베어팹

아두이노

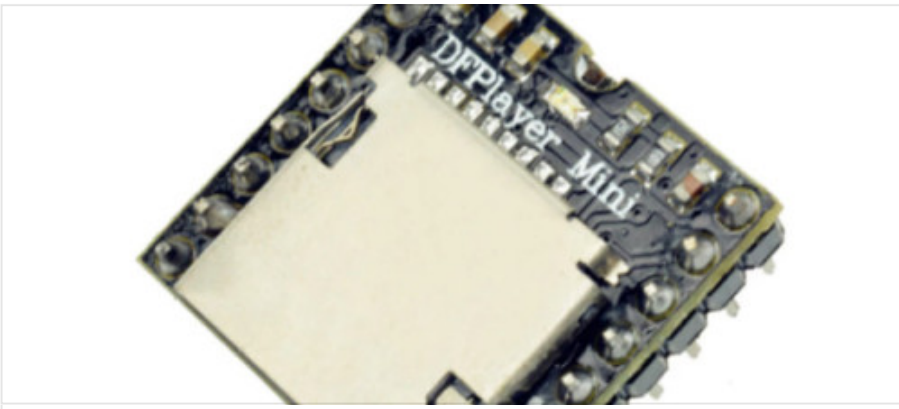
아두이노로 MP3 모듈 제어하기(DFPlayer Mini)

황제곰

2021. 1. 5. 21:54

이웃추가

안녕하세요. 이번 포스팅에서는 소형 사이즈의 MP3 디코딩 모듈(재생모듈)인 DFPlayer Mini라는 모듈을 아두이노로 제어하는 방법에 대해서 알아 보도록 하겠습니다.



아두이노 MP3 플레이어 모듈<DFPlayer Mini> : 베어팹몰

아두이노 MP3 플레이어 모듈<DFPlayer Mini>

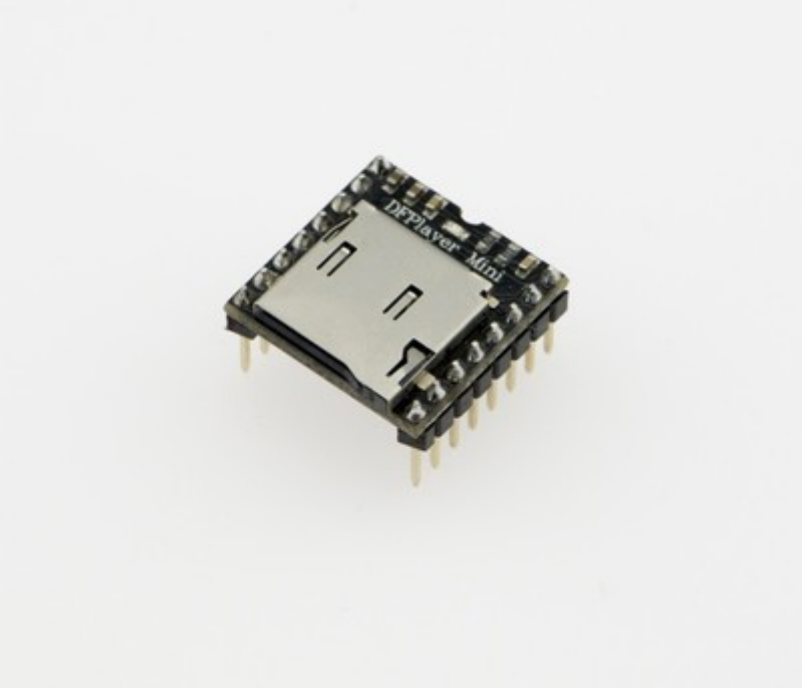
smartstore.naver.com

DFPlayer Mini

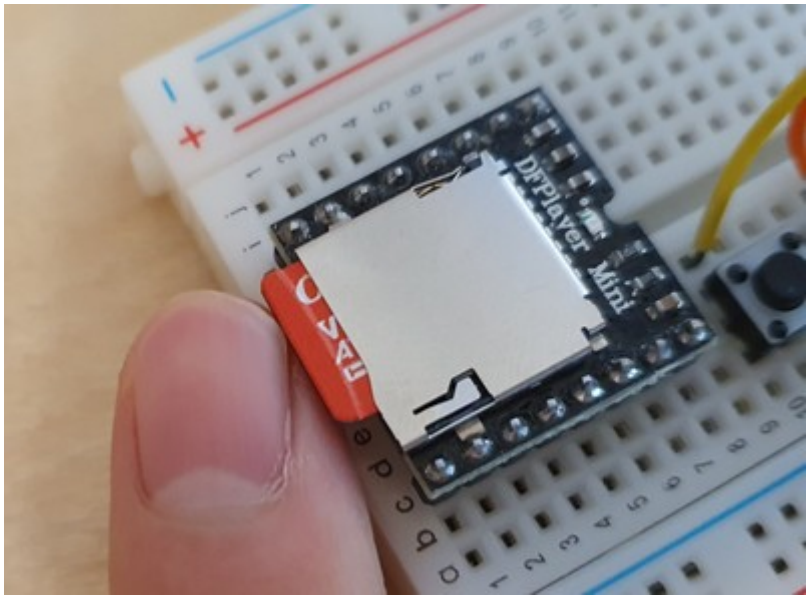
- DFPlayer Mini 모듈 소개 -

2 18

베어팹



DFPlayer 미니 모듈은 mp3 음악파일을 스피커나 이어폰 잭으로 출력해주는 기능을 갖춘 소형 디코딩 모듈인데요. 사이즈가 작고 가격도 저렴한 편이라서 아두이노로 음악관련 프로젝트를 진행할 때 손쉽게 적용이 가능한 모듈입니다.



작은사이즈

사실 마이크로 컨트롤러가 있어야만 이 모듈을 사용할 수 있는건 아니고 저항과 스위치만 몇개 연결하면 별도의 마이크로 컨트롤러 없이 바로 음악 재생 및 몇몇 기능(다음곡, 이전곡 넘기기, 볼륨 조절 등)은 사용이 가능하긴 합니다. 하지만 마이크로 컨트롤러와 시리얼

베어팹

- 모듈 스펙 및 기능 -

데이터 시트에서 제공하고 있는 모듈의 스펙은 아래와 같습니다.

▶ 샘플링속도 (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

▶ FAT16, FAT32파일 시스템, 32기가 TF카드, 32기가 Udisk, 64MB의 NORFLASH 지원

▶ 다양한 제어방법 지원: 입출력 컨트롤제어, 시리얼 제어, AD버튼 제어

▶ 어드벌타이즈 모드 지원

▶ 폴더별로(100개까지) 255개의 노래를 저장할 수 있음.

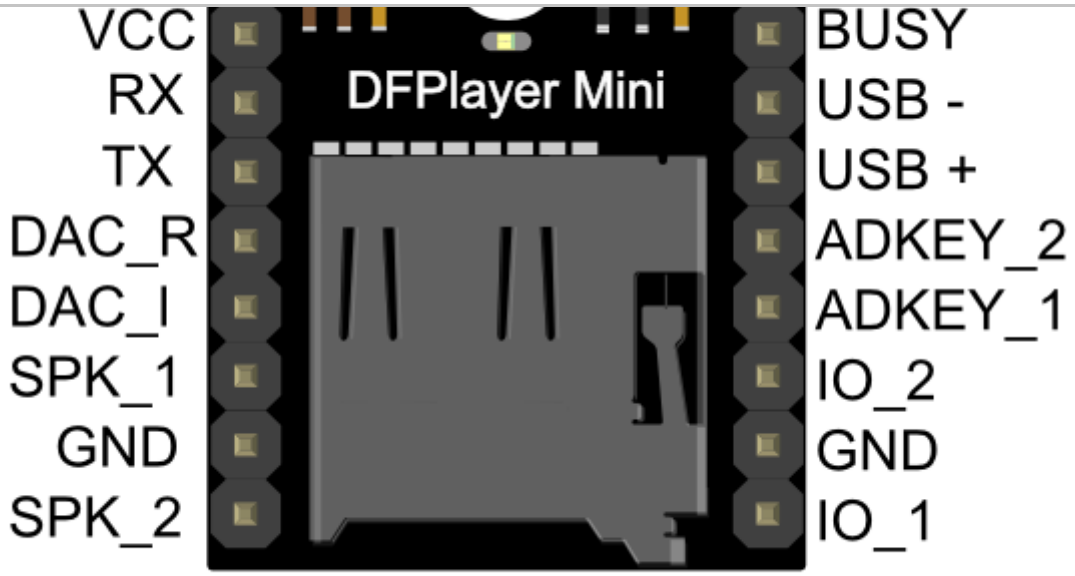
▶ 30단계로 볼륨조절 가능, 6개의 이퀄라이즈 모드 변경 가능

SD카드는 FAT16과 FAT32파일 시스템을 지원하기 때문에 음악을 넣기 전에 해당 포맷으로 먼저 포맷을 해 주어야 합니다.

- 핀맵 및 기능 -

모듈에 사용되는 핀 맵과 기능은 아래그림 및 표와 같습니다.

베어팹



번호	핀	핀 기능	설명
1	VCC	입력 전압	DC3.2V~5V 입력 가능
2	RX	UART 입력	시리얼 통신 입력핀
3	TX	UART 출력	시리얼 통신 출력 핀
4	DAC_R	오디오 출력 오든쪽	이어폰이나 앰프 사용 출력시 사용
5	DAC_L	오디오 출력 왼쪽	
6	SPK_1	스피커 -	스피커 2개의 선 중 하나 연결
7	GND	Ground	전원 GND 연결
8	SPK_2	스피커 +	스피커 2개의 선 중 하나 연결
9	IO_1	간편 입력 1	버든 연결후 클릭시 이전 곡재생 (길게 누르고 있으면 볼륨 감소)
10	GND	Ground	전원 GND 연결
11	IO_2	간편 입력 2	버든 연결후 클릭시 이전 곡재생

13	ADKEY2	AD키 포트 2	AD키로 수동 제어시 사용
14	USB+	USB+ DP	USB용 포트(이 모듈에서는 SD카드를 사용하므로 거의 사용할 일이 없음.)
15	USB-	USB+ DM	USB용 포트(이 모듈에서는 SD카드를 사용하므로 거의 사용할 일이 없음.)
16	BUSY	재생 여부 표시 핀	노래 재생 중일 때LOW, 정지시 HIGH 출력

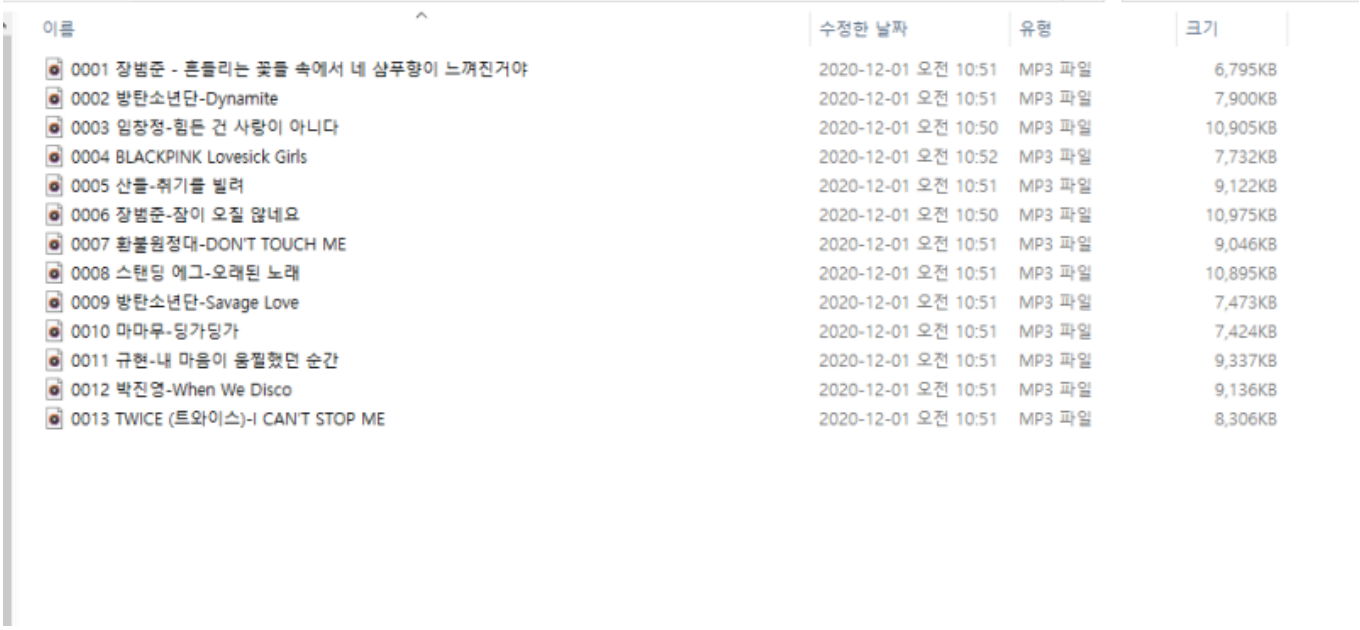
- SD카드에 노래 넣기 -

SD카드에 파일을 넣기 전에 먼저 SD카드의 파일 포맷을 FAT16또는 FAT 32로 포맷을 해 줍니다. 포맷이 된 SD 카드에 mp3파일을 넣으면 되는데요. 음악을 넣는 방법이 4가지 방법이 있습니다. **특히 아두이노로 모듈을 제어할 경우 이 mp3파일을 넣는 방법에 따라 제어 방법(사용함수)이 달라지며, 라이브러리를 사용할 때 제어 함수를 잘 못 사용할 경우(예: mp3폴더에서 mp3파일을 재생하는 함수를 일반 폴더에 보관후 사용했을 경우 등등...) 모듈이 정상 동작하지 않으므로 mp3파일 저장 방법에 따른 재생방법(함수)을 잘 확인해야 합니다.**

▶ **"mp3"** 폴더에 파일 넣기

SD카드의 최상위 위치에서 **"mp3"** 라는 폴더를 만들고 그 하위에 mp3파일을 넣는 방법입니다. 이 때 음악 파일의 이름은 **"nnnnXXXX"**의 형식이어야 합니다. 여기서 nnnn은 숫자 0001부터 9999까지의 값이고 XXXX 파일이름이며 어떤 값이 들어가도 괜찮습니다. mp3 폴더내에 아래 사진과 같이 넣으면 됩니다.

베어팹



이름	수정한 날짜	유형	크기
0001 장범준 - 흔들리는 꽃들 속에서 네 샴푸향이 느껴진거야	2020-12-01 오전 10:51	MP3 파일	6,795KB
0002 방탄소년단-Dynamite	2020-12-01 오전 10:51	MP3 파일	7,900KB
0003 임창정-힘든 건 사랑이 아니다	2020-12-01 오전 10:50	MP3 파일	10,905KB
0004 BLACKPINK Lovesick Girls	2020-12-01 오전 10:52	MP3 파일	7,732KB
0005 산들-숨기를 빌려	2020-12-01 오전 10:51	MP3 파일	9,122KB
0006 장범준-잠이 오질 않네요	2020-12-01 오전 10:50	MP3 파일	10,975KB
0007 환불원정대-DON'T TOUCH ME	2020-12-01 오전 10:51	MP3 파일	9,046KB
0008 스탠딩 에그-오래된 노래	2020-12-01 오전 10:51	MP3 파일	10,895KB
0009 방탄소년단-Savage Love	2020-12-01 오전 10:51	MP3 파일	7,473KB
0010 마마무-딩가딩가	2020-12-01 오전 10:51	MP3 파일	7,424KB
0011 규현-내 마음이 움찔했던 순간	2020-12-01 오전 10:51	MP3 파일	9,337KB
0012 박진영-When We Disco	2020-12-01 오전 10:51	MP3 파일	9,136KB
0013 TWICE (트와이스)-I CAN'T STOP ME	2020-12-01 오전 10:51	MP3 파일	8,306KB

mp3 폴더 아래 음악파일 저장

이렇게 mp3폴더내에 파일이 있을 때는 아래 라이브러리 사용법에서 살펴보게 되겠지만 mp3폴더내의 음악 재생함수는 **playMp3Folder(int fileNumber);** 를 사용합니다.

```
//사용예시
player.playMp3Folder(1);     //mp3 폴더내의 1번 파일 재생

player.playMp3Folder(2);

player.playMp3Folder(211);    //mp3 폴더내의 211번 파일 재생
```

▶ 2자리 숫자로 이루어진 폴더에 음악 파일 넣기(폴더당 최대 999개 저장 가능)

음악을 저장하는 또다른 방법은 아래 사진처럼 2자리의 숫자로 이루어진 폴더를 만들고 해당 폴더 내에 음악파일을 저장하는 것입니다. 폴더내의 음악파일은 **"nnnXXX.mp3"** 의 형식이어야 합니다. 여기서 nnn는 숫자 001에서 999까지의 값이고 XXX는 파일이름으로 어떤 값이 들어가도 괜찮습니다.

베어팹

폴더

음악파일

이 방법으로 음악을 저장했을 때에는 라이브러리를 사용할 때는 **"playFolder(uint8_t folderNumber, uint8_t fileNumber)"** 함수를 사용하면 됩니다.

```
//사용예시

player.playFolder(7, 1);     // 7번 폴더의 1번 파일 재생
player.playFolder(7, 2);
player.playFolder(7, 211);   //7번 폴더의 211번 파일 재생
```

▶ 2자리 숫자로 이루어진 폴더에 음악 파일 넣기(폴더당 최대 9999개 저장 가능)

만약 폴더구조로 음악을 넣고 싶은데 음악의 개수가 999개 이상일 경우는 폴더내의 음악파일의 이름을 **"nnnnXXX.mp3"** 의 형식으로 저장하면 됩니다. 여기서 nnnn은 0001에서 9999까지의 숫자이고 XXX는 파일이름으로 어떤 값이 들어가도 됩니다.

베어팹

예시

이럴 경우 폴더의 음악을 재생할 때는 **"playLargeFolder(uint8_t folderNumber, uint16_t fileNumber)"** 함수를 사용합니다.

```
//사용예시
player.playLargeFolder(5, 1);     //5번 폴더의 1번 파일 재생
player.playLargeFolder(5, 2);
player.playLargeFolder(5, 211);   // 5번 폴더의 211번 파일 재생
```

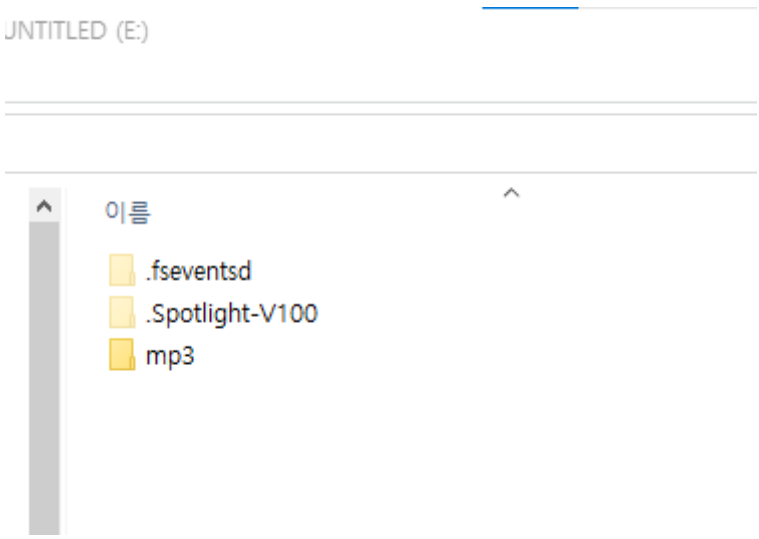
▶ 임의의 위치에 파일 넣기

이 방법은 위 방법들에도 적용되는 방법입니다. 파일을 폴더에 넣어도 되고 그냥 SD카드의 최상위 폴더에 넣어도 됩니다.(파일이름도 상관없습니다.) 라이브러리의 **play( )** 함수를 쓰면 SD카드에 mp3파일이 복사된 순서대로 음악파일이 재생됩니다. 파일이름에 상관없이 SD카드에 저장된 순서대로 내부적으로 임의의 인덱스를 생성되어 그 순서대로 음악이 재생됩니다.

2 18

베어팹

음악 파일을 넣을 때 한가지 주의 해야 할 점이 있는데요. 음악파일을 번호를 잘 지정해서 폴더내 넣었을 때에도 특정운영체제(특히 맥)에서는 음악파일이 저장된 순서에 따라 별도의 인덱스 파일이나 숨김파일이 생성되는 경우가 있습니다. 이런 인덱스 파일이나 숨김파일이 생기면 내가 번호를 매긴 파일이름보다 숨김파일이 먼저 적용되어 내가 원하는 번호대로 음악이 재생되지 않습니다. 그럴 경우는 숨김파일을 표시해서 해당 파일과 인덱스 파일을 삭제 해 주시면 됩니다.



.으로 시작하는 인덱스 파일을 삭제 합니다

마이크로컨트롤러(아두이노) 없이 모듈 사용하기

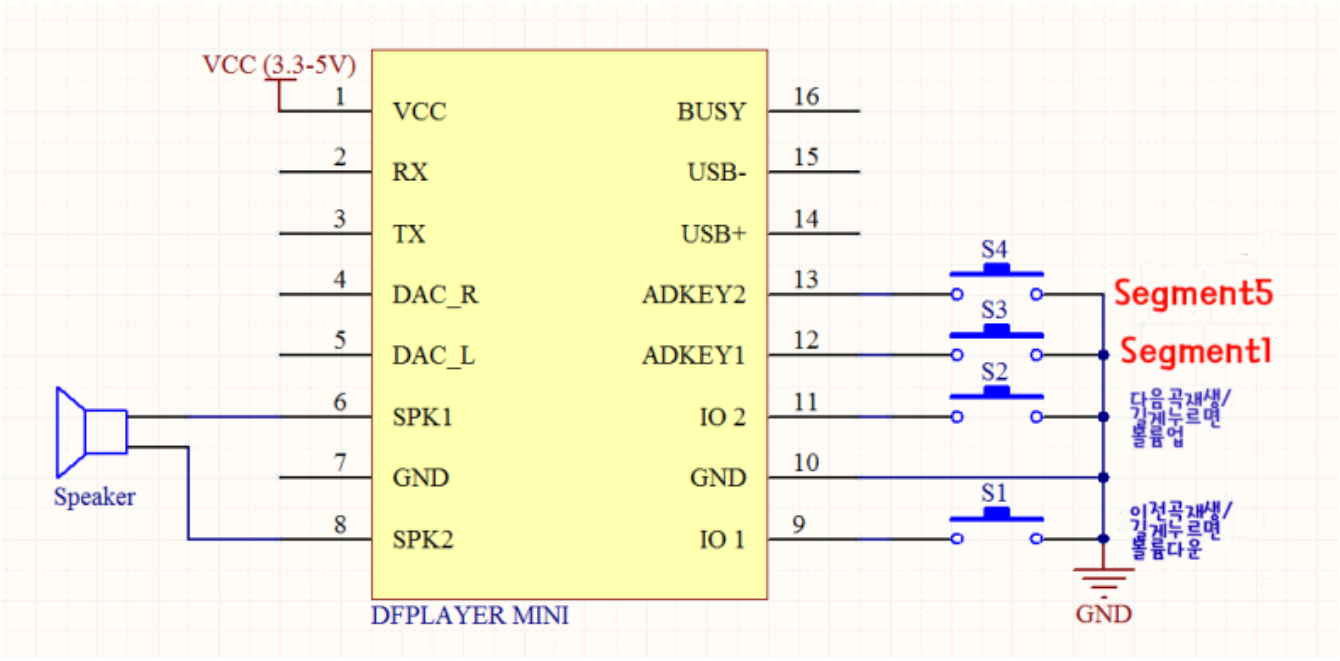
위에서 언급했다시피 이 모듈은 꼭 외부의 마이크로컨트롤러가 있어야만 음악재생을 할 수 있는 것은 아닙니다. 독립된 모듈로 음악을 재생하기 위한 방법이 2가지가 있는데요.

- IO 모드로 사용하기 -

아래와 같이 회로를 연결하면 간단히 4개의 버튼으로 음악을 재생할 수 있습니다. segment1과 연결된 버튼을 누르면 1번 음악파일(첫번째 복사된 파일)이 재생되고 segment5와 연결된 버튼을 누르면 5번 파일(5번째 복사된 파일)이 재생됩니다. S1, S2를

베어팹

- Refer diagram



아래는 I/O 모드로 음악 재생하는 영상입니다. 음질이 나쁘지 않네요^^(모듈도 모듈이지만 스피커도 음질에 많은 영향을 주는 것 같네요.)

DFPlayer Mini I/O 모드

561 2

00:00 00:51 자동

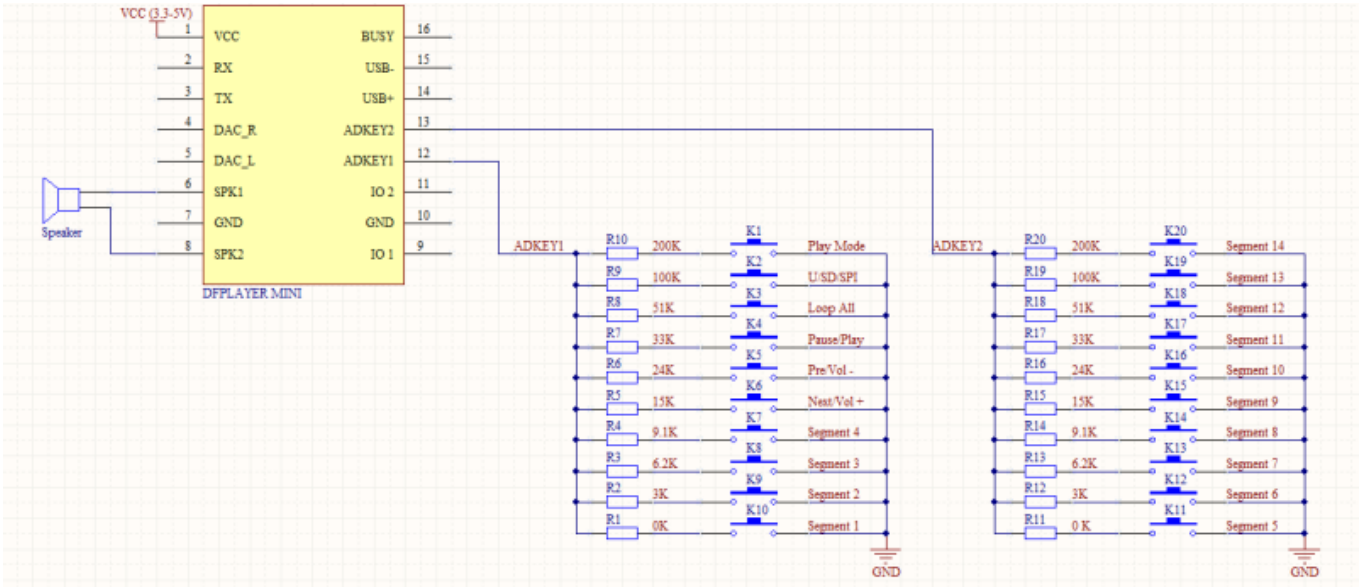
DFPlayer Mini I/O 모드

2 18

베어팹

- AD 키 모드로 사용하기 -

위에서 사용했던 I/O 모드는 간단한 음악 재생 및 볼륨 조절 기능만 사용할 수 있었는데요. AD키 모드로 사용하면 더 많은 기능을 사용할 수 있습니다. 아래 사진과 같이 AD키용 핀 2개에 AD키패드를 연결해 주면 더 많은 기능을 사용할 수 있습니다.



AD 키패드는 아래 사진과 같은 모양의 부품입니다. 각 키에 다른 값의 저항과 연결되어 있어서 이 저항값을 바탕으로 어떤 키가 눌렸는지를 감지할 수 있는 부품입니다.

베어팹

만약 키패드가 없으면 위 회로도에 나온 저항값과 일반 택트스위치를 연결해서 사용해도 동일한 효과를 발휘할 수 있습니다.

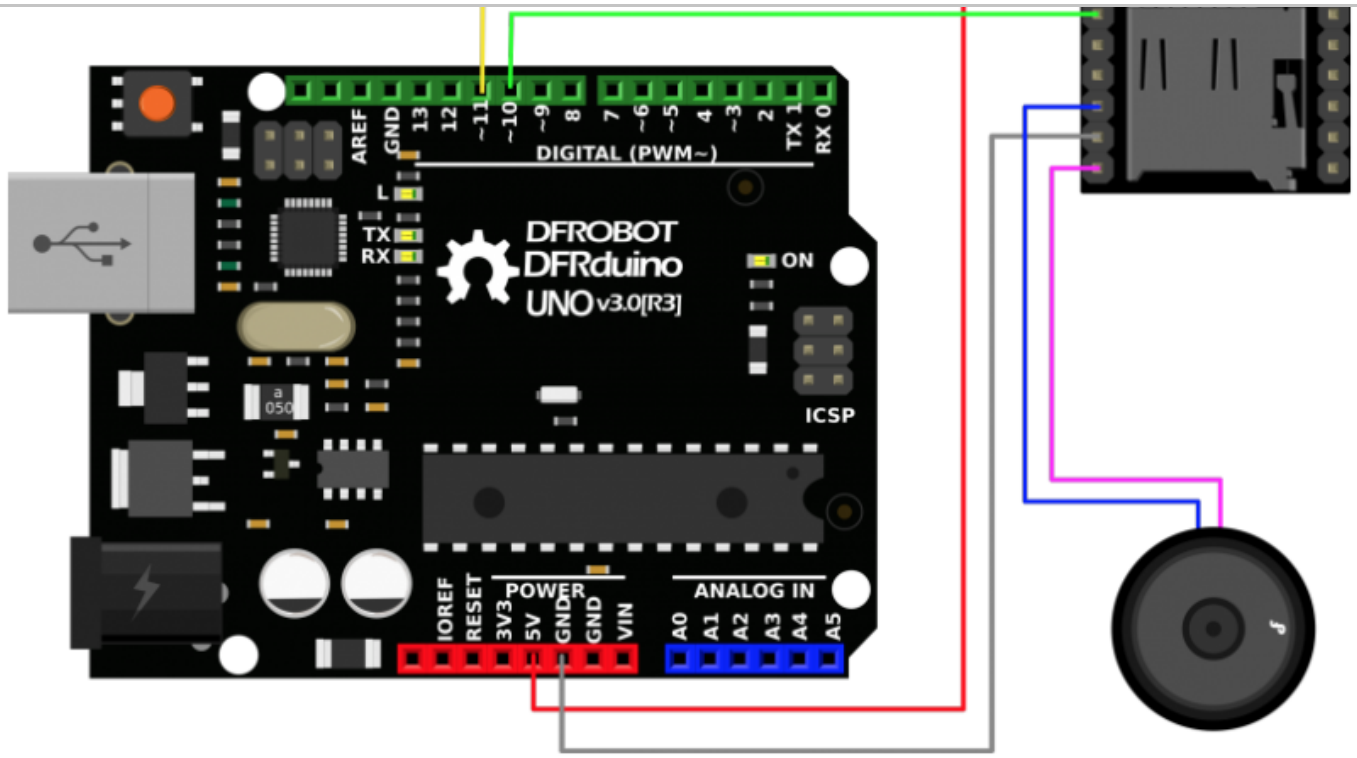
아두이노로 제어하기

사실 요즘은 대부분 스마트폰을 사용해서 음악을 듣기 때문에 위에서처럼 별도로 mp3모듈로 음악을 재생할 일은 별로 없을 것 같네요. 이제 아두이노로 모듈을 제어하는 방법에 대해서 알아 보겠습니다.

- 회로연결 -

회로는 아래 사진과 같이 연결해 주면 되는데요. 아두이노(우노)의 0번 1번은 코드업로드 시에 사용되기 때문에 별도의 핀에 소프트웨어 시리얼 통신을 할 수 있도록 연결해 주면 됩니다. 저는 예시에 나온대로 10번, 11번에 연결해 주었습니다.(소프트웨어 시리얼핀은 다른 핀으로 지정해도 괜찮습니다.)

베어팹



DFMINI Player(핀번호)	아두이노(핀번호)
VCC(1)	5.0V 또는 3.3V
RX(2)	소프트웨어시리얼 TX핀(11번)
TX(3)	소프트웨어시리얼 RX핀(10번)
SPK1(6)-스피커와 연결	
SPK2(8)-스피커와 연결	
Ground(7)	아두이노의 GND와 연결

*** 참고 ***

▶ 데이터시트에 노이즈 발생시 위와 같이 1K옴을 아두이노의 소프트웨어 TX핀과 모듈의

베어팹

연결하면 오류가 발생하기 때문인 것으로 보입니다. 3.3V 신호레벨을 사용하는 아두이노의 경우는 별도의 저항 연결없이 통신이 가능합니다.

▶ mp3 모듈에 전원이 인가되면 mp3모듈이 초기화되고 음악을 재생할 수 있는 준비를 하게 됩니다. 이 시간이 대략 500ms에서 1500ms(sd카드에 보관한 mp3파일이 많은 경우) 걸리고 준비가 되면 모듈에서 시리얼 통신으로 준비가 되었다는 메시지를 송신합니다. 따라서 아두이노에서 모듈을 제어하려면 이 준비시간 동안 기다려 주거나 시리얼 통신으로 모듈로부터 들어오는 수신메시지를 확인해서 모듈이 준비가 되었을 때 제어를 시작해야 합니다. 또한 모듈이 잘못된 제어명령어를 받거나 전력부족 등 여러가지 이유로 시리얼명령어를 수신하지 못하는 상태가 발생할 수 있습니다. 이 때에도 모듈의 전원을 뺐다가 다시 인가해 주면 모듈이 준비상태로 돌아가서 다시 명령어를 수신할 수 있는 상태가 됩니다.(만약에 이 모듈로 케이스가 있는 프로젝트에 활용하려면 모듈에 전원리셋이 가능하도록 별도의 스위치를 빼 주는 것이 좋을 듯 합니다....)

- 시리얼명령어를 통한 제어방법 -

* 이 부분은 동작원리에 대한 설명으로 바로 사용하는 것이 목적이신 분들은 아래 '라이브러리를 통한 제어' 로 넘기셔도 됩니다. *

데이터 시트에 보면 어떻게 모듈을 제어할 수 있는지에 대해서 자세히 나와 있는데요. 시리얼 통신으로 9600bps의 속도로 아래 명령어 구문을아두이노에서 송신하면 mp3모듈에서는 수신된 명령어구문을 해독해서 지정된 모드로 동작하게 됩니다.

명령어 구문의 형식은 아래와 같이 10개의 요소로 이루어져 있습니다.

각각의 요소에 대한 더 자세한 내용은 아래 표와 같습니다.

번호	요소	설명

베어팹

②	VER(버전)	버전 입력 기본값은 0xFF
③	LEN(길이)	전체 명령어 구문의 길이 입력, 시작바이트, 종료바이트, 체크섬은 빼기 때문에 값은 0x06
④	CMD(명령어)	MP3 모듈에 지정할 명령어 입력, 명령어의 내용은 하단의 표 참고
⑤	FEEDBACK(피드백여부)	명령어 구문 송신후 수신확인 피드백을 받을 것인지 지정 확인을 받기 위해서는 0x01, 안 받을 거면 0x00
⑥	PARAM1(파라미터1)	명령어와 더불어 지정해줄 파라미터가 있을 때 넣어줄 추가 파라미터의 상위 바이트
⑦	PARAM2(파라미터2)	명령어와 더불어 지정해줄 파라미터가 있을 때 넣어줄 추가 파라미터의 하위 바이트
⑧	CHECKSUM1(체크섬1)	전체 명령어 구문의 오류 확인을 위한 체크섬 상위 바이트
⑨	CHECKSUM2(체크섬2)	전체 명령어 구문의 오류 확인을 위한 체크섬 하위 바이트
⑩	$O(종료 바이트)	명령어 구문의 종료를 알리기 위한 바이트 값은 0xEF

CMD(명령어)에 들어갈 수 있는 값들은 아래와 같습니다.

베어팹

0x02	Previous	
0x03	Specify tracking(NUM)	0-2999
0x04	Increase volume	
0x05	Decrease volume	
0x06	Specify volume	0-30
0x07	Specify EQ(0/1/2/3/4/5)	Normal/Pop/Rock/Jazz/Classic/Base
0x08	Specify playback mode (0/1/2/3)	Repeat/folder repeat/single repeat/ random
0x09	Specify playback source(0/1/2/3/4)	U/TF/AUX/SLEEP/FLASH
0x0A	Enter into standby – low power loss	
0x0B	Normal working	
0x0C	Reset module	
0x0D	Playback	
0x0E	Pause	
0x0F	Specify folder to playback	1~10(need to set by user)
0x10	Volume adjust set	{DH＝1:Open volume adjust }{DL: set volume gain 0~31}
0x11	Repeat play	{1:start repeat play}{0:stop play}

언듯 보면 굉장히 복잡해 보이는데요. 명령어의 구조를 한번만 파악하고 나면 CMD에 해당하는 실제 명령어만 변경하면 되기 때문에 그다지 복잡하지 않습니다.

간단히 볼륨을 15으로 지정하고 2번 트랙을 재생하는 명령어 구문을 보내는 방법을 예시로 들어 보겠습니다.

기본상태에서 이 모듈의 볼륨은 30으로 지정되어 있습니다. 외부 전원을 사용하지 않고 USB전원으로 연결된 아두이노의 5V로 모듈을 구동시 가끔 전력이 부족해서 그런지 음악 재생중 드르륵 소리가 나면서 모듈이 정상 동작하지 않는 경우가 발생합니다. 이를 방지하기 위해서 먼저 모듈의 볼륨을 15으로 지정해 보겠습니다. 위 명령어 중 볼륨을 조절하는 명령어는 **"0x06" Specify Volume**에 해당합니다. 그리고 이 명령어는 0-30의 파라미터를 추가로 입력 가능한데요. 이 파라미터에 지정하고 싶은 볼륨값을 입력하게 됩니다.(우리는 15였습니다.) 따라서 위 명령어 구문에 이 값을 넣으면 전송할 명령어 구문의 값은 순서대로 아래의 값을 전송하면 됩니다.

정값, 15를 16진수로표현), 체크섬 상위바이트, 체크섬 하위바이트, 0xEF(종료바이트)

체크섬을 계산하는 방법은 아래 공식을 통해 계산할 수 있습니다.

체크섬(2바이트) = 0xFFFF-(버전바이트+명령어길이+CMD+피드백여부 + 파라미터1+파라미터2)+1

따라서 위 볼륨조절을 기준으로 체크섬을 계산해 보면

(0xFF+0x06+0x06+0x00+0x00+0x0F)= 0x011A이므로
체크섬 = 0xFFFF - 0x011A +1
체크섬 = 0xFEE6

이 중 체크섬의 상위바이트는 0xFE, 하위바이트는 0xE6 가 됩니다.

체크섬까지 계산해서 볼륨을 15로 지정하기 위해 보낼 전체 명령어 구문은 아래와 같습니다.

```
0x7E 0xFF 0x06 0x06 0x00 0x00 0x0F 0xFE 0xE6 0xEF
                                  ---------
                                  체크섬부분
```

2번 트랙을 재생하기 위한 방법도 위 볼륨조절과 동일합니다. 위 명령어 표 중 트랙 넘버를 선택하는 명령어는 **"0x03" Specify tracking(NUM)**에 해당합니다. 그리고 이 명령어는 파라미터로 0-2999의 값을 넣을 수 있는데요. 이 파라미터가 트랙번호가 됩니다. 따라서 2번 트랙을 지정할 땐 2번을 넣어주면 되겠죠? 이를 바탕으로 체크섬까지 계산해서 명령어 전체 구문을 계산하면 아래의 구문이 완성됩니다.

2 18

베어팹

play 2번 체크섬

이 명령어 구문을 순서대로 시리얼 통신을 통해 mp3모듈로 보내면 됩니다.

위 방법을 아두이노 코드를 통해 직접 확인 해 보도록 하겠습니다.

```
#include "SoftwareSerial.h"            // 소프트웨어 시리얼 라이브러리

SoftwareSerial mySerial(10, 11);       // 소프트웨어 시리얼 통신 핀 지정

# define Start_Byte 0x7E               // 시작 값
# define Version_Byte 0xFF             // 버전 값
# define Command_Length 0x06           // 길이 값
# define End_Byte 0xEF                 // 종료 값
# define Acknowledge 0x00              // 피드백 미 수신

void setup()
{
  mySerial.begin(9600);                // 소프트웨어 시리얼 통신 개시
  delay(2000);                         // 2초 대기
  specify_Volume(15);                  // 볼륨을 15로 지정
  specify_Track(3);                    // 3번 트랙 재생

}

void loop()
{

}

void specify_Volume(byte level)        // 볼륨 조절 함수
{
  execute_CMD(0x06, 0x00, level);      //볼륨조절 명령어 0x06과 볼륨레벨을 파
}

void specify_Track(int16_t track)      // 트랙 지정 함수
{
  execute_CMD(0x03, highByte(track), lowByte(track));      // 트랙재생 명령
```

베어팹

```
{

  int16_t checksum = -(Version_Byte + Command_Length + CMD + Acknowledge +
  // 보낼 명령어 구문을 배열에 저장
  byte Command_line[10] = { Start_Byte, Version_Byte, Command_Length, CMD

  for (byte k=0; k<10; k++)     // 명령어 구문을 시리얼 통신을 송신
  {
    mySerial.write(Command_line[k]);
  }
}
```

mp3모듈에 전원이 인가되어 있는 상태에서 아두이노에 위 코드를 업로드 하면 3번 트랙의 노래가 15의 볼륨으로 재생되는 것을 확인 할 수 있습니다.

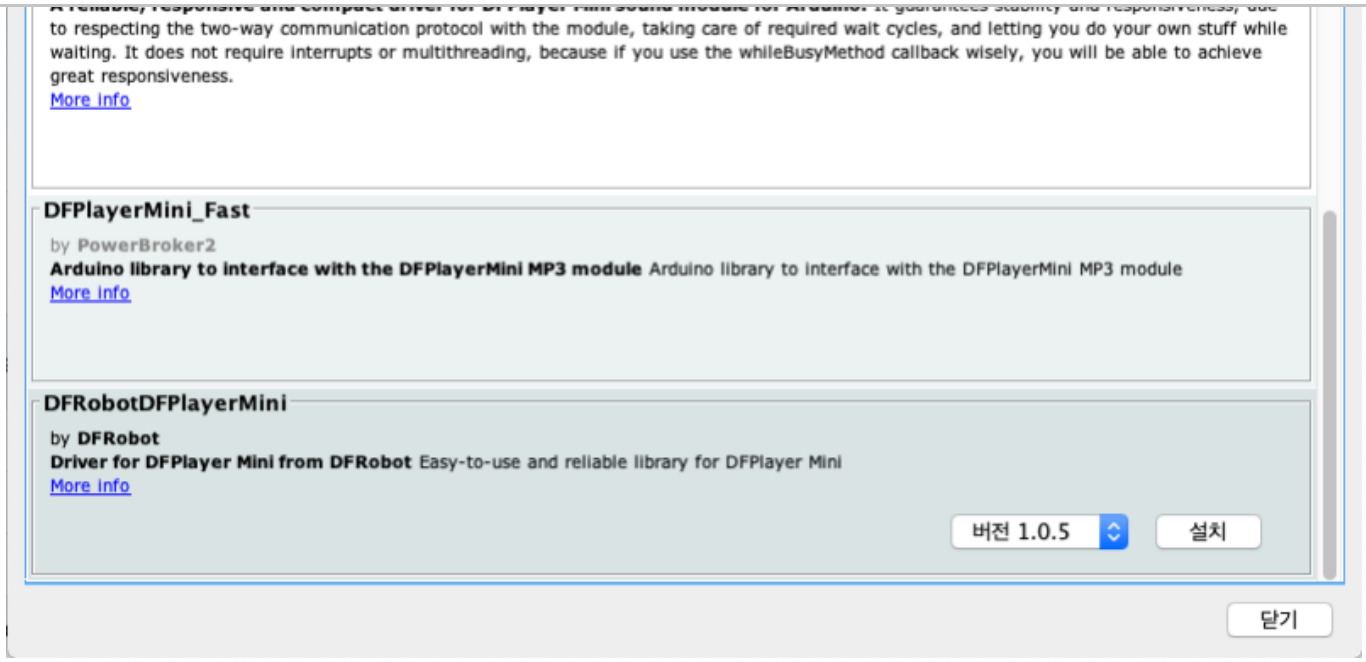
다른 명령어들도 위와 같이 새로 함수를 만들어서 사용할 수 있겠죠? ㅎㅎ

- 라이브러리 설치 -

사실 위 방법처럼 일일히 새로 함수를 작성할 필요없이 이미 개발되어 있는 여러 라이브러리를 사용하면 손쉽게 모듈을 제어할 수 있습니다.

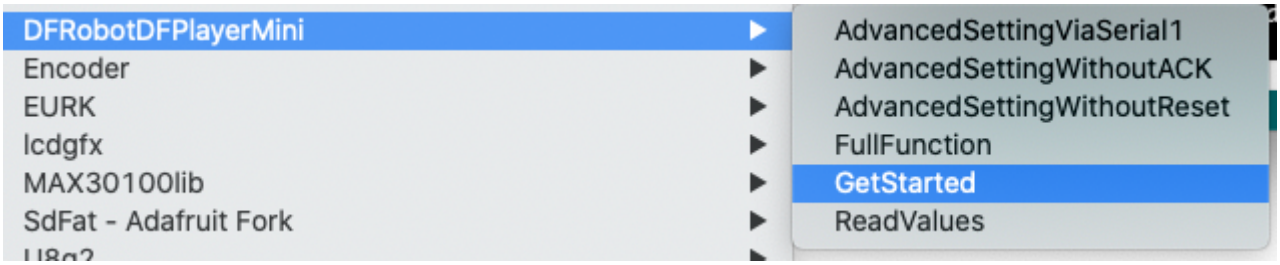
라이브러리 설치를 위해 아두이노의 IDE에서 **[스케치]-[라이브러리 포함하기]-[라이브러리 관리]**를 눌러 줍니다. 라이브러리 매니저에서 **dfplayer**을 검색하면 여러 라이브러리가 나오는데요. 이중 이번 포스팅에서는 DFROBOT사에서 제작한 **DFRobotDFPlayerMini** 라이브러리를 설치해 줍니다.

베어팹



- 라이브러리 예제 실행해 보기 -

라이브러리 설치를 위해 아두이노의 IDE의 **[파일]-[예제]-[DFRobotDFPlayerMini]-[GetStarted]** 를 열어 줍니다.



예제 파일에 보면 객체를 초기화 하는 begin 함수부분이 있는데 기본 상태에서는 라이브러리의 생성자에 feedback 메시지를 수신하는 것으로 설정이 되어 있습니다. 저는 feedback을 수신하는 상태에서는 자꾸 오류가 발생해서 수신하지 않도록 아래와 같이 false를 인자로 넣어주었더니 제대로 동작했습니다. 혹시 오류가 발생하면 아래와 같이 false를 넣어서 사용해 보시길 바랍니다.

베어팹

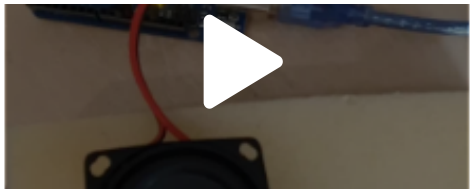
```
Serial.println();
Serial.println(F("DFRobot DFPlayer Mini Demo"));
Serial.println(F("Initializing DFPlayer ... (May take 3~5 seconds)"));

if (!myDFPlayer.begin(mySoftwareSerial, false)) {  //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){
```

이제 소스코드를 업로드 하면 아래 영상과 같이 3초에 한번씩 전체 파일을 순회하면서 음악이 재생되는 것을 확인 할 수 있습니다.(혹시 동작이 잘 안되면 소스코드 업로드 후 mp3 모듈의 전원을 분리했다가 다시 연결해 주면 잠시후 동작합니다.)

아두이노 mp3 모듈 예제

296 0



00:00 00:20 자동

아두이노 mp3 모듈 예제

- 라이브러리 함수 알아 보기 -

이제 라이브러리의 주요 함수에 대해 알아 보겠습니다. **[파일]-[예제]-**

2 18

베어팹

```
#include "Arduino.h"
#include "SoftwareSerial.h"
#include "DFRobotDFPlayerMini.h"

SoftwareSerial mySoftwareSerial(10, 11); // 소프트웨어 시리얼 용 핀 지정
DFRobotDFPlayerMini myDFPlayer;              // 객체 생성
void printDetail(uint8_t type, int value);

void setup()
{
  mySoftwareSerial.begin(9600);          // 소프트웨어 시리얼 통신 개시
  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), false) {  //객체 초기화
    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); //시리얼 통신용 타임아웃 시간 지정

  ----볼륨조절----
  myDFPlayer.volume(10);  //0-30사이의 값을 인수로 입력.
  myDFPlayer.volumeUp(); //볼륨을 1단계씩 키울 때 사용
  myDFPlayer.volumeDown(); //볼륨을 1단계씩 내릴 때 사용

  ----이퀄라이즈 모드 지정시 사용 ----
  myDFPlayer.EQ(DFPLAYER_EQ_NORMAL);    // 일반모드
  myDFPlayer.EQ(DFPLAYER_EQ_POP);      // 팝모드
  myDFPlayer.EQ(DFPLAYER_EQ_ROCK);     // 락모드
  myDFPlayer.EQ(DFPLAYER_EQ_JAZZ);     // 재즈모드
  myDFPlayer.EQ(DFPLAYER_EQ_CLASSIC);  //클래식 모드
  myDFPlayer.EQ(DFPLAYER_EQ_BASS);      // BASS모드
```

베어팹

```
myDFPlayer.outputDevice(DFPLAYER_DEVICE_AUX);
myDFPlayer.outputDevice(DFPLAYER_DEVICE_SLEEP);
myDFPlayer.outputDevice(DFPLAYER_DEVICE_FLASH);

----mp3 모듈 제어 모드----
myDFPlayer.sleep();     //슬립모드
myDFPlayer.reset();     //슬립모드로부터 복귀
myDFPlayer.enableDAC();  //음악파일에서 디코딩 상태로 설정
myDFPlayer.disableDAC();  //음악파일을 디코딩 하지 않음
myDFPlayer.outputSetting(true, 15); //이 모듈에서는 적용되지 않음

----Mp3 재생----
myDFPlayer.next();  //다음 곡 재생
delay(1000);
myDFPlayer.previous();  //이전 곡 재생
delay(1000);
myDFPlayer.play(1);  //1번째 노래 재생(SD카드에 파일이 저장된 순서대로 재생됨
delay(1000);
myDFPlayer.loop(1);  //첫번째 노래 반복재생
delay(1000);
myDFPlayer.pause();  //일시 정지
delay(1000);
myDFPlayer.start();  //다시 재생
delay(1000);
myDFPlayer.playFolder(15, 4);  //지정한 폴더의 트랙 재(15번 폴더의 4번 트랙
delay(1000);
myDFPlayer.enableLoopAll(); //전체 MP3파일 재생(SD카드에 파일이 저장된 순서
delay(1000);
myDFPlayer.disableLoopAll(); //전체 MP3파일 재생 중지(enableLoopAll 함수 (
delay(1000);
myDFPlayer.playMp3Folder(4); //mp3폴더에 저장된 트랙을 재생(SD:/MP3/0004.m
delay(1000);
myDFPlayer.advertise(3); // 광고모드를 실행해서 SD:/ADVERT/0003.mp3 파일을
delay(1000);
myDFPlayer.stopAdvertise(); //광고 모드 해제
delay(1000);
myDFPlayer.playLargeFolder(2, 999); //폴더내에 mp3파일이 여러개 여서 0001.
delay(1000);
myDFPlayer.loopFolder(5); //지정한 폴더엥 있는 모든 음악 순회재생(SD:/05번
delay(1000);
```

2 18

베어팹

```
  myDFPlayer.disableLoop(); //현재 반복 재생 트랙을 반복재생 모드에서 해제
  delay(1000);

  //----Read imformation----
  Serial.println(myDFPlayer.readState()); //mp3 모듈 상태 읽어오기
  Serial.println(myDFPlayer.readVolume()); //현재 볼륨값 읽어오기
  Serial.println(myDFPlayer.readEQ()); //EQ세팅 읽어오기
  Serial.println(myDFPlayer.readFileCounts()); //SD카드내의 전체 음악파일 개
  Serial.println(myDFPlayer.readCurrentFileNumber()); //현재 재생중인 폴더으
  Serial.println(myDFPlayer.readFileCountsInFolder(3)); //특정 폴더내의 파일
}

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

  if (millis() - timer > 3000) {
    timer = millis();
    myDFPlayer.next();  //다음 곡 재생
  }

  if (myDFPlayer.available()) {
    printDetail(myDFPlayer.readType(), myDFPlayer.read()); //Print the det
  }
}
```

주석만 봐도 대부분의 기능을 알수 있기 때문에 추가적으로 별도의 설명은 필요 없을 것 같네요 ㅎ

주석을 봐도 헷깔릴만한 기능만 추가적으로 몇가지 더 알아 보겠습니다.

▶ mp3 모듈 제어모드

```
myDFPlayer.sleep(); //슬립모드
myDFPlayer.reset(); //슬립모드로부터 복귀(동작안됨ㅠ)
myDFPlayer.enableDAC(); //음악파일에서 디코딩 상태로 설정
```

2 18

베어팹

입니다. 해당 모드로 진입한 뒤에는 음악재생 명령을 내려도 음악이 재생되지 않는데요. 다시 일반 모드로 복귀하기 위해서는 reset( ) 함수로 복귀하면 된다고 되어 있는데 해당 함수를 사용해 봤는데 일반모드로 돌아 오지 않더군요ㅠ 일반 모드로 복귀하기 위해 **myDFPlayer.outputDevice(DFPLAYER_DEVICE_SD);** 함수를 사용하니 다시 일반모드로 돌아오는 것을 발견했습니다.

enableDAC( ) 함수는 DAC모드로 동작시킨다는 의미인데요. 기본 상태에서는 DAC모드입니다. 음악트랙을 재생중인 상태에서 **disableDAC( );** 함수를 실행하면 내부적으로는 음악이 재생은 진행되고 있는데 디코딩이 안되기 때문에 소리가 안나는 상태가 됩니다. 다시 enableDAC( ) 함수를 실행하면 그 시점부터 다시 디코딩이 진행됩니다. 이 함수는 MUTE 모드를 만들때 사용하면 좋을 것 같네요.
ouptPutSetting( ) 함수는 사용해 봤는데 별 효과가 없는 것 같습니다.

▶ 광고(Advertise) 모드

위 함수들 중 **advertise( )**라는 함수와 **stopAdvertise( )**라는 함수가 있는데요. 이 함수는 광고모드를 실행하거나 해제할 때 사용되는 함수 입니다. 이 모듈은 광고모드를 지원하는데요. 광고모드는 특정 음악 재생중에 광고모드를 실행하면 먼저 재생중이던 음악을 일시중지 후 광고모드로 실행한 음악파일을 먼저 재생하고 다시 일시중지했던 음악파일을 재생하는 모드입니다. 이 모드를 실행하기 위해서는 SD카드에 ADVERT라는 별도의 폴더를 생성하고 아래 사진처럼 0001, 0002의 번호로 mp3파일을 저장해야 합니다. 음악파일로 음악을 재생하다가 **advertise(트랙 번호)** 함수를 실행하면 해당 음악을 일시 정지되고 ADVERT 폴더에서 지정한 트랙의 음악을 먼저 재생하게 됩니다. 그리고 ADVERT의 음악 재생이 끝나면 이어서 정지되었던 이전 음악을 재생하게 됩니다.

존재하지 않는 이미지입니다.

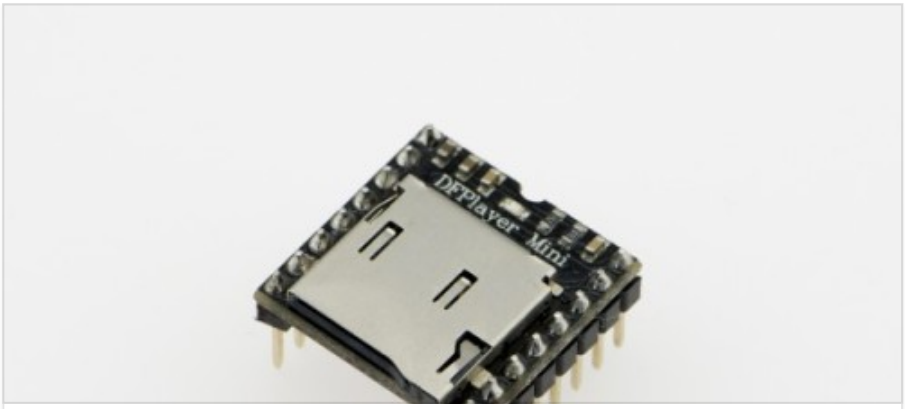
베어팹

존재하지 않는 이미지입니다.

이 밖의 추가기능에 대해서는 데이터 시트를 통해 확인하면 될 것 같네요. 이제 재밌는 음악관련 프로젝트에 이 모듈을 잘 활용할 수 있겠죠?^^

참고자료 및 더 읽어보기

- 모듈설명서: https://wiki.dfrobot.com/DFPlayer_Mini_SKU_DFR0299

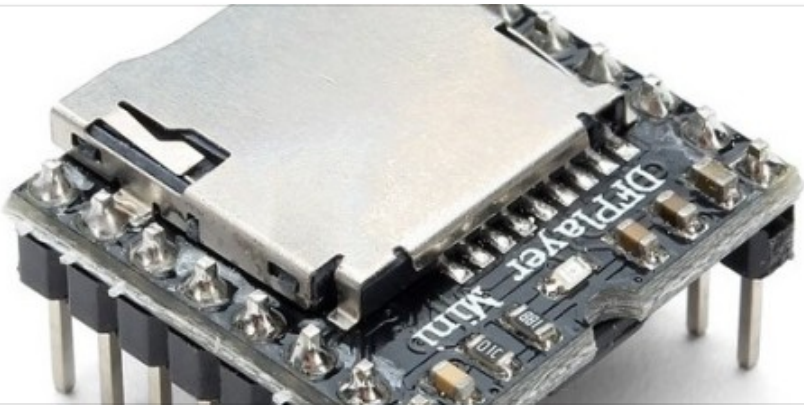


DFPlayer_Mini_SKU_DFR0299-DFRobot
Introduction Specification Application Pin Map Work Mode Connection D…
wiki.dfrobot.com

- 데이터 시트

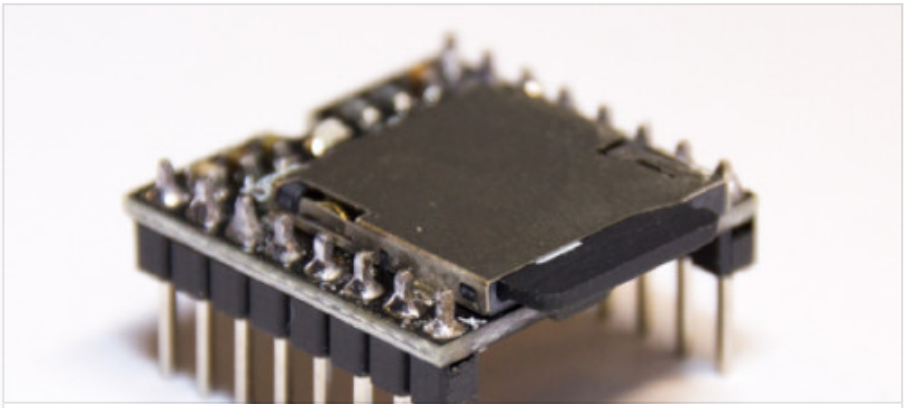
2 18

- 코드 참고: https://www.electronics-lab.com/project/mp3-player-using-arduino-dfplayer-mini/



MP3 player using Arduino and DFPlayer mini - Electronics-Lab.com
Hi guys, welcome to this tutorial. Today, we will build an mp3 player using…
www.electronics-lab.com

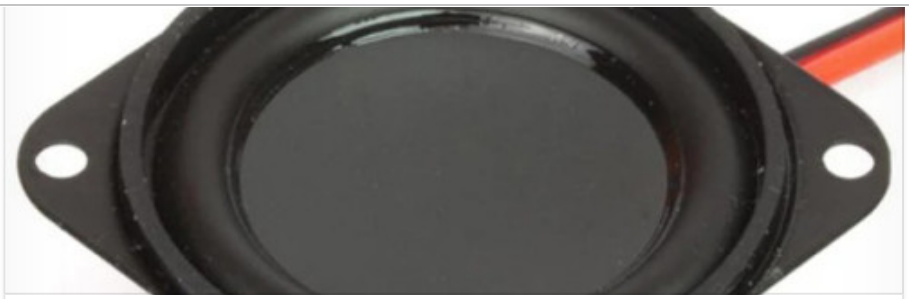
- MP3파일 넣는 방법 참고: https://circuitjournal.com/how-to-use-the-dfplayer-mini-mp3-module-with-an-arduino



Wiring DFPlayer Mini (MP3 Module) to Arduino. Stereo/Mono Diagr…
Wiring DFPlayer Mini (MP3 Module) to Arduino. Stereo/Mono Diagrams. …
circuitjournal.com

- 사용 스피커

베어팹



3W 미니스피커<4옴> : 베어팹몰
[베어팹몰] 메이커를 위한 전자쇼핑몰
smartstore.naver.com

#아두이노 #MP3 #MP3모듈 #DFPlayer #DFPlayerMIni #dfplayermini #디에프플레이어 #디에프플레이어미니 #엠피쓰리 #엠피쓰리모듈 #아두이노음악 #아두이노뮤직 #아두이노mp3 #아두이노mp3제어 #Mp3 #MP3미니 #아두이노음악제어

2 18

황제곰
IT·컴퓨터

곰손도 만든다! 베어팹몰: https://smartstore.naver.com/bearfabmall

이웃추가

이 블로그 아두이노 카테고리 글

아두이노 S148(YS-IRTM) IR송수신 모듈 사용하기
2022. 1. 19.
1 0

2 18

베어팹

아두이노 BME280 온습도 및 기압센서 사용하기

2020. 11. 24.

3 1

아두이노로 맥박, 산소포화도 센서(MAX30100) 사용하기

2020. 11. 13.

1 12

아두이노로 TFT 디스플레이 모듈 제어하기

2020. 11. 10.

0 0

이 블로그의 #아두이노 다른 글

아두이노 S148(YS-IRTM) IR송수신 모듈 사용하기

2022. 1. 19.

1 0

초소형 아두이노 Seeeduino Xiao 사용가이드

2021. 1. 8.

4 4

아두이노 BME280 온습도 및 기압센서 사용하기

2020. 11. 24.

3 1

아두이노로 MQ2 가스센서 사용하기

2 18

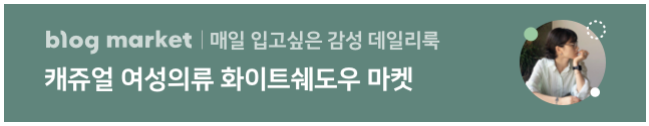
베어팹

아두이노 블루투스 모듈(HC-06)로 무선 통신 하기

2020. 10. 29.

1 0

맨 위로



PC버전으로 보기

2 18