Ladybug VEE Power Meter Code

Overview

This document introduces the user to using Ladybug power sensors in a VEE environment. It assumes familiarity with VEE 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 power meter application (with source code) is included. The user interface is shown in figure 1. It is designed to be easy to understand and uses many of the most common functions. It will work with one or more sensors.

Note: All VEE software is written in VEE 7.5.
Figure 1: VEE Power Meter 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 VEE Power Meter Application

The VEE Power Meter 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 unzip all of the files in the appropriate folder. Note that all files should remain in the same folder. This is especially true of the .DLL and .H files. If you wish to create your own VEE application, the location of these files is critical.
**Setting the Frequency**

To set the frequency, change the value in the frequency control (shown below in yellow). You may also change the units (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.

![Measurement Panel](image)

*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 the drop-down list (shown in yellow). Note that the power units display shows the units for an individual sensor during ratio measurements.

Figure 3: The power units control only controls sensor A. It also displays the current setting for Sensor A.
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 Power, and Crest Factor (all shown in yellow) appear after this selection is made. Also, note that 5xx series sensors cannot make pulse measurements even though this option may be selected.

![Image of pulse measurement setup](Image.png)

**Figure 5:** The additional pulse measurements are shown on the right. The units for duty cycle are always 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.

![Measurement Panel](Image)

**Figure 6:** Note that the ratio measurements are always A-B (or A/B for linear units). To display B-A, simply switch the selected sensors.
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.

Library Setup

Since all calls to the power sensor are done with DLL function calls, the library must be imported prior to calling any of the routines. This is a fairly simple process with an example of the VEE code shown in the following figure.

Figure 7: The crucial setting for an imported library is setting the Library Type to “Compiled Function”. It is also good practice to delete the library prior to importing it.

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 power meter code, these steps are performed as shown in the following figures.

Figure 8: "Get Sensor List" userFunction. Note that the SensorList is an array of records.

Figure 9: "Initialize All Sensors" uses the address fields of each record in the array.
An easy way to initialize all sensors is shown below:

**Sensor Identification**

To make the LED on the current power sensor blink several times, use the LB_BlinkLED_Addr function.
**Setting the Frequency**

To set the power sensor frequency, use the LB_Set_Frequency function. Note that the frequency must be in Hertz. You will need to account for this before calling this function (as was done in the power meter code).

Figure 12: With these parameters, the power sensor at address A would be set to the frequency of the Freq pin.
**Setting the Power Units**

To set the power units, use the LB_SetMeasurementPowerUnits function as shown in the figure. Possible units include dBm, dBW, dBkW, dBuV, dBmV, dBV, Watts, Volts, and dB Relative.

![Diagram showing the process of setting power units](image)

*Figure 13: With these parameters, the LB_SetMeasurementPowerUnits will set the power sensor to dBm. Note that the ordinal output of the radio button control is used (not the enum). Also, the order of the units in the radio buttons control is critical.*
Making a CW Measurement

The Get CW Measurement (Address).vee example will measure the CW power for the sensor at the given address in the selected units of the sensor. This example shows the minimum required to make this measurement, including importing the library, initializing the sensor, setting the frequency, setting the power units, and making the measurements.

Figure 14: 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**

Similarly, the Get Pulse Measurement.vee example will measure the pulse power for the selected sensor.

![Diagram of Pulse Measurement example]

Figure 15: The Pulse Measurement example is similar in format to the CW Measurement example but note that it has four outputs.
**Making a Ratio Measurement**

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

The Ratio Measurement.vee example demonstrates how to perform these measurements. Note that ratio measurements are subtracted for logarithmic units (such as dBm) and divided for linear units.

![Figure 16: This example shows a CW ratio measurement of two sensors at 1 GHz. Note that units are treated differently in ratio measurements.](image)
Adding to Existing Projects

The VEE files distributed with this code can be added directly to any existing project. It is recommended that you check to ensure that the DLL functions are imported correctly at the beginning of the program. This may require modification to the folders specified by the import DLL function.