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 MinGW Eclipse based 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.

NOTE: Some of these steps can also be used for Cygwin with Eclipse…

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 for this environment:

- GCC, GNU Make, PERL from http://www.mingw.org/
- A JVM for Eclipse – from https://www.java.com/en/download/
- Eclipse CDT (see: http://www.eclipse.org/)
- Lua development environment for MinGW (optional)

In addition to MinGW and the utilities mentioned, you will need Silicon Lab VCP driver installed on your system. This can be downloaded from:

http://www.silabs.com/products/mcu/Pages/USBtoUARTBridgeVCPDrivers.aspx

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.

NOTE: There are some incompatibilties with the latest Windows 7/8 VCP driver and the VMB2 firmware. If you are using Windows 7 or 8 on a new install, please download the VCP driver available under the NDA support portal where you downloaded the API from.

MinGW Setup…

In addition of installing the software packages mentioned above, you will need to add to your .profile file the following:

```
export CC=gcc

export PERL_PATH=/bin/perl.exe
```

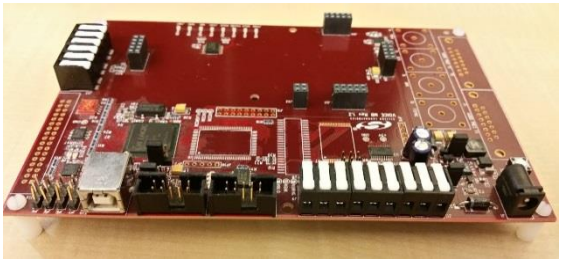
This file is located under c:\MinGW\msys\1.0\home\<your windows id>\.profile. If this file does not exist, you will need to create it. If you instead prefer to edit the file under the MinGW shell (located under c:\MinGW\msys\1.0\msys.bat), the file is under: ~/.profile.

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.

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.



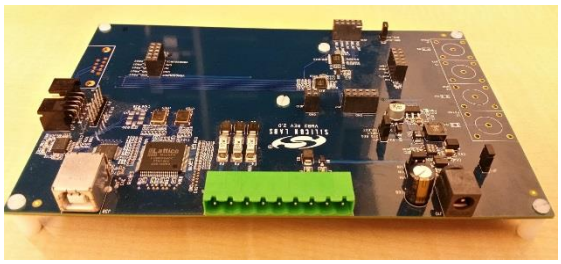


Figure 1 Voice Motherboard1 (left) & Voice Motherboard 2(right)

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
   VMB1: `make VMB1=1`
   VMB2: `make VMB2=1`

   NOTE: MinGW uses Linux style pathing. So C:\Siliabs\ProSLIC\proslic_api_X.Y.Z\proslic_api is /c/SiLabs/ProSLIC/proslic_api_X.Y.Z/proslic_api – note the slashes for MinGW.

   NOTE2: The bash script terminal can be started for MinGW by running C:\MinGW\msys\1.0\msys.bat . It is suggested to create a desktop shortcut for this.

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 VMB1/2 at this point. Your system should recognize that a USB device has been connected.
5) For MinGW, we use the same DLL's as found in Cygwin's development environment: add to your path variable the DLL paths with the following command:

   ```
   export PATH = $PATH:../../platform/cygwin/bin
   ```


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.

Eclipse setup…

At this point we will cover how to configure Eclipse to be used with MinGW to compile/run/debug the ProSLIC API demo. Please note, we are using Eclipse Mars.1 Release 4.5.1 for this document. Menu items are subject to change.

Adding the ProSLIC API to Eclipse

1) Start Eclipse with CDT.
2) When prompted, create a new workspace for this project.
3) Close the Welcome screen.
4) Under File->New->Makefile with exiting code
5) Enter a project name when prompted.
6) Exiting code Location should be where you have the ProSLIC API installed – for example: c:\SiliconLabs\ProSLIC\proslic_api_8.1.1\proslic_api
7) For Languages, the API only uses C, you may deselect C++
8) For Toolchain for Index Settings, Select MinGW GCC
9) Click Finish

At this point you have the C/C++ project view. You will need to configure the build environment and the debug environment.

Build configuration

For the build environment, we need to add the makefile and build options needed to build your project. This write-up will cover just one configuration, but you may want to have at least two configurations: debug and release for each hardware combination under development.

Follow these steps to add to a configuration:

1) Right click on your project top level folder in the Project Explorer Window on the right, select properties.
2) Under C/C++ Build, select Build Variables.
3) Add 3 build variables (click Add… to do so):

Name	Type	Value
ProSLIC_VMB	String	VMB1=1 or VMB2=1, depending on your system
ProSLIC_Converter	String	<build option from id_evb>=1 For example: SI3217X_B_FB=1
ProSLIC_DEBUG	String	DEBUG=1

4) Under C/C++ Build, select Tool Chain Editor. Change the Current Builder to "Gnu Make Builder"
5) Now under C/C++ Build, under Builder Settings:
    a. Deselect "Use default build command"
    b. The build command should be:

        make ${ProSLIC_Converter} ${ProSLIC_VMB} ${ProSLIC_DEBUG} FLUSH_STDOUT=1

    c. Deselect "Generate Makefiles automatically".
    d. Under "Build location", set the Build directory to: <ProSLIC API install directory>\demo\api_demo\build.
    For example: c:\Silabs\ProSLIC\proslic_api_8.1.1\proslic_api\demo\api_demo\build

        NOTE: You may want to have an environment variable to store which version of the ProSLIC API you are compiling against in place of a hard coded path.
    e. Click apply

6) Now under C/C++ Build, Environment:
    a. Add the variable CC and set it to gcc.
    b. Modify your PATH variable to include <ProSLIC API install directory>\demo\platform\cygwin\bin. This will pull in the VMB1/VMB2 DLL's needed by the executable to run.
7) Under C/C++ Run/Debug Settings, append the same path as done in step 6.b.
8) At this point you should be able to build the API demo with the above settings. Click on the little hammer icon in the toolbar to build the API Demo.
9) Under Run/Debug Settings, create a New C/C++ Application setting.
    a. When prompted, for the C/C++ application, add the executable under <ProSLIC API install directory>\demo\api_demo\build\<exe name>.exe - For this walk through, there is a _DBG in the name since we are building the debug version of the program. For the non-debug version, the _DBG suffix is omitted.
    b. Click on the Environment Tab and then click New…
    c. When prompted, enter under Name: Path and for the value: <ProSLIC API install directory>\demo\platform\cygwin\bin.
    d. Make sure the "Append environment to native environment" is selected. Click Apply & then OK.

    NOTE: for Cygwin Users using MinGW compiler: Under Window->Preferences->C/C++ -> Debug-> Souce Lookup Path, add a map for : /cydrive/c c:\
10) At this point you should be able to run or the program. Click on the green circle with the arrow in the toolbar or CTRL-F11 to run. To start the debugger, either click on the little bug in the

toolbar or F11 to debug. Please note, it may take several seconds to get responses initially from the console window in the lower right side. If you do have some issues having the demo find the DLL's, you may need to copy them to where the executable resides.

Where to go from here…

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

- Configuring the EVB for your application - the Silicon Labs provides a configuration tool to modify and add various setting such as ringer presets, tone presets, dc-feed presets and FSK settings. We provide XML files that are inputs used to generate the configuration/constants C files used in the demo under the ProSLIC API install directory. Please refer to the ProSLIC API User's guide for details on this utility.
- 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.