Application Note

AN_338

A Human Voice On FT800

Version 1.0

Issue Date: 2014-10-15

This document shows how to record a human voice into a file and convert it for FT800 audio playback.

Future Technology Devices International Limited (FTDI)
Unit 1, 2 Seaward Place, Glasgow G41 1HH, United Kingdom
Tel.: +44 (0) 141 429 2777 Fax: + 44 (0) 141 429 2758
Web Site: http://ftdichip.com

Table of Contents

1 Introduction

This application note documents how to record a human voice and covert the file to 8 bit u-Law for playback on the FT800. The document also provides an FT800 sample display list to perform the playback. Project source code for Arduino may be downloaded here.

1.1 Scope

This document covers how to record a human voice to 8 bit u-Law files which are FT800 compatible format. Play these files by a VM800P module when different buttons are pressed.

1.2 Software Required

- Audacity (open source, available for free download)
- Aud_Cvt (EVE Audio Converter)
- Arduino IDE (Compile FT800 program for VM800P and program VM800P EEPROM)
- ARDUINO LIBRARY FOR FT800 SERIES (AN 318 Arduino library for FT800 series)

1.3 Hardware Required

- VM800P module

1.4 FT800 Audio file Required

An 8 bit u-Law formatted file is used in this application note.

Figure 1.1 shows how a human voice convents to an 8 bit u-Law file

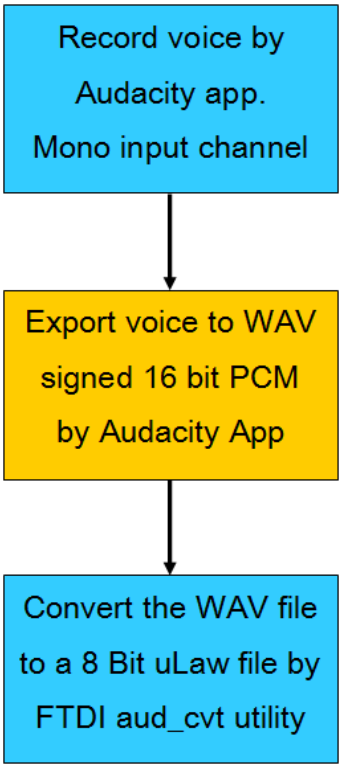

Figure 1.1 How a Human Voice becomes an 8 bit u-Law file

2 Record a Human Voice with Audacity and Aud_cvt

Record a human voice by Audacity with a microphone, a Skype phone, or an internal microphone. Set to (Mono) Input Channel because aud_cvt supports only mono input channel. Figure 2.1 shows how to set to Mono Input Channel

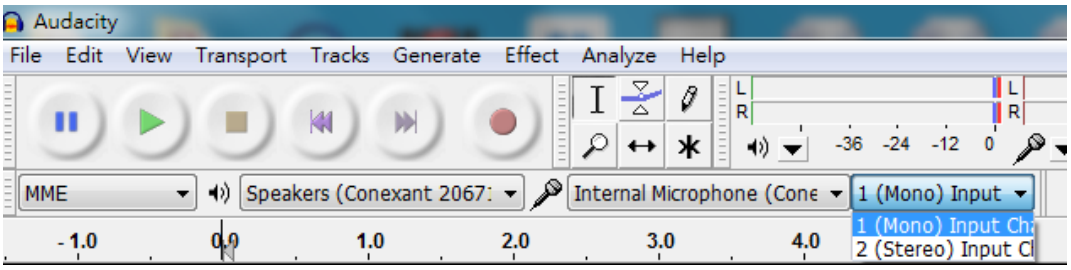

Figure 2.1 Set to (Mono) Input Channel

Set Project Rate to 44100 (Hz) in this application note. Figure 2.2 shows how to set project rate to 44100(Hz)

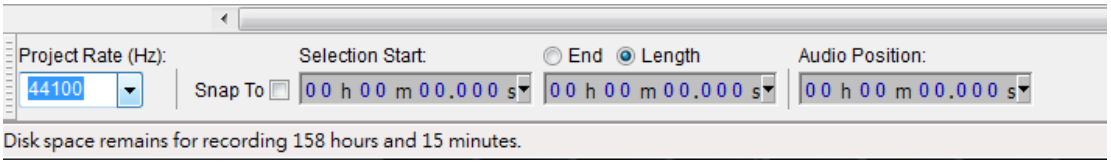

Figure 2.2 Set Project Rate to 44100(Hz)

Click the record button to begin to record voice. Figure 2.2 shows the location of the record button

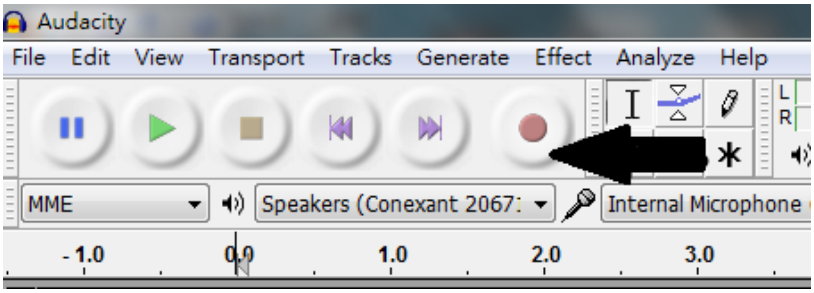

Figure 2.3 Location of The Record Button

Click the stop button when it is completed

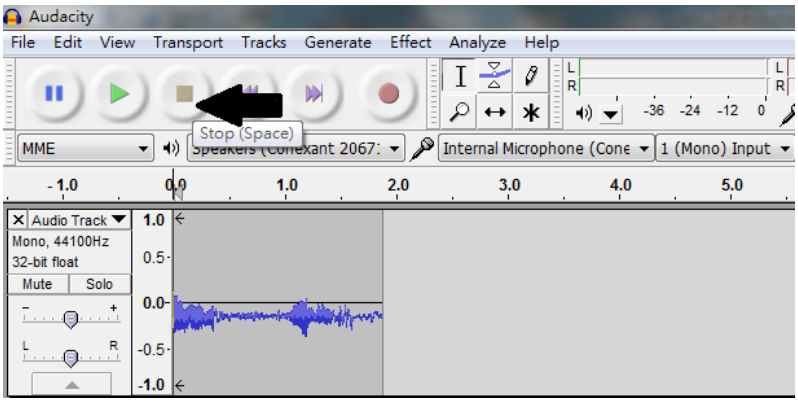

Figure 2.4 Location of The Stop Button

In the File pull down menu, select "Export" and save the file as a 16 bit signed PCM wav file. The saved file was titled test.wav for this application note. After the 16 bit PCM wav file is saved, use aud_cvt to convert it to 8-bit u-Law file.

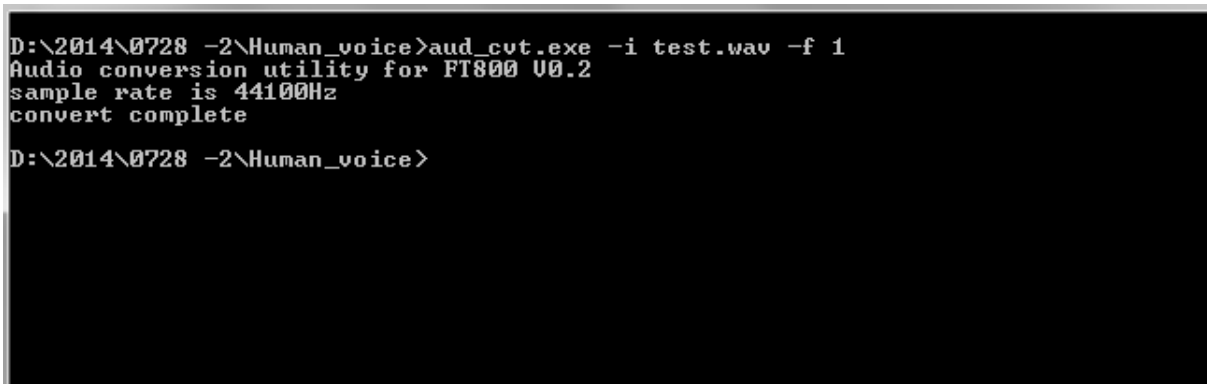

Figure 2.5 Convert test.wav to 8-bit u-Law

There are 4 new files created by aud_cvt conversion. The test.raw will be used would be used in the FT800 display list for playback.

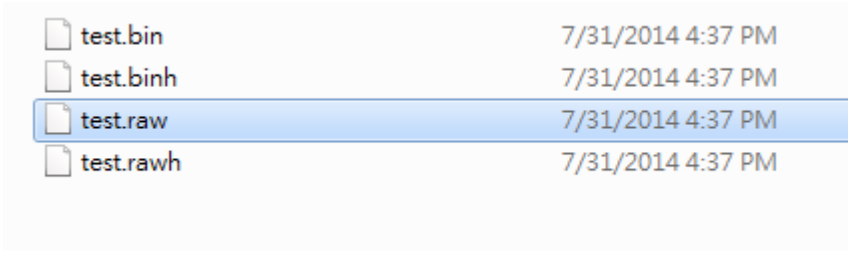

Figure 2.6 4 new files are created after aud_cvt utility

2.1 Recorded files for the sample application

8 voice files were generated using the process described above for playback and stored on an SD card which plugs into the VM800P (FT800 Plus module).

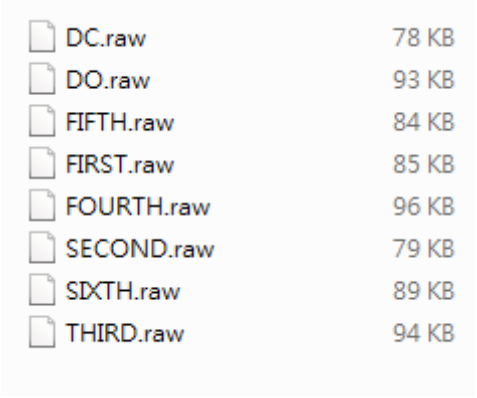

Figure 2.7 8 8-bit uLaw files are needed for this sample program

3 Sample Application

The sample code of this application uses AN_318 ARDUINO LIBRARY FOR FT800 SERIES. The sample code emulates a human voice delivering information that may be used in an elevator such as "door opening", "door closing", "1st floor" ~"6th floor".

When a user presses a button on the VM800P module display, the program will highlight the button and playback a corresponding audio file.

There are 7 functions in this example program

1. BootupConfigure: Initial display resolution, set display enable pin, set audio enable pin, enable display and enable audio

```
int16_t BootupConfigure()
{
        FTImpl.Init(FT_DISPLAY_RESOLUTION);//configure the display to the WQVGA
        /* Set the Display & audio pins */
        FTImpl.SetDisplayEnablePin(FT_DISPENABLE_PIN);
        FTImpl.SetAudioEnablePin(FT_AUDIOENABLE_PIN);
        FTImpl.DisplayOn();
        FTImpl.AudioOn();
        return 0;
}
```

Figure 3.1 BootupConfigure function

2. Read_Keys: return the tag value if any button is pressed

```
static uint8_t keypressed=0;
uint8_t Read_Keys()
{
  static uint8_t Read_tag=0,temp_tag=0,ret_tag=0;
  Read_tag = FTImpl.Read(REG_TOUCH_TAG);
  ret_tag = NULL;
  if(Read_tag!=NULL)            // Allow if the Key is released
  {
    if(temp_tag!=Read_tag)
    {
      temp_tag = Read_tag;
      keypressed = Read_tag;    // Load the Read tag to temp variable
    }
  }
  else
  {
    if(temp_tag!=0)
    {
      ret_tag = temp_tag;
      temp_tag = 0;
    }
    keypressed = 0;
  }
  return ret_tag;
}
```

Figure 3.2 Read_Keys function

3. Calibrate: Calibration is done to align the touch and the display layers on the LCD panel.

4. RedrawBtn: The program will redraw buttons again after users press any button

```
void RedrawBtn()
{

  FTImpl.Clear(1, 1, 1);
  FTImpl.Cmd_FGColor(12632256);
  FTImpl.ColorRGB(0,0,0);
  FTImpl.Cmd_Button(100,50,85,30,30,0,"5");
  FTImpl.Cmd_Button(100,100,85,30,30,0,"3");
  FTImpl.Cmd_Button(100,150,85,30,30,0,"1");
  FTImpl.Cmd_Button(250,50,85,30,30,0,"6");
  FTImpl.Cmd_Button(250,100,85,30,30,0,"4");
  FTImpl.Cmd_Button(250,150,85,30,30,0,"2");
  FTImpl.Cmd_Button(100,200,85,30,18,0,"DOOR OPEN");
  FTImpl.Cmd_Button(250,200,85,30,18,0,"DOOR CLOSE");
}
```

Figure 3.3 RedrawBtn function

5. Info: Draw all the buttons and set the tag on all the buttons. Wait till any button is pressed

```
do {
  FTImpl.DLStart( );
  if( (keypressed!='0') && (keypressed!='C') && (keypressed!='1') && (keypressed!='2') && (keypressed!='3') && (keypressed!='4') && (keypressed!='5') && (keypressed!='6'))
    FTImpl.ColorRGB(255,255,255);
  else FTImpl.Clear(1, 1, 1);
  FTImpl.Cmd_FGColor(12632256);
  FTImpl.ColorRGB(0,0,0);
  FTImpl.Tag('5');
  FTImpl.Cmd_Button(100,50,85,30,30,0,"5");
  FTImpl.Tag('3');
  FTImpl.Cmd_Button(100,100,85,30,30,0,"3");
  FTImpl.Tag('1');
  FTImpl.Cmd_Button(100,150,85,30,30,0,"1");
  FTImpl.Tag('6');
  FTImpl.Cmd_Button(250,50,85,30,30,0,"6");
  FTImpl.Tag('4');
  FTImpl.Cmd_Button(250,100,85,30,30,0,"4");
  FTImpl.Tag('2');
  FTImpl.Cmd_Button(250,150,85,30,30,0,"2");
  FTImpl.Tag('0');
  FTImpl.Cmd_Button(100,200,85,30,18,0,"DOOR OPEN");
  FTImpl.Tag('C');
  FTImpl.Cmd_Button(250,200,85,30,18,0,"DOOR CLOSE");
  FTImpl.DLEnd();
  FTImpl.Finish(); //render the display list and wait for the completion of the DL
  MyKey=Read_Keys();
  } while( (MyKey!='0') && (MyKey!='C') && (MyKey!='1') && (MyKey!='2') && (MyKey!='3') && (MyKey!='4') && (MyKey!='5') && (MyKey!='6')) ;
 /* wait until Play key is not pressed*/
```

Figure 3.4 Info function

6. Load_afile: Read a file from an SD card

```
void Load_afile(uint32_t add, FT_SDFile &r)
{
  uint8_t pbuff[512],temp[512],tval;
  uint16_t z = 0;
  r.ReadSector(pbuff);
  FTImpl.Write(add,pbuff,512L);
  if ((add & 2047L) == 0)       // every 2kb i update the bitmap
  {
    for(z=0;z<512L;z++)
    {
      tval = pbuff[z];
      if (tval & 0x80L)         // ll bits of data
      tval ^= 0x7fL;
      temp[z] = tval;
    }
    FTImpl.Write(8192L,temp,512L);
 }
}
```

Figure 3.5 Load_afile function

7. Player: The real main function of the program. The function gets tag value of a button after calls the Info function. "Turn on the light" of the pressed button, and then open the corresponding voice file. Load the file and then play it.

```
void Player(void)
{
  while(1)
  {
    Info();
    uint32_t ftsize=0;
    uint16_t wp = 0;
    uint32_t rp=0,n,val;
    // Intilaize the audio setting
    FTImpl.Write32(REG_PLAYBACK_FREQ,44100);
    FTImpl.Write32(REG_PLAYBACK_START,0);
    FTImpl.Write32(REG_PLAYBACK_FORMAT,FT_ULAW_SAMPLES);
    FTImpl.Write32(REG_PLAYBACK_LENGTH,8192);
    FTImpl.Write32(REG_PLAYBACK_LOOP,1);
    FTImpl.Write(REG_VOL_PB,255);
    // open the audio file from the SD card
    byte Status = 0;
    switch(MyKey) {
      case '0':  // Door opening
        FTImpl.Cmd_DLStart();
        RedrawBtn();
        FTImpl.ColorRGB(255,0,0);
        FTImpl.Cmd_Button(100,200,85,30,18,0,"DOOR OPEN");
        FTImpl.DLEnd();
        Status = FtSd.OpenFile(Audiofile,"D0.raw");
        break;
```

Figure 3.6 Get the tag value of the pressed button open the corresponding file

```
    // load the audio file
    Load_afile(0UL, Audiofile);
    wp = 512;
    // Initate to play
    FTImpl.Write(REG_PLAYBACK_PLAY,1);
    ftsize = Audiofile.Size;
    while(ftsize > 0)
    {
      rp = FTImpl.Read16(REG_PLAYBACK_READPTR);
      val = 8191 & (rp-wp);
      if (val > 512)
      {
        n = min(512L,ftsize);
        Load_afile(wp, Audiofile);
        wp = (wp +512L) & 8191L;
        ftsize-=n;
      }
    }
    FTImpl.Write(REG_VOL_PB,0);
    FTImpl.Write(REG_PLAYBACK_PLAY,0);
```

Figure 3.7 Load the corresponding file and then play it to make a human voice

4 Testing Results

Figure 4.1 Sample application elevator buttons

Figure 4.2 The "DOOR OPEN" button is pressed and plays back a "door opening" audio file

Figure 4.3 The "5" button is pressed and plays back a "5th floor" audio file

5 Contact Information

Head Office – Glasgow, UK

Future Technology Devices International Limited
Unit 1, 2 Seaward Place, Centurion Business Park
Glasgow G41 1HH
United Kingdom
Tel: +44 (0) 141 429 2777
Fax: +44 (0) 141 429 2758

E-mail (Sales) sales1@ftdichip.com
E-mail (Support) support1@ftdichip.com
E-mail (General Enquiries) admin1@ftdichip.com

Branch Office – Tigard, Oregon, USA

Future Technology Devices International Limited
(USA)
7130 SW Fir Loop
Tigard, OR 97223-8160
USA
Tel: +1 (503) 547 0988
Fax: +1 (503) 547 0987

E-Mail (Sales) us.sales@ftdichip.com
E-Mail (Support) us.support@ftdichip.com
E-Mail (General Enquiries) us.admin@ftdichip.com

Branch Office – Taipei, Taiwan

Future Technology Devices International Limited
(Taiwan)
2F, No. 516, Sec. 1, NeiHu Road
Taipei 114
Taiwan , R.O.C.
Tel: +886 (0) 2 8797 1330
Fax: +886 (0) 2 8751 8737

E-mail (Sales) tw.sales1@ftdichip.com
E-mail (Support) tw.support1@ftdichip.com
E-mail (General Enquiries) tw.admin1@ftdichip.com

Branch Office – Shanghai, China

Future Technology Devices International Limited
(China)
Room 1103, No. 666 West Huaihai Road,
Shanghai, 200052
China
Tel: +86 21 62351596
Fax: +86 21 62351595

E-mail (Sales) cn.sales@ftdichip.com
E-mail (Support) cn.support@ftdichip.com
E-mail (General Enquiries) cn.admin@ftdichip.com

Web Site

http://ftdichip.com

Appendix A – References

Document References

FT800 Datasheet

VM800P datasheet

FT800 Series Programmers Guide

AN338 A Human Voice on FT800 Sample Code

AN276 FT800 Audio File Conversion

AN318 Arduino library for FT800 series

Appendix B – List of Tables & Figures

List of Figures

Appendix C – Revision History

Document Title: AN_338 A Human Voice On FT800
Document Reference No.: FT_001098
Clearance No.: FTDI#416
Product Page: http://www.ftdichip.com/FTProducts.htm
Document Feedback: Send Feedback

Revision	Changes	Date
1.0	Initial Release	2014-10-15