Chipsmall

Chipsmall Limited consists of a professional team with an average of over 10 year of expertise in the distribution of electronic components. Based in Hongkong, we have already established firm and mutual-benefit business relationships with customers from,Europe,America and south Asia,supplying obsolete and hard-to-find components to meet their specific needs.

With the principle of "Quality Parts,Customers Priority,Honest Operation,and Considerate Service",our business mainly focus on the distribution of electronic components. Line cards we deal with include Microchip,ALPS,ROHM,Xilinx,Pulse,ON,Everlight and Freescale. Main products comprise IC,Modules,Potentiometer,IC Socket,Relay,Connector.Our parts cover such applications as commercial,industrial, and automotives areas.

We are looking forward to setting up business relationship with you and hope to provide you with the best service and solution. Let us make a better world for our industry!

Contact us

ISD-VPE15100

User's Manual

ISD15100 Series

Multi-Message Record/Playback Devices

with Digital Audio Interface

Table of Contents

1. Quick Start

The VPE15100 software tool provides easy evaluation of the ISD15100 family and helps users create the ISD15100 binary image file. It serves three main purposes: (1) To create scenarios in a user-defined memory structure; (2) To ease the hassle of configuring the ISD15100 signal paths, volume controls, AGC, I2S…etc; (3) To provide complete evaluation of compression algorithms. The ISD15100 offers several compression algorithms at various bit rates and sampling frequencies to fit various applications' needs. The algorithms include Enhanced ADPCM, VBR (variable bit rate) ADPCM, uLaw, DuLaw (differential uLaw) and PCM.

The VPE15100 provides a powerful index-based Voice Prompt and Voice Macro Editor for script programming, which can be updated on the ISD15100 without the need of updating the code on the host microcontroller as no absolute addresses are used. The path view in the Configuration screen makes it easy to configure signal paths for Record from various inputs or Playback to various outputs of the ISD15100. The Reserved Message/User Data screen allows user to reserve memory area for future message recording or for digital data storage. The VPE15100 also generates binary image file (contains audio data and script programming), which can be burned into ISD15100 memory.

The VPE15100 is a GUI-based program run in Microsoft Windows® environments.

Getting Started

Your evaluation kit should contain the following:

- One CD for ISD15100 GUI Software set-up program and documentations
    - User login ID and Password (without it you cannot install the software)
- ISD15100 evaluation board (ISD-ES15100_USB).
    - With an 8-ohm speaker and a mini-USB cable.

Equipment requirements:
⇒ PC running Windows XP, NT, or 2000

1.1 Standalone Operations

The EV board can be operated standalone or controlled by VPE. To operate standalone, make sure the VPE project is properly downloaded into the board; the playback path setting should be done in users' PU macro (VM1). Attach the keypad onto HJ1 and insert 3 AAA batteries into the battery holder underneath the board. The USB controller will power-on and reset the ISD15108 in 3 seconds. Press the buttons to:

- ExeVM: power-up the on-board ISD15108 or send ExeVM command.
    - USB firmware checks if the ISD15108 is powered up. If not, send power-up command; if yes, send ExeVM command.
- STOP: send STOP command and clear index.
- 0 ~ 9: indexes for executing VM.
- VOLUP/VOLDN: increase/decrease volume.

For example, to execute VM 25, do the following:

- Press ExeVM. ISD15108 will be powered-up.
- Press 2. Users shall see '2' displayed on the 7-segment display.
- Press 5. Users shall see '5' displayed on the 7-segment display.
- Press ExeVM to execute VM 25.

3

1.2   Install VPE

Install VPE before connecting the EV board to PC.
After VPE is properly installed, plug in the EV board to PC; then users shall see a message popped up as Figure 1-1; select 'install from a list or specific location (Advanced)', and then click on 'Next'.

Figure 1-1 Install USB Driver – 1

USB driver is located in the VPE installation folder, which by default is "c:\Program Files\ISD-VPE15100\EmDriver\" as Figure 1-2; Click on 'OK'.

Figure 1-2 Install USB Driver – 2

Users shall then see the installing of USB driver in progress as Figure 1-3.
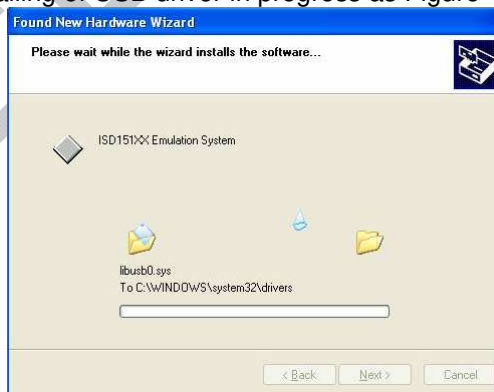

Figure 1-3 Install USB Driver – 3

After the installation of USB driver, reset the USB (either unplug and plug the board or press the RESET_USB button).

4

1.3   Browse the Example Project

Once VPE is properly installed, launch it.

Inside the CD there is an ISD15100 example projects. Do the following:

- Copy the project to PC.
- Open the project:
  - o Project -> Open; load the ISD15100example.prj.
- Browse each screens on VPE and you may gather some basic information about this project:
  - o The clock source is configured as 4.096MHz Crystal. This is to be consistent with the EV board.
  - o There are several voice prompts.
    - ▪ Users are encouraged to try different algorithms and sampling rates to experience the sound quality.
  - o Several empty messages are reserved.
  - o Several voice macros are composed.
    - ▪ In this example project, the voice macro 1 (PU) is composed as:
      - • Set clock source to 4.096MHz Crystal (0x27).
      - • Set a playback path to PWM.
      - • Play beep sounds.
- Go to the Memory Map screen, click on "Create Programming File" to generate the programming file. You will see a query as below. Click on Yes to save project.
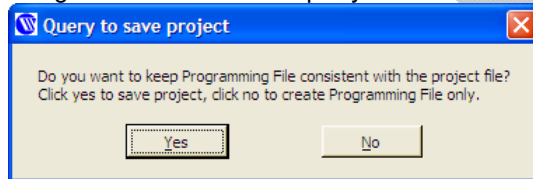
  o
- Whenever the EV board is connected to PC, the button "Burn Device" is effective. Click on it so that VPE will first chip-erases the ISD15100 and then downloads the programming file.
- Go to 'Audio Cmds' screen and try those voice prompts and voice macros.

5

2. Build the Example Project

An example project named ISD15100example is pre-programmed into the EV board; the example project can be found in the CD. This project does the following:

- When powered on (power supplied or a high pulse on RESET pin):
  - o Set clock source to 4MHz crystal,
  - o Set playback path to PWM,
  - o Play a beep sound,
  - o Power down.
- When powered up (received power-up command):
  - o Set clock source to 4MHz crystal,
  - o Set playback path to PWM,
  - o Play two beep sounds.
- A few more voice macros for quality demonstration.

This section introduces the procedure of making the example project.

2.1 Create a New Project

Launch the VPE15100. You shall see your username (A) on the subject bar; your username is part of the project when you create the programming file. In the menu bar, click on 'Project' (B) -> 'New', you shall see a pop-up window as Figure 2-2. Type in the desired project name (C) and assign a project location (D). Click on 'OK' (E), and you shall see the project path shown on the subject bar (F). In the assigned location you shall see that a project folder has been created.
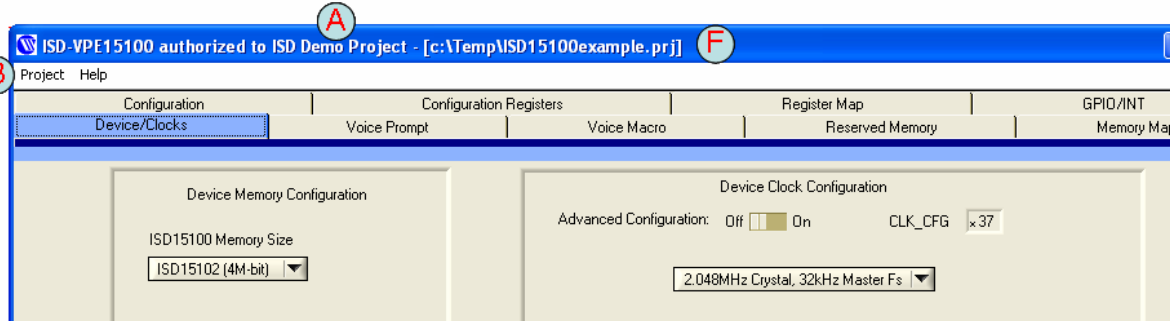

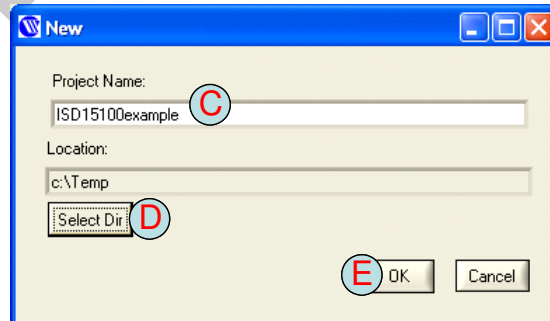**Figure 2-1 Username & Project path on the Subject Bar**

Figure 2-2 Create a New Project

There are three subfolders in the project folder:

6

- BinFiles: where the compressed VPs are stored.
- WavFiles: where the re-sampled and de-compressed VPs are stored.
- WavSrc: where wave sources of VPs are stored.

2.2   Device/Clocks

Users choose the device (A) and clock source (B) on the 'Device/Clocks' screen. The Memory Size selection (A) is for VPE to calculate and show the memory usage (C). If the EV board is present, VPE automatically sets the device (A) to reflect the hardware.

VPE provides several most common clock settings. For every clock setting, seven sub-sampling rates are available for VP compression as shown in Table 2-1. Clock configuration is part of the VP compression, so whenever users change the clock configuration, they have to 'Re-compress All' VPs (button on 'Voice Prompt' screen). The one-byte CLK_CFG (D) is for clock setting in voice macro. If the clock settings that VPE provides do not fit the users' needs, users may switch to the advanced view (E).

Please note that the clock setting (B) here is ONLY for VP compression; it is for making a project. Users also need to set the clock for the ISD chip by either setting it in their POI/PU macros or by sending SPI commands after the chip powers up.
The default clock setting for ISD chip when POI/PU:
- If the POI/PU macro is empty:
  - Clock sets to 0x34, which is internal oscillator with internal resistor.
- If the POI/PU macro exists, but users do not set clock in it:
  - Clock sets to 0x00, which bypasses the PLL.

In the example project, we choose ISD15102 (A) and set the clock to 4.096 MHz Crystal (B).

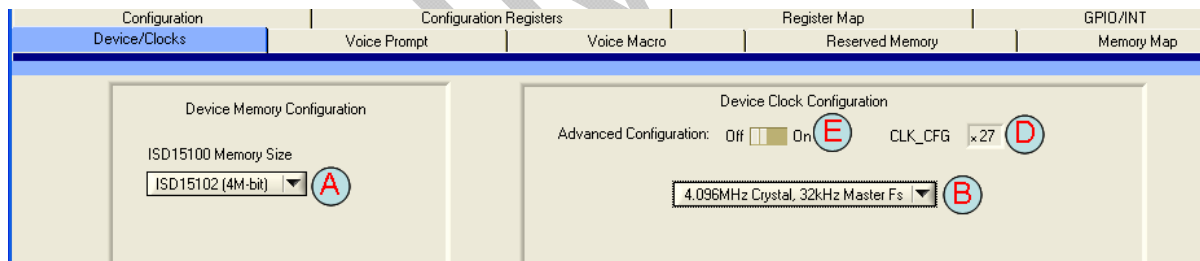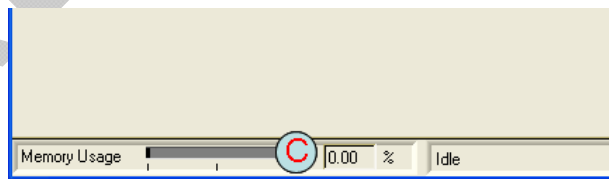**Figure 2-3 Device/Clocks Setting**

Figure 2-4 Memory Usage Indicator

Sub-Sampled ratio	Master Sample Rate Fs (kHz)			
	32	44.1	48	Available Sample Rates
8	4	5.5125	6	
6	5.333	7.35	8	

7

5	6.4	8.82	9.6
4	8	11.025	12
2.5	12.8	17.64	19.2
2	16	22.05	24
1	32	44.1	48

Table 2-1 Sub-sampling Rates

2.2.1 Advanced Clock Configuration

The Clock Source (A) includes four clock sources:

- Internal oscillator,
- Internal oscillator with external resistor,
- I2S clock, and
- Crystal.

The selected clock will feed to the phase-lock-loop (PLL) (B), for generating the internal master clock (MCLK). Refer to Table 2-2 for PLL M & N settings for a required master sampling rate. For every clock setting, there are seven available sub-sampling rates for VP compression (C).

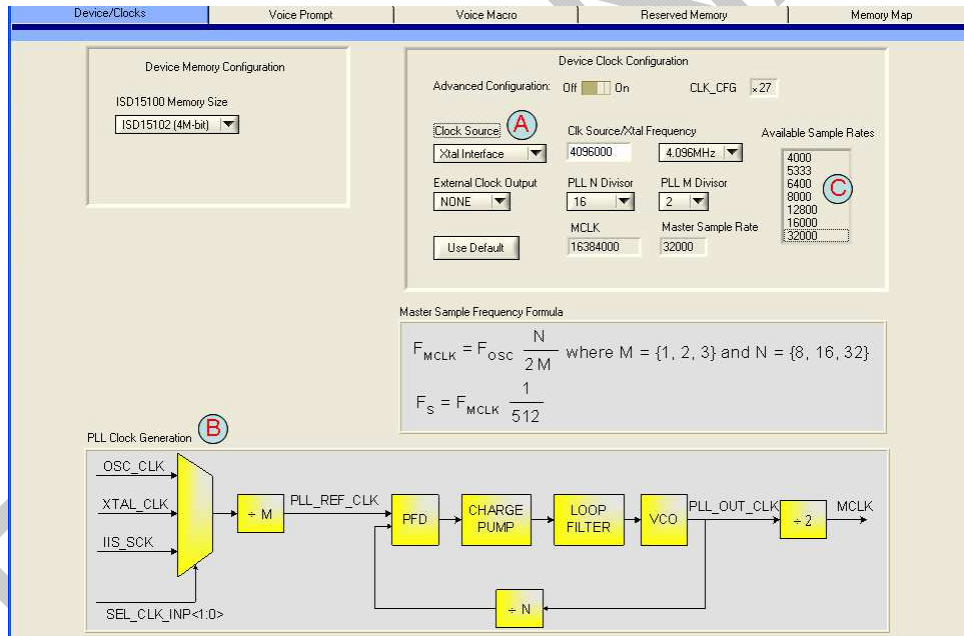

Figure 2-5 Advanced Clock Configuration

Xtal (MHz)	M	N	Fs (kHz)
1.024	1	32	32
2.048	2	32	32
4.096	2	16	32
8.192	2	8	32
12.288	3	8	32
1.4112	1	32	44.1

8

2.8224	2	32	44.1
11.2896	2	8	44.1
3.072	2	32	48
12.288	2	8	48

Table 2-2 PLL M & N Settings

If users need really high quality, they might want to try 44.1 kHz sampling rate. According to Table 2-2, users can use an 11.2896 MHz crystal and set the PLL as M = 2, N = 8. In this case users also have to change the crystal on the EV board.

If the input Clock Source is set to internal Oscillator, the clock rate is fixed at 2.048 MHz. If the input clock source is set to internal Oscillator with external Resistor, the clock rate can vary with resistor values. User can type in different resistor values to see the calculated clock rate. User can change the N and M of the PLL to generate the desired master clock (MCLK) and thus the available sampling rates.

2.3   Voice Prompt

Voice prompts are pre-recorded audio of any length, from short words, phrases or sound effects to long passages of music. These Voice Prompts can be played back in any order as determined by the users and applications. A voice prompt consists of two components:
  (1) An index pointing to the pre-recorded audio.
  (2) Pre-recorded audio.

Before adding voice prompts, first set the default sampling rate (A) and compression algorithm (B).

To add voice prompts, follow the steps below:
  (1) Set sampling rate (C) and compression algorithm (D) for the VPs you are going to add. Users can change the individual VP's sampling rate (E) and/or compression algorithm (F) later when evaluating the quality.
  (2) Add voice prompts (G).
  (3) Repeat steps (1) & (2).
  (4) Change the VP tags (H) if needed; by default, the tags are the filenames.
  (5) Re-order the VPs if needed; select VP by clicking on the square (I), and click on the up/down arrows (J) to re-order the VPs.

If users change the clock configuration on 'Device/Clocks' screen, they must re-compress all VPs (K) because clock setting is part of the VP compression.

Users can evaluate the VPs through PC or EV board:
  • Through PC (L):
    i.   Original voice prompts before re-sampling & compression.
    ii. Re-sampled but not yet compressed voice prompts.
    iii. Compressed voice prompts.
  • Through EV board:
    i.   Plug in the EV board and the 'Play to device' button (M) shows up. Switch to 'Configuration' screen to set a playback path and come back to click on the 'Play to device' button (M).

9

1. This is accomplished by the ISD15100 SPI command: SPI_SND_DEC.

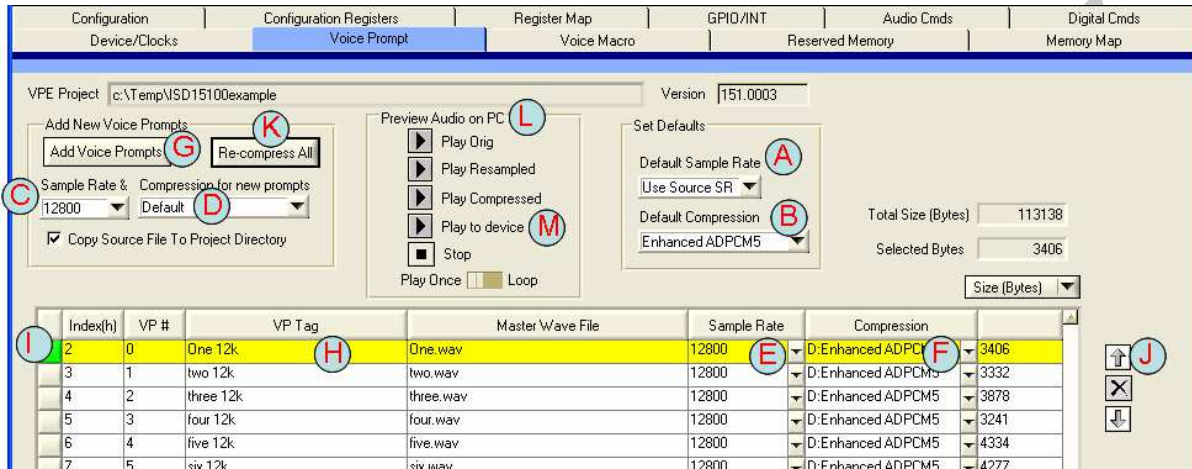The added voice prompts will appear on the 'Voice Macro' screen for voice macro script programming.

Figure 2-6 Voice Prompt screen

2.4 Reserved Memory

Recording Memory Pointer (RMP) divides the ISD15100 memory space into two blocks:

- Reserved Memory,
- Recording Memory.

RMP is a two-byte address pointer pointing to a 4kbyte memory sector which is the first sector available for message recordings. Memory between address zero and the RMP pointer is considered the Reserved Memory for pre-recorded audio (Voice Prompts), pre-programmed macro scripts (Voice Macros), digital read/write access for other applications (User Data), and memory sectors reserved for the first sector of the reserved messages (Reserved Sectors). The memory between the RMP and the end of memory is considered the Recording Memory allocated for recording messages (Message Recordings).
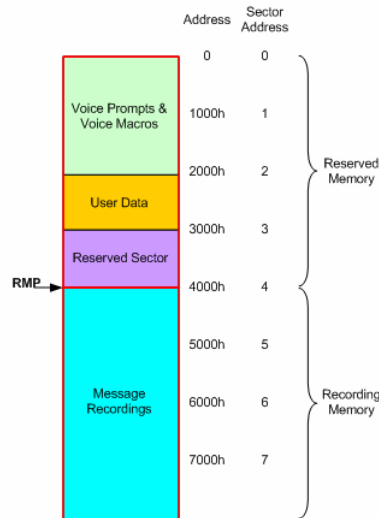
Figure 2-7 Example of ISD15100 Memory Map

On the 'Reserved Memory' screen, users reserve memory sectors for message recordings:
- Empty messages (A):
  - An Empty Message is an empty 4kByte memory sector reserved for message recordings. An SPI record command pointing to the Empty Message starts the message recording and will continue to record to the free memory sectors located in Recording Memory if the message recording is over 4kByte.
- Re-recordable messages (B):
  - Users can have a default message when reserving the memory sectors for a message recording. For a re-recordable message, the first sector will be allocated in 'Reserved Sectors' (refer to Figure 2-7); the rest of the message will be allocated in 'Recording Memory'.

Please note only ADPCM, (D)uLaw, and PCM compressions are supported for reserved messages, not Enhanced-ADPCM or VBR.

To record (or playback) a message, the microcontroller simply issues a record (or playback) message command with starting address pointing to the reserved initial address. VPE operations of reserving re-recordable messages are similar to those of voice prompts. The reserved messages are also listed on the 'Voice Macro' screen for voice macro script programming.

Memory sectors can also be assigned in Reserved Memory for digital read/write access for other application tasks (User Data). To reserve empty memory sectors, click the button 'Add Reserved Memory' (C) and then adjust the 'Number of Sectors' (D). To reserve (import) existing data, click the button 'Add Data' (E); the file size will be shown on 'number of sectors' (F).
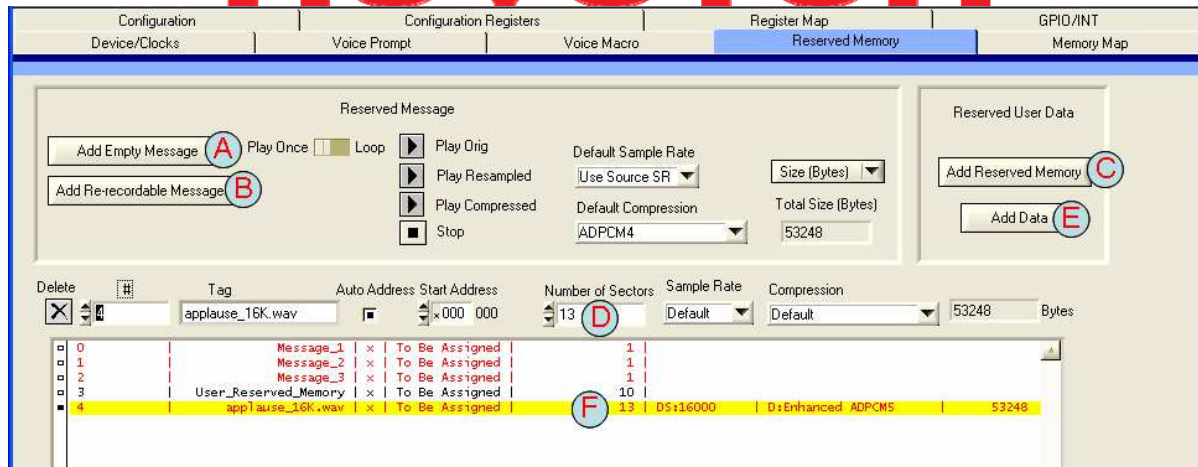
11

Figure 2-8 Reserved Memory

2.5   Voice Macro

Voice Macro is a powerful scripting feature integrated into the ISD15100. Voice Macro editor enables users to create scripted commands to play voice prompts (**A**), play reserved messages (**B**), insert silence (**C**), set clock configuration (**D**), power down the ISD15100 (**E**), and configure the ISD15100 (**F**) such as setting play or record paths. As mentioned earlier, the voice prompts (**A**) are from the 'Voice Prompt' screen, and the reserved messages (**B**) are from the 'Reserved Memory' screen, respectively. Each VM is assigned with a tag and an index, where it can be executed with a single SPI command referring to the index.

There are two special Voice Macros:
- Index 0: VM0, Power-On Initialization (POI).
    - Executed automatically when power supplied or when a high pulse applied on RESET pin.
- Index 1: VM1, Power-Up (PU).
    - Executed automatically when received power-up command.

POI and PU Macros allow users to customize how the part emerges from power-on or power-up conditions.

There are two special VM commands: FINISH (**G**) and WAIT INT (**H**).
- FINISH (**G**):
    - Finish the voice macro and exit. Voice Macros must be ended in either FINISH or PD (power down).
- WAIT INT (**H**):
    - Wait until current play command finishes before executing next macro instruction.

The ISD15100 has a built-in command buffer so that consecutive commands can be executed one after another seamlessly. However, some commands won't go through command buffer; therefore, they won't wait. Users may refer to the design guide; section 13 SPI Commands, Table 13-2, for details. In table 13-2 users may see that some commands can be executed regardless the status bit-1 CBUF_FULL (command buffer full). WAIT_INT (**H**) is used whenever a command in a voice macro intends to wait to be executed until previous play command finishes. A typical example will be

12

adjusting volume in a voice macro. Commands for adjusting volume won't go through the command buffer; therefore, they will be executed immediately. Below is an example where the volume is immediately decreased when VP_A starts to play.

- Play VP_A
- Decrease Volume
- Play VP_B
- Finish.

If users want to decrease volume after VP_A finishes, the voice macro should be modified as below:

- Play VP_A
- WAIT_INT
- Decrease Volume
- Play VP_B
- Finish.

Please note that the commands in a Voice Macro are executed sequentially and all voice macros shall be ended by PD or FINISH to finish the voice macro. The VPE15100 will show up a warning message if it detects a voice macro not ended by PD or FINISH.

The example project has some VMs for demo purpose:

- VM0 (POI macro):
  - o Set clock source to 4MHz crystal,
  - o Set playback path to PWM,
  - o Play a beep sound,
  - o Power down.
- VM1 (PU macro):
  - o Set clock source to 4MHz crystal,
  - o Set playback path to PWM,
  - o Play two beep sounds.
- A few more voice macros for quality demonstration.

Below are the steps of making the POI macro:

1. Click on 'Set CLK CFG' (**D**) to set clock source to 4MHz crystal (0x27).
2. Set CFG17 to 0x00 (**F**) so that registers won't be updated immediately.
3. Switch to 'Configuration' screen; we are going to set a playback path to PWM.
4. Click on '007 PLAYBACK MEMORY-> PWM' (**I**); a green line for the path will light up.
5. Click on 'Sub-group selection' (**J**); leave 'Audio Config' and 'UPDATE' checked; uncheck others.
6. Switch back to 'Voice Macro' screen; click on 'Set All checked group' (**K**).
7. Click on the voice prompt 'FastBeep' (**A**).
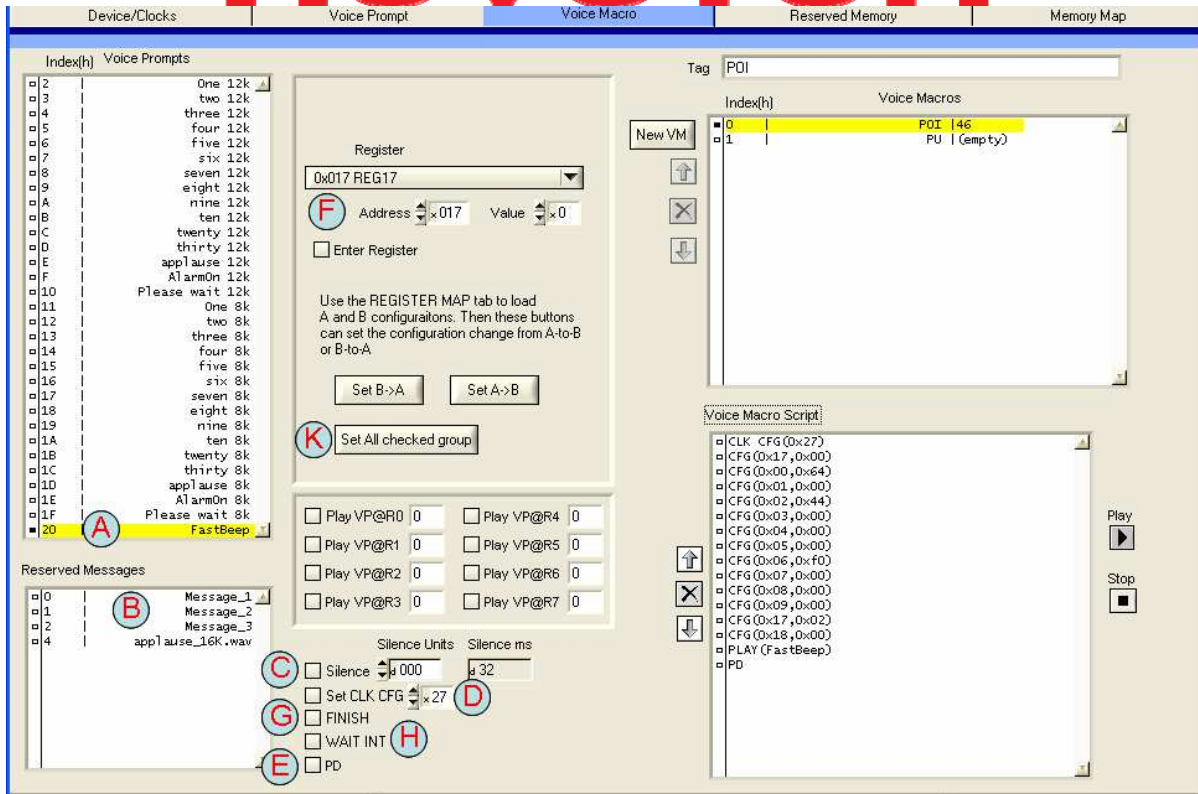8. Click on PD (**E**).

Other VMs are done in similar ways.

13

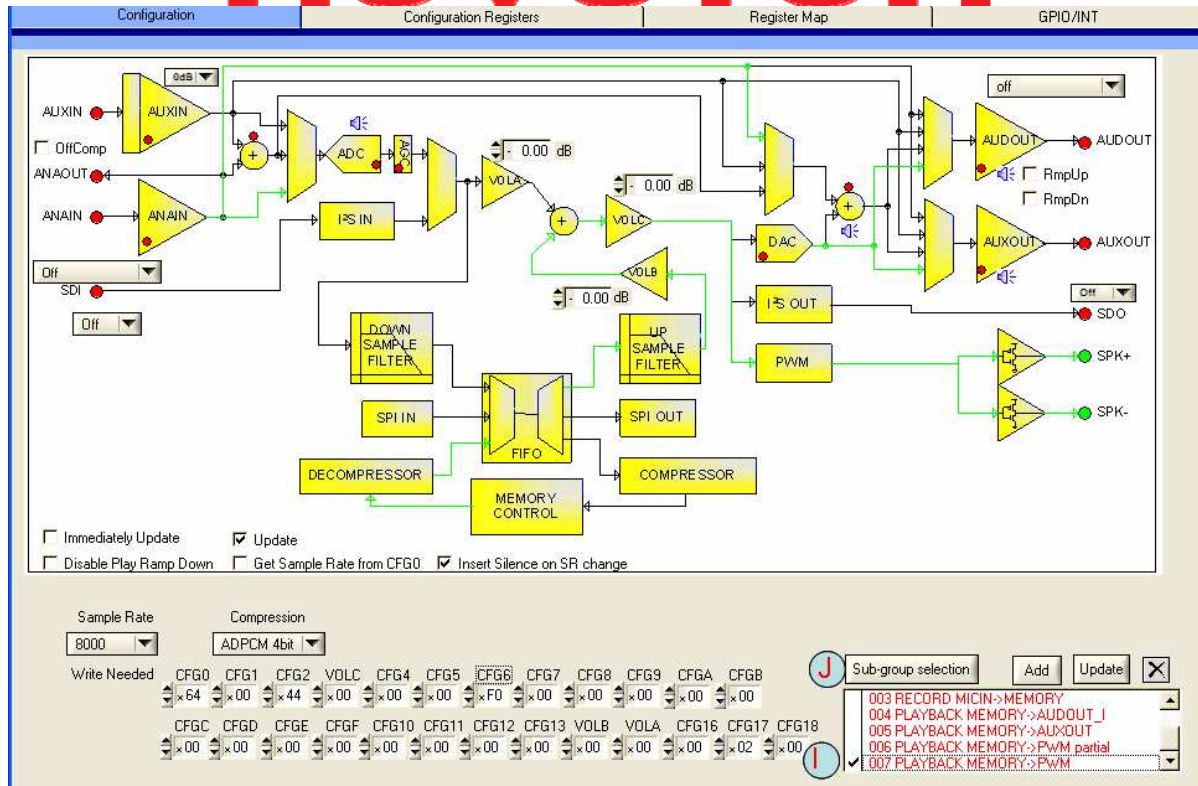**Figure 2-9 Voice Macro Editor**

14

Figure 2-10 Set a Playback Path to PWM

2.6   Configuration

The Configuration screen aims to ease the task of configuring the ISD15100. To set the configuration registers, microcontroller could either send several 'write config register' commands or simply issue a Voice Macro command which consists of a group of 'write config register' commands.

In this screen, users can set a path for record or playback and generate automatic configuration bits that can be ported to the VM editor. Once you set up a path, you can save it by clicking on the 'Add' button (**A**) and assigning a name for the path such as 'Play from Memory to Speaker', or 'REC from ANAIN'. This is useful when you are editing Voice Macro groups.

There are several default configuration paths; users cannot delete these configurations. The Update (**B**) is used to update an existing path (user settings). When selecting a default path by clicking on a configuration path, you can change the path on the diagram and assign a new name to it.

Please note that the CFG17 bit 0 (Immediately Update, **C**) and bit 1 (Update, **D**) are displayed on the path view as these two bits decide how the configuration registers are updated.
- Immediately Update:
  - This bit set to one turns off double buffering of configuration. Thus configuration is updated after each register byte is written. When this bit is set to zero, the configuration register setting is not made active until a one is written to the 'Update' bit.

15

- Update:
  - When a one is written to this bit and 'Immediately Update' is zero, configuration CFG0-CFG17 become active. In this way a path can be set up and applied at the same instant to prevent disturbance on the analog path. (Write Only)
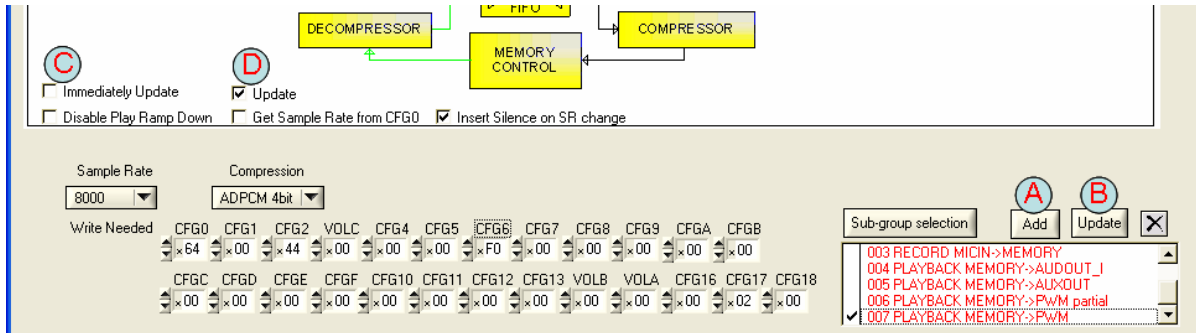

Figure 2-11 Configuration

2.7 Memory Map

Memory Map is the last step to generate the programming file to be burned into the ISD15100 memory. The programming file is a binary file with mem extension. In this screen users will see how each VP, VM, and message is assigned an address and/or an index. To activate the process, click on 'Create Programming File' (**A**). On the left hand side the memory map (**B**) is shown.

On the right hand side, the Memory Protection (**C**) decides how you want the memory contents protected. ISD15100 provides three kinds of protection mechanisms:
- Read Protection:
  - Once enabled, digital-read command is not effective in those protected area.
- Write Protection:
  - Once enabled, digital-write and memory-erase commands are not effective in those protected area.
- Chip-Erase Protection:
  - Once enabled, chip-erase command is not effective.

If both Chip Erase Protect (CEP) and Write Protect (WP) are checked, it will lock the content from beginning of the memory to PMP, preventing ISD15100 from any further modification.

The check boxes under 'Project Information' (**D**) allow users to add project information to the programming file. If you check the users comment box, whatever you type in the comment window, it will be appended to the end of the project information.

If the EV board is connected to PC, the 'Burn Device' button (**E**) will be effective. Once the programming file is created, users can click on 'Burn Device' so that VPE will first chip-erase the ISD15100 and then download the programming file into the ISD15100.

Please note that 'Date' (**F**) is part of the project, so clicking on the 'Create Programming File' (**A**) may change the project contents even though users don't change anything. When users open an existing project, the memory map will be loaded automatically.
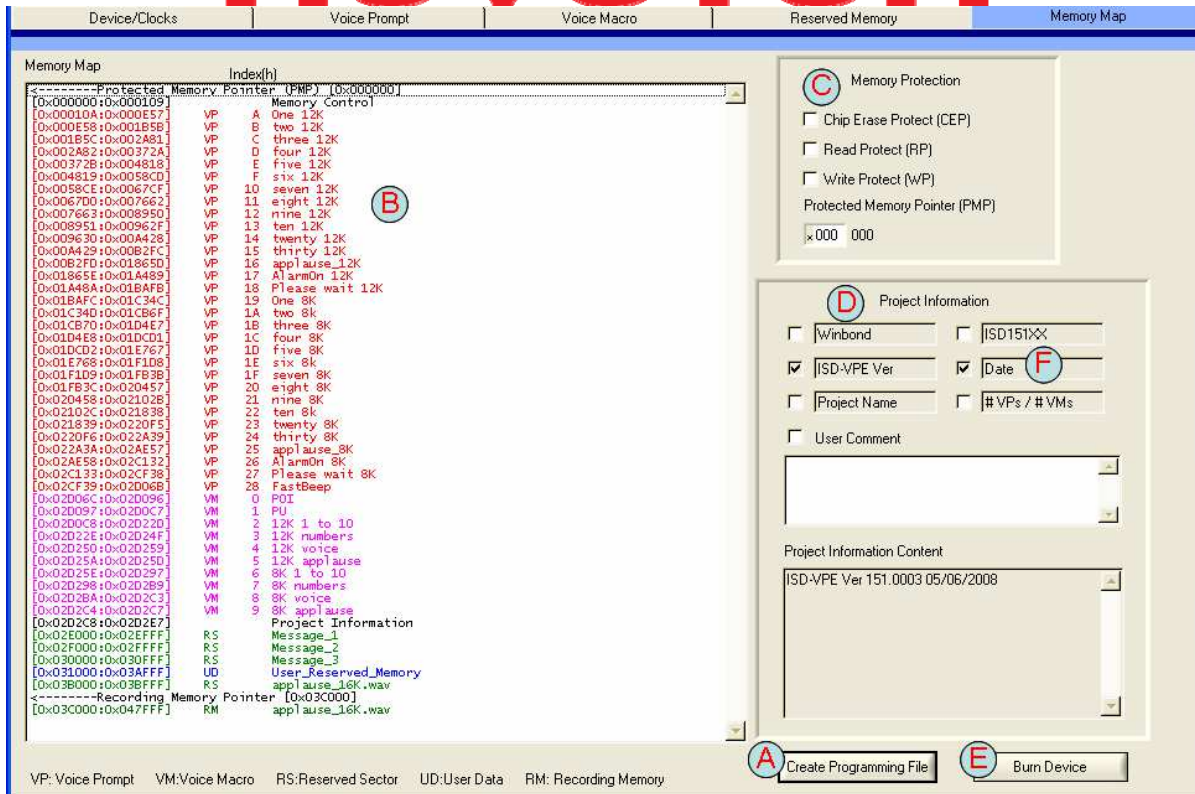
16

Figure 2-12 Memory Map

2.8 Archive and Clean

Under project menu there are two very useful functions.

1. Archive and,
2. Clean

These two functions are used to backup your project files and clean up unnecessary files. They are useful when you want to send an ISD15100 project to other people.

The archiving function duplicates a project to an archive directory with exactly same project name and structure, and changes the project links to point to this directory so that all paths are stored relative to the project directory. For example, if you create an ISD15100 project in a network share folder which is, say, mapped to T drive, you cannot just send this project to other people because "T" drive means nothing to other people. By archiving the project, that project is now self contained and can just be copied to another machine or to backup.
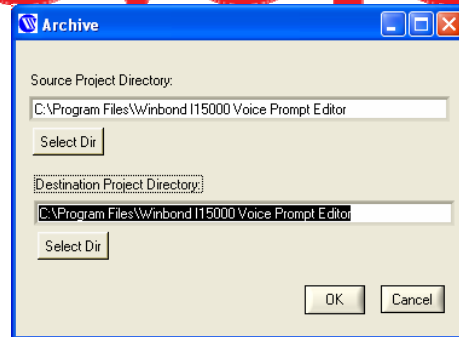
Figure 2-13 Archive

The Clean function deletes the contents in the BinFile and WavFile directories as they can be regenerated. The WavFile directory contains wave files of re-sampled and de-compressed voice prompts. The BinFile directory contains files of compressed voice prompts.

When a project is created, three sub-directories are created:
- BinFiles: where the compressed VPs are stored.
- WavFiles: where the re-sampled and de-compressed VPs are stored.
- WavSrc: where wave source of VPs are stored.

A project file has the prj extension. The binary programming file for the project has mem extension. A header file with txt extension is also included for C language firmware programming.
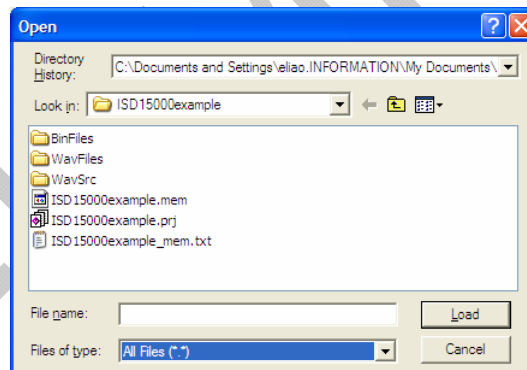

Figure 2-14 Open a Project

18

3. Evaluation System

3.1 EV Board Overview

Figure 3-2 below is the ISD15100 EV board. On the right hand side is the ISD15U110 (**A**), our ISD USB1.1 controller especially designed for ChipCorder ISD15100 family. In the middle is the ISD15108 (**B**), 8-minute duration based on 8 kHz & 4-bit ADPCM. On the left hand side are analog circuitry and connectors for recording and playback. Upper left corner is a 7-segment display (**C**) controlled by four GPIO pins of ISD15108 (**B**). Upper right corner is a hole-array (**D**) for users' prototyping.

The EV board can be operated standalone or controlled by VPE. To operate standalone, make sure the VPE project is properly downloaded into the board; the playback path setting should be done in users' PU macro (VM1). Attach the keypad onto HJ1 (**E**) and insert 3 AAA batteries into the battery holder underneath the board. The USB controller will power-on and reset the ISD15108 in 3 seconds. Press the buttons to:

- ExeVM: power-up the on-board ISD15108 or send ExeVM command.
  - o USB firmware checks if the ISD15108 is powered up. If not, send power-up command; if yes, send ExeVM command.
- STOP: send STOP command and clear index.
- 0 ~ 9: indexes for executing VM.
- VOLUP/VOLDN: increase/decrease volume.

For example, to execute VM 25, do the following:

- Press ExeVM. ISD15108 will be powered-up.
- Press 2. Users shall see '2' displayed on the 7-segment display (**C**).
- Press 5. Users shall see '5' displayed on the 7-segment display (**C**).
- Press ExeVM to execute VM 25.

Figure 3-1 Keypad for Standalone

Around the ISD15108 (**B**) are four pin-headers (**F**) used for adaptation of daughter cards. To attach any daughter card onto the EV board:

- Make sure the on-board ISD15108 has been chip-erased.
- Remove jumpers J17 SSB (**G**) and J21 RDY_BSYB (**H**).
- Attach the daughter card via the four pin-headers (**F**).

On board there are a 4MHz crystal and an 80k-ohm resistor for clock setting (**I**). Users use jumper J12 to select the clock source:

- Internal oscillator with internal resistor:
  - o It doesn't matter how the jumper is setting.

19

- Internal oscillator with external resistor:
  - o Put jumper J12 on 2-3.
  - o Users can change the external resistor for different sampling rate.
- Crystal:
  - o Put jumper J12 on 1-2.
  - o Users can change the crystal for different sampling rate.

On the left hand side are analog circuitry and connectors for recording and playback:
- AUDOUT (**J**): Can be configured as either current-type or voltage-type single-ended output.
  - o When configured as voltage-type output, it is exactly the same as AUXOUT.
- AUXOUT (**K**): Voltage-type single-ended output.
  - o Users need an external amplifier to drive the speaker.
- MICIN/ANAIN (**L**): Can be configured as either differential or single-ended input.
  - o Install jumpers (**M**) 3-4, 7-8, and 11-12 for differential input (MICIN).
  - o Install jumpers (**M**) 1-2, 5-6, and 9-10 for single-ended input (ANAIN).
  - o Please note that we use a general I/O pin to "physically" enable/disable the microphone. Later when users hook up their own microcontroller to command the ISD15100, they can either:
    - ▪ Manually enable the microphone by installing the jumper on J8 (this will increase standby current), or
    - ▪ Use an I/O pin to "physically" enable/disable the microphone as the EV board does.
- AUXIN (**N**): Voltage-type single-ended input.

PWM (**O**) is a direct speaker drive. In ISD15100, it can also be configured as:
- Voltage-type differential output (BTL).
  - o BTL can directly drive the speaker, but then the volume level won't be as loud as PWM. Users can attach an external amplifier for a loud volume.
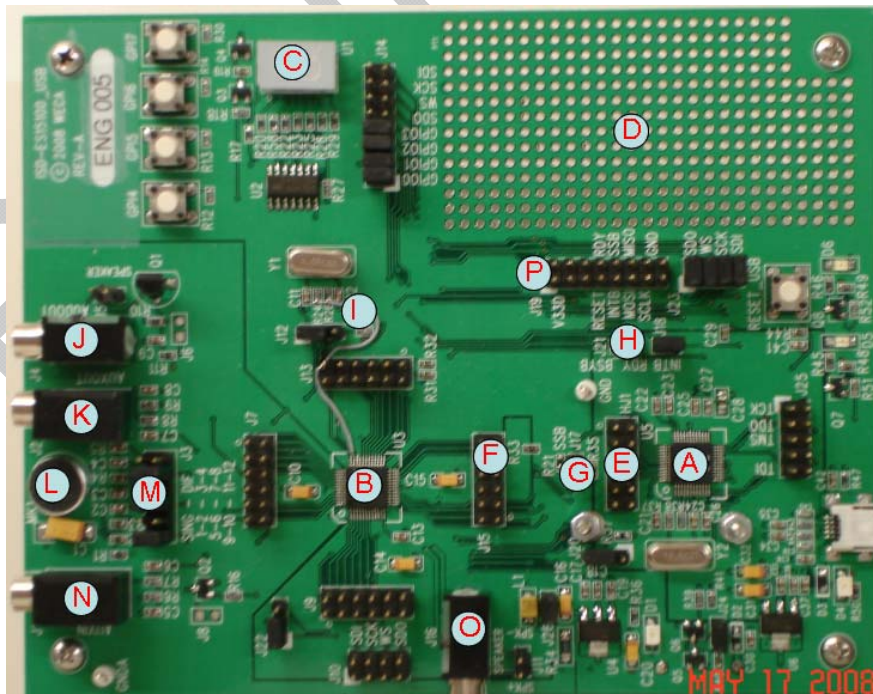- Separately configure SPK+/SPK- as AUDOUT/AUXOUT.

Figure 3-2 ISD-ES15100_USB

Below is a table for the ISD1510 evaluation board configuration.

Jumper Name	Function	Default
J17, J21	• Take off J17 & J21 when daughter card is used. o R21 must be removed. • Install J17 & J21 when daughter card is NOT used.	J17 & J21 are installed to de-select the daughter card.
J12	• Install jumper on 1-2 to select crystal as the clock source. • Install jumper on 2-3 to select the external resistor as the clock source.	Jumper is installed on 1-2 to select crystal as the clock source.
J3	• Install jumpers on 1-2, 5-6, and 9-10 to configure as ANAIN (single-ended). • Install jumpers on 3-4, 7-8, and 11-12 to configure as MICIN (differential).	Configured as MICIN (differential input).
J10	• I2S Connector (SDI,SCK,WS, SDO)	

Table 3-1 EV Board Configuration

3.1.1 Hook up uC to the EV Board

As users get familiar with the ISD15100, they may want to use their own microcontroller instead of the ISD15U110. Users may hook up their microcontroller to the EV board by following the rules below:
- Power EV board by USB, but Do NOT launch VPE:
  - o Users can power the EV board by USB; however, whenever VPE is launch, by default the VPE will read back the ISD15100 status every second. This causes SPI bus confliction if users use their own microcontroller to command ISD15100 at the same time. As long as VPE is not launched, ISD15U110 won't do anything.
- Need at least 5 connections:
  - o Users need at least the four SPI signals plus a common ground. If users intend to implement digital-read or digital-write commands, they need to poll RDY/BSYB pin for data flow control as well. All controlling signals are available at J19 (**P**).
- Start with Read_Status:
  - o Users should start with Read_Status because this command is always needed for debugging.

3.2 Programming the ISD151xx

There are three programmers available for programming the ISD15100:
- ISD-ES15100_PROG:
  - o Single-socket daughter card as shown in Figure 3-3, which can be attached to the ISD15100 EV board.
- Hi-Lo Systems All-100 Gang-Programmer:
  - o 4-socket gang-programmer http://www.hilosystems.com.tw/.
- Prospect IT2000:
  - o 8-socket gang-programmer (under development) http://www.prospect.com.tw/.

Figure 3-3 ISD-ES15100_PROG

4. USB Emulation

When the ISD15100 EV board is connected to PC, two extra screens show up:
- Audio Cmds:
    - Where you can emulate the ISD15100 play and record commands like executing voice macros, erasing recorded messages etc.
- Digital Cmds:
    - Where you can emulate the ISD15100 digital commands like downloading a project, reading back for verification etc.

4.1 Audio Cmds

ISD15100 provides two message recording mechanisms:
- 'Record Msg' (**A**):
    - Record a message without specifying address. ISD15100 automatically searches for available memory sectors starting from the RMP (Recording Memory Pointer) for the message recordings. When a 'Stop' command is issued to finish the recording, or the recording is stopped due to memory full, VPE automatically reads back the recorded message address and length.
- 'Record Msg At' (**B**):
    - Record a message at the user-specified address, which is reserved in advance. By default the reserved messages are allocated before the RMP. Since the addresses of Reserved Messages are allocated in advance, there is no need to read back the recorded message address. Reserved Message is especially useful for a predefined scenario.

The 'Play Msg At' command (**C**) plays a message at the user-specified address. Users can play a message from the middle by specifying an offset (**D**). One offset is one memory sector, which is about one second based on 8kHz 4-bit ADPCM. ISD15100 does not provide a PAUSE command; however, a PAUSE command can be easily done by toggling between 'Stop' and 'Play Msg At' with an offset.

To erase a message, simply specify the address and then click on 'Erase Msg At' (**E**). Addresses of recorded messages are shown on the Message List. (**F**)

'Get Msg List' (**G**) is not an ISD15100 command. It is indeed a bunch of digital-read commands used to scan the memory for messages. The first byte of each memory sector shows whether or not a beginning of message is inside the memory sector.

VM (**H**) & VP (**I**) sub-screens show users the VM/VP tags and indexes that they are working on. It shows up only if the loaded project on VPE is consistent with the project inside the EV board. When the ISD15100 is powered up, VPE automatically reads back checksum from the EV board and compares it with the loaded project. If the two checksums match, the indexes and tags show up. The checksum incorporates the projects details, date of creation, VPE version, and user ID.

To execute a VM or to play a VP, simply click on the VM/VP sub-screens to get the index, and then click on 'Execute VM' (**J**) or 'Play VP' (**K**).

VM and VP are index based, which means to execute a VM or to play a VP the microcontroller only sends a simple command followed by the index. It is address-free! Please note that VM and VP share the indexes, and it always starts from VM. For example, if a project has three VMs and four VPs, then the indexes 0 ~ 2 belong to VMs, and the indexes 3 ~ 6 belong to VPs.

23

'Indirect Voice Prompts' (**L**) is to make voice macro flexible. For example, it could be used to announce time and temperature in users' system. To announce a sentence like '***It is 7:15 am; the outside temperature is 70 degree Fahrenheit***', do the following:

- Microcontroller gets the time (***7:15 am***) and temperature (***70*** degree Fahrenheit) from the system's timer and sensor.
- Write the corresponding voice prompt indexes of ***7, 15, am***, and ***70*** into R0, R1, R2, and R3.
- Execute the voice macro of that sentence.

To emulate it on VPE, click on VP sub-screen (**I**) to get the index, then right-click on Rn, n = 0 ~ 7, to update Rn, then click on 'Play VP@Rn' (**L**).

The volume control bar (**M**) provides an easy way to adjust the volume while playing a VP or a message. ISD15100 provides three volume controls:

- VOLA: volume control for the digital audio data from I2S or analog inputs.
- VOLB: volume control for the digital audio data from decompression block or SPI.
- VOLC: master volume control for PWM, AUDOUT, AUXOUT and I2S outputs.

Resolution of the three volume controls is 0.25dB per step, total 256 steps for each volume control. The volume control bar here is the VOLC. What VPE does is keep writing the CFG3 when the users drag the volume control bar.

'MICIN/ANAIN Enabled' (**N**) indicates that the microphone path is enabled. When the microphone path is enabled (either configured as differential or single-ended), the indicator turns red.

ISD15100 provides four 'SPI Data Transfer' commands. For details, please refer to design guide. Take 'SPI Decode' (**O**) as an example, microcontroller sends compressed audio data (VPs or recorded messages) to ISD15100 to decode. Double-click on 'PLAYBACK MEMORY->PWM' (**P**) to set a playback path for 'SPI Decode'. Check 'CTS SPI' (**Q**) if users want a continuous SPI transaction, which means VPE ignores all other tasks before the 'SPI Data Transfer' command finishes. Right click on the Filename blank (**R**) to assign a compressed audio file, and then click on 'SPI Decode' (**O**).
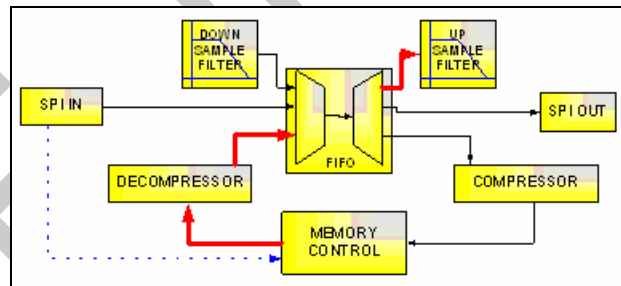

Figure 4-1 SPI Send Compressed Data to Decode

Users can try different clock settings (**S**). Please note that clock setting must match the real configuration on board or the ISD15100 won't work. Below is a table of some common settings:

0x34	Internal oscillator with internal resistor
0x36	Internal oscillator with 80kohm external resistor
0x37	2.048 MHz crystal
0x27	4.049 MHz crystal
0x17	11.2896 MHz crystal

Table 4-1 Some Common Clock Settings

'Read ID' (**T**) reads back the four-byte ISD15100 ID.

24

'Read Status' and 'Read INT' (**U**) both read back the two-byte ISD15100 status. The only difference is that 'Read INT' clears the pending interrupt signal as well. ISD15100 will not generate a new interrupt signal if there is a pending one.

'Saved Configurations' (**P**) is a copy from the 'Configuration' screen. Double-click on a setting to write to the ISD15100.
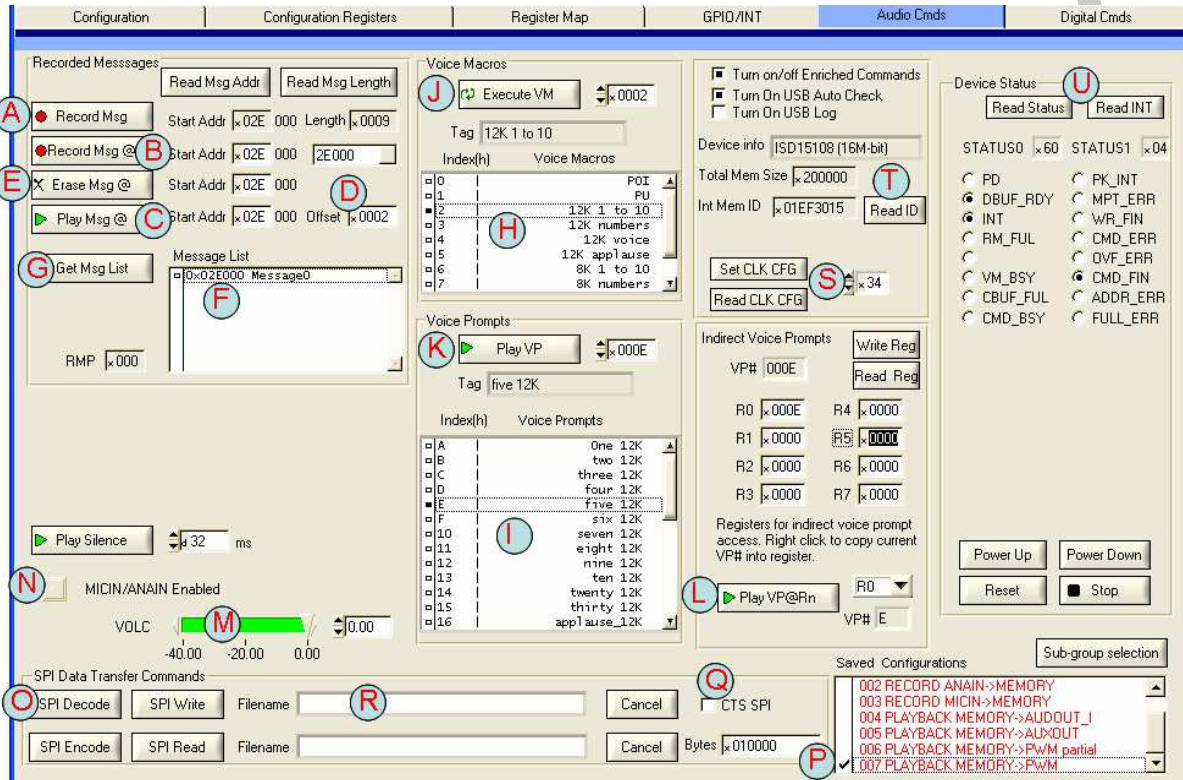

Figure 4-2 Audio Cmds

4.2   Digital Cmds

ISD15100 provides four digital commands: digital-read (**A**), digital-write (**B**), erase-memory (**C**), and chip-erase (**D**).

To erase the whole memory, click on 'Chip Erase' (**D**); while the erasing is in progress, status bit CMD_BSY (**E**) goes high. Time for erasing the whole chip depends on the memory density. Below is a table for reference.

Device	Memory Density	Typical Erasing Time
ISD15102	4Mbit	5 sec
ISD15104	8Mbit	10 sec
ISD15108	16Mbit	25 sec
ISD15116	32Mbit	40 sec
ISD15132	64Mbit	80 sec

25