Ladybug LabVIEW Sample Code

Overview

This document introduces the user to using Ladybug power sensors in a LabVIEW environment. It assumes familiarity with LabVIEW but does not require expertise. It demonstrates key functions including:

- Sensor Initialization
- Sensor Setup (Frequency, Averages, Measurement Units)
- CW Measurements
- Pulse Measurements
- Ratio Measurements

An example test harness (with source code) is included. The user interface is shown in figure 1. It is written as a queue-based state machine and is designed to be easy to understand. It will work with one or more sensors.

Note: All LabVIEW VI's are written in LabVIEW 8.5.1.

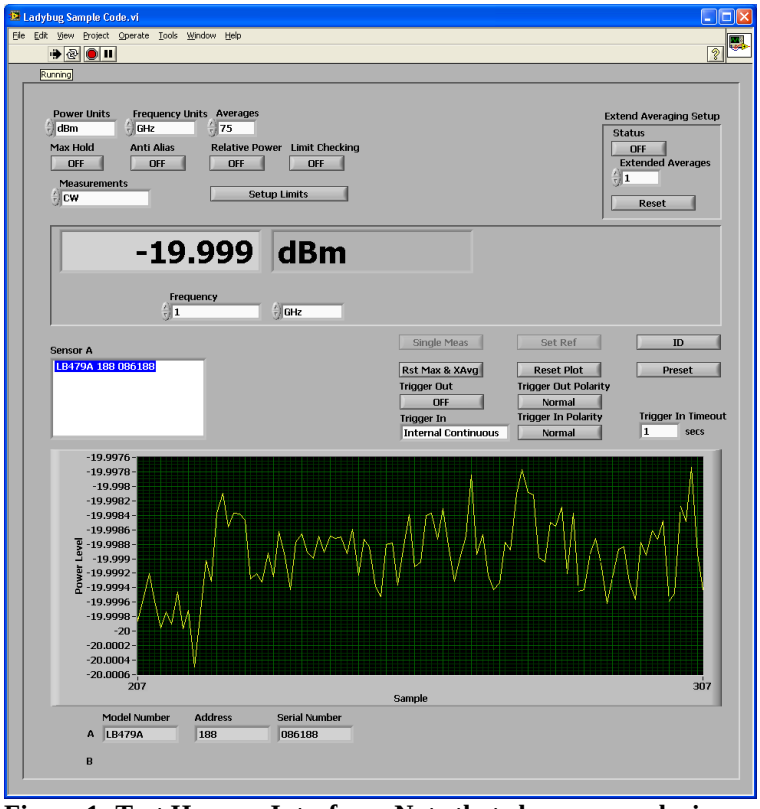


Figure 1: Test Harness Interface. Note that changes may be incorporated for cosmetic reasons or as new functionality is added. The interface may also change depending on some selections.

Exercising the Test Harness

The test harness is an interactive application designed to be similar to the front panel of more traditional power meters. Extra features, such as the plot, were added to emphasize some of the benefits of the Ladybug Sensor.

The software has many capabilities and is easy to use. A few of the settings and techniques are shown here to assist you in getting started.

Software Installation

To install the software, simply run the installer and it will copy all files to the appropriate locations.

Setting the Frequency

To set the frequency, change the value in the frequency control (shown below in yellow). You may also change the units in either place (shown in red) as necessary. These changes will take effect immediately.

Note that this will only change the frequency for the active sensor. If more than one sensor is connected, the others will not change. If you are performing ratio measurements, the settings for Sensor A can be copied to Sensor B by pressing the Mirror Sensor button.

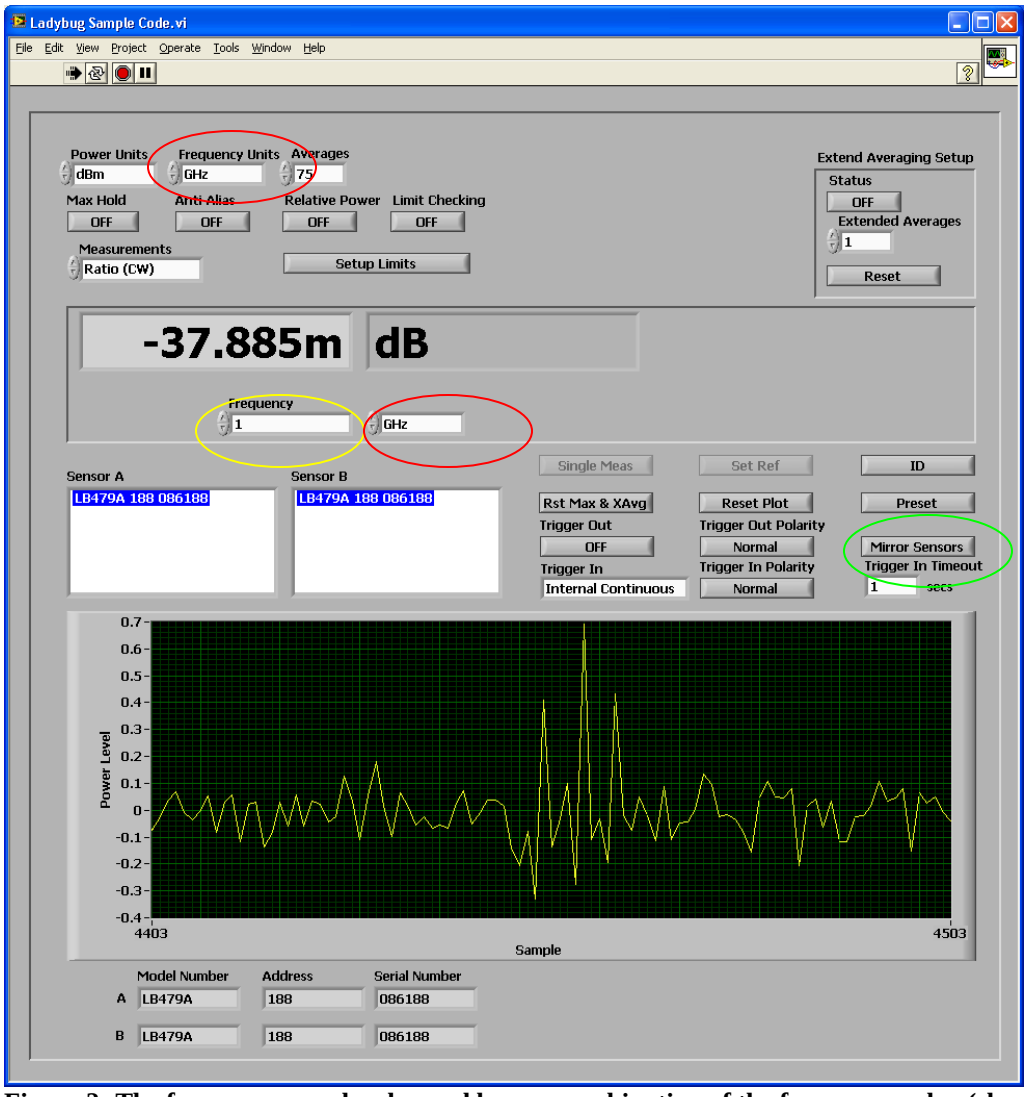


Figure 2: The frequency may be changed by any combination of the frequency value (shown in yellow) and the units (shown in red). Note that the Mirror Sensors button (shown in green) is only visible during ratio measurements.

Setting the Power Units

To set the power units, simply select the units from either of the drop-down lists (shown in yellow). Note that the larger list is not shown during ratio measurements – it will show either "dB" (if the measurement units are in any logarithmic units) or blank (for linear units such as Watts or Volts).

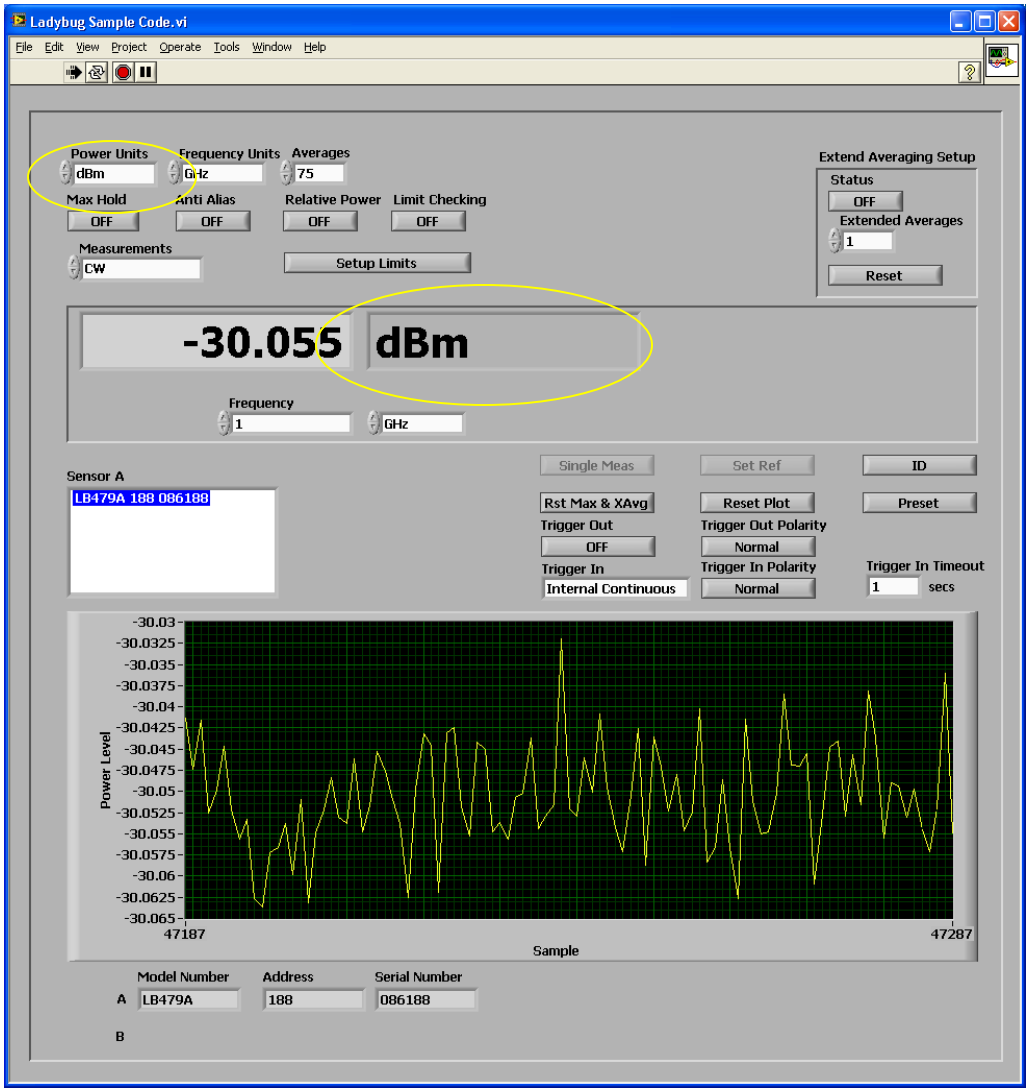


Figure 3: Power units can also be changed in multiple places, except when the measurement is either a CW or Pulse Ratio.

Making a CW Measurement

The default measurement when the system starts is CW. Generally speaking, this can be performed without any difficulty. To switch back to CW from another measurement, simply select CW from the drop-down list (shown below in yellow).

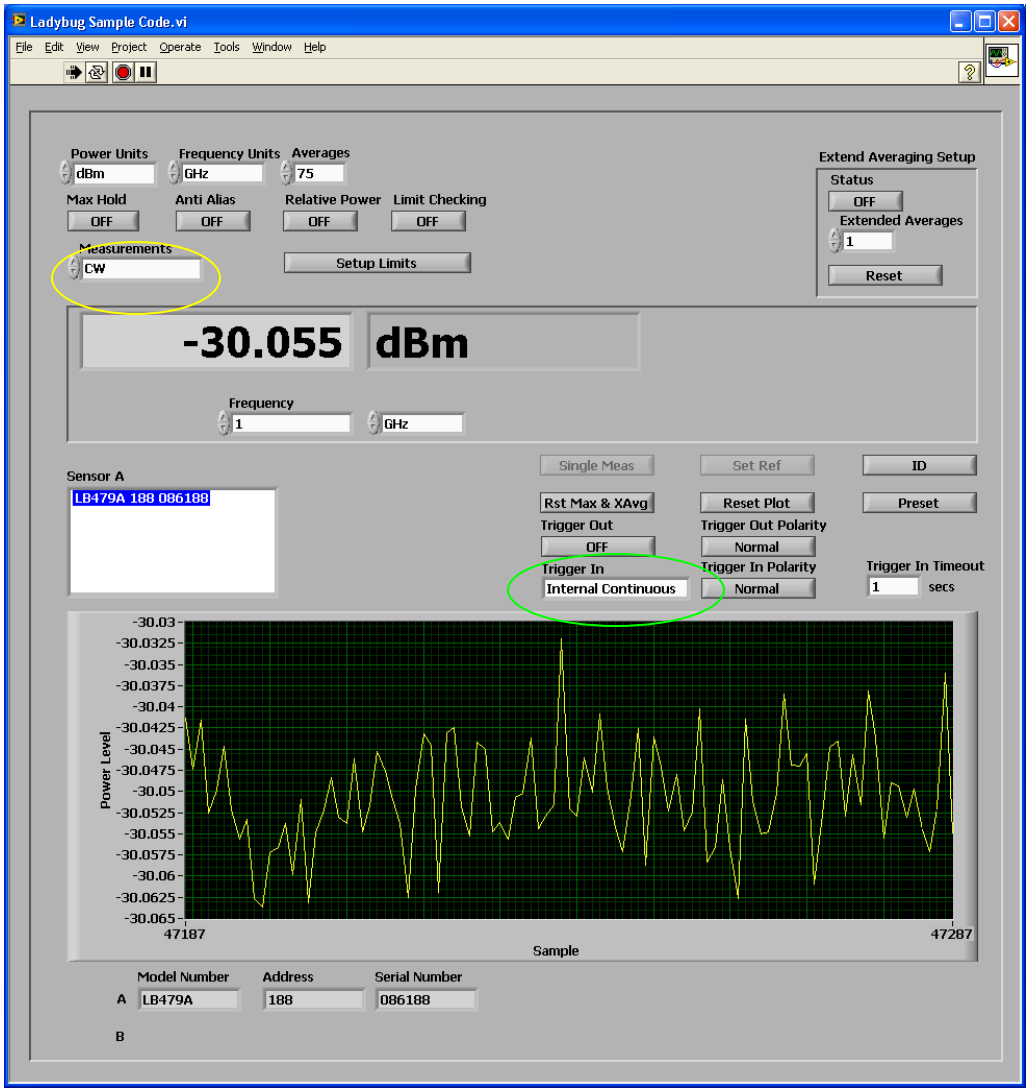


Figure 4: The default measurement type is CW. Measurements will be made continuously unless the Trigger In control (in green) is set to something other than "Internal Continuous".

Making a Pulse Measurement

Similarly, pulse measurements are made by selecting the Pulsed value from the drop-down list. Note that the additional quantities Duty Cycle, Peak, Average, and Crest Factor (all shown in yellow) appear after this selection is made. Note that 5xx series sensors cannot make pulse measurements even though this option may be selected.

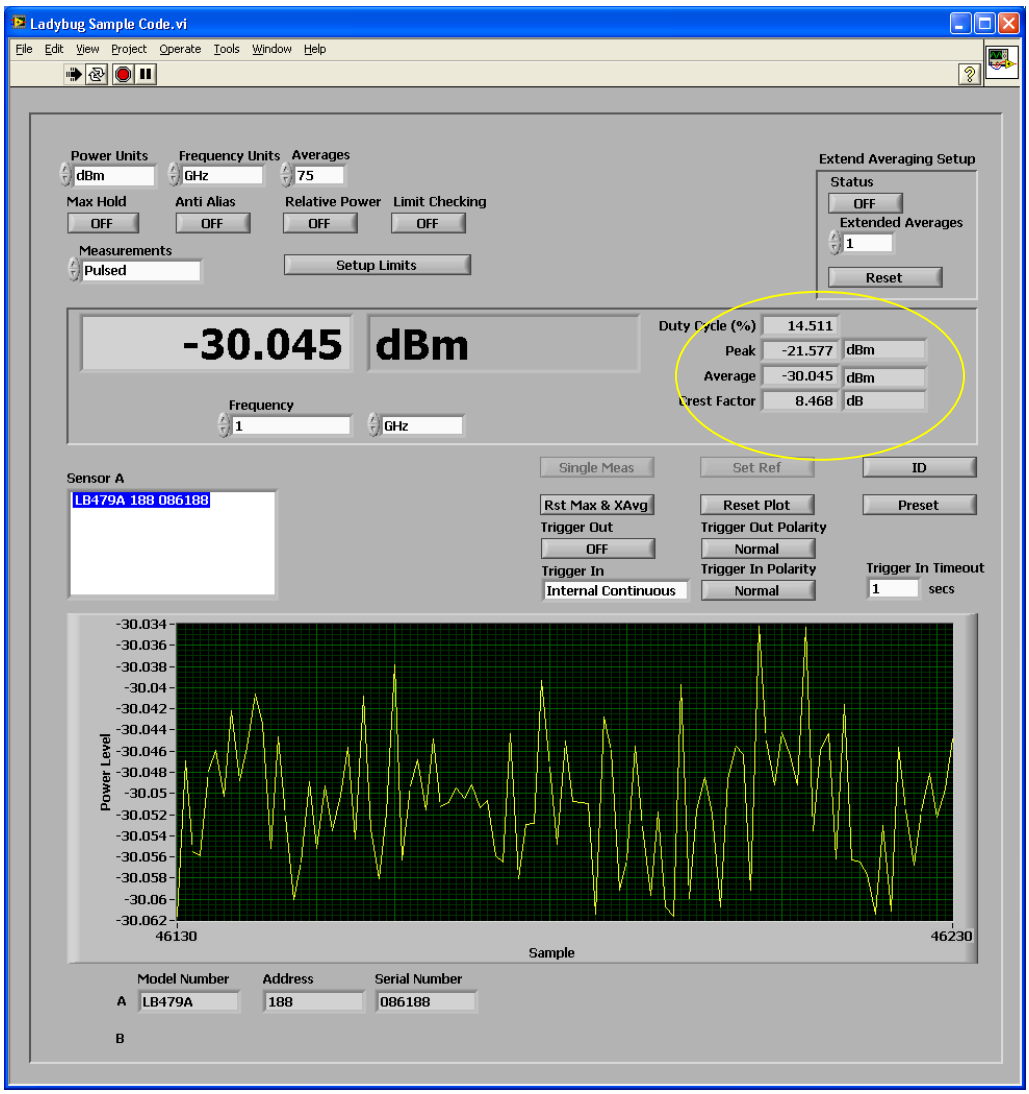
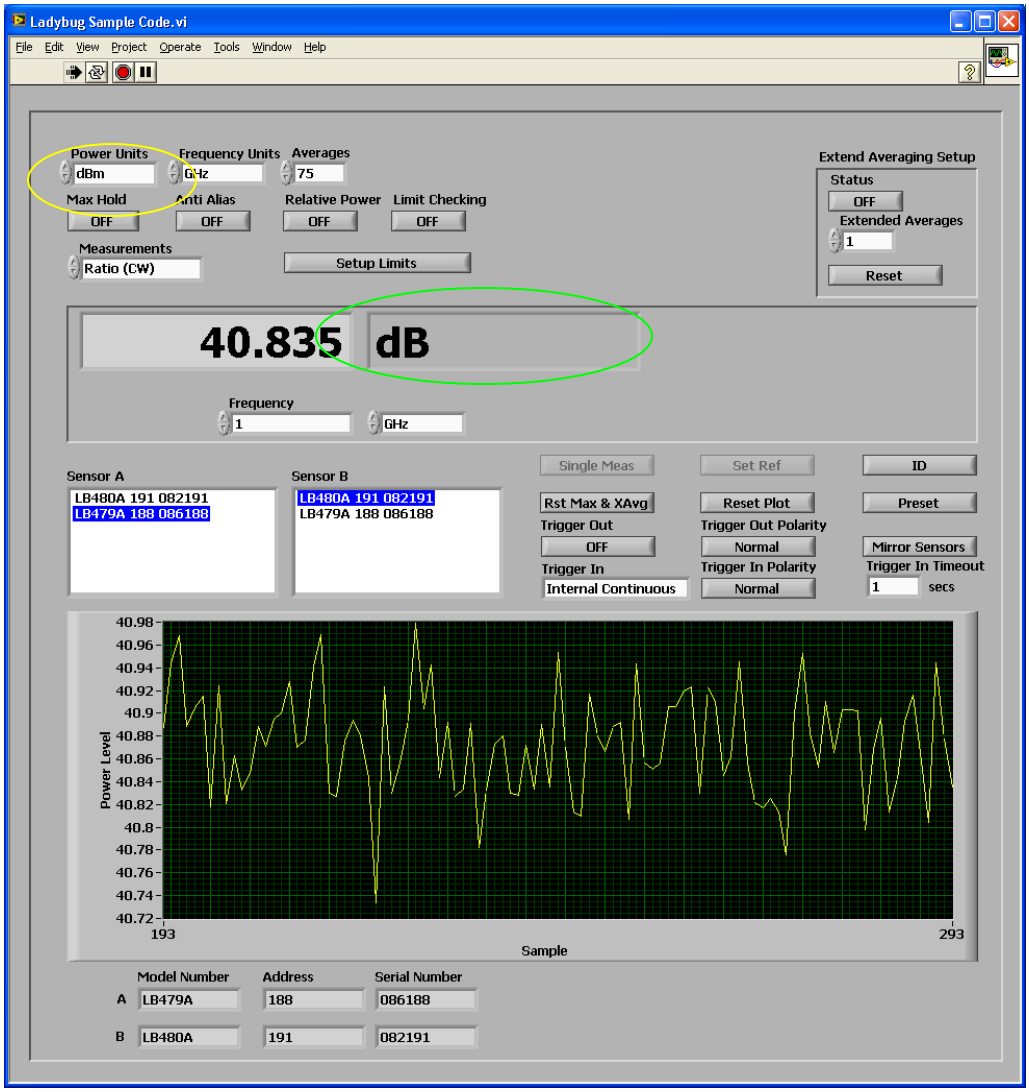


Figure 5: The additional pulse measurements are shown on the right. The units for duty cycle are alway percent and the units for crest factor are always dB. The units for the other parameters are always the same as the power units (except for pulse ratio).

Making a Ratio Measurement

Ratio measurements are made by selecting either of the Ratio options (CW or pulsed). CW is shown here. Not surprisingly, ratio measurements require two sensors for the results to make sense. If the power units (in yellow) are logarithmic, the displayed power units (in green) will be "dB". If the power units are linear (such as watts or volts), the displayed power units will be blank.



Using and Modifying the Code

Sensor Communication

Communication with LadyBug power sensors is done through the index, the serial number, or the address of the sensor. In this example code, all communication is done through the address. This is because the address covers more functionality than the others and because it makes the code more flexible. For example, the user cannot change the serial number of a sensor but they can change the address.

Unless otherwise noted, each VI may be found in the Settings folder.

Additional VI's will be added to give users full functionality as time permits.

Initializing the Sensors

Before the sensors can be used, they must be initialized. The code performs additional steps as well, but we will focus on the steps necessary for any application. You may need the additional steps depending upon your application.

The first step in this process is to obtain the sensor list. This is followed by initializing each sensor. This initialization only needs to be done once.

In the test harness code, these steps are performed as shown in the following figures.

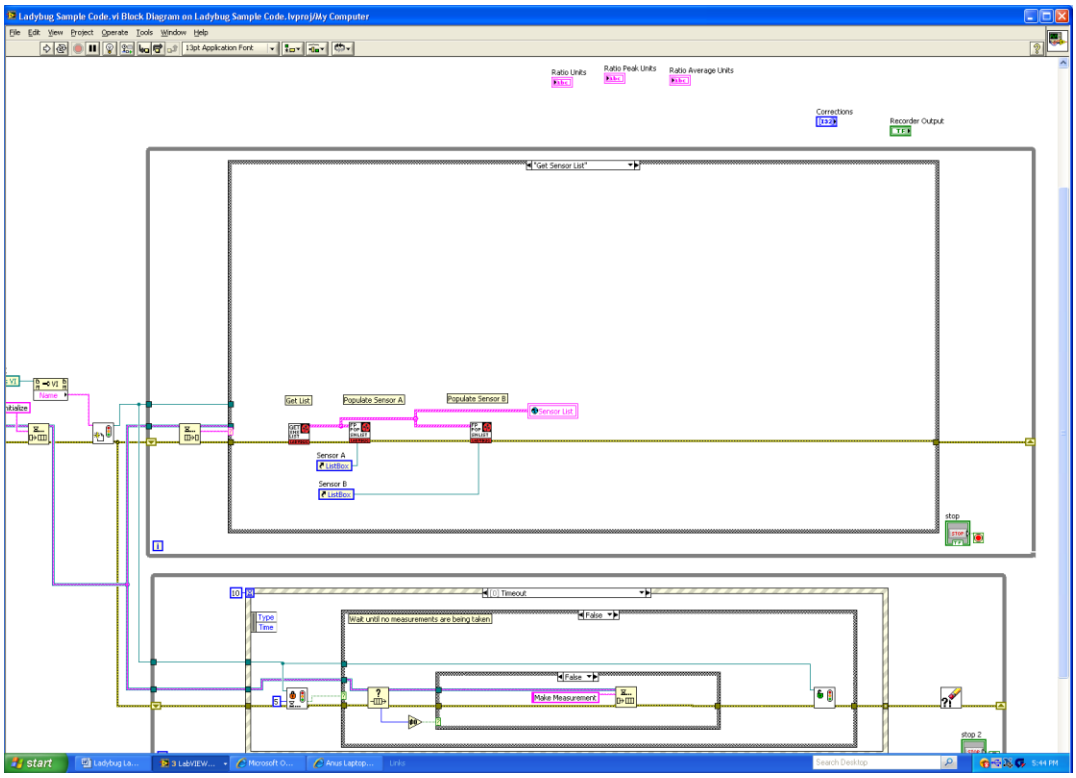


Figure 6: "Get Sensor List" in the case structure.

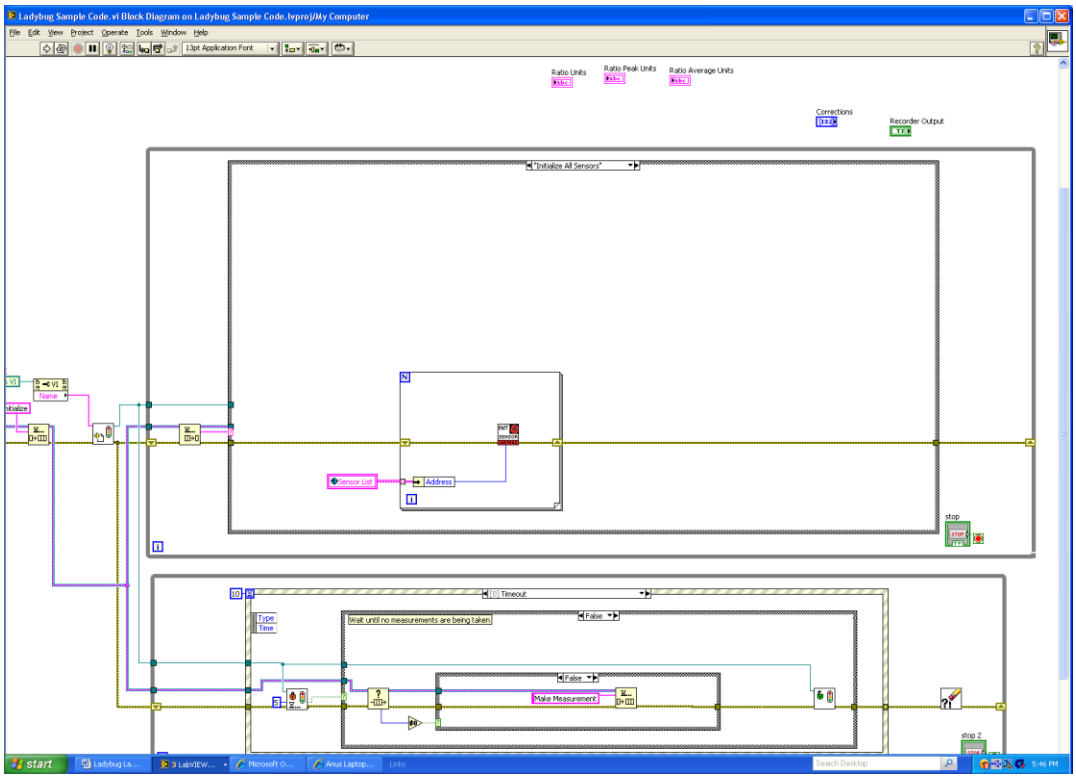


Figure 7: "Initialize All Sensors" in the case structure.

In more detail, the front panel of the Get Sensor List VI is shown below. It retrieves the relevant information from each sensor and returns it in an array of clusters.

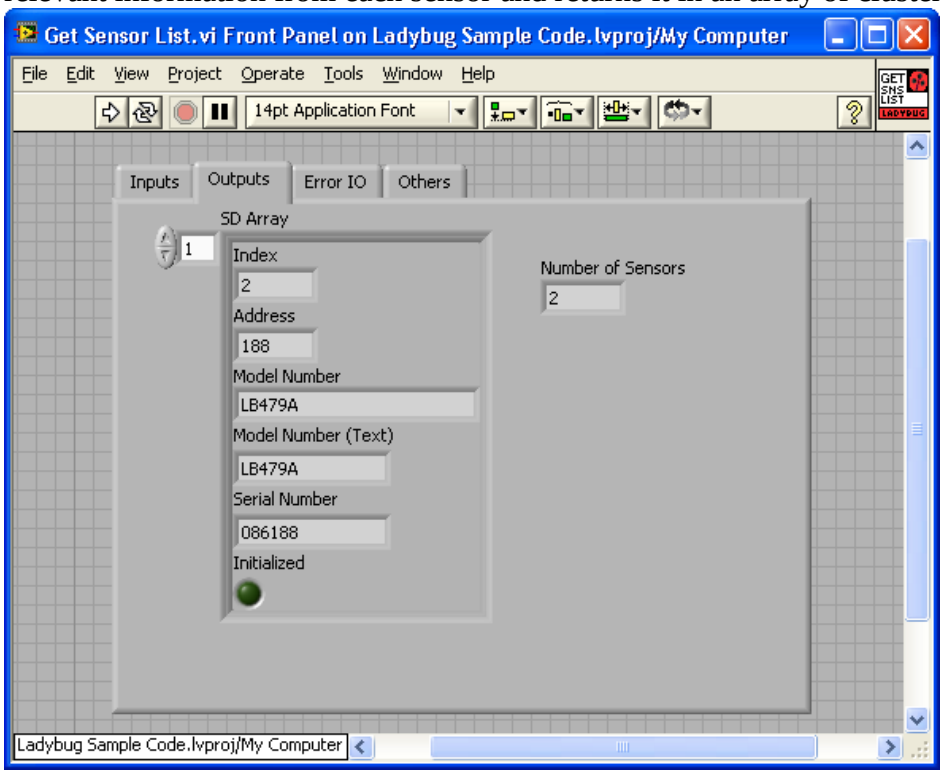


Figure 8: Get Sensor List VI front panel.

Sensors are initialized by address using the Initialize Sensor (Address) VI shown below.

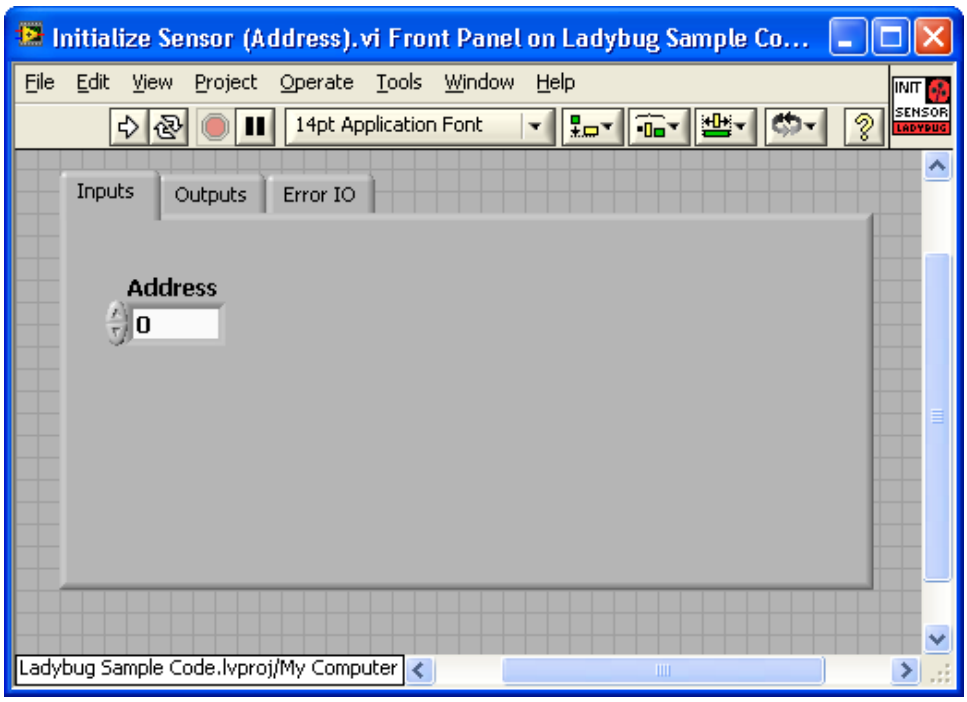


Figure 9: Initialize Sensor (Address) VI front panel.

An easy way to initialize all sensors is shown below:

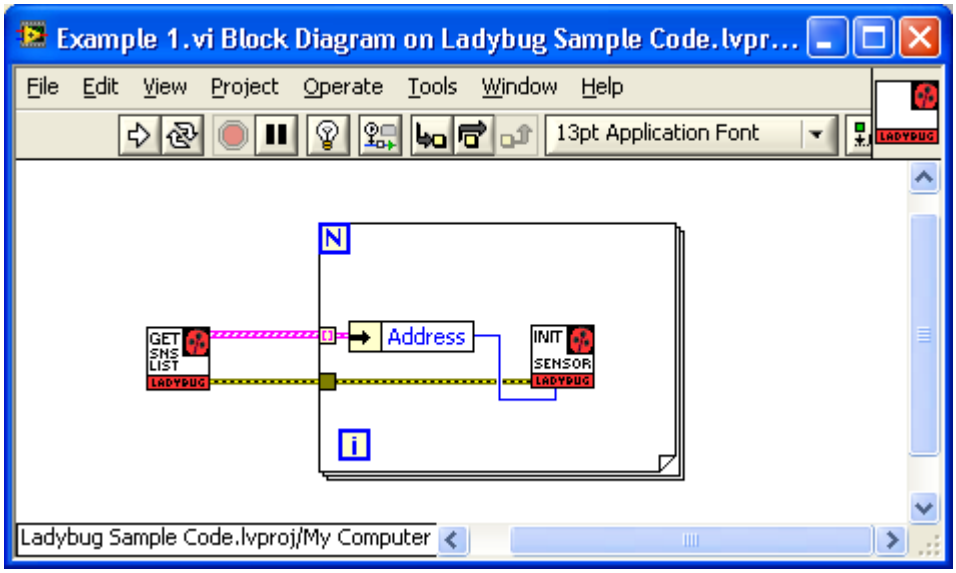


Figure 10: Easy Initialization example (Example 1.vi). This example may be found in the examples folder.

Sensor Identification

The ID VI makes the LED blink several times on the power sensor at the given address. This VI can be run on its own by simply typing a valid value for the address in the Inputs tab of the VI.

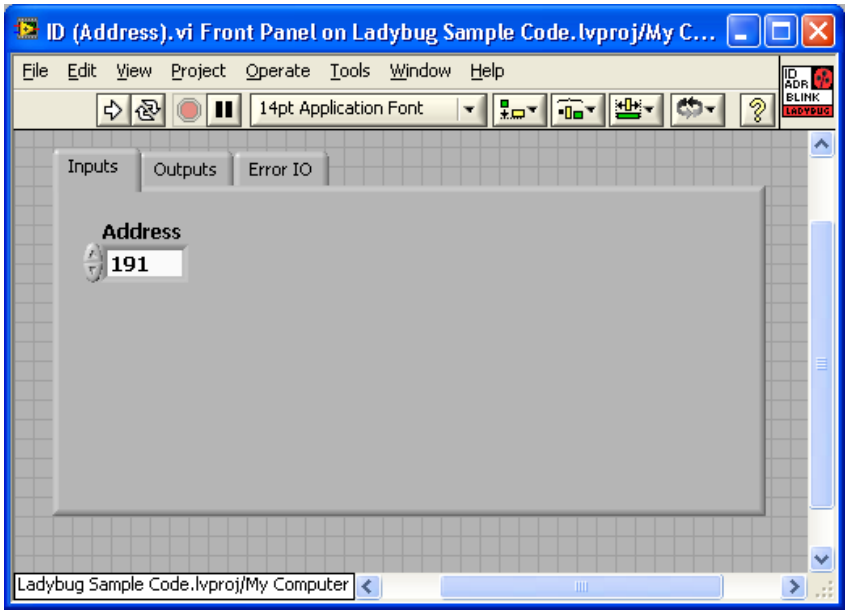


Figure 11: The ID function does not return a value. It is used to physically identify the sensor at the given address.

Setting the Frequency

To set the power sensor frequency, use the Set Frequency (Address) VI. Note that the frequency must be in Hertz. You may want to account for this before calling this VI (as was done in the test harness code).

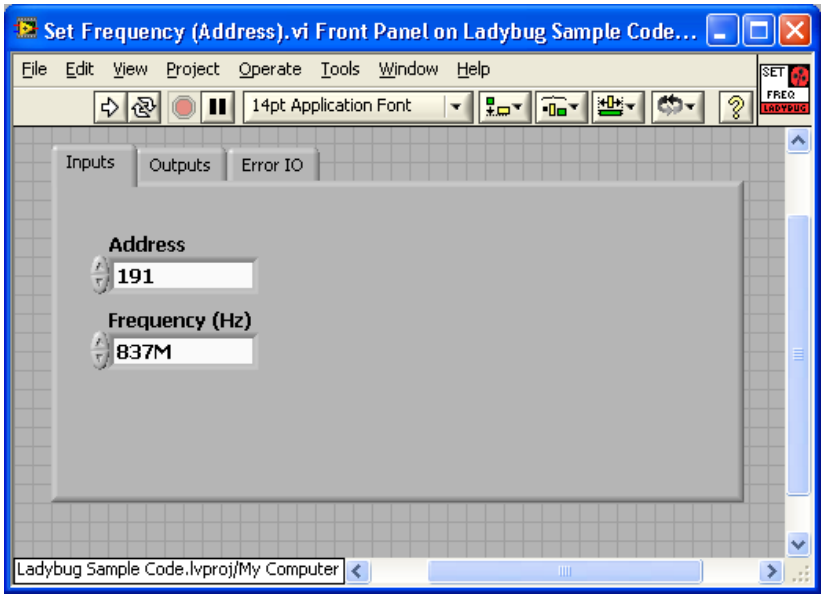


Figure 12: With these parameters, the power sensor at address 191 would be set to a frequency of 837 MHz.

Setting the Power Units

To set the power units, use the Set Power Units VI. Possible units include dBm, dBW, dBkW, dBuV, dBmV, dBV, Watts, Volts, and dB Relative.

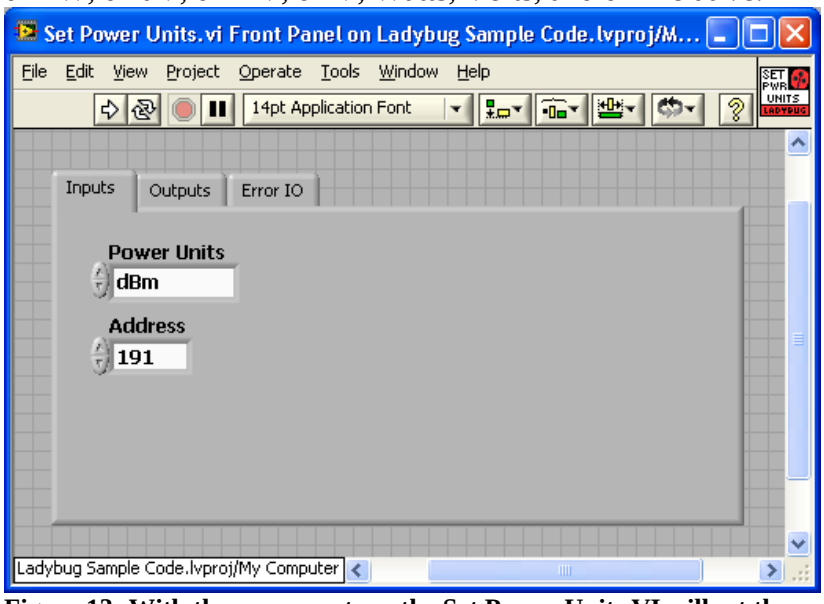


Figure 13: With these parameters, the Set Power Units VI will set the power sensor at address 191 to dBm.

Making a CW Measurement

The Get CW Measurement (Address) VI will measure the CW power for the sensor at the given address in the current units of the sensor.

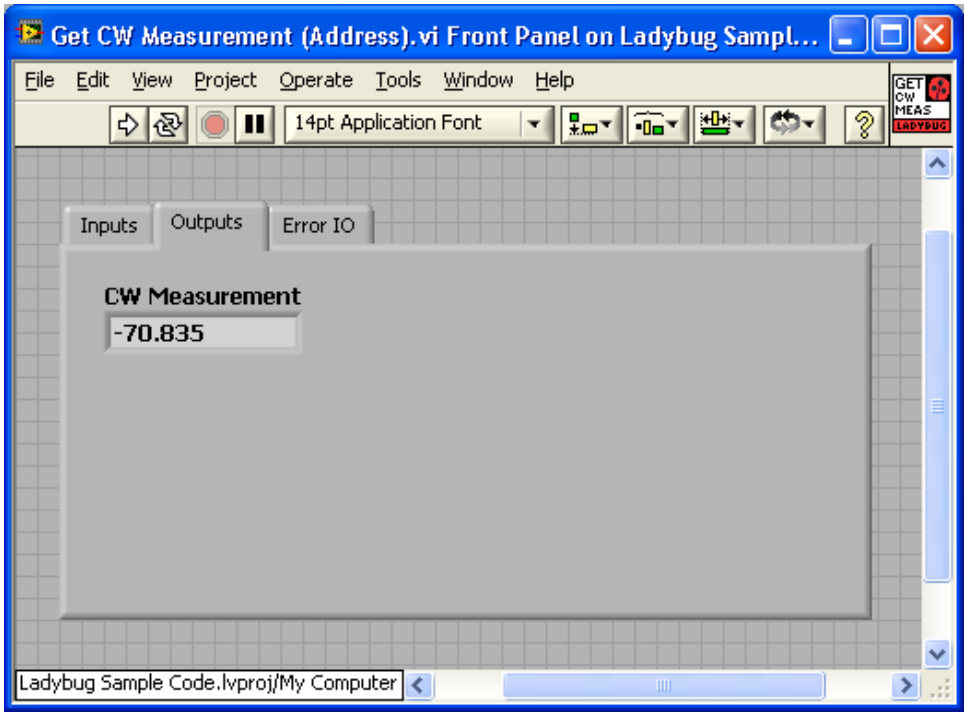


Figure 14: Using this VI, a measurement of -70.835 dBm was taken from the power sensor at address 191. The address was set on the input tab (not shown).

To make a complete measurement, see the CW Measurement Example vi in the example directory.

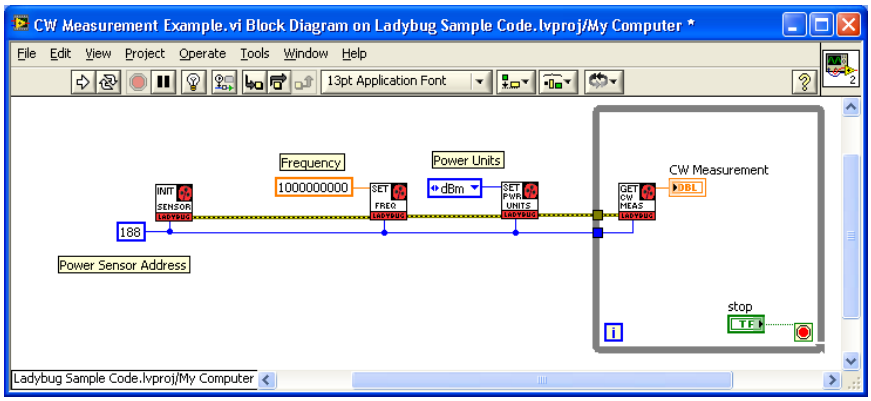


Figure 15: The CW Measurement Example shows how to take continuous CW measurements. Note that the address (188) will probably have to be modified for your needs.

Making a Pulse Measurement

The Get Pulse Measurement (Address) VI will measure the pulse power for the sensor at the given address in the current units of the sensor.

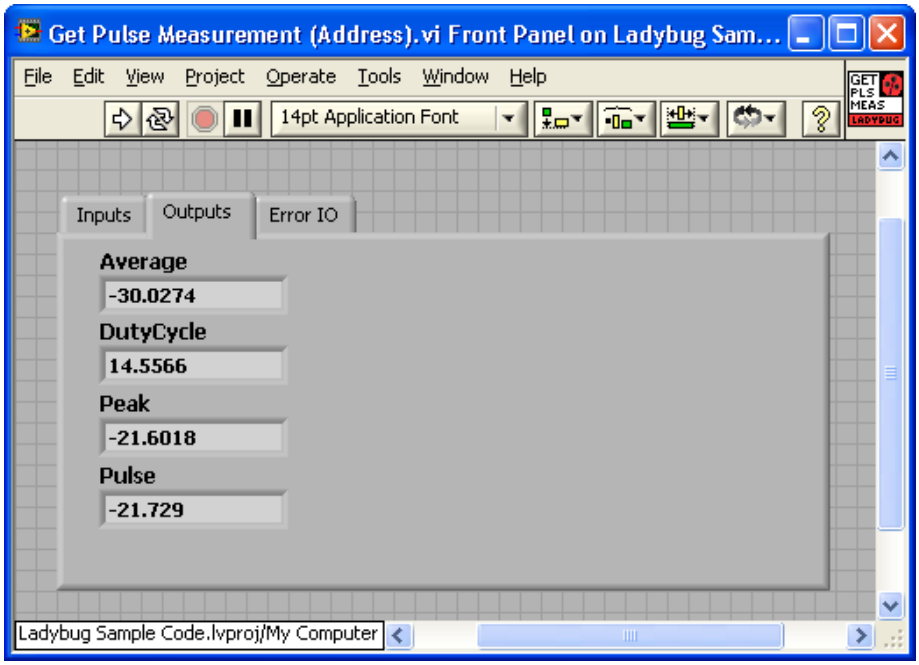


Figure 16: The Pulse Measurement VI is similar in format to the CW Measurement VI but note that it has four outputs.

To make a complete measurement, see the Pulse Measurement Example vi in the example directory.

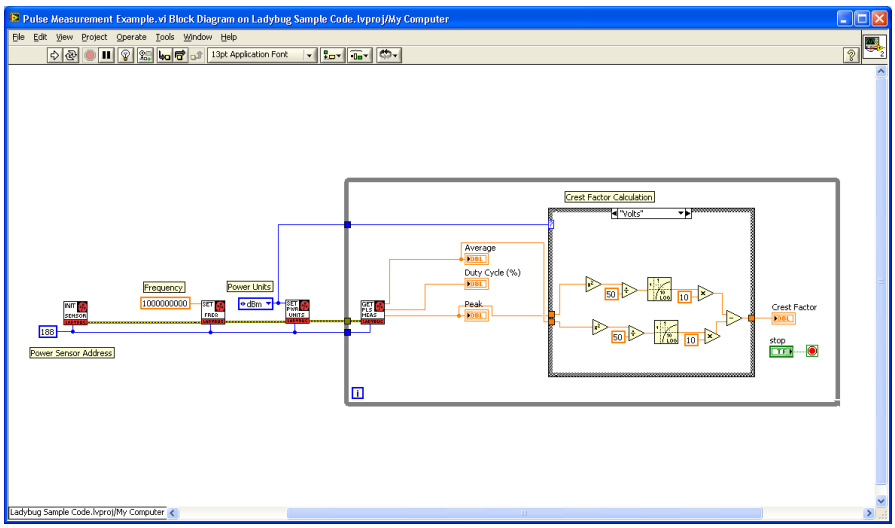


Figure 17: Note that the crest factor calculation is performed externally from the pulse measurement. This may change in future versions.

Making a Ratio Measurement

Ratio measurements simply involve setting up and measuring power levels from two sensors. There is no single VI that performs this measurement.

The Ratio Measurement Example VI demonstrates how to perform these measurements.

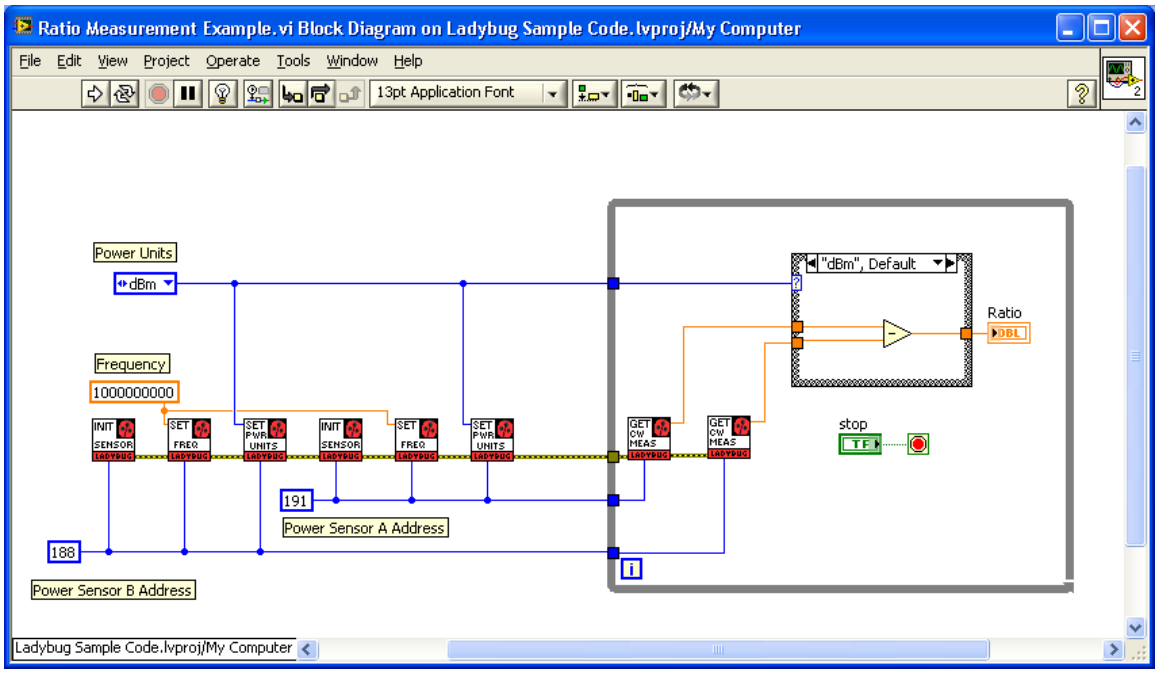


Figure 18: This example shows a CW ratio measurement of two sensors at 1 GHz. Note that units are treated differently in ratio measurements.

Adding to Existing Projects

The VI's distributed with this code can be added directly to any existing project. It is recommended that you add the entire folder to your project to ensure that the functions are readily available and easy to find.

Advanced Code Use

From the list in the appendix, it is clear that there is a substantial amount of functionality that is not supported at this time. We have focused our efforts on those functions which would be most needed.

In the event that you wish to use a function which is not supported, however, it is not difficult to add a function. It is highly recommended that you review some of the existing VI's in the Settings folder before proceeding. Follow these steps:

1) Open the Ladybug Sample Code LabVIEW project.
2) Open Template.vi (at the top level of the project).
3) Use Save As to save the Template.vi the Settings folder. Be sure to select "Copy", "Open Additional Copy", and "Add copy to Ladybug Sample Code.lvproj" as shown below.

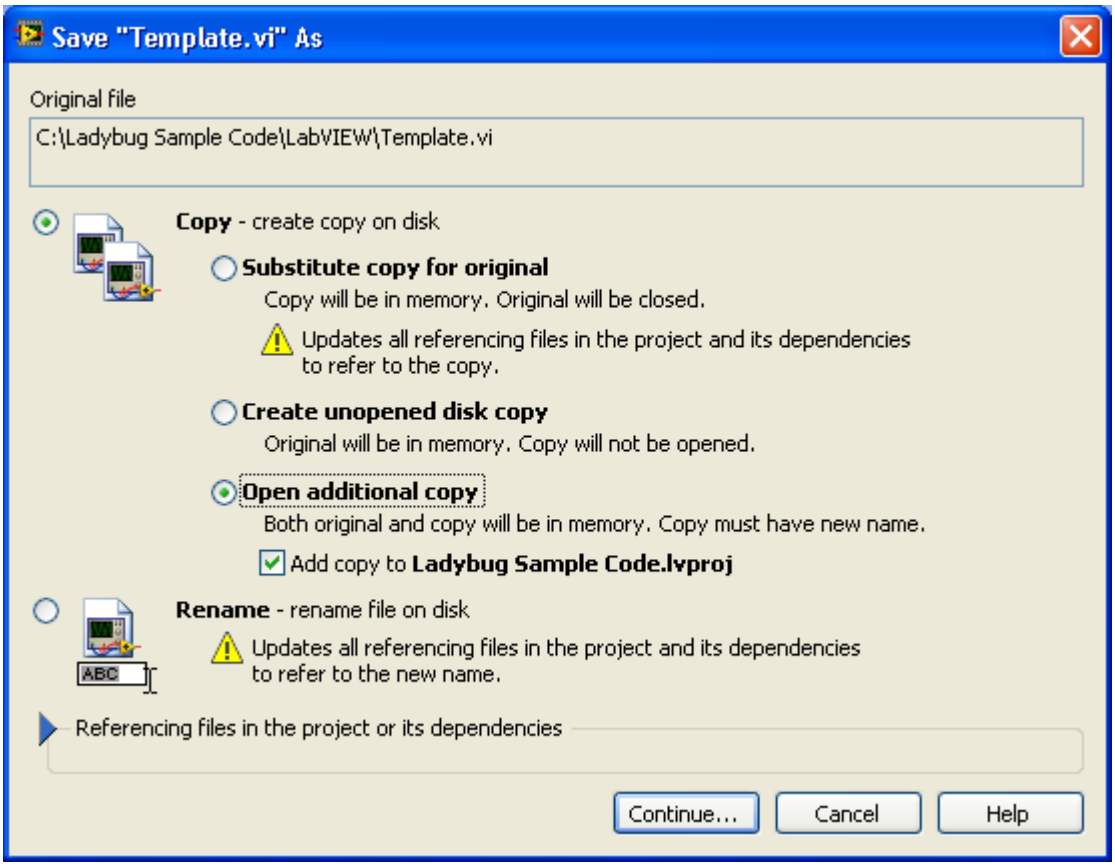


Figure 19: Save As Options must be selected as shown in this figure.

4) From the Project Explorer, expand “Dependencies”, “user.lib”, and “LB_API2.lvlib” as shown below.

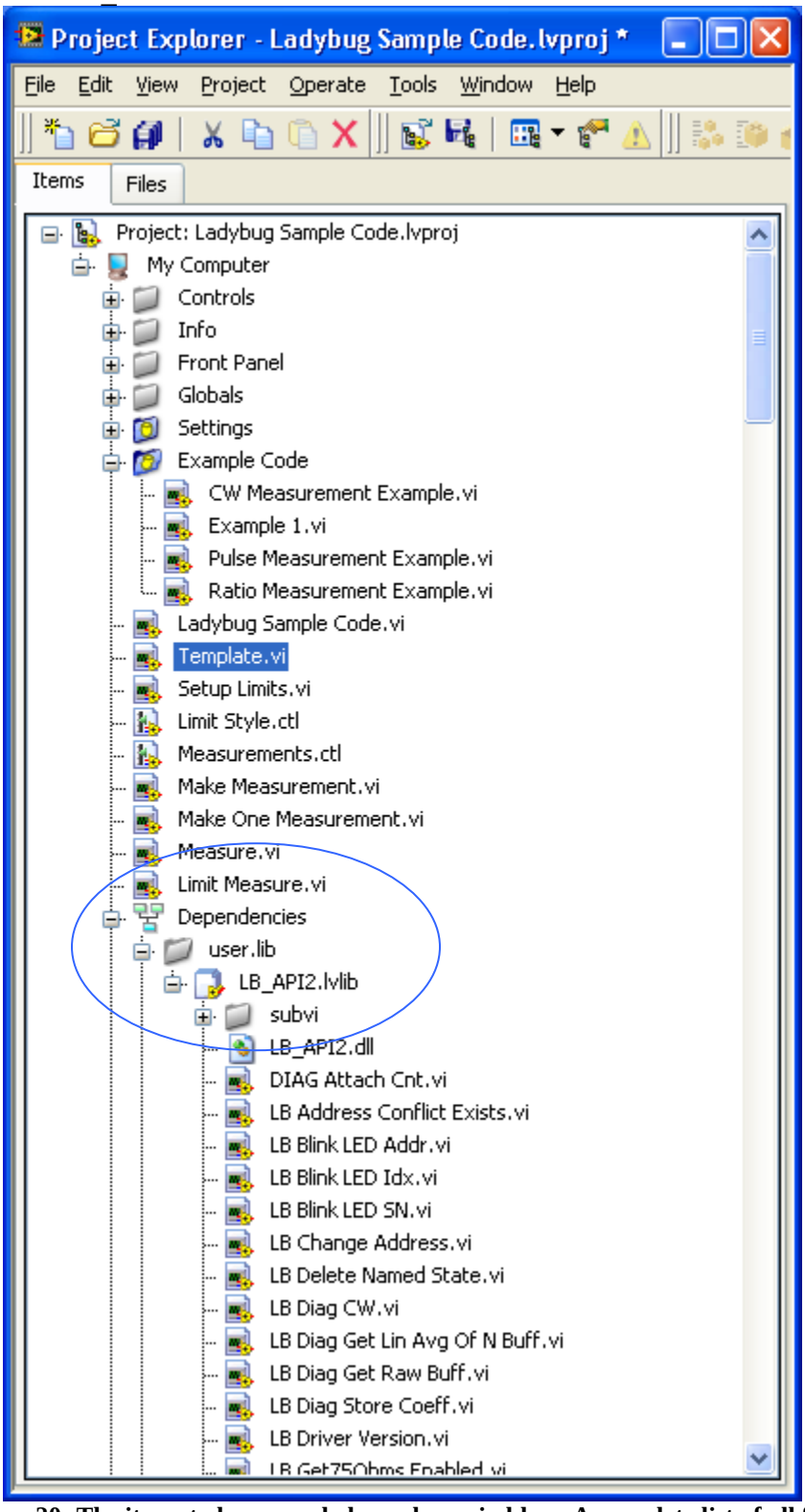


Figure 20: The items to be expanded are shown in blue. A complete list of all functions is included at the lowest level.

5) The VI's listed under LB_API2.lvlib are the lowest level VI's for each of the functions. Select the one that you want and drag it into the (renamed) template VI, as shown in the following figure.

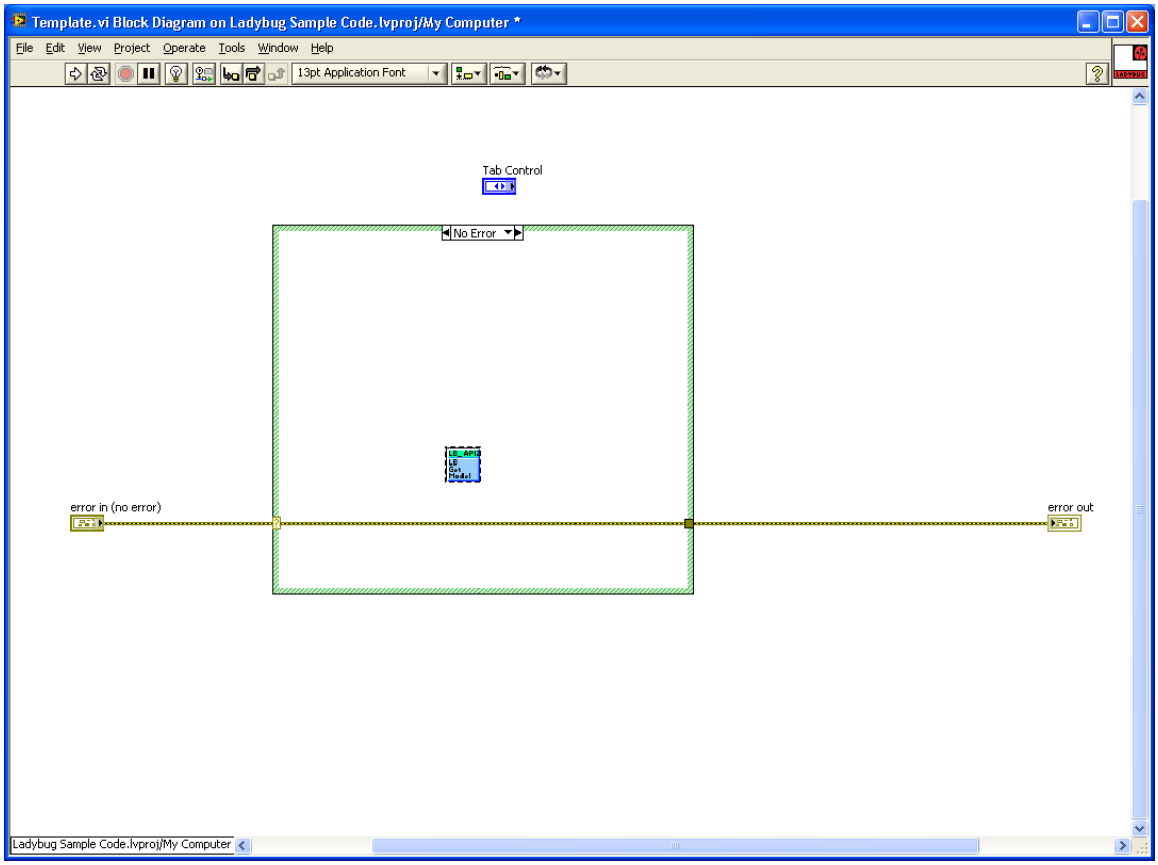


Figure 21: At this point, the low level VI has been added to the template VI but nothing has been connected (including the error in and error out terminals).

6) Create controls and indicators in the template VI as necessary and connect them to the connector pane. Edit the icon as needed. Add comments and documentation as appropriate.
7) Save the VI. It may now be used just as any of the other VI's provided.

APPENDIX

The following table lists all DLL functions and their corresponding VI's. Note that not all functions have VI's at this time. All VI's are in the settings folder.

DLL Function	VI
LB_AddressConflictExists	Not Available
LB_BlinkLED_Addr	ID (Address).vi
LB_BlinkLED_SN	Not Available
LB_BlinkLED_Idx	Not Available
LB_ChangeAddress	Not Available
LB_DriverVersion	Not Available
LB_GetAddress_Idx	Not Available
LB_GetAddress_SN	Not Available
LB_GetAntiAliasingEnabled	Get Anti Alias (Address).vi
LB_GetAutoPulseEnabled	Not Available
LB_GetAutoPulseEnabled	Not Available
LB_GetAverages	Get Averages (Address).vi
LB_GetBestMatchOpt	Not Available
LB_GetCalAndWtyOption	Not Available
LB_GetCalDueDate	Not Available
LB_GetCalOptExpDate	Not Available
LB_GetConnectorOption	Not Available
LB_GetCWReference	Get CW Reference (Address).vi
LB_GetDoubleSidedLimit	Not Available
LB_GetDutyCycleEnabled	Not Available
LB_GetDutyCyclePerCent	Not Available
LB_GetFilterOpt	Not Available
LB_GetFirmwareVersion	Not Available
LB_GetFrequency	Get Frequency (Address).vi
LB_GetIndex_Addr	Not Available
LB_GetIndex_SN	Not Available
LB_GetLimitEnabled	Get Limit Enabled (Address).vi
LB_GetMeasurementPowerUnits	Not Available
LB_GetModelNumber_Addr	Not Available
LB_GetModelNumber_Idx	Not Available
LB_GetModelNumber_SN	Not Available
LB_GetOffset	Not Available
LB_GetOffsetEnabled	Not Available
LB_GetPulseCriteria	Not Available
LB_GetPulseReference	Not Available
LB_GetRecorderOutEnabled	Not Available
LB_GetRecorderOutOption	Not Available
LB_GetRecorderOutSetup	Not Available

LB_GetResponse	Not Available
LB_GetResponseEnabled	Not Available
LB_GetSerNo_Addr	Not Available
LB_GetSerNo_Idx	Not Available
LB_GetSingleSidedLimit	Not Available
LB_GetTriggerOpt	Not Available
LB_GetTTLTriggerInEnabled	Get TTL Trigger In Enabled (Address).vi
LB_GetTTLTriggerInInverted	Get Trigger In Polarity (Address).vi
LB_GetTTLTriggerInTimeOut	Get Trigger In Timeout (Address).vi
LB_GetTTLTriggerOutEnabled	Get Trigger Out (Address).vi
LB_GetTTLTriggerOutInverted	Get Trigger Out Polarity (Address).vi
LB_GetWtyOptExpDate	Not Available
LB_InitializeSensor_Addr	Initialize Sensor (Address).vi
LB_InitializeSensor_Idx	Not Available
LB_InitializeSensor_SN	Not Available
LB_UninitializeSensor_SN	Not Available
LB_IsSensorConnected_Addr	Not Available
LB_IsSensorConnected_SN	Not Available
LB_MeasureCW	Get CW Measurement (Address).vi
LB_MeasureCW_PF	Get CW PF Measurement (Address).vi
LB_MeasurePulse	Get Pulse Measurement (Address).vi
LB_MeasurePulse_PF	Get Pulse PF Measurement (Address).vi
LB_ReadStateFromINI	Not Available
LB_Recall	Not Available
LB_ResetCurrentState	Reset Current State (Address).vi
LB_ResetRegStates	Not Available
LB_SensorCnt	Not Available
LB_SensorList	Not Available
LB_SetAddress_Idx	Not Available
LB_SetAddress_SN	Not Available
LB_SetAntiAliasingEnabled	Set Anti Alias (Address).vi
LB_SetAutoPulseEnabled	Not Available
LB_SetAverages	Set Averages (Address).vi
LB_SetBestMatchOpt	Not Available
LB_SetCalAndWtyOption	Not Available
LB_SetCalDueDate	Not Available
LB_SetCalOptExpDate	Not Available
LB_SetConnectorOption	Not Available
LB_SetCWReference	Set CW Reference (Address).vi
LB_SetDoubleSidedLimit	Set Double Sided Limit (Address).vi
LB_SetDutyCycleEnabled	Not Available
LB_SetDutyCyclePerCent	Not Available
LB_SetFilterOpt	Not Available
LB_SetFrequency	Set Frequency (Address).vi
LB_SetLimitEnabled	Set Limit Enabled (Address).vi

LB_SetMeasurementPowerUnits	Set Power Units.vi
LB_SetModelNumber	Not Available
LB_SetOffset	Not Available
LB_SetOffsetEnabled	Not Available
LB_SetPulseCriteria	Not Available
LB_SetPulseReference	Not Available
LB_SetRecorderOutEnabled	Not Available
LB_SetRecorderOutOption	Not Available
LB_SetRecorderOutSetup	Not Available
LB_SetResponse	Not Available
LB_SetResponseEnabled	Not Available
LB_SetSerialNumber	Not Available
LB_SetSingleSidedLimit	Set Single Sided Limit (Address).vi
LB_SetSysTimeOut	Not Available
LB_SetTriggerOpt	Not Available
LB_SetTTLTriggerInEnabled	Set TTL Trigger In Enabled (Address).vi
LB_SetTTLTriggerInInverted	Set Trigger In Polarity (Address).vi
LB_SetTTLTriggerInTimeOut	Set Trigger In Timeout (Address).vi
LB_SetTTLTriggerOutEnabled	Set Trigger Out (Address).vi
LB_SetTTLTriggerOutInverted	Set Trigger Out Polarity (Address).vi
LB_SetWtyOptExpDate	Not Available
LB_Store	Not Available
LB_WillAddressConflict	Not Available
LB_WriteStateToINI	Not Available
PP_AcquireTrace	Not Available
PP_CheckTrigger	Not Available
PP_CnvtTrace	Not Available
PP_CurrTrace2AnalysisTrace	Not Available
PP_GatePositionIsValid	Not Available
PP_GetAvgMode	Not Available
PP_GetAvgResetSens	Not Available
PP_GetFilter	Not Available
PP_GetGateAveragePower	Not Available
PP_GetGateCrestFactor	Not Available
PP_GetGateDroop	Not Available
PP_GetGateDutyCycle	Not Available
PP_GetGateEndPosition	Not Available
PP_GetGateEndTime	Not Available
PP_GetGateFallTime	Not Available
PP_GetGateMode	Not Available
PP_GetGateOverShoot	Not Available
PP_GetGatePeakPower	Not Available
PP_GetGatePRF	Not Available
PP_GetGatePRT	Not Available
PP_GetGatePulseWidth	Not Available

PP_GetGatePulsePower	Not Available
PP_GetGateRiseTime	Not Available
PP_GetGateStartEndPosition	Not Available
PP_GetGateStartEndPositionTime	Not Available
PP_GetGateStartEndTime	Not Available
PP_GetGateStartPosition	Not Available
PP_GetGateStartTime	Not Available
PP_GetMarkerAmp	Not Available
PP_GetMarkerDeltaAmp	Not Available
PP_GetMarkerDeltaTime	Not Available
PP_GetMarkerMode	Not Available
PP_GetMarkerPosition	Not Available
PP_GetMarkerPositionTime	Not Available
PP_GetMeasurementThreshold	Not Available
PP_GetPeaks_Idx	Not Available
PP_GetPeaks_Val	Not Available
PP_GetPeaksFromTr_Idx	Not Available
PP_GetPeaksFromTr_Val	Not Available
PP_GetPoles	Not Available
PP_GetSweepDelay	Not Available
PP_GetSweepDelayMode	Not Available
PP_GetSweepHoldOff	Not Available
PP_GetSweepTime	Not Available
PP_GetTrace	Not Available
PP_GetTraceAvgPower	Not Available
PP_GetTraceAvgs	Not Available
PP_GetTraceCCDF	Not Available
PP_GetTraceCDF	Not Available
PP_GetTraceCrestFactor	Not Available
PP_GetTraceDC	Not Available
PP_GetTraceLength	Not Available
PP_GetAnalysisTraceLength	Not Available
PP_GetTracePkPwr	Not Available
PP_GetTracePulsePower	Not Available
PP_GetTracePwrDensity	Not Available
PP_GetTriggerEdge	Not Available
PP_GetTriggerLevel	Not Available
PP_GetTriggerOut	Not Available
PP_GetTriggerSoure	Not Available
PP_MarkerNextPk	Not Available
PP_MarkerPkHigher	Not Available
PP_MarkerPkLower	Not Available
PP_MarkerPosIsValid	Not Available
PP_MarkerPrevPk	Not Available
PP_MarkerToFirstPk	Not Available

PP_MarkerToLastPk	Not Available
PP_MarkerToLowestPk	Not Available
PP_MarkerToPk	Not Available
PP_ResendState	Not Available
PP_ResetTraceAveraging	Not Available
PP_SetAnalysisTrace	Not Available
PP_SetAvgMode	Not Available
PP_SetAvgResetSens	Not Available
PP_SetClosestSweepTimeUSEC	Not Available
PP_SetFilter	Not Available
PP_SetGateEndPosition	Not Available
PP_SetGateEndTime	Not Available
PP_SetGateMode	Not Available
PP_SetGateStartEndPosition	Not Available
PP_SetGateStartEndTime	Not Available
PP_SetGateStartPosition	Not Available
PP_SetGateStartTime	Not Available
PP_SetMarkerDeltaTime	Not Available
PP_SetMarkerMode	Not Available
PP_SetMarkerPosition	Not Available
PP_SetMarkerPositionTime	Not Available
PP_SetMeasurementThreshold	Not Available
PP_SetPoles	Not Available
PP_SetState	Not Available
PP_SetSweepDelay	Not Available
PP_SetSweepDelayMode	Not Available
PP_SetSweepHoldOff	Not Available
PP_SetSweepTime	Not Available
PP_SetTraceAvgs	Not Available
PP_SetTriggerEdge	Not Available
PP_SetTriggerLevel	Not Available
PP_SetTriggerOut	Not Available
PP_SetTriggerSoure	Not Available
PP_TakeSweep	Not Available
LB_DiagCW	Not Available
LB_DiagStoreCoeff	Not Available
LB_DiagGetRawBuff	Not Available
LB_GetExtendedAveragingEnabled	Get Extended Averaging (Address).vi
LB_SetExtendedAveragingEnabled	Set Extended Averaging (Address).vi
LB_GetExtendedAveraging	Get Extended Averages (Address).vi
LB_SetExtendedAveraging	Set Extended Averages (Address).vi
LB_ResetExtendedAveraging	Reset XAvg (Address).vi
LB_GetMaxHoldEnabled	Get Max Hold (Address).vi
LB_SetMaxHoldEnabled	Set Max Hold (Address).vi
LB_ResetMaxHold	Reset Max Hold (Address).vi

LB_Get75OhmsEnabled	Not Available
LB_Set75OhmsEnabled	Not Available
PP_GetPulseEdgesPosition	Not Available
PP_GetPulseEdgesTime	Not Available
PP_AnalysisTraceIsValid	Not Available
LB_DiagGetLinAvgOfNBuff	Not Available
DIAG_AttachCnt	Not Available
LB_IsDeviceInUse_Idx	Not Available
LB_IsDeviceInUse_Addr	Not Available
LB_IsDeviceInUse_SN	Not Available
LB_SaveCurrStateTo	Not Available
LB_RestoreCurrStateFrom	Not Available
LB_GetNamedStateCount	Not Available
LB_GetNamedStateNames	Not Available
LB_DeleteNamedState	Not Available