Play an MP3 file with C

By Paeton Dhesi, Jared Murphy, Pavanpal Bhasin, Patrick Burns
Created April 10th, 2023

Guide has been tested on:

BeagleBone (target) - Debian 11.5
PC OS (host) - Debian 11.6
Zen Cape Green

Introduction

This guide will demonstrate the basic usage of the MPG123 C library to play an MP3 file programmatically with C.

Required: Follow the class **Zen Cape Audio Guide** before completing this guide. We will assume that your target can already access the audio hardware on the Zen cape. We will also assume that you can build C programs with the `asound` library.

1. Installation

- On the host, install the `libmpg123-dev` development library. This is the library used to programmatically work with MP3 files and it will be used for the header file.
```
(host) $ sudo apt-get update
(host) $ sudo apt-get install libmpg123-dev
```

- Then we need to make a directory in the shared NFS space with the target where we will later store the libmpg123.so.* file for building your C program. You might recognize the next few steps to be very similar to installing the asound library from the class audio guide.
```
(host) $ mkdir ~/cmpt433/public/mpg123_lib_BBB
(host) $ chmod a+rwx ~/cmpt433/public/mpg123_lib_BBB
```

- On the target, install `libmpg123-0` library:
```
(bbg) $ sudo apt-get update
(bbg) $ sudo apt-get install libmpg123-0
```

- Check for the libmpg123.so files, and copy the .so file with **the most version numbers** to NFS space in the directory we created earlier. Your version numbers might be different.
```
(bbg) $ cd /usr/lib/arm-linux-gnueabihf/
(bbg) $ ls libmpg123*
libmpg123.so.0  libmpg123.so.0.45.3
(bbg) $ cp libmpg123.so.0.45.3 /mnt/remote/mpg123_lib_BBB/libmpg123.so
```

- We should be ready to move on to building C programs with the mpg123 library.

2. Writing a C program to play an MP3 File with the MPG123 library

- Below is a simple C program to play a single MP3 file. Read the comments to understand what is happening at each step. Note that we remove some error checking to condense the lines of code. You should still do this at the appropriate points. In particular you can use the MPG123_OK error code when working with the library.

```
#include <alloca.h>
#include <alsa/asoundlib.h>
#include <mpg123.h>

int main()
{
   char* path = "/mnt/usb/music/01 You Only Live Once.mp3";

   snd_pcm_t* pcmHandle;
   snd_pcm_hw_params_t* params;

   int numChannels, encoding;
   unsigned int bitRate;
   unsigned char* mp3Buffer;       // Stores decoded MP3 audio data
   size_t mp3BufferSize;

   // init the mpg123 library
   mpg123_init();

   // create a new mp3 handle
   mpg123_handle* mp3Handle = mpg123_new(NULL, NULL);
   // Open the MP3 file for decoding
   mpg123_open(mp3Handle, path);

   // get the bit rate, number of channels, and encoding for the file to
be played
   mpg123_getformat(mp3Handle, (long*)&bitRate, &numChannels, &encoding);

   // Allocate a buffer for the mp3 data
   mp3BufferSize = mpg123_outblock(mp3Handle);
   mp3Buffer = (unsigned char*)malloc(mp3BufferSize * sizeof(unsigned
char));

   // Open the ALSA audio device for playback
   snd_pcm_open(&pcmHandle, "default", SND_PCM_STREAM_PLAYBACK, 0);

   // configure parameters for PCM output for the hardware
   snd_pcm_hw_params_alloca(&params);
   snd_pcm_hw_params_any(pcmHandle, params);
   snd_pcm_hw_params_set_access(pcmHandle, params,
SND_PCM_ACCESS_RW_INTERLEAVED);
   snd_pcm_hw_params_set_format(pcmHandle, params, SND_PCM_FORMAT_S16_LE);
   snd_pcm_hw_params_set_channels(pcmHandle, params, numChannels);
   snd_pcm_hw_params_set_rate_near(pcmHandle, params, &bitRate, 0);
   snd_pcm_hw_params(pcmHandle, params);

   // decode the mp3 data, store in the mp3Buffer, and play it back
   size_t bytesRead = 0;
   while (mpg123_read(mp3Handle, mp3Buffer, mp3BufferSize, &bytesRead) ==
MPG123_OK) {
       snd_pcm_prepare(pcmHandle);
       snd_pcm_writei(pcmHandle, mp3Buffer, bytesRead / numChannels /
sizeof(short));
   }

   // free the mp3 buffer and close/delete handles
   free(mp3Buffer);
   snd_pcm_close(pcmHandle);
   mpg123_close(mp3Handle);
   mpg123_delete(mp3Handle);
}
```

3. Building and Compilation

- Ensure that your makefile contains a linker flag that points to the location of the mpg123 library (in NFS space). The linker flag might look like this:
  `-L$(HOME)/cmpt433/public/mpg123_lib_BBB`

- Include the `-lmpg123` linker directive in makefile build commands.

- Similarly to the above two steps, you'll need to have a linker flag pointing to the asound library location, and the `-lasound` linker directive.

4. Extra Information

- Official mpg123 docs: https://www.mpg123.de/api/group__mpg123__init.shtml

- You can also extract the MP3 metadata (song title, album, artist etc.) using the mpg123 library. Here is an example using the id3v2 standard:

```
mpg123_id3v1* id3v1;
mpg123_id3v2* id3v2;
 if (mpg123_id3(mp3Handle, &id3v1, &id3v2) == MPG123_OK) {
     if (id3v2 && id3v2->title && id3v2->artist && id3v2->album) {
         // do what you want with want with the metadata
     }
```

Troubleshooting

- If building is failing, ensure that:
  - You have installed `libmpg123-dev` on the host and `libmpg123-0` on the target.
  - You have copied the libmpg123.so.0.<version> file to NFS space and are properly linking its path in your makefile. Make sure the directory where you copied it has the proper permissions (a + rwx). Also make sure you did the same for the asound library as required by assignment 3.
  - The linker directives `-lmpg123` and `-lasound` are in the build command

- If you get a generic MPG123 error ensure that you aren't forgetting to initialize the library first in the program with the call to `mpg123_init()`