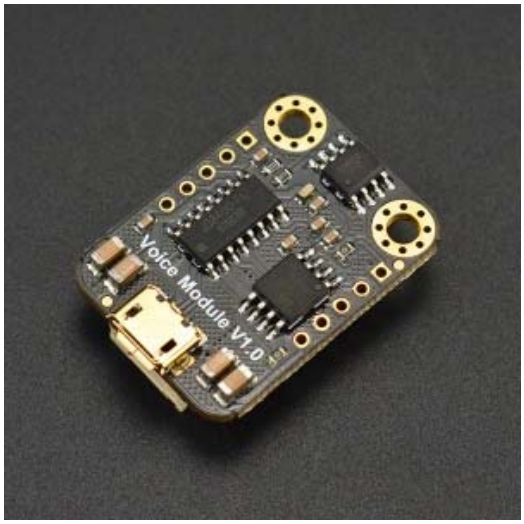


Voice Module SKU: DFR0534

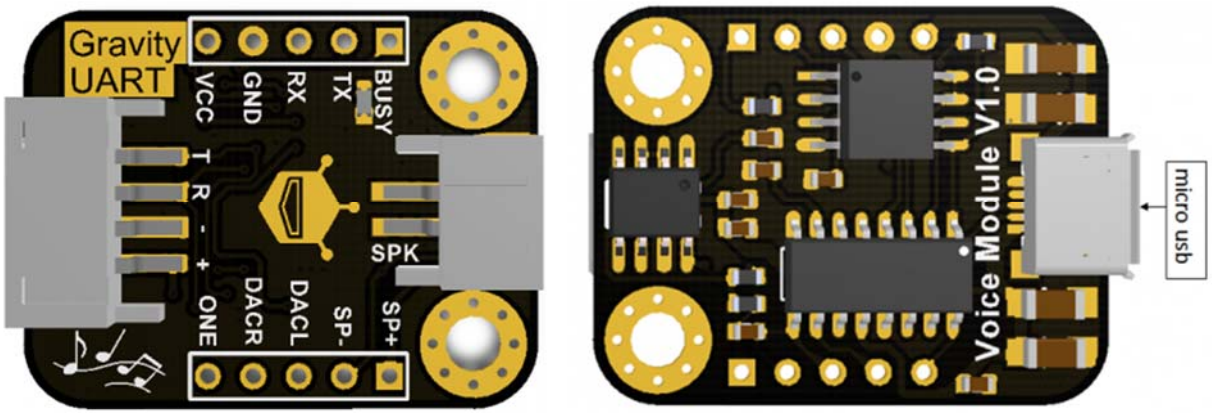
Introduction

Do you always feel like the DIY work is boring, not attractive or no vitality when you design it by yourself?
This new voice module produced by DFRobot can solve these problems. This module can store 8M audio files, which means it can store more than 200 audios for your choice. Its storage method is as simple as USB flash drive, and audios can be updated at any time. It uses the DF-Gravity interface, free from wiring; It has the ability to play the specified audio, play in loop, and adjust volume. It is suitable for all kinds of projects related to sound or voice, such as intelligent car, meteorological stations, smart home devices, vehicle navigation, toll stations, safety monitoring, machine audio navigation and so on.

Specification

- Operating Voltage: 3.3V ~ 5V
- Interface: UART
- Support MP3 WAV Hardware Decoding
- Support Sampling Rate (KHz): 8/11.025/12/16/22.05/24/32/44.1/48
- Support to imitate SPIFLASH to U disk, and audios in SPIFLASH can be updated by the way to operate USB flash drive
- Support 30 levels of volume adjustment
- Dimension: 22x30mm/0.87x1.18in

Board Overview



Num	Name	Description
1	T	TX
2	R	RX
3	-	GND
4	+	3.3V-5V
5	VCC	3.3V-5V
4	GND	GND
5	BUSY	Busy signal pins, high for play, others are low
6	SP+	Trumpet
7	SP-	Trumpet
8	DACL	DAC audio output in left channel
9	DACR	DAC audio output in right channel
10	ONE	One-line serial control pins
11	micro usb	File Update Interface (Connect USB to PC, the storage mode is as simple as U disk)

COMMREQ

```
Check Playback Status(01)
```

Command: AA 01 00 AB

Return: AA 01 01 Playback status SM

Explanation: You can check the play status at any time.

```
Play(02)
```

Command: AA 02 00 AC

Return: None

Explanation: Play the current track from the beginning at any time.

```
Pause(03)
```

Command: AA 03 00 AD

Return: None

```
Playstop(04)
```

Command: AA 04 00 AE

Return: None

```
Prev Audio(05)
```

Command: AA 05 00 AF

Return: None

```
Next Audio(06)
```

Command: AA 06 00 B0

Return: None

```
Specified Audio:(07)
```

Command: AA 07 02 Upper-Byte Lower-Byte SM

Return: None

E.g. AA 07 02 00 08 BB will specify to play the eighth audio in the current drive, and the audio number is from 1 to 65535.

```
Play the audio in specified drive of specified path.(08)
```

Command: AA 08 length drive path SM

Return: None

Explanation: length= drive length + path length=1+ path length

```
Check the current online drive.(09)
```

Command: AA 09 00 B3

Return: AA 09 01 Drive SM

Explanation: The online drive is differentiated by bitwise ***usb:bit (0) sd:bit (1) flash:bit (2)***

It can check the current online drive. It is recommended that you check the current online drive before switching the drive.

```
Check the current drive(0A) to play
```

Command: AA 0A 00 84

Return: AA 0A 01 Drive SM

```
Switch to the specified drive(0B)
```

Command: AA 0B 01 drive SM

Return: None

Explanation: It is drive switching command. If the current drive is online, you can switch it to the corresponding drive and wait to play. After switching, the module plays the first audio of this drive. It is recommended to check whether the drive is online before you switch.

```
Check the total audios(0C)
```

Command: AA 0C 00 B6

Return: AA 0C 02 Upper-byte Lower-byte SM

```
Check the current audio(0D)
```

Command: AA 0D 00 B7

Return: AA 0D 02 upper-byte lower-byte SM

```
The Directory of the previous folder(0E)
```

Command: AA 0E 00 B8

Return: None

Explanation: The first audio in the folder will be played after switching.

```
The directory of the next folder(0F)
```

Command: AA 0F 00 B9

Return: None

Explanation: The first audio in the folder will be played after switching.

```
End to play(10)
```

Command: AA 10 00 BA

Explanation: This command can end the current operations in advance. If the command end a cut-in, it will end in advance and return to the original status.

```
Check the first audio in the folder(11)
```

Command: AA 11 00 BB

Return: AA 11 02 Upper-byte Lower-byte SM

Explanation: It is the sequence number of the first audio.

```
Check the total numbers of audio in all folders(12)
```

Command: AA 12 00 BB

Return: AA 12 02 Upper-byte Lower-byte SM

Explanation: This number does not contain the number of files in subdirectories.

```
Volume settings(13)
```

Command: AA 13 01 VOL SM

Return: None

Explanation: AA 13 01 14 D2 Set volume to level 20.

```
Increase the volume(14)
```

Command: AA 14 00 BE

Return: None

```
Reduce the volume(15)
```

Command: AA 15 00 BF

Return: None

```
Cut-in a specific audio(16)
```

Command: AA 16 03 drive Upper-byte Lower-byte SM

Return: None

E.g. AA 16 03 00 00 09 CC Cut-in the 9th audio in the USB flash drive.

Explanation: Continue to play the original audio when cut-in audio is played.

```
Cut-in the audio on the specified path(17)
```

Command: AA 17 Length Drive Path SM

Return: None

Explanation: length= drive length + path length=1+ path length

```
Set the loop mode(18)
```

Command: AA 18 01 loop mode

Return: None

E.g. Set to end single play AA 18 01 03 C6

```
Set the number of loop times(19)
```

Command: AA 19 02 Upper-byte Lower-byte SM

Return: None

Explain: This command is only valid when the loop mode is Full Loop, Single Loop, Catalog Loop.

E.g. AA 19 02 00 06 CB Repeat 6 times.

```
EQ Setting (1A)
```

Command: AA 1A 01 EQ SM

Return: None

E.g. AA 1A 01 02 C7 Set EQ to ROCK

```
Combination Play(1B)
```

Command: AA 1B Upper-byte of the audio 1, Lower-byte of the audio 1...Upper-byte of the audio n, Lower byte of the audio n.

Return: None

E.g. AA1B04303130328C, audios with file names of 01 and 02 are grouped together to play

Explanation: The combination of file names is very convenient, more accurate than combined file numbers, and is not restricted by the copy order limitation.

```
End Combination Play(1C)
```

Command: AA 1C 00 C6

Return: None

Explanation: End combination playback, and return the play state before the combination.

```
Set channels(1D)
```

Command: AA 1D 01 channels SM

Return: None

```
Check the short-file name(1E)
```

Command: AA 1E 00 C8

Return: AA 1E The length of short-file name Short-file name SM

```
Select audio but do not play(1F)
```

Command: AA 1F 02 Upper-byte Lower-byte SM

Return: None

```
Control of Repetition(20)
```

Command: AA 20 04 beginning-minute beginning-second ending-minute ending-second SM

Return: None

```
End repetition(21)
```

Command: AA 21 00 CB

Return: None

```
Specify the time to rewind(22)
```

Command: AA 22 02 Upper-byte Lower-byte SM

Return: None

Explanation: The unit is seconds

```
Specify the time to wind(23)
```

Command: AA 23 02 Upper-byte Lower-byte SM

Return: None

Explanation: The unit is seconds

```
Get the total time of the current audio(24)
```

Command: AA 24 00 CE

Return: AA 24 03 hour minute second SM

```
Start the playback time sending(25)
```

Command: AA 25 00 CF

Return: AA 25 03 hour minute second SM

Explanation: Start to send playing time, Return automatically when the time is updated.

```
End to send the playing time(26)
```

Command: AA 26 00 D0

Return: None

Explanation: End the playback time sending.

Tutorial

1. Specify the audio to play
2. Specify the audio to loop
3. Modify the volume

Preparation

Hardware

- 1x Arduino UNO
- 1x IO Expansion Board
- 1x Passive Speaker (8Ω3W)
- Dupont lines

Software

- Arduino IDE,click to Download Arduino IDE from Arduino®.

Download Audio File

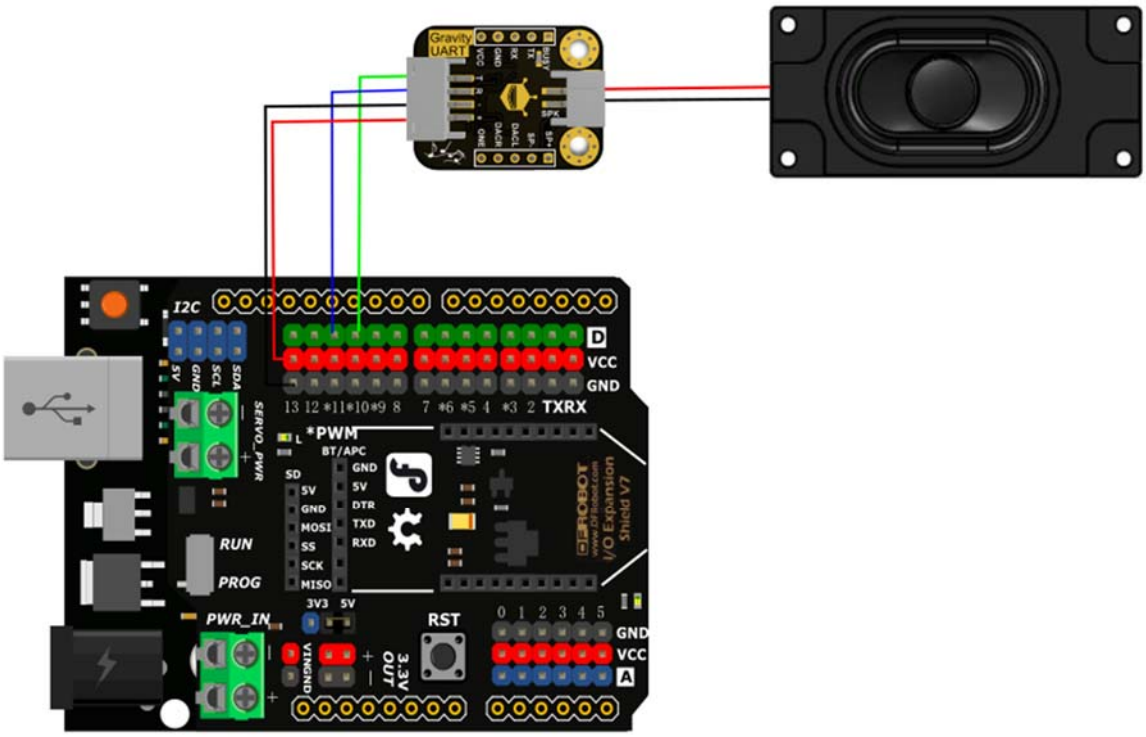
1.The module has built-in audio, if you need to add or replace audio, please use the micro usb to connect to the computer for updating.

2. The way to update the audio is the same as USB flash driver.

3. This module supports MP3 and MAV format audio files.

4. The file should be stored in the "ZH" folder and it is recommended to represent files with numbers. Such as 01.mp3,02.mp3, two letters or one Chinese character are also OK.

Connection Diagram



Sample code

```
/*
 * @file Voice Module.ino
 * @brief
 * @n [Get the module here]
 * @n This example Set the voice module volume and playback
 * @n [Connection and Diagram]()
 *
 * @copyright  [DFRobot](http://www.dfrobot.com), 2016
 * @copyright GNU Lesser General Public License
 *
 * @author [carl](lei.wu@dfrobot.com)
 * @version  V1.0
 * @date  2017-11-3
```

```
 */
#include <SoftwareSerial.h>

SoftwareSerial Serial1(10, 11);

unsigned char order[4] = {0xAA,0x06,0x00,0xB0};


void setup() {
//Serial.begin(115200);
 Serial1.begin(9600);
 volume(0x1E);//Volume settings 0x00-0x1E
}


void loop() {
 play(0x01);//Play the specified audio:0x01-file0001
// Serial1.write(order,4);//order play
 delay(2000);
}

void play(unsigned char Track)
{
 unsigned char play[6] = {0xAA,0x07,0x02,0x00,Track,Track+0xB3};
   Serial1.write(play,6);
}
void volume( unsigned char vol)
{
  unsigned char volume[5] = {0xAA,0x13,0x01,vol,vol+0xBE};
     Serial1.write(volume,5);
 }
```