module, or a dual-channel Optical Head Interface module, use the Channel control to select either Channel 1 or Channel 2.

If your reference Power Meter is a single-channel Power Sensor module, or a Dual Optical Head Interface module, select Channel 1.

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_CHAN_1	0	Channel 1
hp816x_CHAN_2	1	Channel 2

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816xerror_message"](#)"

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_set_ATT_powerReference

Syntax:

```
ViStatus _VI_FUNC hp816x_set_ATT_powerReference(ViSession ihandle,
Vilnt32 ATTSlot, Vilnt32 selection, ViReal64 reference);
```

Use the Set Power Reference function (hp816x_set_ATT_powerReference) to set the reference power value.

Remark: This function only applies to Attenuator modules without power control.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized.
	For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).
	Data Type: ViSession Input/Output: IN

ATTSlot

Use the ATT Slot control to specify the slot used to host the Attenuator module.

Data Type:	Vilnt32
Input/Output:	IN

Values:

Name	Value	Description
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7

hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

selection

Sets the power reference to device limits or if Manual Entry selected, the value entered in the reference control is used..

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_SELECT_MIN	0	Minimum
hp816x_SELECT_MAX	1	Maximum
hp816x_SELECT_DEF	2	Default
hp816x_SELECT_MANUAL	3	Manual Entry

reference

Reference power value in dBm or Watt.

Data Type: ViReal64

Input/Output: IN

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "hp816xerror_message"

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_get_ATT_powerReference_Q

Syntax:

```
ViStatus _VI_FUNC hp816x_get_ATT_powerReference_Q(ViSession ihandle,
ViInt32 ATTSlot, ViPReal64 minimum, ViPReal64 maximum, ViPReal64 default,
ViPReal64 current);
```

Use the Get Power Reference function (hp816x_get_ATT_powerReference_Q) to query the reference power and the device limits.

Remark: This function does not apply to Attenuator modules with power control.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized.

For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function ([hp816x_init](#)).

Data Type: ViSession
Input/Output: IN

ATTSlot

Use the ATT Slot control to specify the slot used to host the Attenuator module.

Data Type: ViInt32
Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5

hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

minimum

Minimum value for the reference power.

Data Type: ViPReal64

Input/Output: OUT

maximum

Maximum value for the reference power.

Data Type: ViPReal64

Input/Output: OUT

default

Default value for the reference power.

Data Type: ViPReal64

Input/Output: OUT

current

Current value for the reference power.

Data Type: ViPReal64

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816xerror_message"](#)"

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_set_ATT_shutterState

Syntax:

```
ViStatus _VI_FUNC hp816x_set_ATT_shutterState(ViSession ihandle, ViInt32
ATTSlot, ViBoolean shutterState);
```

Use the Set Shutter State function (hp816x_set_ATT_shutterState) to set the output path state.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized.
<i>ATTslot</i>	For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).
	Data Type: ViSession Input/Output: IN

ATTslot

Use the ATT Slot control to specify the slot used to host the Attenuator module.

Data Type: ViInt32
Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8

hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

shutterState

Open: Enables the output; the shutter is opened
 Close: Disables the output; the shutter is closed

Data Type: ViBoolean

Input/Output: IN

Values:

Name	Value	Description
VI_TRUE	(1)	Open
VI_FALSE	(0)	Close

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "hp816xerror_message"

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_get_ATT_shutterState_Q

Syntax:

```
ViStatus _VI_FUNC hp816x_get_ATT_shutterState_Q(ViSession ihandle,
Vlnt32 ATTSlot, ViPBoolean shutterState);
```

Use the Get shutter State function (hp816x_get_ATT_shutterState_Q) to query the state of output path for the specified Attenuator module.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized.
<i>ATTSlot</i>	For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).
Data Type: ViSession	
Input/Output:	IN

ATTSlot

Use the ATT Slot control to specify the slot used to host the Attenuator module.

Data Type: Vlnt32
Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8

hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

shutterState

Returns the Attenuator module's shutter state.

If VI_TRUE (1) the shutter is open.

If VI_FALSE (0) the shutter is closed.

Data Type: ViPBoolean

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816xerror_message"](#)"

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_set_ATT_shutterAtPowerOn

Syntax:

```
ViStatus _VI_FUNC hp816x_set_ATT_shutterAtPowerOn(ViSession ihandle,
ViInt32 ATTSlot, ViBoolean shutterState);
```

Use the Set Shutter State at Power On function (hp816x_set_ATT_shutterAtPowerOn) to set the output path state at power on.

Parameter

ihandle

Description

IHandle is the session handle returned from the Initialize Mainframe function ([hp816x_init](#)). A new session handle is created every time the mainframe is initialized.

For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function ([hp816x_init](#)).

Data Type: ViSession

Input/Output: IN

ATTSlot

Use the ATT Slot control to specify the slot used to host the Attenuator module.

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8

hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

shutterState

Open: Enables the output at power on; the shutter is open
 Closed: Disables the output at power on; the shutter is closed

Data Type: ViBoolean

Input/Output: IN

Values:

Name	Value	Description
VI_TRUE	(1)	Open
VI_FALSE	(0)	Closed

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "hp816xerror_message"

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_get_ATT_shutterStateAtPowerOn_Q

Syntax:

```
ViStatus _VI_FUNC hp816x_get_ATT_shutterStateAtPowerOn_Q(ViSession ihandle, Vilnt32 ATTSlot, ViPBoolean shutterState);
```

Use the Get Shutter State at Power On function (hp816x_get_ATT_shutterStateAtPowerOn_Q) to query the state of the specified Attenuator module's output path at power on.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized.
<i>ATTSlot</i>	For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).
	Data Type: ViSession Input/Output: IN

ATTSlot

Use the ATT Slot control to specify the slot used to host the Attenuator module.

Data Type: Vilnt32
Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8

hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

shutterState

Returns the shutter state at power-on for the specified Attenuator module.

If VI_TRUE (1) the shutter is open at power on.
 If VI_FALSE (0) the shutter is closed at power on.

Data Type: ViPBoolean

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816xerror_message"](#)"

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_get_ATT_operStatus

Syntax:

```
ViStatus _VI_FUNC hp816x_get_ATT_operStatus(ViSession ihandle, ViInt32
ATTSlot, ViUInt32 value, ViCharconditionalInfo[ ]);
```

Use the Get Operational Status function (hp816x_get_ATT_operStatus) to query the operational status register and return attenuator-specific status information. To be certain that the Attenuator module is not in an inoperable state, you should also call the Get Module Status function ([hp816x_getModule_Status_Q](#)).

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized.
	For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).

Data Type: ViSession

Input/Output: IN

ATTSlot

Use the ATT Slot control to specify the slot used to host the Attenuator module.

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7

hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

value

Returns the contents of the operational status register. Only bits 3 to 7 of the first byte contain attenuator-specific information:

Bit 3 set - zeroing

Bit 4 set - shutter opened

Bit 5 set - lambda offset active

And, if Bit5 is set (lambda offset active):

Bit6 Bit7

0 0 exact value

1 0 extrapolation below

0 1 extrapolation above

1 1 interpolation

Data Type: ViUInt32

Input/Output: OUT

conditionalInfo

Returns a comma-separated string listing the contents of the operational status register.

Remark: Make sure you have allocated sufficient space for this string.

Data Type: ViChar[]

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816xerror_message"](#)"

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_set_ATT_wlOffsRefPowermeter

Syntax:

```
ViStatus _VI_FUNC hp816x_set_ATT_wlOffsRefPowermeter(ViSession
ihandle, ViInt32 ATTSlot, ViInt32 PWMSlot, ViInt32 channel);
```

Use the Set Wavel. Offs. Ref. PM function (hp816x_set_ATT_wlOffsRefPowermeter) to specify the external powermeter (hosted by the same mainframe as the Attenuator module) used for determine offsets for particular wavelengths. This funtion works in conjunction with the Set Wavelength Offset function when the TOREF control is active. ([hp816x_set_ATT_wavelengthOffset](#))

Remark: This function only applies to Attenuator modules with power control.

<u>Parameter</u>	<u>Description</u>
<i>iHandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized.
<i>ATTSlot</i>	For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).
	Data Type: ViSession Input/Output: IN

ATTSlot

Use the ATT Slot control to specify the slot used to host the Attenuator module.

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5

hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

PWMSlot

Use the PWM Slot control to select a reference powermeter (a Power Sensor or Optical Head Interface Module).

Data Type: ViInt32

Input/Output: IN

Values:

<u>Name</u>	<u>Value</u>	<u>Description</u>
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

channel

Use the Channel control to select the channel that you want to reference.

If you choose a Dual Power Sensor Module or a Dual Optical Head Interface Module, you can use the Channel control to choose between Channel 1 and Channel 2 of your module.

If you choose a single--channel Power Meter, use the Channel control to choose Channel 1.

Data Type: ViInt32

Input/Output: IN

Values:

<u>Name</u>	<u>Value</u>	<u>Description</u>
hp816x_CHAN_1	0	Channel 1
hp816x_CHAN_2	1	Channel 2

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "hp816xerror_message"

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_get_ATT_wIOffsRefPowermeter_Q

Syntax:

```
ViStatus _VI_FUNC hp816x_get_ATT_wIOffsRefPowermeter_Q(ViSession
ihandle, ViInt32 ATTSlot, ViPInt32 slot, ViPInt32 channel);
```

Use the Get Power Reference function (hp816x_get_ATT_wIOffsRefPowermeter_Q) to query the slot and channel used to host the reference powermeter.

Remark: This function only applies to Attenuator modules with power control.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized. For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).
	Data Type: ViSession Input/Output: IN

ATTslot

Use the ATT Slot control to specify the slot used to host the Attenuator module.

Data Type: ViInt32
Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6

hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

slot

Returns the slot number of the powermeter used as reference powermeter.

Data Type: ViPInt32
Input/Output: OUT

channel

Returns the channel of the powermeter used as the reference powermeter.

Data Type: ViPInt32
Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816xerror_message"](#)"

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_set_ATT_wavelengthOffsetState

Syntax:

```
ViStatus _VI_FUNC hp816x_set_ATT_wavelengthOffsetState(ViSession ihandle, Vilnt32 ATTSlot, ViBoolean offsetDependant);
```

Use the Set Wavelength Offset State function (hp816x_set_ATT_wavelengthOffsetState) to turn the Attenuator module's wavelength offset table on(off).

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized.
<i>ATTSlot</i>	For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).
	Data Type: ViSession Input/Output: IN

ATTSlot

Use the ATT Slot control to specify the slot used to host the Attenuator module.

Data Type: Vilnt32
Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8

hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

offsetDependant

Use this control to turn on (off) the Attenuator module's wavelength-offset table.

Data Type: ViBoolean

Input/Output: IN

Values:

Name	Value	Description
VI_TRUE	(1)	On
VI_FALSE	(0)	Off

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "hp816xerror_message"

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_get_ATT_wavelengthOffsetState_Q

Syntax:

```
ViStatus _VI_FUNC hp816x_get_ATT_wavelengthOffsetState_Q(ViSession ihandle, Vilnt32 ATTSlot, ViPBoolean offsetDependancy);
```

Use the Get Wavelength Offset State function (hp816x_get_ATT_wavelengthOffsetState_Q) to query whether a wavelength-offset table is being used by the specified Attenuator module.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized.
<i>ATTSlot</i>	For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).
	Data Type: ViSession Input/Output: IN

ATTSlot

Use the ATT Slot control to specify the slot used to host the Attenuator module.

Data Type: Vilnt32
Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8

hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

offsetDependency

Returns whether a wavelength-offset table is used by the specified Attenuator module.

If VI_TRUE (1) a wavelength-offset table is used.

If VI_FALSE (0) a wavelength-offset table is not used.

Data Type: ViPBoolean

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816xerror_message"](#)

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_set_ATT_wavelengthOffset

Syntax:

```
ViStatus _VI_FUNC hp816x_set_ATT_wavelengthOffset(ViSession ihandle,
ViInt32 ATTSlot, ViReal64 wavelength, ViBoolean offsetSource, ViReal64
offsetValue);
```

Use the Set Wavelength Offset function (hp816x_set_ATT_wavelengthOffset) to set for an offset for a particular wavelength. This function generates a wavelength-offset pair entry in the Attenuator module's Wavelength Offset table.

Remark: If a wavelength-offset pair entry already exists in the Attenuator module's Wavelength Offset table, the offset for this wavelength entry is overwritten.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized. For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).
	Data Type: ViSession Input/Output: IN

ATTSlot

Use the ATT Slot control to specify the slot used to host the Attenuator module.

Data Type: ViInt32
Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4

hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

wavelength

Use this control to enter the wavelength.

Data Type: ViReal64

Input/Output: IN

offsetSource

This control selects whether the value entered in the Offset Value input control (Offset Value), or the power reading from an external reference powermeter (TOREF), is stored.

Remark: If TOREF is selected the following conditions must be met:

1. The Attenuator module must include an integrated powermeter.
2. A reference powermeter must have been previously selected using hp816x_set_ATT_wlOffsRefPM.

Data Type: ViBoolean

Input/Output: IN

Values:

Name	Value	Description
VI_TRUE	(1)	TOREF
VI_FALSE	(0)	Offset Value

offsetValue

Use this control to enter an offset for the specified wavelength.

Data Type: ViReal64

Input/Output: IN

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "hp816xerror_message"

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_get_ATT_wavelengthOffsetIndex_Q

Syntax:

```
ViStatus _VI_FUNC hp816x_get_ATT_wavelengthOffsetIndex_Q(ViSession
ihandle, ViInt32 ATTSlot, ViUInt32 tableIndex, ViPReal64 wavelength, ViPReal64
offset);
```

Use the Get WL/Offset from Index function (hp816x_get_ATT_wavelengthOffsetIndex_Q) to query the wavelength offset applied for a particular wavelength offset table index for a specified Attenuator module.

Parameter

ihandle

Description

IHandle is the session handle returned from the Initialize Mainframe function ([hp816x_init](#)). A new session handle is created every time the mainframe is initialized.

For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function ([hp816x_init](#)).

Data Type: ViSession

Input/Output: IN

ATTslot

Use the ATT Slot control to specify the slot used to host the Attenuator module.

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7

hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

tableIndex

Use this control to specify a particular index for a wavelength offset table entry. Index entries start from 1.

Data Type: ViUInt32

Input/Output: IN

wavelength

Returns the wavelength for the *tableIndex*.

Data Type: ViPReal64

Input/Output: OUT

offset

This control returns the offset for the *tableIndex*.

Data Type: ViPReal64

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816xerror_message"](#)"

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_get_ATT_offsetFromWavelength_Q

Syntax:

```
ViStatus _VI_FUNC hp816x_get_ATT_offsetFromWavelength_Q(ViSession ihandle, Vilnt32 ATTSlot, ViReal64 wavelength, ViPReal64 offset);
```

Use the Get Offset from Wavelength function (hp816x_get_ATT_offsetFromWavelength_Q) to query the attenuation offset applied to a particular wavelength.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized.
<i>ATTSlot</i>	For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init)
	Data Type: ViSession Input/Output: IN

ATTSlot

Use the ATT Slot control to specify the slot used to host the Attenuator module.

Data Type: Vilnt32
Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8

hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

wavelength

The wavelength for which you wish to query the offset applied.

Data Type: ViReal64

Input/Output: IN

offset

The offset applied to *wavelength*

Data Type: ViPReal64

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816xerror_message"](#)"

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_delete_ATT_offsetTblEntries

Syntax:

```
ViStatus _VI_FUNC hp816x_delete_ATT_offsetTblEntries(ViSession ihandle,
ViInt32 ATTSlot, ViBoolean noOfEntries, ViReal64 wavelengthOrIndex);
```

Use the Delete Table Entries function (hp816x_delete_ATT_offsetTblEntries) to delete all entries, or a particular entry, from an attenuator module's wavelength-offset table.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized.
<i>ATTSlot</i>	For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).
	Data Type: ViSession Input/Output: IN

ATTSlot

Use the ATT Slot control to specify the slot used to host the Attenuator module.

Data Type: ViInt32
Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8

hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

noOfEntries

Use the noOfEntries control to specify whether you want all wavelength-offset entries deleted (VI_TRUE) deleted, or a single wavelength-offset entry deleted (VI_FALSE).

Data Type: ViBoolean

Input/Output: IN

Values:

Name	Value	Description
VI_TRUE	(1)	All
VI_FALSE	(0)	Single

wavelengthOrIndex

Use the wavelengthOrIndex control to specify the wavelength or index entry in the wavelength-offset table that you want to delete.

If the input is less than 1, the entry corresponding to the specified wavelength is deleted.

If the input is greater or equal 1, the entry corresponding to the specified index is deleted.

Data Type: ViReal64

Input/Output: IN

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816xerror_message"](#)"

E0615, Jun 2015



hp816x_get_ATT_offsetTable_Q

Syntax:

```
ViStatus _VI_FUNC hp816x_get_ATT_offsetTable_Q(ViSession ihandle,
ViInt32 ATTSlot, ViReal64 wavelengthArray[ ], ViReal64 offsetArray[ ]);
```

Use the Get Offset Table Entries function (hp816x_get_ATT_offsetTable_Q) to return the wavelength offset table of the specified Attenuator module.

Parameter

ihandle

Description

IHandle is the session handle returned from the Initialize Mainframe function ([hp816x_init](#)). A new session handle is created every time the mainframe is initialized.

For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function ([hp816x_init](#)).

Data Type: ViSession

Input/Output: IN

ATTSlot

Use the ATT Slot control to specify the slot used to host the Attenuator module.

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8

hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

wavelengthArray

Returns the wavelengths saved in the wavelength-offset table as a ViReal64 array.

Remark: Make sure that the array is large enough to hold all the entries.

Use [**hp816x_get_ATT_offsetTblSize_Q**](#) to query the number of entries that will be returned.

Data Type: ViReal64[]

Input/Output: OUT

offsetArray

Returns the offsets saved in the wavelength-offset table as a ViReal64 array.

Remark: Make sure that the array is large enough to hold all the entries.

Use [**hp816x_get_ATT_offsetTblSize_Q**](#) to query the number of entries that will be returned.

Data Type: ViReal64[]

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "[**hp816xerror_message**](#)"

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_get_ATT_offsetTblSize_Q

Syntax:

```
ViStatus _VI_FUNC hp816x_get_ATT_offsetTblSize_Q(ViSession ihandle,
ViInt32 ATTSlot, ViUInt32 currentSize, ViUInt32 maximumSize);
```

Use the Get Table Size function (hp816x_get_ATT_offsetTblSize_Q) to query the dimensions of the wavelength-offset table for the specified Attenuator module.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized. For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).
ATTSlot	Data Type: ViInt32 Input/Output: IN

ATTSlot

Use the ATT Slot control to specify the slot used to host the Attenuator module.

Data Type: ViInt32
Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8

hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

currentSize

This control returns the number of entries currently saved in the wavelength-offset table.

Data Type: ViUInt32

Input/Output: OUT

maximumSize

This parameter returns the maximum capacity of the wavelength-offset table .

Data Type: ViUInt32

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816xerror_message"](#)"

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_read_ATT_outputPower

Syntax:

```
ViStatus _VI_FUNC hp816x_read_ATT_outputPower(ViSession
ihandle,ViPInt32 ATT_Slot, Real64 * Output_Power[ ]);
```

The Read ATT Output Power function (hp816x_read_ATT_outputPower) forces the specified attenuator module's builtin powermeter to begin a new measurement cycle. The evaluated output power value is not returned until the measurement averaging time has past.

Remark: Applies to Attenuator modules with Power Control only.

Parameter

ihandle

Description

IHandle is the session handle returned from the Initialize Mainframe function ([hp816x_init](#)). A new session handle is created every time the mainframe is initialized.

For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function ([hp816x_init](#)).

Data Type: ViSession

Input/Output: IN

ATT_Slot

Use the ATT Slot control to specify the slot used to host the Attenuator module.

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5

hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

Output_Power

Returns the current power in W or dBm.

Data Type: ViPReal64

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)"

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_fetch_ATT_outputPower

Syntax:

```
ViStatus _VI_FUNC hp816x_fetch_ATT_outputPower(ViSession
ihandle,ViPInt32 ATT_Slot, Real64 * Output_Power[ ] );
```

The Fetch ATT Output Power function (hp816x_fetch_ATT_outputPower) forces the specified attenuator module's builtin powermeter to return a power value without averaging power measurements over the averaging time period.

If this function is called more than once within an averaging cycle, the returned values will be identical.

Remark: Applies to Attenuator modules with Power Control only.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized.
<i>ATT_Slot</i>	For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).
	Data Type: ViSession Input/Output: IN

ATT_Slot

Use the ATT Slot control to specify the slot used to host the Attenuator module.

Data Type: ViInt32
Input/Output: IN

Values:

<u>Name</u>	<u>Value</u>	<u>Description</u>
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3

hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

Output_Power

Returns the current power in W or dBm.

Data Type: ViPReal64

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)"

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_set_ATT_powerOffset

Syntax:

```
ViStatus _VI_FUNC hp816x_set_ATT_powerOffset(ViSession ihandle, ViInt32
ATTSlot, ViInt32 selection, ViReal64 offset);
```

Use the Set Power Offset function (hp816x_set_ATT_powerOffset) to set the offset factor applied to the power measured at the attenuator module. This factor does not affect the Attenuator module's filter, or its power output.

$$P_{set} = P_{out} - P_{offset}$$

Remark: This function only applies to Attenuator modules with power control.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized.
	For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).
	Data Type: ViSession Input/Output: IN

ATTSlot

Use the ATT Slot control to specify the slot used to host the Attenuator module.

Data Type: ViInt32
Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5

hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

selection

Sets the offset reference to device limits or if Manual Entry selected, the value entered in the offset control is used.

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_SELECT_MIN	0	Minimum
hp816x_SELECT_MAX	1	Maximum
hp816x_SELECT_DEF	2	Default
hp816x_SELECT_MANUAL	3	Manual Entry

offset

Use this control to specify the power offset applied.

Remark: Applying a power offset changes the apparent output power limits.

Data Type: ViReal64

Input/Output: IN

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "hp816xerror_message"

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_get_ATT_powerOffset_Q

Syntax:

```
ViStatus _VI_FUNC hp816x_get_ATT_powerOffset_Q(ViSession ihandle,
ViInt32 ATTSlot, ViPReal64 minimum, ViPReal64 maximum, ViPReal64 default,
ViPReal64 current);
```

Use the Get Power Offset function (hp816x_get_ATT_powerOffset_Q) to query the power offset limits and the current power offset setting for the specified Attenuator module.

Pset = Pout -Poffset

Remark: This function only applies to Attenuator modules with power control.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized. For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).
Data Type: ViSession Input/Output: IN	

ATTSlot

Use the ATT Slot control to specify the slot used to host the Attenuator module.

Data Type: ViInt32
Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4

hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

minimum

Minimum value of power offset.

Remark: This value is changed whenever the power offset is changed.

Data Type: ViPReal64

Input/Output: OUT

maximum

Maximum value of power offset.

Remark: This value is changed whenever the power offset is changed.

Data Type: ViPReal64

Input/Output: OUT

default

Default value of power offset.

Data Type: ViPReal64

Input/Output: OUT

current

Current value of power offset.

Data Type: ViPReal64
Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "[hp816xerror_message](#)"

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_set_ATT_pwrOffsRefPM

Syntax:

```
ViStatus _VI_FUNC hp816x_set_ATT_pwrOffsRefPM(ViSession ihandle,
ViInt32 ATTSlot, ViInt32 PWMSlot, ViInt32 PWMChannel);
```

Use the Set Power Offset Ref. PM function (hp816x_set_ATT_pwrOffsRefPM) to calculate a power offset factor using the builtin Powermeter and an external reference powermeter. The difference P offset = P out -- P ext is copied to the Power offset parameter.

Parameter

ihandle

Description

IHandle is the session handle returned from the Initialize Mainframe function ([hp816x_init](#)). A new session handle is created every time the mainframe is initialized.

For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function ([hp816x_init](#)).

Data Type: ViSession

Input/Output: IN

ATTSlot

Use the ATT Slot control to specify the slot used to host the Attenuator module.

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7

hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

PWMSlot

Use the PWM Slot control to specify the slot used to host a reference powermeter (a Power Sensor or Optical Head Interface Module).

Data Type: VlInt32

Input/Output: IN

Values:

<u>Name</u>	<u>Value</u>	<u>Description</u>
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

PWMChannel

Use the Channel control to specify the channel that you want to reference.

If you choose a Dual Power Sensor Module or a Dual Optical Head Interface Module, you can use the Channel control to choose between Channel 1 and Channel 2 of your module.

If you choose a single--channel Power Meter, specify Channel 1.

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_CHAN_1	0	Channel 1
hp816x_CHAN_2	1	Channel 2

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "hp816xerror_message"

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_set_ATT_offsByRefPM

Syntax:

```
ViStatus _VI_FUNC hp816x_set_ATT_offsByRefPM(ViSession ihandle, ViInt32 ATTSlot, ViInt32 PWMSlot, ViInt32 PWMChannel);
```

Use the Set Att. Off. by EXT PM Loop function (hp816x_set_ATT_offsByRefPM) to calculate an attenuation offset factor by subtracting the power value measured by an external reference powermeter from the power value measured by the Attenuator module's integrated powermeter. The result is stored as the attenuation offset factor.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized.
	For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).

Data Type: ViSession

Input/Output: IN

ATTSlot

Use the ATT Slot control to specify the slot used to host the Attenuator module.

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7

hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

PWMSlot

Use the PWM Slot control to specify the slot used to host the external reference powermeter (a Power Sensor or Optical Head Interface Module).

Data Type: Vlnt32

Input/Output: IN

Values:

<u>Name</u>	<u>Value</u>	<u>Description</u>
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

PWMChannel

Use the Channel control to specify the channel that you want to reference.

If you choose a Dual Power Sensor Module or a Dual Optical Head Interface Module, you can use the Channel control to choose between Channel 1 and Channel 2 of your module.

If you choose a single--channel Power Meter, specify Channel 1.

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_CHAN_1	0	Channel 1
hp816x_CHAN_2	1	Channel 2

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "hp816xerror_message"

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_set_ATT_controlLoopState

Syntax:

```
ViStatus _VI_FUNC hp816x_set_ATT_controlLoopState(ViSession ihandle,
Vlnt32 ATTSlot, ViBoolean controlLoop);
```

Use the Set Control Loop State function (hp816x_set_ATT_controlLoopState) to enable or disable an Attenuator module's power control loop.

Parameter

ihandle

Description

IHandle is the session handle returned from the Initialize Mainframe function ([hp816x_init](#)). A new session handle is created every time the mainframe is initialized.

For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function ([hp816x_init](#)).

Data Type: ViSession

Input/Output: IN

ATTSlot

Use the ATT Slot control to specify the slot used to host the Attenuator module.

Data Type: Vlnt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8

hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

controlLoop

Use this control to enable (VI_TRUE) or disable (VI_FALSE) the power control loop.

Data Type: ViBoolean

Input/Output: IN

Values:

Name	Value	Description
VI_TRUE	(1)	Enable
VI_FALSE	(0)	Disable

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "hp816xerror_message"

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_get_ATT_controlLoopState_Q

Syntax:

```
ViStatus _VI_FUNC hp816x_get_ATT_controlLoopState_Q(ViSession ihandle,
ViInt32 ATTSlot, ViPBoolean controlLoopState);
```

Use the Get Control Loop State function (hp816x_get_ATT_controlLoopState_Q) to query the power control loop state.

Remark: Applies to Attenuator modules with Power Control only.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized.

For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function ([hp816x_init](#)).

Data Type: ViSession
Input/Output: IN

ATTslot

Use the ATT Slot control to specify the slot used to host the Attenuator module.

Data Type: ViInt32
Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6

hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

controlLoopState

This control return the state of the power control loop.

If VI_TRUE is returned the power control loop is enabled,

If VI_FALSE is returned the power control loop is disabled.

Data Type: ViPBoolean

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816xerror_message"](#)"

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_set_ATT_avTime

Syntax:

```
ViStatus _VI_FUNC hp816x_set_ATT_avTime(ViSession ihandle, ViInt32
ATTSlot, ViReal64 averagingTime);
```

Use the Set Averaging Time function (hp816x_set_ATT_avTime) to set the averaging time of an Attenuator module's integrated powermeter.

Parameter

ihandle

Description

IHandle is the session handle returned from the Initialize Mainframe function ([hp816x_init](#)). A new session handle is created every time the mainframe is initialized.

For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function ([hp816x_init](#)).

Data Type: ViSession

Input/Output: IN

ATTSlot

Use the ATT Slot control to specify the slot used to host the Attenuator module.

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8

hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

averagingTime

Use this control to set the averaging time of the integrated powermeter.

Data Type: ViReal64

Input/Output: IN

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "hp816xerror_message"

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_get_ATT_avtime_Q

Syntax:

```
ViStatus _VI_FUNC hp816x_get_ATT_avtime_Q(ViSession ihandle, ViInt32
ATTSlot, ViPReal64 averagingTime);
```

Use the Get Averaging Time function (hp816x_get_ATT_avtime_Q) to query the averaging time set for the Attenuator module's built-in powermeter.

Remark: Applies to Attenuator modules with Power Control only.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized.

For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function ([hp816x_init](#)).

Data Type: ViSession
Input/Output: IN

ATTslot

Use the ATT Slot control to specify the slot used to host the Attenuator module.

Data Type: ViInt32
Input/Output: IN

Values:

<u>Name</u>	<u>Value</u>	<u>Description</u>
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6

hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

averagingTime

This control returns the averaging time set for the Attenuator module's built-in powermeter.

Data Type: ViPReal64

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816xerror_message"](#)"

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_set_ATT_TriggerConfig

Syntax:

```
ViStatus _VI_FUNC hp816x_set_ATT_triggerConfig(ViSession ihandle,
ViInt32 ATT_Slot, ViInt32 Trigger_In);
```

The Set ATT Trigger Configuration (hp816x_set_ATT_triggerConfig) function configures input trigger handling for a builtin powermeter of an attenuator.

Remark: Applies to Attenuator modules with Power Control only.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized.
<i>ATT_Slot</i>	For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).
	Data Type: ViSession Input/Output: IN

ATT_Slot

Use the ATT Slot control to specify the slot used to host the Attenuator module.

Data Type: ViInt32
Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5

hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

Trigger_In

Use the Trigger In control to select one of the following input trigger configurations:

Ignore (hp816x_PWM_TRIGIN_IGN), input triggers are ignored,

Single Measurement
(hp816x_PWM_TRIGIN_SME), an input trigger triggers a single power measurement, or

Complete Measurement
(hp816x_PWM_TRIGIN_CME), as
(hp816x_PWM_TRIGIN_SME), an input trigger triggers a single complete power measurement.

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_PWM_TRIGIN_IGN	0	Ignore
hp816x_PWM_TRIGIN_SME	1	Single Measurement
hp816x_PWM_TRIGIN_CME	2	Complete Measurement

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)"

E0615, Jun 2015



hp816x_get_ATT_triggerConfig_Q

Syntax:

```
ViStatus _VI_FUNC hp816x_get_ATT_triggerConfig_Q(ViSession ihandle,
ViInt32 ATT_Slot, ViPInt32 * Trigger_In);
```

The Get ATT Trigger Configuration (hp816x_get_ATT_triggerConfig_Q) returns the input trigger configuration of a builtin powermeter of an attenuator.

Remark: Applies to Attenuator modules with Power Control only.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized.
<i>ATT_Slot</i>	For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).
	Data Type: ViSession Input/Output: IN

ATT_Slot

Use the ATT Slot control to specify the slot used to host the Attenuator module.

Data Type: ViInt32
Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6

hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

Trigger_In

Trigger In returns one of the following input trigger configurations:

0 (hp816x_PWM_TRIGIN_IGN), input triggers are ignored,

1 (hp816x_PWM_TRIGIN_SME), an input trigger triggers a single power measurement, or

2 (hp816x_PWM_TRIGIN_CME), as (hp816x_PWM_TRIGIN_SME), an input trigger triggers a single complete power measurement.

Data Type: ViPInt32

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_zero_ATT_powermeter

Syntax:

```
ViStatus _VI_FUNC hp816x_zero_ATT_powermeter(ViSession ihandle, ViInt32
ATTslot);
```

Use the Zero Builtin Powermeter function (hp816x_zero_ATT_powermeter) to zero the powermeter built-in to the specified Attenuator module

Remark: Applies to Attenuator modules with Power Control only.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized.

For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function ([hp816x_init](#)).

Data Type: ViSession
Input/Output: IN

ATTslot

Use the ATT Slot control to specify the slot used to host the Attenuator module.

Data Type: ViInt32
Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6

hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816xerror_message"](#)"

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_get_ATT_zeroResult_Q

Syntax:

```
ViStatus _VI_FUNC hp816x_get_ATT_zeroResult_Q(ViSession ihandle, ViInt32  
ATTSlot, ViPInt32 lastZeroResult);
```

Use the Get Last Zero Result function (hp816x_get_ATT_zeroResult_Q) to query the result of the last zeroing operation for the powermeter built-in to the specified Attenuator module.

Remark: Applies to Attenuator modules with Power Control only.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized. For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).
	Data Type: ViSession Input/Output: IN

ATTslot

Use the ATT Slot control to specify the slot used to host the Attenuator module.

Data Type: ViInt32
Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6

hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

lastZeroResult

Returns the result of the last zeroing operation.

If the operation was successful 0 is returned.

Data Type: ViPInt32

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816xerror_message"](#)"

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_zero_ATT_all

Syntax:

ViStatus _VI_FUNC hp816x_zero_ATT_all(ViSession *ihandle*);

Use the Zero All function (hp816x_zero_ATT_all) to zero all powermeters (including the powermeter built-in to each Attenuator module with Power Control, and returnloss modules).

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized.
	For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).
	Data Type: ViSession Input/Output: IN

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "[hp816xerror_message](#)"

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_spectralCalibration

Syntax:

```
ViStatus _VI_FUNC hp816x_spectralCalibration(ViSession ihandle, ViInt32 Slot, ViUInt32 Size_of_Spectrum, ViReal64 Wavelength[], ViReal64 Power[], ViReal64 * Wavelength_Result, ViChar Error_Diagnose );
```

Use the Spectral Calibration function (hp816x_spectralCalibration) to calibrate an attenuator module's built-in power meter or a power meter to an effective wavelength for a multiwavelength or broadband signal. For accurate power measurement of a multiwavelength or broadband signal, you can set the power meter to its effective wavelength (corresponding to the required weighted-averageresponsivity factor) for the signal.

Remark: This function only applies to power meters or Attenuator modules with power control, specifically the 81576A and 81577A. These modules use a filter technology with minimum wavelength dependence, but incorporate a power meter whose responsivity varies with wavelength.

Parameter

ihandle

Description

IHandle is the session handle returned from the Initialize Mainframe function ([hp816x_init](#)). A new session handle is created every time the mainframe is initialized.

For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function ([hp816x_init](#)).

Data Type: ViSession

Input/Output: IN

Slot

Use the Slot control to specify the slot used to host the Attenuator module (or powermeter).

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_1	1	Slot 1

hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

Size_of_Spectrum

Use the Size of Spectrum control to specifies the number of wavelength:power pairs in the multiwavelength DWDM signal.

Data Type: ViUInt32

Input/Output: IN

Wavelength

Input the array of wavelength values

Data Type: ViReal64[]

Input/Output: IN

Power

Input the array of power values in W

Data Type: ViReal64[]

Input/Output: IN

Wavelength_Result

Returns the effective wavelength corresponding to the required powermeter responsivity factor for the DWDM signal.

If 0.0 is returned, the spectral calibration failed.

Data Type: ViReal64

Input/Output: OUT

Error_Diagnose

Returns a diagnostic message, which may help you understand why a spectral calibration failed.

Data Type: ViChar

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)"

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_getWIRespTblSize

Syntax:

```
ViStatus _VI_FUNC hp816x_getWIRespTblSize (ViSession ihandle, ViInt32
ATT_Slot, ViUInt32 * Size, ViUInt32 * CSV_Size);
```

Use the Get Wavelength Resp. Table Size function (hp816x_getWIRespTblSize) to query the size of the wavelength response table used to calibrate a power meter or an Attenuator module's built-in power meter.

Remark: This function only applies to power meters or Attenuator modules with power control, specifically the 81576A and 81577A. These modules use a filter technology with minimum wavelength dependence, but incorporate a power meter whose responsivity varies with wavelength.

Parameter

ihandle

Description

IHandle is the session handle returned from the Initialize Mainframe function ([hp816x_init](#)). A new session handle is created every time the mainframe is initialized.

For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function ([hp816x_init](#)).

Data Type: ViSession

Input/Output: IN

ATT_Slot

Use the ATT Slot control to specify the slot used to host the Attenuator module.

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4

hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

Size

Returns the size of the powermeter's wavelength response calibration table.

Remark: Use this value to allocate memory prior to using the Read Wavelength Response Table function ([hp816x_readWIRespTable](#)).

Data Type: ViUInt32

Input/Output: OUT

CSV_Size

Returns the number of bytes required by the powermeter's wavelength response calibration table in CSV format.

Remark: Use this value to allocate memory prior to using the Read Wavelength Response Table function ([hp816x_readWIRepTblCSV](#)).

Data Type: ViUInt32

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "[hp816xerror_message](#)"

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_readWIRespTable

Syntax:

```
ViStatus _VI_FUNC hp816x_getWIRespTable (ViSession ihandle, ViInt32
ATT_Slot, ViReal64 Wavelength [ ], ViReal64 Response_Factor [ ]);
```

Use the Read Wavelength Resp. Table function (hp816x_readWIRespTable) to read the wavelength response table used to calibrate a power meter or an Attenuator module's built-in power meter.

Use the Get Wavelength Resp. Table Size function ([hp816x_getWvlRespTblSize](#)) to query the size of this table.

Remark: This function only applies to power meters or Attenuator modules with power control, specifically the 81576A and 81577A. These modules use a filter technology with minimum wavelength dependence, but incorporate a power meter whose responsivity varies with wavelength.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized. For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).

Data Type: ViSession

Input/Output: IN

ATT_Slot

Use the ATT Slot control to specify the slot used to host the Attenuator module.

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2

hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

Wavelength

Wavelength is both an input and an output.

As an input, use a Real64 array with size given by the Size result from the hp816x_getWRespTblSize query.

Returns the wavelength entries from the powermeter's wavelength response calibration table.

Data Type: ViReal64[]

Input/Output: OUT

Response_Factor

Response_Factor is both an input and an output.

As an input, use a Real64 array with size given by the Size result from the hp816x_getWRespTblSize query.

Returns the response factor entries from the powermeter's wavelength response calibration table.

Data Type: ViReal64[]

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816xerror_message"](#)"

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_readWIRepTblCSV

Syntax:

```
ViStatus _VI_FUNC hp816x_getWIRepTblCSV (ViSession ihandle, ViInt32
ATT_Slot, ViChar CSV_List []);
```

Use the Read WI Rep. Table CSV function (hp816x_readWIRepTblCSV) to read the wavelength response table used to calibrate a power meter or an Attenuator module's built-in power meter in CSV format.

Use the Get Wavelength Resp. Table Size function ([hp816x_getWvlRespTblSize](#)) to query the size of this table.

Remark: This function only applies to power meters or Attenuator modules with power control, specifically the 81576A and 81577A. These modules use a filter technology with minimum wavelength dependence, but incorporate a power meter whose responsivity varies with wavelength.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized. For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).

Data Type: ViSession

Input/Output: IN

ATT_Slot

Use the ATT Slot control to specify the slot used to host the Attenuator module.

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2

hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

CSV_List

CSV_List is both an input and an output.

As an input, use a text string with length given by the CSV_Size result from the hp816x_getWRespTblSize query.

Returns the wavelength, response_factor entries from the powermeter's wavelength response calibration table in CSV format.

Data Type: ViChar[]

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "[hp816xerror_message](#)"

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



Information for N77 Series Instruments

This version of the 816x VXIplug&play instrument driver has been extended to support the N773xA, N774xA optical power meter and the N775xA and N776xA Optical Attenuator instruments. These are stand-alone instruments that do not require installation in an 816x mainframe. The SCPI command set has been designed for almost complete programming compatibility with the modular instruments of the 816x family. Please be aware of the following details and limitations, when using this driver.

- The N77-series instruments can be controlled via USB, LAN or GPIB interfaces. Especially for transferring large data volumes, the USB and usually LAN interfaces provide faster performance.
- The identification of the individual ports of these multiport instruments in the SCPI and hp816x functions corresponds to the slot number. So for example the ports of the N7745A can be addressed with the slot numbers 1 to 8. The channel number in the commands and functions is not used and can generally be omitted or set to Channel 1 (index 0).
- The N771x-series and 81950A tunable lasers are not supported by the 816x VXIplug&play driver.
- The N773x-series optical switches are supported by the 816x VXIplug&play functions for switches and relevant functions for the mainframes.
- The N774x-series power meters are supported by the 816x VXIplug&play functions for power meters, relevant functions for the mainframes like for standard trigger configuration, and the Multi Frame Lambda Scan application functions.
- The N776x-series optical attenuators are supported by the 816x VXIplug&play functions for attenuators and the relevant functions for the mainframes.
- The N775x-series optical attenuators and power meters are supported by the 816x VXIplug&play functions for attenuators and the relevant functions for the mainframes and the basic functions for the power meters, excluding the advanced functions using data power logging. The choices of averaging time is also more limited for these power meters.
- The averaging time parameter for the power meters is limited in the 816x VXIplug&play functions to a limited set of discrete values, similar to the modular 816x power meters. This set includes: 1, 2, 5, 10, 25, 100, 200, and 500 μ s, 1, 2, 5, 10, 20, 50, 100, 200, 500 ms, and 1, 2, 5, and 10s.
- The maximum number of logging data points that can be acquired is limited by the 816x VXIplug&play data acquisition functions to 100,000 per port.
- The shortest averaging time used by the Multi Frame Lambda Scan application is 25 μ s.
- For the Multi Frame Lambda Scan application, the N774xA instruments can be included simply by registering the instrument as a mainframe, after the first

mainframe with the tunable laser has been registered.

Further details to available functions are found under the categories below:

[Utility Functions](#)

[Mainframe Specific Functions](#)

[Power Sensor Specific Functions](#)

[Applications](#)

hp816x_PWM_slaveChannelCheck

Syntax:

```
ViStatus _VI_FUNC hp816x_PWM_slaveChannelCheck(ViSession ihandle,
Vilnt32 PWMSlot, ViBoolean slaveChannelCheck);
```

The Slave Channel Check function (hp816x_PWM_slaveChannelCheck) allows you to automatically correct parameters that are passed mistakenly to the slave channel of a dual Power Meter.

Some parameters for dual Power Meters can only be passed to the master channel, channel 1. The parameter affects the slave channel, channel 2. See your mainframe's programming guide for a table of GPIB commands that can only be configured for the master channel.

If Slave Channel Check is On, the driver generates a run-time error, if a command is passed mistakenly to the slave channel of a dual Power Meter.

If Slave Channel Check is Off, the driver corrects parameters that are passed mistakenly to the slave channel of a dual Power Meter. Off is the default.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized. For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).
	Data Type: ViSession Input/Output: IN
<i>PWMSlot</i>	Use the PWM Slot control to choose an installed Power Meter (a Power Sensor or Optical Head Interface Module). Data Type: Vilnt32 Input/Output: IN

Values:

<u>Name</u>	<u>Value</u>	<u>Description</u>
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

slaveChannelCheck

If Slave Channel Check is On, the driver generates a runtime error, if a command is passed mistakenly to the slave channel of a dual Power Meter.

If Slave Channel Check is Off, the driver corrects parameters that are passed mistakenly to the slave channel of a dual Power Meter. Off is the default.

Data Type: ViBoolean

Input/Output: IN

Values:

<u>Name</u>	<u>Value</u>	<u>Description</u>
VI_TRUE	(1)	On
VI_FALSE	(0)	Off

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine

["hp816x_error_message"](#)

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_set_PWM_parameters

Syntax:

```
ViStatus _VI_FUNC hp816x_set_PWM_parameters(ViSession ihandle, ViInt32 PWMSlot, ViInt32
channelNumber, ViBoolean rangeMode, ViBoolean powerUnit, ViBoolean internalTrigger,
ViReal64 wavelength, ViReal64 averagingTime, ViReal64 powerRange);
```

The Set PWM Parameters function (hp816x_set_PWM_parameters) sets the most important parameters for a sensor, preparing the instrument to start a measurement.

If a value is not yet applied to the instrument and the Force Transaction function (hp816x_forceTransaction) is switched off, the new values will be set.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized. For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).

Data Type: ViSession
Input/Output: IN

<i>PWMSlot</i>	Use the PWM Slot control to choose an installed Power Meter (a Power Sensor or Optical Head Interface Module).
----------------	--

Data Type: ViInt32
Input/Output: IN

Values:

<u>Name</u>	<u>Value</u>	<u>Description</u>
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10

hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

channelNumber

If you choose a Dual Power Sensor Module or a Dual Optical Head Interface Module, you can use the Channel Number control to choose between Channel 1 and Channel 2 of your module.

If you choose a single-channel Power Meter, use the Channel Number to choose Channel 1.

Data Type: ViInt32

Input/Output: IN

Values:

<u>Name</u>	<u>Value</u>	<u>Description</u>
hp816x_CHAN_1	0	Channel 1
hp816x_CHAN_2	1	Channel 2

rangeMode

Use the MinMax control to choose one of the following range modes:

Auto (hp816x_PWM_RANGE_AUTO), the auto-ranging mode, ensures that the result has a displayed value between 9% and 100% of full scale. The default state is for automatic ranging to be enabled.

Manual (hp816x_PWM_RANGE_MANUAL), which allows you to set a user-defined range.

Data Type: ViBoolean

Input/Output: IN

Values:

<u>Name</u>	<u>Value</u>	<u>Description</u>
hp816x_PWM_RANGE_AUTO	1	Auto
hp816x_PWM_RANGE_MANUAL	0	manual

powerUnit

Use the Power Unit control to choose between power units of Watts (hp816x_PU_WATT) and dBm (hp816x_PU_DBM)

Data Type: ViBoolean

Input/Output: IN

Values:

Name	Value	Description
hp816x_PU_WATT	1	Watt
hp816x_PU_DBM	0	dBm

internalTrigger

Use the Internal Trigger control to choose between Continuous triggering and Immediate triggering.

If you select Immediate triggering (hp816x_PWM_IMMEDIATE) one measurement cycle is performed. When the averaging time is over, the result is available. No further measurement cycles are started. If the result is read with the Fetch PWM Value function, the value will always be the same.

If you select Continuous triggering (hp816x_PWM_CONTINUOUS) a new measurement cycle is immediately triggered after each measurement cycle is completed.

Data Type: ViBoolean

Input/Output: IN

Values:

Name	Value	Description
hp816x_PWM_CONTINUOUS	1	Continuous
hp816x_PWM_IMMEDIATE	0	Immediate

wavelength

Use Wavelength to set the measurement wavelength of the Power Meter in meters (m).

This is the Wavelength value. The responsivity of the Power Meter varies with wavelength. For accurate power measurement, you need to input the wavelength of the optical input

Data Type: ViReal64

Input/Output: IN

averagingTime

Use Averaging Time to set the length of time over which a signal is averaged.

Longer averaging times increase the accuracy and improve the noise rejection of the measurement. Longer averaging times also decrease sensitivity.

If you use a 8153A Series Power Meter and you set Trigger In to Single Measurement using the Set PWM Trigger Configuration ([hp816x_set_PWM_triggerConfiguration](#)) function, it is possible set the averaging time to 5 ms.

Data Type: ViReal64

Input/Output: IN

powerRange

If you choose Manual ([hp816x_PWM_RANGE_MANUAL](#)) using the Range Mode control, you must set a Power Range.

The Power Range is the highest power level that can be measured. If the measured power is less than the Power Range, measured power is displayed.

Data Type: ViReal64

Input/Output: IN

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "[hp816x_error_message](#)"

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_get_PWM_parameters_Q

Syntax:

```
ViStatus _VI_FUNC hp816x_get_PWM_parameters_Q(ViSession ihandle,
ViInt32 PWMSlot, ViInt32 channelNumber, ViPBoolean rangeMode, ViPBoolean
powerUnit, ViPInt32 internalTrigger, ViPReal64 wavelength, ViPReal64
averagingTime, ViPReal64 powerRange);
```

The Get PWM parameters function (hp816x_get_PWM_parameters_Q) returns the most important parameters for a sensor.

Parameter

ihandle

Description

IHandle is the session handle returned from the Initialize Mainframe function ([hp816x_init](#)). A new session handle is created every time the mainframe is initialized.

For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function ([hp816x_init](#)).

Data Type: ViSession

Input/Output: IN

PWMSlot

Use the PWM Slot control to choose an installed Power Meter (a Power Sensor or Optical Head Interface Module).

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4

hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

channelNumber

If you choose a Dual Power Sensor Module or a Dual Optical Head Interface Module, you can use the Channel Number control to choose between Channel 1 and Channel 2 of your module.

If you choose a single-channel Power Meter, use the Channel Number to choose Channel 1.

Data Type: ViInt32

Input/Output: IN

Values:

<u>Name</u>	<u>Value</u>	<u>Description</u>
hp816x_CHAN_1	0	Channel 1
hp816x_CHAN_2	1	Channel 2

rangeMode

Range Mode returns one of the following range modes:

1, the auto-ranging mode (hp816x_PWM_RANGE_AUTO), ensures that the result has a displayed value between 9% and 100% of full scale.

0, manual-ranging mode (hp816x_PWM_RANGE_MANUAL), you set the range.

Data Type: ViPBoolean

Input/Output: OUT

powerUnit

Power Unit returns the following:

0 (hp816x_PU_DBM) units of dBm or

1 (hp816x_PU_WATT) units of Watts.

Data Type: ViPBoolean

Input/Output: OUT

internalTrigger

Internal Trigger returns one of the following internal trigger configurations:

0 (hp816x_TRIG_IMMEDIATE), where only one power measurement is triggered, or

1 (hp816x_TRIG_CONTINUOUS), where power measurements are continuously triggered.

Data Type: ViPInt32

Input/Output: OUT

wavelength

Wavelength returns the measurement wavelength of the Power Meter in meters (m).

This is the Wavelength value. The responsivity of the Power Meter varies with wavelength. For accurate power measurement, you need to input the wavelength of the optical input.

Data Type: ViPReal64

Input/Output: OUT

averagingTime

Averaging Time returns the length of time in seconds (s) over which a signal is averaged.

Longer averaging times increase the accuracy and improve the noise rejection of the measurement. Longer averaging times also decrease sensitivity.

Data Type: ViPReal64

Input/Output: OUT

powerRange

Power Range returns the power range in dBm.

Data Type: ViPReal64

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)"

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_set_PWM_averagingTime

Syntax:

```
ViStatus _VI_FUNC hp816x_set_PWM_averagingTime(ViSession ihandle,
Vilnt32 PWMSlot, Vilnt32 channelNumber, ViReal64 averagingTime);
```

The Set PWM Averaging Time function (hp816x_set_PWM_averagingTime) sets the averaging time of a Power Meter.

Parameter

ihandle

Description

IHandle is the session handle returned from the Initialize Mainframe function ([hp816x_init](#)). A new session handle is created every time the mainframe is initialized.

For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function ([hp816x_init](#)).

Data Type: ViSession

Input/Output: IN

PWMSlot

Use the PWM Slot control to choose an installed Power Meter (a Power Sensor or Optical Head Interface Module).

Data Type: Vilnt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6

hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

channelNumber

If you choose a Dual Power Sensor Module or a Dual Optical Head Interface Module, you can use the Channel Number control to choose between Channel 1 and Channel 2 of your module.

If you choose a single-channel Power Meter, use the Channel Number to choose Channel 1.

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_CHAN_1	0	Channel 1
hp816x_CHAN_2	1	Channel 2

averagingTime

Use Averaging Time to set the length of time over which a signal is averaged.

Longer averaging times increase the accuracy and improve the noise rejection of the measurement. Longer averaging times also decrease sensitivity.

If you use a 8153A Series Power Meter and you set Trigger In to Single Measurement using the Set PWM Trigger Configuration ([\(hp816x_set_PWM_triggerConfiguration\)](#) function, it is possible set the averaging time to 5 ms.

Data Type: ViReal64
Input/Output: IN

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_get_PWM_averagingTime_Q

Syntax:

```
ViStatus _VI_FUNC hp816x_get_PWM_averagingTime_Q(ViSession ihandle,
Vlnt32 PWMSlot, Vlnt32 channelNumber, ViPReal64 averagingTime);
```

The Get PWM Averaging Time (hp816x_get_PWM_averagingTime_Q) function returns the averaging time of a Power Meter.

Parameter

ihandle

Description

IHandle is the session handle returned from the Initialize Mainframe function ([hp816x_init](#)). A new session handle is created every time the mainframe is initialized.

For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function ([hp816x_init](#)).

Data Type: ViSession

Input/Output: IN

PWMSlot

Use the PWM Slot control to choose an installed Power Meter (a Power Sensor or Optical Head Interface Module).

Data Type: Vlnt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6

hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

channelNumber

If you choose a Dual Power Sensor Module or a Dual Optical Head Interface Module, you can use the Channel Number control to choose between Channel 1 and Channel 2 of your module.

If you choose a single-channel Power Meter, use the Channel Number to choose Channel 1.

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_CHAN_1	0	Channel 1
hp816x_CHAN_2	1	Channel 2

averagingTime

Averaging Time returns the length of time over which a signal is averaged.

Longer averaging times increase the accuracy and improve the noise rejection of the measurement. Longer averaging times also decrease sensitivity.

Data Type: ViPReal64

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)"

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_set_PWM_wavelength

Syntax:

```
ViStatus _VI_FUNC hp816x_set_PWM_wavelength(ViSession ihandle, ViInt32 PWMSlot, ViInt32 channelNumber, ViReal64 wavelength);
```

The Set PWM Wavelength function (hp816x_set_PWM_wavelength) sets the wavelength of a sensor.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized. For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).

Data Type: ViSession

Input/Output: IN

PWMSlot

Use the PWM Slot control to choose an installed Power Meter (a Power Sensor or Optical Head Interface Module).

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7

hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

channelNumber

If you choose a Dual Power Sensor Module or a Dual Optical Head Interface Module, you can use the Channel Number control to choose between Channel 1 and Channel 2 of your module.

If you choose a single-channel Power Meter, use the Channel Number to choose Channel 1.

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_CHAN_1	0	Channel 1
hp816x_CHAN_2	1	Channel 2

wavelength

Use Wavelength to set the measurement wavelength of the Power Meter in meters (m).

The responsivity of the Power Meter varies with wavelength. For accurate power measurement, you need to input the wavelength of the optical input.

Data Type: ViReal64

Input/Output: IN

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)"

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_get_PWM_wavelength_Q

Syntax:

```
ViStatus _VI_FUNC hp816x_get_PWM_wavelength_Q(ViSession ihandle,
ViInt32 PWMSlot, ViInt32 channelNumber, ViPReal64 minWavelength, ViPReal64
maxWavelength, ViPReal64 currentWavelength);
```

The Get PWM Wavelength function (hp816x_get_PWM_wavelength_Q) returns the minimum, maximum, and current wavelength of a Power Meter.

The responsivity of the Power Meter varies with wavelength. For accurate power measurement, you need to input the wavelength of the optical input.

Parameter

ihandle

Description

IHandle is the session handle returned from the Initialize Mainframe function ([hp816x_init](#)). A new session handle is created every time the mainframe is initialized.

For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function ([hp816x_init](#)).

Data Type: ViSession

Input/Output: IN

PWMSlot

Use the PWM Slot control to choose an installed Power Meter (a Power Sensor or Optical Head Interface Module).

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3

hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

channelNumber

If you choose a Dual Power Sensor Module or a Dual Optical Head Interface Module, you can use the Channel Number control to choose between Channel 1 and Channel 2 of your module.

If you choose a single-channel Power Meter, use the Channel Number to choose Channel 1.

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_CHAN_1	0	Channel 1
hp816x_CHAN_2	1	Channel 2

minWavelength

Min Wavelength returns the lowest wavelength that can be set for the Power Meter in meters (m).

Data Type: ViPReal64

Input/Output: OUT

maxWavelength

Max Wavelength returns the highest wavelength that can be set for the Power Meter in meters (m).

Data Type: ViPReal64

Input/Output: OUT

currentWavelength

Current Wavelength returns the wavelength in meters (m) that the Power Meter is set to.

Data Type: ViPReal64

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_set_PWM_calibration

Syntax:

```
ViStatus _VI_FUNC hp816x_set_PWM_calibration(ViSession ihandle, ViInt32 PWMSlot, ViInt32 channelNumber, ViReal64 calibration);
```

The Set PWM Calibration function (hp816x_set_PWM_calibration) sets the calibration factor of a Power Meter.

This is a calibration offset that you can enter to compensate for external optical circuitry, for example, the 81022FF Integrating Sphere or a 3 dB coupler. This value is automatically subtracted from the input signal.

Parameter

ihandle

Description

IHandle is the session handle returned from the Initialize Mainframe function ([hp816x_init](#)). A new session handle is created every time the mainframe is initialized.

For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function ([hp816x_init](#)).

Data Type: ViSession

Input/Output: IN

PWMSlot

Use the PWM Slot control to choose an installed Power Meter (a Power Sensor or Optical Head Interface Module).

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4

hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

channelNumber

If you choose a Dual Power Sensor Module or a Dual Optical Head Interface Module, you can use the Channel Number control to choose between Channel 1 and Channel 2 of your module.

If you choose a single-channel Power Meter, use the Channel Number to choose Channel 1.

Data Type: ViInt32

Input/Output: IN

Values:

<u>Name</u>	<u>Value</u>	<u>Description</u>
hp816x_CHAN_1	0	Channel 1
hp816x_CHAN_2	1	Channel 2

calibration

Use the Calibration input to set the calibration factor of the sensor in decibels (dB).

Data Type: ViReal64

Input/Output: IN

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "[hp816x_error_message](#)"

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_get_PWM_calibration_Q

Syntax:

```
ViStatus _VI_FUNC hp816x_get_PWM_calibration_Q(ViSession ihandle,
Vilnt32 PWMSlot, Vilnt32 channelNumber, ViPReal64 calibration);
```

The Get PWM Calibration function (hp816x_get_PWM_calibration_Q) returns the calibration factor of your Power Meter in dBm.

Parameter

ihandle

Description

IHandle is the session handle returned from the Initialize Mainframe function ([hp816x_init](#)). A new session handle is created every time the mainframe is initialized.

For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function ([hp816x_init](#)).

Data Type: ViSession

Input/Output: IN

PWMSlot

Use the PWM Slot control to choose an installed Power Meter (a Power Sensor or Optical Head Interface Module).

Data Type: Vilnt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6

hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

channelNumber

If you choose a Dual Power Sensor Module or a Dual Optical Head Interface Module, you can use the Channel Number control to choose between Channel 1 and Channel 2 of your module.

If you choose a single-channel Power Meter, use the Channel Number to choose Channel 1.

Data Type: ViInt32

Input/Output: IN

Values:

<u>Name</u>	<u>Value</u>	<u>Description</u>
hp816x_CHAN_1	0	Channel 1
hp816x_CHAN_2	1	Channel 2

calibration

Calibration returns the calibration factor of the Power Meter in decibels (dB).

Data Type: ViPReal64

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)"

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_set_PWM_powerRange

Syntax:

```
ViStatus _VI_FUNC hp816x_set_PWM_powerRange(ViSession ihandle, ViInt32 PWMSlot,
ViInt32 channelNumber, ViBoolean rangeMode, ViReal64 powerRange);
```

The Set PWM Range function (hp816x_set_PWM_powerRange) sets the power range of a Power Meter.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized. For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).
	Data Type: ViSession Input/Output: IN

PWMSlot

Use the PWM Slot control to choose an installed Power Meter (a Power Sensor or Optical Head Interface Module).

Data Type: ViInt32
Input/Output: IN

Values:

<u>Name</u>	<u>Value</u>	<u>Description</u>
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15

hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

channelNumber

If you choose a Dual Power Sensor Module or a Dual Optical Head Interface Module, you can use the Channel Number control to choose between Channel 1 and Channel 2 of your module.

If you choose a single-channel Power Meter, use the Channel Number to choose Channel 1.

Data Type: ViInt32

Input/Output: IN

Values:

<u>Name</u>	<u>Value</u>	<u>Description</u>
hp816x_CHAN_1	0	Channel 1
hp816x_CHAN_2	1	Channel 2

rangeMode

Use the Range Mode control to choose one of the following range modes:

Auto, the auto-ranging mode, ensures that the result has a displayed value between 9% and 100% of full scale. The default state is for automatic ranging to be enabled.

Manual, which allows you to set a user-defined range.

Data Type: ViBoolean

Input/Output: IN

Values:

<u>Name</u>	<u>Value</u>	<u>Description</u>
hp816x_PWM_RANGE_AUTO	1	Auto
hp816x_PWM_RANGE_MANUAL	0	Manual

powerRange

If you choose Manual using the Range Mode control, you must set a Power Range in dBm.

The Power Range determines the range of power levels that can be measured.

The Upper Power Limit is always 3 dBm greater than the chosen Power Range value. The resolution is always 40 dBm less than the chosen Power Range value.

For example, if you set the Power Range to -30 dBm, you can measure power between -27 dBm

and -70 dBm, at a resolution of -70 dBm (0.1 nW).

Data Type: ViReal64

Input/Output: IN

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "[hp816x_error_message](#)"

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_get_PWM_powerRange_Q

Syntax:

```
ViStatus _VI_FUNC hp816x_get_PWM_powerRange_Q(ViSession ihandle,
ViInt32 PWMSlot, ViInt32 channelNumber, ViPBoolean rangeMode, ViPReal64
powerRange);
```

The Get PWM Range function (hp816x_get_PWM_powerRange_Q) returns the power range setting of a Power Meter.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized.
<i>PWMSlot</i>	For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).

Data Type: ViSession

Input/Output: IN

PWMSlot

Use the PWM Slot control to choose an installed Power Meter (a Power Sensor or Optical Head Interface Module).

Data Type: ViInt32

Input/Output: IN

Values:

<u>Name</u>	<u>Value</u>	<u>Description</u>
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5

hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

channelNumber

If you choose a Dual Power Sensor Module or a Dual Optical Head Interface Module, you can use the Channel Number control to choose between Channel 1 and Channel 2 of your module.

If you choose a single-channel Power Meter, use the Channel Number to choose Channel 1.

Data Type: ViInt32

Input/Output: IN

Values:

<u>Name</u>	<u>Value</u>	<u>Description</u>
hp816x_CHAN_1	0	Channel 1
hp816x_CHAN_2	1	Channel 2

rangeMode

Range Mode returns one of the following range modes:

1, the auto-ranging mode
 (hp816x_PWM_RANGE_AUTO), ensures that the result has a displayed value between 9% and 100% of full scale.

0, manual-ranging mode
 (hp816x_PWM_RANGE_MANUAL), allows you to set the range.

Data Type: ViPBoolean

Input/Output: OUT

powerRange

Power Range returns the power range in dBm.

The Power Range determines the range of power levels that can be measured.

If the measured power is less than 3 dBm greater than the Power Range, measured power can be returned.

If the measured power is less than 40 dBm less than the Power Range, measured power can be returned.

For example, if you set the Power Range to -30 dBm, you can measure power between -27 dBm and -70 dBm, at a resolution of -70 dBm (0.1 nW).

Data Type: ViPReal64

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_set_PWM_powerUnit

Syntax:

```
ViStatus _VI_FUNC hp816x_set_PWM_powerUnit(ViSession ihandle, ViInt32 PWMSlot, ViInt32 channelNumber, ViInt32 powerUnit);
```

The set PWM Power Unit function (hp816x_set_PWM_powerUnit) sets the power unit of a Power Meter.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized.
	For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).

Data Type: ViSession

Input/Output: IN

PWMSlot

Use the PWM Slot control to choose an installed Power Meter (a Power Sensor or Optical Head Interface Module).

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7

hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

channelNumber

If you choose a Dual Power Sensor Module or a Dual Optical Head Interface Module, you can use the Channel Number control to choose between Channel 1 and Channel 2 of your module.

If you choose a single-channel Power Meter, use the Channel Number to choose Channel 1.

Data Type: Vilnt32

Input/Output: IN

Values:

<u>Name</u>	<u>Value</u>	<u>Description</u>
hp816x_CHAN_1	0	Channel 1
hp816x_CHAN_2	1	Channel 2

powerUnit

Use the Power Unit control to choose between power units of Watts (hp816x_PU_WATT) and dBm (hp816x_PU_DBM).

Data Type: Vilnt32

Input/Output: IN

Values:

<u>Name</u>	<u>Value</u>	<u>Description</u>
hp816x_PU_WATT	1	Watt
hp816x_PU_DBM	0	dBm

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)"

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_get_PWM_powerUnit_Q

Syntax:

```
ViStatus _VI_FUNC hp816x_get_PWM_powerUnit_Q(ViSession ihandle,
Vilnt32 PWMSlot, Vilnt32 channelNumber, ViPlnt32 powerUnit);
```

The Get PWM Power Unit function (hp816x_get_PWM_powerUnit_Q) returns the power units of a Power Meter.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized.

For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function ([hp816x_init](#)).

Data Type: ViSession

Input/Output: IN

PWMSlot

Use the PWM Slot control to choose an installed Power Meter (a Power Sensor or Optical Head Interface Module).

Data Type: Vilnt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7

hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

channelNumber

If you choose a Dual Power Sensor Module or a Dual Optical Head Interface Module, you can use the Channel Number control to choose between Channel 1 and Channel 2 of your module.

If you choose a single-channel Power Meter, use the Channel Number to choose Channel 1.

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_CHAN_1	0	Channel 1
hp816x_CHAN_2	1	Channel 2

powerUnit

Power Unit returns the selected power units:

0, if dBm (hp816x_PU_DBM) are selected, or

1, if Watts (hp816x_PU_WATT) are selected.

Data Type: ViPInt32

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "[hp816x_error_message](#)"

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_set_PWM_referenceSource

Syntax:

```
ViStatus _VI_FUNC hp816x_set_PWM_referenceSource(ViSession ihandle, ViInt32 PWMSlot,
ViInt32 channelNumber, ViInt32 measureMode, ViInt32 referenceSource, ViInt32 slot, ViInt32
channel);
```

The Set PWM Reference Source function (hp816x_set_PWM_referenceSource) sets the reference control parameters.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized.
	For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).
	Data Type: ViSession Input/Output: IN

PWMSlot

Use the PWM Slot control to choose an installed Power Meter (a Power Sensor or Optical Head Interface Module).

Data Type: ViInt32
Input/Output: IN

Values:

<u>Name</u>	<u>Value</u>	<u>Description</u>
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13

hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

channelNumber

If you choose a Dual Power Sensor Module or a Dual Optical Head Interface Module, you can use the Channel Number control to choose between Channel 1 and Channel 2 of your module.

If you choose a single-channel Power Meter, use the Channel Number to choose Channel 1.

Data Type: Vilnt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_CHAN_1	0	Channel 1
hp816x_CHAN_2	1	Channel 2

measureMode

Use the Measure Mode control to set power measurement to one of the following:

Relative measurement
(hp816x_PWM_REF_RELATIV); decibels, dB or

Absolute measurement
(hp816x_PWM_REF_ABSOLUTE); dBm or Watts.

If Relative is chosen using the Measure control, the given reference source will be checked. The function returns an error if the selected reference source does not exist.

Data Type: Vilnt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_PWM_REF_RELATIV	1	Relative
hp816x_PWM_REF_ABSOLUTE	0	Absolute

referenceSource

Use the Reference Source control to choose between the Power Meter's Internal reference and referencing the power reading of a Channel.

Use the Set PWM Reference Value function
([hp816x_set_PWM_referenceValue](#)) to set the internal reference value.

If you reference a Channel you must select this channel using the Slot and Channel inputs. The power reading from this channel updates the power reference continuously.

Data Type: Vilnt32

Input/Output: IN

Values:

<u>Name</u>	<u>Value</u>	<u>Description</u>
hp816x_PWM_TO_REF	0	Internal
hp816x_PWM_TO_MOD	1	Channel

slot

Use the Slot control to choose an installed Power Meter (a Power Sensor or Optical Head Interface Module) as your reference for Relative power measurement.

Data Type: Vilnt32

Input/Output: IN

Values:

<u>Name</u>	<u>Value</u>	<u>Description</u>
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

channel

Use the Channel control to select the channel that you want to reference.

If you choose a Dual Power Sensor Module or a Dual Optical Head Interface Module, you can use the Channel control to choose between Channel 1 and Channel 2 of your module.

If you choose a single-channel Power Meter, use the Channel control to choose Channel 1.

Data Type: Vilnt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_CHAN_1	0	Channel 1
hp816x_CHAN_2	1	Channel 2

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "[hp816x_error_message](#)"

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_get_PWM_referenceSource_Q

Syntax:

```
ViStatus _VI_FUNC hp816x_get_PWM_referenceSource_Q(ViSession ihandle, ViInt32
PWMSlot, ViInt32 channelNumber, ViInt32 measureMode, ViInt32 referenceSource, ViInt32 slot,
ViInt32 channel);
```

The Get PWM Reference Source function (hp816x_get_PWM_referenceSource_Q) gets the reference control parameters.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized.
<i>PWMSlot</i>	For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).
	Data Type: ViSession Input/Output: IN

PWMSlot

Use the PWM Slot control to choose an installed Power Meter (a Power Sensor or Optical Head Interface Module).

Data Type: ViInt32
Input/Output: IN

Values:

<u>Name</u>	<u>Value</u>	<u>Description</u>
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13

hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

channelNumber

If you choose a Dual Power Sensor Module or a Dual Optical Head Interface Module, you can use the Channel Number control to choose between Channel 1 and Channel 2 of your module.

If you choose a single-channel Power Meter, use the Channel Number to choose Channel 1.

Data Type: Vilnt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_CHAN_1	0	Channel 1
hp816x_CHAN_2	1	Channel 2

measureMode

The Measure Mode control returns the power measurement setting according to one of the following:

Relative measurement
(hp816x_PWM_REF_RELATIV); decibels, dB or

Absolute measurement
(hp816x_PWM_REF_ABSOLUTE); dBm or Watts.

If Relative is chosen using the Measure control, the given reference source will be checked. The function returns an error if the selected reference source does not exist.

Data Type: Vilnt32

Input/Output: OUT

Values:

Name	Value	Description
hp816x_PWM_REF_RELATIV	1	Relative
hp816x_PWM_REF_ABSOLUTE	0	Absolute

referenceSource

The Reference Source control returns the selection between the Power Meter's Internal reference and referencing the power reading of a Channel.

Use the Set PWM Reference Value function

[\(hp816x_set_PWM_referenceValue\)](#) to set the internal reference value.

If a Channel is referenced, this channel is selected using the Slot and Channel values. The power reading from this channel updates the power reference continuously.

Data Type: Vilnt32

Input/Output: OUT

Values:

<u>Name</u>	<u>Value</u>	<u>Description</u>
hp816x_PWM_TO_REF	0	Internal
hp816x_PWM_TO_MOD	1	Channel

slot

The Slot control returns the installed Power Meter (a Power Sensor or Optical Head Interface Module) indicated as reference for Relative power measurement.

Data Type: Vilnt32

Input/Output: OUT

Values:

<u>Name</u>	<u>Value</u>	<u>Description</u>
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

channel

The Channel control returns the channel that is referenced.

If you choose a Dual Power Sensor Module or a

Dual Optical Head Interface Module, you can use the Channel control to choose between Channel 1 and Channel 2 of your module.

If you choose a single-channel Power Meter, use the Channel control to choose Channel 1.

Data Type: ViInt32

Input/Output: OUT

Values:

Name	Value	Description
hp816x_CHAN_1	0	Channel 1
hp816x_CHAN_2	1	Channel 2

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)"

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_set_PWM_referenceValue

Syntax:

```
ViStatus _VI_FUNC hp816x_set_PWM_referenceValue(ViSession ihandle,
ViInt32 PWMSlot, ViInt32 channelNumber, ViReal64 internalReference, ViReal64
referenceChannel);
```

The Set PWM Reference Value function (hp816x_set_PWM_referenceValue) sets the internal reference value and the reference channel.

These two reference values are independent and have separate memory locations.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized.

For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function ([hp816x_init](#)).

Data Type: ViSession
Input/Output: IN

PWMSlot

Use the PWM Slot control to choose an installed Power Meter (a Power Sensor or Optical Head Interface Module).

Data Type: ViInt32
Input/Output: IN

Values:

<u>Name</u>	<u>Value</u>	<u>Description</u>
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3

hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

channelNumber

If you choose a Dual Power Sensor Module or a Dual Optical Head Interface Module, you can use the Channel Number control to choose between Channel 1 and Channel 2 of your module.

If you choose a single-channel Power Meter, use the Channel Number to choose Channel 1.

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_CHAN_1	0	Channel 1
hp816x_CHAN_2	1	Channel 2

internalReference

Use Internal Reference to set the internal reference value to a value in dBm or Watts, depending on the units you choose using the Set PWM Power Unit function ([hp816x_set_PWM_powerUnit](#)).

For this value to be chosen as the reference, use the Set PWM Reference Source function ([hp816x_set_PWM_referenceSource](#)) to choose Internal as the Reference Source.

Data Type: ViReal64

Input/Output: IN

referenceChannel

Use Reference Channel to set an additional reference value in decibels (dB), depending on the units you choose using the Set PWM Power Unit function ([hp816x_set_PWM_powerUnit](#)) that will be used if you measure relative to a second channel by using the Set PWM Reference Source function ([hp816x_set_PWM_referenceSource](#)) to choose Channel as the Reference Source.

The power reading from a chosen channel is continuously added to the value you set here and the sum is used to calculate the power ratio value in decibels (dB) using one of the following equations:

$$P(\text{dB}) = Pa(\text{dBm}) - Pb(\text{dBm}) - Prc(\text{dB})$$

$$P(\text{dB}) = 10\log(Pa(W)/Pb(W)) - Prc(\text{dB})$$

where:

P is the power returned in decibels (dB),

Pa is the absolute power in dBm or Watts measured by the current Power Meter channel (this value includes any calibration factor set for the channel),

Pb is the absolute power in dBm or Watts measured by the chosen reference Power Meter channel (this value includes any calibration factor set for the channel), and

Prc is the absolute power in decibels (dB) you set using the Reference Channel input.

Data Type: ViReal64

Input/Output: IN

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)"

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_get_PWM_referenceValue_Q

Syntax:

```
ViStatus _VI_FUNC hp816x_get_PWM_referenceValue_Q(ViSession ihandle,
ViInt32 PWMSlot, ViInt32 channelNumber, ViPInt32 referenceMode, ViPReal64
internalReference, ViPReal64 referenceChannel);
```

The Get PWM Reference Value function (hp816x_get_PWM_referenceValue_Q) returns the current internal, external reference values and the reference mode.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized.
<i>PWMSlot</i>	For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).
	Data Type: ViSession Input/Output: IN

PWMSlot

Use the PWM Slot control to choose an installed Power Meter (a Power Sensor or Optical Head Interface Module).

Data Type: ViInt32
Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5

hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

channelNumber

If you choose a Dual Power Sensor Module or a Dual Optical Head Interface Module, you can use the Channel Number control to choose between Channel 1 and Channel 2 of your module.

If you choose a single-channel Power Meter, use the Channel Number to choose Channel 1.

Data Type: ViInt32

Input/Output: IN

Values:

<u>Name</u>	<u>Value</u>	<u>Description</u>
hp816x_CHAN_1	0	Channel 1
hp816x_CHAN_2	1	Channel 2

referenceMode

Reference Mode returns the following:

0 (hp816x_PWM_REF_ABSOLUTE), for absolute measurement mode, units of dBm or Watts (W) are chosen, or

1 (hp816x_PWM_REF_RELATIV), for relative measurement mode, units of decibels (dB) are chosen.

Data Type: ViPInt32

Input/Output: OUT

internalReference

Internal Reference returns the internal reference value in dBm.

Data Type: ViPReal64

Input/Output: OUT

referenceChannel

Reference Channel returns the additional reference value in dBm that will be used if you measure relative to a second channel.

The power reading from a chosen channel is continuously added to the value you set here and the sum is used to calculate the power ratio value in decibels (dB).

For this channel to be chosen as the reference, use the Set PWM Reference Source function ([hp816x_set_PWM_referenceSource](#)) to choose Channel as the Reference Source.

Data Type: ViPReal64

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "[hp816x_error_message](#)"

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_set_PWM_triggerConfiguration

Syntax:

```
ViStatus _VI_FUNC hp816x_set_PWM_triggerConfiguration(ViSession ihandle, Vilnt32 PWMSlot, Vilnt32 triggerIn, Vilnt32 triggerOut);
```

The Set PWM Trigger Configuration (hp816x_set_PWM_triggerConfiguration) function configures input trigger handling and trigger output generation.

Parameter

ihandle

Description

IHandle is the session handle returned from the Initialize Mainframe function ([hp816x_init](#)). A new session handle is created every time the mainframe is initialized.

For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function ([hp816x_init](#)).

Data Type: ViSession

Input/Output: IN

PWMSlot

Use the PWM Slot control to choose an installed Power Meter (a Power Sensor or Optical Head Interface Module).

Data Type: Vilnt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6

hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

triggerIn

Use the Trigger In control to select one of the following input trigger configurations:

Ignore (hp816x_PWM_TRIGIN_IGN), input triggers are ignored,

Single Measurement
(hp816x_PWM_TRIGIN_SME), an input trigger triggers a single power measurement, or

Complete Measurement
(hp816x_PWM_TRIGIN_CME), an input trigger triggers the data acquisition function that was started most recently to start. The Power Meter data acquisition functions are: the Set PWM Logging ([hp816x_set_PWM_logging](#)) function, the Set PWM Stability ([hp816x_set_PWM_stability](#)) function, and the Set PWM MinMax ([hp816x_set_PWM_minMax](#)) function.

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_PWM_TRIGIN_IGN	0	Ignore
hp816x_PWM_TRIGIN_SME	1	Single Measurement
hp816x_PWM_TRIGIN_CME	2	Complete Measurement

triggerOut

Use the Trigger Out control to select one of the

following output trigger configurations:

Disabled (hp816x_PWM_TRIGOUT_NONE), no output trigger will be generated, or

End of Averaging
(hp816x_PWM_TRIGOUT_AVG), an output trigger is generated at the end of the averaging time for a power measurement, or

Beginning of Averaging
(hp816x_PWM_TRIGOUT_MEAS), an output trigger is generated at the beginning of the averaging time for a power measurement.

Data Type: ViInt32

Input/Output: IN

Values:

<u>Name</u>	<u>Value</u>	<u>Description</u>
hp816x_PWM_TRIGOUT_NONE	0	Disabled
hp816x_PWM_TRIGOUT_AVG	1	End of Averaging
hp816x_PWM_TRIGOUT_MEAS	2	Beginning of Averaging

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_get_PWM_triggerConfiguration

Syntax:

```
ViStatus _VI_FUNC hp816x_get_PWM_triggerConfiguration(ViSession ihandle, ViInt32 PWMSlot, ViPInt32 triggerIn, ViPInt32 triggerOut);
```

The Get PWM Trigger Configuration (hp816x_get_PWM_triggerConfiguration) function returns the input trigger handling configuration and the output trigger generation configuration for a Power Meter.

Parameter

ihandle

Description

IHandle is the session handle returned from the Initialize Mainframe function ([hp816x_init](#)). A new session handle is created every time the mainframe is initialized.

For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function ([hp816x_init](#)).

Data Type: ViSession

Input/Output: IN

PWMSlot

Use the PWM Slot control to choose an installed Power Meter (a Power Sensor or Optical Head Interface Module).

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6

hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

triggerIn

Trigger In returns one of the following input trigger configurations:

0 (hp816x_PWM_TRIGIN_IGN), input triggers are ignored,

1 (hp816x_PWM_TRIGIN_SME), an input trigger triggers a single power measurement, or

2 (hp816x_PWM_TRIGIN_CME), an input trigger triggers the data acquisition function that was started most recently to start. The Power Meter data acquisition functions are: the Set PWM Logging ([hp816x_set_PWM_logging](#)) function, the Set PWM Stability (hp816x_set_PWM_stability) function, and the Set PWM MinMax ([hp816x_set_PWM_minMax](#)) function.

Data Type: ViPInt32

Input/Output: OUT

triggerOut

Trigger Out returns one of the following output trigger configurations:

0 (hp816x_PWM_TRIGOUT_NONE), no output trigger will be generated, or

1 (hp816x_PWM_TRIGOUT_AVG), an output trigger is generated at the end of the averaging time for a power measurement, or

2 (hp816x_PWM_TRIGOUT_MEAS), an output trigger is generated at the beginning of the averaging time for a power measurement.

Data Type: ViPInt32

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)"

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_PWM_displayToReference

Syntax:

```
ViStatus _VI_FUNC hp816x_PWM_displayToReference(ViSession ihandle,
Vilnt32 PWMSlot, Vilnt32 channelNumber);
```

Use the PWM Display to Reference function (hp816x_PWM_displayToReference) to choose the currently displayed power value as the new internal reference.

Parameter

ihandle

Description

IHandle is the session handle returned from the Initialize Mainframe function ([hp816x_init](#)). A new session handle is created every time the mainframe is initialized.

For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function ([hp816x_init](#)).

Data Type: ViSession

Input/Output: IN

PWMSlot

Use the PWM Slot control to choose an installed Power Meter (a Power Sensor or Optical Head Interface Module).

Data Type: Vilnt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6

hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

channelNumber

If you choose a Dual Power Sensor Module or a Dual Optical Head Interface Module, you can use the Channel Number control to choose between Channel 1 and Channel 2 of your module.

If you choose a single--channel Power Meter, use the Channel Number to choose Channel 1.

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_CHAN_1	0	Channel 1
hp816x_CHAN_2	1	Channel 2

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)"



hp816x_set_PWM_internalTrigger

Syntax:

```
ViStatus _VI_FUNC hp816x_set_PWM_internalTrigger(ViSession ihandle, ViInt32
PWMSlot, ViInt32 channelNumber, ViBoolean internalTrigger);
```

The Set PWM Internal Trigger function (hp816x_set_PWM_internalTrigger) determine how the Power Meter starts measuring.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized.

For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function ([hp816x_init](#)).

Data Type: ViSession
Input/Output: IN

PWMSlot

Use the PWM Slot control to choose an installed Power Meter (a Power Sensor or Optical Head Interface Module).

Data Type: ViInt32
Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12

hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

channelNumber

If you choose a Dual Power Sensor Module or a Dual Optical Head Interface Module, you can use the Channel Number control to choose between Channel 1 and Channel 2 of your module.

If you choose a single-channel Power Meter, use the Channel Number to choose Channel 1.

Data Type: ViInt32

Input/Output: IN

Values:

<u>Name</u>	<u>Value</u>	<u>Description</u>
hp816x_CHAN_1	0	Channel 1
hp816x_CHAN_2	1	Channel 2

internalTrigger

Use the Internal Trigger control to choose between Continuous triggering (**hp816x_PWM_CONTINUOUS**) and Immediate triggering (**hp816x_PWM_IMMEDIATE**).

Selecting Immediate triggering performs one measurement cycle. When the averaging time is over, the result is available. No further measurement cycles are started. If the result is read with the Fetch PWM Value function, the value will always be the same .

Selecting Continuous triggering will produce different results, because after finishing one measure cycle, a new cycle is immediately triggered.

Data Type: ViBoolean

Input/Output: IN

Values:

<u>Name</u>	<u>Value</u>	<u>Description</u>
hp816x_PWM_CONTINUOUS	1	Continuous
hp816x_PWM_IMMEDIATE	0	Immediate

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "[hp816x_error_message](#)"

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_start_PWM_internalTrigger

Syntax:

```
ViStatus _VI_FUNC hp816x_start_PWM_internalTrigger(ViSession ihandle,
Vilnt32 PWMSlot, Vilnt32 channelNumber);
```

The Start PWM Internal Trigger function (hp816x_start_PWM_internalTrigger) starts a Power Meter measurement cycle.

If you have previously chosen Continuous Triggering mode, see the Set PWM Internal Trigger function, the function chooses Immediate Triggering mode automatically. You need to use the Start PWM Trigger function to start another Power Meter measurement cycle.

Parameter

ihandle

Description

IHandle is the session handle returned from the Initialize Mainframe function ([hp816x_init](#)). A new session handle is created every time the mainframe is initialized.

For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function ([hp816x_init](#)).

Data Type: ViSession

Input/Output: IN

PWMSlot

Use the PWM Slot control to choose an installed Power Meter (a Power Sensor or Optical Head Interface Module).

Data Type: Vilnt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_0	0	Slot 0
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2

hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

channelNumber

If you choose a Dual Power Sensor Module or a Dual Optical Head Interface Module, you can use the Channel Number control to choose between Channel 1 and Channel 2 of your module.

If you choose a single-channel Power Meter, use the Channel Number to choose Channel 1.

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_CHAN_1	0	Channel 1
hp816x_CHAN_2	1	Channel 2

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_PWM_readValue

Syntax:

```
ViStatus _VI_FUNC hp816x_PWM_readValue(ViSession ihandle, ViInt32
PWMSlot, ViUInt32 channelNumber, ViPReal64 measuredValue);
```

The Read PWM Value function (hp816x_PWM_readValue) forces the instrument to start a measurement cycle. The evaluated power value will not be returned until the averaging time period is finished.

Parameter

ihandle

Description

IHandle is the session handle returned from the Initialize Mainframe function ([hp816x_init](#)). A new session handle is created every time the mainframe is initialized.

For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function ([hp816x_init](#)).

Data Type: ViSession

Input/Output: IN

PWMSlot

Use the PWM Slot control to choose an installed Power Meter (a Power Sensor or Optical Head Interface Module).

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6

hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

channelNumber

If you choose a Dual Power Sensor Module or a Dual Optical Head Interface Module, you can use the Channel Number control to choose between Channel 1 and Channel 2 of your module.

If you choose a single-channel Power Meter, use the Channel Number to choose Channel 1.

Data Type: ViUInt32

Input/Output: IN

Values:

<u>Name</u>	<u>Value</u>	<u>Description</u>
hp816x_CHAN_1	0	Channel 1
hp816x_CHAN_2	1	Channel 2

measuredValue

Measured Value returns the measured power value in Watts (W) or dBm.

Data Type: ViPReal64

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)"

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_PWM_fetchValue

Syntax:

```
ViStatus _VI_FUNC hp816x_PWM_fetchValue(ViSession ihandle, ViUInt32
PWMSlot, ViUInt32 channelNumber, ViPReal64 measuredValue);
```

The Fetch PWM Value function (hp816x_PWM_fetchValue) immediately returns a power value without averaging power measurements over the averaging time period.

REMARKS: If this function is called more than once within an averaging cycle, the returned values will be identical.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized.

For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function ([hp816x_init](#)).

Data Type: ViSession
Input/Output: IN

PWMSlot

Use the PWM Slot control to choose an installed Power Meter (a Power Sensor or Optical Head Interface Module).

Data Type: ViInt32
Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4

hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

channelNumber

If you choose a Dual Power Sensor Module or a Dual Optical Head Interface Module, you can use the Channel Number control to choose between Channel 1 and Channel 2 of your module.

If you choose a single-channel Power Meter, use the Channel Number to choose Channel 1.

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_CHAN_1	0	Channel 1
hp816x_CHAN_2	1	Channel 2

measuredValue

Measured Value returns the measured power value in Watts (W) or dBm.

Data Type: ViPReal64

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine [hp816x_error_message](#)

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_PWM_readAll

Syntax:

```
ViStatus _VI_FUNC hp816x_PWM_readValue(ViSession Ihandle, ViUInt32 *  
Number_of_Channels, ViInt32 Slots[ ], ViInt32 Channels[ ], ViPReal64 Values[ ] );
```

The Read All PWM Values function (hp816x_PWM_readAll) forces the instrument to begin a measurement cycle. Power values are not returned until the averaging time required to read every powermeter channel hosted by the mainframe is complete.

The returned information is stored in three arrays. Each element in the "Values" array (value[index]) was read from the powermeter channel identified by the corresponding element in the "Slots" array (slot[index]) and "Channel" array (channel[index]).

Note:

The caller of this function is responsible for allocating enough memory for the passed arrays. (The maximum array size is 34 for an 8166 mainframe)

Every powermeter module read by this function must have firmware version 3.0 or higher.

<u>Parameter</u>	<u>Description</u>
<i>Ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized. For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).
	Data Type: ViSession Input/Output: IN
<i>Number_of_Channels</i>	Returns the number of powermeter channels that were read. Data Type: ViUInt32 * Input/Output: OUT

Slots[]

Returns an array containing the mainframe slot number for each powermeter read.

Data Type: ViInt32[]

Input/Output: OUT

Channels[]

Returns an array containing the channel number for each powermeter read.

Data Type: ViInt32[]

Input/Output: OUT

Values:

Name	Value	Description
hp816x_CHAN_1	0	Channel 1
hp816x_CHAN_2	1	Channel 2

Values[]

Returns an array containing the measured power value (in Watts (W) or dBm) for each powermeter read.

Data Type: ViReal64[]

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_PWM_zeroing

Syntax:

```
ViStatus _VI_FUNC hp816x_PWM_zeroing(ViSession ihandle, ViPInt32
zeroingResult, ViInt32 PWMSlot, ViInt32 channelNumber);
```

The PWM Zeroing function (hp816x_PWM_zeroing) removes electrical offsets for one Power Meter.

The instrument measures by converting optical power to electrical power, and then measuring electrical power. An electrical offset is power that is always present, even if there is no light at the input. If this offset is not removed, it will affect power measurement.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized.

For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function ([hp816x_init](#)).

Data Type: ViSession
Input/Output: IN

zeroingResult

Zeroing Result returns an integer value.

If 0 is returned, the zeroing operation has been successfully completed.

If 98 is returned, the zeroing operation has failed because the Power Meter received light. The most common reason for zeroing to fail is if: - a source is connected to the Power Meter's input connector, - the fiber connected to the Power Meter's input connector is collecting light, or - the Power Meter receives ambient light because the input connector is uncovered.

Any other value is an error code and can be looked up in your instrument's Programming Guide.

Data Type: ViPInt32

Input/Output: OUT

PWMSlot

Use the PWM Slot control to choose an installed Power Meter (a Power Sensor or Optical Head Interface Module).

Data Type: ViInt32

Input/Output: IN

Values:

<u>Name</u>	<u>Value</u>	<u>Description</u>
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

channelNumber

If you choose a Dual Power Sensor Module or a Dual Optical Head Interface Module, you can use the Channel Number control to choose between Channel 1 and Channel 2 of your module.

If you choose a single-channel Power Meter, use the Channel Number to choose Channel 1.

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_CHAN_1	0	Channel 1
hp816x_CHAN_2	1	Channel 2

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)"

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_PWM_zeroingAll

Syntax:

```
ViStatus _VI_FUNC hp816x_PWM_zeroingAll(ViSession ihandle, ViPInt32
summaryofZeroingAll);
```

The PWM Zeroing All function (hp816x_PWM_zeroingAll) removes electrical offsets for all Power Meter modules.

The instrument measures by converting optical power to electrical power, and then measuring electrical power. An electrical offset is power that is always present, even if there is no light at the input. If this offset is not removed, it will affect power measurement.

NOTE: Because zeroing is a time consuming operation, the timeout is temporarily increased.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized. For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).
	Data Type: ViSession Input/Output: IN

summaryofZeroingAll

Summary of Zeroing All returns a summary of the results of every Return Loss module and Power Meter in the instrument.

If 0 is returned, the zeroing operations have been successfully completed.

If 98 is returned, a zeroing operation has failed because a Return Loss module or Power Meter has received light. The most common reason for zeroing to fail is if: - a source is connected to a Return Loss module's or Power Meter's input connector, - the fiber connected to a Return Loss module's or Power Meter's input connector is collecting light, or - a Return Loss module or

Power Meter received ambient light because the input connector is uncovered.

Any other value is an error code and can be looked up in your instrument's Programming Guide.

Data Type: ViPInt32

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_PWM_ignoreError

Syntax:

```
ViStatus _VI_FUNC hp816x_PWM_ignoreError(ViSession ihandle, ViInt32 PWMSlot, ViInt32 channelNumber, ViBoolean ignoreError, ViInt32 instrumentErrorNumber);
```

Use the Ignore Instrument Error function (hp816x_PWM_ignoreError) to trap a specific instrument error for a Power Meter.

If the specified error occurs, the returned status of a call causing this error will be VI_SUCCESS.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized.

For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function ([hp816x_init](#)).

Data Type: ViSession
Input/Output: IN

PWMSlot

Use the PWM Slot control to choose an installed Power Meter (a Power Sensor or Optical Head Interface Module).

Data Type: ViInt32
Input/Output: IN

Values:

<u>Name</u>	<u>Value</u>	<u>Description</u>
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3

hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

channelNumber

If you choose a Dual Power Sensor Module or a Dual Optical Head Interface Module, you can use the Channel Number control to choose between Channel 1 and Channel 2 of your module.

If you choose a single-channel Power Meter, use the Channel Number to choose Channel 1.

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_CHAN_1	0	Channel 1
hp816x_CHAN_2	1	Channel 2

ignoreError

Use the Ignore Error control to set the status of the instrument error specified by the Instrument Error Number control.

If you want the instrument error to be ignored (call returns VI-SUCCESS), select "Yes".

Otherwise, select "No".

Data Type: ViBoolean

Input/Output: IN

Values:

Name	Value	Description
VI_TRUE	(1)	Yes
VI_FALSE	(0)	No

InstrumentErrorNumber

Use the Instrument Error Number control to specify the instrument error number.

See your instrument's Programming Guide for a listing of the instrument error numbers and messages.

Data Type: ViInt32

Input/Output: IN

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)"

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_set_PWM_logging

Syntax:

```
ViStatus _VI_FUNC hp816x_set_PWM_logging(ViSession ihandle, ViInt32 PWMSlot, ViInt32 channelNumber, ViReal64 averagingTime, ViInt32 dataPoints, ViPInt32 estimatedTimeout);
```

The Set PWM Logging function (hp816x_set_PWM_logging) sets the parameters for a logging operation and starts the logging.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized.
<i>PWMSlot</i>	For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).
	Data Type: ViSession Input/Output: IN

PWMSlot

Use the PWM Slot control to choose an installed Power Meter (a Power Sensor or Optical Head Interface Module).

Data Type: ViInt32
Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5

hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

channelNumber

If you choose a Dual Power Sensor Module or a Dual Optical Head Interface Module, you can use the Channel Number control to choose between Channel 1 and Channel 2 of your module.

If you choose a single-channel Power Meter, use the Channel Number to choose Channel 1.

Data Type: ViInt32

Input/Output: IN

Values:

<u>Name</u>	<u>Value</u>	<u>Description</u>
hp816x_CHAN_1	0	Channel 1
hp816x_CHAN_2	1	Channel 2

averagingTime

Use Averaging Time to set the length of time in seconds (s) over which a logging operation power measurement is averaged.

Data Type: ViReal64

Input/Output: IN

dataPoints

Use Data Points to set the number of values to be taken during a logging operation.

The maximum number of values is 12000.

Allocate a one-dimensional array with the same dimensions as this value and use as the Logging Result input for the Get PWM Logging Results function ([hp816x_get_PWM_loggingResults_Q](#)).

Data Type: ViInt32

Input/Output: IN

estimatedTimeout

Estimated Timeout returns the estimated time in milliseconds (ms), the logging operation will take; this value can be used to adjust the Timeout function ([hp816x_timeOut](#)).

Data Type: ViPInt32

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "[hp816x_error_message](#)"

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_get_PWM_loggingResults_Q

Syntax:

```
ViStatus _VI_FUNC hp816x_get_PWM_loggingResults_Q(ViSession ihandle,
ViInt32 PWMSlot, ViInt32 channelNumber, ViBoolean waitforCompletion,
ViBoolean resultUnit, ViPBoolean loggingStatus, ViReal64/loggingResult[ ]);
```

The Get PWM Logging Results function (hp816x_get_PWM_loggingResults_Q) returns the result of a previously started logging operation.

NOTE: The Wait for Completion control should only be set to Yes if the application is thoroughly tested because using this control may result in an infinite loop that blocks the calling function.

Also, if the function detects that logging is finished, the logging operation stops automatically.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized. For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).

Data Type: ViSession

Input/Output: IN

PWMSlot

Use the PWM Slot control to choose an installed Power Meter (a Power Sensor or Optical Head Interface Module).

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_1	1	Slot 1

hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

channelNumber

If you choose a Dual Power Sensor Module or a Dual Optical Head Interface Module, you can use the Channel Number control to choose between Channel 1 and Channel 2 of your module.

If you choose a single-channel Power Meter, use the Channel Number to choose Channel 1.

Data Type: ViInt32

Input/Output: IN

Values:

<u>Name</u>	<u>Value</u>	<u>Description</u>
hp816x_CHAN_1	0	Channel 1
hp816x_CHAN_2	1	Channel 2

waitForCompletion

Use the Wait for Completion control to choose one of the following:

Yes (VI_TRUE), the logging operation will wait until the Total Time specified in the Set PWM Logging function has elapsed, or

No (VI_FALSE), the logging operation will stop

immediately.

Data Type: ViBoolean

Input/Output: IN

Values:

Name	Value	Description
VI_TRUE	(1)	Yes
VI_FALSE	(0)	No

resultUnit

Use the Result Unit control to choose power units of dBm (hp816x_PU_DBM) or Watts (hp816x_PU_WATT) for the logging operation result.

Data Type: ViBoolean

Input/Output: IN

Values:

Name	Value	Description
hp816x_PU_DBM	0	dBm
hp816x_PU_WATT	1	Watt

loggingStatus

Logging Status returns a Boolean representing the current status of the logging operation.

If Logging Status equals 1, the operation is completed.

If Logging Status equals 0, the operation is being performed.

Data Type: ViPBoolean

Input/Output: OUT

loggingResult

Logging Result is both an input and an output.

As an input, Logging Result accepts a one-

dimensional integer array that determines the size of the Logging Result output. Allocate a one-dimensional integer array with the same dimensions as the Data Points value you have set for the Set PWM Logging function ([hp816x_set_PWM_logging](#)).

As an output, Logging Result returns the result of a logging operation as a one-dimensional integer array.

Data Type: ViReal64[]

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "[hp816x_error_message](#)"

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_set_PWM_stability

Syntax:

```
ViStatus _VI_FUNC hp816x_set_PWM_stability(ViSession ihandle, ViInt32 PWMSlot, ViInt32 channelNumber, ViReal64 averagingTime, ViReal64 delayTime, ViReal64 totalTime, ViPInt32 estimatedResults);
```

The Set PWM Stability function (hp816x_set_PWM_stability) sets up a stability operation and starts the data acquisition.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized.
<i>PWMSlot</i>	For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).
	Data Type: ViSession Input/Output: IN

PWMSlot

Use the PWM Slot control to choose an installed Power Meter (a Power Sensor or Optical Head Interface Module).

Data Type: ViInt32
Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5

hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

channelNumber

If you choose a Dual Power Sensor Module or a Dual Optical Head Interface Module, you can use the Channel Number control to choose between Channel 1 and Channel 2 of your module.

If you choose a single-channel Power Meter, use the Channel Number to choose Channel 1.

Data Type: ViInt32

Input/Output: IN

Values:

<u>Name</u>	<u>Value</u>	<u>Description</u>
hp816x_CHAN_1	0	Channel 1
hp816x_CHAN_2	1	Channel 2

averagingTime

Use Averaging Time to set the length of time in seconds (s) over which a stability operation power measurement is averaged.

Data Type: ViReal64

Input/Output: IN

delayTime

Use the Period time to set the time in seconds between the beginning of each averaging cycle.

Data Type: ViReal64

Input/Output: IN

totalTime

Use Total Time to how long in seconds (s) the stability operation lasts.

Data Type: ViReal64

Input/Output: IN

estimatedResults

Estimated Results returns an estimate of the number of results that the stability operation will record.

Use Estimated Results to allocate a one-dimensional array with the same dimensions as this value and use the array as the Stability Result input for the Get PWM Stability Results function ([hp816x_get_PWM_stabilityResults_Q](#)).

Data Type: ViPlnt32

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "[hp816x_error_message](#)"

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_get_PWM_stabilityResults_Q

Syntax:

```
ViStatus _VI_FUNC hp816x_get_PWM_stabilityResults_Q(ViSession ihandle,
ViInt32 PWMSlot, ViInt32 channelNumber, ViBoolean waitforCompletion,
ViBoolean resultUnit, ViPBoolean stabilityStatus, ViReal64stabilityResult[ ]);
```

The Get PWM Stability Result function (hp816x_get_PWM_stabilityResults_Q) returns the result of a previously started stability operation.

REMARK The Wait for Completion control should only be set to Yes if the application is thoroughly tested because using this control may result in an infinite loop that blocks the calling function.

Also, if the function detects that logging is finished, the stability operation stops automatically.

<u>Parameter</u>	<u>Description</u>	
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized.	
<i>PWMSlot</i>	For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).	
	Data Type: ViSession Input/Output: IN	
<i>channelNumber</i>	Use the PWM Slot control to choose an installed Power Meter (a Power Sensor or Optical Head Interface Module).	
	Data Type: ViInt32 Input/Output: IN	
Values:		
<u>Name</u>	<u>Value</u>	<u>Description</u>
hp816x_SLOT_1	1	Slot 1

hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

channelNumber

If you choose a Dual Power Sensor Module or a Dual Optical Head Interface Module, you can use the Channel Number control to choose between Channel 1 and Channel 2 of your module.

If you choose a single-channel Power Meter, use the Channel Number to choose Channel 1.

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_CHAN_1	0	Channel 1
hp816x_CHAN_2	1	Channel 2

waitforCompletion

Use the Wait for Completion control to choose one of the following:

Yes (VI_TRUE), the stability operation will wait until the Total Time specified in the Set PWM Stability function has elapsed, or

No (VI_FALSE), the stability operation will stop

immediately.

Data Type: ViBoolean

Input/Output: IN

Values:

Name	Value	Description
VI_TRUE	(1)	Yes
VI_FALSE	(0)	No

resultUnit

Use the Result Unit control to choose power units of dBm (hp816x_PU_DBM) or Watts (hp816x_PU_WATT) for the stability operation result.

Data Type: ViBoolean

Input/Output: IN

Values:

Name	Value	Description
hp816x_PU_DBM	0	dBm
hp816x_PU_WATT	1	Watt

stabilityStatus

Stability Status returns a Boolean representing the current status of the stability operation.

If Stability Status equals 1, the operation is completed.

If Stability Status equals 0, the operation is being performed.

Data Type: ViPBoolean

Input/Output: OUT

stabilityResult

Stability Result is both an input and an output.

As an input, Stability Result accepts a one-

dimensional integer array that determines the size of the Stability Result output. Allocate a one-dimensional integer array with the same dimensions as the Estimated Results value you have set for the Set PWM Stability function ([hp816x_set_PWM_stability](#)).

As an output, Stability Result returns the result of a stability operation as a one-dimensional integer array.

Data Type: ViReal64[]

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "[hp816x_error_message](#)"

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_set_PWM_minMax

Syntax:

```
ViStatus _VI_FUNC hp816x_set_PWM_minMax(ViSession ihandle, ViInt32 PWMSlot, ViInt32 channelNumber, ViInt32 minmaxMode, ViInt32 dataPoints, ViUInt32 estimatedTime);
```

The Set PWM MinMax function (hp816x_set_PWM_minMax) sets the parameters for a MinMax logging operation.

A MinMax logging operation is active as long as it is not explicitly stopped. Before you start a different Power Meter data acquisition function or different MinMax logging operation, you must call the Stop PWM Function function.

<u>Parameter</u>	<u>Description</u>	
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized. For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).	
	Data Type: ViSession Input/Output: IN	
<i>PWMSlot</i>	Use the PWM Slot control to choose an installed Power Meter (a Power Sensor or Optical Head Interface Module). Data Type: ViInt32 Input/Output: IN	
Values:		
<u>Name</u>	<u>Value</u>	<u>Description</u>
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3

hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

channelNumber

If you choose a Dual Power Sensor Module or a Dual Optical Head Interface Module, you can use the Channel Number control to choose between Channel 1 and Channel 2 of your module.

If you choose a single-channel Power Meter, use the Channel Number to choose Channel 1.

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_CHAN_1	0	Channel 1
hp816x_CHAN_2	1	Channel 2

minmaxMode

Use the MinMax control to choose one of the following MinMax modes:

Continuous mode (hp816x_MM_CONT),

Window mode (hp816x_MM_WIN), or

Refresh mode (hp816x_MM_REFRESH).

See your instrument's User's Guide for

information on MinMax modes.

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_MM_CONT	0	Continuous
hp816x_MM_WIN	1	Window
hp816x_MM_REFRESH	2	Refresh

dataPoints

Use Data Points to define the number of samples used for the MinMax logging operation. Data Points affects Refresh mode and Window mode.

See your instrument's User's Guide for information on Data Points.

Data Type: ViInt32

Input/Output: IN

estimatedTime

Estimated Time returns the time, in seconds, it will take until a minimum or maximum value is available.

This value can be used to adjust the time before you can call the Get PWM MinMax Results function (get_PWM_minMaxResults_Q). This does not apply to Continuous MinMax mode.

Data Type: ViUInt32

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "[hp816x_error_message](#)"

E0615, Jun 2015



hp816x_get_PWM_minMaxResults_Q

Syntax:

```
ViStatus _VI_FUNC hp816x_get_PWM_minMaxResults_Q(ViSession ihandle,
Vilnt32 PWMSlot, Vilnt32 channelNumber, ViPReal64 minimum, ViPReal64
maximum, ViPReal64 current);
```

The Get PWM MinMax function (hp816x_get_PWM_minMax) returns the minimum, maximum and current values of a previously started MinMax operation.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized.
<i>PWMSlot</i>	For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).
	Data Type: ViSession Input/Output: IN

PWMSlot

Use the PWM Slot control to choose an installed Power Meter (a Power Sensor or Optical Head Interface Module).

Data Type: Vilnt32
Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5

hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

channelNumber

If you choose a Dual Power Sensor Module or a Dual Optical Head Interface Module, you can use the Channel Number control to choose between Channel 1 and Channel 2 of your module.

If you choose a single-channel Power Meter, use the Channel Number to choose Channel 1.

Data Type: ViInt32

Input/Output: IN

Values:

<u>Name</u>	<u>Value</u>	<u>Description</u>
hp816x_CHAN_1	0	Channel 1
hp816x_CHAN_2	1	Channel 2

minimum

Minimum returns the minimum power value in Watts for the MinMax operation.

Data Type: ViPReal64

Input/Output: OUT

maximum

Maximum returns the maximum power value in Watts for the MinMax operation.

Data Type: ViPReal64
Input/Output: OUT

current

Current returns the current power value in Watts.

Data Type: ViPReal64
Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_PWM_functionStop

Syntax:

```
ViStatus _VI_FUNC hp816x_PWM_functionStop(ViSession ihandle, ViInt32 PWMSlot, ViInt32 channelNumber);
```

The Stop PWM Function function (hp816x_PWM_functionStop) stops (aborts) any activated logging, stability or Min Max data aquistion operation.

Parameter

ihandle

Description

IHandle is the session handle returned from the Initialize Mainframe function ([hp816x_init](#)). A new session handle is created every time the mainframe is initialized.

For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function ([hp816x_init](#)).

Data Type: ViSession

Input/Output: IN

PWMSlot

Use the PWM Slot control to choose an installed Power Meter (a Power Sensor or Optical Head Interface Module).

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6

hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

channelNumber

If you choose a Dual Power Sensor Module or a Dual Optical Head Interface Module, you can use the Channel Number control to choose between Channel 1 and Channel 2 of your module.

If you choose a single-channel Power Meter, use the Channel Number to choose Channel 1.

Data Type: ViInt32

Input/Output: IN

Values:

<u>Name</u>	<u>Value</u>	<u>Description</u>
hp816x_CHAN_1	0	Channel 1
hp816x_CHAN_2	1	Channel 2

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)"

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_spectralCalibrationEx

Syntax:

```
ViStatus _VI_FUNC hp816x_spectralCalibrationEx(ViSession ihandle, ViInt32 Slot, ViInt32 channelNumber, ViUInt32 Size_of_Spectrum, ViReal64 Wavelength[], ViReal64 Power[], ViReal64 * Wavelength_Result, ViChar Error_Diagnose );
```

Use the Spectral Calibration function (hp816x_spectralCalibrationEx) to calibrate a power meter to an effective wavelength for a multiwavelength or broadband signal. For accurate power measurement of a multiwavelength or broadband signal, you can set the power meter to its effective wavelength (corresponding to the required weighted-average responsivity factor) for the signal. This extended "Ex" function allows selection of the channel number for dual-channel power meter modules.

Remark: This function only applies to power meters or Attenuator modules with power control, specifically the 81576A and 81577A. These modules use a filter technology with minimum wavelength dependence, but incorporate a power meter whose responsivity varies with wavelength.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized.

For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function ([hp816x_init](#)).

Data Type: ViSession
Input/Output: IN

Slot

Use the Slot control to specify the slot used to host the Attenuator module (or powermeter).

Data Type: ViInt32
Input/Output: IN

Values:

Name	Value	Description
------	-------	-------------

hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

channelNumber

If you choose a Dual Power Sensor Module or a Dual Optical Head Interface Module, you can use the Channel Number control to choose between Channel 1 and Channel 2 of your module.

If you choose a single-channel Power Meter, use the Channel Number to choose Channel 1.

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_CHAN_1	0	Channel 1
hp816x_CHAN_2	1	Channel 2

Size_of_Spectrum

Use the Size of Spectrum control to specifies the number of wavelength:power pairs in the multiwavelength DWDM signal.

Data Type: ViUInt32

Input/Output: IN

Wavelength

Input the array of wavelength values

Data Type: ViReal64[]

Input/Output: IN

Power

Input the array of power values in W

Data Type: ViReal64[]

Input/Output: IN

Wavelength_Result

Returns the effective wavelength corresponding to the required powermeter responsivity factor for the DWDM signal.

If 0.0 is returned, the spectral calibration failed.

Data Type: ViReal64

Input/Output: OUT

Error_Diagnose

Returns a diagnostic message, which may help you understand why a spectral calibration failed.

Data Type: ViChar

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)"

E0615, Jun 2015



hp816x_getWIRespTblSizeEx

Syntax:

```
ViStatus _VI_FUNC hp816x_getWIRespTblSizeEx (ViSession ihandle, ViInt32 Slot, ViInt32 channelNumber, ViUInt32 * Size, ViUInt32 * CSV_Size);
```

Use the Get Wavelength Resp. Table Size function (hp816x_getWIRespTblSizeEx) to query the size of the wavelength response table used to calibrate a power meter or an Attenuator module's built-in power meter.

Remark: This function only applies to power meters or Attenuator modules with power control, specifically the 81576A and 81577A. These modules use a filter technology with minimum wavelength dependence, but incorporate a power meter whose responsivity varies with wavelength. This extended "Ex" function allows selection of the channel number for dual-channel power meter modules.

Parameter

ihandle

Description

IHandle is the session handle returned from the Initialize Mainframe function ([hp816x_init](#)). A new session handle is created every time the mainframe is initialized.

For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function ([hp816x_init](#)).

Data Type: ViSession

Input/Output: IN

Slot

Use the Slot control to specify the slot used to host the power meter.

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3

hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

channelNumber

If you choose a Dual Power Sensor Module or a Dual Optical Head Interface Module, you can use the Channel Number control to choose between Channel 1 and Channel 2 of your module.

If you choose a single-channel Power Meter, use the Channel Number to choose Channel 1.

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_CHAN_1	0	Channel 1
hp816x_CHAN_2	1	Channel 2

Size

Returns the size of the power meter's wavelength response calibration table.

Remark: Use this value to allocate memory prior to using the Read Wavelength Response Table function ([hp816x_readWIRespTableEx](#)).

Data Type: ViUInt32

Input/Output: OUT

CSV_Size

Returns the number of bytes required by the power meter's wavelength response calibration table in CSV format.

Remark: Use this value to allocate memory prior to using the Read Wavelength Response Table function ([hp816x_readWlRepTblCSV_Ex](#)).

Data Type: ViUInt32

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "[hp816xerror_message](#)"

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_readWIRespTableEx

Syntax:

```
ViStatus _VI_FUNC hp816x_getWIRespTableEx (ViSession ihandle, ViInt32
Slot, ViInt32 channelNumber, ViReal64 Wavelength [], ViReal64
Response_Factor []);
```

Use the Read Wavelength Resp. Table function (hp816x_readWIRespTableEx) to read the wavelength response table used to calibrate a power meter or an Attenuator module's built-in power meter.

Use the Get Wavelength Resp. Table Size function ([hp816x_getWvlRespTblSizeEx](#)) to query the size of this table.

Remark: This function only applies to power meters or Attenuator modules with power control, specifically the 81576A and 81577A. These modules use a filter technology with minimum wavelength dependence, but incorporate a power meter whose responsivity varies with wavelength. This extended "Ex" function allows selection of the channel number for dual-channel power meter modules.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized. For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).
Slot	Data Type: ViSession Input/Output: IN
Slot	Use the Slot control to specify the slot used to host the power meter. Data Type: ViInt32 Input/Output: IN

Values:

<u>Name</u>	<u>Value</u>	<u>Description</u>
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

channelNumber

If you choose a Dual Power Sensor Module or a Dual Optical Head Interface Module, you can use the Channel Number control to choose between Channel 1 and Channel 2 of your module.

If you choose a single-channel Power Meter, use the Channel Number to choose Channel 1.

Data Type: ViInt32

Input/Output: IN

Values:

<u>Name</u>	<u>Value</u>	<u>Description</u>
hp816x_CHAN_1	0	Channel 1
hp816x_CHAN_2	1	Channel 2

Wavelength

Wavelength is both an input and an output.

As an input, use a Real64 array with size given by the Size result from the hp816x_getWRespTblSizeEx query.

Returns the wavelength entries from the powermeter's wavelength response calibration

table.

Data Type: ViReal64[]

Input/Output: OUT

Response_Factor

Response_Factor is both an input and an output.

As an input, use a Real64 array with size given by the Size result from the hp816x_getWRespTblSizeEx query.

Returns the response factor entries from the powermeter's wavelength response calibration table.

Data Type: ViReal64[]

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816xerror_message"](#)"

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_readWIRepTblCSV_Ex

Syntax:

```
ViStatus _VI_FUNC hp816x_getWIRepTblCSV_Ex (ViSession ihandle, ViInt32 Slot, ViInt32 channelNumber, ViChar CSV_List []);
```

Use the Read WI Rep. Table CSV function (hp816x_readWIRepTblCSV_Ex) to read the wavelength response table used to calibrate a power meter or an Attenuator module's built-in power meter in CSV format.

Use the Get Wavelength Resp. Table Size function ([hp816x_getWvlRespTblSizeEx](#)) to query the size of this table.

Remark: This function only applies to power meters or Attenuator modules with power control, specifically the 81576A and 81577A. These modules use a filter technology with minimum wavelength dependence, but incorporate a power meter whose responsivity varies with wavelength.

Parameter

ihandle

Description

IHandle is the session handle returned from the Initialize Mainframe function ([hp816x_init](#)). A new session handle is created every time the mainframe is initialized.

For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function ([hp816x_init](#)).

Data Type: ViSession

Input/Output: IN

Slot

Use the ATT Slot control to specify the slot used to host the Attenuator module.

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_1	1	Slot 1

hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

channelNumber

If you choose a Dual Power Sensor Module or a Dual Optical Head Interface Module, you can use the Channel Number control to choose between Channel 1 and Channel 2 of your module.

If you choose a single-channel Power Meter, use the Channel Number to choose Channel 1.

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_CHAN_1	0	Channel 1
hp816x_CHAN_2	1	Channel 2

CSV_List

CSV_List is both an input and an output.

As an input, use a text string with length given by the CSV_Size result from the hp816x_getWRespTblSizeEx query.

Returns the wavelength, response_factor entries from the powermeter's wavelength response calibration table in CSV format.

Data Type: ViChar[]

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "[hp816xerror_message](#)"

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_set_FLS_parameters

Syntax:

```
ViStatus _VI_FUNC hp816x_set_FLS_parameters(ViSession ihandle, ViInt32
FLSSlot, ViInt32 wavelengthSource, ViBoolean turnLaser, ViInt32
modulationLowerSource, ViInt32 modulationUpperSource, ViReal64
modulationFreqLowerSource, ViReal64 modulationFreqUpperSource, ViReal64
attenuationLowerSource, ViReal64 attenuationUpperSource);
```

The Set FLS Parameters function (hp816x_set_FLS_parameters) enables you to set up Laser Sources quickly. In case of a Single-Wavelength Laser Source Module, the inputs for the lower source will be used.

Remark: For dual-wavelength Laser Sources with part numbers less than 81600, the modulation settings for the upper source will be ignored; only the lower source modulation settings are valid.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized.

For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function ([hp816x_init](#)).

Data Type: ViSession
Input/Output: IN

<i>FLSSlot</i>	Use the FLS Slot control to choose an installed Fixed Laser Source Module.
----------------	--

Data Type: ViInt32
Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3

hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

wavelengthSource

For a Laser Source Module, you can use the Wavelength Source control to select one of the following:

Lower (hp816x_LOWER_SRC), the shorter wavelength,

Upper (hp816x_UPPER_SRC), the longer wavelength, or

Both (hp816x_BOTH_SRC), both the Upper and Lower wavelengths.

This control is ignored for a Single-Wavelength Laser Source Module.

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_LOWER_SRC	0	Lower
hp816x_UPPER_SRC	1	Upper
hp816x_BOTH_SRC	2	Both

turnLaser

Use the Turn Laser On/Off control to turn the Laser Source On (VI_TRUE) or Off (VI_FALSE).

Data Type: ViBoolean

Input/Output: IN

Values:

Name	Value	Description
VI_TRUE	(1)	On
VI_FALSE	(0)	Off

modulationLowerSource

Use the Modulation Lower Source control to select the modulation source for a single-wavelength Fixed Laser Source. You can choose one of the following options:

Off (hp816x_AM_OFF) turns the modulation off;

Internal (hp816x_AM_INT), internal digital modulation, or

Coherence Control (hp816x_AM_CC), coherence control.

See your instrument's User's Guide for more information on modulation sources.

NOTE:

Coherence control is not available for modules with product IDs less than 81600.

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_AM_OFF	0	Off
hp816x_AM_INT	1	Internal
hp816x_AM_CC	2	Coherence Control

modulationUpperSource

Use the Modulation Upper Source control to select the modulation source for the upper wavelength source:

Off (hp816x_AM_OFF) turns the modulation off,

Internal (hp816x_AM_INT), internal digital modulation, or

Coherence Control (hp816x_AM_CC), coherence control.

See your instrument's User's Guide for more information on modulation sources.

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_AM_OFF	0	Off
hp816x_AM_INT	1	Internal
hp816x_AM_CC	2	Coherence Control

modulationFreqLowerSource

Use the Modulation Freq Lower Source control to set the modulation frequency to a value in Hertz (Hz) for the lower wavelength source.

This input sets the modulation frequency for a single-wavelength Laser Source.

Data Type: ViReal64

Input/Output: IN

modulationFreqUpperSource

Use this Modulation Freq Upper Source control to set the modulation frequency to a value in Hertz (Hz) for the upper wavelength source.

Data Type: ViReal64

Input/Output: IN

attenuationLowerSource

Use the Attenuation Lower Source control to set the attenuation to a value in decibels (dB) for the lower wavelength source.

Data Type: ViReal64

Input/Output: IN

attenuationUpperSource

The Attenuation Upper Source input sets the attenuation to a value in decibels (dB) for the upper wavelength Laser Source.

This control only applies to dual-wavelength Laser Source.

Data Type: ViReal64

Input/Output: IN

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)"

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_get_FLS_parameters_Q

Syntax:

```
ViStatus _VI_FUNC hp816x_get_FLS_parameters_Q(ViSession ihandle,
ViInt32 FLSSlot, ViPInt32 wavelengthSource, ViPBoolean turnLaser, ViPInt32
modulationLowerSource, ViPInt32 modulationUpperSource, ViPReal64
modulationFreqLowerSource, ViPReal64 modulationFreqUpperSource, ViPReal64
attenuationLowerSource, ViPReal64 attenuationUpperSource);
```

The Set FLS Parameters function ([hp816x_set_FLS_parameters](#)) enables you to set up Laser Sources quickly. In case of a Laser Source with a single optical output, the inputs for the lower source will be used.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized.

For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function ([hp816x_init](#)).

Data Type: ViSession
Input/Output: IN

FLSSlot

Use the FLS Slot control to choose an installed Fixed Laser Source Module.

Data Type: ViInt32
Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4

hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

wavelengthSource

Wavelength Source returns the selected wavelength source as one of the following:

0 (hp816x_LOWER_SRC), the shorter wavelength, or

1 (hp816x_UPPER_SRC), the longer wavelength, or

2 (hp816x_BOTH_SRC), both the Upper and Lower wavelengths.

For a Single-Wavelength Laser Source Module, this control is ignored.

Data Type: ViPInt32

Input/Output: OUT

turnLaser

Turn Laser returns the laser state as one of the following:

0 (VI_TRUE), Laser Source On, or

1 (VI_FALSE), Laser Source Off.

Data Type: ViPBoolean

Input/Output: OUT

modulationLowerSource

Modulation Lower Source returns the one of the following modulation sources for a single-wavelength Fixed Laser Source and for both upper and lower output signals of the 81554SM dual-wavelength Fixed Laser Source.

1 (hp816x_AM_INT), internal digital modulation,
or

2 (hp816x_AM_CC), coherence control.

Modulation Lower Source returns the modulation source for a single-wavelength Fixed Laser Source and for both upper and lower output signals of the 81554SM dual-wavelength Fixed Laser Source.

See your instrument's User's Guide for more information on modulation sources.

NOTE:

Coherence control is not available for modules with product IDs less than 81600.

Data Type: ViPInt32

Input/Output: OUT

modulationUpperSource

Modulation Upper Source returns the modulation source for the upper wavelength source:

1 (hp816x_AM_INT), internal digital modulation,
or

2 (hp816x_AM_CC), coherence control.

See your instrument's User's Guide for more information on modulation sources.

Data Type: ViPInt32
Input/Output: OUT

modulationFreqLowerSource

Modulation Freq Lower Source returns the modulation frequency as a value in Hertz (Hz) for the lower wavelength source.

Modulation Freq Lower Source returns the modulation frequency for a single-wavelength Fixed Laser Source and for both upper and lower output signals of than 81554SM dual-wavelength Fixed Laser Source.

Data Type: ViPReal64
Input/Output: OUT

modulationFreqUpperSource

Modulation Freq Upper Source returns the modulation frequency as a value in Hertz (Hz) for the upper wavelength Source.

This control only applies to dual-wavelength Fixed Laser Source product numbers greater than 81600.

Data Type: ViPReal64
Input/Output: OUT

attenuationLowerSource

Attenuation Lower Source returns the attenuation as a value in decibels (dB) for the lower wavelength source.

Data Type: ViPReal64
Input/Output: OUT

attenuationUpperSource

Attenuation Upper Source returns the attenuation level as a value in decibels (dB) for the upper wavelength Laser Source.

This control only applies to dual-wavelength Laser Source.

Data Type: ViPReal64

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_set_FLS_modulation

Syntax:

```
ViStatus _VI_FUNC hp816x_set_FLS_modulation(ViSession ihandle, ViInt32  
FLSSlot, ViInt32 laserSource, ViInt32 modulationFrequency, ViInt32  
modulationSource, ViReal64 manualFrequency);
```

The Setup FLS Modulation function (hp816x_set_FLS_modulation) adjusts the internal modulation of a Fixed Laser Source.

Internal Modulation is described in your instrument's User's Guide.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized. For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).
	Data Type: ViSession Input/Output: IN
<i>FLSSlot</i>	

FLSSlot
Use the FLS Slot control to choose an installed Fixed Laser Source Module.

Data Type: ViInt32
Input/Output: IN

Values:

<u>Name</u>	<u>Value</u>	<u>Description</u>
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5

hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

laserSource

For a dual-wavelength Laser Source Module, you can use the Wavelength Source control to select the longer wavelength, Upper (hp816x_UPPER_SRC), or the shorter wavelength, Lower (hp816x_LOWER_SRC).

For a single-wavelength Laser Source Module, the default is Lower.

Data Type: VlInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_LOWER_SRC	0	Lower
hp816x_UPPER_SRC	1	Upper

modulationFrequency

You can use the Modulation Frequency control to select one of the following:

Minimum (hp816x_AM_MIN), which is the minimum frequency,

Maximum (hp816x_AM_MAX), which is the maximum frequency,

Default (hp816x_AM_DEFAULT), which is the sum of the minimum frequency and the maximum frequency divided by two (this is not the same value that the Reset function, the Preset function, or the Preset hardkey sets the frequency to), and

Manual Input (hp816x_AM_MANUAL), which is the value entered in the Manual Frequency input.

Data Type: VlInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_AM_MIN	0	Minimum
hp816x_AM_MAX	1	Maximum
hp816x_AM_DEFAULT	2	Default
hp816x_AM_MANUAL	3	Manual Input

modulationSource

Use Modulation Source to choose one of the following modulation sources:

Off (hp816x_AM_OFF) turns the modulation off;

Internal (hp816x_AM_INT), internal digital modulation, or

Coherence Control (hp816x_AM_CC), coherence control.

See your instrument's User's Guide for more information on modulation sources.

ATTENTION:

Coherence Control is only supported by modules with a product number greater than 81600.

Data Type: VlInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_AM_OFF	0	Off
hp816x_AM_INT	1	Internal
hp816x_AM_CC	2	Coherence Control

manualFrequency

If you choose Manual Input as the Modulation Frequency input, you can use the Manual Frequency input to set the modulation frequency to a value in Hertz (Hz).

Data Type: ViReal64

Input/Output: IN

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)"

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_get_FLS_modulation_Q

Syntax:

```
ViStatus _VI_FUNC hp816x_get_FLS_modulationSettings_Q(ViSession
ihandle, ViInt32 FLSSlot, ViInt32 wavelengthSource, ViPBoolean
modulationState, ViPIInt32 modulationSource, ViPReal64 minimumFrequency,
ViPReal64 maximumFrequency, ViPReal64 defaultFrequency, ViPReal64
currentFrequency);
```

The Get FLS Modulation Settings function (hp816x_get_FLS_modulationSettings_Q) returns the current values of the following: Modulation State, Minimum Frequency, Maximum Frequency, Default Frequency, and Current Frequency.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized.
<i>FLSSlot</i>	For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).

Data Type: ViSession
Input/Output: IN

FLSSlot

Use the FLS Slot control to choose an installed Fixed Laser Source Module.

Data Type: ViInt32
Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2

hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

wavelengthSource

For a dual-wavelength Laser Source Module, you can use the Wavelength Source control to select the longer wavelength, Upper (hp816x_UPPER_SRC), or the shorter wavelength, Lower (hp816x_LOWER_SRC).

For a single-wavelength Laser Source Module, the default is Lower.

An error is returned for modules with part numbers lower than 81600 if you choose Upper.

Data Type: Vilnt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_LOWER_SRC	0	Lower
hp816x_UPPER_SRC	1	Upper

modulationState

Modulation returns whether the output signal is modulated:

1 (hp816x_MOD_ENABLED), if modulation is enabled, or

0 (hp816x_MOD_DISABLED), if modulation is disabled.

Data Type: ViPBoolean

Input/Output: OUT

modulationSource

Modulation Source returns the chosen modulation source as one of the following:

1 (hp816x_MOD_INT): Internal digital modulation is chosen, or

2 (hp816x_MOD_CC): Coherence Control is chosen.

See your instrument's User's Guide for more information on modulation sources.

Data Type: ViPInt32

Input/Output: OUT

minimumFrequency

Minimum Frequency returns the lowest frequency that your Fixed Laser Source can be set to.

Frequency is returned in units of Hertz (Hz).

Data Type: ViReal64

Input/Output: OUT

maximumFrequency

Maximum Frequency returns the highest frequency that your Fixed Laser Source can be set to.

Frequency is returned in units of Hertz (Hz).

Data Type: ViReal64

Input/Output: OUT

defaultFrequency

Default Frequency returns the default frequency of your Fixed Laser Source. This is identical to the default value set by setting the Modulation Frequency control of the Set FLS Modulation function to Default; it is the sum of the Minimum Frequency and the Maximum Frequency divided by two. This is not the same value that the Reset function, the Preset function, or the Preset hardkey sets the frequency to.

Frequency is returned in units of Hertz (Hz).

Data Type: ViReal64

Input/Output: OUT

currentFrequency

Current Frequency returns the modulation frequency that your Fixed Laser Source can be set to..

Frequency is returned in units of Hertz (Hz).

Data Type: ViReal64

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_set_FLS_laserSource

Syntax:

```
ViStatus _VI_FUNC hp816x_set_FLS_laserSource(ViSession ihandle, ViInt32
FLSSlot, ViInt32 laserSource);
```

The Set FLS Laser Source function (hp816x_set_FLS_laserSource) chooses, which laser output, of a dual-wavelength Fixed Laser Source, is output.

This function is irrelevant for single-wavelength Fixed Laser Sources.

Parameter

ihandle

Description

IHandle is the session handle returned from the Initialize Mainframe function ([hp816x_init](#)). A new session handle is created every time the mainframe is initialized.

For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function ([hp816x_init](#)).

Data Type: ViSession

Input/Output: IN

FLSSlot

Use the FLS Slot control to choose an installed Fixed Laser Source Module.

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5

hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

/laserSource

For a Laser Source Module, you can use the Laser Source control to select one of the following:

Upper (hp816x_UPPER_SRC), the longer wavelength, Lower (hp816x_LOWER_SRC), the shorter wavelength, or Both (hp816x_BOTH_SRC), both the Upper and Lower wavelengths.

For a Single-Wavelength Laser Source Module, the default is Lower.

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_LOWER_SRC	0	Lower
hp816x_UPPER_SRC	1	Upper
hp816x_BOTH_SRC	2	Both

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)"

■

hp816x_get_FLS_laserSource_Q

Syntax:

```
ViStatus _VI_FUNC hp816x_get_FLS_laserSource_Q(ViSession ihandle,
Vilnt32 FLSSlot, ViPInt32 laserSource);
```

The Get Laser Source function (hp816x_get_FLS_laserSource_Q) returns, for dual-wavelength Fixed Laser Sources, which wavelength is switched to the output.

This function is irrelevant for a single-wavelength Fixed Laser Source.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized.
<i>FLSSlot</i>	For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).
	Data Type: ViSession Input/Output: IN

FLSSlot

Use the FLS Slot control to choose an installed Fixed Laser Source Module.

Data Type: Vilnt32
Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5

hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

/laserSource

Laser Source returns one of the following values:

0 (hp816x_LOWER_SRC): the shorter wavelength is the selected output.

1 (hp816x_UPPER_SRC): the longer wavelength is the selected output.

2 (hp816x_L_SRC_BOTH): both wavelengths are output simultaneously.

Data Type: ViPInt32

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)"

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_get_FLS_wavelength_Q

Syntax:

```
ViStatus _VI_FUNC hp816x_get_FLS_wavelength_Q(ViSession ihandle,
ViInt32 FLSSlot, ViPReal64 wavelengthLowerSource, ViPReal64
wavelengthUpperSource);
```

The Get FLS Wavelength function (hp816x_get_FLS_wavelength_Q) returns the wavelength of a Fixed Laser Source.

REMARK:

Wavelength Upper Source returns 0.0 if a single-wavelength Fixed Laser Source is installed.

Parameter

ihandle

Description

IHandle is the session handle returned from the Initialize Mainframe function ([hp816x_init](#)). A new session handle is created every time the mainframe is initialized.

For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function ([hp816x_init](#)).

Data Type: ViSession

Input/Output: IN

FLSSlot

Use the FLS Slot control to choose an installed Fixed Laser Source Module.

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2

hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

wavelengthLowerSource

Wavelength Lower Source returns the wavelength of a single-wavelength Fixed Laser Source or the shorter wavelength of a dual-wavelength Fixed Laser Source.

Wavelength is returned in meters (m).

Data Type: ViPReal64

Input/Output: OUT

wavelengthUpperSource

Wavelength Upper Source returns the upper wavelength of a dual-wavelength Fixed Laser Source.

Wavelength is returned in meters (m).

The value, 0, is returned for a single-wavelength Fixed Laser Source.

Data Type: ViPReal64

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)"

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_set_FLS_attenuation

Syntax:

```
ViStatus _VI_FUNC hp816x_set_FLS_attenuation(ViSession ihandle, ViInt32
```

```
FLSSlot, ViReal64 attenuationLowerSource, ViReal64 attenuationUpperSource);
```

The Set FLS Attenuation function (hp816x_set_FLS_attenuation) sets the attenuation of a Fixed Laser Source.

Parameter

ihandle

Description

IHandle is the session handle returned from the Initialize Mainframe function ([hp816x_init](#)). A new session handle is created every time the mainframe is initialized.

For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function ([hp816x_init](#)).

Data Type: ViSession

Input/Output: IN

FLSSlot

Use the FLS Slot control to choose an installed Fixed Laser Source Module.

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7

hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

attenuationLowerSource

Attenuation Lower Source sets the attenuation of a single-wavelength Fixed Laser Source or the attenuation of the shorter wavelength source of a dual-wavelength Fixed Laser Source.

Attenuation is set in units of decibels (dB).

Data Type: ViReal64

Input/Output: IN

attenuationUpperSource

Attenuation Upper Source sets the attenuation of the longer wavelength source of a dual-wavelength Fixed Laser Source.

Attenuation is set in units of decibels (dB).

Data Type: ViReal64

Input/Output: IN

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)"

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_get_FLS_attenuation_Q

Syntax:

```
ViStatus _VI_FUNC hp816x_get_FLS_attenuation_Q(ViSession ihandle,
ViInt32 FLSSlot, ViPReal64 attenuationLowerSource, ViPReal64
attenuationUpperSource);
```

The Get FLS Attenuation function (hp816x_get_FLS_attenuation_Q) returns the set attenuation of a Fixed Laser Source.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized.
<i>FLSSlot</i>	For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).
	Data Type: ViSession Input/Output: IN

FLSSlot

Use the FLS Slot control to choose an installed Fixed Laser Source Module.

Data Type: ViInt32
Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6

hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

attenuationLowerSource

Attenuation Lower Source returns the set attenuation of a single-wavelength Fixed Laser Source or the attenuation of the shorter wavelength source of a dual-wavelength Fixed Laser Source.

Attenuation is returned in units of decibels (dB).

Data Type: ViPReal64

Input/Output: OUT

attenuationUpperSource

Attenuation Upper Source returns the set attenuation of the longer wavelength source of a dual-wavelength Fixed Laser Source.

0.0 is returned for a single-wavelength Fixed Laser Source.

Attenuation is returned in units of decibels (dB).

Data Type: ViPReal64

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)"

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_set_FLS_triggerState

Syntax:

```
ViStatus _VI_FUNC hp816x_set_FLS_triggerState(ViSession ihandle, ViInt32
FLSSlot, ViBoolean outputTrigger);
```

The Set FLS Trigger State (hp816x_set_FLS_triggerState) function specifies when an output trigger is generated.

You can enable an output trigger to be generated for every leading edge of a digitally-modulated TTL signal.

Parameter

ihandle

Description

IHandle is the session handle returned from the Initialize Mainframe function ([hp816x_init](#)). A new session handle is created every time the mainframe is initialized.

For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function ([hp816x_init](#)).

Data Type: ViSession

Input/Output: IN

FLSSlot

Use the FLS Slot control to choose an installed Fixed Laser Source Module.

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5

hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

outputTrigger

Use Output Trigger to enable an output trigger to be generated for every leading edge of a digitally-modulated TTL signal.

Choose Enable (VI_TRUE) to enable an output trigger to be generated for every leading edge of a digitally-modulated TTL signal.

Choose Disable (VI_FALSE) to disable output trigger for the chosen Laser Source.

Data Type: ViBoolean

Input/Output: IN

Values:

Name	Value	Description
VI_TRUE	(1)	Enable
VI_FALSE	(0)	Disable

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)"

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_get_FLS_laserState_Q

Syntax:

```
ViStatus _VI_FUNC hp816x_get_FLS_laserState_Q(ViSession ihandle, ViInt32
FLSSlot, ViBoolean outputTrigger);
```

The Get FLS Trigger State (hp816x_get_FLS_triggerState) function returns whether an output trigger is generated for every leading edge of a digitally-modulated TTL signal.

Parameter

ihandle

Description

IHandle is the session handle returned from the Initialize Mainframe function ([hp816x_init](#)). A new session handle is created every time the mainframe is initialized.

For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function ([hp816x_init](#)).

Data Type: ViSession

Input/Output: IN

FLSSlot

Use the FLS Slot control to choose an installed Fixed Laser Source Module.

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6

hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

outputTrigger

Output Trigger returns whether an output trigger is generated for every leading edge of a digitally-modulated TTL signal.

If 0 (VI_FALSE) is returned, an output trigger is not generated for every leading edge of a digitally-modulated TTL signal.

If 1 (VI_TRUE) is returned, an output trigger is generated for every leading edge of a digitally-modulated TTL signal.

Data Type: ViBoolean

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_get_FLS_power

Syntax:

```
ViStatus _VI_FUNC hp816x_get_FLS_power(ViSession ihandle, ViInt32
FLSSlot, ViInt32 laserSource, ViPReal64 laserPower);
```

The Get FLS Power function (hp816x_get_FLS_power_Q) returns the output power of a fixed-wavelength Laser Source in Watts.

Parameter

ihandle

Description

IHandle is the session handle returned from the Initialize Mainframe function ([hp816x_init](#)). A new session handle is created every time the mainframe is initialized.

For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function ([hp816x_init](#)).

Data Type: ViSession

Input/Output: IN

FLSSlot

Use the FLS Slot control to choose an installed Fixed Laser Source Module.

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7

hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

laserSource

For a Laser Source Module, you can use the Laser Source control to select one of the following:

Upper (hp816x_UPPER_SRC), the longer wavelength source, or Lower (hp816x_LOWER_SRC), the shorter wavelength source.

For a Single-Wavelength Laser Source Module, the default is Lower.

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_LOWER_SRC	0	Lower
hp816x_UPPER_SRC	1	Upper

laserPower

Laser Power returns the the laser beam power of the current selected Laser Source in Watts.

Data Type: ViPReal64

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "[hp816x_error_message](#)"

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_set_FLS_laserState

Syntax:

```
ViStatus _VI_FUNC hp816x_set_FLS_laserState(ViSession ihandle, ViInt32
FLSSlot, ViBoolean laserState);
```

The Set FLS Laser State function (hp816x_set_FLS_laserState) turns the laser on or off.

Parameter

ihandle

Description

IHandle is the session handle returned from the Initialize Mainframe function ([hp816x_init](#)). A new session handle is created every time the mainframe is initialized.

For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function ([hp816x_init](#)).

Data Type: ViSession

Input/Output: IN

FLSSlot

Use the FLS Slot control to choose an installed Fixed Laser Source Module.

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7

hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

laserState

Use the Laser State control to turn the laser output On (VI_TRUE) or Off (VI_FALSE).

Data Type: ViBoolean

Input/Output: IN

Values:

Name	Value	Description
VI_TRUE	(1)	On
VI_FALSE	(0)	Off

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_get_FLS_laserState_Q

Syntax:

```
ViStatus _VI_FUNC hp816x_get_FLS_laserState_Q(ViSession ihandle, ViInt32
FLSSlot, ViPBoolean laserState);
```

The Get FLS Laser State function (hp816x_get_FLS_laserState_Q) returns whether the laser is turned on or off.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized.
<i>FLSSlot</i>	For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).
	Data Type: ViSession Input/Output: IN

FLSSlot

Use the FLS Slot control to choose an installed Fixed Laser Source Module.

Data Type: ViInt32
Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7

hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

laserState

Laser State returns whether the laser is turned on or off.

If 0 (VI_FALSE) is returned, the laser is turned off.

If 1 (VI_TRUE) is returned, the laser is turned on.

Data Type: ViPBoolean

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)"

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



Information for N77 Series Instruments

This version of the 816x VXIplug&play instrument driver has been extended to support the N773xA, N774xA optical power meter and the N775xA and N776xA Optical Attenuator instruments. These are stand-alone instruments that do not require installation in an 816x mainframe. The SCPI command set has been designed for almost complete programming compatibility with the modular instruments of the 816x family. Please be aware of the following details and limitations, when using this driver.

- The N77-series instruments can be controlled via USB, LAN or GPIB interfaces. Especially for transferring large data volumes, the USB and usually LAN interfaces provide faster performance.
- The identification of the individual ports of these multiport instruments in the SCPI and hp816x functions corresponds to the slot number. So for example the ports of the N7745A can be addressed with the slot numbers 1 to 8. The channel number in the commands and functions is not used and can generally be omitted or set to Channel 1 (index 0).
- The N771x-series and 81950A tunable lasers are not supported by the 816x VXIplug&play driver.
- The N773x-series optical switches are supported by the 816x VXIplug&play functions for switches and relevant functions for the mainframes.
- The N774x-series power meters are supported by the 816x VXIplug&play functions for power meters, relevant functions for the mainframes like for standard trigger configuration, and the Multi Frame Lambda Scan application functions.
- The N776x-series optical attenuators are supported by the 816x VXIplug&play functions for attenuators and the relevant functions for the mainframes.
- The N775x-series optical attenuators and power meters are supported by the 816x VXIplug&play functions for attenuators and the relevant functions for the mainframes and the basic functions for the power meters, excluding the advanced functions using data power logging. The choices of averaging time is also more limited for these power meters.
- The averaging time parameter for the power meters is limited in the 816x VXIplug&play functions to a limited set of discrete values, similar to the modular 816x power meters. This set includes: 1, 2, 5, 10, 25, 100, 200, and 500 μ s, 1, 2, 5, 10, 20, 50, 100, 200, 500 ms, and 1, 2, 5, and 10s.
- The maximum number of logging data points that can be acquired is limited by the 816x VXIplug&play data acquisition functions to 100,000 per port.
- The shortest averaging time used by the Multi Frame Lambda Scan application is 25 μ s.
- For the Multi Frame Lambda Scan application, the N774xA instruments can be included simply by registering the instrument as a mainframe, after the first

mainframe with the tunable laser has been registered.

Further details to available functions are found under the categories below:

[Utility Functions](#)

[Mainframe Specific Functions](#)

[Power Sensor Specific Functions](#)

[Applications](#)

hp816x_get_SWT_type

Syntax:

```
ViStatus _VI_FUNC hp816x_get_SWT_type(ViSession ihandle, ViInt32
SWT_Slot, ViPInt32 * Switch_Type, ViChar Switch_Description );
```

The Get Switch Type function (hp816x_get_SWT_type) returns a number and description corresponding to the switch module hosted by a specified slot.

Parameter

ihandle

Description

IHandle is the session handle returned from the Initialize Mainframe function ([hp816x_init](#)). A new session handle is created every time the mainframe is initialized.

For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function ([hp816x_init](#)).

Data Type: ViSession

Input/Output: IN

SWT_Slot

Use the SWT Slot control to specify the slot used to host the Switch module.

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6

hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

Switch_Type

Switch Type returns an integer corresponding to the type of switch module hosted in SWT Slot.

Data Type: ViInt32

Input/Output: OUT

Switch_Description

Switch Description returns a textual description of the specified switch module, including:

```
"1:2 switch module"
"2*1:2 switch module"
"2x2 switch module"
"1:4 switch module"
"2:4 switch module"
"1:8 switch module"
"unknown"
```

Data Type: ViChar

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)"

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_set_SWT_route

Syntax:

```
ViStatus _VI_FUNC hp816x_set_SWT_route(ViSession ihandle, ViInt32
SWT_Slot, ViPInt32 Input, ViInt32 Output );
```

Use the Set Route function (hp816x_set_SWT_route) to set the path between port A and another numbered port within a switch module and, if applicable, port B and another numbered port.

Valid paths depend on the type of switch module. Refer to the Get Switch Type function ([hp816x_get_SWT_type](#)) and the Get Route Table function ([hp816x_get_SWT_routeTable](#)).

Remarks: Although ports A and B are referred to as inputs, and numbered ports as outputs, the paths set are bi-directional.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized. For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).

Data Type: ViSession
Input/Output: IN

SWT_Slot

Use the SWT Slot control to specify the slot used to host the Switch module.

Data Type: ViInt32
Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2

hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

Input

Use the Input control to choose either port A or, if applicable to the switch type, port B.

Data Type: Vilnt32

Input/Output: IN

Values:

<u>Name</u>	<u>Value</u>	<u>Description</u>
hp816x_SWT_INP_A	0	Port A
hp816x_SWT_INP_B	1	Port B

Output

Use the Output control to choose the numbered port to be connected to the port specified by the Input control.

Remarks: If the switch is a dual channel device, the channels are entirely separate. So, port A can be connected to port 1 and port 2, and port B can be connected to another port 1 and another port 2.

Data Type: Vilnt32

Input/Output: IN

Values:

<u>Name</u>	<u>Value</u>	<u>Description</u>
hp816x_SWT_OUT_1	0	Port 1
hp816x_SWT_OUT_2	1	Port 2
hp816x_SWT_OUT_3	2	Port 3
hp816x_SWT_OUT_4	3	Port 4
hp816x_SWT_OUT_5	4	Port 5
hp816x_SWT_OUT_6	5	Port 6
hp816x_SWT_OUT_7	6	Port 7
hp816x_SWT_OUT_8	7	Port 8

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)"

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_get_SWT_route

Syntax:

```
ViStatus _VI_FUNC hp816x_get_SWT_route(ViSession ihandle, ViInt32
SWT_Slot, ViPInt32 Input, ViInt32 * Output );
```

Use the Get Route function (hp816x_get_SWT_route) to query the path between port A and another numbered port within a switch module and, if applicable, port B and another numbered port.

Remarks: Although ports A and B are referred to as inputs, and numbered ports as outputs, the paths set are bi-directional.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized.
For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).	
Data Type: ViSession Input/Output: IN	

SWT_Slot

Use the SWT Slot control to specify the slot used to host the Switch module.

Data Type: ViInt32
Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4

hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

Input

Use the Input control to choose either port A or, if applicable to the switch type, port B.

Use the Get Switch Type function ([hp816x_get_SWT_type](#)) to query the switch type hosted by a specified slot.

Data Type: VlInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_SWT_INP_A	0	Port A
hp816x_SWT_INP_B	1	Port B

Output

Returns the number of the port connected to the port specified by the Input control.

Remarks: If the switch is a dual channel device, the channels are entirely separate. So, port A can be connected to port 1 and port 2, and port B can be connected to another port 1 and another port 2.

Data Type: VlInt32

Input/Output: OUT

Values:

<u>Name</u>	<u>Value</u>	<u>Description</u>
hp816x_SWT_OUT_1	0	Port 1
hp816x_SWT_OUT_2	1	Port 2
hp816x_SWT_OUT_3	2	Port 3
hp816x_SWT_OUT_4	3	Port 4
hp816x_SWT_OUT_5	4	Port 5
hp816x_SWT_OUT_6	5	Port 6
hp816x_SWT_OUT_7	6	Port 7
hp816x_SWT_OUT_8	7	Port 8

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)"

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_get_SWT_routeTable

Syntax:

```
ViStatus _VI_FUNC hp816x_get_SWT_routeTable (ViSession iHandle, ViInt32 SWT_Slot, ViPInt32 Input, ViChar RouteTable [ ]);
```

Use the Get Route Table function (hp816x_get_SWT_routeTable) to query the paths allowed between a specified switch module's ports.

Remarks: Although ports A and B are referred to as inputs, and numbered ports as outputs, the allowed paths are bi-directional.

<u>Parameter</u>	<u>Description</u>
<i>iHandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized.

For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function ([hp816x_init](#)).

Data Type: ViSession

Input/Output: IN

SWT_Slot

Use the SWT Slot control to specify the slot used to host the Switch module.

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4

hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

Input

Use the Input control to choose either port A or, if applicable to the switch type, port B.

Use the Get Switch Type function ([hp816x_get_SWT_type](#)) to query the switch type hosted by a specified slot.

Data Type: VlInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_SWT_INP_A	0	Port A
hp816x_SWT_INP_B	1	Port B

Route_Table

Returns the route table for the specified switch module as an array of strings.

Remarks: If the switch is a dual channel device, the channels are entirely separate. So, port A can be connected to port 1 and port 2, and port B can be connected to another port 1 and another port 2.

Data Type: VlInt32

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "[hp816x_error_message](#)"

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_set_RLM_parameters

Syntax:

```
ViStatus _VI_FUNC hp816x_set_RLM_parameters(ViSession ihandle, ViInt32 RLMSlot,
ViBoolean internalTrigger, ViReal64 wavelength, ViReal64 averagingTime, ViBoolean
laserSource, ViBoolean laserState);
```

The Set RLM Parameters function (hp816x_set_RLM_parameters) sets the most important parameters for a Return Loss module, preparing the instrument to start a measurement.

If a value is not yet applied to the instrument and the Force Transaction function ([hp816x_forceTransaction](#)) is switched off, the new values will be set.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized. For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).

Data Type: ViSession
Input/Output: IN

<i>RLMSlot</i>	Use the RLM Slot control to choose an installed Return Loss Module.
----------------	---

Data Type: ViInt32
Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8

hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

internalTrigger

Use the Internal Trigger control to choose between Continuous triggering and Immediate triggering.

If you select Immediate triggering (hp816x_RLM_IMMEDIATE) one measurement cycle is performed. When the averaging time is over, the result is available. No further measurement cycles are started. If the result is read with the Fetch RLM Return Loss (hp816x_RLM_fetchReturnLoss) function, the value will always be the same.

Selecting Continuous triggering (hp816x_RLM_CONTINUOUS) will continuous measurements are performed, because after finishing one measurement cycle, a new cycle is immediately triggered.

Data Type: ViBoolean

Input/Output: IN

Values:

Name	Value	Description
hp816x_RLM_IMMEDIATE	0	Immediate
hp816x_RLM_CONTINUOUS	1	Continuous

wavelength

Use Wavelength to set the measurement wavelength in meters (m).

This is the Wavelength value. The responsivity of the Power Sensor varies with wavelength. For accurate power measurement, you need to input the wavelength of the optical input.

Data Type: ViReal64

Input/Output: IN

averagingTime

Use Averaging Time to set the length of time in seconds (s) over which a signal is averaged.

Longer averaging times increase the accuracy and improve the noise rejection of the measurement. Longer averaging times also decrease sensitivity.

Data Type: ViReal64

Input/Output: IN

laserSource

For a Return Loss module, you can use the Laser Source control to select one of the following:

Upper (hp816x_LOWER_SRC), the longer Laser Source wavelength or Lower (hp816x_UPPER_SRC), the shorter Laser Source wavelength.

For a single Laser Source wavelength Return Loss module, the default is Lower.

Data Type: ViBoolean

Input/Output: IN

Values:

Name	Value	Description
hp816x_UPPER_SRC	1	Upper
hp816x_LOWER_SRC	0	Lower

laserState

Use the Laser State control to turn the laser output On (VI_TRUE) or Off (VI_FALSE).

Data Type: ViBoolean

Input/Output: IN

Values:

Name	Value	Description
VI_TRUE	(1)	On
VI_FALSE	(0)	Off

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine

["hp816x_error_message"](#)

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_get_RLM_parameters_Q

Syntax:

```
ViStatus _VI_FUNC hp816x_get_RLM_parameters_Q(ViSession ihandle,
ViInt32 RLMSlot, ViPInt32 internalTrigger, ViPReal64 wavelength, ViPReal64
averagingTime, ViPBoolean laserSource, ViPBoolean laserState);
```

The Get RLM parameters function (hp816x_get_RLM_parameters_Q) returns the most important parameters for a Return Loss module.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized.
<i>RLMSlot</i>	For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).
	Data Type: ViSession Input/Output: IN

RLMSlot

Use the RLM Slot control to choose an installed Return Loss Module.

Data Type: ViInt32
Input/Output: IN

Values:

<u>Name</u>	<u>Value</u>	<u>Description</u>
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6

hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

internalTrigger

Internal Trigger returns one of the following internal trigger configurations:

0 (hp816x_RLM_IMMEDIATE), where only one power measurement is triggered, or

1 (hp816x_RLM_CONTINUOUS), where power measurements are continuously triggered.

Data Type: ViPInt32

Input/Output: OUT

wavelength

Wavelength returns the measurement wavelength in meters (m).

This is the Wavelength value. The responsivity of the Power Sensor varies with wavelength. For accurate power measurement, you need to input the wavelength of the optical input.

Data Type: ViPReal64

Input/Output: OUT

averagingTime

Averaging Time returns the length of time in seconds (s) over which a signal is averaged.

Longer averaging times increase the accuracy and improve the noise rejection of the measurement. Longer averaging times also decrease sensitivity.

Data Type: ViPReal64

Input/Output: OUT

laserSource

Laser Source returns one of the following values:

0 (hp816x_LOWER_SRC) :the shorter wavelength is the selected output.

1 (hp816x_UPPER_SRC) :the longer wavelength is the selected output.

Data Type: ViPBoolean

Input/Output: OUT

laserState

Laser State returns whether the laser is turned on or off.

If 0 (VI_FALSE) is returned, the laser is turned off.

If 1 (VI_TRUE) is returned, the laser is turned on.

Data Type: ViPBoolean

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)"

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_set_RLM_internalTrigger

Syntax:

```
ViStatus _VI_FUNC hp816x_set_RLM_internalTrigger(ViSession ihandle, ViInt32
RLMSlot, ViBoolean internalTrigger);
```

The Set RLM Internal Trigger function (hp816x_set_RLM_internalTrigger) determines how the Return Loss module starts measuring.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized.

For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function ([hp816x_init](#)).

Data Type: ViSession
Input/Output: IN

RLMSlot

Use the RLM Slot control to choose an installed Return Loss Module.

Data Type: ViInt32
Input/Output: IN

Values:

<u>Name</u>	<u>Value</u>	<u>Description</u>
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11

hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

internalTrigger

Use the Internal Trigger control to choose between Continuous triggering (**hp816x_RLM_CONTINUOUS**) and Immediate triggering (**hp816x_RLM_IMMEDIATE**).

Selecting Immediate triggering performs one measurement cycle. When the averaging time is over, the result is available. No further measurement cycles are started. If the result is read with the Fetch RLM Return Loss ([hp816x_RLM_fetchReturnLoss](#)) function , the value will always be the same.

Selecting Continuous triggering will produce different results, because after finishing one measure cycle, a new cycle is immediately triggered.

Data Type: ViBoolean

Input/Output: IN

Values:

Name	Value	Description
hp816x_RLM_CONTINUOUS	1	Continuous
hp816x_RLM_IMMEDIATE	0	Immediate

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "[hp816x_error_message](#)"

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_set_RLM_averagingTime

Syntax:

```
ViStatus _VI_FUNC hp816x_set_RLM_averagingTime(ViSession ihandle,
Vilnt32 RLMSlot, ViReal64 averagingTime);
```

The Set RLM Averaging Time function (hp816x_set_RLM_averagingTime) sets the averaging time of a Return Loss module.

Parameter

ihandle

Description

IHandle is the session handle returned from the Initialize Mainframe function ([hp816x_init](#)). A new session handle is created every time the mainframe is initialized.

For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function ([hp816x_init](#)).

Data Type: ViSession

Input/Output: IN

RLMSlot

Use the RLM Slot control to choose an installed Return Loss Module.

Data Type: Vilnt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7

hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

averagingTime

Use Averaging Time to set the length of time over which a signal is averaged.

Longer averaging times increase the accuracy and improve the noise rejection of the measurement. Longer averaging times also decrease sensitivity.

Data Type: ViReal64

Input/Output: IN

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_get_RLM_averagingTime_Q

Syntax:

```
ViStatus _VI_FUNC hp816x_get_RLM_averagingTime_Q(ViSession ihandle,
Vlnt32 RLMSlot, ViPReal64 averagingTime);
```

The Get RLM Averaging Time (hp816x_get_RLM_averagingTime_Q) function returns the averaging time of a Return Loss module.

Parameter

ihandle

Description

IHandle is the session handle returned from the Initialize Mainframe function ([hp816x_init](#)). A new session handle is created every time the mainframe is initialized.

For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function ([hp816x_init](#)).

Data Type: ViSession

Input/Output: IN

RLMSlot

Use the RLM Slot control to choose an installed Return Loss Module.

Data Type: Vlnt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7

hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

averagingTime

Averaging Time returns the length of time over which a signal is averaged.

Longer averaging times increase the accuracy and improve the noise rejection of the measurement. Longer averaging times also decrease sensitivity.

Data Type: ViPReal64

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)"

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_set_RLM_wavelength

Syntax:

```
ViStatus _VI_FUNC hp816x_set_RLM_wavelength(ViSession ihandle, ViInt32 RLMSlot, ViReal64 wavelength);
```

The Set RLM Wavelength function (hp816x_set_RLM_wavelength) sets the wavelength of a Return Loss module.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized.

For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function ([hp816x_init](#)).

Data Type: ViSession

Input/Output: IN

RLMSlot

Use the RLM Slot control to choose an installed Return Loss Module.

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7

hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

wavelength

Use Wavelength to set the measurement wavelength in meters (m).

The responsivity of the Power Sensor varies with wavelength. For accurate power measurement, you need to input the wavelength of the optical input.

Data Type: ViReal64

Input/Output: IN

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_get_RLM_wavelength_Q

Syntax:

```
ViStatus _VI_FUNC hp816x_get_RLM_wavelength_Q(ViSession ihandle,
ViInt32 RLMSlot, ViPReal64 minWavelength, ViPReal64 maxWavelength,
ViPReal64 defaultWavelength, ViPReal64 currentWavelength);
```

The Get RLM Wavelength function (hp816x_get_RLM_wavelength_Q) returns the minimum, maximum, and current wavelength of a Return Loss module.

The responsivity of the Power Sensor varies with wavelength. For accurate power measurement, you need to input the wavelength of the optical input.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized.

For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function ([hp816x_init](#)).

Data Type: ViSession
Input/Output: IN

RLMSlot

Use the RLM Slot control to choose an installed Return Loss Module.

Data Type: ViInt32
Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3

hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

minWavelength

Min Wavelength returns the lowest measurement wavelength that can be set in meters (m).

Data Type: ViPReal64

Input/Output: OUT

maxWavelength

Max Wavelength returns the highest measurement wavelength that can be set in meters (m).

Data Type: ViPReal64

Input/Output: OUT

defaultWavelength

Default Wavelength returns the default measurement wavelength of your Return Loss module. This is the sum of the Minimum Wavelength and the Maximum Wavelength divided by two. This is not the same value that the Reset function, the Preset function, or the Preset hardkey sets the wavelength to.

Wavelength is returned in meters (m).

Data Type: ViPReal64
Input/Output: OUT

currentWavelength

Current Wavelength returns the measurement wavelength in meters (m) that is currently set.

Data Type: ViPReal64
Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)"

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_set_RLM_powerRange

Syntax:

```
ViStatus _VI_FUNC hp816x_set_RLM_powerRange(ViSession ihandle, ViInt32 RLMSlot,
ViBoolean rangeMode, ViReal64 powerRange, ViReal64 powerRangeSecondSensor);
```

The Set RLM Range function (hp816x_set_RLM_powerRange) sets the power range of a Return Loss module.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized.
	For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).
	Data Type: ViSession Input/Output: IN

RLMSlot

Use the RLM Slot control to choose an installed Return Loss Module.

Data Type: ViInt32
Input/Output: IN

Values:

<u>Name</u>	<u>Value</u>	<u>Description</u>
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15

hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

rangeMode

Use the Range Mode control to choose one of the following range modes:

Auto, the auto-ranging mode, ensures that the result has a displayed value between 9% and 100% of full scale. The default state is for automatic ranging to be enabled.

Manual, which allows you to set a user-defined range.

Data Type: ViBoolean

Input/Output: IN

Values:

Name	Value	Description
hp816x_PWM_RANGE_AUTO	1	Auto
hp816x_PWM_RANGE_MANUAL	0	Manual

powerRange

If you choose Manual using the Range Mode control, you must set a Power Range in dBm.

The Power Range determines the range of power levels that can be measured.

The Upper Power Limit is always 3 dBm greater than the chosen Power Range value. The resolution is always 40 dBm less than the chosen Power Range value.

For example, if you set the Power Range to -30 dBm, you can measure power between -27 dBm and -70 dBm, at a resolution of -70 dBm (0.1 nW).

Data Type: ViReal64

Input/Output: IN

powerRangeSecondSensor

If you choose Manual using the Range Mode control, you must set a Power Range in dBm.

The Power Range determines the range of power levels that can be measured.

The Upper Power Limit is always 3 dBm greater than the chosen Power Range value. The resolution is always 40 dBm less than the

chosen Power Range value.

For example, if you set the Power Range to -30 dBm, you can measure power between -27 dBm and -70 dBm, at a resolution of -70 dBm (0.1 nW).

Data Type: ViReal64

Input/Output: IN

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "[hp816x_error_message](#)"

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_get_RLM_powerRange_Q

Syntax:

```
ViStatus _VI_FUNC hp816x_get_RLM_powerRange_Q(ViSession ihandle,
ViInt32 RLMSlot, ViPBoolean rangeMode, ViPReal64 powerRange, ViPReal64
powerRangeSecondSensor);
```

The Get RLM Range function (hp816x_get_RLM_powerRange_Q) returns the power range setting of a Return Loss module.

Parameter

ihandle

Description

IHandle is the session handle returned from the Initialize Mainframe function ([hp816x_init](#)). A new session handle is created every time the mainframe is initialized.

For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function ([hp816x_init](#)).

Data Type: ViSession

Input/Output: IN

RLMSlot

Use the RLM Slot control to choose an installed Return Loss Module.

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6

hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

rangeMode

Range Mode returns one of the following range modes:

1, the auto-ranging mode (hp816x_RLM_RANGE_AUTO), ensures that the result has a displayed value between 9% and 100% of full scale.

0, manual-ranging mode (hp816x_RLM_RANGE_MANUAL), allows you to set the range.

Data Type: ViPBoolean

Input/Output: OUT

powerRange

Power Range returns the power range in dBm.

The Power Range determines the range of power levels that can be measured.

If the measured power is less than 3 dBm greater than the Power Range, measured power can be returned.

If the measured power is less than 40 dBm less than the Power Range, measured power can be returned.

For example, if you set the Power Range to -30 dBm, you can measure power between -27 dBm and -70 dBm, at a resolution of -70 dBm (0.1 nW).

Data Type: ViPReal64

Input/Output: OUT

powerRangeSecondSensor

Power Range returns the power range in dBm.

The Power Range determines the range of power levels that can be measured.

If the measured power is less than 3 dBm greater than the Power Range, measured power can be returned.

If the measured power is less than 40 dBm less than the Power Range, measured power can be returned.

For example, if you set the Power Range to -30 dBm, you can measure power between -27 dBm and -70 dBm, at a resolution of -70 dBm (0.1 nW).

Data Type: ViPReal64

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_set_RLM_triggerConfiguration

Syntax:

```
ViStatus _VI_FUNC hp816x_set_RLM_triggerConfiguration(ViSession ihandle, ViInt32
RLMSlot, ViInt32 triggerIn, ViInt32 triggerOut);
```

The Set RLM Trigger Configuration (hp816x_set_RLM_triggerConfiguration) function configures input trigger handling and trigger output generation.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized. For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).

Data Type: ViSession
Input/Output: IN

RLMSlot

Use the RLM Slot control to choose an installed Return Loss Module.

Data Type: ViInt32
Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13

hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

triggerIn

Use the Trigger In control to select one of the following input trigger configurations:

Ignore (hp816x_RLM_TRIGIN_IGN), input triggers are ignored,

Single Measurement (hp816x_RLM_TRIGIN_SME), an input trigger triggers a single power measurement, or

Continuous Measurement (hp816x_RLM_TRIGIN_CME), an input trigger triggers continuous power measurement.

Data Type: Vilnt32

Input/Output: IN

Values:

<u>Name</u>	<u>Value</u>	<u>Description</u>
hp816x_RLM_TRIGIN_IGN	0	Ignore
hp816x_RLM_TRIGIN_SME	1	Single Measurement
hp816x_RLM_TRIGIN_CME	2	Continuous Measurement

triggerOut

Use the Trigger Out control to select one of the following output trigger configurations:

Disabled (hp816x_RLM_TRIGOUT_NONE), no output trigger will be generated, or

End of Averaging (hp816x_RLM_TRIGOUT_AVG), at the end of the averaging time for a power measurement an output trigger is generated.

Data Type: Vilnt32

Input/Output: IN

Values:

<u>Name</u>	<u>Value</u>	<u>Description</u>
hp816x_RLM_TRIGOUT_NONE	0	Disabled
hp816x_RLM_TRIGOUT_AVG	1	End of Averaging

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "[hp816x_error_message](#)"

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_start_RLM_internalTrigger

Syntax:

```
ViStatus _VI_FUNC hp816x_start_RLM_internalTrigger(ViSession ihandle,
Vlnt32 RLMSlot);
```

The Start RLM Trigger function (hp816x_start_RLM_internalTrigger) starts a Return Loss measurement cycle.

If you have previously chosen Continuous Triggering mode, see the Set RLM Internal Trigger function, the function chooses Immediate Triggering mode automatically. You need to use the Start RLM Trigger function to start another Return Loss measurement cycle.

Parameter

ihandle

Description

IHandle is the session handle returned from the Initialize Mainframe function ([hp816x_init](#)). A new session handle is created every time the mainframe is initialized.

For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function ([hp816x_init](#)).

Data Type: ViSession

Input/Output: IN

RLMSlot

Use the RLM Slot control to choose an installed Return Loss Module.

Data Type: Vlnt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4

hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)"

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_RLM_readReturnLoss

Syntax:

```
ViStatus _VI_FUNC hp816x_RLM_readReturnLoss(ViSession ihandle, ViInt32 RLMSlot, ViPReal64 returnLoss);
```

The Read RLM Return Loss function (hp816x_RLM_readReturnLoss) forces the instrument to start a measurement cycle. The evaluated return loss value will not be returned, until the averaging time period is finished.

Parameter

ihandle

Description

IHandle is the session handle returned from the Initialize Mainframe function ([hp816x_init](#)). A new session handle is created every time the mainframe is initialized.

For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function ([hp816x_init](#)).

Data Type: ViSession

Input/Output: IN

RLMSlot

Use the RLM Slot control to choose an installed Return Loss Module.

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6

hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

returnLoss

Measured Value returns the measured return loss
in dB

Data Type: ViPReal64

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine
["hp816x_error_message"](#)

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_RLM_fetchReturnLoss

Syntax:

```
ViStatus _VI_FUNC hp816x_RLM_fetchReturnLoss(ViSession ihandle, ViInt32 RLMSlot, ViPReal64 returnLoss);
```

The Fetch RLM Return Loss function (hp816x_RLM_fetchReturnLoss) immediately returns a return loss value without averaging return loss measurements over the averaging time period.

REMARKS: If this function is called more than once within an averaging cycle, the returned values will be identical.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized.

For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function ([hp816x_init](#)).

Data Type: ViSession
Input/Output: IN

RLMSlot

Use the RLM Slot control to choose an installed Return Loss Module.

Data Type: ViInt32
Input/Output: IN

Values:

<u>Name</u>	<u>Value</u>	<u>Description</u>
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4

hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

returnLoss

Measured Value returns the measured return loss
in dB

Data Type: ViPReal64

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine
["hp816x_error_message"](#)

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_RLM_readValue

Syntax:

```
ViStatus _VI_FUNC hp816x_RLM_readValue(ViSession ihandle, ViInt32
RLMSlot, ViBoolean monitorDiode, ViPReal64 powerValue);
```

The Read RLM Power Value function (hp816x_RLM_readValue) forces the instrument to start a measurement cycle. The Power Value will not be returned until the averaging time period is finished. You may choose whether you want to read a power value for the internal monitor diode, or for the return loss diode.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized. For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).

Data Type: ViSession

Input/Output: IN

RLMSlot

Use the RLM Slot control to choose an installed Return Loss Module.

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6

hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

monitorDiode

Use the Monitor Diode control to choose whether you want to read a power value for the internal monitor diode (Yes), or for the return loss diode (No).

Data Type: ViBoolean

Input/Output: IN

Values:

Name	Value	Description
VI_TRUE	(1)	Yes
VI_FALSE	(0)	No

powerValue

Power Value returns the measured power value in Watts (W).

Data Type: ViPReal64

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)"

■

[Alphabetical Reference](#)[Hierarchical Reference](#)

hp816x_RLM_fetchValue

Syntax:

```
ViStatus _VI_FUNC hp816x_RLM_fetchValue(ViSession ihandle, ViInt32  
RLMSlot, ViBoolean monitorDiode, ViPReal64 powerValue);
```

The Fetch RLM Power Value function (hp816x_RLM_fetchValue) immediately returns a power value for either the internal monitor diode, or the return loss diode, without averaging power measurements over the averaging time period.

REMARKS: If this function is called more than once within an averaging cycle, the returned values will be identical.

Parameter

ihandle

Description

IHandle is the session handle returned from the Initialize Mainframe function ([hp816x_init](#)). A new session handle is created every time the mainframe is initialized.

For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function ([hp816x_init](#)).

Data Type: ViSession

Input/Output: IN

RLMSlot

Use the RLM Slot control to choose an installed Return Loss Module.

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4

hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

monitorDiode

Use the Monitor Diode control to choose whether you want to fetch a power value for the internal monitor diode (Yes), or for the return loss diode (No).

Data Type: ViBoolean

Input/Output: IN

Values:

Name	Value	Description
VI_TRUE	(1)	Yes
VI_FALSE	(0)	No

powerValue

Power Value returns the measured power value in Watts (W)

Data Type: ViPReal64

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "[hp816x_error_message](#)"

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_set_RLM_rlReference

Syntax:

```
ViStatus _VI_FUNC hp816x_set_RLM_rlReference(ViSession ihandle, ViInt32 RLMSlot, ViReal64 returnLossReference);
```

The Set RLM Return Loss Reference function (hp816x_set_RLM_rlReference) sets the return loss reference for a Return Loss module.

Parameter

ihandle

Description

IHandle is the session handle returned from the Initialize Mainframe function ([hp816x_init](#)). A new session handle is created every time the mainframe is initialized.

For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function ([hp816x_init](#)).

Data Type: ViSession

Input/Output: IN

RLMSlot

Use the RLM Slot control to choose an installed Return Loss Module.

Data Type: ViInt32

Input/Output: IN

Values:

<u>Name</u>	<u>Value</u>	<u>Description</u>
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7

hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

returnLossReference

Use Return Loss Reference to set the return loss value of your reference reflector in decibels (dB).

For example, the 81610CC reference cable provides a calibrated and stable reference near 14.8 dB.

Data Type: ViReal64

Input/Output: IN

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



[Alphabetical Reference](#)[Hierarchical Reference](#)

hp816x_get_RLM_rlReference_Q

Syntax:

```
ViStatus _VI_FUNC hp816x_get_RLM_rlReference_Q(ViSession ihandle,  
ViInt32 RLMSlot, ViPReal64 returnLossReference);
```

The Get RLM Return Loss Reference function (hp816x_get_RLM_rlReference_Q) returns the return loss reference for a Return Loss module.

Parameter

ihandle

Description

IHandle is the session handle returned from the Initialize Mainframe function ([hp816x_init](#)). A new session handle is created every time the mainframe is initialized.

For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function ([hp816x_init](#)).

Data Type: ViSession

Input/Output: IN

RLMSlot

Use the RLM Slot control to choose an installed Return Loss Module.

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7

hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

returnLossReference

Return Loss Reference returns the return loss value that has been set for your reference reflector in decibels (dB).

For example, the 81610CC reference cable provides a calibrated and stable reference near 14.8 dB return loss.

Data Type: ViPReal64

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)"

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_set_RLM_FPDelta

Syntax:

```
ViStatus _VI_FUNC hp816x_set_RLM_FPDelta(ViSession ihandle, ViInt32 RLMSlot, ViReal64 frontPanelDelta);
```

The Set RLM Front Panel Delta function (hp816x_set_RLM_FPDelta) sets the loss variation value due to the front panel connector.

Follow the instructions in the User's Guide to measure this value.

Parameter

ihandle

Description

IHandle is the session handle returned from the Initialize Mainframe function ([hp816x_init](#)). A new session handle is created every time the mainframe is initialized.

For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function ([hp816x_init](#)).

Data Type: ViSession

Input/Output: IN

RLMSlot

Use the RLM Slot control to choose an installed Return Loss Module.

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5

hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

frontPanelDelta

Use Front Panel Delta to set the correction value for losses that are caused by the front panel connector and other losses.

Data Type: ViReal64

Input/Output: IN

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)"

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_get_RLM_FPDelta_Q

Syntax:

```
ViStatus _VI_FUNC hp816x_get_RLM_FPDelta_Q(ViSession ihandle, ViInt32 RLMSlot, ViPReal64 frontPanelDelta);
```

The Get RLM Front Panel Delta function (hp816x_get_RLM_FPDelta_Q) returns the loss variation value due to the front panel connector.

Parameter

ihandle

Description

IHandle is the session handle returned from the Initialize Mainframe function ([hp816x_init](#)). A new session handle is created every time the mainframe is initialized.

For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function ([hp816x_init](#)).

Data Type: ViSession

Input/Output: IN

RLMSlot

Use the RLM Slot control to choose an installed Return Loss Module.

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7

hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

frontPanelDelta

Front Panel Delta returns the correction value for losses that are caused by the front panel connector and other losses.

Data Type: ViPReal64

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)"

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_calibrate_RLM

Syntax:

```
ViStatus _VI_FUNC hp816x_calibrate_RLM(ViSession ihandle, ViInt32 RLMSlot, ViInt32
calibrate);
```

The Calibrate RLM function (hp816x_calibrate_RLM) allows you to set the Reflection Reference calibration values and the Termination Reference calibration values. You may set calibration values specific to your setup or choose the factory-set calibration values.

Calibration helps to eliminate the affect of wavelength dependencies, coupler directivity, insertion losses, backscattering, and other non-ideal system characteristics.

Performing a Reflection Reference calibration and a Termination Reference calibration will help you perform more accurate return loss measurements than if you use the factory-set calibration values. If you wish to perform a calibration, set up your instrumentation as explained in the mainframe's User's Guide and then call this function.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized. For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).

Data Type: ViSession
Input/Output: IN

<i>RLMSlot</i>	Use the RLM Slot control to choose an installed Return Loss Module.
----------------	---

Data Type: ViInt32
Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10

hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

calibrate

Use Calibrate to select one of the following options:

Reflection Reference (hp816x_CAL_REFL) - sets the Reflection Reference calibration values to the values currently measured by the chosen Return Loss module,

Termination Reference (hp816x_CAL_TERM) - sets the Termination Reference calibration values to the values currently measured by the chosen Return Loss module, or

Factory-Set Calibration (hp816x_CAL_FACTORY) - chooses the factory-set calibration for Reflection Reference calibration values and the Termination Reference calibration values.

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_CAL_REFL	0	Reflection Reference
hp816x_CAL_TERM	1	Termination Reference
hp816x_CAL_FACTORY	2	Factory-Set Calibration

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_RLM_zeroing

Syntax:

```
ViStatus _VI_FUNC hp816x_RLM_zeroing(ViSession ihandle, ViInt32 RLMSlot,
ViPInt32 zeroingResult);
```

The RLM Zeroing function (hp816x_RLM_zeroing) removes electrical offsets for one Return Loss module.

The instrument measures by converting optical power to electrical power, and then measuring electrical power. An electrical offset is power that is always present, even if there is no light at the input. If this offset is not removed, it will affect power measurement.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized.

For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function ([hp816x_init](#)).

Data Type: ViSession
Input/Output: IN

RLMSlot

Use the RLM Slot control to choose an installed Return Loss Module.

Data Type: ViInt32
Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4

hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

zeroingResult

Zeroing Result returns an integer value.

If 0 is returned, the zeroing operation has been successful completed.

If 98 is returned, the zeroing operation has failed because the Power Meter received light. The most common reason for zeroing to fail is if:

- a source is connected to the Power Meter's input connector,
- the fiber connected to the Power Meter's input connector is collecting light, or
- the Power Meter receives ambient light because the input connector is uncovered.

Any other value is an error code and can be looked up in your instrument's Programming Guide.

Data Type: ViPInt32

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine

["hp816x_error_message"](#)

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_RLM_zeroingAll

Syntax:

```
ViStatus _VI_FUNC hp816x_RLM_zeroingAll(ViSession ihandle, ViPInt32
summaryofZeroingAll);
```

The RLM Zeroing All function (hp816x_RLM_zeroingAll) removes electrical offsets for all Return Loss modules and Power Meter modules and Power Meter modules.

The instrument measures by converting optical power to electrical power, and then measuring electrical power. An electrical offset is power that is always present, even if there is no light at the input. If this offset is not removed, it will affect power measurement.

NOTE: Because zeroing is a time consuming operation, the timeout is temporarily increased.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized. For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).

Data Type: ViSession

Input/Output: IN

summaryofZeroingAll

Summary of Zeroing All returns a summary of the results of every Return Loss module and Power Meter in the instrument.

If 0 is returned, the zeroing operations have been successfully completed.

If 98 is returned, a zeroing operation has failed because a Return Loss module or Power Meter has received light. The most common reason for zeroing to fail is if:

- a source is connected to a Return Loss

module's or Power Meter's input connector,

- the fiber connected to a Return Loss module's or Power Meter's input connector is collecting light, or

- a Return Loss module or Power Meter received ambient light because the input connector is uncovered.

Any other value is an error code and can be looked up in your instrument's Programming Guide.

Data Type: ViPInt32

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_enable_RLM_sweep

Syntax:

```
ViStatus _VI_FUNC hp816x_enable_RLM_sweep(ViSession ihandle,
ViBoolean enableRLMLambdaSweep);
```

The Enable RLM Lambda Sweep function (hp816x_enable_RLM_sweep) includes or excludes all Return Loss modules from a lambda scan operation.

A Return Loss module included in a lambda scan appears in the channel list as if it were a dual-channel power meter whose first channel addresses the power measured at its power diode, and whose second channel addresses the power measured at its monitor diode.

For example, if slot 1 is occupied by a single-channel power meter, and slot 2 by a Return Loss module:

- channel 0 addresses the power meter's power diode,
- channel 1 addresses the Return Loss module's power diode,
- channel 3 addresses the Return Loss module's monitor diode.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized.

For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function ([hp816x_init](#)).

Data Type: ViSession

Input/Output: IN

enableRLMLambdaSweep

Use the Enable RLM Lambda Sweep control to include (VI_TRUE) or exclude (VI_FALSE) Return Loss modules from a lambda scan operation.

Data Type: ViBoolean

Input/Output: IN

Values:

Name	Value	Description
VI_TRUE	(1)	Include in sweep
VI_FALSE	(0)	Exclude from sweep

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)"

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_get_RLM_reflectanceValues_Q

Syntax:

```
ViStatus _VI_FUNC hp816x_get_RLM_reflectanceValues_Q(ViSession ihandle,
Vilnt32 RLMSlot, ViPReal64 mref, ViPReal64 pref);
```

The Get RLM Reflectance Values function (get_RLM_reflectanceValues_Q) returns Reflection Reference calibration values. These calibration values are required to calculate the return loss.

If you wish to perform a calibration, set up your instrumentation for a Reflection Reference calibration as explained in the mainframe's User's Guide and then call the Calibrate RLM function ([hp816x_calibrate_RLM](#)).

You can also use the Calibrate RLM function ([hp816x_calibrate_RLM](#)) to choose factory-set calibration values.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized.
<i>RLMSlot</i>	For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).
	Data Type: ViSession Input/Output: IN

RLMSlot

Use the RLM Slot control to choose an installed Return Loss Module.

Data Type: Vilnt32
Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2

hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

mref

Mref returns the Reflection Reference calibration value for the internal monitor diode.

Data Type: ViPReal64

Input/Output: OUT

pref

Pref returns the Reflection Reference calibration value for the internal power diode

Data Type: ViPReal64

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)"

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_get_RLM_terminationValues_Q

Syntax:

```
ViStatus _VI_FUNC hp816x_get_RLM_terminationValues_Q(ViSession
ihandle, ViInt32 RLMSlot, ViPReal64 mpara, ViPReal64 ppara);
```

The Get RLM Termination Values function (get_RLM_terminationValues_Q) returns Termination Reference calibration values. These calibration values are required to calculate the return loss.

If you wish to perform a calibration, set up your instrumentation for a Termination Reference calibration as explained in the mainframe's User's Guide and then call the Calibrate RLM function ([hp816x_calibrate_RLM](#)).

You can also use the Calibrate RLM function ([hp816x_calibrate_RLM](#)) to choose factory-set calibration values.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized. For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).

Data Type: ViSession
Input/Output: IN

<i>RLMSlot</i>	Use the RLM Slot control to choose an installed Return Loss Module.
----------------	---

Data Type: ViInt32
Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_1	1	Slot 1

hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

mpara

Mpara returns the Termination Reference calibration value for the internal monitor diode.

Data Type: ViPReal64

Input/Output: OUT

ppara

Ppara returns the Termination Reference calibration value for the internal power diode

Data Type: ViPReal64

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)



hp816x_get_RLM_dutValues_Q

Syntax:

```
ViStatus _VI_FUNC hp816x_get_RLM_dutValues_Q(ViSession ihandle, ViInt32 RLMSlot, ViPReal64 mdut, ViPReal64 pdut);
```

The Get RLM DUT Values function (hp816x_get_RLM_dutValues_Q) reads the power values at the internal monitor and power diode. Set up your instrumentation to measure the Return Loss of the DUT as explained in the mainframe's User's Guide and then call this function.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized. For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).

Data Type: ViSession
Input/Output: IN

RLMSlot

Use the RLM Slot control to choose an installed Return Loss Module.

Data Type: ViInt32
Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6

hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

mdut

Mdut returns the power value measured by the internal monitor diode.

Data Type: ViPReal64

Input/Output: OUT

pdu

Pdut returns the power value measured by the internal power diode.

Data Type: ViPReal64

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)"

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_calculate_RL

Syntax:

```
ViStatus _VI_FUNC hp816x_calculate_RL(ViSession ihandle, ViInt32 RLMSlot,
ViReal64 mref, ViReal64 mpara, ViReal64 pref, ViReal64 ppara, ViReal64 mdut,
ViReal64 pdut, ViReal64 frontPanelDelta, ViPReal64 returnLoss);
```

The Calculate Return loss function (hp816x_calculate_RL) returns the Return Loss and the Return Loss of the DUT.

This function is intended to be used in more complex test set-ups. Use this function in combination with the following functions:

- the Calibrate RLM function ([hp816x_calibrate_RLM](#)),
- the Get Front Panel Delta function ([hp816x_get_RLM_FPDelta_Q](#)),
- the Get RLM Reflectance Values function ([hp816x_get_RLM_reflectanceValues_Q](#)),
- the Get RLM Termination Values function ([hp816x_get_RLM_terminationValues_Q](#)), and
- the Get RLM DUT Values function ([hp816x_get_RLM_dutValues_Q](#)).

The last four functions return the input values that enable this function to calculate the Return Loss .

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized. For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).
	Data Type: ViSession Input/Output: IN

RLMSlot

Use the RLM Slot control to choose an installed

Return Loss Module.

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

mref

Use Mref to input the Reflection Reference calibration value for the internal monitor diode. This value is returned by the Get RLM Reflectance Values function ([hp816x_get_RLM_reflectanceValues_Q](#)).

Data Type: ViReal64

Input/Output: IN

mpara

Use Mpara to input the Termination Reference calibration value for the internal monitor diode. This value is returned by the Get RLM Reference Values function ([hp816x_get_RLM_terminationValues_Q](#)).

Data Type: ViReal64

Input/Output: IN

pref

Use Pref to input the Reflection Reference calibration value for the internal power diode. This value is returned by the Get RLM Reflectance Values function ([hp816x_get_RLM_reflectanceValues_Q](#)).

Data Type: ViReal64

Input/Output: IN

ppara

Use Ppara to input the Termination Reference calibration value for the internal power diode. This value is returned by the Get RLM Reference Values function ([hp816x_get_RLM_terminationValues_Q](#)).

Data Type: ViReal64

Input/Output: IN

mdut

Use Mdut to input the power value for the internal monitor diode. This value is returned by the Get RLM DUT Values function ([hp816x_get_RLM_dutValues_Q](#)).

Data Type: ViReal64

Input/Output: IN

pdu

Use Pdut to input the power value for the internal power diode. This value is returned by the Get RLM DUT Values function ([hp816x_get_RLM_dutValues_Q](#)).

Data Type: ViReal64

Input/Output: IN

frontPanelDelta

Use Front Panel Delta to input the power value for the internal power diode. This value is returned by the Get Front Panel Delta function (hp816x_get_FPDelta_Q).

Data Type: ViReal64

Input/Output: IN

returnLoss

Return Loss returns the return loss.

This value includes the loss variation due to the front panel connector.

Data Type: ViPReal64

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "[hp816x_error_message](#)"

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_get_RLM_srcWavelength_Q

Syntax:

```
ViStatus _VI_FUNC hp816x_get_RLM_srcWavelength_Q(ViSession ihandle,
ViInt32 RLMSlot, ViPReal64 wavelengthLowerSource, ViPReal64
wavelengthUpperSource);
```

The Get RLM Source Wavelength function (hp816x_get_RLM_srcWavelength) returns the wavelength of a laser source that is built into a Return Loss module.

Parameter

ihandle

Description

IHandle is the session handle returned from the Initialize Mainframe function ([hp816x_init](#)). A new session handle is created every time the mainframe is initialized.

For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function ([hp816x_init](#)).

Data Type: ViSession

Input/Output: IN

RLMSlot

Use the RLM Slot control to choose an installed Return Loss Module.

Data Type: ViInt32

Input/Output: IN

Values:

<u>Name</u>	<u>Value</u>	<u>Description</u>
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6

hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

wavelengthLowerSource

If the Return Loss Module has one built-in laser source, Wavelength Lower Source returns its wavelength.

If there are two built-in laser sources, Wavelength Lower Source returns the wavelength value for the source with the shorter wavelength.

If the Return Loss Module does not have a built-in laser source, or Laser Source is set to External, the value returned is 0.0

Data Type: ViPReal64

Input/Output: OUT

wavelengthUpperSource

If there are two built-in laser sources, Wavelength Upper Source returns the wavelength value for the source with the longer wavelength.

If the Return Loss Module does not have a second built-in laser source, or Laser Source is set to External, the value returned is 0.0

Data Type: ViPReal64

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)"

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_set_RLM_laserSourceParameters

Syntax:

```
ViStatus _VI_FUNC hp816x_set_RLM_laserSourceParameters(ViSession ihandle, Vilnt32 RLMSlot, Vilnt32 laserSource, ViBoolean turnLaser);
```

The Set RLM Laser Source Parameters function (hp816x_set_RLM_laserSourceParameters) sets all parameters for a Return Loss module with an internal Laser Source.

In case of a single frequency Laser Source, the inputs for the lower source will be used.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized.

For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function ([hp816x_init](#)).

Data Type: ViSession
Input/Output: IN

RLMSlot

Use the RLM Slot control to choose an installed Return Loss Module.

Data Type: Vilnt32
Input/Output: IN

Values:

<u>Name</u>	<u>Value</u>	<u>Description</u>
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5

hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

/laserSource

For a Return Loss module, you can use the Laser Source control to select one of the following:

Upper (hp816x_LOWER_SRC), the longer wavelength Laser Source or

Lower (hp816x_UPPER_SRC), the shorter wavelength Laser Source.

For a single-wavelength Return Loss module, this control is ignored.

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_LOWER_SRC	0	Lower
hp816x_UPPER_SRC	1	Upper

turnLaser

Use the Turn Laser On/Off control to turn the Laser Source On (VI_TRUE) or Off (VI_FALSE).

Data Type: ViBoolean

Input/Output: IN

Values:

Name	Value	Description
VI_TRUE	(1)	On
VI_FALSE	(0)	Off

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)"

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_get_RLM_laserSourceParameters_Q

Syntax:

```
ViStatus _VI_FUNC hp816x_get_RLM_laserSourceParameters_Q(ViSession
iHandle, ViInt32 RLMSlot, ViPInt32 laserSource, ViPBoolean laserState);
```

The Set RLM Laser Source Parameters function ([hp816x_set_RLM_laserSourceParameters](#)) returns all parameters for a Return Loss module internal Laser Source.

In case of a single frequency Laser Source, the inputs for the lower source will be used.

Parameter

iHandle

Description

IHandle is the session handle returned from the Initialize Mainframe function ([hp816x_init](#)). A new session handle is created every time the mainframe is initialized.

For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function ([hp816x_init](#)).

Data Type: ViSession

Input/Output: IN

RLMSlot

Use the RLM Slot control to choose an installed Return Loss Module.

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5

hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

laserSource

Laser Source returns one of the following values:

0 (hp816x_LOWER_SRC): the shorter wavelength is the selected output.

1 (hp816x_UPPER_SRC): the longer wavelength is the selected output.

Data Type: ViPInt32

Input/Output: OUT

laserState

Laser State returns whether the laser is turned on or off.

If 0 (VI_FALSE) is returned, the laser is turned off.

If 1 (VI_TRUE) is returned, the laser is turned on.

Data Type: ViPBoolean

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine

["hp816x_error_message"](#)

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_set_RLM_modulationState

Syntax:

```
ViStatus _VI_FUNC hp816x_set_RLM_modulationState(ViSession ihandle,
Vilnt32 RLMSlot, Vilnt32 laserSource, ViBoolean lowFrequencyControl);
```

The Set RLM Modulation State function (hp816x_set_RLM_modulationState) adjusts the internal modulation of a laser source that is built into a Return Loss module.

Internal Modulation is described in your instrument's User's Guide.

Parameter

ihandle

Description

IHandle is the session handle returned from the Initialize Mainframe function ([hp816x_init](#)). A new session handle is created every time the mainframe is initialized.

For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function ([hp816x_init](#)).

Data Type: ViSession

Input/Output: IN

RLMSlot

Use the RLM Slot control to choose an installed Return Loss Module.

Data Type: Vilnt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5

hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

/laserSource

For a Return Loss module that includes two built-in laser sources, you can use the Laser Source control to select one of the following:

Lower (hp816x_RLM_LOWER_SRC), the shorter wavelength built-in Laser Source; or Upper (hp816x_RLM_UPPER_SRC), the longer wavelength built-in Laser Source.

For a Return Loss module with one built-in laser source, this control is ignored.

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_LOWER_SRC	0	Lower
hp816x_UPPER_SRC		

/lowFrequencyControl

Use the Low Frequency Control to turn internal low frequency modulation On (VI_TRUE) or Off (VI_FALSE).

Data Type: ViBoolean

Input/Output: IN

Values:

<u>Name</u>	<u>Value</u>	<u>Description</u>
VI_TRUE	(1)	On
VI_FALSE	(0)	Off

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)"

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_get_RLM_modulationState_Q

Syntax:

```
ViStatus _VI_FUNC hp816x_get_RLM_modulationState_Q(ViSession ihandle,
Vilnt32 RLMSlot, Vilnt32 laserSource, ViPBoolean lowFrequencyControl);
```

The Get RLM Modulation function (hp816x_get_RLM_modulationState_Q) returns the internal modulation state of a laser source that is built into a Return Loss module.

Internal Modulation is described in your instrument's User's Guide.

Parameter

ihandle

Description

IHandle is the session handle returned from the Initialize Mainframe function ([hp816x_init](#)). A new session handle is created every time the mainframe is initialized.

For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function ([hp816x_init](#)).

Data Type: ViSession

Input/Output: IN

RLMSlot

Use the RLM Slot control to choose an installed Return Loss Module.

Data Type: Vilnt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5

hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

/laserSource

For a Return Loss module that includes two built-in laser sources, you can use the Laser Source control to select one of the following:

Lower (hp816x_RLM_LOWER_SRC), the shorter wavelength built-in Laser Source; or Upper (hp816x_RLM_UPPER_SRC), the longer wavelength built-in Laser Source.

For a Return Loss module with one built-in laser source, this control is ignored.

Data Type: VInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_LOWER_SRC	0	Lower
hp816x_UPPER_SRC	1	Upper

/lowFrequencyControl

Low Frequency Control returns the low frequency modulation mode.

If 1 (VI_TRUE) is returned, low frequency modulation is turned on.

If 0 (VI_FALSE) is returned, low frequency modulation is turned off.

Data Type: ViPBoolean

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_set_RLM_laserState

Syntax:

```
ViStatus _VI_FUNC hp816x_set_RLM_laserState(ViSession ihandle, ViInt32 RLMSlot, ViBoolean laserState);
```

The Set RLM Laser State function (hp816x_set_RLM_laserState) turns the laser on or off.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized.
	For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).

Data Type: ViSession

Input/Output: IN

RLMSlot

Use the RLM Slot control to choose an installed Return Loss Module.

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7

hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

laserState

Use the Laser State control to turn the laser output On (VI_TRUE) or Off (VI_FALSE).

Data Type: ViBoolean

Input/Output: IN

Values:

Name	Value	Description
VI_TRUE	(1)	On
VI_FALSE	(0)	Off

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_get_RLM_laserState_Q

Syntax:

```
ViStatus _VI_FUNC hp816x_get_RLM_laserState_Q(ViSession ihandle, ViInt32 RLMSlot, ViPBoolean laserState);
```

The Get RLM Laser State function (hp816x_get_RLM_laserState) turns the laser on or off.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized.
<i>RLMSlot</i>	For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).
	Data Type: ViSession Input/Output: IN

RLMSlot

Use the RLM Slot control to choose an installed Return Loss Module.

Data Type: ViInt32
Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7

hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

laserState

Laser State returns whether the laser is turned on or off.

If 0 (VI_FALSE) is returned, the laser is turned off.

If 1 (VI_TRUE) is returned, the laser is turned on.

Data Type: ViPBoolean

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)"

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_set_RLM_logging

Syntax:

```
ViStatus _VI_FUNC hp816x_set_RLM_logging(ViSession ihandle, ViInt32 RLMSlot, ViReal64 averagingTime, ViInt32 dataPoints, ViPInt32 estimatedTimeout);
```

The Set RLM Logging function (hp816x_set_RLM_logging) sets the parameters for a logging operation and starts the logging.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized.
	For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).
	Data Type: ViSession Input/Output: IN

RLMSlot

Use the RLM Slot control to choose an installed Return Loss Module.

Data Type: ViInt32
Input/Output: IN

Values:

<u>Name</u>	<u>Value</u>	<u>Description</u>
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6

hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

averagingTime

Use Averaging Time to set the length of time in seconds (s) over which a logging operation power measurement is averaged.

Data Type: ViReal64

Input/Output: IN

dataPoints

Use Data Points to set the number of values to be taken during a logging operation.

The maximum number of values is 12000.

Allocate a one-dimensional array with the same dimensions as this value and use as the Logging Result input for the Get RLM Logging Results function ([hp816x_get_RLM_loggingResults_Q](#)).

Data Type: ViInt32

Input/Output: IN

estimatedTimeout

Estimated Timeout returns the estimated time in milliseconds (ms), the logging operation will take; this value can be used to adjust the Timeout function ([hp816x_timeOut](#)).

Data Type: ViPInt32

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "[hp816x_error_message](#)"

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_get_RLM_loggingResults_Q

Syntax:

```
ViStatus _VI_FUNC hp816x_get_RLM_loggingResults_Q(ViSession ihandle,
ViInt32 RLMSlot, ViBoolean waitforCompletion, ViBoolean resultUnit,
ViPBoolean loggingStatus, ViReal64/loggingResult[ ]);
```

The Get RLM Logging Results function (hp816x_get_RLM_loggingResults_Q) returns the result of a previously started logging operation.

NOTE: The Wait for Completion control should only be set to Yes if the application is thoroughly tested because using this control may result in an infinite loop that blocks the calling function.

Also, if the function detects that logging is finished, the logging operation stops automatically.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized. For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).

Data Type: ViSession

Input/Output: IN

RLMSlot

Use the RLM Slot control to choose an installed Return Loss Module.

Data Type: ViInt32

Input/Output: IN

Values:

<u>Name</u>	<u>Value</u>	<u>Description</u>
hp816x_SLOT_1	1	Slot 1

hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

waitForCompletion

Use the Wait for Completion control to choose one of the following:

Yes (VI_TRUE), the stability operation will wait until the Total Time specified in the Set PWM Stability function has elapsed, or

No (VI_FALSE), the stability operation will stop immediately.

Data Type: ViBoolean

Input/Output: IN

Values:

Name	Value	Description
VI_TRUE	(1)	Yes
VI_FALSE	(0)	No

resultUnit

Use the Result Unit control to choose whether to log the return loss value in dB (hp816x_PU_DBM) or in Watts (hp816x_PU_WATT).

Data Type: ViBoolean

Input/Output: IN

Values:

Name	Value	Description
hp816x_PU_DBM	0	dB
hp816x_PU_WATT	1	Watts

loggingStatus

Logging Status returns a Boolean representing the current status of the logging operation.

If Logging Status equals 1, the operation is completed.

If Logging Status equals 0, the operation is being performed.

Data Type: ViPBoolean

Input/Output: OUT

loggingResult

Logging Result is both an input and an output.

As an input, Logging Result accepts a one-dimensional integer array that determines the size of the Logging Result output. Allocate a one-dimensional integer array with the same dimensions as the Data Points value you have set for the Set RLM Logging function ([hp816x_set_RLM_logging](#)).

As an output, Logging Result returns the result of a logging operation as a one-dimensional integer array.

Data Type: ViReal64[]

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine

["hp816x_error_message"](#)

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_set_RLM_stability

Syntax:

```
ViStatus _VI_FUNC hp816x_set_RLM_stability(ViSession ihandle, ViInt32 RLMSlot, ViReal64 averagingTime, ViReal64 periodTime, ViReal64 totalTime, ViPInt32 estimatedResults);
```

The Set RLM Stability function (hp816x_set_RLM_stability) sets up a stability operation and starts the data acquisition.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized.
	For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).

Data Type: ViSession

Input/Output: IN

RLMSlot

Use the RLM Slot control to choose an installed Return Loss Module.

Data Type: ViInt32

Input/Output: IN

Values:

<u>Name</u>	<u>Value</u>	<u>Description</u>
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6

hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

averagingTime

Use Averaging Time to set the length of time in seconds (s) over which a stability operation power measurement is averaged.

Data Type: ViReal64

Input/Output: IN

periodTime

Use the Period time to set the time in seconds between the beginning of each averaging cycle.

Data Type: ViReal64

Input/Output: IN

totalTime

Use Total Time to how long in seconds (s) the stability operation lasts.

Data Type: ViReal64

Input/Output: IN

estimatedResults

Estimated Results returns an estimate of the number of results that the stability operation will record.

Use Estimated Results to allocate a one-dimensional array with the same dimensions as this value and use the array as the Stability Result input for the Get RLM Stability function ([hp816x_get_RLM_stabilityResults_Q](#)).

Data Type: ViPInt32

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "[hp816x_error_message](#)"

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_get_RLM_stabilityResults_Q

Syntax:

```
ViStatus _VI_FUNC hp816x_get_RLM_stabilityResults_Q(ViSession ihandle,
ViInt32 RLMSlot, ViBoolean waitforCompletion, ViBoolean resultUnit,
ViPBoolean stabilityStatus, ViReal64stabilityResult[ ]);
```

The Get RLM Stability Results function (hp816x_get_RLM_stabilityResults_Q) returns the result of a previously started stability operation.

REMARK The Wait for Completion control should only be set to Yes if the application is thoroughly tested because using this control may result in an infinite loop that blocks the calling function.

Also, if the function detects that logging is finished, the stability operation stops automatically.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized. For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).

Data Type: ViSession

Input/Output: IN

RLMSlot

Use the RLM Slot control to choose an installed Return Loss Module.

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_1	1	Slot 1

hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

waitForCompletion

Use the Wait for Completion control to choose one of the following:

Yes (VI_TRUE), the stability operation will wait until the Total Time specified in the Set PWM Stability function has elapsed, or

No (VI_FALSE), the stability operation will stop immediately.

Data Type: ViBoolean

Input/Output: IN

Values:

Name	Value	Description
VI_TRUE	(1)	Yes
VI_FALSE	(0)	No

resultUnit

Use the Result Unit control to choose whether to log the return loss value in dB (hp816x_PU_DBM) or in Watts (hp816x_PU_WATT).

Data Type: ViBoolean

Input/Output: IN

Values:

Name	Value	Description
hp816x_PU_DBM	0	dB
hp816x_PU_WATT	1	Watts

stabilityStatus

Stability Status returns a Boolean representing the current status of the stability operation.

If Stability Status equals 1, the operation is completed.

If Stability Status equals 0, the operation is being performed.

Data Type: ViPBoolean

Input/Output: OUT

stabilityResult

Stability Result is both an input and an output.

As an input, Stability Result accepts a one-dimensional integer array that determines the size of the Stability Result output. Allocate a one-dimensional integer array with the same dimensions as the Estimated Results value you have set for the Set RLM Stability function ([hp816x_set_RLM_stability](#)).

As an output, Stability Result returns the result of a stability operation as a one-dimensional integer array.

Data Type: ViReal64[]

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine

["hp816x_error_message"](#)

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_set_RLM_minMax

Syntax:

```
ViStatus _VI_FUNC hp816x_set_RLM_minMax(ViSession ihandle, ViInt32 RLMSlot, ViInt32 minmaxMode, ViInt32 dataPoints, ViUInt32 estimatedTime);
```

The Set RLM MinMax function (hp816x_set_RLM_minMax) sets the parameters for a MinMax logging operation.

A MinMax logging operation is active as long as it is not explicitly stopped. Before you start a different Return Loss module data acquisition function or different MinMax logging operation, you must call the Stop RLM Function ([hp816x_RLM_functionStop](#)) function.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized. For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).

Data Type: ViSession

Input/Output: IN

RLMSlot

Use the RLM Slot control to choose an installed Return Loss Module.

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5

hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

minmaxMode

Use the MinMax control to choose one of the following MinMax modes:

Continuous mode (hp816x_MM_CONT),

Window mode (hp816x_MM_WIN), or

Refresh mode (hp816x_MM_REFRESH).

See your instrument's User's Guide for information on MinMax modes.

Data Type: Vlnt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_MM_CONT	0	Continuous
hp816x_MM_WIN	1	Window
hp816x_MM_REFRESH	2	Refresh

dataPoints

Use Data Points to define the number of samples used for the MinMax logging operation. Data Points affects Refresh mode and Window mode.

See your instrument's User's Guide for information on Data Points.

Data Type: ViInt32

Input/Output: IN

estimatedTime

Estimated Time returns the time, in seconds, it will take until a minimum or maximum value is available.

This value can be used to adjust the time before you can call the Get RLM MinMax Results function (get_RLM_minMaxResults_Q). This does not apply to Continuous mode.

Data Type: ViUInt32

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_get_RLM_minMaxResults_Q

Syntax:

```
ViStatus _VI_FUNC hp816x_get_RLM_minMaxResults_Q(ViSession ihandle,  
ViInt32 RLMSlot, ViPReal64 minimum, ViPReal64 maximum, ViPReal64 current);
```

The Get RLM MinMax Results function (hp816x_get_RLM_minMaxResults_Q) returns the minimum, maximum and current values of a previously started MinMax operation.

Parameter

iHandle

Description

IHandle is the session handle returned from the Initialize Mainframe function ([hp816x_init](#)). A new session handle is created every time the mainframe is initialized.

For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function ([hp816x_init](#)).

Data Type: ViSession

Input/Output: IN

RLMSlot

Use the RLM Slot control to choose an installed Return Loss Module.

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7

hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

minimum

Minimum returns the minimum power value in Watts for the MinMax operation.

Data Type: ViPReal64
Input/Output: OUT

maximum

Maximum returns the maximum power value in Watts for the MinMax operation.

Data Type: ViPReal64
Input/Output: OUT

current

Current returns the current power value in Watts.

Data Type: ViPReal64
Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)"

E0615, Jun 2015



hp816x_RLM_functionStop

Syntax:

```
ViStatus _VI_FUNC hp816x_RLM_functionStop(ViSession ihandle, ViInt32 RLMSlot);
```

The Stop RLM Function function (hp816x_RLM_functionStop) stops (aborts) any activated logging, stability or Min Max data acquisition operation.

Parameter

ihandle

Description

IHandle is the session handle returned from the Initialize Mainframe function ([hp816x_init](#)). A new session handle is created every time the mainframe is initialized.

For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function ([hp816x_init](#)).

Data Type: ViSession

Input/Output: IN

RLMSlot

Use the RLM Slot control to choose an installed Return Loss Module.

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7

hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)"

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_enable_RLM_sweep

Syntax:

```
ViStatus _VI_FUNC hp816x_enable_RLM_sweep(ViSession ihandle,
ViBoolean enableRLMLambdaSweep);
```

The Enable RLM Lambda Sweep function (hp816x_enable_RLM_sweep) includes or excludes all Return Loss modules from a lambda scan operation.

A Return Loss module included in a lambda scan appears in the channel list as if it were a dual-channel power meter whose first channel addresses the power measured at its power diode, and whose second channel addresses the power measured at its monitor diode.

For example, if slot 1 is occupied by a single-channel power meter, and slot 2 by a Return Loss module:

- channel 0 addresses the power meter's power diode,
- channel 1 addresses the Return Loss module's power diode,
- channel 3 addresses the Return Loss module's monitor diode.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized.

For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function ([hp816x_init](#)).

Data Type: ViSession

Input/Output: IN

enableRLMLambdaSweep

Use the Enable RLM Lambda Sweep control to include (VI_TRUE) or exclude (VI_FALSE) Return Loss modules from a lambda scan operation.

Data Type: ViBoolean

Input/Output: IN

Values:

Name	Value	Description
VI_TRUE	(1)	Include in sweep
VI_FALSE	(0)	Exclude from sweep

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)"

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



Information for N77 Series Instruments

This version of the 816x VXIplug&play instrument driver has been extended to support the N773xA, N774xA optical power meter and the N775xA and N776xA Optical Attenuator instruments. These are stand-alone instruments that do not require installation in an 816x mainframe. The SCPI command set has been designed for almost complete programming compatibility with the modular instruments of the 816x family. Please be aware of the following details and limitations, when using this driver.

- The N77-series instruments can be controlled via USB, LAN or GPIB interfaces. Especially for transferring large data volumes, the USB and usually LAN interfaces provide faster performance.
- The identification of the individual ports of these multiport instruments in the SCPI and hp816x functions corresponds to the slot number. So for example the ports of the N7745A can be addressed with the slot numbers 1 to 8. The channel number in the commands and functions is not used and can generally be omitted or set to Channel 1 (index 0).
- The N771x-series and 81950A tunable lasers are not supported by the 816x VXIplug&play driver.
- The N773x-series optical switches are supported by the 816x VXIplug&play functions for switches and relevant functions for the mainframes.
- The N774x-series power meters are supported by the 816x VXIplug&play functions for power meters, relevant functions for the mainframes like for standard trigger configuration, and the Multi Frame Lambda Scan application functions.
- The N776x-series optical attenuators are supported by the 816x VXIplug&play functions for attenuators and the relevant functions for the mainframes.
- The N775x-series optical attenuators and power meters are supported by the 816x VXIplug&play functions for attenuators and the relevant functions for the mainframes and the basic functions for the power meters, excluding the advanced functions using data power logging. The choices of averaging time is also more limited for these power meters.
- The averaging time parameter for the power meters is limited in the 816x VXIplug&play functions to a limited set of discrete values, similar to the modular 816x power meters. This set includes: 1, 2, 5, 10, 25, 100, 200, and 500 μ s, 1, 2, 5, 10, 20, 50, 100, 200, 500 ms, and 1, 2, 5, and 10s.
- The maximum number of logging data points that can be acquired is limited by the 816x VXIplug&play data acquisition functions to 100,000 per port.
- The shortest averaging time used by the Multi Frame Lambda Scan application is 25 μ s.
- For the Multi Frame Lambda Scan application, the N774xA instruments can be included simply by registering the instrument as a mainframe, after the first

mainframe with the tunable laser has been registered.

Further details to available functions are found under the categories below:

[Utility Functions](#)

[Mainframe Specific Functions](#)

[Power Sensor Specific Functions](#)

[Applications](#)

hp816x_WaitForOPC

Syntax:

```
ViStatus _VI_FUNC hp816x_WaitForOPC(ViSession IHandle, ViInt32 TLS_Slot,
ViBoolean Wait_for_Operation_Complete );
```

The Wait for Operation Complete function (hp816x_WaitForOPC) can be useful if DFB Laser Sources are installed.

Since changing the wavelength of a DFB source is a time consuming operation, this function may be used to return immediately from a wavelength setting operation.

<u>Parameter</u>	<u>Description</u>
<i>IHandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized.
	For LabView Programmers
	This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).
	Data Type: ViSession Input/Output: IN
<i>TLS_Slot</i>	Use the TLS Slot control to choose an installed Tunable Laser module.
	Data Type: ViInt32 Input/Output: IN
Wait_for_Operation_Complete	Use the Wait_for_Operation_Complete control, a boolean input, to set whether to "Wait for OPC" or not.

If "Wait for OPC" is set to "No", the VXI-PnP driver will return immediately without checking the OPC flag. The desired parameter adjustment may be incomplete.

Data Type: ViBoolean
Input/Output: IN

Values:

Name	Value	Description
VI_TRUE	(1)	Wait for OPC
VI_FALSE	(0)	Do not wait for OPC

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "[hp816x_error_message](#)"

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_set_TLS_parameters

Syntax:

```
ViStatus _VI_FUNC hp816x_set_TLS_parameters(ViSession ihandle, ViInt32
TLSSlot, ViInt32 powerUnit, ViInt32 opticalOutput, ViBoolean turnLaser, ViReal64
power, ViReal64 attenuation, ViReal64 wavelength);
```

The Set TLS Parameters (hp816x_set_TLS_parameters) function sets the most important parameters of a Tunable Laser module.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized.
<i>TLSSlot</i>	For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).
	Data Type: ViSession Input/Output: IN

TLSSlot

Use the TLS Slot control to choose an installed Tunable Laser module.

Data Type: ViInt32
Input/Output: IN

Values:

<u>Name</u>	<u>Value</u>	<u>Description</u>
hp816x_SLOT_0	0	Slot 0
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5

hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

powerUnit

You can use Power Unit to choose either Watts (hp816x_PU_WATT) or dBm (hp816x_PU_DBM) as the power unit.

Data Type: VlInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_PU_WATT	1	Watt
hp816x_PU_DBM	0	dBm

opticalOutput

Use the Optical Output control to choose the optical output for a TLS module with two outputs. You can choose one of the following:

High Power (hp816x_HIGHPOW), where Output 2, the High Power output is the regulated output.

Low SSE (hp816x_LOWSSE), where Output 1, the Low SSE output is the regulated output.

BHR (hp816x_BHR), is the same as choosing Both (Master 2) from the instrument's front panel, Output 2, the High Power output is the regulated output and Output 1, the Low SSE output is the unregulated output.

BLR (hp816x_BLR), is the same as choosing

Both (Master 1) from the instrument's front panel,
the Low SSE output is the regulated output and
Output 2, the High Power output is the
unregulated output.

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_HIGHPOW	0	High Power
hp816x_LOWSSE	1	Low SSE
hp816x_BHR	2	BHR
hp816x_BLR	3	BLR

turnLaser

You can use the Turn Laser On/Off control to turn
the Laser Source On (VI_TRUE) or Off
(VI_FALSE).

Data Type: ViBoolean

Input/Output: IN

Values:

Name	Value	Description
VI_TRUE	(1)	On
VI_FALSE	(0)	Off

power

Use Power to set the output power.

You must choose the power unit using the Power
Unit control.

Data Type: ViReal64

Input/Output: IN

attenuation

If your Tunable Laser module has an in-built
optical attenuator, use Attenuation to set the
attenuation to a value in decibels (dB).

Data Type: ViReal64

Input/Output: IN

wavelength

Use Wavelength to set the output wavelength in meters (m) of the Tunable Laser module.

Data Type: ViReal64

Input/Output: IN

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)"

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_get_TLS_parameters_Q

Syntax:

```
ViStatus _VI_FUNC hp816x_get_TLS_parameters_Q(ViSession ihandle,
ViInt32 TLSSlot, ViPInt32 powerUnit, ViPBoolean laserState, ViPInt32
opticalOutput, ViPReal64 power, ViPReal64 attenuation, ViPReal64 wavelength);
```

The Get TLS Parameters (hp816x_get_TLS_parameters_Q) function returns the most important parameters of a Tunable Laser module.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized.
<i>TLSSlot</i>	For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).
	Data Type: ViSession Input/Output: IN

TLSSlot

Use the TLS Slot control to choose an installed Tunable Laser module.

Data Type: ViInt32
Input/Output: IN

Values:

<u>Name</u>	<u>Value</u>	<u>Description</u>
hp816x_SLOT_0	0	Slot 0
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5

hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

powerUnit

Power Unit returns the following:

0 (hp816x_PU_DBM) units of dBm or

1 (hp816x_PU_WATT) units of Watts.

Data Type: ViPInt32

Input/Output: OUT

laserState

Laser State returns the following:

0 (VI_FALSE), the laser is turned off.

1 (VI_TRUE), the laser is turned on.

Data Type: ViPBoolean

Input/Output: OUT

opticalOutput

Optical Output returns the regulated output of the Tunable Laser module, whereby the one the following values is returned:

0 (hp816x_HIGHPOW); when Output 2, the High

Power output is the regulated output.

1 (hp816x_LOWSSE); when Output 1, the Low SSE output is the regulated output.

2 (hp816x_BHR); when Output 2, the High Power output is the regulated output and Output 1, the Low SSE output is the unregulated output.

3 (hp816x_BLR); when Output 1, the Low SSE output is the regulated output and Output 2, the High Power output is the unregulated output.

Data Type: ViPInt32

Input/Output: OUT

power

Power returns the output power of the TLS in the chosen power unit.

Use the Unit control of the Set TLS Power function ([hp816x_set_TLS_power](#)) to set the units to dBm or Watts.

Use the value returned by Power Unit to find out whether Power is returned in Watts or dBm.

Data Type: ViPReal64

Input/Output: OUT

attenuation

Attenuation returns the attenuation value in decibels (dB).

Check if your Tunable Laser module has an in-built optical attenuator.

Data Type: ViPReal64

Input/Output: OUT

wavelength

Wavelength returns the wavelength value of the Tunable Laser module in meters (m).

Data Type: ViPReal64

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)"

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_set_TLS_wavelength

Syntax:

```
ViStatus _VI_FUNC hp816x_set_TLS_wavelength(ViSession ihandle, ViInt32
TLSSlot, ViInt32 wavelengthSelection, ViReal64 wavelength);
```

The Set TLS Wavelength (hp816x_set_TLS_wavelength) function sets the wavelength of a Tunable Laser module.

Parameter

ihandle

Description

IHandle is the session handle returned from the Initialize Mainframe function ([hp816x_init](#)). A new session handle is created every time the mainframe is initialized.

For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function ([hp816x_init](#)).

Data Type: ViSession

Input/Output: IN

TLSSlot

Use the TLS Slot control to choose an installed Tunable Laser module.

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_0	0	Slot 0
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6

hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

wavelengthSelection

You can use the Wavelength Selection control to select one of the following:

Minimum (hp816x_INP_MIN), which is the minimum wavelength,

Maximum (hp816x_INP_MAX), which is the maximum wavelength,

Default (hp816x_INP_DEF), which is the sum of the minimum wavelength and the maximum wavelength divided by two (this is not the same value that the Reset function, the Preset function, or the Preset hardkey sets the frequency to), and

Manual Input (hp816x_INP_MAN), which is the value entered in the Wavelength input.

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_INP_MIN	0	Minimum
hp816x_INP_DEF	1	Default
hp816x_INP_MAX	2	Maximum
hp816x_INP_MAN	3	Manual Input

wavelength

If you choose Manual Input as the Wavelength

Selection input, you can use the Wavelength input to set the wavelength to a value in meters (m).

Data Type: ViReal64

Input/Output: IN

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)"

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_get_TLS_wavelength_Q

Syntax:

```
ViStatus _VI_FUNC hp816x_get_TLS_wavelength_Q(ViSession ihandle,
ViInt32 TLSSlot, ViPReal64 minimumWavelength, ViPReal64 defaultWavelength,
ViPReal64 maximumWavelength, ViPReal64 currentWavelength);
```

The Get TLS Wavelength (hp816x_get_TLS_wavelength_Q) function returns the current values of the following: Minimum Wavelength, Maximum Wavelength, Default Wavelength, and Current Wavelength.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized.
<i>TLSSlot</i>	For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).
	Data Type: ViSession Input/Output: IN

TLSSlot

Use the TLS Slot control to choose an installed Tunable Laser module.

Data Type: ViInt32
Input/Output: IN

Values:

<u>Name</u>	<u>Value</u>	<u>Description</u>
hp816x_SLOT_0	0	Slot 0
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5

hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

minimumWavelength

Minimum Wavelength returns the lowest wavelength that your Tunable Laser module can be set to.

Wavelength is returned in meters (m).

Data Type: ViPReal64

Input/Output: OUT

defaultWavelength

Default Wavelength returns the default wavelength of your Tunable Laser module. This is identical to the default value set by setting the Wavelength Selection control of the Set TLS Wavelength function to Default; it is the sum of the Minimum Wavelength and the Maximum Wavelength divided by two. This is not the same value that the Reset function, the Preset function, or the Preset hardkey sets the wavelength to.

Wavelength is returned in meters (m).

Data Type: ViPReal64

Input/Output: OUT

maximumWavelength

Maximum Wavelength returns the highest wavelength that your Tunable Laser module can

be set to.

Wavelength is returned in meters (m).

Data Type: ViPReal64

Input/Output: OUT

currentWavelength

Current Wavelength returns the wavelength that your Tunable Laser module is set to.

Wavelength is returned in meters (m).

Data Type: ViPReal64

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)"

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_set_TLS_power

Syntax:

```
ViStatus _VI_FUNC hp816x_set_TLS_power(ViSession ihandle, ViInt32
TLSSlot, ViInt32 unit, ViInt32 powerSelection, ViReal64 manualPower);
```

The Set TLS Power (hp816x_set_TLS_power) function sets the power of a Tunable Laser module.

REMARK If you choose Manual Input as the Power Selection input, you can use the Manual Power input to set the power to a value in dBm or Watts, depending on how you set the Unit control.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized.
<i>TLSSlot</i>	For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).
	Data Type: ViSession Input/Output: IN

TLSSlot

Use the TLS Slot control to choose an installed Tunable Laser module.

Data Type: ViInt32
Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_0	0	Slot 0
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4

hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

unit

You can use the Unit control to choose either Watts (hp816x_PU_WATT) or dBm (hp816x_PU_DBM) as the power unit.

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_PU_DBM	0	dBm
hp816x_PU_WATT	1	Watt

powerSelection

You can use the Power Selection control to select one of the following:

Minimum (hp816x_INP_MIN), which is the minimum power,

Maximum (hp816x_INP_MAX), which is the maximum power,

Default (hp816x_INP_DEF), which is the sum of the minimum power and the maximum power divided by two (this is not the same value that the Reset function, the Preset function, or the Preset hardkey sets the frequency to), and

Manual Power (hp816x_INP_MAN), which is the

value entered in the Wavelength input.

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_INP_MIN	0	Minimum
hp816x_INP_DEF	1	Default
hp816x_INP_MAX	2	Maximum
hp816x_INP_MAN	3	Manual Input

manualPower

If you choose Manual Input as the Power Selection input, you can use the Manual Power input to set the power to a value in dBm or Watts, depending on how you set the Unit control.

Data Type: ViReal64

Input/Output: IN

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)"

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_get_TLS_power_Q

Syntax:

```
ViStatus _VI_FUNC hp816x_get_TLS_power_Q(ViSession ihandle, ViInt32 TLSSlot, ViPInt32 powerUnit, ViPReal64 minimumPower, ViPReal64 defaultPower, ViPReal64 maximumPower, ViPReal64 currentPower);
```

The Get TLS Power (hp816x_get_TLS_power_Q) function returns the current values of the following: the Power Units, Minimum Power, Maximum Power, Default Power, and Current Power. The power limits depend on the current optical output.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized.
<i>TLSSlot</i>	For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).
	Data Type: ViSession Input/Output: IN

TLSSlot

Use the TLS Slot control to choose an installed Tunable Laser module.

Data Type: ViInt32
Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_0	0	Slot 0
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4

hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

powerUnit

Power Units returns the currently selected power units for your Tunable Laser module.

0 (hp816x_PU_DBM) is returned if the units are dBm.

1 (hp816x_PU_WATT) is returned if the units are Watts (W).

Data Type: ViPInt32

Input/Output: OUT

minimumPower

Minimum Power returns the lowest power that your Tunable Laser module can be set to.

Power is returned in units of dBm or Watts.

Use the Unit control of the Set TLS Power function ([hp816x_set_TLS_power](#)) to set the units to dBm or Watts.

Use the value returned by Power Unit to find out whether Power is returned in Watts or dBm.

Data Type: ViPReal64

Input/Output: OUT

defaultPower

Default Power returns the default power of your Tunable Laser module. This is identical to the default value set by setting the Power Selection control of the Set TLS Power function to Default; it is the sum of the Minimum Power and the Maximum Power divided by two. This is not the same value that the Reset function, the Preset function, or the Preset hardkey sets the wavelength to.

Power is returned in units of dBm or Watts. Use the Unit control of the Set TLS power function ([hp816x_set_TLS_power](#)) to set the units to dBm or Watts.

Data Type: ViPReal64

Input/Output: OUT

maximumPower

Maximum Power returns the highest power that your Tunable Laser module can be set to.

Power is returned in units of dBm or Watts. Use the Unit control of the Set TLS power function ([hp816x_set_TLS_power](#)) to set the units to dBm or Watts.

Data Type: ViPReal64

Input/Output: OUT

currentPower

Current Power returns the power that your Tunable Laser module is set to.

Power is returned in units of dBm or Watts. Use the Unit control of the Set TLS power function ([hp816x_set_TLS_power](#)) to set the units to dBm or Watts.

Data Type: ViPReal64

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)"

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_set_TLS_opticalOutput

Syntax:

```
ViStatus _VI_FUNC hp816x_set_TLS_opticalOutput(ViSession ihandle, ViInt32 TLSSlot, ViInt32 setOpticalOutput);
```

The Set TLS Optical Output (hp816x_set_TLS_opticalOutput) function configures the optical output of Tunable Laser module with two optical outputs.

Parameter

ihandle

Description

IHandle is the session handle returned from the Initialize Mainframe function ([hp816x_init](#)). A new session handle is created every time the mainframe is initialized.

For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function ([hp816x_init](#)).

Data Type: ViSession

Input/Output: IN

TLSSlot

Use the TLS Slot control to choose an installed Tunable Laser module.

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_0	0	Slot 0
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6

hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

setOpticalOutput

Use the Optical Output control to choose the optical output for a TLS module with two outputs. You can choose one of the following:

High Power (hp816x_HIGHPOW), where Output 2, the High Power output is the regulated output.

Low SSE (hp816x_LOWSSE), where Output 1, the Low SSE output is the regulated output.

BHR (hp816x_BHR) is the same as choosing Both (Master 2) from the instrument's front panel, Output 2, the High Power output is the regulated output and Output 1, the Low SSE output is the unregulated output.

BLR (hp816x_BLR) is the same as choosing Both (Master 1) from the instrument's front panel, the Low SSE output is the regulated output and Output 2, the High Power output is the unregulated output.

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_HIGHPOW	0	High Power
hp816x_LOWSSE	1	Low SSE
hp816x_BHR	2	BHR
hp816x_BLR	3	BLR

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_get_TLS_opticalOutput_Q

Syntax:

```
ViStatus _VI_FUNC hp816x_get_TLS_opticalOutput_Q(ViSession ihandle,
Vilnt32 TLSSlot, ViPInt32 opticalOutput);
```

The Get TLS Optical Output (hp816x_get_TLS_opticalOutput_Q) function returns the optical output configuration of a Tunable Laser module with two optical outputs.

REMARK

If this function addresses a module with only one optical output, an error is generated.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized.

For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function ([hp816x_init](#)).

Data Type: ViSession
Input/Output: IN

TLSSlot

Use the TLS Slot control to choose an installed Tunable Laser module.

Data Type: Vilnt32
Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_0	0	Slot 0
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2

hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

opticalOutput

Optical Output returns the regulated output of the Tunable Laser module, whereby the one the following values is returned:

0 (hp816x_HIGHPOW); when Output 2, the High Power output is the regulated output.

1 (hp816x_LOWSSE); when Output 1, the Low SSE output is the regulated output.

2 (hp816x_BHR); when Output 2, the High Power output is the regulated output and Output 1, the Low SSE output is the unregulated output.

3 (hp816x_BLR); when Output 1, the Low SSE output is the regulated output and Output 2, the High Power output is the unregulated output.

Data Type: ViPInt32

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)"

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_get_TLS_powerMaxInRange_Q

Syntax:

```
ViStatus _VI_FUNC hp816x_get_TLS_powerMaxInRange_Q(ViSession ihandle, Vilnt32 TLSSlot, ViReal64 startofRange, ViReal64 endofRange, ViPReal64 maximumPower);
```

The Get TLS Max Power in Range (hp816x_get_TLS_powerMaxInRange_Q) function returns the maximum permitted power for a given wavelength interval.

This function can be used to calculate the maximum permitted power for a wavelength sweep.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized.

For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function ([hp816x_init](#)).

Data Type: ViSession
Input/Output: IN

TLSSlot

Use the TLS Slot control to choose an installed Tunable Laser module.

Data Type: Vilnt32
Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_0	0	Slot 0
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3

hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

startofRange

The Start of Range input is the lowest value of the wavelength interval in meters (m).

Data Type: ViReal64

Input/Output: IN

endofRange

The End of Range input is the highest value of the wavelength interval in meters (m).

Data Type: ViReal64

Input/Output: IN

maximumPower

Maximum Power returns the maximum permitted power for your Tunable Laser module in the range between the Start of Range and End of Range inputs you have entered.

Maximum Power is returned in dBm or Watts depending on the units you choose using the Set TLS Power function.

Data Type: ViPReal64

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "[hp816x_error_message](#)"

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_set_TLS_laserState

Syntax:

```
ViStatus _VI_FUNC hp816x_set_TLS_laserState(ViSession ihandle, ViInt32 TLSSlot, ViBoolean laserState);
```

The Set TLS Laser State (hp816x_set_TLS_laserState) function turns the laser on or off.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized.
	For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).

Data Type: ViSession

Input/Output: IN

TLSSlot

Use the TLS Slot control to choose an installed Tunable Laser module.

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_0	0	Slot 0
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6

hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

laserState

Use the Laser State control to turn the laser output On (VI_TRUE) or Off (VI_FALSE).

Data Type: ViBoolean

Input/Output: IN

Values:

Name	Value	Description
VI_TRUE	(1)	On
VI_FALSE	(0)	Off

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_get_TLS_laserState_Q

Syntax:

```
ViStatus _VI_FUNC hp816x_get_TLS_laserState_Q(ViSession ihandle, ViInt32 TLSSlot, ViPBoolean laserState);
```

The Get TLS laser State (hp816x_get_TLS_laserState_Q) function returns whether the laser is turned on or off.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized.
<i>TLSSlot</i>	For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).

Data Type: ViSession

Input/Output: IN

TLSSlot

Use the TLS Slot control to choose an installed Tunable Laser module.

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_0	0	Slot 0
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6

hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

laserState

Laser State returns whether the laser is turned on or off.

If 0 (VI_FALSE) is returned, the laser is turned off.

If 1 (VI_TRUE) is returned, the laser is turned on.

Data Type: ViPBoolean

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)"

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_set_TLS_laserRiseTime

Syntax:

```
ViStatus _VI_FUNC hp816x_set_TLS_laserRiseTime(ViSession ihandle,
Vilnt32 TLSSlot, ViReal64 laserRiseTime);
```

The Set TLS Laser Rise Time (hp816x_set_TLS_laserRiseTime) function adjusts the laser rise time.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized.
<i>TLSSlot</i>	For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).

Data Type: ViSession

Input/Output: IN

TLSSlot

Use the TLS Slot control to choose an installed Tunable Laser module.

Data Type: Vilnt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_0	0	Slot 0
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6

hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

laserRiseTime

Use Laser Rise Time to input the laser rise time in meters (0-3).

Data Type: ViReal64

Input/Output: IN

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)"

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_get_TLS_laserRiseTime

Syntax:

```
ViStatus _VI_FUNC hp816x_get_TLS_laserRiseTime(ViSession ihandle,
Vilnt32 TLSSlot, ViPReal64 laserRiseTime);
```

The Get TLS Laser Rise Time (hp816x_get_TLS_laserRiseTime) function returns the laser rise time.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized.
<i>TLSSlot</i>	For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).
	Data Type: ViSession Input/Output: IN
	Data Type: Vilnt32 Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_0	0	Slot 0
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6

hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

laserRiseTime

Laser Rise Time returns the laser rise time in meters.

Data Type: ViPReal64

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)"

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_set_TLS_triggerConfiguration

Syntax:

```
ViStatus _VI_FUNC hp816x_set_TLS_triggerConfiguration(ViSession ihandle, ViInt32 TLSSlot, ViInt32 triggerIn, ViInt32 triggerOut);
```

The Set TLS Trigger Configuration (hp816x_set_TLS_triggerConfiguration) function configures:

- how the Tunable Laser module processes input triggers, and
- how an output trigger is generated.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	<p>IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized.</p> <p>For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).</p> <p>Data Type: ViSession Input/Output: IN</p>

TLSSlot

Use the TLS Slot control to choose an installed Tunable Laser module.

Data Type: ViInt32
Input/Output: IN

Values:

<u>Name</u>	<u>Value</u>	<u>Description</u>
hp816x_SLOT_0	0	Slot 0
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9

hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

triggerIn

You can use the Trigger In control to choose one of the following input trigger configurations:

Ignore (hp816x_TLS_TRIGIN_IGN), where input triggers are ignored,

Next Step (hp816x_TLS_TRIGIN_NEXTSTEP), where an input trigger will cause the next step of a sweep to be performed, and

Run Sweep (hp816x_TLS_TRIGIN_SWEEPSTART), where an input trigger will cause a sweep to run.

Trigger In only applies to Tunable Laser modules.

Data Type: ViInt32
Input/Output: IN

Values:

<u>Name</u>	<u>Value</u>	<u>Description</u>
hp816x_TLS_TRIGIN_IGN	0	Ignore
hp816x_TLS_TRIGIN_NEXTSTEP	1	Next Step
hp816x_TLS_TRIGIN_SWEEPSTART	2	Run Sweep

triggerOut

You can use the Trigger Out to choose one of the following output trigger configurations:

Disabled (hp816x_TLS_TRIGOUT_DISABLED), where output trigger mode is disabled,

Modulation (hp816x_TLS_TRIGOUT_MOD), where the output trigger connector outputs a TTL signal at the frequency of the modulation (this signal is output whether the laser is switched on or off),

Step Finished (hp816x_TLS_TRIGOUT_STEPEnd), where a trigger is output after every step of a sweep finishes,

Sweep Finished

(hp816x_TLS_TRIGOUT_SWEND), where a trigger is output after a sweep finishes, and

Sweep Started
(hp816x_TLS_TRIGOUT_SWSTART), where a trigger is output after a sweep starts.

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_TLS_TRIGOUT_DISABLED	0	Disabled
hp816x_TLS_TRIGOUT_MOD	1	Modulation
hp816x_TLS_TRIGOUT_STEPEND	2	Step Finished
hp816x_TLS_TRIGOUT_SWSTART	3	Sweep Started
hp816x_TLS_TRIGOUT_SWEND	4	Sweep Finished

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)"

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_get_TLS_triggerConfiguration

Syntax:

```
ViStatus _VI_FUNC hp816x_get_TLS_triggerConfiguration(ViSession ihandle,
ViInt32 TLSSlot, ViPInt32 triggerIn, ViPInt32 triggerOut);
```

The Get TLS Trigger Configuration (hp816x_get_TLS_triggerConfiguration) function returns:

- the selected Tunable Laser module's input triggers configuration and
- the selected Tunable Laser module's output triggers configuration.

Parameter

ihandle

Description

IHandle is the session handle returned from the Initialize Mainframe function ([hp816x_init](#)). A new session handle is created every time the mainframe is initialized.

For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function ([hp816x_init](#)).

Data Type: ViSession

Input/Output: IN

TLSSlot

Use the TLS Slot control to choose an installed Tunable Laser module.

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_0	0	Slot 0
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3

hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

triggerIn

You can use the Trigger In to choose one of the following input trigger configurations:

0 (hp816x_TLS_TRIGIN_IGN), where input triggers are ignored,

1 (hp816x_TLS_TRIGIN_NEXTSTEP), where an input trigger will cause the next step of a sweep to be performed, and

2 (hp816x_TLS_TRIGIN_SWEEPSTART), where an input trigger will cause a sweep to be run.

Data Type: ViPInt32

Input/Output: OUT

triggerOut

You can use the Trigger Out to choose one of the following output trigger configurations:

0 (hp816x_TLS_TRIGOUT_DISABLED), where output trigger mode is disabled,

1 (hp816x_TLS_TRIGOUT_MOD), where the output trigger connector outputs a TTL signal at the frequency of the internal modulation (this signal is output whether the laser is switched on or off),

2 (hp816x_TLS_TRIGOUT_STEPEND), where a trigger is output after every step of a sweep finishes,

3 (hp816x_TLS_TRIGOUT_SWSTART), where a trigger is output after a sweep starts, and

4 (hp816x_TLS_TRIGOUT_SWEND), where a trigger is output after a sweep finishes.

Data Type: ViPInt32

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)"

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_set_TLS_attenuation

Syntax:

```
ViStatus _VI_FUNC hp816x_set_TLS_attenuation(ViSession ihandle, ViInt32 TLSSlot, ViBoolean powerMode, ViBoolean darkenLaser, ViReal64 attenuation);
```

The Set TLS Attenuation (hp816x_set_TLS_attenuation) function adjusts the attenuation of a Tunable Laser module.

NOTE If this function addresses a module without a built-in attenuator, an error is generated.

Parameter

ihandle

Description

IHandle is the session handle returned from the Initialize Mainframe function ([hp816x_init](#)). A new session handle is created every time the mainframe is initialized.

For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function ([hp816x_init](#)).

Data Type: ViSession

Input/Output: IN

TLSSlot

Use the TLS Slot control to choose an installed Tunable Laser module.

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_0	0	Slot 0
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4

hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

powerMode

For the Power Mode control, you can choose the following:

Auto (VI_TRUE), where the module automatically regulates the optimum combination between built-in attenuator and the laser current to achieve lowest noise, or

Manual (VI_FALSE), where you can enter the attenuation.

Data Type: ViBoolean

Input/Output: IN

Values:

Name	Value	Description
VI_TRUE	(1)	Auto
VI_FALSE	(0)	Manual

darkenLaser

Use the Darken Laser control to darken the laser output.

You use this control when you want to turn the laser On (VI_TRUE) and Off (VI_FALSE) without any settling delays when you turn the laser on.

Your Tunable Laser module must have built-in attenuation.

Data Type: ViBoolean
Input/Output: IN

Values:

Name	Value	Description
VI_TRUE	(1)	On
VI_FALSE	(0)	Off

attenuation

If you choose Manual as the Power Mode, you can use the Attenuation input to set the attenuation level in decibels (dB).

Data Type: ViReal64
Input/Output: IN

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "[hp816x_error_message](#)"

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_get_TLS_attenuationSettings_Q

Syntax:

```
ViStatus _VI_FUNC hp816x_get_TLS_attenuationSettings_Q(ViSession ihandle, ViInt32 TLSSlot,
ViPBoolean powerMode, ViPBoolean dark, ViPReal64 attenuation);
```

The Get TLS Attenuation Settings (hp816x_get_TLS_attenuation_Q) function returns the attenuation level, power mode and dark position of a Tunable Laser module.

NOTE:

If this function addresses a module that does not have an built-in attenuator, an error is generated.

Parameter

ihandle

Description

IHandle is the session handle returned from the Initialize Mainframe function ([hp816x_init](#)). A new session handle is created every time the mainframe is initialized.

For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function ([hp816x_init](#)).

Data Type: ViSession

Input/Output: IN

TLSSlot

Use the TLS Slot control to choose an installed Tunable Laser module.

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_0	0	Slot 0
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3

hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

powerMode

Power Mode returns one of the following values for a Tunable Laser module with built-in attenuation:

1 (VI_TRUE), when the module automatically regulates the optimum combination between built-in attenuator and the laser current to achieve lowest noise, or

0 (VI_FALSE), when the instrument uses a manually-entered attenuation value.

Data Type: ViPBoolean

Input/Output: OUT

dark

Dark returns whether the laser output your Tunable Laser module with built-in attenuation is darkened.

1 (VI_TRUE), when the optical output is darkened, or

0 (VI_FALSE), when the optical output is not darkened.

Data Type: ViPBoolean

Input/Output: OUT

attenuation

Attenuation returns the attenuation level of your Tunable Laser module with built-in attenuation in decibels (dB).

Data Type: ViPReal64

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)"

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_set_TLS_dark

Syntax:

```
ViStatus _VI_FUNC hp816x_set_TLS_dark(ViSession ihandle, ViInt32 TLSSlot,
ViBoolean darkenLaser);
```

The Set TLS Dark (hp816x_set_TLS_dark) function sets the attenuation of your Tunable Laser module with built-in attenuation to the dark position.

No light is emitted from the optical output if the TLS is set to the dark position.

Use this function to turn the laser on or off without any settling delays.

NOTE If this function addresses a module without a built-in attenuator, an error is generated.

Parameter	Description
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized. For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).

Data Type: ViSession
Input/Output: IN

TLSSlot

Use the TLS Slot control to choose an installed Tunable Laser module.

Data Type: ViInt32
Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_0	0	Slot 0

hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

darkenLaser

Use the Darken Laser control to darken the laser output.

You use this control when you want to turn the laser On (VI_TRUE) and Off (VI_FALSE) without any settling delays when you turn the laser on.

Your Tunable Laser module must have built-in attenuation.

Data Type: ViBoolean

Input/Output: IN

Values:

Name	Value	Description
VI_TRUE	(1)	On
VI_FALSE	(0)	Off

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)"

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_get_TLS_darkState_Q

Syntax:

```
ViStatus _VI_FUNC hp816x_get_TLS_darkState_Q(ViSession ihandle, ViInt32 TLSSlot, ViPBoolean dark);
```

The Get TLS Dark State (hp816x_get_TLS_darkState_Q) function returns whether the optical output of your Tunable Laser module with built-in attenuation is set to the dark position.

NOTE If this function addresses a module without a built-in attenuator, an error is generated.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized.

For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function ([hp816x_init](#)).

Data Type: ViSession
Input/Output: IN

TLSSlot

Use the TLS Slot control to choose an installed Tunable Laser module.

Data Type: ViInt32
Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_0	0	Slot 0
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4

hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

dark

Dark returns whether the laser output your Tunable Laser module with built-in attenuation is darkened. One of the folowing values is returned:

1 (VI_TRUE), when the optical output is darkened,
or

0 (VI_FALSE), when the optical output is not darkened.

Data Type: ViPBoolean

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_get_TLS_temperatures

Syntax:

```
ViStatus _VI_FUNC hp816x_get_TLS_temperatures(ViSession Ihandle,
ViReal64 * Actual_Temperature, ViReal64 * Temperature_Difference, ViReal64 *  

Temperature_Last_Zero);
```

The Get TLS Temperatures (hp816x_get_temperatures) function returns temperatures related to the auto zeroing of backloadable tuneable laser sources.

<u>Parameter</u>	<u>Description</u>
<i>Ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized. For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).
	Data Type: ViSession Input/Output: IN
<i>Actual_Temperature</i>	Returns the current lamda zero temperature. Data Type: ViReal64 * Input/Output: OUT
<i>TemperatureDifference</i>	Returns the temperature delta when an auto lambda zero will take place. Data Type: ViReal64. *. Input/Output: OUT

Temperature_Last_Zero

Returns the temperature at which last auto lambda zero took place.

Data Type: ViReal64.*

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "[hp816x_error_message](#)"

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_get_TLS_temperaturesEx

Syntax:

```
ViStatus _VI_FUNC hp816x_get_TLS_temperaturesEx(ViSession Ihandle,
ViInt32 TLSSlot, ViReal64 * Actual_Temperature, ViReal64 *
Temperature_Difference, ViReal64 * Temperature_Last_Zero);
```

The Get TLS Temperatures Ex (hp816x_get_temperaturesEx) function returns temperatures related to the auto zeroing of a selected tuneable laser source.

<u>Parameter</u>	<u>Description</u>
<i>Ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized.
<i>TLSSlot</i>	For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).

Data Type: ViSession

Input/Output: IN

TLSSlot

Use the TLS Slot control to choose an installed Tunable Laser module.

Data Type: ViInt32

Input/Output: IN

Values:

<u>Name</u>	<u>Value</u>	<u>Description</u>
hp816x_SLOT_0	0	Slot 0
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5

hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

Actual_Temperature

Returns the current lamda zero temperature.

Data Type: ViReal64 *

Input/Output: OUT

TemperatureDifference

Returns the temperature delta when an auto lamda zero will take place.

Data Type: ViReal64. *

Input/Output: OUT

Temperature_Last_Zero

Returns the temperature at which last auto lamda zero took place.

Data Type: ViReal64.*

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)"

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_TLS_zeroing

Syntax:

ViStatus _VI_FUNC hp816x_TLS_zeroing (ViSession *ihandle*);

The TLS Zeroing (hp816x_TLS_zeroing) function starts a lambda zeroing operation for a Tunable Laser module.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized. For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).
	Data Type: ViSession Input/Output: IN

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "[hp816x_error_message](#)"

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_TLS_zeroingAll

Syntax:

ViStatus _VI_FUNC hp816x_TLS_zeroingAll (ViSession ihandle);

The TLS Zeroing All (hp816x_TLS_zeroingAll) function starts a lambda zeroing operation for all Tunable Laser modules within the mainframe.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized. For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).
	Data Type: ViSession Input/Output: IN

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "[hp816x_error_message](#)"

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



[Alphabetical Reference](#)[Hierarchical Reference](#)

hp816x_TLS_zeroingEx

Syntax:

```
ViStatus _VI_FUNC hp816x_TLS_zeroingEx (ViSession ihandle, ViInt32 TLSSlot);
```

The TLS Zeroing Ex (hp816x_TLS_zeroingEx) function starts a lambda zeroing operation for a selected Tunable Laser module.

Parameter

ihandle

Description

IHandle is the session handle returned from the Initialize Mainframe function ([hp816x_init](#)). A new session handle is created every time the mainframe is initialized.

For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function ([hp816x_init](#)).

Data Type: ViSession

Input/Output: IN

TLSSlot

Use the TLS Slot control to choose an installed Tunable Laser module.

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_0	0	Slot 0
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5

hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)"

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_set_TLS_autoCalibration

Syntax:

```
ViStatus _VI_FUNC hp816x_set_TLS_autoCalibration(ViSession ihandle,
Vilnt32 TLS_Slot, ViBoolean Autocalibration);
```

Use the Set Auto Calibration function (hp816x_set_TLS_autoCalibration) to choose whether a backloadable Tunable Laser module's autocalibration function is ON or OFF.

A backloadable TLS automatically performs a lambda zeroing routine when the instrument boots.

Setting autocalibration ON allows the TLS to perform lambda zeroing from time to time, to compensate for drift caused, for example, by changes in temperature or other environmental factors.

Setting autocalibration OFF allows the TLS to operate continuously without interruption by lambda zeroing.

You can force the TLS to perform lambda zeroing at a time of your choice using the TLS Zeroing function ([hp816x_TLS_zeroing](#))

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized. For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).

Data Type: ViSession

Input/Output: IN

TLS_Slot

Use the TLS Slot control to choose an installed Tunable Laser module.

Data Type: Vilnt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_0	0	Slot 0
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

Autocalibration

Use the Autocalibration control to choose whether a backloadable Tunable Laser module's autocalibration function is ON or OFF.

Data Type: ViBoolean

Input/Output: IN

Values:

Name	Value	Description
VI_TRUE	(1)	Autocalibration ON
VI_FALSE	(0)	Autocalibration OFF

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)

■

hp816x_get_TLS_autoCalState

Syntax:

```
ViStatus _VI_FUNC hp816x_get_TLS_autoCalState(ViSession ihandle, ViInt32 TLS_Slot, ViBoolean * Auto_Calibration_State);
```

Use the Get Auto Calibration Sta function (hp816x_get_TLS_autoCalState) to query whether a backloadable Tunable Laser module's autocalibration function is ON or OFF.

If autocalibration is ON, the TLS performs lambda zeroing from time to time, to compensate for drift caused, for example, by changes in temperature or other enviromental factors.

If autocalibration is OFF, the TLS operates continuously without interruption by lambda zeroing.

You can force the TLS to perform lambda zeroing at a time of your choice using the TLS Zeroing function ([hp816x_TLS_zeroing](#))

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized.

For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function ([hp816x_init](#)).

Data Type: ViSession
Input/Output: IN

TLS_Slot

Use the TLS Slot control to choose an installed Tunable Laser module.

Data Type: ViInt32
Input/Output: IN

Values:

<u>Name</u>	<u>Value</u>	<u>Description</u>
hp816x_SLOT_0	0	Slot 0
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

Auto_Calibration_State

Auto Calibration State returns whether a backloadable Tunable Laser module's autocalibration function is ON or OFF.

If 0 (VI_FALSE) is returned, autocalibration is turned off.

If 1 (VI_TRUE) is returned, autocalibration is turned on.

Data Type: ViBoolean

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)"

■

hp816x_get_TLS_accClass

Syntax:

```
ViStatus _VI_FUNC hp816x_get_TLS_accClass(ViSession ihandle, ViInt32
TLS_Slot, ViInt32 *Accuracy_Class);
```

Use the Get Accuracy Class function (hp816x_get_TLS_autoCalState) to query the accuracy class of a backloadable Tunable Laser module.

Parameter

ihandle

Description

IHandle is the session handle returned from the Initialize Mainframe function ([hp816x_init](#)). A new session handle is created every time the mainframe is initialized.

For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function ([hp816x_init](#)).

Data Type: ViSession

Input/Output: IN

TLS_Slot

Use the TLS Slot control to choose an installed Tunable Laser module.

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_0	0	Slot 0
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5

hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

Accuracy_Class

This control returns the accuracy class of backloadable tunable laser source.

If the TLS is of the highest accuracy, 1 is returned.

Data Type: ViInt32

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)"

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_displayToLambdaZero

Syntax:

```
ViStatus _VI_FUNC hp816x_displayToLambdaZero(ViSession ihandle, ViInt32 TLSSlot);
```

The TLS Display to Lambda Zero function ([hp816x_get_TLS_lambdaZero_Q](#)) transfers the actual displayed wavelength to Lambda0, the base wavelength for calculating the frequency offset.

See your instrument's User's Guide for more information on the frequency offset.

Parameter

ihandle

Description

IHandle is the session handle returned from the Initialize Mainframe function ([hp816x_init](#)). A new session handle is created every time the mainframe is initialized.

For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function ([hp816x_init](#)).

Data Type: ViSession

Input/Output: IN

TLSSlot

Use the TLS Slot control to choose an installed Tunable Laser module.

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_0	0	Slot 0
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4

hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)"

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_get_TLS_lambdaZero_Q

Syntax:

```
ViStatus _VI_FUNC hp816x_get_TLS_lambdaZero_Q(ViSession ihandle,
Vilnt32 TLSSlot, ViPReal64 lambda0);
```

The Get Lambda Zero function (hp816x_get_TLS_lambdaZero_Q) returns the base wavelength (Lambda0) of a Tunable Laser module.

See your instrument's User's Guide for more information on the frequency offset.

Parameter

ihandle

Description

IHandle is the session handle returned from the Initialize Mainframe function ([hp816x_init](#)). A new session handle is created every time the mainframe is initialized.

For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function ([hp816x_init](#)).

Data Type: ViSession

Input/Output: IN

TLSSlot

Use the TLS Slot control to choose an installed Tunable Laser module.

Data Type: Vilnt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_0	0	Slot 0
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4

hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

lambda0

Lambda0 outputs the base wavelength of your Tunable Laser module in meters (m).

Data Type: ViPReal64

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)"

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



[Alphabetical Reference](#)[Hierarchical Reference](#)

hp816x_set_TLS_frequencyOffset

Syntax:

```
ViStatus _VI_FUNC hp816x_set_TLS_frequencyOffset(ViSession ihandle,  
ViInt32 TLSSlot, ViReal64 offset);
```

The Set TLS Frequency Offset function (hp816x_set_TLS_frequencyOffset) sets the reference wavelength in Hertz (Hz).

See your instrument's User's Guide for more information on the frequency offset.

Parameter

ihandle

Description

IHandle is the session handle returned from the Initialize Mainframe function ([hp816x_init](#)). A new session handle is created every time the mainframe is initialized.

For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function ([hp816x_init](#)).

Data Type: ViSession

Input/Output: IN

TLSSlot

Use the TLS Slot control to choose an installed Tunable Laser module.

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_0	0	Slot 0
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4

hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

offset

You can use the Offset input to set the frequency offset of your Tunable Laser module in Hertz (Hz).

Data Type: ViReal64

Input/Output: IN

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)"

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_get_TLS_frequencyOffset_Q

Syntax:

```
ViStatus _VI_FUNC hp816x_get_TLS_frequencyOffset_Q(ViSession ihandle,
Vilnt32 TLSSlot, ViPReal64 offset);
```

The Get TLS Frequency Offset function (hp816x_get_TLS_frequencyOffset_Q) function returns the frequency offset in Hertz (Hz).

See your instrument's User's Guide for more information on the frequency offset.

Parameter

ihandle

Description

IHandle is the session handle returned from the Initialize Mainframe function ([hp816x_init](#)). A new session handle is created every time the mainframe is initialized.

For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function ([hp816x_init](#)).

Data Type: ViSession

Input/Output: IN

TLSSlot

Use the TLS Slot control to choose an installed Tunable Laser module.

Data Type: Vilnt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_0	0	Slot 0
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4

hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

offset

Offset returns the frequency offset of your Tunable Laser module in Hertz (Hz).

Data Type: ViPReal64

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)"

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_set_TLS_sweep

Syntax:

```
ViStatus _VI_FUNC hp816x_set_TLS_sweep(ViSession ihandle, ViInt32 TLSSlot, ViInt32
sweepMode, ViInt32 repeatMode, ViUInt32 cycles, ViReal64 dwellTime, ViReal64
startWavelength, ViReal64 stopWavelength, ViReal64 stepSize, ViReal64 sweepSpeed);
```

The Set TLS Sweep (hp816x_set_TLS_Sweep) function sets up a sweep operation for a Tunable Laser module.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized. For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).
	Data Type: ViSession Input/Output: IN

TLSSlot

Use the TLS Slot control to choose an installed Tunable Laser module.

Data Type: ViInt32
Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_0	0	Slot 0
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10

hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

sweepMode

Use the Sweep Mode control to choose one of the following sweep modes:

Stepped (hp816x_SWEEP_STEP), which stops at wavelengths that are separated by a certain step size,

Manual (hp816x_SWEEP_MAN), which you can run each step manually, and

Continuous (hp816x_SWEEP_CONT), which sweeps continually at the speed you set.

Data Type: VlInt32

Input/Output: IN

Values:

<u>Name</u>	<u>Value</u>	<u>Description</u>
hp816x_SWEEP_STEP	0	Stepped
hp816x_SWEEP_MAN	1	Manual
hp816x_SWEEP_CONT	2	Continuous

repeatMode

Use the Repeat Mode control to choose between:

Oneway (hp816x_ONEWAY), where every cycle of a sweep starts at the Start Wavelength, and

Twoway (hp816x_TWOWAY), where every odd cycle of the sweep starts at the start wavelength and every even cycle starts at the Stop Wavelength.

Data Type: VlInt32

Input/Output: IN

Values:

<u>Name</u>	<u>Value</u>	<u>Description</u>
-------------	--------------	--------------------

hp816x_TWOWAY	1	Twoway
hp816x_ONEWAY	0	Oneway

cycles

Use Cycles to set the number of times a sweep is repeated.

Data Type: ViUInt32

Input/Output: IN

Values:

Cycles_MIN	1
Cycles_MAX	65535

dwellTime

Use Dwell time to enter the amount of time in seconds spent at each step, for a stepped sweep.

Data Type: ViReal64

Input/Output: IN

startWavelength

Use Start Wavelength to enter the wavelength in meters (m) at which the sweep begins.

Data Type: ViReal64

Input/Output: IN

stopWavelength

Use Stop Wavelength to enter the wavelength in meters (m) at which the sweep ends.

Data Type: ViReal64

Input/Output: IN

stepSize

Use Step Size in "Stepped" sweep mode to enter the size of the change in the wavelength in meters (m) for each wavelength step.

Use Step Size in "Continuous" sweep mode to set the wavelength interval that determines when the instrument triggers. This allows you to take power

measurements at specific wavelength intervals using a logging or stability data acquisition function.

Data Type: ViReal64
Input/Output: IN

sweepSpeed

Use Sweep Speed to select the speed of a continuous sweep. You can choose one of the following values:

This parameter is only used if you use continuous sweep mode.

Data Type: ViReal64
Input/Output: IN

Values:

Name	Value	Description
hp816x_SWEEP_SPEED_LOW	0.5e-9	0.5 nm/s
hp816x_SWEEP_SPEED_MEDIUM	5.0e-9	5 nm/s
	10.0e-9	10 nm/s
	20.0e-9	20 nm/s
hp816x_SWEEP_SPEED_HIGH	40.0e-9	40 nm/s
	80.0e-9	80 nm/s

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)"

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_TLS_sweepControl

Syntax:

```
ViStatus _VI_FUNC hp816x_TLS_sweepControl(ViSession ihandle, ViInt32 TLSSlot,
ViInt32 action);
```

The TLS Sweep Control (hp816x_TLS_sweepControl) function controls a wavelength sweep operation.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized.

For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function ([hp816x_init](#)).

Data Type: ViSession

Input/Output: IN

TLSSlot

Use the TLS Slot control to choose an installed Tunable Laser module.

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_0	0	Slot 0
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9

hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

action

Use the Action control to perform one of the following wavelength sweep actions:

Start (hp816x_SW_CMD_START) , start a wavelength sweep,

Stop (hp816x_SW_CMD_STOP), stop a wavelength sweep,

Pause (hp816x_SW_CMD_PAUSE) , pause a wavelength sweep,

Continue (hp816x_SWEEP_CONT), continue a paused wavelength sweep.

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_SW_CMD_STOP	0	Stop
hp816x_SW_CMD_START	1	Start
hp816x_SW_CMD_PAUSE	2	Pause
hp816x_SW_CMD_CONT	3	Continue

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)"

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_get_TLS_sweepState_Q

Syntax:

```
ViStatus _VI_FUNC hp816x_get_TLS_sweepState_Q(ViSession ihandle,
Vilnt32 TLSSlot, ViPInt32 sweepState);
```

The Get TLS Sweep State (hp816x_get_TLS_sweepState_Q) function returns the current state of a sweep.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized.
<i>TLSSlot</i>	For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).
	Data Type: ViSession Input/Output: IN
	Data Type: Vilnt32 Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_0	0	Slot 0
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6

hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

sweepState

Sweep State returns the one of the following sweep states:

0 (hp816x_SW_CMD_STOP), no sweep is being performed,

1 (hp816x_SW_CMD_START), sweeping has been started, or

2 (hp816x_SW_CMD_PAUSE), sweeping has been paused.

Data Type: ViPInt32

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)"

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_TLS_sweepNextStep

Syntax:

```
ViStatus _VI_FUNC hp816x_TLS_sweepNextStep(ViSession ihandle, ViInt32 TLSSlot);
```

The TLS Next Step (hp816x_TLS_sweepNextStep) function sweeps one step forward.

Parameter

ihandle

Description

IHandle is the session handle returned from the Initialize Mainframe function ([hp816x_init](#)). A new session handle is created every time the mainframe is initialized.

For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function ([hp816x_init](#)).

Data Type: ViSession

Input/Output: IN

TLSSlot

Use the TLS Slot control to choose an installed Tunable Laser module.

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_0	0	Slot 0
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6

hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_TLS_sweepPreviousStep

Syntax:

```
ViStatus _VI_FUNC hp816x_TLS_sweepPreviousStep(ViSession ihandle,  
Vilnt32 TLSSlot);
```

The TLS Previous Step (hp816x_TLS_sweepPreviousStep) function sweeps one step backward.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized.
<i>TLSSlot</i>	For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).

Data Type: ViSession

Input/Output: IN

TLSSlot

Use the TLS Slot control to choose an installed Tunable Laser module.

Data Type: Vilnt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_0	0	Slot 0
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6

hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_TLS_sweepWait

Syntax:

```
ViStatus _VI_FUNC hp816x_TLS_sweepWait(ViSession ihandle, ViInt32 TLSSlot);
```

The Wait Sweep Finished (hp816x_TLS_sweepWait) function forces the program to wait until the sweep finishes.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized.
<i>TLSSlot</i>	For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).

Data Type: ViSession

Input/Output: IN

TLSSlot

Use the TLS Slot control to choose an installed Tunable Laser module.

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_0	0	Slot 0
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6

hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_set_TLS_modulation

Syntax:

```
ViStatus _VI_FUNC hp816x_set_TLS_modulation(ViSession ihandle, ViInt32 TLSSlot, ViInt32
modulationSource, ViInt32 modulationOutput, ViBoolean modulation, ViReal64
modulationFrequency);
```

The Set TLS Modulation (hp816x_set_TLS_modulation) function sets the modulation of your Tunable Laser module.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized.

For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function ([hp816x_init](#)).

Data Type: ViSession

Input/Output: IN

TLSSlot

Use the TLS Slot control to choose an installed Tunable Laser module.

Data Type: ViInt32

Input/Output: IN

Values:

<u>Name</u>	<u>Value</u>	<u>Description</u>
hp816x_SLOT_0	0	Slot 0
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13

hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

modulationSource

Use Modulation Source to choose one of the following modulation sources:

Internal (hp816x_MOD_INT), internal digital modulation,

Coherence Control (hp816x_MOD_CC), coherence control,

External Analog (hp816x_MOD_AEXT), external analog modulation,

External Digital (hp816x_MOD_DEXT), external digital modulation,

Wavelength Locking (hp816x_MOD_VWLOCK), wavelength locking,

Backplane (hp816x_MOD_BACKPL), external digital modulation using the Input Trigger connector, or

ATTENTION Not all modulation sources are available for all Tunable Laser modules.

See your instrument's User's Guide for more information on modulation sources.

Data Type: Vilnt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_MOD_INT	0	Internal
hp816x_MOD_CC	1	Coherence Control
hp816x_MOD_AEXT	2	External Analog
hp816x_MOD_DEXT	3	External Digital
hp816x_MOD_VWLOCK	4	Wavelength Locking
hp816x_MOD_BACKPL	5	Backplane

modulationOutput

Use the Modulation Output control to specify whether the modulation output signal is only active when the laser is active, a Laser Ready signal (hp816x_MOD_LREADY), or if the modulation output is always active regardless of laser state, Always (hp816x_MOD_ALWAYS).

Modulation Output only applies to backloadable

Tunable Laser modules.

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_MOD_LREADY	1	Laser Ready
hp816x_MOD_ALWAYS	0	Always

modulation

Use the Modulation control to turn modulation On (hp816x_MOD_ENABLED) or Off (hp816x_MOD_DISABLED).

Data Type: ViBoolean

Input/Output: IN

Values:

Name	Value	Description
hp816x_MOD_ENABLED	1	On
hp816x_MOD_DISABLED	0	Off

modulationFrequency

Use Modulation Frequency to set the modulation frequency of an internally modulated signal in Hertz (Hz).

Data Type: ViReal64

Input/Output: IN

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "[hp816x_error_message](#)"

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_get_TLS_modulationSettings_Q

Syntax:

```
ViStatus _VI_FUNC hp816x_get_TLS_modulationSettings_Q(ViSession ihandle, Vilnt32 TLSSlot, ViPInt32 modulationSource, ViPBoolean modulationOutput, ViPBoolean modulationState, ViPReal64 frequency);
```

The Get TLS Modulation Setting (hp816x_get_TLS_modulationSettings_Q) function returns the current modulation setting of a Tunable Laser module.

Parameter

ihandle

Description

IHandle is the session handle returned from the Initialize Mainframe function ([hp816x_init](#)). A new session handle is created every time the mainframe is initialized.

For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function ([hp816x_init](#)).

Data Type: ViSession

Input/Output: IN

TLSSlot

Use the TLS Slot control to choose an installed Tunable Laser module.

Data Type: Vilnt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_0	0	Slot 0
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5

hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

modulationSource

Modulation Source returns the one of the following modulation sources:

0 (hp816x_MOD_INT), internal digital modulation,

1 (hp816x_MOD_CC), coherence control,

2 (hp816x_MOD_AEXT), external analog modulation,

3 (hp816x_MOD_DEXT), external digital modulation using the front connector,

4 (hp816x_MOD_VWLOCK), wavelength locking,

5 (hp816x_MOD_BACKPL), external digital modulation using the Input Trigger connector at the backplane, or

See your instrument's User's Guide for more information on modulation sources.

Data Type: ViPInt32

Input/Output: OUT

modulationOutput

Modulation Output returns one of the following:

1 (hp816x_MOD_LREADY), the modulation output signal is only active when the laser is active, or

0 (hp816x_MOD_ALWAYS), the modulation output is always active regardless of laser state.

Data Type: ViPBoolean

Input/Output: OUT

modulationState

Modulation State returns whether the output signal is modulated:

1 (hp816x_MOD_ENABLED), if modulation is enabled, or

0 (hp816x_MOD_DISABLED), if modulation is disabled.

Data Type: ViPBoolean

Input/Output: OUT

frequency

Frequency returns the frequency in Hertz (Hz) for internally modulated signals.

Data Type: ViPReal64

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)"

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_TLS_configureBNC

Syntax:

```
ViStatus _VI_FUNC hp816x_TLS_configureBNC(ViSession ihandle, ViInt32 TLSSlot, ViInt32 BNCOOutput);
```

The TLS Configure BNC Output (hp816x_TLS_configureBNC) function configures the BNC output connector on the front panel of some Tunable Laser modules.

Parameter

ihandle

Description

IHandle is the session handle returned from the Initialize Mainframe function ([hp816x_init](#)). A new session handle is created every time the mainframe is initialized.

For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function ([hp816x_init](#)).

Data Type: ViSession

Input/Output: IN

TLSSlot

Use the TLS Slot control to choose an installed Tunable Laser module.

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_0	0	Slot 0
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6

hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

BNCOutput

You can use the BNC Output control to choose between the following:

Modulation (`hp816x_BNC_MOD`), where the BNC output signal is a TTL signal at the same modulation frequency as the optical output signal (if the optical output signal is digitally modulated), and

VPP (`hp816x_BNC_VPP`), where the BNC output signal is proportional to the power of the optical signal.

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_BNC_MOD	0	Modulation
hp816x_BNC_VPP	1	VPP

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)"



[Alphabetical Reference](#)[Hierarchical Reference](#)

hp816x_get_TLS_BNC_config_Q

Syntax:

```
ViStatus _VI_FUNC hp816x_get_TLS_BNC_config_Q(ViSession ihandle,  
Vilnt32 TLSSlot, ViPInt32 BNCOutput);
```

The Get TLS BNC Configuration (hp816x_get_TLS_BNC_config_Q) function returns the configuration of the BNC output connector on the front panel of some Tunable Laser modules.

Parameter

ihandle

Description

IHandle is the session handle returned from the Initialize Mainframe function ([hp816x_init](#)). A new session handle is created every time the mainframe is initialized.

For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function ([hp816x_init](#)).

Data Type: ViSession

Input/Output: IN

TLSSlot

Use the TLS Slot control to choose an installed Tunable Laser module.

Data Type: Vilnt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_0	0	Slot 0
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6

hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

BNCOutput

BNC Output returns the following BNC output connector configurations:

0 (hp816x_BNC_MOD), when the BNC output signal is a TTL signal at modulation frequency if your optical output signal is digitally modulated, or

1 (hp816x_BNC_VPP), when the BNC output signal is proportional to the power of the optical signal.

Data Type: ViPInt32

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)"

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_set_TLS_ccLevel

Syntax:

```
ViStatus _VI_FUNC hp816x_set_TLS_ccLevel(ViSession ihandle, ViInt32
TLS_Slot, ViReal64 CC_Level );
```

Use the Set Coherence Control Lev (DFB) function (hp816x_set_TLS_ccLevel) to specify the coherence control level (1-99.98%) of a DFB laser source.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized. For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).
	Data Type: ViSession Input/Output: IN

TLS_Slot

Use the TLS Slot control to choose an installed Tunable Laser module.

Data Type: ViInt32
Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_0	0	Slot 0
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5

hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

CC_Level

Use this control to set the coherence control level in percent.

Allowed values are between 1 and 99.98

Data Type: ViReal64

Input/Output: IN

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)"

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_get_TLS_ccLevel_Q

Syntax:

```
ViStatus _VI_FUNC hp816x_get_TLS_ccLevel_Q(ViSession ihandle, ViInt32
TLS_Slot, ViReal64 * CC_Level);
```

Use the Get Coherence Control Lev (DFB) function (hp816x_set_TLS_ccLevel) to query the coherence control level set for a DFB laser source.

Parameter

ihandle

Description

IHandle is the session handle returned from the Initialize Mainframe function ([hp816x_init](#)). A new session handle is created every time the mainframe is initialized.

For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function ([hp816x_init](#)).

Data Type: ViSession

Input/Output: IN

TLS_Slot

Use the TLS Slot control to choose an installed Tunable Laser module.

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_0	0	Slot 0
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5

hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

CC_Level

Returns the coherence control level set for a DFB source, in percent.

Allowed values are between 1 and 99.98

Data Type: ViReal64

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)"

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_set_TLS_SBSLevel

Syntax:

```
ViStatus _VI_FUNC hp816x_set_TLS_SBSLevel(ViSession ihandle, ViInt32
TLS_Slot, ViReal64 SBSLevel );
```

Use the Set SBS Level function (hp816x_set_TLS_SBSLevel) to specify the SBS Suppression level (1-99.98%) of a tunable laser source.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized.
<i>TLS_Slot</i>	For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).
	Data Type: ViSession Input/Output: IN

TLS_Slot

Use the TLS Slot control to choose an installed Tunable Laser module.

Data Type: ViInt32
Input/Output: IN

Values:

<u>Name</u>	<u>Value</u>	<u>Description</u>
hp816x_SLOT_0	0	Slot 0
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5

hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

SBSLevel

Use this control to set the SBS Suppression level in percent.

Allowed values are between 1 and 99.98

Data Type: ViReal64

Input/Output: IN

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)"

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_get_TLS_SBSLevel_Q

Syntax:

```
ViStatus _VI_FUNC hp816x_get_TLS_SBSLevel_Q(ViSession ihandle, ViInt32
TLS_Slot, ViReal64 * SBSLevel);
```

Use the Get SBS Level function (hp816x_set_TLS_SBSLevel) to query the SBS Suppression level set for a tunable laser source.

Parameter

ihandle

Description

IHandle is the session handle returned from the Initialize Mainframe function ([hp816x_init](#)). A new session handle is created every time the mainframe is initialized.

For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function ([hp816x_init](#)).

Data Type: ViSession

Input/Output: IN

TLS_Slot

Use the TLS Slot control to choose an installed Tunable Laser module.

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_0	0	Slot 0
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5

hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

SBSLevel

Returns the SBS Suppression level set for a tunable laser source, in percent.

Allowed values are between 1 and 99.98

Data Type: ViReal64

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)"

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_set_TLS_SBS_control

Syntax:

```
ViStatus _VI_FUNC hp816x_set_TLS_SBS_control(ViSession ihandle, ViInt32
TLSSlot, ViBoolean mod_state, ViReal64 modulationFrequency);
```

The Set TLS SBS Control (hp816x_set_TLS_SBS_control) function enables or disables the SBS Suppression feature of an 8194xA or 8198xA compact Tunable Laser Source module.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized.

For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function ([hp816x_init](#)).

Data Type: ViSession

Input/Output: IN

TLSSlot

Use the TLS Slot control to choose an installed Tunable Laser module.

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_0	0	Slot 0
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8

hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

mod_state

Use the Modulation State control to enable SBS suppression (hp816x_SBS_CONTROL_ENABLED), or disable SBS suppression (hp816x_SBS_CONTROL_DISABLED)

SBS Suppression is a feature of 8194xA and 8196xA compact TLS modules.

Data Type: ViBoolean

Input/Output: IN

Values:

<u>Name</u>	<u>Value</u>	<u>Description</u>
hp816x_SBS_CONTROL_ENABLED	1	On
hp816x_SBS_CONTROL_DISABLED	0	Off

modulationFrequency

Use Modulation Frequency to set the modulation frequency of the SBS Suppression in Hertz (Hz).

Data Type: ViReal64

Input/Output: IN

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "[hp816x_error_message](#)"



hp816x_get_TLS_SBS_control_Q

Syntax:

```
ViStatus _VI_FUNC hp816x_get_TLS_SBS_control_Q(ViSession iHandle,
ViInt32 TLSSlot,ViPBoolean *modState, ViPReal64 *frequency);
```

The Get TLS SBS Control (hp816x_get_TLS_SBS_control_Q) function returns the current SBS suppression settings for an 8194xA or 8198xA compact Tunable Laser Source module.

Parameter

iHandle

Description

IHandle is the session handle returned from the Initialize Mainframe function ([hp816x_init](#)). A new session handle is created every time the mainframe is initialized.

For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function ([hp816x_init](#)).

Data Type: ViSession

Input/Output: IN

TLSSlot

Use the TLS Slot control to choose an installed Tunable Laser module.

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_0	0	Slot 0
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6

hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

**mod_state*

Modulation State returns:

1 (hp816x_SBS_CONTROL_ENABLED), if SBS suppression is on, or

0 (hp816x_SBS_CONTROL_DISABLED), if SBS suppression is off.

Data Type: ViPBoolean

Input/Output: OUT

**frequency*

Frequency returns the frequency of SBS suppression in Hertz (Hz).

Data Type: ViPReal64

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)"

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_get_TLS_powerPoints_Q

Syntax:

```
ViStatus _VI_FUNC hp816x_get_TLS_powerPoints_Q(ViSession ihandle,
Vilnt32 TLSSlot, ViPInt32 numberofPowerPoints);
```

The Get Number of Power Points (hp816x_get_TLS_powerPoints_Q) function returns the maximum number of data items (wavelength, power) which will be returned from the Get Logged Power Data (hp816x_get_TLS_power_data_Q) function.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized. For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).

Data Type: ViSession
Input/Output: IN

TLSSlot

Use the TLS Slot control to choose an installed Tunable Laser module.

Data Type: Vilnt32
Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_0	0	Slot 0
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5

hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

numberofPowerPoints

Number of Power Points returns the maximum number of data items (wavelength, power) which will be returned from the Get Logged Power Data (hp816x_get_TLS_power_data_Q) function.

Use to generate the following 3 inputs for the Get Logged Power Data (hp816x_get_TLS_power_data_Q) function:

- set the Number of Data Items input to the same value as this output,

- allocate a one-dimensional ViReal64 array of the same dimension as this output and use as the Wavelength Data input, and

- allocate a one-dimensional ViReal64 array of the same dimension as this output and use as the Power Data input.

Data Type: ViPInt32

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)"

■

hp816x_get_TLS_powerData_Q

Syntax:

```
ViStatus _VI_FUNC hp816x_get_TLS_powerData_Q(ViSession ihandle, ViInt32
TLSSlot, ViInt32 numberOfDataItems, ViReal64 wavelengthData[ ],
ViReal64 powerData[ ]);
```

The Get Logged Power Data (hp816x_get_TLS_power_data_Q) function returns two arrays, the first array returns wavelength values and the second array returns corresponding power values.

The position of each wavelength and power value in each array corresponds to describe the maximum power available at the corresponding wavelength. In this way the two arrays describe the maximum power curve for the Tunable Laser module.

Parameter

ihandle

Description

IHandle is the session handle returned from the Initialize Mainframe function ([hp816x_init](#)). A new session handle is created every time the mainframe is initialized.

For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function ([hp816x_init](#)).

Data Type: ViSession

Input/Output: IN

TLSSlot

Use the TLS Slot control to choose an installed Tunable Laser module.

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_0	0	Slot 0
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2

hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

numberofDataItems

Data Type: ViInt32

Input/Output: IN

wavelengthData

Wavelength Data is both an input and a output.

As an input, allocate Wavelength Data as a one-dimensional ViReal64 array of the same dimension as returned by the Number of Power Values output of the Get Number of Power Values ([hp816x_get_TLS_powerPoints_Q](#)) function.

As a return, Wavelength Data returns a one dimensional array. Each component contains a wavelength. The wavelength corresponds to a power value at the same component index in the Power Data array.

Data Type: ViReal64[]

Input/Output: OUT

powerData

Power Data is both an input and an output.

As an input, allocate Power Data as a one-dimensional ViReal64 array of the same dimension as returned by the Number of Power Values output of the Get Number of Power Values ([hp816x_get_TLS_powerPoints_Q](#)) function.

As a return, Power Data returns a one-dimensional array. The power value corresponds to a wavelength at the same component index in the Wavelength Data array.

Data Type: ViReal64[]

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)"

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_set_TLS_lambdaLoggingState

Syntax:

```
ViStatus _VI_FUNC hp816x_set_TLS_lambdaLoggingState(ViSession ihandle,
ViBoolean lambdaLoggingState);
```

The Set TLS Lambda Logging State (hp816x_set_TLS_lambdaLoggingState) function turns Lambda Logging On or Off.

Lambda Logging is a feature that records the exact wavelength of a Tunable Laser module when a trigger is generated during a continuous sweep.

Lambda Logging is disabled at the end of a sweep. If you want to start a new wavelength sweep with Lambda Logging activated, you must be reset Lambda Logging to On.

NOTE: Lambda Logging is not available if your Tunable Laser module firmware revision is lower than 2.0.

NOTE: After this function completes, Trigger Out mode of the Set TLS Trigger Configuration ([hp816x_set_TLS_triggerConfiguration](#)) function will be set to Step Finished (hp816x_TRIGGER_STEP_FINISHED). Call the Set TLS Trigger Configuration ([hp816x_set_TLS_triggerConfiguration](#)) function to readjust the Trigger Out mode.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized. For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).

Data Type: ViSession

Input/Output: IN

lambdaLoggingState

Use the Lambda Logging State control to turn the Lambda Logging On (VI_TRUE) or Off (VI_FALSE).

Data Type: ViBoolean

Input/Output: IN

Values:

Name	Value	Description
VI_TRUE	(1)	On
VI_FALSE	(0)	Off

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)"

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_get_TLS_lambdaLoggingState_Q

Syntax:

```
ViStatus _VI_FUNC hp816x_get_TLS_lambdaLoggingState_Q(ViSession
iHandle, ViPBoolean lambdaLoggingState);
```

The Get TLS Lambda Logging State (hp816x_get_TLS_lambdaLoggingState_Q) function returns whether Lambda Logging is turned On or Off.

Lambda Logging is a feature that records the exact wavelength of a Tunable Laser module when a trigger is generated during a continuous sweep.

You can use this function to determine when a Lambda Logging operation is completed. This is useful because the Get Number of Wavelength Values Extended([hp816x_get_TLS_wavelengthPoints_Q](#)) function and the Get Logged Wavelength Data ([hp816x_get_TLS_wavelengthData_Q](#)) function return meaningless values unless the Lambda Logging operation is completed.

<u>Parameter</u>	<u>Description</u>
<i>iHandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized. For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).
	Data Type: ViSession Input/Output: IN
<i>lambdaLoggingState</i>	Lambda Logging State returns whether the Lambda Logging is turned on or off. If 0 (VI_FALSE) is returned, the Lambda Logging is turned off. If 1 (VI_TRUE) is returned, the Lambda Logging is turned on. Data Type: ViPBoolean

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_set_TLS_lambdaLoggingStateEx

Syntax:

```
ViStatus _VI_FUNC hp816x_set_TLS_lambdaLoggingStateEx(ViSession ihandle, ViInt32 TLSSlot, ViBoolean lambdaLoggingState);
```

The Set TLS Lambda Logging State Ex (hp816x_set_TLS_lambdaLoggingStateEx) function turns Lambda Logging On or Off for a selected Tunable Laser Module.

Lambda Logging is a feature that records the exact wavelength of a Tunable Laser module when a trigger is generated during a continuous sweep.

Lambda Logging is disabled at the end of a sweep. If you want to start a new wavelength sweep with Lambda Logging activated, you must be reset Lambda Logging to On.

NOTE: Lambda Logging is not available if your Tunable Laser module firmware revision is lower than 2.0.

NOTE: After this function completes, Trigger Out mode of the Set TLS Trigger Configuration ([hp816x_set_TLS_triggerConfiguration](#)) function will be set to Step Finished (hp816x_TRIGOUT_STEP_FINISHED). Call the Set TLS Trigger Configuration ([hp816x_set_TLS_triggerConfiguration](#)) function to readjust the Trigger Out mode.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized. For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).

Data Type: ViSession

Input/Output: IN

TLSSlot

Use the TLS Slot control to choose an installed Tunable Laser module.

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_0	0	Slot 0
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

lambdaLoggingState

Use the Lambda Logging State control to turn the Lambda Logging On (VI_TRUE) or Off (VI_FALSE).

Data Type: ViBoolean

Input/Output: IN

Values:

Name	Value	Description
VI_TRUE	(1)	On
VI_FALSE	(0)	Off

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)"

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_get_TLS_lambdaLoggingStateEx_Q

Syntax:

```
ViStatus _VI_FUNC hp816x_get_TLS_lambdaLoggingStateEx_Q(ViSession ihandle, ViInt32 TLSSlot, ViPBoolean lambdaLoggingState);
```

The Get TLS Lambda Logging State Extended (hp816x_get_TLS_lambdaLoggingStateEx_Q) function returns, for a selected Tunable Laser module, whether Lambda Logging is turned On or Off.

Lambda Logging is a feature that records the exact wavelength of a Tunable Laser module when a trigger is generated during a continuous sweep.

You can use this function to determine when a Lambda Logging operation is completed. This is useful because the Get Number of Wavelength Values Ex (hp816x_get_TLS_wavelengthPointsEx_Q) function and the Get Logged Wavelength Data Ex ([hp816x_get_TLS_wavelengthDataEx_Q](#)) function return meaningless values unless the Lambda Logging operation is completed.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized. For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).

Data Type: ViSession

Input/Output: IN

TLSSlot

Use the TLS Slot control to choose an installed Tunable Laser module.

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_0	0	Slot 0

hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

lambdaLoggingState

Lambda Logging State returns whether the Lambda Logging is turned on or off.

If 0 (VI_FALSE) is returned, the Lambda Logging is turned off.

If 1 (VI_TRUE) is returned, the Lambda Logging is turned on.

Data Type: ViPBoolean

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_get_TLS_wavelengthPoints_Q

Syntax:

```
ViStatus _VI_FUNC hp816x_get_TLS_wavelengthPoints_Q(ViSession ihandle,
ViPReal64 numberofWavelengthValues);
```

The Get Number of Wavelength Values (hp816x_get_TLS_wavelengthPoints_Q) function returns the number of wavelength points that will be used in a Lambda Logging operation.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized. For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).
	Data Type: ViSession Input/Output: IN
<i>numberofWavelengthValues</i>	Number of Wavelength Values returns the number of wavelength values that were returned by the last Lambda Logging operation. If you call this function before the Lambda Logging operation completes, the returned value is meaningless. Use the Get Lambda Logging State function (hp816x_get_TLS_lambdaLoggingState_Q) to confirm that the Lambda Logging operation is completed.
	Data Type: ViPReal64 Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)"

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_get_TLS_wavelengthPointsEx_Q

Syntax:

```
ViStatus _VI_FUNC hp816x_get_TLS_wavelengthPointsEx_Q(ViSession ihandle, Vilnt32 TLSSlot, ViPReal64 numberofWavelengthValues);
```

The Get Number of Wavelength Values Ex (hp816x_get_TLS_wavelengthPointsEx_Q) function returns, for a selected Tunable Laser Module, the number of wavelength points that will be used in a Lambda Logging operation.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	<p>IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized.</p> <p>For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).</p>

Data Type: ViSession

Input/Output: IN

TLSSlot

Use the TLS Slot control to choose an installed Tunable Laser module.

Data Type: Vilnt32

Input/Output: IN

Values:

<u>Name</u>	<u>Value</u>	<u>Description</u>
hp816x_SLOT_0	0	Slot 0
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3
hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5

hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

numberofWavelengthValues

Number of Wavelength Values returns the number of wavelength values that were returned by the last Lambda Logging operation.

If you call this function before the Lambda Logging operation completes, the returned value is meaningless. Use the Get Lambda Logging State Ex function ([hp816x_get_TLS_lambdaLoggingStateEx_Q](#)) to confirm that the Lambda Logging operation is completed.

Data Type: ViPReal64

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "[hp816x_error_message](#)"

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_get_TLS_wavelengthData_Q

Syntax:

```
ViStatus _VI_FUNC hp816x_get_TLS_wavelengthData_Q(ViSession ihandle,  
Vilnt32 arraySize, ViReal64wavelengthData[ ]);
```

The Get Logged Wavelength Data (`hp816x_get_TLS_wavelengthData_Q`) function returns the results of a Lambda Logging operation.

If you call this function before the Lambda Logging operation completes, the returned value is meaningless. Use the Get Lambda Logging State function ([hp816x_get_TLS_lambdaLoggingState_Q](#)) to confirm that the Lambda Logging operation is completed.

<u>Parameter</u>	<u>Description</u>
<i>iHandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized.
	For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).
	Data Type: ViSession Input/Output: IN
<i>arraySize</i>	Use Array Size to set the size of the memory buffer in bytes needed for the function to complete.
	Use the Number of Wavelength Values output of the Get Number of Wavelength Values (hp816x_get_TLS_wavelengthPoints_Q) to set this value. You should multiply this value by 8 to ensure that the function completes.
	Data Type: ViInt32 Input/Output: IN

wavelengthData

Wavelength Data is both an input and an output.

As an input, Wavelength Data accepts a one-dimensional real array that determines the size of the Wavelength Data output. Allocate a one-dimensional real array using the the Number of Wavelength Values output of the Get Number of Wavelength Values (hp816x_get_TLS_wavelengthPoints_Q) to set the dimension of the array.

As an output, Wavelength Data returns a one dimensional array containing the exact wavelength values of the Tunable Laser module when triggers were generated during the continuous sweep.

Data Type: ViReal64[]

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "[hp816x_error_message](#)"

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_get_TLS_wavelengthDataEx_Q

Syntax:

```
ViStatus _VI_FUNC hp816x_get_TLS_wavelengthDataEx_Q(ViSession
iHandle, ViInt32 TLSSlot, ViInt32 arraySize, ViReal64 wavelengthData[ ]);
```

The Get Logged Wavelength Data Extended (hp816x_get_TLS_wavelengthDataEx_Q) function returns, for a selected Tunable Laser Module, the results of a Lambda Logging operation.

If you call this function before the Lambda Logging operation completes, the returned value is meaningless. Use the Get Lambda Logging State Ex function ([hp816x_get_TLS_lambdaLoggingStateEx_Q](#)) to confirm that the Lambda Logging operation is completed.

<u>Parameter</u>	<u>Description</u>
<i>iHandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized.

For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function ([hp816x_init](#)).

Data Type: ViSession
Input/Output: IN

TLSSlot

Use the TLS Slot control to choose an installed Tunable Laser module.

Data Type: ViInt32
Input/Output: IN

Values:

Name	Value	Description
hp816x_SLOT_0	0	Slot 0
hp816x_SLOT_1	1	Slot 1
hp816x_SLOT_2	2	Slot 2
hp816x_SLOT_3	3	Slot 3

hp816x_SLOT_4	4	Slot 4
hp816x_SLOT_5	5	Slot 5
hp816x_SLOT_6	6	Slot 6
hp816x_SLOT_7	7	Slot 7
hp816x_SLOT_8	8	Slot 8
hp816x_SLOT_9	9	Slot 9
hp816x_SLOT_10	10	Slot 10
hp816x_SLOT_11	11	Slot 11
hp816x_SLOT_12	12	Slot 12
hp816x_SLOT_13	13	Slot 13
hp816x_SLOT_14	14	Slot 14
hp816x_SLOT_15	15	Slot 15
hp816x_SLOT_16	16	Slot 16
hp816x_SLOT_17	17	Slot 17

arraySize

Use Array Size to set the size of the memory buffer in bytes needed for the function to complete.

Use the Number of Wavelength Values output of the Get Number of Wavelength Values Extended ([hp816x_get_TLS_wavelengthPointsEx_Q](#)) to set this value. You should multiply this value by 8 to ensure that the function completes.

Data Type: VlInt32

Input/Output: IN

wavelengthData

Wavelength Data is both an input and an output.

As an input, Wavelength Data accepts a one-dimensional real array that determines the size of the Wavelength Data output. Allocate a one-dimensional real array using the the Number of Wavelength Values output of the Get Number of Wavelength Values (hp816x_get_TLS_wavelengthPoints_Q) to set the dimension of the array.

As an output, Wavelength Data returns a one dimensional array containing the exact wavelength values of the Tunable Laser module when triggers were generated during the

continuous sweep.

Data Type: ViReal64[]

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)"

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_set_LambdaScan_wavelength

Syntax:

```
ViStatus _VI_FUNC hp816x_set_LambdaScan_wavelength(ViSession ihandle,
ViReal64 powerMeterWavelength);
```

The Set Lambda Scan Wavelength (hp816x_set_LambdaScan_wavelength) function allows you to use a different wavelength than 1550 nm during a Lambda Scan operation. All Power Meters taking part in the Lambda Scan operation will be set to the chosen wavelength.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized. For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).
	Data Type: ViSession Input/Output: IN
<i>powerMeterWavelength</i>	Use Power Meter Wavelength to change the wavelength for the Lambda Scan operation. Data Type: ViReal64 Input/Output: IN

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "[hp816x_error_message](#)"



hp816x_enableHighSweepSpeed

Syntax:

```
ViStatus _VI_FUNC hp816x_enableHighSweepSpeed(ViSession ihandle,
ViBoolean fastSweepSpeed);
```

The Enable High Sweep Speed (hp816x_enableHighSweepSpeed) function enables/disables the highest available sweep speed (40 nanometers per second) for Lambda Scan operations. The Lambda Scan operation chooses the highest possible sweep speed for the chosen step size.

If you choose Enable, the highest sweep speed possible will be used. This may lead to less accurate measurements.

If you choose Disable, the highest sweep speed will not be used.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized. For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).

Data Type: ViSession

Input/Output: IN

fastSweepSpeed

Use the Fast Sweep Speed control to enable/disable the highest available sweep speed (40 nanometers per second) for Lambda Scan operations.

If you choose Enable (VI_TRUE), the highest sweep speed possible will be used. This may lead to less accurate measurements.

If you choose Disable (VI_FALSE), the highest sweep speed will not be used.

Data Type: ViBoolean

Input/Output: IN

Values:

Name	Value	Description
VI_TRUE	(1)	Enable
VI_FALSE	(0)	Disable

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)"

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_prepareLambdaScan

Syntax:

```
ViStatus _VI_FUNC hp816x_prepareLambdaScan(ViSession ihandle, ViInt32
powerUnit, ViReal64 power, ViInt32 opticalOutput, ViInt32 numberofScans, ViInt32
PWMChannels, ViReal64 startWavelength, ViReal64 stopWavelength, ViReal64 stepSize,
ViUInt32 numberofDatapoints, ViUInt32 numberofChannels);
```

The Prepare Lambda Scan (hp816x_prepareLambdaScan) function prepares a Lambda Scan operation.

That is, it prepares an operation where a 8164A/B Lightwave Measurement System with a back-loadable Tunable Laser module and up to four Power Sensors installed, performs a wavelength sweep where the Tunable Laser module and Power Sensors are coordinated with each other.

The Prepare Lambda Scan (hp816x_prepareLambdaScan) function must be called before a Lambda Scan operation is executed. Use the return values of this function (Number of Datapoints and Number of Power Arrays) to allocate arrays for the Execute Lambda Scan ([hp816x_executeLambdaScan](#)) function.

To obtain a higher precision, the Tunable Laser Source is set 1 nm before the Start Wavelength, this means, you have to choose a Start Wavelength 1 nm greater than the minimum possible wavelength. Also, the wavelength sweep is actually started 90 pm before the Start Wavelength and ends 90 pm after the Stop Wavelength, this means, you have to choose a Stop Wavelength 90 pm less than the maximum possible wavelength.

Triggers coordinate the Tunable Laser module with all Power Meters. The function sets for the lowest possible averaging time available for the installed Power Meters and, then, sets the highest possible sweep speed for the selected Tunable Laser module sweep.

If one of the following circumstances occurs, the "parameter mismatch" error will be returned:

1. If one Power Meter is out of the specification at 1550 nm, the error "powermeter wavelength does not span 1550nm" will be returned. For example, the 81530A Power Sensor and the 81520A Optical Head are out of specification at 1550 nm. Remove the Power Meter that is out of specification at 1550 nm from the mainframe.
2. If the Step Size is too small and results in a trigger frequency that is to high for the installed Power Meters, the error "could not calculate a sweep speed!" will be returned. Increase the Step Size.
3. If the chosen wavelength range is too large and Step Size is too small, the error "too many datapoints to log!" will be returned. In this case, reduce the wavelength range and/or increase the Step Size.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized.
	For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).

Data Type: ViSession
Input/Output: IN

powerUnit

You can use the Unit control to choose either Watts (hp816x_PU_WATT) or dBm (hp816x_PU_DBM) as the power unit.

Data Type: ViInt32
Input/Output: IN

Values:

Name	Value	Description
hp816x_PU_DBM	0	dBm
hp816x_PU_WATT	1	Watt

power

You can use the Power input to set the output power to a value in dBm.

Data Type: ViReal64
Input/Output: IN

opticalOutput

Use the Optical Output control to choose the optical output for a TLS module with two outputs. You can choose one of the following:

High Power (hp816x_HIGHPOW), where Output 2, the High Power output is the regulated output.

Low SSE (hp816x_LOWSSE), where Output 1, the Low SSE output is the regulated output.

BHR (hp816x_BHR) is the same as choosing Both (Master 2) from the instrument's front panel. Output 2, the High Power output is the regulated output and Output 1, the Low SSE output is the unregulated output.

BLR (hp816x_BLR) is the same as choosing Both (Master 1) from the instrument's front panel, the Low SSE output is the regulated output and Output 2, the High Power output is the unregulated output.

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_HIGHPOW	0	High Power
hp816x_LOWSSE	1	Low SSE
hp816x_BHR	2	BHR
hp816x_BLR	3	BLR

numberofScans

Number of Scans sets the number of sweep cycles that will be performed.

To perform 1 scan enter 0
(hp816x_NO_OF_SCANS_1).

To perform 2 scans enter 1
(hp816x_NO_OF_SCANS_2).

To perform 3 scans enter 2
(hp816x_NO_OF_SCANS_3).

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_NO_OF_SCANS_1	0	1
hp816x_NO_OF_SCANS_2	1	2
hp816x_NO_OF_SCANS_3	2	3

PWMChannels

Use the PWM Channels control to specify how many Power Meter channels should be used.

Data Type: ViInt32

Input/Output: IN

startWavelength

Use Start Wavelength to enter the wavelength in meters (m) at which the sweep begins.

Data Type: ViReal64

Input/Output: IN

stopWavelength

Use Stop Wavelength to enter the wavelength in meters (m) at which the sweep ends.

Data Type: ViReal64

Input/Output: IN

stepSize

Use Step Size to enter the size of the change in the wavelength in meters (m) for each wavelength step.

Data Type: ViReal64

Input/Output: IN

numberofDatapoints

Number of Datapoints returns the number of wavelength steps that the Lambda Scan will perform.

Each Power Meter channel will make the same number of measurements, use the returned value for allocating the size of power value arrays.

Data Type: ViUInt32

Input/Output: OUT

numberofChannels

Number of Channels returns the number of Power Meter channel that a Lambda Scan operation will use.

Use this value to allocate the number of power value arrays for the Execute Lambda Scan function ([hp816x_executeLambdaScan](#)).

Data Type: ViUInt32

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "[hp816x_error_message](#)"

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_getLambdaScanParameters_Q

Syntax:

```
ViStatus _VI_FUNC hp816x_getLambdaScanParameters_Q(ViSession ihandle,
ViPReal64 startWavelength, ViPReal64 stopWavelength, ViPReal64
averagingTime, ViPReal64 sweepSpeed);
```

The Get Lambda Scan Parameters function (hp816x_getLambdaScanParameters_Q) returns all parameters that the Prepare Lambda Scan ([hp816x_prepareLambdaScan](#)) function adjusts or automatically calculates.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized. For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).
	Data Type: ViSession Input/Output: IN
<i>startWavelength</i>	Start Wavelength returns the adjusted wavelength value in meters that the Prepare Lambda Scan (hp816x_prepareLambdaScan) function generates. This Start Wavelength is used as the Tunable Laser module start wavelength for the Lambda Scan operation.
	Data Type: ViPReal64 Input/Output: OUT
<i>stopWavelength</i>	Stop Wavelength returns the adjusted wavelength value in meters that the Prepare Lambda Scan

([hp816x_prepareLambdaScan](#)) function generates.

This Stop Wavelength is used as the Tunable Laser module stop wavelength for the Lambda Scan operation.

Data Type: ViPReal64

Input/Output: OUT

averagingTime

Averaging Time returns the averaging time value in seconds that the Prepare Lambda Scan ([hp816x_prepareLambdaScan](#)) function generates.

This Averaging Time is used for all installed Power Meters for the Lambda Scan operation.

Data Type: ViPReal64

Input/Output: OUT

sweepSpeed

Sweep Speed returns the sweep speed in meters per second (m/s) that the Prepare Lambda Scan ([hp816x_prepareLambdaScan](#)) function generates.

This Sweep Speed is used as the Tunable Laser module sweep speed for the Lambda Scan operation.

Data Type: ViPReal64

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "[hp816x_error_message](#)"

E0615, Jun 2015



hp816x_executeLambdaScan

Syntax:

```
ViStatus _VI_FUNC hp816x_executeLambdaScan(ViSession ihandle,
ViReal64 wavelengthArray[ ], ViReal64 powerArray1[ ], ViReal64 powerArray2[ ],
ViReal64 powerArray3[ ], ViReal64 powerArray4[ ], ViReal64 powerArray5[ ],
ViReal64 powerArray6[ ], ViReal64 powerArray7[ ], ViReal64 powerArray8[ ]);
```

The Execute Lambda Scan (hp816x_executeLambdaScan) function runs and returns the results of a Lambda Scan operation.

That is, it executes an operation where a 8164A/B Lightwave Measurement System with a back-loadable Tunable Laser module and up to four Power Sensors installed, performs a wavelength sweep where the Tunable Laser module and Power Sensors are coordinated with each other.

The Prepare Lambda Scan ([hp816x_prepareLambdaScan](#)) function must be called before a Lambda Scan operation is executed. Use the return values of this function (Number of Datapoints and Number of Power Arrays) to allocate arrays for the Execute Lambda Scan (hp816x_executeLambdaScan) function.

Equally Spaced Datapoints is enabled as part of this function and cannot be disabled, see the Windows Help file or your Programming Guide for more details. Use Multi Frame Lambda Scan if you need to have inequally spaced datapoints.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized. For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).
	Data Type: ViSession Input/Output: IN
<i>wavelengthArray</i>	Wavelength Array returns the wavelength points at which power values were measured. The wavelength points returned by this array are always equally spaced.

Data Type: ViReal64[]

Input/Output: OUT

powerArray1

Power Array 1 returns the measured power values for the first Power Meter channel.

Equally Spaced Datapoints is enabled as part of this function, see the Windows Help file or your Programming Guide for more details.

Data Type: ViReal64[]

Input/Output: OUT

powerArray2

Power Array 2 returns the measured power values for the second Power Meter channel.

Equally Spaced Datapoints is enabled as part of this function, see the Windows Help file or your Programming Guide for more details.

Data Type: ViReal64[]

Input/Output: OUT

powerArray3

Power Array 3 returns the measured power values for the third Power Meter channel.

Equally Spaced Datapoints is enabled as part of this function, see the Windows Help file or your Programming Guide for more details.

Data Type: ViReal64[]

Input/Output: OUT

powerArray4

powerArray4

Power Array 4 returns the measured power values for the fourth Power Meter channel.

Equally Spaced Datapoints is enabled as part of this function, see the Windows Help file or your Programming Guide for more details.

Data Type: ViReal64[]

Input/Output: OUT

powerArray5

Power Array 5 returns the measured power values for the fifth Power Meter channel.

Equally Spaced Datapoints is enabled as part of this function, see the Windows Help file or your Programming Guide for more details.

Data Type: ViReal64[]

Input/Output: OUT

powerArray6

Power Array 6 returns the measured power values for the sixth Power Meter channel.

Equally Spaced Datapoints is enabled as part of this function, see the Windows Help file or your Programming Guide for more details.

Data Type: ViReal64[]

Input/Output: OUT

powerArray7

Power Array 7 returns the measured power values for the seventh Power Meter channel.

Equally Spaced Datapoints is enabled as part of this function, see the Windows Help file or your Programming Guide for more details.

Data Type: ViReal64[]

Input/Output: OUT

powerArray8

Power Array 8 returns the measured power values for the eighth Power Meter channel.

Equally Spaced Datapoints is enabled as part of this function, see the Windows Help file or your Programming Guide for more details.

Data Type: ViReal64[]

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



Information for N77 Series Instruments

This version of the 816x VXIplug&play instrument driver has been extended to support the N773xA, N774xA optical power meter and the N775xA and N776xA Optical Attenuator instruments. These are stand-alone instruments that do not require installation in an 816x mainframe. The SCPI command set has been designed for almost complete programming compatibility with the modular instruments of the 816x family. Please be aware of the following details and limitations, when using this driver.

- The N77-series instruments can be controlled via USB, LAN or GPIB interfaces. Especially for transferring large data volumes, the USB and usually LAN interfaces provide faster performance.
- The identification of the individual ports of these multiport instruments in the SCPI and hp816x functions corresponds to the slot number. So for example the ports of the N7745A can be addressed with the slot numbers 1 to 8. The channel number in the commands and functions is not used and can generally be omitted or set to Channel 1 (index 0).
- The N771x-series and 81950A tunable lasers are not supported by the 816x VXIplug&play driver.
- The N773x-series optical switches are supported by the 816x VXIplug&play functions for switches and relevant functions for the mainframes.
- The N774x-series power meters are supported by the 816x VXIplug&play functions for power meters, relevant functions for the mainframes like for standard trigger configuration, and the Multi Frame Lambda Scan application functions.
- The N776x-series optical attenuators are supported by the 816x VXIplug&play functions for attenuators and the relevant functions for the mainframes.
- The N775x-series optical attenuators and power meters are supported by the 816x VXIplug&play functions for attenuators and the relevant functions for the mainframes and the basic functions for the power meters, excluding the advanced functions using data power logging. The choices of averaging time is also more limited for these power meters.
- The averaging time parameter for the power meters is limited in the 816x VXIplug&play functions to a limited set of discrete values, similar to the modular 816x power meters. This set includes: 1, 2, 5, 10, 25, 100, 200, and 500 μ s, 1, 2, 5, 10, 20, 50, 100, 200, 500 ms, and 1, 2, 5, and 10s.
- The maximum number of logging data points that can be acquired is limited by the 816x VXIplug&play data acquisition functions to 100,000 per port.
- The shortest averaging time used by the Multi Frame Lambda Scan application is 25 μ s.
- For the Multi Frame Lambda Scan application, the N774xA instruments can be included simply by registering the instrument as a mainframe, after the first

mainframe with the tunable laser has been registered.

Further details to available functions are found under the categories below:

[Utility Functions](#)

[Mainframe Specific Functions](#)

[Power Sensor Specific Functions](#)

[Applications](#)

hp816x_returnEquidistantData

Syntax:

```
ViStatus _VI_FUNC hp816x_returnEquidistantData(ViSession ihandle,
ViBoolean equallySpacedDatapoints);
```

The Equally Spaced Datapoints (hp816x_returnEquidistantData) function performs a linear interpolation on the wavelength point and power measurement data and returns an equidistant wavelength array.

This function is used because Lambda Scan functions make use of Lambda Logging to log the exact wavelength that measurements were triggered at. This results in Lambda Array wavelength points that are not equally spaced.

NOTE: Lambda Logging is not available if your Tunable Laser module firmware revision is lower than 2.0.

Equally Spaced Datapoints is enabled as a default.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized. For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).
	Data Type: ViSession Input/Output: IN
<i>equallySpacedDatapoints</i>	The Equally Spaced Datapoints control enables/disables the Equally Spaced Datapoints (hp816x_returnEquidistantData) function. If you choose Enable (VI_TRUE), a linear interpolation is performed on the wavelength point and power measurement data and an equidistant wavelength array is returned.

If you choose Disable (VI_FALSE), the results are returned directly from the instrument.

Data Type: ViBoolean

Input/Output: IN

Values:

Name	Value	Description
VI_TRUE	(1)	Enable
VI_FALSE	(0)	Disable

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)"

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_registerMainframe

Syntax:

ViStatus _VI_FUNC hp816x_registerMainframe(ViSession *iHandle*);

Use the Register Mainframe (hp816x_registerMainframe) function to register your mainframe as a participant in a Multi Frame Lambda Scan operation. The mainframe must be connected to the GPIB bus and have its Input Trigger Connector connected to the Output Trigger Connector of the 8164A/B Lightwave Measurement System mainframe that the Tunable Laser module is installed in.

The IHandle for the session is internally registered and is used to identify the instrument for the Multi Frame Lambda Scan operation.

<u>Parameter</u>	<u>Description</u>
<i>iHandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized. For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).
	Data Type: ViSession Input/Output: IN

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "[hp816x_error_message](#)"



hp816x_unregisterMainframe

Syntax:

```
ViStatus _VI_FUNC hp816x_unregisterMainframe(ViSession ihandle);
```

Use the Unregister Mainframe function (hp816x_unregisterMainframe) to remove a mainframe from a Multi Frame Lambda Scan operation and clear the driver's internal data structures.

If you use LabView 5.0 the following items should be noted:

- All multi frame functions are not re-entrant, if the driver is running and initialized more than once, results may be unpredictable.
- To avoid wrong results, call the Unregister Mainframe function prior to the Initialize function ([hp816x_init](#)). This is especially necessary during program debugging, if the Close function ([hp816x_close](#)) is not called.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized.
	For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).
	You can unregister all mainframes if you call this function with IHandle set to zero.
	Data Type: ViSession Input/Output: IN

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "[hp816x_error_message](#)"

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_setSweepSpeed

Syntax:

```
ViStatus _VI_FUNC hp816x_setSweepSpeed(ViSession ihandle, ViInt32 Sweep_Speed);
```

Use the Set Sweep Speed function (hp816x_setSweepSpeed) to specify a sweep speed for a multi frame lambda scan.

Call this function before calling the Prepare Multi Frame Lambda Scan function ([hp816x_prepareMfLambdaScan](#)).

The Prepare Multi Frame Lambda Scan function checks whether your desired Sweep Speed meets the following condition:

Averageing Time (of all the powermeters) < (Step Size / Sweep Speed)

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized. For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).

Data Type: ViSession

Input/Output: IN

Sweep_Speed

Use Sweep Speed to select the speed of a sweep.

Auto Select:

Calculates, and sets, the highest possible sweep speed, which depends on the step size and the minimum averaging time for all powermeters.

Remark: If it is not possible to use a specified

sweep speed, the Prepare Multi Frame Lambda Scan function ([hp816x_prepareMfLambdaScan](#)) will return an error.

Data Type: ViInt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_SPEED_80NM	-1	80 nm/s
hp816x_SPEED_40NM	0	40 nm/s
hp816x_SPEED_20NM	1	20 nm/s
hp816x_SPEED_10NM	2	10 nm/s
hp816x_SPEED_5NM	3	5 nm/s
hp816x_SPEED_05NM	4	0.5 nm/s
hp816x_SPEED_AUTO	5	Auto Select

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)"

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_prepareMfLambdaScan

Syntax:

```
ViStatus _VI_FUNC hp816x_prepareMfLambdaScan(ViSession ihandle, ViInt32
powerUnit, ViReal64 power, ViInt32 opticalOutput, ViInt32 numberofScans, ViInt32
PWMChannels, ViReal64 startWavelength, ViReal64 stopWavelength, ViReal64
stepSize, ViUInt32 numberofDatapoints, ViUInt32 numberofChannels);
```

The Prepare Multi Frame Lambda Scan (hp816x_prepareMfLambdaScan) function prepares a Lambda Scan operation for multiple Mainframes.

That is, it prepares an operation where an 8164A/B Lightwave Measurement System with a sweepable Tunable Laser module and up to 1000 Power Sensor Channels located in different Mainframes are installed. The function performs a wavelength sweep where the Tunable Laser module and Power Sensors are co-ordinated with each other.

The Prepare Multi Frame Lambda Scan (hp816x_prepareMfLambdaScan) function must be called before a Multi Frame Lambda Scan is executed. Use the return values of this function (Number of Datapoints and Number of Power Arrays) to allocate arrays for the Execute Multi Frame Lambda Scan ([hp816x_executeMfLambdaScan](#)) function.

The function scans all mainframes to find swept-wavelength Tunable Laser Sources. The function scans each mainframe in the order that they were originally registered by the Register Mainframe function ([hp816x_registerMainframe](#)). The first Tunable Laser Source found will perform the sweep operation.

To obtain a higher precision, the wavelength sweep is started 90 pm before the Start Wavelength (or, if the Step Size is > 90 pm, one step before the Start Wavelength) and ends 90 pm after the Stop Wavelength (or, if the Step Size is > 90 pm, one step after the Stop Wavelength). This means that you have to choose a Stop Wavelength 90 pm less than the maximum possible wavelength (or, if the Step Size is > 90 pm, one step less than the maximum possible wavelength).

Triggers coordinate the Tunable Laser module with all Power Meters. The function sets for the lowest possible averaging time available for the installed Power Meters and, then, sets the highest possible sweep speed for the selected Tunable Laser module sweep. All mainframes must be connected to the GPIB bus and have their Input Trigger Connector connected to the Output Trigger Connector of the 8164A/B Lightwave Measurement System mainframe that the Tunable Laser module is installed in.

If one of the following circumstances occurs, the "parameter mismatch" error will be returned:

1. If one Power Meter is out of the specification at 1550 nm, the error "powermeter wavelength does not span 1550nm" will be returned. For example, the 81530A Power Sensor and the 81520A Optical Head are out of specification at 1550 nm. Remove the Power Meter that is out of specification at 1550 nm from the mainframe.
2. If the Step Size is too small and results in a trigger frequency that is to high for the installed Power Meters, the error "could not calculate a sweep speed!" will be returned. Increase the Step Size.
3. If the chosen wavelength range is too large and Step Size is too small, the error "too many datapoints to log!" will be returned. In this case, reduce the wavelength range and/or increase the Step Size.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new

session handle is created every time the mainframe is initialized.

For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function ([hp816x_init](#)).

Data Type: ViSession
Input/Output: IN

powerUnit

You can use the Unit control to choose either Watts (hp816x_PU_WATT) or dBm (hp816x_PU_DBM) as the power unit.

Data Type: ViInt32
Input/Output: IN

Values:

Name	Value	Description
hp816x_PU_DBM	0	dBm
hp816x_PU_WATT	1	Watt

power

You can use the Power input to set the output power to a value in dBm.

Data Type: ViReal64
Input/Output: IN

opticalOutput

Use the Optical Output control to choose the optical output for a TLS module with two outputs. You can choose one of the following:

High Power (hp816x_HIGHPOW), where Output 2, the High Power output is the regulated output.

Low SSE (hp816x_LOWSSE), where Output 1, the Low SSE output is the regulated output.

BHR (hp816x_BHR) is the same as choosing Both (Master 2) from the instrument's front panel, Output 2, the High Power output is the regulated output and Output 1, the Low SSE output is the

unregulated output.

BLR (hp816x_BLR) is the same as choosing Both (Master 1) from the instrument's front panel, the Low SSE output is the regulated output and Output 2, the High Power output is the unregulated output.

Data Type: Vilnt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_HIGHPOW	0	High Power
hp816x_LOWSSE	1	Low SSE
hp816x_BHR	2	BHR
hp816x_BLR	3	BLR

numberofScans

Number of Scans sets the number of sweep cycles that will be performed.

To perform 1 scan enter 0
(hp816x_NO_OF_SCANS_1).

To perform 2 scans enter 1
(hp816x_NO_OF_SCANS_2).

To perform 3 scans enter 2
(hp816x_NO_OF_SCANS_3).

Data Type: Vilnt32

Input/Output: IN

Values:

Name	Value	Description
hp816x_NO_OF_SCANS_1	0	1
hp816x_NO_OF_SCANS_2	1	2
hp816x_NO_OF_SCANS_3	2	3

PWMChannels

Use the PWM Channels input to specify how many Power Meter channels should be used.

If some channels have been excluded, the PWMChannels control must be high enough to include the original index of the highest included channel index. See: [hp816x_excludeChannel](#)

The maximum number of channels that may be specified is 1000.

Data Type: ViInt32
Input/Output: IN

startWavelength

Use Start Wavelength to enter the wavelength in meters (m) at which the sweep begins.

Data Type: ViReal64
Input/Output: IN

stopWavelength

Use Stop Wavelength to enter the wavelength in meters (m) at which the sweep ends.

Data Type: ViReal64
Input/Output: IN

stepSize

Use Step Size to enter the size of the change in the wavelength in meters (m) for each wavelength step.

Data Type: ViReal64
Input/Output: IN

numberofDatapoints

Number of Datapoints returns the number of wavelength steps that the Multi Frame Lambda Scan will perform.

Each Power Meter channel will make the same number of measurements, use the returned value for allocating power value arrays.

Data Type: ViUInt32
Input/Output: OUT

numberofChannels

Number of Channels returns the number of Power Meter channel that a Lambda Scan operation will use.

Use this value to allocate the number of power value arrays for the Execute Multi Frame Lambda Scan function (hp816x_executemflambdaScan).

Data Type: ViUInt32

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)"

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_getMFLambdaScanParameters_Q

Syntax:

```
ViStatus _VI_FUNC hp816x_getMFLambdaScanParameters_Q(ViSession
ihandle, ViPReal64 startWavelength, ViPReal64 stopWavelength, ViPReal64
averagingTime, ViPReal64 sweepSpeed);
```

The Get MF Lambda Scan Parameters (hp816x_getMFLambdaScanParameters_Q) function returns all parameters that the Prepare Multi Frame Lambda Scan ([hp816x_prepareMfLambdaScan](#)) function adjusts or automatically calculates.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized. For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).
	Data Type: ViSession Input/Output: IN
<i>startWavelength</i>	Start Wavelength returns the adjusted wavelength value in meters that the Prepare Multi Frame Lambda Scan (hp816x_prepareMfLambdaScan) function generates. This Start Wavelength is used as the Tunable Laser module start wavelength for the Lambda Scan operation.
	Data Type: ViPReal64 Input/Output: OUT
<i>stopWavelength</i>	Stop Wavelength returns the adjusted wavelength

value in meters that the Prepare Multi Frame Lambda Scan ([hp816x_prepareMfLambdaScan](#)) function generates.

This Stop Wavelength is used as the Tunable Laser module stop wavelength for the Lambda Scan operation.

Data Type: ViPReal64

Input/Output: OUT

averagingTime

Averaging Time returns the averaging time value in seconds that the Prepare Multi Frame Lambda Scan ([hp816x_prepareMfLambdaScan](#)) function generates.

This Averaging Time is used for all installed Power Meters for the Lambda Scan operation.

Data Type: ViPReal64

Input/Output: OUT

sweepSpeed

Sweep Speed returns the sweep speed in meters per second (m/s) that the Prepare Multi Frame Lambda Scan ([hp816x_prepareMfLambdaScan](#)) function generates.

This Sweep Speed is used as the Tunable Laser module sweep speed for the Lambda Scan operation.

Data Type: ViPReal64

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "[hp816x_error_message](#)"

E0615, Jun 2015



hp816x_executeMfLambdaScan

Syntax:

```
ViStatus _VI_FUNC hp816x_executeMfLambdaScan(ViSession ihandle,
ViReal64wavelengthArray[ ]);
```

The Execute Multi Frame Lambda Scan (hp816x_executeMfLambdaScan) function runs a Lambda Scan operation and returns an array that contains the wavelength values at which power measurements are made.

That is, it executes an operation where a 8164A/B Lightwave Measurement System with a sweepable Tunable Laser module and up to 1000 Power Sensors installed, performs a wavelength sweep where the Tunable Laser module and Power Sensors are coordinated with each other.

Use the values returned from the Prepare Multi Frame Lambda Scan ([hp816x_prepareMfLambdaScan](#)) function to set the parameters of the Execute Multi Frame Lambda Scan (hp816x_executeMfLambdaScan) function.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized. For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).
	Data Type: ViSession Input/Output: IN
<i>wavelengthArray</i>	Wavelength Array returns the wavelength values for the Lambda Scan operation. The Get Lambda Scan Result function (hp816x_getLambdaScanResult) returns the corresponding power value arrays.
	Data Type: ViReal64[] Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "[hp816x_error_message](#)"

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_getLambdaScanResult

Syntax:

```
ViStatus _VI_FUNC hp816x_getLambdaScanResult(ViSession ihandle, ViInt32
PWMChannel, ViBoolean clipToLimit, ViReal64 clippingLimit,
ViReal64 powerArray[ ], ViReal64 lambdaArray[ ]);
```

The Get Lambda Scan Result function (hp816x_getLambdaScanResult) returns for a given Power Meter channel a power value array and a wavelength value array.

These arrays contains the results of the last Multi Frame Lambda Scan operation.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized. For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).
	Data Type: ViSession Input/Output: IN

PWMChannel

Use the PWM Channel input to choose the Power Meter channel for which the power value array should be returned.

The maximum number of channels that may be specified is 1000.

Data Type: ViInt32
Input/Output: IN

clipToLimit

The Clip to Limit control turns the clipping function

On (VI_TRUE) or Off (VI_FALSE).

If the control is turned on any power value less than the Clipping Limit value will be set to this Clipping Limit value.

Data Type: ViBoolean

Input/Output: IN

Values:

Name	Value	Description
VI_TRUE	(1)	On
VI_FALSE	(0)	Off

clippingLimit

The Clipping Limit value is the value to which a power measurement will be clipped to if a value is less than the Clipping Limit.

The Clipping Limit will be used if you turn On the Clip to Limit control.

Data Type: ViReal64

Input/Output: IN

powerArray

Power Array returns for the chosen Power Meter channel the results of the last Multi Frame Lambda Scan operation.

Data Type: ViReal64[]

Input/Output: OUT

lambdaArray

Lambda Array returns for the logged wavelength points of a channel.

Data Type: ViReal64[]

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "[hp816x_error_message](#)"

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_getNoOfRegPWMChannels_Q

Syntax:

```
ViStatus _VI_FUNC hp816x_getNoOfRegPWMChannels_Q(ViSession ihandle,
ViUInt32 numberofPWMChannels);
```

The Get Number of PWM Channels (hp816x_getNoOfRegPWMChannels_Q) function returns the number of Power Meter channels in a test setup.

Only Power Meters whose mainframe was registered using the Register Mainframe ([hp816x_registerMainframe](#)) function are counted.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized.

For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function ([hp816x_init](#)).

Data Type: ViSession

Input/Output: IN

numberofPWMChannels

Get Number of PWM Channels returns the number of available Power Meter channels.

The channels are numbered from 0 to n, where n is the number of available channels minus 1.

Data Type: ViUInt32

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine

["hp816x_error_message"](#)

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_getChannelLocation

Syntax:

```
ViStatus _VI_FUNC hp816x_getChannelLocation(ViSession ihandle, ViInt32
PWMChannel, ViPInt32 mainframeNumber, ViPInt32 slotNumber, ViPInt32
channelNumber);
```

The Get Channel Location function (hp816x_getChannelLocation_Q) returns the location of the chosen Power Meter channel as used in a Multi Frame Lambda Scan operation.

The maximum number of channels that may be specified is 1000.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized. For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).
	Data Type: ViSession Input/Output: IN
<i>PWMChannel</i>	Use the PWM Channels input to select the Power Meter channel location. The maximum input may be 999. Remember that Power Meter channels are numbered from 0 to 999.
	Data Type: ViInt32 Input/Output: IN
<i>mainframeNumber</i>	

Mainframe Number returns the mainframe number for the requested channel.

This number corresponds to the order in which you registered mainframes using the Register Mainframe function ([hp816x_registerMainframe](#)).

Data Type: ViPInt32

Input/Output: OUT

slotNumber

Slot Number returns the slot in which the chosen Power Meter is located.

Data Type: ViPInt32

Input/Output: OUT

channelNumber

Channel Number returns the channel number of the chosen Power Meter channel.

For a dual Power Meter the number of the channel (1 or 2) is returned.

For single Power Meter the returned value is always 1.

Data Type: ViPInt32

Input/Output: OUT

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "[hp816x_error_message](#)"

■

hp816x_excludeChannel

Syntax:

```
ViStatus _VI_FUNC hp816x_excludeChannel(ViSession ihandle, ViInt32
PWMChannel);
```

The Exclude Channel function (hp816x_excludeChannel) excludes a specified PWM channel from a Lambda Scan operation. This function should be called before hp816x_prepareMFLambdaScan, so the settings are not limited by excluded channels.

The original indexing of the power meter channels, resulting from the hp816x_registerMainframe calls, is not changed by the Exclude Channel function. The hp816x_getLambdaScanResult command should be used with the same values for *PWMChannel* for the included channels as if no channels were excluded. In addition, all registered channels up to the last included channel, including those excluded, should be considered for setting *PWMChannels* in 816x_prepareMFLambdaScan.

Example: if 8 power meters are registered, they receive the indices 0 to 7. If the first two are excluded, then the Lambda Scan Result should be read for *PWMChannel* 2 to 7. And the *PWMChannels* parameter in hp816x_prepareMFLambdaScan should be set to 8.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized. For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).

Data Type: ViSession

Input/Output: IN

PWMChannel

Use the PWM Channels control to choose the Power Meter channel to be excluded from the Lambda Scan operation.

The maximum input may be 999.

Remember that Power Meter channels are numbered from 0 to 999.

Data Type: ViInt32

Input/Output: IN

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "[hp816x_error_message](#)"

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_setInitialRangeParams

Syntax:

```
ViStatus _VI_FUNC hp816x_setInitialRangeParams(ViSession ihandle, ViInt32
PWMChannel, ViBoolean resettoDefault, ViReal64 initialRange, ViReal64
rangeDecrement);
```

Use the Set Initial Range Parameters function (hp816x_setInitialRangeParams) to set the Initial Range, and the Range Decrement, for a selected PWM Channel.

Call this function after Prepare Multi Frame Lambda Scan, and before Execute Multi Frame Lambda Scan.

If this function is not called default values are used.

The default value for the Initial Range used in a lambda scan depends on the initial power.

The default value for the Range Decrement in subsequent scans is 20.0 dBm

Parameter	Description
<i>ihandle</i>	<p>IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized.</p> <p>For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).</p>
	<p>Data Type: ViSession Input/Output: IN</p>

PWMChannel

Use the PWM Channel control to select the Power Meter channel to which the intial range parameters will be applied.

The maximum input may be 999.

Remember that Power Meter channels are numbered from 0 to 999.

Data Type: ViInt32

Input/Output: IN

resettoDefault

Use the Reset to Default control to reset the initial range parameters to the built-in default calculated values.

If you want to reset the initial range parameters to the default values, select "Yes".

If you want to use different values, select "NO".

Data Type: ViBoolean

Input/Output: IN

Values:

Name	Value	Description
VI_TRUE	(1)	Yes
VI_FALSE	(0)	No

initialRange

Use Initial Range control to specify the range for the initial scan.

Data Type: ViReal64

Input/Output: IN

rangeDecrement

Use the Range Decrement control to specify a range decrement for subsequent scans.

Data Type: ViReal64

Input/Output: IN

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "["hp816x_error_message"](#)"

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



hp816x_setScanAttenuation

Syntax:

```
ViStatus _VI_FUNC hp816x_setScanAttenuation(ViSession ihandle, ViReal64
Scan_Attenuation);
```

Use the Set Sweep Speed function (hp816x_setScanAttenuation) to specify the attenuation applied by a backloadable TLS with a built-in optical attenuator during a multi frame lambda scan operation.

<u>Parameter</u>	<u>Description</u>
<i>ihandle</i>	IHandle is the session handle returned from the Initialize Mainframe function (hp816x_init). A new session handle is created every time the mainframe is initialized. For LabView Programmers: This input terminal needs to be connected to the session handle, IHandle, that is returned from the Initialize Mainframe function (hp816x_init).
	Data Type: ViSession Input/Output: IN
<i>Scan_Attenuation</i>	Use this control to specify the attenuation applied in decibels (dB). Data Type: ViReal64 Input/Output: IN

Return Value:

VI_SUCCESS: No error

Non VI_SUCCESS: Indicates error condition. To determine error message, pass the return value to routine "[hp816x_error_message](#)"



[Alphabetical Reference](#)[Hierarchical Reference](#)

Function Reference (Alphabetical)

[hp816x_ATT_displayToOffset](#)
[hp816x_PWM_displayToReference](#)
[hp816x_PWM_fetchValue](#)
[hp816x_PWM_functionStop](#)
[hp816x_PWM_ignoreError](#)
[hp816x_PWM_readAll](#)
[hp816x_PWM_readValue](#)
[hp816x_PWM_slaveChannelCheck](#)
[hp816x_PWM_zeroing](#)
[hp816x_PWM_zeroingAll](#)
[hp816x_RLM_fetchReturnLoss](#)
[hp816x_RLM_fetchValue](#)
[hp816x_RLM_functionStop](#)
[hp816x_RLM_readReturnLoss](#)
[hp816x_RLM_readValue](#)
[hp816x_RLM_zeroing](#)
[hp816x_RLM_zeroingAll](#)
[hp816x_TLS_configureBNC](#)
[hp816x_TLS_sweepControl](#)
[hp816x_TLS_sweepNextStep](#)
[hp816x_TLS_sweepPreviousStep](#)
[hp816x_TLS_sweepWait](#)
[hp816x_TLS_zeroing](#)
[hp816x_TLS_zeroingAll](#)
[hp816x_TLS_zeroing_Ex](#)
[hp816x_WaitForOPC](#)
[hp816x_calculate_RL](#)
[hp816x_calibrate_RLM](#)
[hp816x_close](#)
[hp816x_cls](#)
[hp816x_cmd](#)
[hp816x_cmdInt](#)
[hp816x_cmdInt32_Q](#)
[hp816x_cmdReal](#)
[hp816x_cmdReal64_Q](#)
[hp816x_cmdString_Q](#)
[hp816x_convertQuestionableStatus_Q](#)
[hp816x_cp_ATT_refFromExtPM](#)

[hp816x_dbmToWatt](#)
[hp816x_dcl](#)
[hp816x_delete_ATT_offsetTblEntries](#)
[hp816x_displayToLambdaZero](#)
[hp816x_driverLogg](#)
[hp816x_enableDisableDisplay](#)
[hp816x_enableHighSweepSpeed](#)
[hp816x_enable_RLM_sweep](#)
[hp816x_errorQueryDetect](#)
[hp816x_error_message](#)
[hp816x_error_query](#)
[hp816x_excludeChannel](#)
[hp816x_executeLambdaScan](#)
[hp816x_executeMfLambdaScan](#)
[hp816x_fetch_ATT_outputPower](#)
[hp816x_forceTransaction](#)
[hp816x_generateTrigger](#)
[hp816x_getChannelLocation](#)
[hp816x_getDate](#)
[hp816x_getHandle](#)
[hp816x_getInstrumentId_Q](#)
[hp816x_getLambdaScanParameters_Q](#)
[hp816x_getLambdaScanResult](#)
[hp816x_getLockState](#)
[hp816x_getMFLambdaScanParameters_Q](#)
[hp816x_getModuleStatus_Q](#)
[hp816x_getNoOfRegPWMChannels_Q](#)
[hp816x_getSlotInformation_Q](#)
[hp816x_getTime](#)
[hp816x_getWIRespTblSize](#)
[hp816x_getWIRespTblSizeEx](#)
[hp816x_get_ATT_absPowerMode_Q](#)
[hp816x_get_ATT_attenuationOffset_Q](#)
[hp816x_get_ATT_attenuation_Q](#)
[hp816x_get_ATT_attenuatorSpeed_Q](#)
[hp816x_get_ATT_avtime_Q](#)
[hp816x_get_ATT_controlLoopState_Q](#)
[hp816x_get_ATT_offsetFromWavelength_Q](#)
[hp816x_get_ATT_offsetTable_Q](#)
[hp816x_get_ATT_offsetTblSize_Q](#)
[hp816x_get_ATT_operStatus](#)
[hp816x_get_ATT_powerOffset_Q](#)
[hp816x_get_ATT_powerReference_Q](#)
[hp816x_get_ATT_powerUnit_Q](#)

[hp816x_get_ATT_power_Q](#)
[hp816x_get_ATT_shutterStateAtPowerOn_Q](#)
[hp816x_get_ATT_shutterState_Q](#)
[hp816x_get_ATT_triggerConfig_Q](#)
[hp816x_get_ATT_wavelengthOffsetIndex_Q](#)
[hp816x_get_ATT_wavelengthOffsetState_Q](#)
[hp816x_get_ATT_wavelength_Q](#)
[hp816x_get_ATT_wlOffsRefPowermeter_Q](#)
[hp816x_get_ATT_zeroResult_Q](#)
[hp816x_get_FLS_attenuation_Q](#)
[hp816x_get_FLS_laserSource_Q](#)
[hp816x_get_FLS_laserState_Q](#)
[hp816x_get_FLS_modulationSettings_Q](#)
[hp816x_get_FLS_parameters_Q](#)
[hp816x_get_FLS_power](#)
[hp816x_get_FLS_triggerState_Q](#)
[hp816x_get_FLS_wavelength_Q](#)
[hp816x_get_PWM_averagingTime_Q](#)
[hp816x_get_PWM_calibration_Q](#)
[hp816x_get_PWM_loggingResults_Q](#)
[hp816x_get_PWM_minMaxResults_Q](#)
[hp816x_get_PWM_parameters_Q](#)
[hp816x_get_PWM_powerRange_Q](#)
[hp816x_get_PWM_powerUnit_Q](#)
[hp816x_get_PWM_referenceSource](#)
[hp816x_get_PWM_referenceValue_Q](#)
[hp816x_get_PWM_stabilityResults_Q](#)
[hp816x_get_PWM_triggerConfiguration](#)
[hp816x_get_PWM_wavelength_Q](#)
[hp816x_get_RLM_FPDelta_Q](#)
[hp816x_get_RLM_averagingTime_Q](#)
[hp816x_get_RLM_dutValues_Q](#)
[hp816x_get_RLM_laserSourceParameters_Q](#)
[hp816x_get_RLM_laserState_Q](#)
[hp816x_get_RLM_loggingResults_Q](#)
[hp816x_get_RLM_minMaxResults_Q](#)
[hp816x_get_RLM_modulationState_Q](#)
[hp816x_get_RLM_parameters_Q](#)
[hp816x_get_RLM_powerRange_Q](#)
[hp816x_get_RLM_reflectanceValues_Q](#)
[hp816x_get_RLM_rIReference_Q](#)
[hp816x_get_RLM_srcWavelength_Q](#)
[hp816x_get_RLM_stabilityResults_Q](#)
[hp816x_get_RLM_terminationValues_Q](#)

[hp816x_get_RLM_wavelength_Q](#)
[hp816x_get_SWT_route](#)
[hp816x_get_SWT_routeTable](#)
[hp816x_get_SWT_type](#)
[hp816x_get_TLS_BNC_config_Q](#)
[hp816x_get_TLS_accClass](#)
[hp816x_get_TLS_attenuationSettings_Q](#)
[hp816x_get_TLS_autoCalState](#)
[hp816x_get_TLS_ccLevel_Q](#)
[hp816x_get_TLS_darkState_Q](#)
[hp816x_get_TLS_frequencyOffset_Q](#)
[hp816x_get_TLS_lambdaLoggingState_Q](#)
[hp816x_get_TLS_lambdaLoggingStateEx_Q](#)
[hp816x_get_TLS_lambdaZero_Q](#)
[hp816x_get_TLS_laserRiseTime](#)
[hp816x_get_TLS_laserState_Q](#)
[hp816x_get_TLS_modulationSettings_Q](#)
[hp816x_get_TLS_opticalOutput_Q](#)
[hp816x_get_TLS_parameters_Q](#)
[hp816x_get_TLS_powerData_Q](#)
[hp816x_get_TLS_powerMaxInRange_Q](#)
[hp816x_get_TLS_powerPoints_Q](#)
[hp816x_get_TLS_power_Q](#)
[hp816x_get_TLS_SBS_control_Q](#)
[hp816x_get_TLS_SBSLevel_Q](#)
[hp816x_get_TLS_sweepState_Q](#)
[hp816x_get_TLS_temperatures](#)
[hp816x_get_TLS_temperaturesEx](#)
[hp816x_get_TLS_triggerConfiguration](#)
[hp816x_get_TLS_wavelengthData_Q](#)
[hp816x_get_TLS_wavelengthDataEx_Q](#)
[hp816x_get_TLS_wavelengthPoints_Q](#)
[hp816x_get_TLS_wavelengthPointsEx_Q](#)
[hp816x_get_TLS_wavelength_Q](#)
[hp816x_init](#)
[hp816x_listVisa_Q](#)
[hp816x_lockUnlockInstrument](#)
[hp816x_mainframeSelftest](#)
[hp816x_moduleSelftest](#)
[hp816x_nodeInputConfiguration](#)
[hp816x_opc_Q](#)
[hp816x_prepareLambdaScan](#)
[hp816x_prepareMfLambdaScan](#)
[hp816x_preset](#)

[hp816x_readWIRepTblCSV](#)
[hp816x_readWIRepTblCSV_Ex](#)
[hp816x_readWIRespTable](#)
[hp816x_readWIRespTableEx](#)
[hp816x_read_ATT_outputPower](#)
[hp816x_registerMainframe](#)
[hp816x_reset](#)
[hp816x_returnEquidistantData](#)
[hp816x_revision_Q](#)
[hp816x_revision_query](#)
[hp816x_self_test](#)
[hp816x_setBaudrate](#)
[hp816x_setDate](#)
[hp816x_setInitialRangeParams](#)
[hp816x_setScanAttenuation](#)
[hp816x_setSweepSpeed](#)
[hp816x_setTime](#)
[hp816x_set_ATT_absPowerMode](#)
[hp816x_set_ATT_attenuation](#)
[hp816x_set_ATT_attenuationOffset](#)
[hp816x_set_ATT_attenuatorSpeed](#)
[hp816x_set_ATT_avTime](#)
[hp816x_set_ATT_controlLoopState](#)
[hp816x_set_ATT_offsByRefPM](#)
[hp816x_set_ATT_power](#)
[hp816x_set_ATT_powerOffset](#)
[hp816x_set_ATT_powerReference](#)
[hp816x_set_ATT_powerUnit](#)
[hp816x_set_ATT_pwrOffsRefPM](#)
[hp816x_set_ATT_shutterAtPowerOn](#)
[hp816x_set_ATT_shutterState](#)
[hp816x_set_ATT_triggerConfig](#)
[hp816x_set_ATT_wavelength](#)
[hp816x_set_ATT_wavelengthOffset](#)
[hp816x_set_ATT_wavelengthOffsetState](#)
[hp816x_set_ATT_wlOffsRefPowermeter](#)
[hp816x_set_FLS_attenuation](#)
[hp816x_set_FLS_laserSource](#)
[hp816x_set_FLS_laserState](#)
[hp816x_set_FLS_modulation](#)
[hp816x_set_FLS_parameters](#)
[hp816x_set_FLS_triggerState](#)
[hp816x_set_LambdaScan_wavelength](#)
[hp816x_set_PWM_averagingTime](#)

[hp816x_set_PWM_calibration](#)
[hp816x_set_PWM_internalTrigger](#)
[hp816x_set_PWM_logging](#)
[hp816x_set_PWM_minMax](#)
[hp816x_set_PWM_parameters](#)
[hp816x_set_PWM_powerRange](#)
[hp816x_set_PWM_powerUnit](#)
[hp816x_set_PWM_referenceSource](#)
[hp816x_set_PWM_referenceValue](#)
[hp816x_set_PWM_stability](#)
[hp816x_set_PWM_triggerConfiguration](#)
[hp816x_set_PWM_wavelength](#)
[hp816x_set_RLM_FPDelta](#)
[hp816x_set_RLM_averagingTime](#)
[hp816x_set_RLM_internalTrigger](#)
[hp816x_set_RLM_laserSourceParameters](#)
[hp816x_set_RLM_laserState](#)
[hp816x_set_RLM_logging](#)
[hp816x_set_RLM_minMax](#)
[hp816x_set_RLM_modulationState](#)
[hp816x_set_RLM_parameters](#)
[hp816x_set_RLM_powerRange](#)
[hp816x_set_RLM_rIReference](#)
[hp816x_set_RLM_stability](#)
[hp816x_set_RLM_triggerConfiguration](#)
[hp816x_set_RLM_wavelength](#)
[hp816x_set_SWT_route](#)
[hp816x_set_TLS_attenuation](#)
[hp816x_set_TLS_autoCalibration](#)
[hp816x_set_TLS_ccLevel](#)
[hp816x_set_TLS_dark](#)
[hp816x_set_TLS_frequencyOffset](#)
[hp816x_set_TLS_lambdaLoggingState](#)
[hp816x_set_TLS_lambdaLoggingStateEx](#)
[hp816x_set_TLS_laserRiseTime](#)
[hp816x_set_TLS_laserState](#)
[hp816x_set_TLS_modulation](#)
[hp816x_set_TLS_opticalOutput](#)
[hp816x_set_TLS_parameters](#)
[hp816x_set_TLS_power](#)
[hp816x_set_TLS_SBS_Control](#)
[hp816x_set_TLS_SBSLevel](#)
[hp816x_set_TLS_sweep](#)
[hp816x_set_TLS_triggerConfiguration](#)

[hp816x_set_TLS_wavelength](#)
[hp816x_spectralCalibration](#)
[hp816x_spectralCalibrationEx](#)
[hp816x_standardTriggerConfiguration](#)
[hp816x_standardTriggerConfiguration_Q](#)
[hp816x_start_PWM_internalTrigger](#)
[hp816x_start_RLM_internalTrigger](#)
[hp816x_timeOut](#)
[hp816x_timeOut_Q](#)
[hp816x_trigOutConfiguration](#)
[hp816x_unregisterMainframe](#)
[hp816x_wattToDBm](#)
[hp816x_zero_ATT_all](#)
[hp816x_zero_ATT_powermeter](#)

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



[Alphabetical Reference](#)[Hierarchical Reference](#)

Function Reference (Hierarchical)

The hierarchical function reference lists the hp816x VXI*plug&play* functions by classes as defined by the hp816x function panel (hp816x.fp). The function panel runs under VEE, LabWindows®, and LabVIEW®.

DRIVER: hp816x

[Initialize](#)[Get Session Handle](#)[revision query](#)[Self Test](#)[Utility](#)[Timeout](#)[Get Instrument ID](#)[List Visa Addresses](#)[Set Baudrate](#)[Query Timeout](#)[Device Clear](#)[Instrument Error Query](#)[Error Message](#)[Automatic Error Detection](#)[dBm to Watt](#)[WattToDBm](#)[Set Date](#)[Get Date](#)[Set Time](#)[Get Time](#)[Clear Error Queue](#)[Force Transaction](#)[Enable Driver Logging](#)[Pass Through Functions](#)[Send String Command](#)[Send String Query](#)[Send Integer Command](#)[Send Integer Query](#)[Send Real Command](#)[Send Real Query](#)[Mainframe Specific Functions](#)[Information for N77 Series Instruments](#)[Reset](#)

[Preset](#)

[Mainframe Self-Test](#)

[Module Self-Test](#)

[Revision Query](#)

[Operation Complete](#)

[Lock-Unlock Instrument](#)

[Get Lock State](#)

[Enable-Disable Display](#)

[Get Slot Information](#)

[Get Module Status](#)

[Convert Questionable Status](#)

[Standard Trigger Configuration](#)

[Get Standard Trigger Configuration](#)

Custom Trigger

[Trigger Input Configuration](#)

[Trigger Output Configuration](#)

[Generate Trigger](#)

Attenuators

[Information for N77 Series Instruments](#)

[Set Attenuation](#)

[Get Attenuation](#)

[Set Attenuation Offset](#)

[Get Attenuation Offset](#)

[Set Attenuation Speed](#)

[Get Attenuation Speed](#)

[Display to Offset](#)

[Set Wavelength](#)

[Get Wavelength](#)

[Set Power Unit](#)

[Get Power Unit](#)

[Set Absolute Power Mode](#)

[Get Absolute Power Mode](#)

[Set Power](#)

[Get Power](#)

[Copy Ref. from Powermeter](#)

[Set Power Reference](#)

[Get Power Reference](#)

[Set Shutter State](#)

[Get Shutter State](#)

[Set Shutter State at Power On](#)

[Get Shutter State at Power On](#)

[Get Operational Status](#)

Wavelength-Offset Table Control

[Set Wavel. Offs. Ref. PM](#)

[Get Wavel. Offs. Ref. PM](#)
[Set Wavelength Offset State](#)
[Get Wavelength Offset State](#)
[Set Wavelength Offset](#)
[Get WL/Offset from Index](#)
[Get Offset from Wavelength](#)
[Delete Table Entries](#)
[Get Offset Table Entries](#)
[Get Table Size](#)
Builtin Powermeter Control
[Read ATT Output Power](#)
[Fetch ATT Output Power](#)
[Set Power Offset](#)
[Get Power Offset](#)
[Set Power Offset Ref. PM](#)
[Set Att. Offs. by Ext PM](#)
[Set Control Loop State](#)
[Get Control Loop State](#)
[Set Averaging Time](#)
[Get Averaging Time](#)
[Set ATT Trigger Configuration](#)
[Get ATT Trigger configuration](#)
[Zero Builtin Powermeter](#)
[Get Last Zero Result](#)
[Zero All](#)
Multi Wavelength Calibration
[Spectral Calibration](#)
[Get Wavelength Resp. Tbl Size](#)
[Read Wavelength Response Table](#)
[Read WI Resp Table CSV](#)
Power Meter Modules
[Information for N77 Series Instruments](#)
[Slave Channel Check](#)
[Set PWM Parameters](#)
[Get PWM Parameters](#)
[Set PWM Averaging Time](#)
[Get PWM Averaging Time](#)
[Set PWM Wavelength](#)
[Get PWM Wavelength](#)
[Set PWM Calibration](#)
[Get PWM Calibration](#)
[Set PWM Power Range](#)
[Get PWM Power Range](#)
[Set PWM Power Unit](#)

[Get PWM Power Unit](#)
[Set PWM Reference Source](#)
[Get PWM Reference Source](#)
[Set PWM Reference Value](#)
[Get PWM Reference Value](#)
[Set PWM Trigger Configuration](#)
[Get PWM Trigger Configuration](#)
[PWM Display to Reference](#)
[Set PWM Internal Trigger](#)
[Start PWM Internal Trigger](#)
[Read PWM Value](#)
[Fetch PWM Value](#)
[Read All PWM Values](#)
[PWM Zeroing](#)
[PWM Zeroing All](#)
[Ignore Instrument Error](#)
PWM Data Acquisition
[Set PWM Logging](#)
[Get PWM Logging Results](#)
[Set PWM Stability](#)
[Get PWM Stability Results](#)
[Set PWM MinMax](#)
[Get PWM MinMax Results](#)
[Stop PWM Function](#)
[PWM Zeroing](#)
[Ignore Instrument Error](#)
Multi Wavelength Calibration
[Spectral CalibrationEx](#)
[Get Wavelength Resp. Tbl SizeEx](#)
[Read Wavelength Response TableEx](#)
[Read WI Resp Table CSV_Ex](#)
Fixed Laser Sources
[Set FLS Parameters](#)
[Get FLS Parameters](#)
[Set FLS Modulation](#)
[Get FLS Modulation Settings](#)
[Set FLS Laser Source](#)
[Get FLS Laser Source](#)
[Get FLS Wavelength](#)
[Set FLS Attenuation](#)
[Get FLS Attenuation](#)
[Set FLS Trigger State](#)
[Get FLS Trigger State](#)
[Get FLS Power](#)

[Set FLS Laser State](#)

[Get FLS Laser State](#)

Switch Modules

[Get Switch Type](#)

[Set Route](#)

[Get Route](#)

[Get Route Table](#)

Return Loss Modules

[Set RLM Parameters](#)

[Get RLM Parameters](#)

[Set RLM Internal Trigger](#)

[Set RLM Averaging Time](#)

[Get RLM Averaging Time](#)

[Set RLM Wavelength](#)

[Get RLM Wavelength](#)

[Set RLM Power Range](#)

[Get RLM Power Range](#)

[Set RLM Trigger Configuration](#)

[Start RLM Trigger](#)

[Read RLM Return Loss](#)

[Fetch RLM Return Loss](#)

[Read RLM Power Value](#)

[Fetch RLM Power Value](#)

[Set RLM Return Loss Reference](#)

[Get RLM Return Loss Reference](#)

[Set RLM Front Panel Delta](#)

[Get RLM Front Panel Delta](#)

[Calibrate RLM](#)

[RLM Zeroing](#)

[RLM Zeroing All](#)

[Enable RLM Sweep](#)

Advanced Functions

[Get RLM Reflectance Values](#)

[Get RLM Termination Values](#)

[Get RLM DUT Values](#)

[Calculate Return Loss](#)

Optional Laser Source Control

[Get RLM Source Wavelength](#)

[Set RLM Laser Source Parameters](#)

[Get RLM Laser Source Parameters](#)

[Set RLM Modulation State](#)

[Get RLM Modulation State](#)

[Set RLM Laser State](#)

[Get RLM Laser State](#)

RLM Data Acquisition

[Set RLM Logging](#)
[Get RLM Logging Results](#)
[Set RLM Stability](#)
[Get RLM Stability Results](#)
[Set RLM MinMax](#)
[Get RLM MinMax Results](#)
[Stop RLM Function](#)
[Enable RLM Sweep](#)

Tunable Laser Sources

[Information for N77 Series Instruments](#)
[Wait for OPC](#)
[Set TLS Parameters](#)
[Get TLS Parameters](#)
[Set TLS Wavelength](#)
[Get TLS Wavelength](#)
[Set TLS Power](#)
[Get TLS Power](#)
[Set TLS Optical Output](#)
[Get TLS Optical Output](#)
[Get TLS Max Power in Range](#)
[Set TLS Laser State](#)
[Get TLS Laser State](#)
[Set Laser Rise Time](#)
[Get Laser Rise Time](#)
[Set TLS Trigger Configuration](#)
[Get TLS Trigger Configuration](#)
[Set TLS Attenuation](#)
[Get TLS Attenuation Settings](#)
[Set TLS Dark](#)
[Get TLS Dark State](#)
[Get TLS Temperatures](#)
[Get TLS Temperatures Ex](#)
[TLS Zeroing](#)
[TLS ZeroingAll](#)
[TLS Zeroing Ex](#)
[Set Auto Calibration](#)
[Get Auto Calibration State](#)
[Get Accuracy Class](#)
TLS Frequency Offset
[TLS Display to Lambda Zero](#)
[Get TLS Lambda Zero](#)
[Set TLS Frequency Offset](#)
[Get TLS Frequency Offset](#)

TLS Sweep Functions

[Set TLS Sweep](#)

[TLS Sweep Control](#)

[Get TLS Sweep State](#)

[TLS Next Step](#)

[TLS Previous Step](#)

[Wait Sweep Finished](#)

TLS Modulation

[Set TLS Modulation](#)

[Get TLS Modulation Settings](#)

[TLS Configure BNC Output](#)

[TLS Get BNC Configuration](#)

[Set Coherence Control Lev \(DFB\)](#)

[Get Coherence Control Lev \(DFB\)](#)

TLS SBS Suppression

[Set TLS SBS Control](#)

[Get TLS SBS Control](#)

[Set TLS SBS Level](#)

[Get TLS SBS Level](#)

TLS Maximum Power Curve

[Get Number of Power Points](#)

[Get Logged Power Data](#)

TLS Logging Functions

[Set Lambda Logging State](#)

[Set Lambda Logging State Ex](#)

[Get Lambda Logging State](#)

[Get Lambda Logging State Ex](#)

[Get Number of Wavelength Values](#)

[Get Number of Wavelength Values Ex](#)

[Get Logged Wavelength Data](#)

[Get Logged Wavelength Data Ex](#)

Applications

[Set Lambda Scan Wavelength](#)

[Enable High Sweep Speed](#)

Lambda Scan

[Prepare Lambda Scan](#)

[Get Lambda Scan Parameters](#)

[Execute Lambda Scan](#)

Multi Frame Lambda Scan

[Information for N774x Series Instruments](#)

[Equally Spaced Datapoints](#)

[Register Mainframe](#)

[Unregister Mainframe](#)

[Set Sweep Speed](#)

[Prepare Multi Frame Lambda Scan](#)

[Get MF Lambda Scan Parameters](#)

[Execute Multi Frame Lambda Scan](#)

[Get Lambda Scan Result](#)

[Get Number of PWM Channels](#)

[Get Channel Location](#)

[Exclude Channel](#)

[Set Initial Range Parameters](#)

[Set Scan Attenuation](#)

[Close](#)

Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015



Where to find Online and User Manual Information

Online Information

<http://www.keysight.com/find/optical>

Contact information is available via the Keysight Technologies customer website:

<http://www.keysight.com/find/contactus>

User Manual Information

Using the Keysight Technologies 816x VXIplug&play Instrument Driver requires you to use both the online help and the instrument user's manual. It is assumed that VISA and a GP-IB card if required has been installed and properly configured. The following is a high-level directory of where specific information can be found:

SCPI programming information, SCPI Command Reference, and SCPI example programs	<i>Keysight 8163A/B Lightwave Multimeter, Keysight 8164A/B Lightwave Measurement System, & Keysight 8166A/B Lightwave Multichannel System Programming Guide</i> <i>Keysight N77xx Series Programming Guide</i>
SCPI example programs	<i>SCPI 1995 Volume2: Command Reference</i>
Keysight Technologies 816x VXIplug&play instrument driver programming information	<i>Online Help</i>
Keysight Technologies 816x VXIplug&play instrument driver example programs	<i>Online Help</i>
Keysight Technologies 816x VXIplug&play instrument driver function reference	<i>Online Help</i>
VISA Language Information	<i>VISA User's Manual</i>
VEE programming information	<i>VEE User's Manual</i>

▲

[Alphabetical Reference](#)

[Hierarchical Reference](#)

In Case of Trouble

A variety of support is available through the Web including troubleshooting hints, technical information and access to the most up-to-date drivers.

For general troubleshooting support and technical information see:

<http://www.keysight.com/find/optical>

Contact Keysight:

<http://www.keysight.com/find/contactus>

For access to the most up to date instrument drivers see:

<http://www.keysight.com/find/octfirmware>

© Copyright 1999-2015 Keysight Technologies. All rights reserved.

E0615, Jun 2015

