Symphony SoundBite Demo

User's Guide

(Including Factory Board Test Procedure)

Document Number: SNDBDMOUG
Rev. 2.0
09/08

freescale™
semiconductor

How to Reach Us:

Home Page:
www.freescale.com

E-mail:
support@freescale.com

USA/Europe or Locations Not Listed:
Freescale Semiconductor
Technical Information Center, CH370
1300 N. Alma School Road
Chandler, Arizona 85224
+1-800-521-6274 or +1-480-768-2130
support@freescale.com

Europe, Middle East, and Africa:
Freescale Halbleiter Deutschland GmbH
Technical Information Center
Schatzbogen 7
81829 Muenchen, Germany
+44 1296 380 456 (English)
+46 8 52200080 (English)
+49 89 921 03 559 (German)
+33 1 69 35 48 48 (French)
support@freescale.com

Japan:
Freescale Semiconductor Japan Ltd.
Headquarters
ARCO Tower 15F
1-8-1, Shimo-Meguro, Meguro-ku,
Tokyo 153-0064, Japan
0120 191014 or +81 3 5437 9125
support.japan@freescale.com

Asia/Pacific:
Freescale Semiconductor Hong Kong Ltd.
Technical Information Center
2 Dai King Street
Tai Po Industrial Estate
Tai Po, N.T., Hong Kong
+800 2666 8080
support.asia@freescale.com

For Literature Requests Only:
Freescale Semiconductor Literature Distribution Center
P.O. Box 5405
Denver, Colorado 80217
1-800-521-6274 or 303-675-2140
Fax: 303-675-2150
LDCForFreescaleSemiconductor@hibbertgroup.com

Contents

About This Book

This document describes how to use and execute the demonstration application that is preprogrammed into the Symphony SoundBite board. This application doubles as the board test software during manufacture of the board. The details of the board test procedure and how to build the demonstration application from the source files are also included. The source code for the demonstration application is provided separately.

Audience

This guide is intended for users of the Symphony SoundBite audio development board.

Organization

This document is organized into the following chapters:

Chapter 1 Describes the Symphony SoundBite board.

Chapter 2 How to run the preprogrammed demonstration application.

Chapter 3 How to perform the factory board test procedure.

Chapter 4 How to build the demonstration application from the source files.

References

Also see:

- *Symphony SoundBite Board Reference Manual (SNDBITERM)*
- *Symphony SoundBite Assembly Project Template*

Chapter 1 About Symphony SoundBite

1.1 Hardware Features

Built around a single DSPB56371 device, the Symphony SoundBite is an inexpensive, feature-packed solution for audio digital signal processing.

- The Symphony SoundBite supports the simultaneous input and output of 8 channels of analog audio (or 6 channels of analog with digital stereo optical S/DIF) through the eight 3.5 mm stereo jacks on the board. Sample rates of up to 192 kHz are supported with the four 24-bit stereo codecs provided on the board.
- A serial I^2C EEPROM provides non-volatile storage for the DSP, as well as allowing the board to operate in a stand-alone configuration (the DSP application code can be stored in the EEPROM).
- An 8-position DIP switch and 9 LEDs are connected to GPIO pins on the DSP and are available for user interaction and indications. An expansion header is provided to facilitate off-board expansion.
- An on-board USB interface provides low-level JTAG/OnCE debugging capability. The USB interface also provides high-level SPI and I^2C communication via the DSP's SHI port. The board can be powered via the USB jack, but optimum audio performance requires the use of an external power supply.
- The analog inputs and outputs are considered to be line-level. Since two different types of codecs are used, there is a slight difference in the full-scale maximum input and output voltages between them. Please refer to the data sheets for the AKM codecs AK4556 and AK4584 for more information.

NOTE: Although headphones may be plugged into the output jacks and sound may be heard in them, there is no headphone amplifier built into the codecs nor is one provided on the Symphony board. Clipping will occur at higher output levels due to the load headphones present to the codec outputs. An external amplifier should be employed when using headphones.

1.2 Symphony SoundBite Board Layout

Figure 1-1 shows the Symphony SoundBite board.

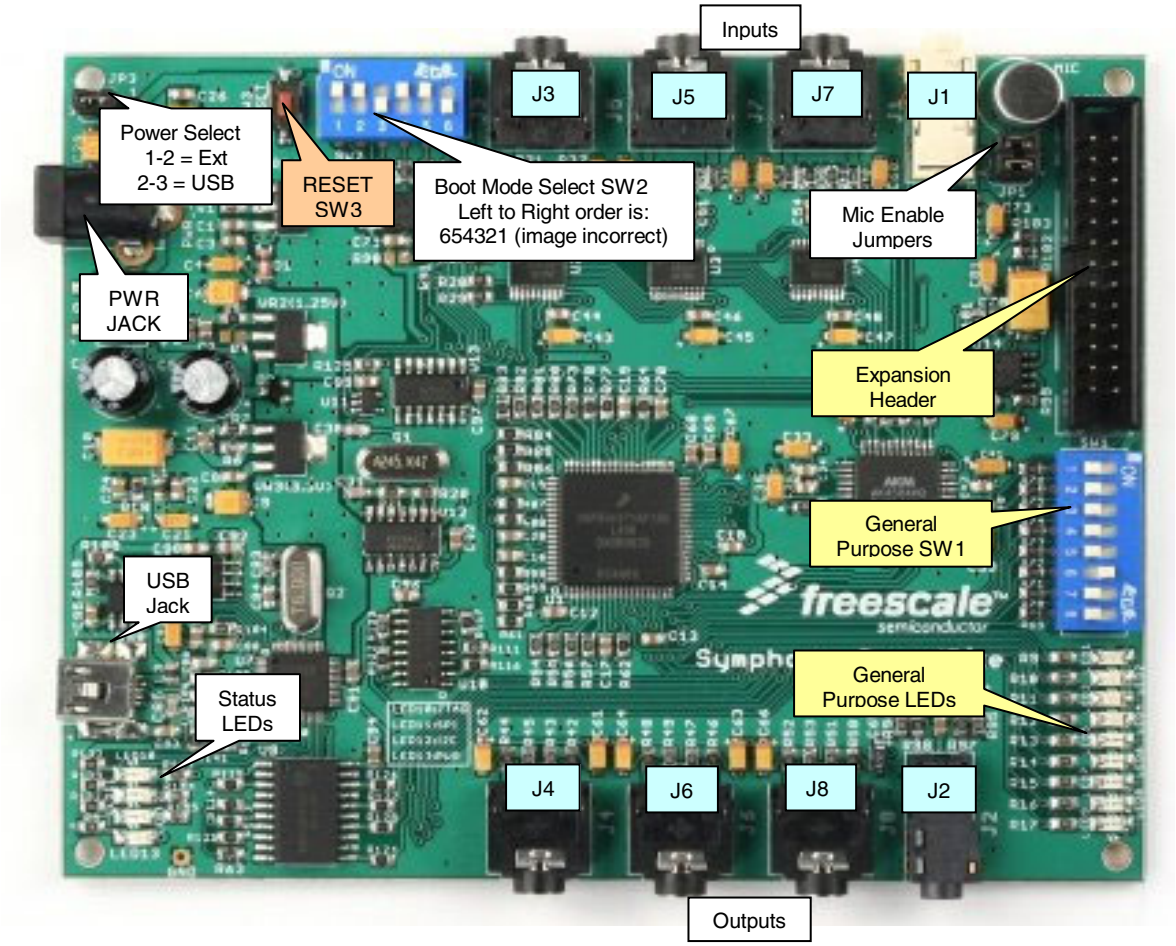

Figure 1-1: Symphony SoundBite Board Layout

Chapter 2 Using the Demonstration Application

At the factory, the EEPROM on the Symphony SoundBite is programmed with a demonstration application that also doubles as the functional test software for the board during manufacture. The demonstration application allows you to do "something" with the Symphony SoundBite board directly out of the box, without installing drivers or software.

2.1 Configuring the Jumpers and Switches

On board start-up, the demonstration application reads the status of the bank of 8 general purpose switches in the SW1 DIP switch, which selects how the board passes/generates audio. This demonstration application originated from test software written to allow the measurement of the audio performance of the various subsystems of the Symphony SoundBite with an Audio Precision analyzer.

To restore the Symphony SoundBite to the original factory configuration and to provide a known starting point:

Set the switches and jumpers as indicated in Table 2-1, and choose the appropriate power source for the board (which is fixed at the factory) that you want to use.

Table 2-1: Factory Default Configuration

Jumper or Switch		Effect
JP3	Jumper 2-3 or Jumper 1-2	Board is powered from USB. or Board is powered from PWR_JACK.
JP3	Jumper 1-2, 3-4	Enable microphone input.
SW2	110110 (positions 654321, 1=ON)	Enable EEPROM and Boot Mode 9 (Boot from serial I^2C EEPROM).
SW1	1000 0000 (positions 1234 5678, 1=ON)	J1 analog input passed to all outputs.

2.2 Running the Demonstration Application

When power is applied to the Symphony SoundBite board, the status LED (LED13) will light. The reset manager (U11) de-asserts the reset line of the DSP (U1) after a period of time, causing the DSP to read the boot mode switches (SW2 positions 4:1). As configured in Table 2-1, the boot mode switches select Boot Mode 9 (boot from serial I^2C EEPROM). SW2 switch positions 6:5 connect the SDA and SCL lines of the I^2C EEPROM (U6) to the DSP (U1). The DSP downloads the application code from EEPROM to internal RAM and then executes it. Depressing the RESET switch (SW3) is equivalent to power-cycling the board.

When the demonstration application starts up, it configures the DSP's GPIO pins such that the DIP switches may be read and the general purpose LEDs may be toggled on and off. The status of the DIP switch is then read to determine the behavior (as described in Table 2-2) that the user desires, and the AK4584 (U5) is configured accordingly. Next, interrupts are enabled, which allows the audio signals to be processed by the DSP. In the main loop of the application, an LED (LED9) flashes a heart beat to indicate that the DSP is up and running. The DIP switch status is continually read, with the state of each switch constantly displayed at LED1 through LED8.

To run a different configuration of the demonstration application, change SW1 to the desired setting (according to Table 2-2) and reset the board using the RESET switch (SW3) (or power-cycle the Symphony SoundBite board).

For more about the internal workings of the demonstration application, refer to the application source code and the *Symphony SoundBite Assembly Project Template* document.

Table 2-2: Configurations for SW1 DIP Switch

SW1 1234 5678	Demonstration Application Configuration
1000 0000	J1 optical input passes through to all outputs.
0100 0000	J3 analog input passes through to all outputs.
0010 0000	J5 analog input passes through to all outputs.
0001 0000	J7 analog input passes through to all outputs.
0000 1000	Individual channel pass-through: J1>J2, J3>J4 , J5>J6, J7>J8.
0000 0100	Generates 1 kHz on all outputs
0000 0010	Generates sine waves of various frequencies.
0000 0001	Generates sine waves of various frequencies (different from above).
1111 1111	J7 analog input > J2 optical output

Chapter 3 Factory Board Test Procedure

The demonstration application is used during the testing of the Symphony SoundBite during manufacture. As such, you may perform the factory test procedure to verify proper board operation and to return the board back to the original factory state.

About the Multi-Protocol USB Communication Interface:
The USB interface on the Symphony SoundBite is based on the FT-2232 dual USB UART which is manufactured by Future Technology Devices International (FTDI). This device can be configured to use various built-in synchronous serial modes. Only three serial modes are of interest with regard to the Symphony SoundBite: JTAG, I2C, and SPI. For more about the multi-protocol interface on the Symphony SoundBite, refer to the *Symphony SoundBite Hardware Reference Guide*.

In order for the Symphony SoundBite to be properly recognized by the host PC, the non-volatile memory (U8) attached to the FT-2232 device (U7) must be programmed with the appropriate "personality" information using the template file ***SymphonySoundBite.ept***, which is provided in the accompanying zip file archive.

Re-programming U8: In normal use, reprogramming U8 should never be required. Detailed information regarding reprogramming U8 is beyond the scope of this document. The information below is provided for informational purposes only.

To program U8, you need the appropriate Windows driver and a free utility provided by FTDI, ***MProg 3.0a***:

http://ftdichip.com/Drivers/D2XX.htm for the driver

http://ftdichip.com/Resources/Utilities.htm for the programming utility, ***Mprog 3.0a***

About Programming the Firmware:
The demonstration application is programmed into the serial EEPROM (U6) using a small subset of the Symphony Studio software development tool suite (OpenOCD and the command line GDB debugger for DSP563xx/7xx, to be specific) with several Windows batch files. To download the demonstration application code to the DSP, these files make a connection from the host PC via the USB port through the JTAG debugging port. Once downloaded, the program counter of the DSP is changed to the address of the routine within the application that programs the demonstration application into the EEPROM.

3.1 Performing the Factory Board Test

To run a functional test on the Symphony SoundBite board:

1) Configure jumpers and switches on the board according to Table 2-1.
2) Verify that the FTDI drivers are installed and that the FT-2232 EEPROM has been programmed with the Symphony SoundBite template.
3) Connect the USB cable between the host PC and the Symphony SoundBite board. If using an external power supply, connect it now and turn it on. LED13 (power indication) should light.

   If LED13 does not light, verify that jumper JP3 is set for the correct power supply source and that the USB drivers on the host PC are properly installed.
4) Launch the batch file ***00-OpenOCD.bat***, which establishes the *host PC* to JTAG debugging connection. (Note: Sometimes OpenOCD is not always able to connect to the DSP after the board is freshly powered up and has been freshly connected to the host PC.)

If OpenOCD does not connect to the board, an error occurs (indicated by more than one line appearing in the command window console or if the window closes), so re-launch the batch file.
After OpenOCD succeeds in connecting to the board, a single line in a command window will appear similar to:

```
Info: openocd.c:82 main(): Open On-Chip Debugger ps001 (2007-10-19 18:00 CEST)
```

5) Launch the batch file ***01-programSoundBite.bat***, which starts the EEPROM programming cycle.
Another command window should pop up. Some text should scroll in this new window as well as in the OpenOCD window, indicating that a successful connection was made between the debugger executable and OpenOCD.

   After the scrolling stops, the general purpose LEDs on the Symphony SoundBite board will begin to flash. If the LEDs do not start flashing after the text stops scrolling, check the boot mode and EEPROM select DIP switch SW2 (particularly SW2 positions 5 and 6, which connect the EEPROM to the DSP).

   LED9 flashes a constant rate heartbeat throughout the programming cycle. LED1-LED8 will count up in binary throughout the programming cycle until all of the LEDs (LED1-LED8) are lighted.

   At the completion of the programming cycle, LED9 goes out (while leaving LED1-LED8 lighted), which indicates that the board is fully programmed.

6) Remove the USB cable (if it is not being used to power the board). Depress the RESET switch (SW3), which reboots the DSP and resets the board. After a short pause, LED9 should start flashing a heartbeat. LED1 should be steadily ON while all of the general purpose LEDs remain OFF, reflecting the state of the switches of SW1. Note that the default state according to Table 2-1 is 1000 0000 (1234 5678).

7) Test all of the positions of the SW1 DIP switch by toggling each position ON and OFF, verifying that each switch and LED pair works correctly. For each switch, the corresponding general purpose LED should display the ON/OFF state of the switch.

8) Verify that the microphone on the Symphony SoundBite board works.
Connect headphones or an amplified speaker to any of the output jacks J2, J4, J6 or J8, and make a sound or tap the microphone to verify that it works correctly (you should hear what you do).

   If no sound is heard, verify that the jumpers on JP1 are properly configured, and that the board was rebooted with 1000 0000 (1234 5678) set on SW1. Recall that according to Table 2-2, the SW1 switch configuration routes analog input on J1 (the microphone with JP1 jumpered) to all the analog outputs.

9) Test the S/PDIF optical input and output.
The S/PDIF optical input and output can be tested without any additional optical equipment, by using the Symphony SoundBite optical input and output to test itself. To do that you apply an analog signal to J7, routing that to the optical output through the DSP, looping the optical J2 output to J1 through an optical cable, and routing that optical input through the DSP to all the remaining analog outputs (J4, J6 and J8). To do that:
   a. Set SW1 to 1111 1111 (1234 5678) and reboot by depressing SW1 (RESET). (See Table 2-2.)
   b. Apply a 1 kHz 2.4V peak-to-peak sin wave to J7.
   c. Connect an oscilloscope to any output (J4, J6 or J8).
   d. Connect J1 to J2 with an optical cable with appropriate adapters
   e. After the optical connection is made, the 1 kHz sin wave should appear with the same frequency and approximately the same amplitude at the output, but with some phase shift.

10) Test the analog inputs and outputs.

   To test the analog inputs and outputs, you apply a known signal to each input and look for the same signal on the corresponding output. To do that:

   a. Set SW1 to 0000 1000 (1234 5678) and reboot by depressing SW1 (RESET). (See Table 2-2.)
   b. Apply a 1 kHz 2.4V peak-to-peak sin wave successively to inputs J1, J3, J5, and J7.
   c. Connect an oscilloscope to the corresponding outputs (J2, J4, J6, and J8).

d. For each pair of corresponding jacks, the output should have a 1 kHz sine wave with approximately the same amplitude as the input, but with some phase shift.

If all of the tests (listed in steps 1-10) pass without any problems, the Symphony SoundBite board is considered to be fully functional.

Chapter 4 Building the Demonstration Application

The demonstration application source consists of a collection of assembly code and include files that are individually assembled and then linked together into an executable object for the DSP. The detailed function of the code within the source is beyond the scope of this document. For more information about the internal workings of the demonstration application, please refer to *Symphony SoundBite: Assembly Project Template* document.

This chapter details the procedure to build the demonstration application from the source. Some working knowledge of Symphony Studio is assumed.

4.1 Importing and Preparing the Project

The source of the demonstration application is included in the accompanying zip file archive within a compressed workspace file (Symphony Studio can directly import the source project from the compressed archive). To import the demonstration application source into a Symphony Studio project:

1) Launch Symphony Studio.
2) In the ***C/C++*** perspective, right-click in the ***Projects*** pane (or use the ***File*** menu to select ***Import***) and choose ***Import***.
3) In the pop-up dialog box, select ***General*** > ***Existing Projects into Workspace***. Click ***Next***.
4) Choose the ***Select archive file*** radio button, browse to the demonstration application workspace archive, and ***Open*** it.
5) Make sure that the check box next to the Symphony SoundBite board test project is checked.
6) Check "***Copy projects into workspace.***"
7) Click ***Finish***.
8) Right-click on the project and select ***Properties***.
9) Select ***C/C++ Build*** > ***Tool Settings***.
10) Select ***56k ASM Assembler*** > ***Options***. Verify that the ***Debug*** option is checked and that there is a "%" in the ***List File*** field.

    The checkbox tells the assembler to place symbolic information for debugging into the intermediate object files.

    The "%" tells the assembler to generate a listing file for each input source file (the make file assembles each .asm file separately and the %" is a placeholder for the filename being acted upon).
11) Select ***56k ASM Linker*** > ***Options.*** Verify that the ***Debug*** option is checked. In the ***Map File*** field, there should be a filename "***mapfile.txt,***" which is what was used in the original project (although this name is arbitrary). The text in the ***Memory Control File*** field should be "***..\sb_boardtest_lnk.ctl,***" which points to an existing file in the project folder.

    The checkbox directs the linker to include the symbolic debugging information in the object output file.

    The map file name specifies the filename to use for saving the map file (which contains textual information about the symbolics in the object file, where the labels are located and so on).

The memory control file contained within the project specifies where each section should be placed. This is important, so that there is no application code within the vector table, and for looking up the start address of the EEPROM programming routine.

12) Click ***OK*** to save the settings. Your project is now ready for building.

4.2 Running and Debugging the Application

It is assumed that an External Tool launch in Symphony Studio has been properly configured for use with the Symphony SoundBite. For more information, refer to *Symphony SoundBite with Symphony Studio: Quick Start*.

To configure a Debug launch for the newly built project:

1) Select the menu ***Run > Debug…***
2) On the left side, select ***Freescale 56371***. Click the ***New*** button or double-click the ***Freescale 56371*** to create a new debug configuration.
3) Enter a name for the debug configuration at the top.
4) If the board test application is not pre-filled in the ***Project*** field, choose the project by clicking the ***Browse*** button and selecting the appropriate project.
5) Select the executable object file (the ***C/C++ Application*** field) by clicking the ***Search Project*** button, choose the object file and click OK.
6) Verify that the Core Index is 0.
7) Verify that ***Download onto Target*** is checked.
8) Verify that ***Run at Startup*** and ***Stop on Startup*** are unchecked.
9) Click ***Apply*** to save the changes.
10) Click ***Close*** to save the debug configuration.

To download and execute the demonstration application:

1) Launch the External Tool configured for the Symphony SoundBite.
2) Launch the Debug configuration created above by selecting ***Run > Debug…*** and selecting it from the list under ***Freescale 56371***. Alternatively, if the configuration has previously been used, it may appear under the ***Run > Debug History*** menu.
3) Run or step through the application.

To program the on-board EEPROM so that the Symphony SoundBite will self-boot:
(Note that this is an alternative way to burn the EEPROM from the previously described method)

1) In the ***C/C++*** perspective, open the ***Debug*** folder within the application project by clicking the “+” next to it or by double-clicking it.
2) Double-click the file ***makefile.txt*** (or whatever filename for the map file was used in the project) to open it with the editor. Under the ***P: Memory*** heading, look for the section name ***soundbite_eeprogram***. Make note of the start address for this section.
3) Launch the External Tool configured for the Symphony SoundBite.

4) Launch the Debug configuration created above by selecting ***Run*** > ***Debug…*** and selecting it from the list under ***Freescale 56371***. Alternatively, if the configuration has previously been used, it may appear under the ***Run*** > ***Debug History*** menu.

5) In the ***Registers*** tab, expand the ***core-PCU*** register set. Change the ***PC*** register to the start address noted in Step 2, which is the start address of the EEPROM programming routine.

   Note that the default number of words to store in EEPROM is 0x450, so if the start address is greater than this, the programming routine must be modified and the application rebuilt. See the project file ***sb_eeprogram.asm*** and refer to the *Symphony SoundBite Assembly Project Template Project* document for more information.

6) Run the application. The LEDs should begin to flash and then stop flashing as described in section 3.1.

THIS PAGE INTENTIONALLY LEFT BLANK

THIS PAGE INTENTIONALLY LEFT BLANK