Data communication using single board computers

A breif Review of the program

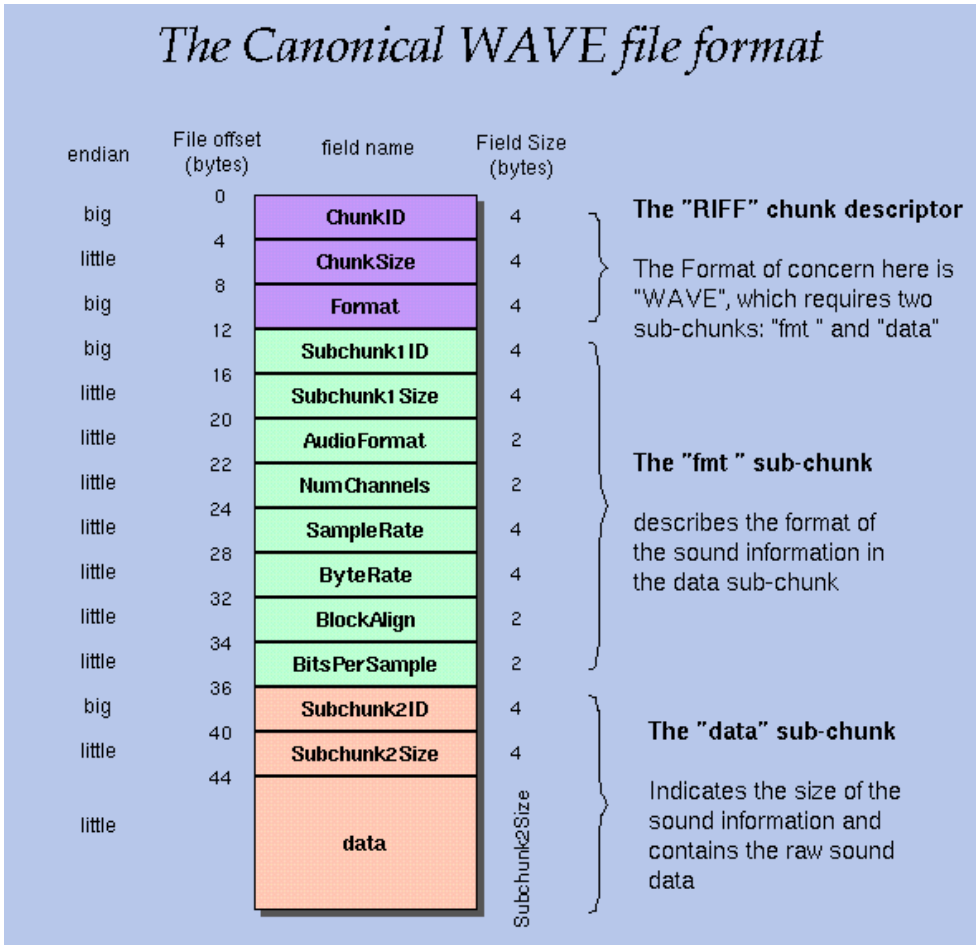
Presented by : Prakruti M Bilagi
2014IEN36
Central university of Karnataka

Guide : Dr. G V V Sharma
Dept of EE
Indian Institute of Technology Huderabad

The code package

- *Transmitter.h*
- *Transmitter.cpp*
- *Wave_reader.h*
- *Wave_reader.cpp*
- *Error_reporter.h*
- *Error_reporter.cpp*
- *Pcm_wave_header.h*
- *Audioformat.h*
- *Makefile*
- *Acaustic_guiter.wav*

Pcm_wave_header




```
struct PCMWaveHeader
{
char chunkID[4];
unsigned chunkSize;
char format[4];
char subchunk1ID[4];
unsigned subchunk1Size;
unsigned short audioFormat;
unsigned short channels;
unsigned sampleRate;
unsigned byteRate;
unsigned short blockAlign;
unsigned short bitsPerSam-
ple;
char subchunk2ID[4];
unsigned subchunk2Size;
};
```

Audioformat.h

```
struct AudioFormat
{
    unsigned short channels;
    unsigned short bitsPerSample;
    unsigned sampleRate;
};
```

Tranmitter.h

```
class Transmitter
{
    public:
        virtual ~Transmitter();
        void play(string filename, double frequency, bool loop);
        void stop();
        static Transmitter* getInstance();
    private:
        Transmitter();
        bool forceStop, eof;
        static void* peripherals;
        static vectorfloat *buffer;
        static bool transmitting, restart;
        static unsigned frameOffset, clockDivisor;
        static void* transmit(void* params);
};
```

Wave_reader.h

```
class WaveReader
{
    public:
        WaveReader(string filename, bool forceStop);
        virtual  WaveReader();
        AudioFormat* getFormat();
        vector¡float¿* getFrames(unsigned frameCount, bool forceStop);
        bool setFrameOffset(unsigned frameOffset);
    private:
        string filename;
        PCMWaveHeader header;
        unsigned dataOffset, currentOffset;
        int fileDescriptor;
        vectorchar *readData(unsignedbytesToRead, bool
        headerBytes, bool forceStop);
        string getFilename();
};
```

Tranmitter

- Include header files
- Assign the base addresses for gpio base clock base clockdividor base and a counter
- Assign initial values for variables to transmitter class members
- Check for the system, type and version of the host system.
- Assign memory for the GPIO pins using memFd and mmap system calls and this memory would be the base for assessing the GPIO and clock etc.
- In the function play()

a. It checks if the transmitter is already transmitting something.

b. Creates the objects of class Transmitter(Transmitter) and Wave_reader.(reader)

c. Grab the format of the reader and store it in "format" variable.

Transmitter conti:.

d Set the value to the clockDivisor.

e Create the bufferFrames that will be stored in the vector "frames"

f Create a vector "Frames" and use the fuction getFrames from class reader to stack the bufferFrames into it.

g Create thread to start the transmittion.

h Read thw whole data frames one by one.

i Set "transmitting" variable "false".

j Join the thread.

k Delete the reader and format finally while exiting.

Transmitter conti:.

- Next comes the transmit function where in does the changes in the clock frequecy for the transmittion purpose.