# Модуль воспроизведения звука для озвучивания робототехнических проектов



В магазинах этот модуль можно найти под двумя маркировками: DFPlayer MP3 Mini и FN-M16P. Сути однако это не меняет. Модуль достаточно распространенный и в сети о нем очень много информации. Однако, в этой информации столь же много неточностей. А в библиотеках, которые я встречал для Arduino очень много ошибок.

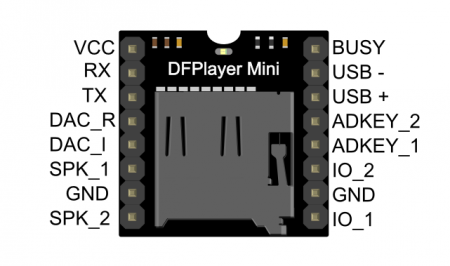
Мне удалось найти один из наиболее полных даташитов на него. По нему я и сделал свою библиотеку. Опять же на полноту она не претендует, но призвана решать выше означенную задачу – создание непрерывного звукового сопровождения для проекта.

## Основные характеристики модуля

* Воспроизведение файлов в форматах mp3 и wav
* Поддержка файловых систем FAT16 и FAT32
* Поддержка SD карт памяти объемом до 32Гб
* Управление громкостью воспроизведения
* Эквалайзер с шестью предустановленными режимами
* Возможность воспроизведения с карты памяти microSD и внешнего USB Flash носителя

Есть еще возможность работы в автономном режиме с кнопочным управлением. Но в свете решаемой задачи, этот режим меня совершенно не интересует. Поэтому и освещаться не будет.

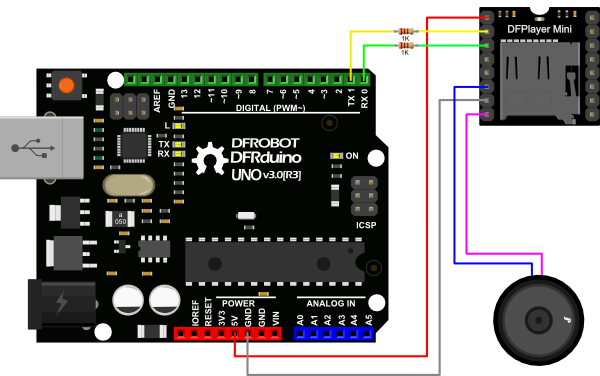
## Распиновка



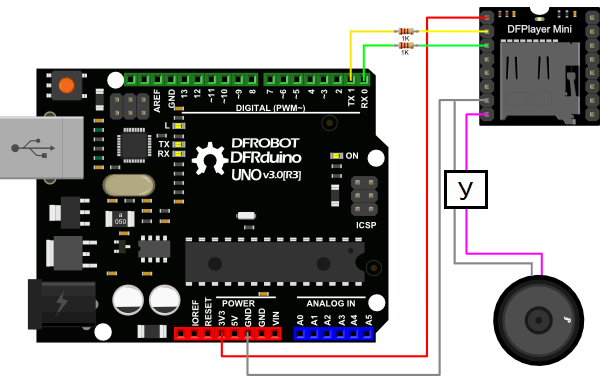
|  |  |  |  |
| --- | --- | --- | --- |
| № | Обозначение | Назначение | Примечание |
| 1 | VCC | Напряжение питания 3.2~5.0В | Лучше питать от 3.3В |
| 2 | RX | UART serial input | Скорость порта 9600 бод |
| 3 | TX | UART serial output |
| 4 | DAC\_R | Линейный выход; правый канал |  |
| 5 | DAC\_L | Линейный выход; левый канал |  |
| 6 | SPK2 | Выход на динамик; левый канал | Лучше подключать наушники |
| 7 | GND | «Земля» | Общий контакт для подключения динамиков |
| 8 | SPK1 | Выход на динамик; правый канал | Лучше подключать наушники |
| 9 | IO1 |  | Кнопочное управление |
| 10 | GND | «Земля» |  |
| 11 | IO2 |  | Кнопочное управление |
| 12 | ADKEY1 |  | Кнопочное управление |
| 13 | ADKEY2 |  | Кнопочное управление |
| 14 | USB+ | USB+ DP | Подключение к ПК или внешний накопитель |
| 15 | USB- | USB- DM |
| 16 | Busy | Индикатор состояния | LOW – идет воспроизведение; HIGH - ожидание |

К выходам SPK1 и SPK2 лучше подключать только наушники. Динамики же стоит подключать к линейному выходу через усилитель.

В инете везде можно найти вот такую схему подключения:



В таком виде звука от плеера вы никогда не услышите. Схема должна быть такой:



На 5В, без резисторов и усилителя тоже работает, но «хрипит» и сам модуль сильно греется. Решил не выяснять, чем это может кончиться.

В схеме показано подключение к аппаратному UART, но в своих проектах я использую этот модуль с программным портом (SoftwareSerial).

## Формат передачи команд

Команда передается модулю в виде массива из 10 байт

|  |  |  |
| --- | --- | --- |
| № | Наименование | Назначение |
| 0 | $S | Стартовый байт = 0x7E |
| 1 | Ver. | Версия. По умолчанию = 0xFF |
| 2 | Number | Кол-во байт в пакете от Ver. до Param\_LSB. Должно быть равно 0x06. |
| 3 | Command | Команда |
| 4 | Feedback | 0x01 – ожидаем ответ от модуля; 0x00 – нет. |
| 5 | Param\_MSB | Старший байт параметра команды |
| 6 | Param\_LSB | Младший байт параметра команды |
| 7 | Check\_MSB | Старший байт контрольной суммы |
| 8 | Check\_LSB | Младший байт контрольной суммы |
| 9 | $O | Завершающий байт = 0xEF |

Полный список поддерживаемых команд перечислять не буду. Все их можно посмотреть в найденном мной даташите.

## Функции библиотеки

### void init (SoftwareSerial\*);

Инициализация объекта. Необходимо вызвать один раз перед первым обращением к модулю. В виде параметра передается порт.

### void playNext();

Перейти к проигрыванию следующего трека.

### void playPrev();

Перейти к проигрыванию предыдущего трека.

### void playTrackPhisical(uint16\_t track\_num);

Проиграть трек с порядковым номером track\_num. Надо сказать, что треки на карте памяти могут иметь любые имена. Под порядковым номером в данном случае подразумевается положение файла в таблице FAT карты памяти. Фактически это порядок записи файлов на карту. Его главное преимущество в скорости. Проигрывание файла начинается практически мгновенно без пауз и задержек.

### void playRepeatSingle(uint16\_t track\_num);

Равно как и предыдущий метод, проигрывает трек с порядковым номером track\_num. Как не трудно догадаться, трек проигрывается по кругу без остановки пока не будет передана команда stop. Сигнал на выходе Bisy при этом появится только в момент остановки трека.

### void playFolder(uint8\_t folder, uint8\_t track);

Проиграть заданный трек из определенной папки. Производитель рекомендует именовать папки двузначными числами. Допустимо использовать 99 папок, в каждой из которых не более 255 файлов. Файлы при этом сортируются по имени в алфавитном порядке. Номер трека указывается в соответствии с этой сортировкой.

### void playFolder2(uint8\_t folder, uint16\_t track);

Аналогично предыдущему методу, проигрывает заданный трек из определенной папки. Допустимо использовать не более 15 папок, в каждой из которых до 3000 файлов. Файлы при этом сортируются по имени в алфавитном порядке. Номер трека указывается в соответствии с этой сортировкой.

### void play();

Продолжить проигрывание файла. Например, после паузы.

### void pause();

Приостановить воспроизведение.

### void stop();

Остановка воспроизведения.

### void playAllRepeat(bool start);

Проигрывать все треки по кругу из корневой папки.

### void playMP3Folder(uint16\_t track);

Проиграть трек по номеру. Звуковые файлы при этом размещаются в папке с именем «mp3». Номер трека соответствует порядковому номеру файла при сортировке по имени в алфавитном порядке. Допустимо иметь в папке до 65536 треков.

### void playFolderRepeat(uint8\_t folder);

Проигрывать по кругу все треки из заданной папки. Производитель рекомендует именовать папки двузначными числами.

### void playRandom();

Проигрывание случайным образом выбранных треков.

### void repeatCurrent(bool start);

Проигрывать по кругу текущий трек. Метод вызывается пока трек воспроизводится.

### void volumeSet(uint8\_t val);

Установка громкости. Допустимо значение в интервале 0 – 30. При включении модуля устанавливается громкость 30.

### void setEQ(uint8\_t val);

Установка эквалайзера:

MP3\_EQ\_NORMAL 0

MP3\_EQ\_POP 1

MP3\_EQ\_ROCK 2

MP3\_EQ\_JAZZ 3

MP3\_EQ\_CLASSIC 4

MP3\_EQ\_BASS 5

### void setInterCut(uint16\_t track);

Очень интересная функция. Фактически ставит текущий проигрываемый трек на паузу и проигрывает заданный. Но чтобы она правильно работала, на карте надо создать папку «ADVERT», куда поместить копии всех файлов в том же порядке, что и в корневой папке. На выводе bisy появится сигнал только в момент приостановки основного трека. После проигрывания заданного трека модуль продолжит проигрывание приостановленного автоматически. Никаких сообщений на этот счет микроконтроллеру не будет.

### void stopInterCut();

Функция остановки для предыдущей функции