Recording with a USB Mic Guide

CMPT 433: Embedded Systems

This guide has been tested on:

BeagleY-AI (Target): Debian 12.10 (Bookworm)
PC OS (host): Debian 11.11 (Bullseye)
Tools: alsa-utils v1.2.8-1

This guide covers the following steps:

1. Configuring a USB mic to the Beagle board
2. Programmatically saving the created file to a folder
3. Troubleshooting tips

1. Configuring a USB mic to the Beagle board

In order to configure recording, the alsa-utils library will first have to be installed.

(byai)$ ***sudo apt install alsa-utils***

List all detected devices before plugging in the USB mic with the following command. You can determine which device your mic corresponds to by running the following command before and after plugging in the USB mic. Look for a device in the format “hw:_,_”. For example, the mic might appear as “hw:CARD=Device,DEV=0”.

(byai)$ ***arecord -L***

Once we have identified the device, we need to check its configuration. We need to identify the **Format** and **Rate** of our mic so that we can explicitly pass it to the record command, otherwise ALSA might choose to record with a sample rate not supported by our mic.

(byai)$ ***arecord -D <USB_MIC> --dump-hw-params***

```
calbertan@cat8sfu:~$ arecord -D hw:CARD=Device,DEV=0 --dump-hw-params
Warning: Some sources (like microphones) may produce inaudible results
         with 8-bit sampling. Use '-f' argument to increase resolution
         e.g. '-f S16_LE'.
Recording WAVE 'stdin' : Unsigned 8 bit, Rate 8000 Hz, Mono
HW Params of device "hw:CARD=Device,DEV=0":
--------------------
ACCESS:  MMAP_INTERLEAVED RW_INTERLEAVED
FORMAT:  S16_LE
SUBFORMAT:  STD
SAMPLE_BITS: 16
FRAME_BITS: 16
CHANNELS: 1
RATE: [44100 48000]
PERIOD_TIME: [1000 1000000]
PERIOD_SIZE: [45 48000]
PERIOD_BYTES: [90 96000]
PERIODS: [2 1024]
BUFFER_TIME: [1875 2000000]
BUFFER_SIZE: [90 96000]
BUFFER_BYTES: [180 192000]
TICK_TIME: ALL
--------------------
arecord: set_params:1352: Sample format non available
Available formats:
- S16_LE
```

The following command will create a recording test.wav for 10 seconds:

(byai)$ ***arecord -D <USB_MIC> -f <FORMAT> -r <RATE> --duration=10 test.wav***

2. Programmatically saving the created file to a folder

The next step is to run the above command in our code. Since the command will generate a file, we need to make sure that the folder that ALSA is generating the .wav file has the proper permissions.

The important thing to note is that we are running the “arecord” command on the Beagle, but we are writing to a mounted folder from the Host running Linux which might lead to permission issues.

To solve this, create a folder with the necessary permissions that will be copied over during the build process. That folder will contain all the recording outputs.

1. Create a “Audio-Ouput” folder in the project’s home directory

2. Add the following lines to the app/CMakeLists.txt to copy over the specified folder to proper location with the necessary permission. In this case, it will copy over the Audio Output in the project to the /public/myApps folder where the program is cross-compiled

```
add_custom_command(TARGET mic_record_sample POST_BUILD
  COMMAND "${CMAKE_COMMAND}" -E copy_directory
     "${CMAKE_SOURCE_DIR}/Audio-Output"
     "~/cmpt433/public/myApps/Audio-Output"
  COMMAND chmod u+w "~/cmpt433/public/myApps/Audio-Output"
  COMMAND chmod 777 "~/cmpt433/public/myApps/Audio-Output"
  COMMENT "Copying Audio-Files folder to public NFS directory")
```

Then, we can simply call a fork and run the recording command to write the output to the folder. We have provided a code snippet to show what that might look like.

```
void startRecording(){
  pid_t pid = fork();
  if(pid < 0){
    perror("Error creating fork");
  }

  if(pid > 0){                              // Parent process
    wait(NULL);                             // Waits for child process to finish
  }
  else{                                     // Child process
    char *filename = "Audio-Output";
    char *args[] = {
      "arecord",
      "-D", MIC_DEVICE,
      "-f", MIC_FORMAT,
      "-r", MIC_RATE,
      "--duration", "10”,
      filename,
      NULL                                 //needs to be null terminated
    };

    execvp("arecord", args);
    perror("Error when executing arecord");
    free(filename);
    exit(EXIT_FAILURE);
  }
}
```

3. Troubleshooting Tips

1. If your mic has multiple channels, you should also add a “-c” flag. This is what the command looked like when we used a different mic
   (byai)$ ***arecord -D hw:CARD=SoloCast,DEV=0 -f S16_LE -r 44100 -c 2 --duration=10 test.wav***

2. If you encounter a permission denied, try running your program on the Beagle instead of the mounted folders then make sure the folder path for the .wav file exists.

We have created a mic_record_sample project to test recording with a mic in code. The project is based on Dr. Brian’s cmake_starter.zip

Reference

https://opencoursehub.cs.sfu.ca/bfraser/solutions/433/04-BuildingSoftware/