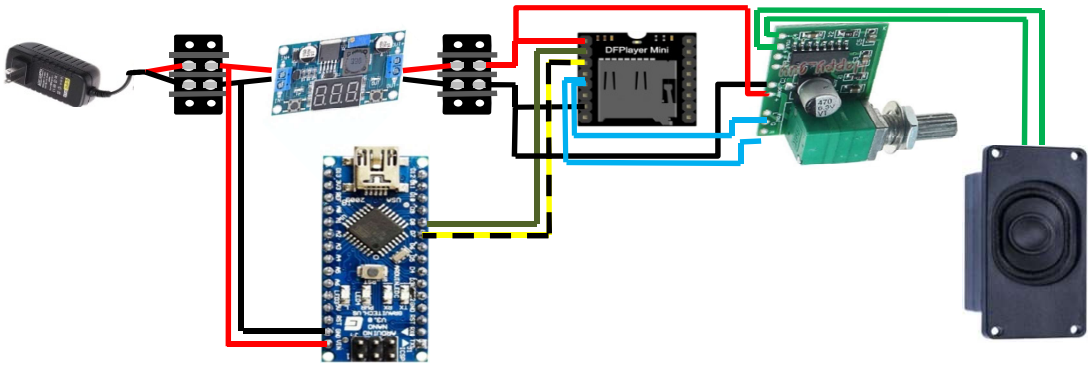
Part 9 – Adding Audio to an Arduino

April, 2020
David Ackmann – Gateway Division NMRA

With the proper hardware, an Arduino can play MP3 files stored on a micro SD card, such as are commonly found in cameras and cell phones. This tutorial explains how this can be accomplished.

In addition to an Arduino (I usually use an Arduino Nano), external power supply and a micro SD card, you will need a MP3 player chip (I use a DFPlayer Mini, AMAZON ASIN: B07JGWMPTF), a speaker (AMAZON ASIN: B0738NLFTG), a voltage regulator (AMAZON ASIN: B07WQJ2GD6), barrier strips, spade connectors and jumper wires. While you can control the volume level through software, I prefer to use a small amplifier with a knob to adjust the volume (AMAZON ASIN: B01MYTZGYM). For all my Nano projects I also a Nano Expansion Board (AMAZON ASIN: B07MGVC18K). The diagram below gives the basic block diagram.



The 12VDC power supply is connected to a barrier strip (red and black wires), and from there to a Buck Converter (voltage regulator) and to the GND and VIN pins of the Arduino (make sure you get the polarities correct). Run wires from the output of the Buck Converter to the second barrier strip. Set the Buck Converter to 4.2V before hooking up power from the second barrier strip to the MP3 player and the amplifier. Run wires from pins 7 and 8 of the Arduino (green and yellow/black) to pins 2 and 3 of the MP3 player (it doesn't matter which pins on the Arduino go to which pins on the MP3 player). Run wires from pins 4 and 5 of the MP3 Player (blue) to pins 1 and 2 of the amplifier. Note that the amplifier does not have pins in positions 4 and 7. Run wires from the amplifier pin positions 10 and 11 (green) to the speaker.

It may seem a bit odd that we need a Buck Converter (voltage regulator) in addition to the external power supply. But this is necessary because the MP3 Player circuit likes to runs on about 4.2 volts, whereas the Nano will run on anything from about 7 volts to 12 volts DC. There are less expensive voltage regulators on the market, but I like this one because it has a display making it easier to dial in the proper voltage, and it runs cooler than a simple 3 pin voltage regulator. Deviate from the recommendation at your own risk.

I like to use barrier strips between the power supply and the Buck Converter/Nano, and also between the Buck Converter and the MP3 Player/Amplifier. It just makes connections easier. You don't have to use spade connectors at the end of your jumpers when attaching them to the barrier strips (you could just tin the wire ends and wrap the ends around the screws), but spade connectors make for more reliable connections.

Before we start connecting components, let's discuss where we find model railroad sounds, and how we edit the sound files we find. As you might expect, we start by doing a web search on "model railroad sounds". Visit http:// http://soundbible.com/tags-train.html for starters. Check out their "Royalty Free Sounds" and their "Sound Effects" tab, but if you need something specific, use the site's "Search" feature. Follow the instructions to download a MP3 file to your local PC. Also, do a web search on "sound effects"; not all are free, but if you can find just what you want, it might be worth a few dollars. When you find a sound file or files you like, download them to a folder on your PC, then move one or more to your SD card (you don't need a very large card to store a lot of sounds). But be advised that playing an SD card with a lot of sound files can be tricky, because the Arduino software references sound files by their position on the SD card, not by their file name.

If you are lucky, you may find exactly what you want. For my diorama of a lumberjack chopping down a tree I needed a chopping sound and a falling tree sound, and I found them in separate files. I downloaded free sound editing software called "Audacity" and used it to trim the files to meet my exact needs, and to adjust their volume. Audacity has many other sound editing features as well, and it now is my preferred tool for sound editing. Visit http://www.audacityteam.org and download a copy; there are some good Audacity tutorials available on YouTube as well.

If you have read "Part 7 – Why an Arduino Nano instead of An Uno", you know that I prefer a Nano because the available Expansion Board (AMAZON ASIN: B07MGVC18K) makes it easier to connect jumper wires to the Arduino. I also discovered that I can take the same expansion board, saw it down the middle on a band saw, then saw it again perpendicular to the first cut, cutting it at the ninth connection point, then placing the chip into the two pieces to make an expansion board for the MP3 player. **Need a photo here** I hot glue the expansion board pieces onto a piece of styrene to make it easier to mount on a fascia board. One piece of 0.040 thick styrene should do the trick. This makes connecting the player and the Arduino easier and more

secure (and because for the MP3 player we don't need to use pins 9-15, we only need connection points for pins 1-8 and sometimes pin16).

But be careful. Power tools like band saws can be intimidating and dangerous if you are not experienced, especially when you saw through the metal pieces that make up pin 9, or any other metal part of the original expansion board. Always follow basic safety instructions and wear eye protection. If you feel uncomfortable making these cuts, then don't attempt them and find someone with a bit more experience to help; there is no shame in staying safe.

I use the other half of the remaining cutoff to build an expansion board for the amplifier. Just 11 pins were needed here, so I made the crosscut at the 12th pin. Again, "Safety First".

Now on to the wiring diagrams, and we will be showing two of them. In the first one the MP3 player drives the speakers directly, which means there is no potentiometer to control the volume of the sound clip, it is done by setting a volume parameter in the software; in the second one we place a small amplifier between the MP3 player and the speakers and can change the volume without changing the software. For my money, the amplifier is worth the extra cost. If you want to try it without the amplifier first, well it is a simple change later on to add an amplifier.

I have placed an Arduino sketch which plays the sound of a lumberjack felling a tree on GitHib at http://www.github.com/daackm, so visit the site and open the "Arduino-Animations-Tutorials", and copy the "Part 9" code. Open a new sketch in the IDE, delete all the original code, paste my code into the IDE and save the sketch to your PC.

I do need to mention some details about placing your files on the SD card. To play a sound byte on my MP3 player, you need to reference the file by its numeric place on the card, not by its name; If you want to play the 3rd file, the line in the sketch would be "myDFPlayer.play(3);". Keeping track of which sound is in which file is tricky, and I suggest that you name your files starting with a number, such as "001Chopping Sound.mp3", or "002Falling Tree.mp3", and then copy the files to the SD card in order, one by one. And don't place more files on the card than you need for your current animation.

Also, take a look at the line "boolean play_state=digitalRead(8);". See that in the wiring diagrams we have a jumper running from pin 16 of the MP3 player to pin D8 of the Arduino. This "digitalRead" on Arduino pin 8 allows us to monitor whether the MP3 player is still playing a song, or has completed the task. In our demo we take a 10 second break after playing the sound file before we repeat the operation. If you look ahead at my Carousel project you will see code where I have multiple sound files that I

play at random, so it is important to know when one song ends so I can start another tune at the appropriate time.