

Keyes Player Mini MP3 Module

(Red & Environmental-protection)



1. Introduction

It is an affordable small MP3 module that can directly be connected to the speaker. The module can be used alone with the power supply battery, speaker as well as button, and also can be used as a module of Arduino UNO or any serial ports through the serial port control. The module itself is perfectly integrated with MP3, WAV, WMA hard decoding, and meanwhile the software supports TF card driver, as well as FAT16 and FAT32 file system. It can play the specified music and other functions just via a simple serial commands, without cumbersome underlying operation, which is easy-to-use, stable and reliable.

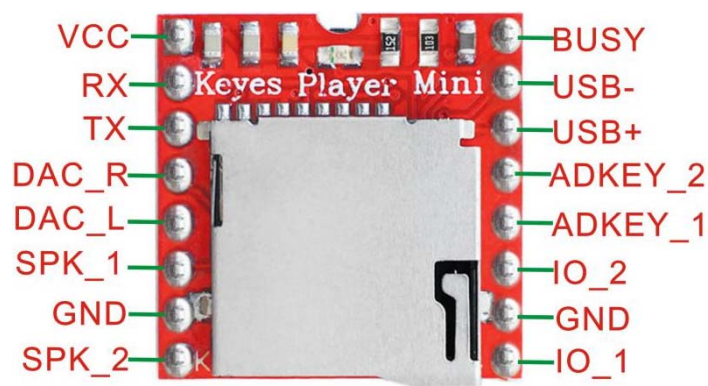
2. Parameters

- Support Sampling Rate (KHz): 8/11.025 / 12/16 / 22.05 / 24/32 / 44.1 / 48
- 24-bit DAC output, dynamic range supporting 90dB, signal to noise ratio supporting 85dB
- Full support FAT16, FAT32 file system; support the largest 32G TF card; support 32G U disk; 64M bytes NORFLASH.
- Various control modes are available. IO control mode, serial port mode, AD button control mode.
- The interstitial function of the broadcaster can pause the background music being played.

After ad. broadcasting, it will play back to the background music.

- Audio data sorted by folder, up to 100 folders, each folder can be assigned with 255 songs.
- 30 levels of sound volume and 6 levels of EQ are adjustable.

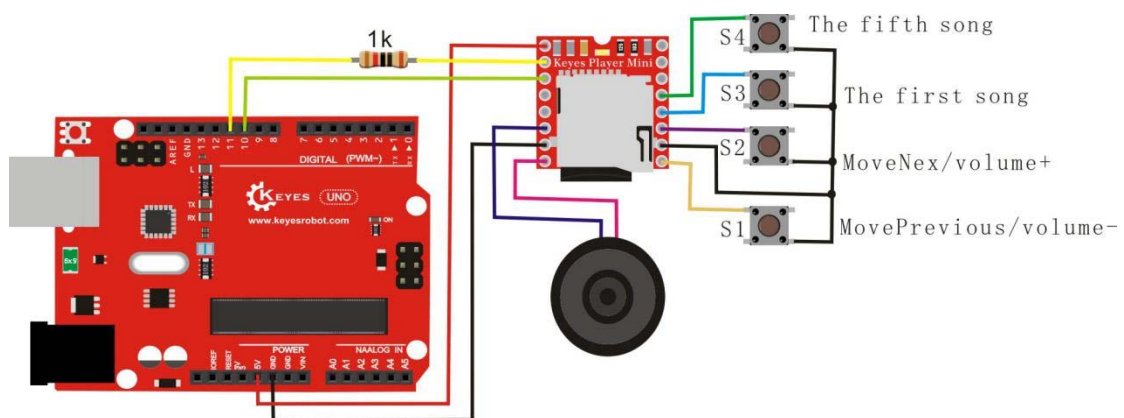
3. Pin Explanation



Pin No.	Pin Name	Function	Notes
1	VCC	Power Supply Input	3.3-5V, recommend 5V, no greater than 5.2V
2	RX	UART Serial Value Input	
3	TX	UART Serial Value Output	
4	DAC_R	Audio output right channel	Drive earphones, power amplifier
5	DAC_L	Audio output left channel	Drive earphones, power amplifier
6	SPK_1	Connect to speaker	Drive the speaker less than 3W

7	GND	Ground	Power ground
8	SPK_2	Connect to speaker	Drive the speaker less than 3W
9	IO_1	Trigger Port	Default the previous song (long press the volume to minus it)
10	GND	Ground	Power ground
11	IO_2	Trigger Port	Default the next song (long press the volume to plus it)
12	ADKEY_1	AD □ 1	When the trigger is the first one (long press to cycle it)
13	ADKEY_2	AD □ 2	When the trigger is the fifth one (long press to cycle it)
14	USB+	USB+ DP	Connect to U disk or plug into the USB port of PC
15	USB-	USB+ DM	Connect to U disk or plug into the USB port of PC
16	BUSY	Play Status	With audio, the output is low; no audio, the output is high.

4. Connection Diagram



5. Sample Code

```
#include "Arduino.h"

#include "SoftwareSerial.h"

#include "DFRobotDFPlayerMini.h"

SoftwareSerial mySoftwareSerial(10, 11); // RX, TX

DFRobotDFPlayerMini myDFPlayer;

void printDetail(uint8_t type, int value);

void setup()
{
    mySoftwareSerial.begin(9600);
    Serial.begin(115200);

    Serial.println();
    Serial.println(F("DFRobot DFPlayer Mini Demo"));
    Serial.println(F("Initializing DFPlayer ... (May take 3~5 seconds)"));

    if (!myDFPlayer.begin(mySoftwareSerial)) { //Use softwareSerial to communicate with
mp3.
        Serial.println(F("Unable to begin:"));
        Serial.println(F("1.Please recheck the connection!"));
        Serial.println(F("2.Please insert the SD card!"));
        while(true);
    }

    Serial.println(F("DFPlayer Mini online.));

    myDFPlayer.volume(10); //Set volume value. From 0 to 30

    myDFPlayer.play(1); //Play the first mp3
```

```

}

void loop()
{
    static unsigned long timer = millis();

    if (millis() - timer > 30000) {
        timer = millis();
        myDFPlayer.next(); //Play next mp3 every 30 second.
    }

    if (myDFPlayer.available()) {
        printDetail(myDFPlayer.readType(), myDFPlayer.read()); //Print the detail message from
        DFPlayer to handle different errors and states.
    }
}

void printDetail(uint8_t type, int value){
    switch (type) {
        case TimeOut:
            Serial.println(F("Time Out!"));
            break;
        case WrongStack:
            Serial.println(F("Stack Wrong!"));
            break;
        case DFPlayerCardInserted:
            Serial.println(F("Card Inserted!"));
            break;
        case DFPlayerCardRemoved:
            Serial.println(F("Card Removed!"));

```

```

        break;

case DFPlayerCardOnline:

    Serial.println(F("Card Online!"));

    break;

case DFPlayerPlayFinished:

    Serial.print(F("Number:"));

    Serial.print(value);

    Serial.println(F(" Play Finished!"));

    break;

case DFPlayerError:

    Serial.print(F("DFPlayerError:"));

    switch (value) {

        case Busy:

            Serial.println(F("Card not found"));

            break;

        case Sleeping:

            Serial.println(F("Sleeping"));

            break;

        case SerialWrongStack:

            Serial.println(F("Get Wrong Stack"));

            break;

        case CheckSumNotMatch:

            Serial.println(F("Check Sum Not Match"));

            break;

        case FileIndexOut:

            Serial.println(F("File Index Out of Bound"));

            break;

        case FileMismatch:

            Serial.println(F("Cannot Find File"));

```

```

        break;
    case Advertise:
        Serial.println(F("In Advertise"));
        break;
    default:
        break;
    }
    break;
default:
    break;
}
}

```

6. Test Result

Wiring and uploading the code, after powered up, use the code to control MP3 module, about 3 seconds later, you can hear a song played by speaker and switched to the next one for 3 seconds. Then pull out the RX and TX connection cable, and use the button to test it. Powered on, you can hear the song, press S1 to move to the previous song, long press S1 to reduce the sound volume. Press S2 to move to the next song, long press S2 to increase the sound volume. Long press S3 to go to the fifth song, long press S3 to loop the fifth one. Press S4 for the first song, long press it to cycle.

7. Related Data Link

<http://url.cn/5yfu0SZ>