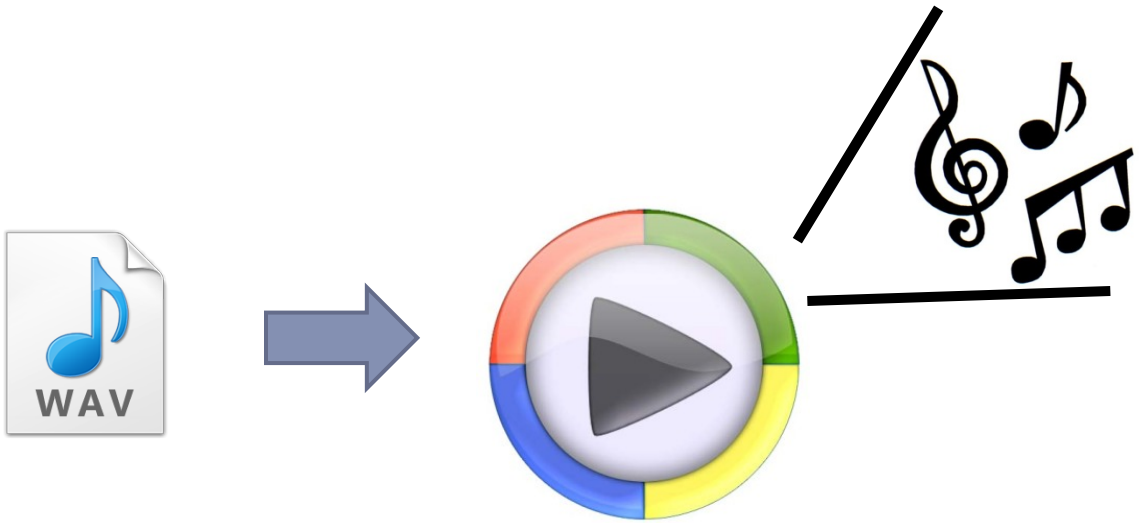
CSCI 3280 Tutorial 4
Mini Project Specification
Karaoke System（C++）

Main topic:

- 1, Low Level API
- 2, Wave File Format
- 3, Audio Play Procedure

Low Level API

- Kalaoke System
- Music Player
- Please do NOT leave your questions to the final day



Low Level API

- Windows APIs

High-level: calling one function ?

- *PlaySound("c:\\abc.wav",0,SND_ASYNC|SND_FILENAME)*

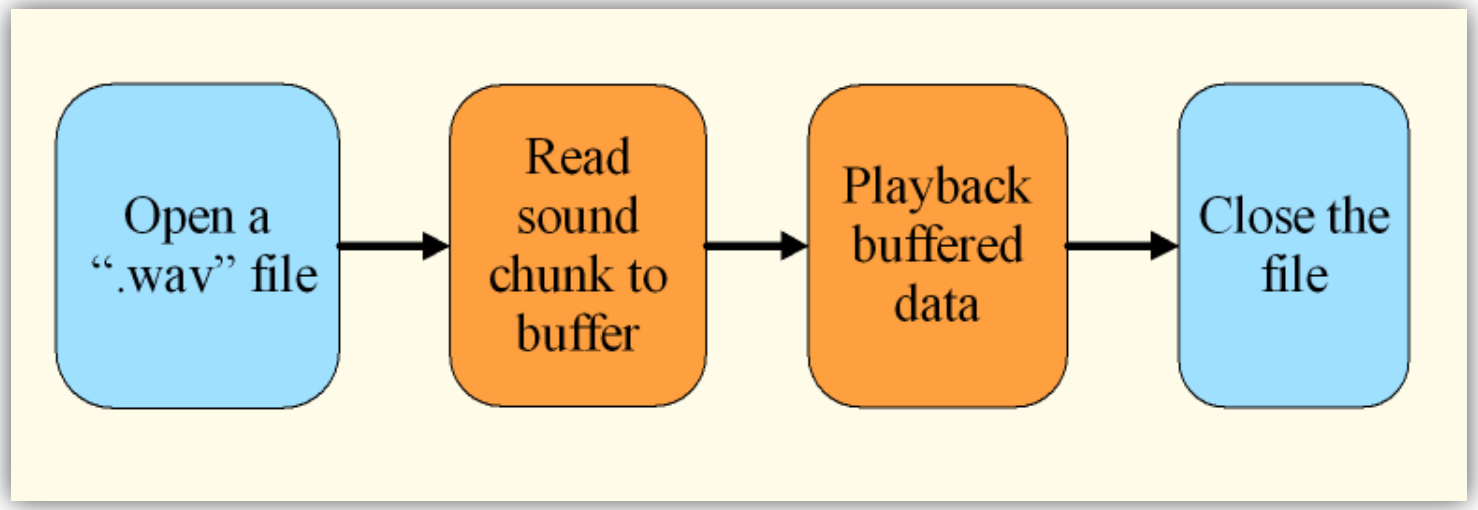
- **Low-level: manipulating the sound data manually**

- DirectX APIs

- Third-party codes

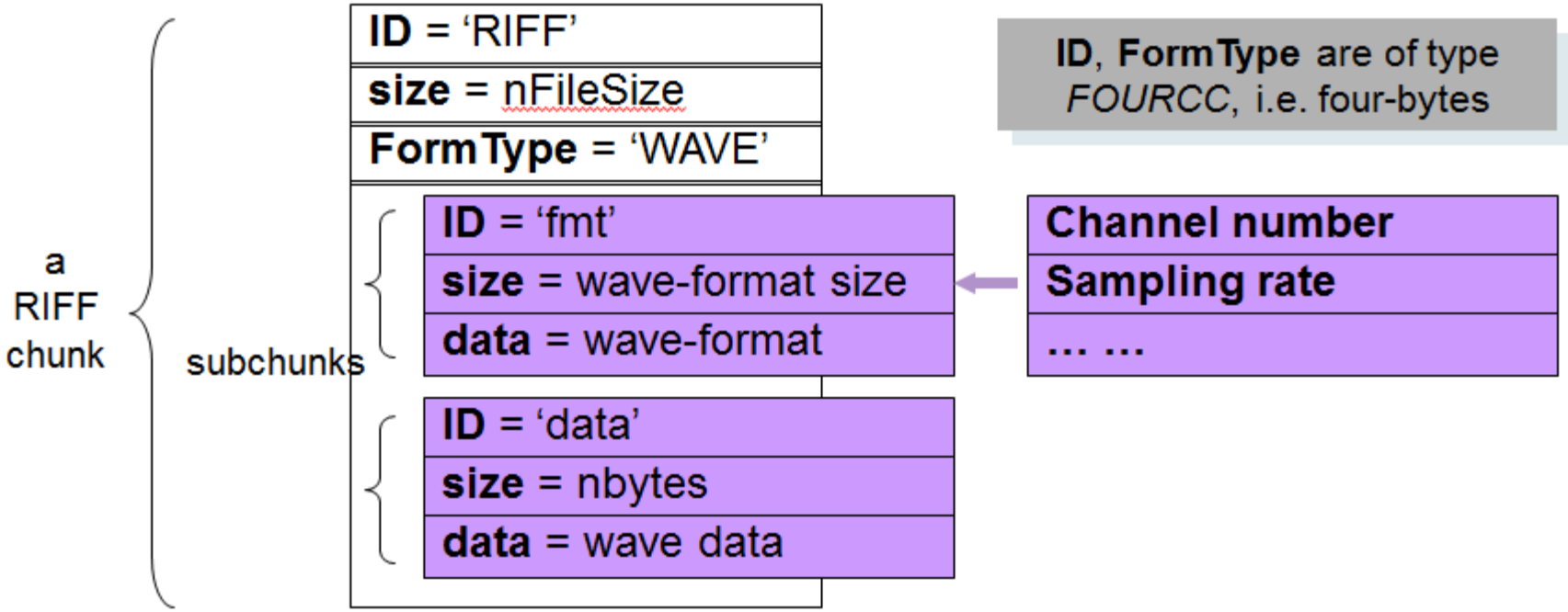
Low Level API

- Full control by your code!

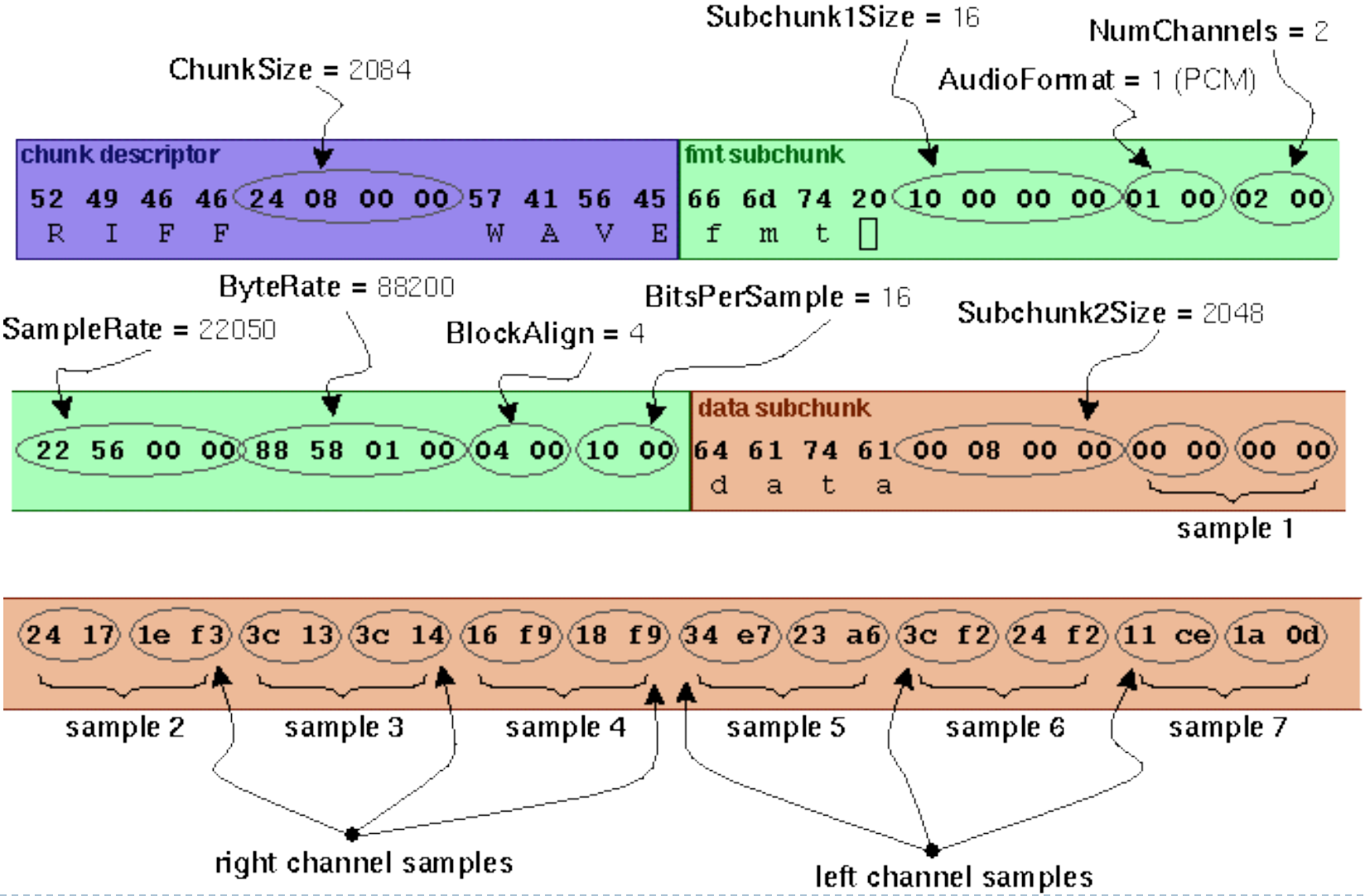


WAV File format

- RIFF (Resource Interchange File Format)



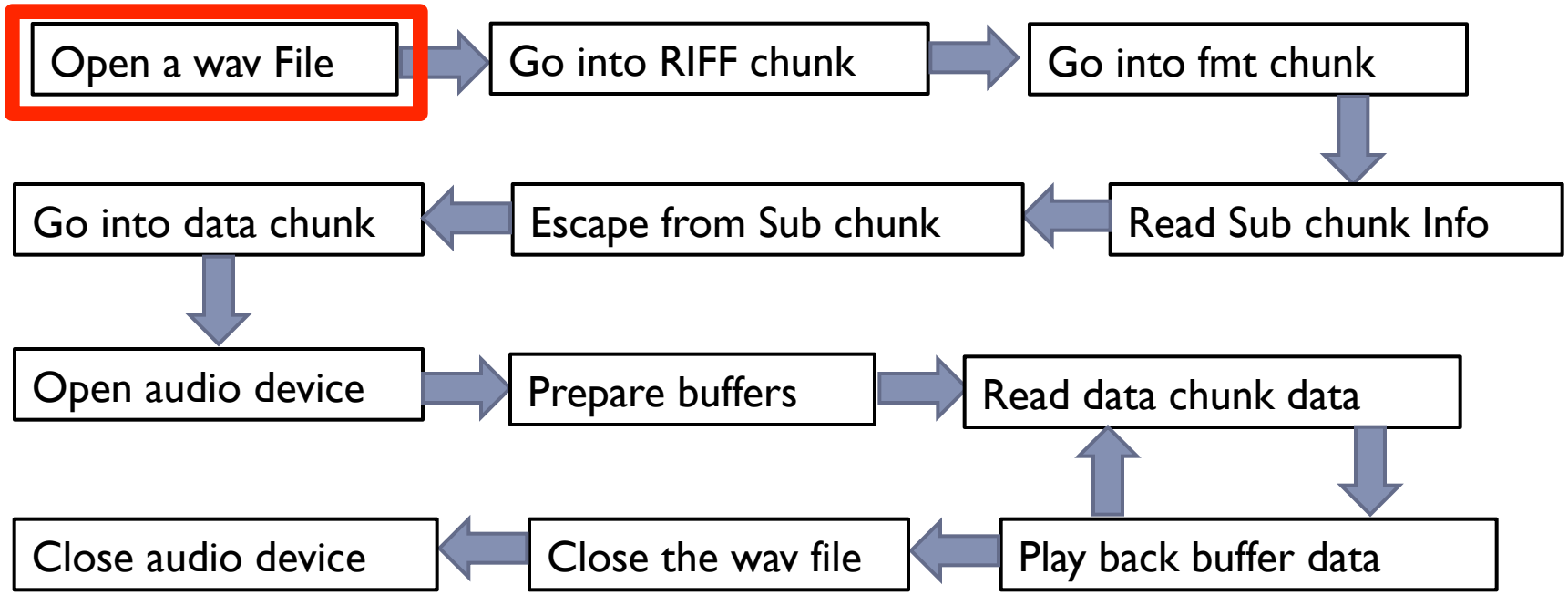
WAV File format



Windows API

- windows.h
- mmsystem.h
- Function:
  - mmioXXXX()
  - waveOutXXXX( )
- You can read the sample program.
- Hints:
  - Google is your friend!

Audio Play Procedure

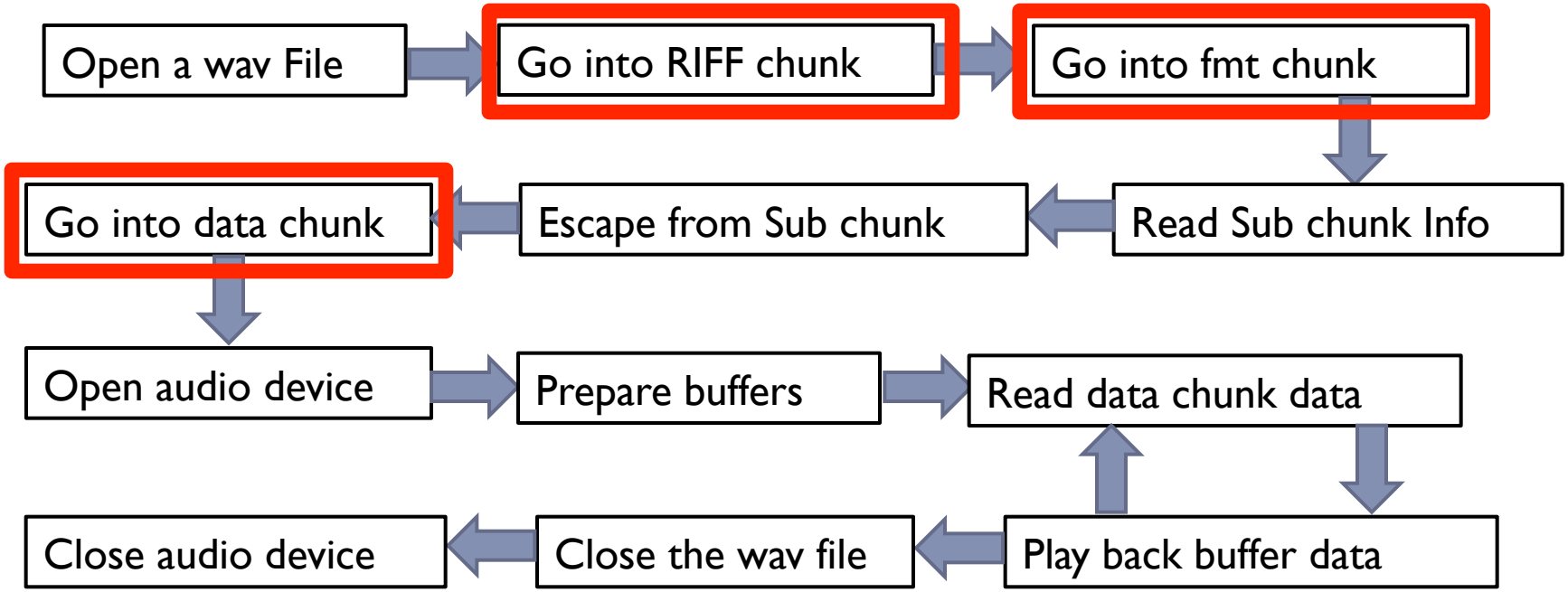


HMMIO mmioOpen (LPSTR *filename*, LPMMIOINFO *info*, DWORD *flags*);

Opens a file for unbuffered or buffered I/O

```
if ((hmmioIn = mmioOpen(filename, NULL, MMIO_READ)) == NULL) {
        fprintf(stderr, "Error: mmioOpen error on %s\n", filename);
        exit(-1);
}
```

Audio Play Procedure

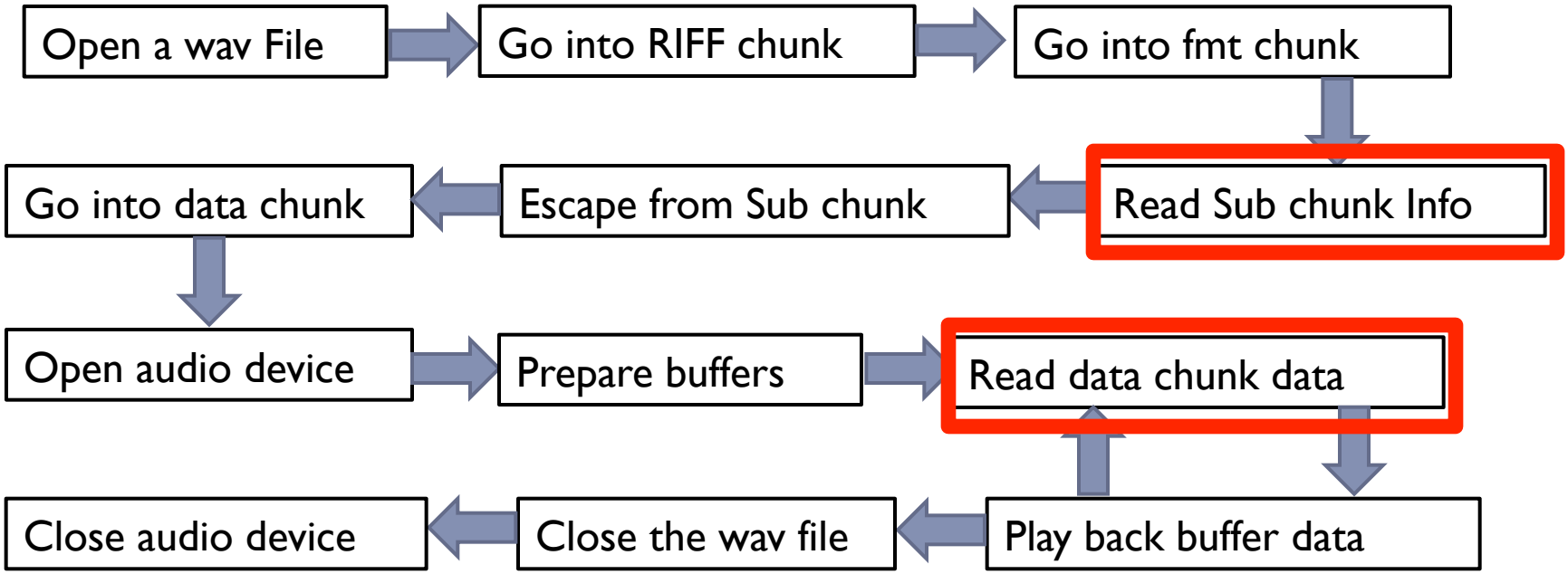


mmioDescend (HMMIO *h*, LPMMCKINO *lpck*, LPMMCKINFO *lpckParent*, UNIT *flags*);

Descends into a chunk of a RIFF file

```
pckInRIFF.fccType = mmioFOURCC('W', 'A', 'V', 'E');
if (mmioDescend(hmmioIn, &pckInRIFF, NULL, MMIO_FINDRIFF)) {
      fprintf(stderr, "Error: could not find WAVE chunk or ");
      fprintf(stderr, "the file is not in RIFF\n");
      mmioClose(hmmioIn, 0);
      exit(-1);
}
```

Audio Play Procedure

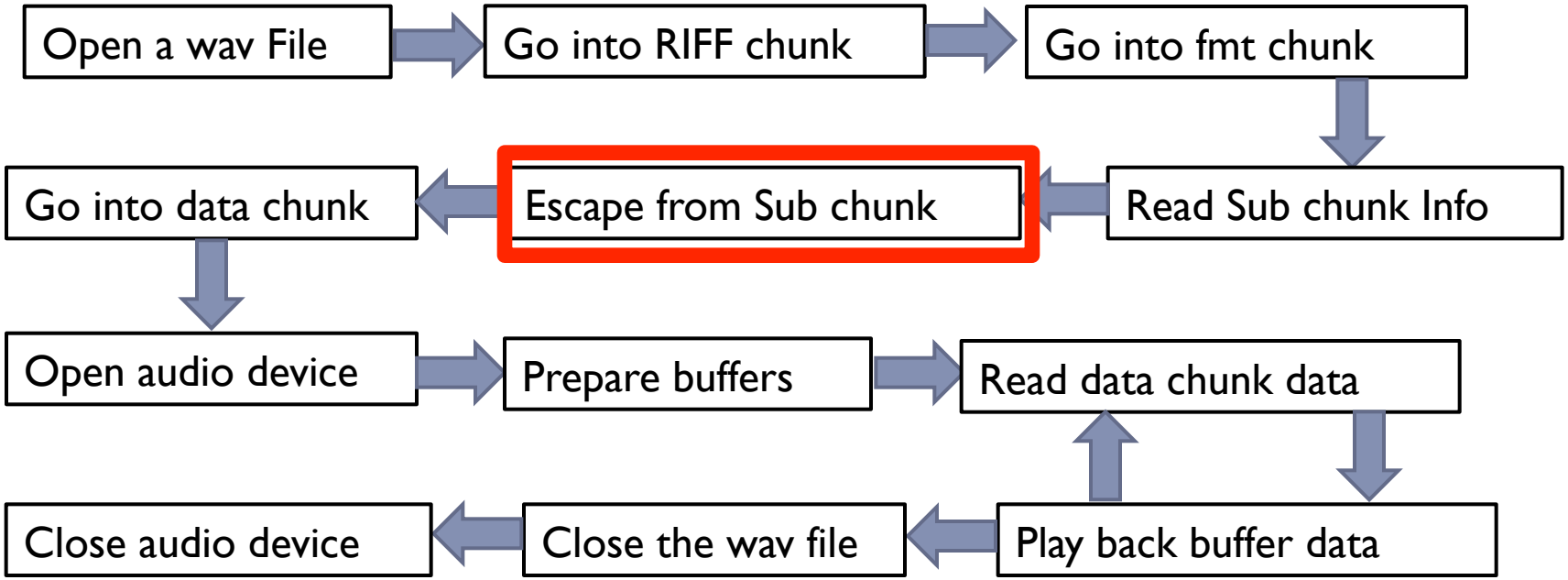


LONG mmioRead (HMMIO *h*, HPSTR *pch*, LONG *cch*);

reads a specified number of bytes from a file

```
if (mmioRead(hmmioIn, (HPSTR)&pcmWaveFormat, waveformatSize) != waveformatSize) {
        fprintf(stderr, "Error in reading the waveformat\n");
        mmioClose(hmmioIn, 0);
        exit(-1);
}
```

Audio Play Procedure

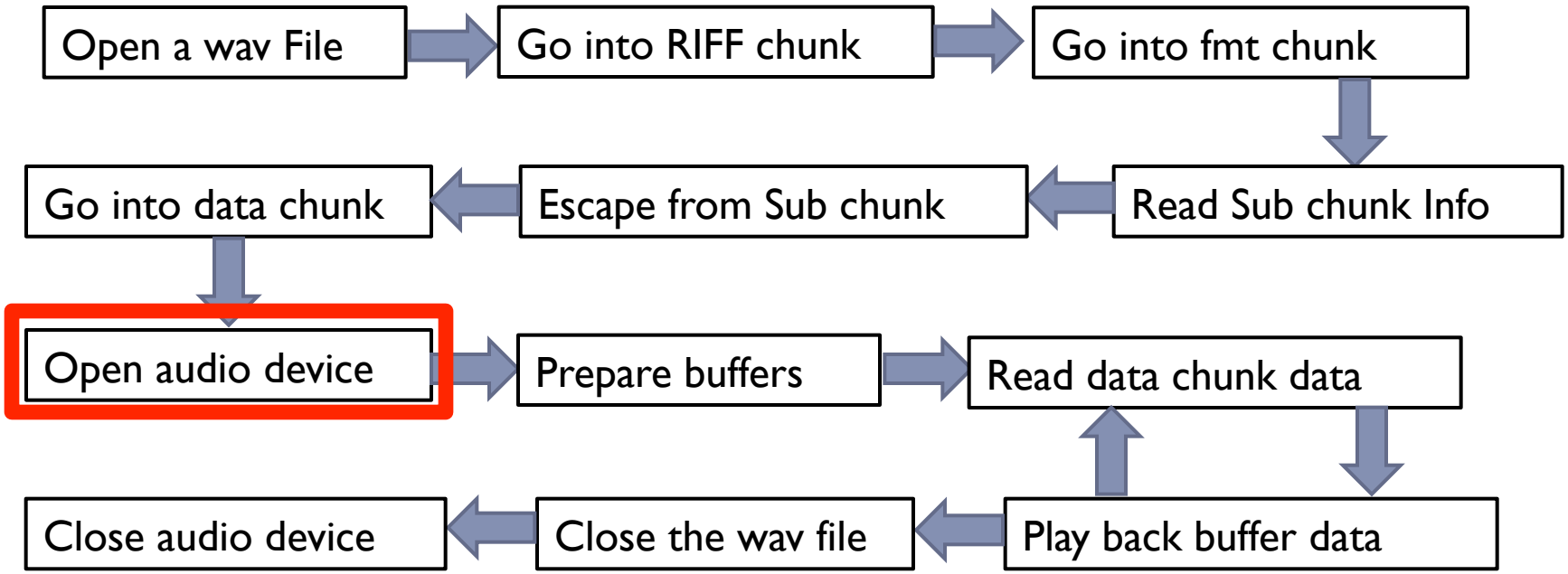


mmioAscend(HMMIO *h*, LPMMCKINFO *lpck*, UINT *flags*);

Ascends out of a chunk in a RIFF file

```
if (mmioAscend(hmmioIn, (LPMMCKINFO)&ckIn, 0)) {
    fprintf(stderr, "Error in mmioAscend\n");
    mmioClose(hmmioIn, 0);
    exit(-1);
}
```

Audio Play Procedure

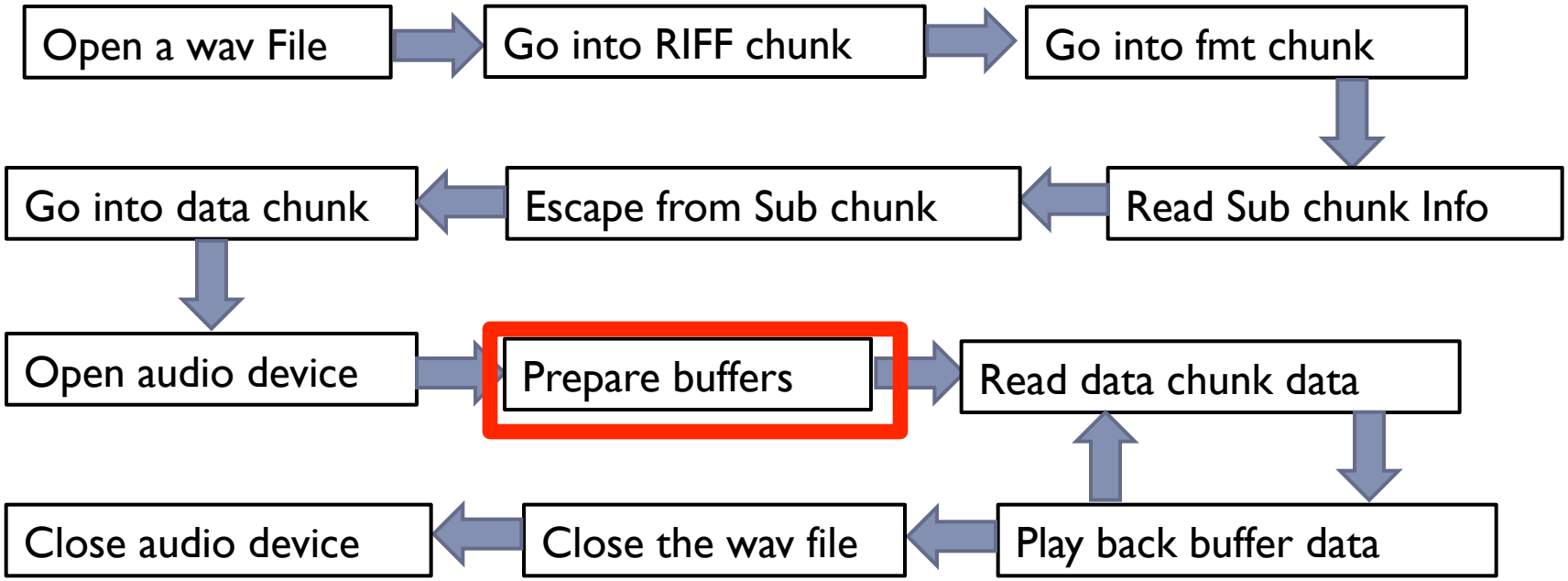


waveOutOpen(LPHWAVEOUT phwo, UINT uDeviceID, LPWAVEFORMATEX pwfx, DWORD dwCallback, DWORD Instance, DWORD fdwOpen);

```
err = waveOutOpen(&hAudioOut, WAVE_MAPPER, (WAVEFORMATEX *)&pcmWaveFormat, (DWORD)playSamples, 0L, CALLBACK_FUNCTION);

if (err) {
      waveOutGetErrorText(err, (LPSTR)errorMsg, sizeof(errorMsg));
      fprintf(stderr, "Error in WaveOutOpen: %s\n", errorMsg);
      return;
}
```

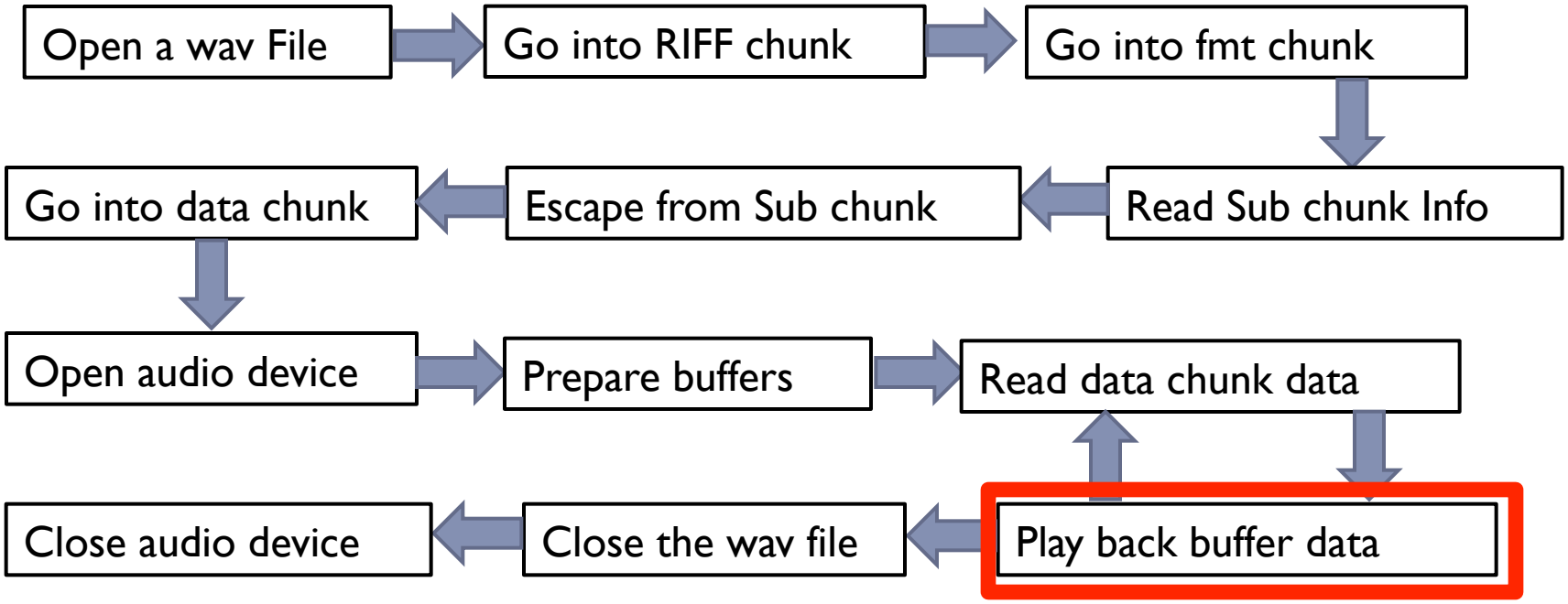
Audio Play Procedure



waveOutPrepareHeader(HWAVEOUT hwo, LPWAVEHDR pwh, UINT cbwh);

```
databuffer[i].lpData          = new char[BUFFER_SIZE];
databuffer[i].dwBufferLength = BUFFER_SIZE;
databuffer[i].dwFlags         = 0;
memset(databuffer[i].lpData, 0, BUFFER_SIZE);
err = waveOutPrepareHeader(hAudioOut, &databuffer[i], sizeof(WAVEHDR));
```

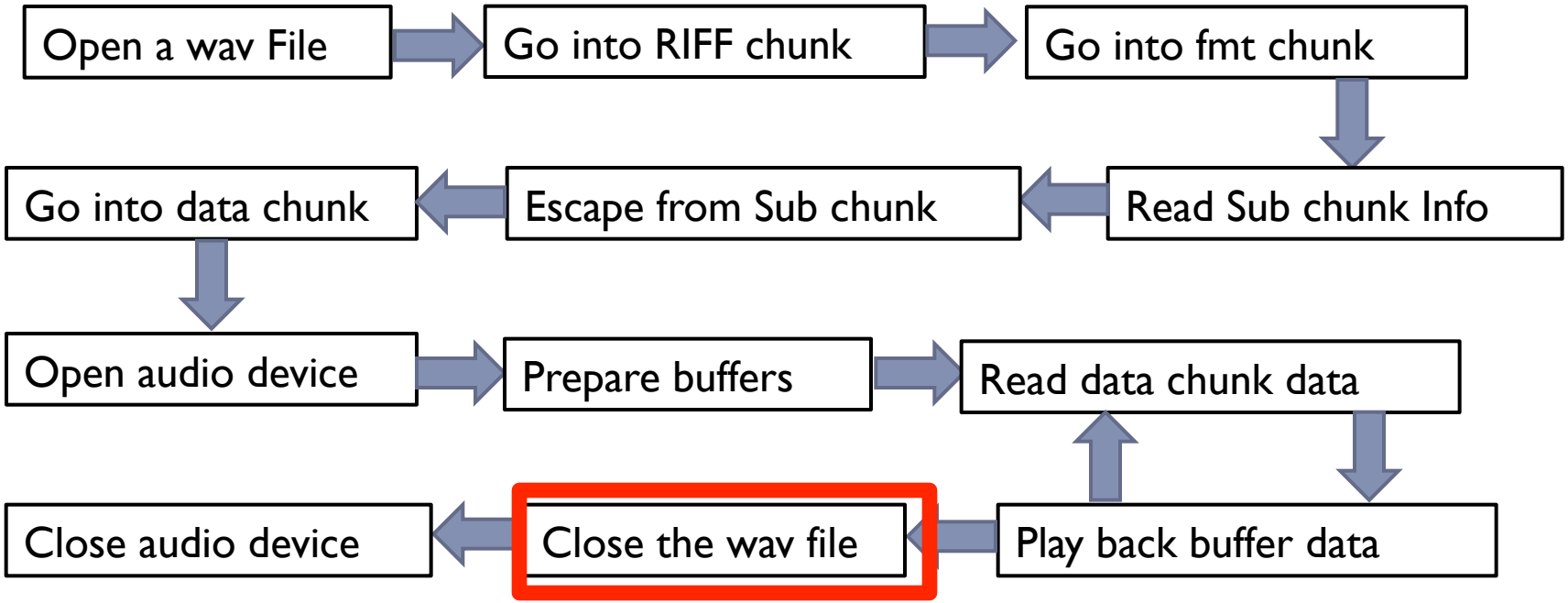
Audio Play Procedure



waveOutWrite (HWAVEOUT hwo, LPWAVEHDR pwh, UINT cbwh);

```
waveOutWrite(hAudioOut, &databuffer[0], sizeof(WAVEHDR));
```

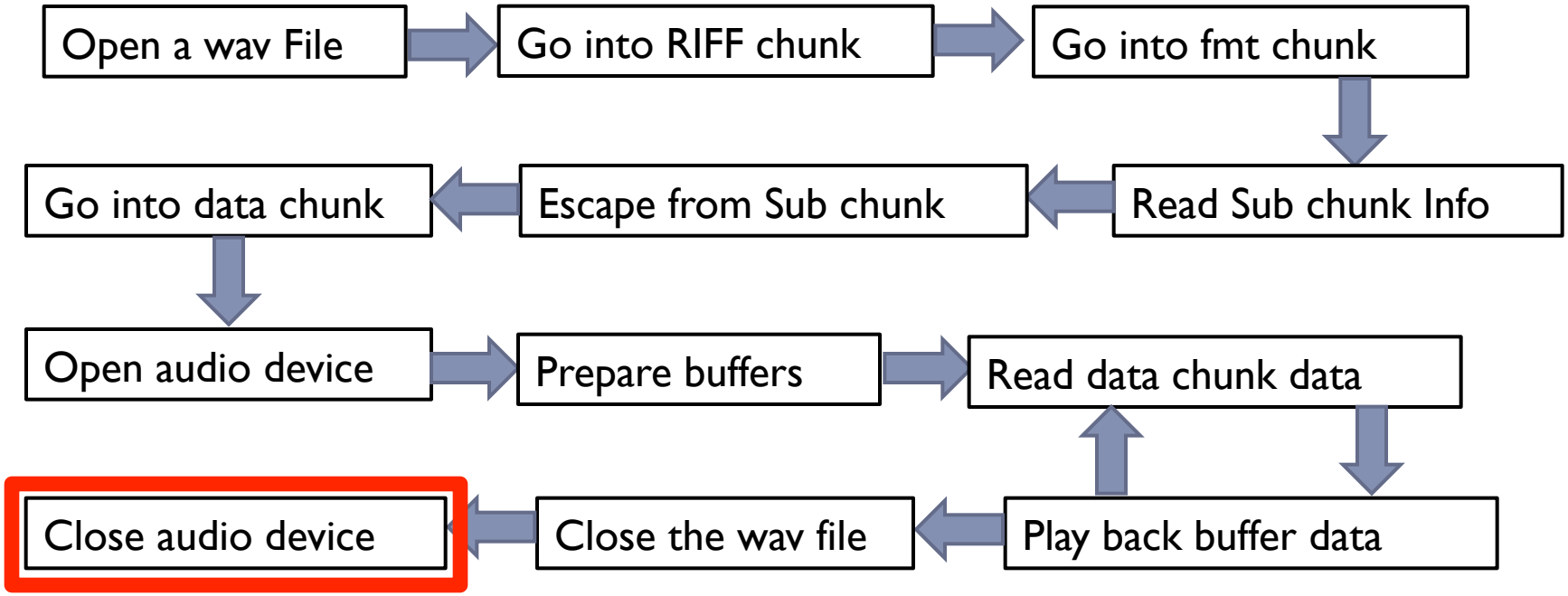
Audio Play Procedure



mmioClose (HMMIO h, UINT wflags);

mmioClose (hmmioIn, 0);

Audio Play Procedure



waveOutClose (HWAVEOUT hwo);

waveOutClose (hAudioOut);

Tutorial Next week

- Some useful functions
- Lyrics display

Thank you.