Thank you for selecting Silicon Labs for your SLIC and VDAA solution. This quick start guide covers how to bring up the ProSLIC API demo application on your development system with the evaluation hardware. With the ProSLIC API demo, you can explore the extensive feature set of the ProSLIC devices, as well as implement custom code to evaluate system requirements prior to committing to a final hardware design.

Development environment

It is assumed (since you are reading this) that you have already installed on your development system the ProSLIC API release. In addition to the ProSLIC API software, you will need to have the following installed:

- GCC or equivalent (LLVM/Clang)
- GNU Make
- BASH shell interpreter
- Grep & cut text utilities
- Recommended: PERL

Make sure you have the utitlies mentioned above installed. Please refer to your Linux®[1] distribution on how to do this. See the ProSLIC API Users Guide for other suggested software in the demo section.

The VCP driver should already be included as part of your distribution's Linux kernel. If not, please refer to your distribution's instructions on how to add the Silicon Labs VCP driver as either a kernel module or as part of your kernel.

It is highly recommended that your system is running in a non-virtual environment. We have seen cases of the USB drivers not behaving correctly in a virtual environment.

Identifying the hardware

The Silicon Labs evaluation system comprises of two boards that need to be identified:

- Voice Motherboard platform that provides the interface from the Windows system via USB cable and generates the needed PCM and control interfaces.
- ProSLIC or VDAA evaluation board (EVB) that implements the circuit being evaluated.

[1] Linux® is the registered trademark of Linus Torvalds in the U.S. and other countries.

Silicon Labs has two variants of voice motherboards: the Voice Motherboard 1 (VMB1) and the Voice Motherboard 2 (VMB2). Either board can be used for your evaluation of the ProSLIC/VDAA solution. Please identify which board you have since the software will need to be configured correctly to communicate with the given board.

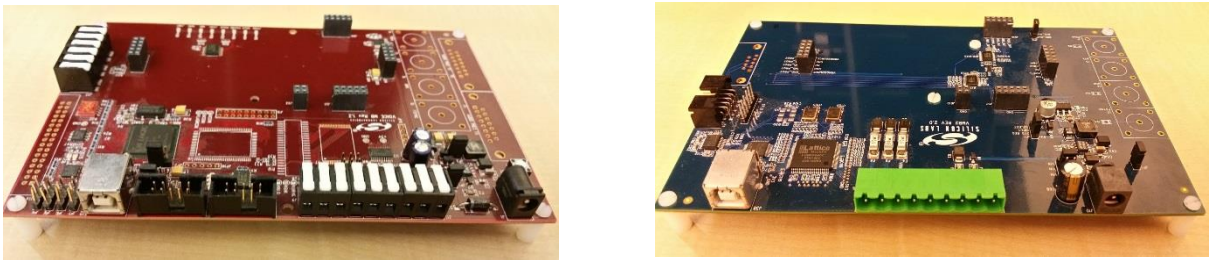

Figure 1 Voice Motherboard1 (left) & Voice Motherboard 2(right)

For the ProSLIC & VDAA evaluation board, Silicon Labs provides a basic utility called id_evb that identifies the EVB installed on the motherboard. To build this utility on your system, do the following steps from your BASH shell/terminal:

1) cd <ProSLIC API install directory>/demo/id_evb/build
   VMB2: `make VMB2=1`
2) After the software compiles, you should have an executable called id_evb in this directory.
3) If the EVB has not already been plugged into the Voice Motherboard, plug it in while the motherboard is not connected to anything.
4) Make sure you have power and USB connected to the VMB2 at this point. Your system should recognize that a USB device has been connected.
5) For Desktop Linux, you need to specify which tty port to use to communicate with the VMB2 by setting the SIVOICE_SPI_IF environment variable.

   For example, if the VMB2 was instantiated on /dev/ttyUSB0, you would type the following:
   `export SIVOICE_SPI_IF=/dev/ttyUSB0`

   If you are unsure which tty device to use, please check your system log with the dmesg command.

6) Enable the shell script to be executed with the following command: `chmod 700 map.sh`
7) Run the shell script map.sh that will identify the EVB installed on your system. The following is a sample run:

```
./map.sh
Reading EEPROM string - this will take a few seconds
Rev = B
SI3217X_B_FB
```


The software identified that the EVB installed is a Si3217x Rev B Flyback EVB. The last string is what is needed to build the ProSLIC API demo program covered in the next section.

Building the ProSLIC API Demo

Once you have identified the hardware installed on your system. Perform the following steps:

1) cd <ProSLIC API install directory>/demo/api_demo/build
2) For VMB2: `make VMB2=1 <EVB DETECTED>=1`

   For the earlier example this would be on a VMB2 based system:
   `make VMB2=1 SI3217X_B_FB=1`

3) At this point you build directory should have an executable called si3217x_b_fb. Execute it with:
   `./si3217x_b_fb`

You may at a later stage specify your own configuration/constants file with the PROSLIC_CFG_SRC makefile option. If you decide to use this, please make sure you have your constants file located in the <ProSLIC API install directory>/demo/api_demo/custom directory.

Now you can start the ProSLIC API demo and try the various options shown in its text based menu system.

Where to go from here…

Here are a few topics you may want to explore after exploring the ProSLIC API demo:

- Learn more about the capabilities found in the ProSLIC API – such as inward testing, porting to an embedded system, interrupt handling, and demos. You can learn more in the ProSLIC API Users Guide.
- Embedded Linux Integration guide has information related to both SPIDEV (user space SPI drivers) and kernel space drivers. It has information on how to build the example drivers and how to run the demos under an embedded system.
- Metallic Line Testing (MLT) – such as foreign voltage detection, Ringers/REN load, and Receiver Offhook tests, please refer to the ProSLIC MLT API software package.
- Caller ID FSK generation - Caller ID Framework as a separate package. This framework consists of several different FSK encoders and signaling statemachines. Please refer to its documentation on how to compile and run this framework.

If you have questions or comments, please let us know – either through your local Silicon Labs support group or via the support portal at www.silabs.com. Finally, please do subscribe in the NDA portal for notifications about updates about the ProSLIC API and other ProSLIC collateral.

Thank you again for selecting Silicon Labs.