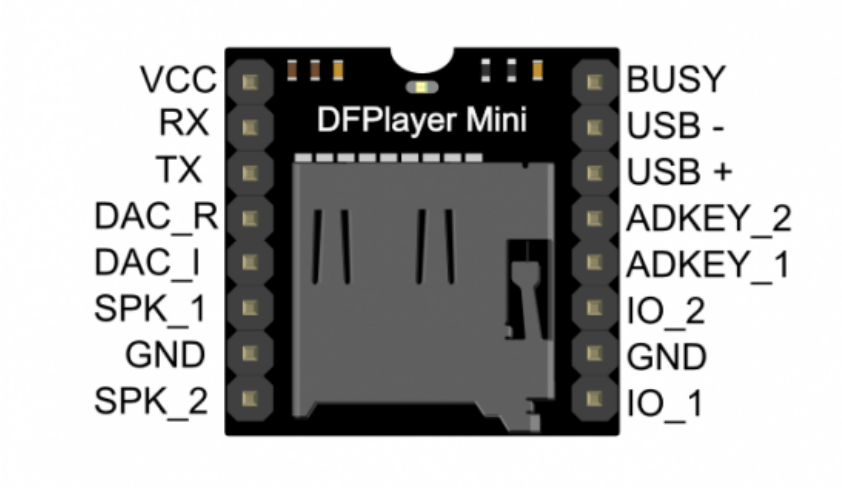
MP3 Module - DFPlayer Mini

The **DFPlayer Mini MP3 Player For Arduino** is a small and low price MP3 module with an simplified output directly to the speaker. The module can be used as a stand alone module with attached battery, speaker and push buttons or used in combination with an **Arduino UNO** or any other with RX/TX capabilities.

SPECIFICATION

- Supported sampling rates (kHz): 8/11.025/12/16/22.05/24/32/44.1/48
- 24 -bit DAC output, support for dynamic range 90dB, SNR support 85dB
- Fully supports **FAT16, FAT32 file system, maximum support 32G** of the TF card, support 32G of U disk, 64M bytes NOR FLASH
- A variety of control modes, I/O control mode, serial mode, AD button control mode
- Advertising sound waiting function, the music can be suspended. when advertising is over in the music continue to play
- Audio data sorted by folder supports up to 100 folders, every folder can hold up to 255 songs
- 30 level adjustable volume, 6 -level EQ adjustable

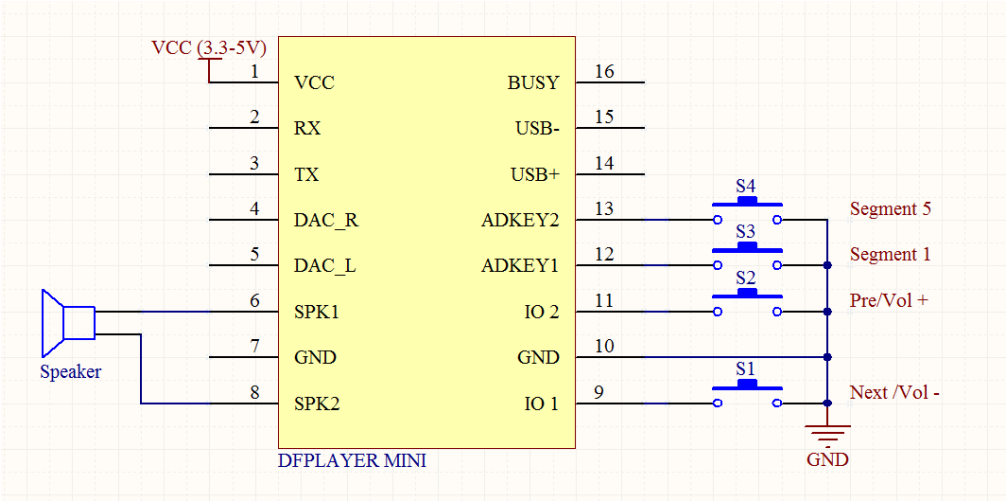
NOTE: Need 5V to drive the speaker.

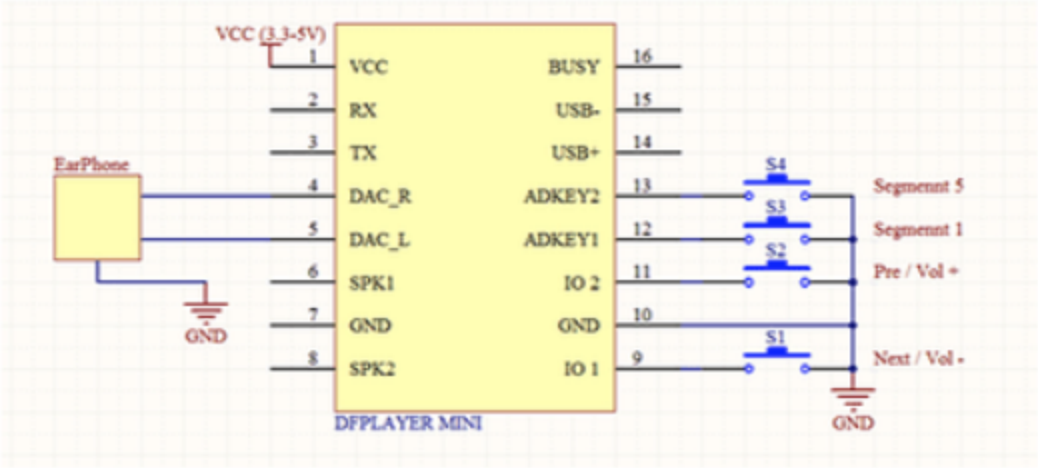


Pin	Description	Note
VCC	Input Voltage	DC3.2~5.0V;Type: DC4.2V
RX	UART serial input	
TX	UART serial output	
DAC_R	Audio output right channel	Drive earphone and amplifier
DAC_L	Audio output left channel	Drive earphone and amplifier
SPK2	Speaker-	Drive speaker less than 3W
GND	Ground	Power GND
SPK1	Speaker+	Drive speaker less than 3W
IO1	Trigger port 1	Short press to play previous（long press to decrease volume）
GND	Ground	Power GND
IO2	Trigger port 2	Short press to play next（long press to increase volume）
ADKEY1	AD Port 1	Trigger play first segment
ADKEY2	AD Port 2	Trigger play fifth segment
USB+	USB+ DP	USB Port
USB-	USB- DM	USB Port
BUSY	Playing Status	Low means playing \High means no

Simple Configuration

This configuration is the simplest way to run this board.





Running from a Micro Controller

Using a micro-controller with serial support will allow you to take full advantage of this board.

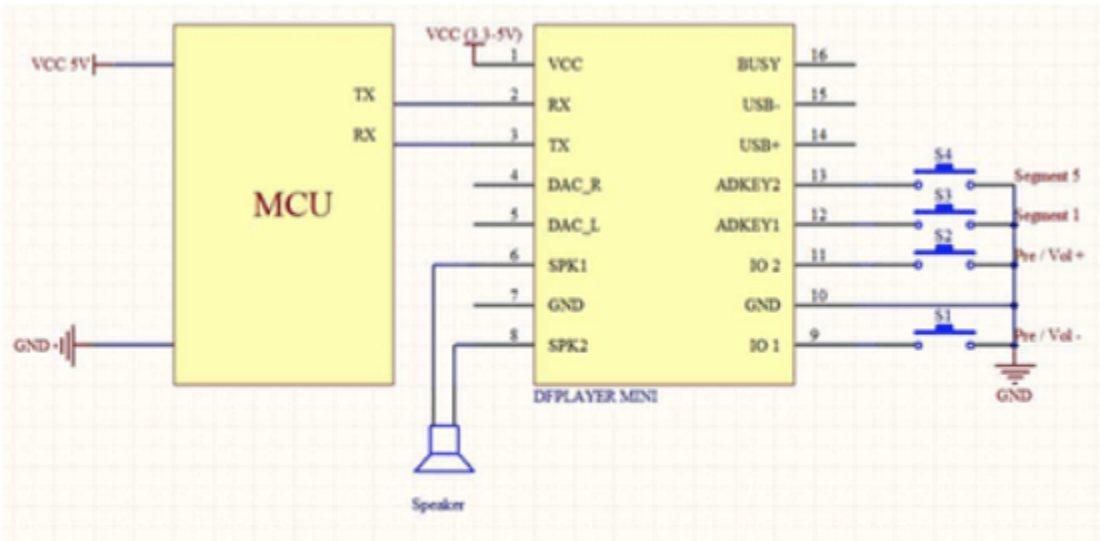


Figure 4.2 Serial Connect (5v)

In the above diagram, the TX/RX of the micro-controller are connected to the TX/RX of the DFPlayer board in order to take advantage of the more advanced functionality of the board.

Full Sample Code

```
/**************************************************
 DFPlayer - A Mini MP3 Player For Arduino
 <https://www.dfrobot.com/index.php?route=product/product&product_id=1121>

 **************************************************
 This example shows the all the function of library for DFPlayer.

 Created 2016-12-07
 By [Angelo qiao](Angelo.qiao@dfrobot.com)
```

```
 GNU Lesser General Public License.
 See <http://www.gnu.org/licenses/> for details.
 All above must be included in any redistribution
 ****************************************************/

/***********Notice and Trouble shooting***************
 1.Connection and Diagram can be found here
<https://www.dfrobot.com/wiki/index.php/DFPlayer_Mini_SKU:DFR0299#Connection_Diagram>
 2.This code is tested on Arduino Uno, Leonardo, Mega boards.
 ****************************************************/

#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.setTimeOut(500); //Set serial communictaion time out 500ms

  //----Set volume----
  myDFPlayer.volume(10);  //Set volume value (0~30).
  myDFPlayer.volumeUp(); //Volume Up
  myDFPlayer.volumeDown(); //Volume Down

  //----Set different EQ----
  myDFPlayer.EQ(DFPLAYER_EQ_NORMAL);
//  myDFPlayer.EQ(DFPLAYER_EQ_POP);
//  myDFPlayer.EQ(DFPLAYER_EQ_ROCK);
//  myDFPlayer.EQ(DFPLAYER_EQ_JAZZ);
//  myDFPlayer.EQ(DFPLAYER_EQ_CLASSIC);
//  myDFPlayer.EQ(DFPLAYER_EQ_BASS);

  //----Set device we use SD as default----
//  myDFPlayer.outputDevice(DFPLAYER_DEVICE_U_DISK);
  myDFPlayer.outputDevice(DFPLAYER_DEVICE_SD);
//  myDFPlayer.outputDevice(DFPLAYER_DEVICE_AUX);
//  myDFPlayer.outputDevice(DFPLAYER_DEVICE_SLEEP);
//  myDFPlayer.outputDevice(DFPLAYER_DEVICE_FLASH);

  //----Mp3 control----
//  myDFPlayer.sleep();     //sleep
//  myDFPlayer.reset();     //Reset the module
//  myDFPlayer.enableDAC();  //Enable On-chip DAC
//  myDFPlayer.disableDAC();  //Disable On-chip DAC
//  myDFPlayer.outputSetting(true, 15); //output setting, enable the output and set the gain to 15

  //----Mp3 play----
  myDFPlayer.next();  //Play next mp3
  delay(1000);
  myDFPlayer.previous();  //Play previous mp3
  delay(1000);
```

```
  myDFPlayer.play(1);  //Play the first mp3
  delay(1000);
  myDFPlayer.loop(1);  //Loop the first mp3
  delay(1000);
  myDFPlayer.pause();  //pause the mp3
  delay(1000);
  myDFPlayer.start();  //start the mp3 from the pause
  delay(1000);
  myDFPlayer.playFolder(15, 4);  //play specific mp3 in SD:/15/004.mp3; Folder Name(1~99); File Name(1~255)
  delay(1000);
  myDFPlayer.enableLoopAll(); //loop all mp3 files.
  delay(1000);
  myDFPlayer.disableLoopAll(); //stop loop all mp3 files.
  delay(1000);
  myDFPlayer.playMp3Folder(4); //play specific mp3 in SD:/MP3/0004.mp3; File Name(0~65535)
  delay(1000);
  myDFPlayer.advertise(3); //advertise specific mp3 in SD:/ADVERT/0003.mp3; File Name(0~65535)
  delay(1000);
  myDFPlayer.stopAdvertise(); //stop advertise
  delay(1000);
  myDFPlayer.playLargeFolder(2, 999); //play specific mp3 in SD:/02/004.mp3; Folder Name(1~10); File Name
(1~1000)
  delay(1000);
  myDFPlayer.loopFolder(5); //loop all mp3 files in folder SD:/05.
  delay(1000);
  myDFPlayer.randomAll(); //Random play all the mp3.
  delay(1000);
  myDFPlayer.enableLoop(); //enable loop.
  delay(1000);
  myDFPlayer.disableLoop(); //disable loop.
  delay(1000);

  //----Read imformation----
  Serial.println(myDFPlayer.readState()); //read mp3 state
  Serial.println(myDFPlayer.readVolume()); //read current volume
  Serial.println(myDFPlayer.readEQ()); //read EQ setting
  Serial.println(myDFPlayer.readFileCounts()); //read all file counts in SD card
  Serial.println(myDFPlayer.readCurrentFileNumber()); //read current play file number
  Serial.println(myDFPlayer.readFileCountsInFolder(3)); //read fill counts in folder SD:/03
}

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

  if (millis() - timer > 3000) {
    timer = millis();
    myDFPlayer.next();  //Play next mp3 every 3 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;
  }
}
```

Copying Mp3s to your Micro SD Card

Format your micro SD card as FAT32.

Copy the mp3 files to the SD Card

```
cp *.mp3 /Volumes/<VOLUME_NAME>/.
```

Remove all unneeded files if on a MAC

```
dot_clean /Volumes/<VOLUME_NAME>
```

NOTE: The order you copy the mp3 into micro SD card will affect the order mp3 played , which means play(1) function will play the first mp3 copied into micro SD card.

For Mac User

NOTE: If you are using Mac OS X to copy the mp3, the file system will automatically add hidden files like: "._0001.mp3" for index, which this module will handle as valid mp3 files. It is really annoying. So you can run following command in terminal to eliminate those files.

```
dot_clean /Volumes/<SDVolumeName>
```

Please replace the to the volume name of your SD card.

Reference

Reference	URL
DFPlayer	https://wiki.dfrobot.com/DFPlayer_Mini_SKU_DFR0299
Manual	https://www.dfrobot.com/wiki/index.php/DFPlayer_Mini_SKU:DFR0299
Manual	http://www.picaxe.com/docs/by8001.pdf
Example for ESP32	https://github.com/pcbreflux/espressif/blob/master/esp32/arduino/sketchbook/ESP32_DFPlayer_full/ESP32_DFPlayer_full.ino
A great & very cheap MP3 sound module without need for a library!	https://community.particle.io/t/a-great-very-cheap-mp3-sound-module-without-need-for-a-library/20111/24