This document is created for the purpose of ensuring the correct usage of the ISD.DLL.

1. What is the ISD API (ISD.DLL)?

ISD API, delivered as ISD.DLL file, can be used to dynamically create a digital ChipCorder application image according to a user-defined configuration file. It gives the customer application the capability of generating customized sound effect file on the fly and therefore makes it possible to change the application sound effect in field.

The ISD API version 1.0 can create .mem image file usable for digital ChipCorder devices including ISD15100/3900/15C00, ISD15D00/3800, and ISD2100 series. It supports flexible compression sample rate and algorithm.

To use the ISD API, certain rules have to be followed, as explained in the following sections.

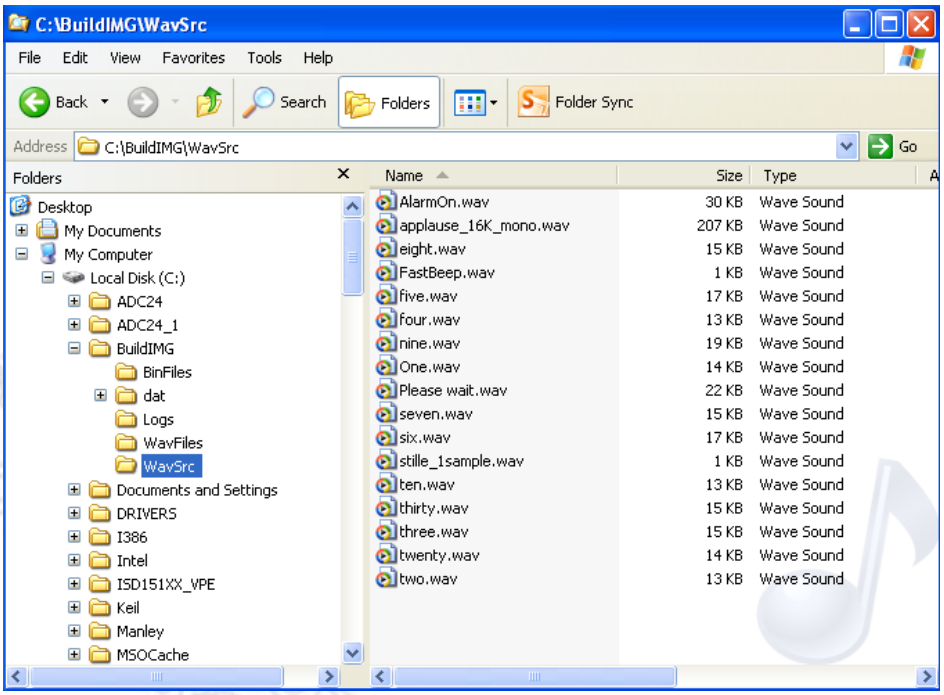
2. Three steps to use the ISD API

Step 1: Prepare the wave source files, and pack them into a folder called WavSrc.

Step 2: Prepare a configuration file, which tell the DLL where to find the source wave, where to save the intermediate files and what should be the final image file.

Step 1: Prepare the working directory: copy all the files/folders into places following a certain directory structure, and run the application which utilizes the DLL.

2.1 Prepare Wave Source Files



Copy all source wave files into the WavSrc subfolder. The source wave files must be in 16bit PCM format, and it is recommended be mono wave files. There is no requirement on source wave file sample rate.

2.2 Prepare the configuration file

Configuration File Syntax

The configuration file follows the INI file convention. It consists of multiple sections; and each section can have each section can have multiple items. Each line inside of a section starts with the item name followed by the item value. For example, a sample INI file can be like:

```
 [String_Setion1]
Item1 = string1
Item2 = string2

[Number_Setion3]
Item1 = Value1
Item2 = Value2
…
```

Configuration File Content

The configuration file must provide the following information:

- In [Info] section
  - Main working directory →the directory must be pre-existed;
  - Sub directory for storing bin files and intermediate eave files → BinFiles and wavFiles directories
  - Clock configuration byte → must be hardcoded 0x34
  - Clock frequency → must be hardcoded 2048000
  - The expected .mem file name
  - Compression algorithm type
  - Compression sample rate
- In [VPWaveList] section
  - All the wave files full name with path → these will be convert to voice prompts by API
- In [ReservedMemoryList] section
  - Each line needs describes a reserved item: Reserved data file full name with path, reserved memory block start address, and the amount of reserved sector (sector size must be 4K byte)
- In [VoicemacroList] section
  - One or multiple lines to describe a Voice Macro by literally listing all the command byte for that voice macro command script.

A Sample Configuration file

A real configuration file is shown below. The descriptions are given after each line starting with "//", and at the end of each INI section.

```
[Info]
DeviceID=18// ***
ProjectDir=C:\BuildIMG// the main working directory, needs to be created in advance
BinDir=C:\BuildIMG\BinFiles// indicate where to save the compressed binary file, API will create this folder
ResampleDIr=C:\BuildIMG\WavFiles// indicate where to save the intermediate compressed/resampled wave file, API will create this folder
SourceFileSampleRate=8000// ***
SourceFileBitsPerSample=16// ***
IntMem=64     // ***
ClockConfigurationByte=0x34// this property value must be 0x34
ClockInputFrequency=2048000// this property value must be 204800
MemFileName=C:\BuildIMG\APIDemo.mem// indicate the final image name
VoicePromptsEncoding=0x64// ***
CompressionType=12// indicate the compression algorithm
CompressionSRCode=3// indicate the desired sample rate for compression
```

"// ***" means the value for that property is a don't-care value; and line ends with don't-care value serves as aplace holder. User should simply leave that line as is, and do not delete that line.

The available CompressionType and CompressionSRCode are shown below:

CompressionType		CompressionSRCode	
Algorithm Type Decimal Value	Algorithm	Sample rate code	Sample rate
10	ADPCM2	0	4K
11	ADPCM3	1	5.3K
12	ADPCM4	2	6.4K
13	ADPCM5	3	8K
16	ULAW6	4	12K
17	ULAW7	5	16K
18	ULAW8	6	32K
20	DULAW6		
21	DULAW7		
22	DULAW8		
28	VBR High Compression		
29	VBR Moderate Compression		

30	VBR high Quality		
24	PCM8		
25	PCM10		
27	PCM12		

[VPWaveList]
VW_1 =C:\BuildIMG\WavSrc\one.wav
VW_2 =C:\BuildIMG\WavSrc\two.wav
VW_3 =C:\BuildIMG\WavSrc\three.wav
VW_4 =C:\BuildIMG\WavSrc\four.wav
VW_5 =C:\BuildIMG\WavSrc\five.wav
VW_6 =C:\BuildIMG\WavSrc\six.wav
VW_7 =C:\BuildIMG\WavSrc\seven.wav
;VW_8 =C:\BuildIMG\WavSrc\eight.wav
;VW_9 =C:\BuildIMG\WavSrc\nine.wav
;VW_10 =C:\BuildIMG\WavSrc\ten.wav

Every wave file that user wants to add into the project needs to be listed here, and also copied into the WavSrc subfolder.
Semi Colon ";" means that line is commented out.

[ReservedMemoryList]
RM_1 = C:\BuildIMG\dat\Konfigurationsdatei.dat [0x1000, 1]
;RM_2 = C:\BuildIMG\dat\a.dat [0x4000, 5]

A block of memory "RM_1" will be reserved. The reserved block starts from byte address 0x1000, with size of 1 sector. Also, the content of user data file Konfigurationsdatei.dat will be copied to reserved block, starting from the beginning of the block which is byte address 0x1000.

The reserved block should start from a sector address. This API support 4kByte sector size only, so the block starting address will always end with hex decimal 0x000.

Please note that if the reserved block size is smaller than the user data file, then the API will automatically increase the block size by the multiplication of 4K byte, until it allocate enough sector to hold the user data.
RM_2 will not be reserved, because it is commented out.

[VoiceMacroList]
VM_1 = 0x00, [0xFF]
VM_2 = 0x01, [0xB4 0x34, 0x00, 0xFF]
VM_1 = 0x02,[0xB8,0x02,0x44,0xB8,0x03,0x00,0xa6,0x00,0x05]
VM_2 = 0x02,[0xa6,0x00,0x06]
VM_3 = 0x02,[0xa6,0x00,0x07]

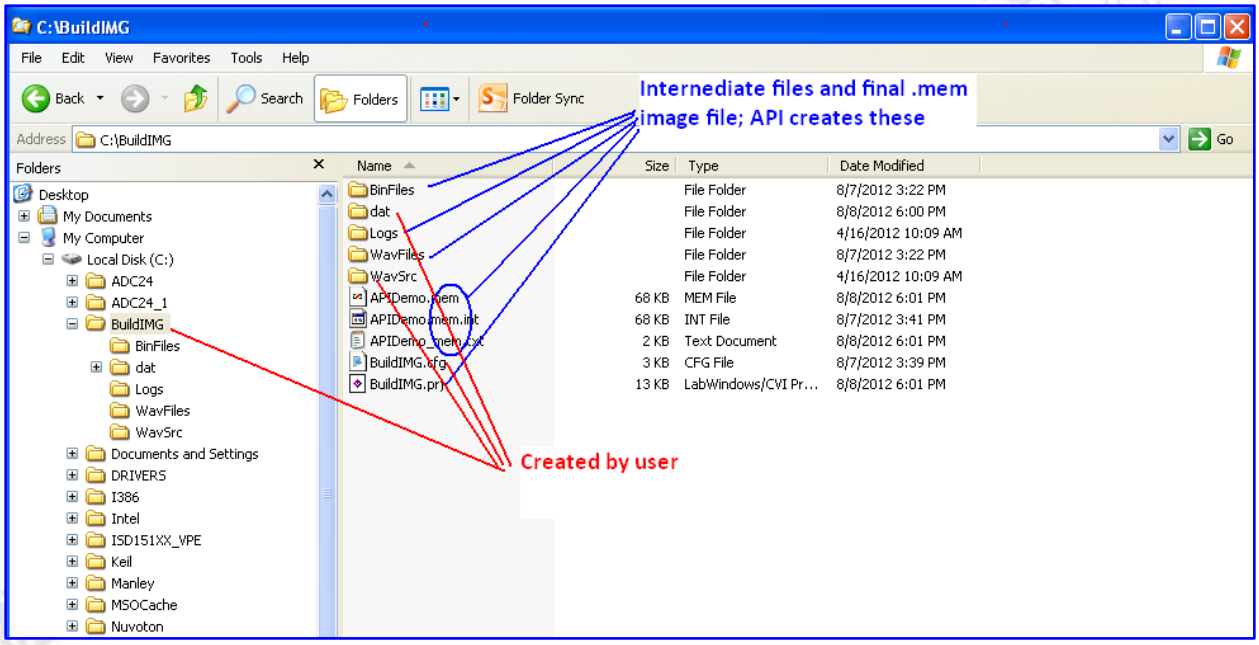
VM_4 = 0x02,[0xFF]
VM_5 = 0x03,[0xFF]
VM_6 = 0x04,[0xFF]

To add VM function, user needs to first translate the VM script command in to hex code byte by byte, then add all the bytes here. This is a number value section which means that all the values here are numbers (in Hex), not strings.

The above creates:

- POI VM, i.e. VM0x00, with a single command: FINISH.
- PU VM, i.e. VM0x01; it sets CLK register value as 0x34, and then ends with FINISH.
- VM0x02; it configures the playback path, sets output volume as maximum, plays VP 0x05, VP 0x06, VP 0x07 and then ends with FINISH.
- VM0x03: with a single command: FINISH.
- VM0x04: with a single command: FINISH.

2.3 Prepare the Working directory structure



3. Other Restrictions

3.1 ISD15100/3900/15C00 application

If the ISD API is used for creating images for ISD15100/3900/15C00 series device, then user must create VM 0x00 (i.e. POI VM) and VM 0x01 (i.e. PU VM) in VoiceMacroList section.

Please note that the VM 0x01 – PU VM cannot end with PD command.

3.2 ISD15D00/3800 application

If the ISD API is used for creating images for ISD15D00/3800 series device, then user must create VM 0x00 (i.e. POI VM), VM 0x01 (i.e. PU VM) and VM 0x02 (Wakeup VM in VoiceMacroList section.
Again note that the VM 0x01 – PU VM and VM 0x02 – Wakeup VM cannot end with PD command.

4. Reference: Voice Macro Command Table

VM command	Command Byte index						remark
	1	2	3	4	5	6	
PWR_DN	0x12	-	-				
SILENCE	0xA8	Data	0x00*				
Play_VP	0xA6	Index[15:8]	Index[7:0]				
Play_VP@Rn	0xAE	*n* = 0 …7	0x00*				
Play_VP_LoopN	0xA4	Index[15:8]	Index[7:0]	N[15:8]	N[7:0]	0x00*	
Play_VP_LoopN@Rn	0xB2	*n* = 0 …7	0x00*	N[15:8]	N[7:0]	0x00*	
EXE_VM	0xB0	Index[15:8]	Index[7:0]				
EXE_VM@Rn	0xBC	*n* = 0 …7	0x00*				
SET_CLK_REG	0xB4	Data	0x00*				Not to be used for ISD2360
WR_CFG_REG	0xB8	Reg_Addr	Data				
MASK_GOTO	0xE0	Addr[23:16]	Addr[15:8]	Addr[7:0]	0x00*	0x00*	Only Available to ISD2360
GOTO	0xE1	Addr[23:16]	Addr[15:8]	Addr[7:0]	0x00*	0x00*	Only Available to ISD2360
WAIT_CHN_CNT	0xEE	0x00*	0x00*				Only Available to ISD2360
WAIT_INT	0xFC	0x00*	0x00*				Not available to ISD2360
FINISH	0xFF	-	-				

Note: * stands for the dummy byte.

5. Setup demo project

The distribution package includes the following:

- ISD.DLL
- A visual c++ 6.0 demo project: EP4.esw
- A sample working folder

The user can use the demo project as the reference to build the target application. Also, please follow the example configuration file “BuildIMG.cfg” to create the customized configuration file.

For any further questions regarding tech support, please contact: ChipCorder@nuvoton.com.

新唐科技 NUVOTON INTELLECTUAL PROPERTY

REVISION HISTORY

Version	Date	Description
1.0	Aug 09, 2012	First Release

新唐科技 nuvoTon INTELLECTUAL PROPERTY