Lab 24

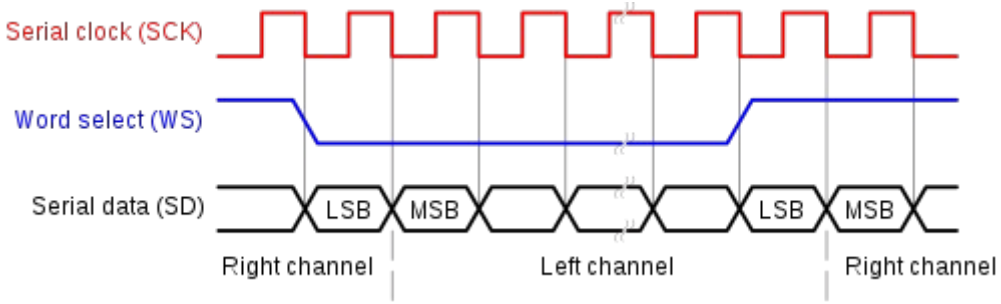
WEB/BT radio with PCM5102 on I2S bus and WiFiManager configuration

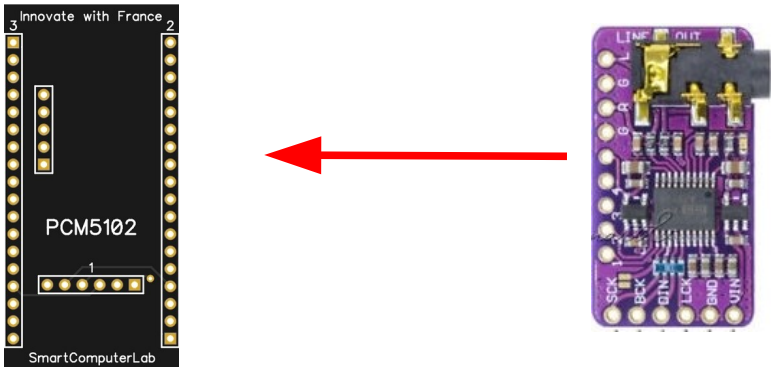
Content

Introduction

In this Lab we are going to develop the WEB radio applications using I2S bus and PCM5102 receiver and amplifier.
The following example shows how to exploit the Audio.h library containing MP3 and AAC decoders to decode the received media flow vi the WiFi connection.
Then the decoded PCM flow is sent to the external module via the I2S bus.

I2S (**Inter-IC Sound**), pronounced eye-squared-ess, is an electrical serial bus interface standard used for connecting digital audio devices together. It is used to communicate PCM audio data between integrated circuits in an electronic device. The I2S bus separates clock and serial data signals, resulting in simpler receivers than those required for asynchronous communications systems that need to recover the clock from the data stream. Despite the similar name, **I2S is unrelated** to the bidirectional **I2C bus**.





24.1 Simple WEB Radio for BBC stations

The following example is a simple WEB radio with predefined WiFi connection and fixed choice of BBC radio flows.

```
#include "Arduino.h"
#include "WiFiMulti.h"
#include "Audio.h"
// pin configuration for I2S bus - may be modified
#define I2S_DOUT      32  // D2
#define I2S_BCLK      14  // D3
#define I2S_LRC       2   // D1
#define buttonPin     0 //17
#include <U8x8lib.h>
U8X8_SSD1306_128X64_NONAME_SW_I2C u8x8(15, 4, 16);
Audio audio;
WiFiMulti wifiMulti;
String ssid1 =     "Livebox-08B0";
String password1 = "...";
String ssid2 = "PhoneAP";          //  your network SSID (name)
String password2 = "smartcomputerlab";   // your network passw
int setparam()
{
int count=0, max_count=10;;
int  buttonState;
char dbuf[32];
u8x8.clear();
u8x8.drawString(0, 0,"Set Station");
while(1)
  {
  buttonState = digitalRead(buttonPin);
  if(buttonState)
    {
      if(!count)count=1;else count=0;
      Serial.println(count);
      if(count) u8x8.drawString(4, 2,"YES");
      else u8x8.drawString(4, 2,"NO ");
      max_count--; if(!max_count) return 0;
    }
  else
    {
     Serial.println(count);
     if(count)
       {
       u8x8.drawString(0, 2,"BBC1");
       u8x8.drawString(0, 3,"BBC2");
       u8x8.drawString(0, 4,"BBC3");
       u8x8.drawString(0, 5,"BBC4");
       u8x8.drawString(0, 6,"BBC5_live");
       u8x8.drawString(0, 7,"BBC6_music");
       }
     else
       {
       u8x8.drawString(0, 2,"Setting Default");
       u8x8.drawString(0, 3,"BBC2");
       }
    delay(3000);
    return count;
    }
  delay(3000);
  }
}
```

```
int getvol()
{
int count=0, max_count=10;;
int  buttonState;
char dbuff[32];
u8x8.clear();
u8x8.drawString(0, 0,"Set Volume");
u8x8.drawString(0, 1,"Min:1, Max:21");
while(1)
  {
  buttonState = digitalRead(buttonPin);

  if(buttonState)
    {
      if(count==21)count=1;else count++;
      Serial.println(count);
      sprintf(dbuff,"%d",count);
      u8x8.drawString(4, 3,dbuff);
      max_count--; if(!max_count) return 2;
    }
  else
    {
    sprintf(dbuff,"Set to: %d",count);
    u8x8.drawString(4, 6,dbuff);
    delay(3000);
    return count;

    }
  delay(3000);
  }
}

int setstation()
{
int sf=1;
int  buttonState;
u8x8.clear();
char dbuf[32];
u8x8.drawString(0, 0,"Setting Station");
while(1)
  {
  buttonState = digitalRead(buttonPin);
  if(buttonState) { sf++; if(sf==7) sf=1;
                    Serial.println(sf);
                    sprintf(dbuf,"Station=%2.2d",sf);
                    u8x8.drawString(0, 2,dbuf);
                  }
  else
     {
      Serial.println(sf);
      u8x8.drawString(0, 4,"Station");
      sprintf(dbuf,"BBC%2.2d",sf);
             u8x8.drawString(0, 5,dbuf);
      return sf;
      }
  delay(3000);
  }
}

void setup() {
  int ret=0,sta=0;
    Serial.begin(9600);
    pinMode(buttonPin,INPUT_PULLUP);
```

```
    WiFi.mode(WIFI_STA);
    wifiMulti.addAP(ssid1.c_str(), password1.c_str());
    wifiMulti.addAP(ssid2.c_str(), password2.c_str());
    wifiMulti.run();
    if(WiFi.status() != WL_CONNECTED){
        WiFi.disconnect(true);
        wifiMulti.run();
      }
    Serial.println("WiFi connected");
    u8x8.begin();  // initialize OLED
    u8x8.setFont(u8x8_font_chroma48medium8_r);
    u8x8.clear();
    u8x8.drawString(0,0,"WiFi connected");
    audio.setPinout(I2S_BCLK, I2S_LRC, I2S_DOUT);
    audio.setVolume(getvol()); // 0...21
    ret=setparam();
    if(ret)
      {
        sta=setstation();
        if(sta==1)
audio.connecttohost("http://bbcmedia.ic.llnwd.net/stream/bbcmedia_radio1_mf_p");
        if(sta==2)
audio.connecttohost("http://bbcmedia.ic.llnwd.net/stream/bbcmedia_radio2_mf_p");
        if(sta==3)
audio.connecttohost("http://bbcmedia.ic.llnwd.net/stream/bbcmedia_radio3_mf_p");
        if(sta==4)
audio.connecttohost("http://bbcmedia.ic.llnwd.net/stream/bbcmedia_radio4fm_mf_p"
);
        if(sta==5)
audio.connecttohost("http://bbcmedia.ic.llnwd.net/stream/bbcmedia_radio5live_mf_
p");
        if(sta==6)
audio.connecttohost("http://bbcmedia.ic.llnwd.net/stream/bbcmedia_6music_mf_p");
      }
      else
audio.connecttohost("http://bbcmedia.ic.llnwd.net/stream/bbcmedia_radio2_mf_p");
}

void loop()
{
    audio.loop();
}

void audio_info(const char *info){
    Serial.print("info        "); Serial.println(info);
}
void audio_id3data(const char *info){  //id3 metadata
    Serial.print("id3data     ");Serial.println(info);
}
void audio_eof_mp3(const char *info){  //end of file
    Serial.print("eof_mp3     ");Serial.println(info);
}
void audio_showstation(const char *info){
    Serial.print("station     ");Serial.println(info);
    u8x8.drawString(0,2,"Webradio");
    u8x8.drawString(0,3,info);
}
void audio_showstreaminfo(const char *info){
    Serial.print("streaminfo  ");Serial.println(info);
}
void audio_showstreamtitle(const char *info){
    Serial.print("streamtitle ");Serial.println(info);
}
```

```
void audio_bitrate(const char *info){
    Serial.print("bitrate     ");Serial.println(info);
    u8x8.drawString(0,4,"bitrate");
    u8x8.drawString(8,4,info);
}
void audio_commercial(const char *info){  //duration in sec
    Serial.print("commercial  ");Serial.println(info);
}
void audio_icyurl(const char *info){  //homepage
    Serial.print("icyurl      ");Serial.println(info);
}
void audio_lasthost(const char *info){  //stream URL played
    Serial.print("lasthost    ");Serial.println(info);
}
void audio_eof_speech(const char *info){
    Serial.print("eof_speech  ");Serial.println(info);
}
```

24.2 Configurable WEB Radio for some Brazilian stations

The following code exploits the WiFiManager library that allows us to connect to the initially unknown WiFi Access Point. The application needs the use of smartphone connected to the given Access Point that will be used for our WEB radio.
During the first connection the WiFi of the WEB radio switches to **Access Point Mode** and we have to connect to it with our smartphone and open a WEB browser at IP address: **192.168.4.1**. Then we have to fill in the required fields : **SSID** and **Password** of the AP point to be used for streaming.
After few moments the program receives these data and switches to **Station Mode** in order to receive the valid IP address and start the streaming.

24.2.1 The code for ESP32-HELTEC WiFi-LoRa bard (V2)

```
#include "Arduino.h"
#include "Audio.h"
#include <WiFiManager.h>
WiFiManager wm;
const char* ssid = "ESP32";
const char* password = "smartcomputerlab";
#define I2S_DOUT      32
#define I2S_BCLK      14
#define I2S_LRC       2
#define buttonPin     17 //0
#include <U8x8lib.h>
U8X8_SSD1306_128X64_NONAME_SW_I2C u8x8(15, 4, 16);
Audio audio;

int setparam()
{
int count=0, max_count=10;;
int  buttonState;
char dbuf[32];
u8x8.clear();
u8x8.drawString(0, 1,"    Marcella    ");
u8x8.drawString(0, 3,"  and Jeremie");
u8x8.drawString(0, 5," Bresil station ");
delay(3000);
u8x8.clear();

while(1)
  {
  buttonState = digitalRead(buttonPin);
  if(buttonState)
    {
      if(!count)count=1;else count=0;
```

```
      Serial.println(count);
      if(count) u8x8.drawString(4, 2,"YES");
      else u8x8.drawString(4, 2,"NO ");
      max_count--; if(!max_count) return 0;
    }
  else
    {
     Serial.println(count);
     if(count)
       {
       u8x8.drawString(0, 2,"MA-Regae");
       u8x8.drawString(0, 3,"Bossa Nova Rio");
       u8x8.drawString(0, 4,"Radio Brazil");
       u8x8.drawString(0, 5,"Love Classic SP");
       u8x8.drawString(0, 6,"Viva Samba");
       u8x8.drawString(0, 7,"Studio Crancia");
       }
     else
       {
       u8x8.drawString(0, 1,"Setting Default");
       u8x8.drawString(0, 3,"Radio Nova SL");
       }
    delay(3000);
    return count;
    }
  delay(3000);
  }
}

int getvol()
{
int count=0, max_count=30;
int  buttonState;
char dbuff[32];
u8x8.clear();
u8x8.drawString(0, 0,"Set Volume");
u8x8.drawString(0, 1,"Min:1, Max:21");
while(1)
  {
  buttonState = digitalRead(buttonPin);

  if(buttonState)
    {
      if(count==21)count=1;else count++;
      Serial.println(count);
      sprintf(dbuff,"%d",count);
      u8x8.drawString(4, 3,dbuff);
      max_count--;
      if(!max_count)
        { u8x8.drawString(4, 6,"Set to: 10");return 10; }
    }
  else
    {
    sprintf(dbuff,"Set to: %d",count);
    u8x8.drawString(4, 6,dbuff);
    delay(3000);
    return count;

    }
  delay(3000);
  }
}
```

```
int setstation()
{
int sf=1;
int  buttonState;
u8x8.clear();
char dbuf[32];
u8x8.drawString(0, 0,"Setting Station");
while(1)
  {
  buttonState = digitalRead(buttonPin);
  if(buttonState)
    {
    sf++; if(sf==7) sf=1;
    Serial.println(sf);
      switch(sf)
       {
        case 1:  u8x8.drawString(0, 5,"MA-Regae        "); break;
        case 2:  u8x8.drawString(0, 5,"Bossa Nova Rio  "); break;
        case 3:  u8x8.drawString(0, 5,"Radio Brazil    "); break;
        case 4:  u8x8.drawString(0, 5,"Love Classic SP "); break;
        case 5:  u8x8.drawString(0, 5,"Viva Samba      "); break;
        case 6:  u8x8.drawString(0, 5,"Studio Crancia  "); break;
        default:  u8x8.drawString(0, 5,"Radio Nova SL   "); break;
       }
     }
  else
     {
      Serial.println(sf);
      u8x8.drawString(0, 4,"Station");
      switch(sf)
       {
        case 1:  u8x8.drawString(0, 5,"MA-Regae        "); break;
        case 2:  u8x8.drawString(0, 5,"Bossa Nova Rio  "); break;
        case 3:  u8x8.drawString(0, 5,"Radio Brazil    "); break;
        case 4:  u8x8.drawString(0, 5,"Love Classic SP "); break;
        case 5:  u8x8.drawString(0, 5,"Viva Samba      "); break;
        case 6:  u8x8.drawString(0, 5,"Studio Crancia  "); break;
        default:  u8x8.drawString(0, 5,"Radio Nova SL   "); break;
       }
      return sf;
      }
  delay(3000);
  }
}

int count=20;
char dbuff[32];

void setup() {
 int ret=0,sta=0;
    u8x8.begin();  // initialize OLED
    u8x8.setFont(u8x8_font_chroma48medium8_r);
    u8x8.clear();
    u8x8.drawString(0,0,"Starting WiFi");
    pinMode(buttonPin,INPUT_PULLUP);
  Serial.begin(9600);  // Initialisation du moniteur série
  delay(1000);
  Serial.println("\n");
  WiFi.mode(WIFI_STA);
  Serial.begin(9600);
  WiFi.begin();
 //Wait for WiFi to connect to AP
  Serial.println("Waiting for WiFi");
```

```
  while (WiFi.status()!= WL_CONNECTED && count>0) {
    delay(500); count--;
    Serial.print(count);
  }
 if(!count)
    {
    if(!wm.autoConnect(ssid, password))  // Test d'auto-connexion
      Serial.println("Erreur de connexion.");  // Si pas de connexion = Erreur
    else
      Serial.println("Connexion etablie !");  // Si connexion = OK
    }
  Serial.println("WiFi Connected.");
  u8x8.drawString(0,5,"WiFi Connected");
  Serial.print("IP Address: ");
  Serial.println(WiFi.localIP());
  delay(3000);
  audio.setPinout(I2S_BCLK, I2S_LRC, I2S_DOUT);
  audio.setVolume(getvol()); // 0...21

    ret=setparam();
    if(ret)
      {
        sta=setstation();
        if(sta==1)
audio.connecttohost("http://stm21.srvaudio.com.br:10782/stream?1588957954462");
        if(sta==2) audio.connecttohost("http://54.38.43.201:8009/stream-128kmp3-
BossaNovaBrazil");
        if(sta==3)
audio.connecttohost("http://streams.calmradio.com:20028/stream/1/");
        if(sta==4) audio.connecttohost("http://sc-loveclassic.1.fm:10052/");
        if(sta==5) audio.connecttohost("http://74.63.237.84:8020/live");
        if(sta==6) audio.connecttohost("http://192.99.150.31:9431/stream/1/");
      }
}

int scon=0;

void loop()
{
  audio.loop();
}

void audio_info(const char *info){
    Serial.print("info        "); Serial.println(info);
}
void audio_id3data(const char *info){  //id3 metadata
    Serial.print("id3data     ");Serial.println(info);
}
void audio_eof_mp3(const char *info){  //end of file
    Serial.print("eof_mp3     ");Serial.println(info);
}
void audio_showstation(const char *info){
    Serial.print("station     ");Serial.println(info);
    u8x8.drawString(0,4,info);
}
void audio_showstreaminfo(const char *info){
    Serial.print("streaminfo  ");Serial.println(info);
}
void audio_showstreamtitle(const char *info){
    Serial.print("streamtitle ");Serial.println(info);
}
void audio_bitrate(const char *info){
    Serial.print("bitrate     ");Serial.println(info);
```

```
}
void audio_commercial(const char *info){  //duration in sec
    Serial.print("commercial  ");Serial.println(info);
}
void audio_icyurl(const char *info){  //homepage
    Serial.print("icyurl      ");Serial.println(info);
}
void audio_lasthost(const char *info){  //stream URL played
    Serial.print("lasthost    ");Serial.println(info);
}
void audio_eof_speech(const char *info){
    Serial.print("eof_speech  ");Serial.println(info);
}
```

24.2.2 The code for ESP32-Wemos D1 – MH ET LIVE MiniKit

The following cod operates on ESP32 Wemos -D1 (**MH ET LIVE MiniKit**) board with separate OLED screen.

```
#include "Arduino.h"
#include "Audio.h"
#include <WiFiManager.h>
#include <Wire.h>
#include "SSD1306.h"

SSD1306  display(0x3c, 21, 22);

WiFiManager wm;
const char* ssid = "ESP32";
const char* password = "smartcomputerlab";
// Digit
#define I2S_DOUT      32
#define I2S_BCLK      14
#define I2S_LRC       2
#define buttonPin     17

Audio audio;

int setparam()
{
int count=0, max_count=10;;
int  buttonState;
char dbuf[32];
display.clear();
display.setTextAlignment(TEXT_ALIGN_CENTER);display.setFont(ArialMT_Plain_16);
display.drawString(64, 0,"Set Station");      display.display();
delay(2000);
while(1)
  {
  buttonState = digitalRead(buttonPin);
  if(buttonState)
    {
      display.clear();
      if(!count)count=1;else count=0;

Serial.println(count);display.clear();display.setTextAlignment(TEXT_ALIGN_CENTER
);
      display.setFont(ArialMT_Plain_24);
      if(count) display.drawString(64, 0,"YES");
      else display.drawString(64, 0,"NO ");
      display.display();
```

```
      max_count--; if(!max_count) return 0;
    }
  else
    {
     Serial.println(count);
     if(count)
       {

display.clear();display.setFont(ArialMT_Plain_10);display.setTextAlignment(TEXT_
ALIGN_CENTER);
       display.drawString(64, 0,"MA-Regae");
       display.drawString(64, 8,"Bossa Nova Rio");
       display.drawString(64, 16,"Radio Brazil");
       display.drawString(64, 24,"Love Classic SP");
       display.drawString(64, 32,"Viva Samba");
       display.drawString(64, 48,"Studio Crancia");
       display.display();
       }
     else
       {
       display.clear();display.setFont(ArialMT_Plain_16);
       display.setTextAlignment(TEXT_ALIGN_CENTER);
       display.drawString(64, 0,"Radio Nova SL");
       display.display();
       }
    delay(3000);
    return count;
    }
  delay(3000);
  }
}

int getvol()
{
int count=0, max_count=30;
int  buttonState;
char dbuff[32];
display.clear();     display.setFont(ArialMT_Plain_16);
display.setTextAlignment(TEXT_ALIGN_CENTER);
display.drawString(64, 0,"Set Volume");
display.drawString(64, 20,"1 to 21");display.display();
delay(2000);
while(1)
  {
  buttonState = digitalRead(buttonPin);

  if(buttonState)
    {
      if(count==21)count=1;else count++;
      Serial.println(count);
      sprintf(dbuff,"%d",count);display.clear();
      display.setFont(ArialMT_Plain_24);
      display.setTextAlignment(TEXT_ALIGN_CENTER);
      display.drawString(64, 20,dbuff);display.display();
      max_count--;
      if(!max_count)
        { display.drawString(4, 30,"Set to: 10");return 10; display.display();}
    }
  else
    {
    display.clear();display.setTextAlignment(TEXT_ALIGN_CENTER);
    display.setFont(ArialMT_Plain_16);
    sprintf(dbuff,"Set to: %d",count);
```

```
    display.drawString(64, 0,dbuff);display.display();
    delay(3000);
    return count;
    }
  delay(2000);
  }
}

int setstation()
{
int sf=1;
int  buttonState;
display.clear();
char dbuf[32];
display.setFont(ArialMT_Plain_16);display.setTextAlignment(TEXT_ALIGN_CENTER);
display.drawString(64, 0,"Setting Station");display.display();
delay(2000);
while(1)
  {
  buttonState = digitalRead(buttonPin);
  if(buttonState)
    {
    sf++; if(sf==7) sf=1;
    Serial.println(sf);
    display.clear();
      switch(sf)
       {
        case 1:  display.drawString(64, 0,"MA-Regae        "); break;
        case 2:  display.drawString(64, 0,"Bossa Nova Rio  "); break;
        case 3:  display.drawString(64, 0,"Radio Brazil    "); break;
        case 4:  display.drawString(64, 0,"Love Classic SP "); break;
        case 5:  display.drawString(64, 0,"Viva Samba      "); break;
        case 6:  display.drawString(64, 0,"Studio Crancia  "); break;
        default:  display.drawString(64, 0,"Radio Nova SL   "); break;
       }
     display.display();
     }
  else
     {
      Serial.println(sf);
      display.clear();
      switch(sf)
       {
        case 1:  display.drawString(64, 0,"MA-Regae        "); break;
        case 2:  display.drawString(64, 0,"Bossa Nova Rio  "); break;
        case 3:  display.drawString(64, 0,"Radio Brazil    "); break;
        case 4:  display.drawString(64, 0,"Love Classic SP "); break;
        case 5:  display.drawString(64, 0,"Viva Samba      "); break;
        case 6:  display.drawString(64, 0,"Studio Crancia  "); break;
        default:  display.drawString(64, 0,"Radio Nova SL   "); break;
       }
      display.display();
      return sf;
      }
  delay(3000);
  }
}

int count=20;

void setup() {
  int ret=0,sta=0;
  Serial.begin(9600);
```

```
    display.init();  // initialize OLED
    display.setFont(ArialMT_Plain_16);
    display.drawString(0,0,"Starting WiFi"); display.display();
    pinMode(buttonPin,INPUT_PULLUP);
  // -- Initializing the configuration.
  WiFi.begin();
 //Wait for WiFi to connect to AP
  Serial.println("Waiting for WiFi");
  while (WiFi.status()!= WL_CONNECTED && count>0) {
    delay(500); count--;
    Serial.print(count);
  }
  if(!count)
    {
    display.drawString(0,1,"SmartConfig");
    display.drawString(0,2,"Launch App");
    display.drawString(0,3,"Set SSID");
    display.drawString(0,4,"Enter PASS");
    delay(2000);
    if(!wm.autoConnect(ssid, password))
    Serial.println("Erreur de connexion.");
  else
    Serial.println("Connexion etablie!");
    delay(1000);
    Serial.println("");
    //Wait for WiFi to connect to AP
    Serial.println("Waiting for WiFi");
    while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
    }
  }
  Serial.println("WiFi Connected.");
  display.drawString(0,20,"WiFi Connected"); display.display();
  Serial.print("IP Address: ");
  Serial.println(WiFi.localIP());
  delay(3000);
  audio.setPinout(I2S_BCLK, I2S_LRC, I2S_DOUT);
  audio.setVolume(getvol()); // 0...21
  delay(2000);
  ret=setparam();
  if(ret)
    {
    sta=setstation();
        if(sta==1)
audio.connecttohost("http://stm21.srvaudio.com.br:10782/stream?1588957954462");
        if(sta==2) audio.connecttohost("http://54.38.43.201:8009/stream-128kmp3-
BossaNovaBrazil");
        if(sta==3)
audio.connecttohost("http://streams.calmradio.com:20028/stream/1/");
        if(sta==4) audio.connecttohost("http://sc-loveclassic.1.fm:10052/");
        if(sta==5) audio.connecttohost("http://74.63.237.84:8020/live");
        if(sta==6) audio.connecttohost("http://192.99.150.31:9431/stream/1/");
      }
      else
        {
        display.clear();display.setFont(ArialMT_Plain_16);
        display.setTextAlignment(TEXT_ALIGN_CENTER);
        display.drawString(64, 0,"Radio Nova SL   ");display.display();
          audio.connecttohost("http://cast3.hoost.com.br:9071/live");
          // Radio Nova Sao Luis
        }
}
```

```
int scon=0;

void loop()
{
  audio.loop();
}

void displine(char *lines)
{
display.clear();display.setFont(ArialMT_Plain_10);display.setTextAlignment(TEXT_
ALIGN_LEFT);
display.drawStringMaxWidth(0, 0,128,lines);display.display();
}
// optional
void audio_info(const char *info){
    Serial.print("info        "); Serial.println(info);
}
void audio_id3data(const char *info){  //id3 metadata
    Serial.print("id3data     ");Serial.println(info);
}
void audio_eof_mp3(const char *info){  //end of file
    Serial.print("eof_mp3     ");Serial.println(info);
}
void audio_showstation(const char *info){
    Serial.print("station     ");Serial.println(info);
    //displine((char *)info);
}
void audio_showstreaminfo(const char *info){
    Serial.print("streaminfo  ");Serial.println(info);
}
void audio_showstreamtitle(const char *info){
    Serial.print("streamtitle ");Serial.println(info);
    displine((char *)info);
}
void audio_bitrate(const char *info){
    Serial.print("bitrate     ");Serial.println(info);
}
void audio_commercial(const char *info){  //duration in sec
    Serial.print("commercial  ");Serial.println(info);
}
void audio_icyurl(const char *info){  //homepage
    Serial.print("icyurl      ");Serial.println(info);
}
void audio_lasthost(const char *info){  //stream URL played
    Serial.print("lasthost    ");Serial.println(info);
}
void audio_eof_speech(const char *info){
    Serial.print("eof_speech  ");Serial.println(info);
}
```

24.3 Simple Web Radio receiver (BBC2) – short ext board

The following code has been developed for small extension card and PCM5102 I2S receiver and audio player.

```
#include "Arduino.h"
#include "WiFiMulti.h"
#include "Audio.h"
// pin configuration for I2S bus - may be modified
#define I2S_DOUT      21  //32 // 21  // D2
#define I2S_BCLK      17 //14   //17  // D3
#define I2S_LRC       22   // 2 D1
#define buttonPin     0 //m
#include <U8x8lib.h>
U8X8_SSD1306_128X64_NONAME_SW_I2C u8x8(15, 4, 16);
Audio audio;
WiFiMulti wifiMulti;

String ssid1 = "Livebox-08B0";
String password1 = "G79ji6dtEptVTPWmZP";
String ssid2 = "PhoneAP";          //  your network SSID (name)
String password2 = "smartcomputerlab";   // your network passw

void setup() {
  int ret=0,sta=0;
    Serial.begin(9600);

pinMode(0,INPUT_PULLUP);//pinMode(17,OUTPUT);pinMode(21,OUTPUT);pinMode(22,OUTPU
T);
    WiFi.mode(WIFI_STA);
    wifiMulti.addAP(ssid1.c_str(), password1.c_str());
    wifiMulti.addAP(ssid2.c_str(), password2.c_str());
    wifiMulti.run();
    if(WiFi.status() != WL_CONNECTED){
        WiFi.disconnect(true);
        wifiMulti.run();
      }
    Serial.println("WiFi connected");
    u8x8.begin();  // initialize OLED
    u8x8.setFont(u8x8_font_chroma48medium8_r);
    u8x8.clear();
    u8x8.drawString(0,0,"WiFi connected");
    audio.setPinout(I2S_BCLK, I2S_LRC, I2S_DOUT);
    audio.setVolume(6); // 0...21

audio.connecttohost("http://bbcmedia.ic.llnwd.net/stream/bbcmedia_radio2_mf_p");
}

void loop()
{
    audio.loop();
}

void audio_info(const char *info){
    Serial.print("info        "); Serial.println(info);
}
```

```
void audio_id3data(const char *info){  //id3 metadata
    Serial.print("id3data     ");Serial.println(info);
}

void audio_eof_mp3(const char *info){  //end of file
    Serial.print("eof_mp3     ");Serial.println(info);
}

void audio_showstation(const char *info){
    Serial.print("station     ");Serial.println(info);
    u8x8.drawString(0,2,"Webradio");
    u8x8.drawString(0,3,info);
}

void audio_showstreaminfo(const char *info){
    Serial.print("streaminfo  ");Serial.println(info);
}

void audio_showstreamtitle(const char *info){
    Serial.print("streamtitle ");Serial.println(info);
}

void audio_bitrate(const char *info){
    Serial.print("bitrate     ");Serial.println(info);
    u8x8.drawString(0,4,"bitrate");
    u8x8.drawString(8,4,info);
}

void audio_commercial(const char *info){  //duration in sec
    Serial.print("commercial  ");Serial.println(info);
}

void audio_icyurl(const char *info){  //homepage
    Serial.print("icyurl      ");Serial.println(info);
}

void audio_lasthost(const char *info){  //stream URL played
    Serial.print("lasthost    ");Serial.println(info);
}

void audio_eof_speech(const char *info){
    Serial.print("eof_speech  ");Serial.println(info);
}
```

24.4 WebRadio with RF transmitter

The following example shows how to build the combined WebRadio with FM transmitter.
We start with the initial example of simple WebRadio receiver for one Brazilian station (Sao Luis) , and we add the RF transmitter from ELECHOUSE.
The RF transmitter may be controlled via an I2C bus to set the transmission frequency and some other parameters.
Below we show two versions of the code, one including the control of all parameters, the second only with the setting of the transmission frequency. We start with the simple WebRadio receiver operating on our ESP32 mini board.

24.4.1 Complete code for WebRadio receiver for ESP32 mini board

```
#include "Arduino.h"
#include "Audio.h"
#include <WiFi.h>
char *ssid = "Livebox-08B0";
char *pass = "G79ji6dtE..VTPWmZP";

#define I2S_DOUT      12  // mini
#define I2S_BCLK      16  // mini
#define I2S_LRC       14  // mini

Audio audio;

void setup() {
    Serial.begin(9600);
    WiFi.mode(WIFI_STA);
    WiFi.begin(ssid,pass);
    if(WiFi.status() != WL_CONNECTED){
    delay(500); Serial.print(".");
      }
  Serial.println("WiFi connected");
  delay(3000);
  Serial.println("WiFi connected");
  audio.setPinout(I2S_BCLK, I2S_LRC, I2S_DOUT);
  audio.setVolume(8); // 0...21
  audio.connecttohost("http://cast3.hoost.com.br:9071/live");
}

void loop() {
      audio.loop();
}

void audio_info(const char *info){
    Serial.print("info        "); Serial.println(info);}

void audio_id3data(const char *info){  //id3 metadata
    Serial.print("id3data     ");Serial.println(info);}

void audio_eof_mp3(const char *info){  //end of file
    Serial.print("eof_mp3     ");Serial.println(info);}

void audio_showstation(const char *info){
    Serial.print("station     ");Serial.println(info);}

void audio_showstreaminfo(const char *info){
    Serial.print("streaminfo  ");Serial.println(info);}

void audio_showstreamtitle(const char *info){
    Serial.print("streamtitle ");Serial.println(info);}

void audio_bitrate(const char *info){
    Serial.print("bitrate     ");Serial.println(info);}

void audio_commercial(const char *info){  //duration in sec
    Serial.print("commercial  ");Serial.println(info);}

void audio_icyurl(const char *info){  //homepage
    Serial.print("icyurl      ");Serial.println(info);}

void audio_lasthost(const char *info){  //stream URL played
    Serial.print("lasthost    ");Serial.println(info);
}

void audio_eof_speech(const char *info){
    Serial.print("eof_speech  ");Serial.println(info);}
```

Below we show two versions of the code, one including the control of all parameters, the second only with the setting of the transmission frequency.

24.4.2 Complete code for WebRadio receiver and RF transmitter for ESP32 mini board

The following code includes all functions of RF transmitter activated from the additional task. The I2C bus is mapped on two lines **0** – for **SDA** and **2** for **SCL** (note that to program this board you need disconnect these lines during code uploading).

```
#include <XantoI2C.h>
#include "XantoKT0803L.h"
#include "Arduino.h"
#include "Audio.h"
#include <WiFi.h>

char *ssid = "Livebox-08B0";
char *pass = "G79ji6dtE..VTPWmZP";

#define I2S_DOUT      12 //mini // D2
#define I2S_BCLK      16 //mini // D3
#define I2S_LRC       14  //mini // 2 D1
Audio audio;

const uint8_t PIN_SCL = 2;  // configuration task
const uint8_t PIN_SDA = 0;
XantoKT0803L fm(PIN_SCL, PIN_SDA);
char snprintf_buffer[19] = "0x00 76543210 0x00";

void TRANS_Task(void * pvParameters)
{
while(true)
  {
   if (Serial.available()) {
    byte cmd = Serial.read();
    if (cmd == 'W') {
      String str = Serial.readString();
      int register_address, value;
      sscanf(str.c_str(), "%i %i", &register_address, &value);
      writeAndPrintRegister(register_address, value);
    } else if (cmd == 'R') {
      String str = Serial.readString();
      int register_address;
      sscanf(str.c_str(), "%i %i", &register_address);
      readAndPrintRegister(register_address);
    } else if (cmd == 'F') {
      setFrequency(Serial.parseInt());
    } else if (cmd == 'P') {
      readAndPrintAllRegisters();
    } else if (cmd == 'M') {
      Serial.println("Mute");
      fm.mute(1);
      printErrorIfExists();
    } else if (cmd == 'U') {
      Serial.println("Unmute");
      fm.mute(0);
      printErrorIfExists();
    } else {
      Serial.println("Unknown command");
      printUsage();
    }
  }
 delay(10);
 }
}

void setup() {
  Serial.begin(9600);
  WiFi.mode(WIFI_STA);
  WiFi.begin(ssid,pass);
  if(WiFi.status() != WL_CONNECTED){
  delay(500); Serial.print(".");  }
  Serial.println("WiFi connected");
  delay(3000);
  Serial.println("WiFi connected");
  printUsage();
```

```
  xTaskCreatePinnedToCore(
                    TRANS_Task,   /* Function to implement the task */
                    "TRANS_Task", /* Name of the task */
                    10000,      /* Stack size in words */
                    NULL,       /* Task input parameter */
                    0,          /* Priority of the task */
                    NULL,       /* Task handle. */
                    0);  /* Core where the task should run */
  Serial.println("TRANS_Task created...");
  audio.setPinout(I2S_BCLK, I2S_LRC, I2S_DOUT);
  audio.setVolume(8); // 0...21
  audio.connecttohost("http://cast3.hoost.com.br:9071/live");
}

void printUsage() {
  Serial.println("Usage:");
  Serial.println(" -Write register: W register_address_hex value_hex");
  Serial.println(" -Read register: R register_address_hex");
  Serial.println(" -Set Frequency: F frequency*10");
  Serial.println(" -Print all registers: P");
  Serial.println(" -Mute: M");
  Serial.println(" -Unmute: U");
  Serial.println("E.g.: \"W 0x02 0x40\" - write value=0x40 to the register with address=0x02");
  Serial.println("E.g.: \"R 0x02\" - read the register with address=0x02");
  Serial.println("E.g.: \"F 997\" - set radio frequency=99.7MHz");
}

void printErrorIfExists() {
  if (fm.error > 0) {
    Serial.print("Error: ");
    Serial.println(fm.error);
    fm.error = 0;
  }
}

void printRegister(uint8_t register_address, uint8_t value) {
  snprintf(snprintf_buffer, 19, "0x%02X %d%d%d%d%d%d%d%d 0x%02X",
           register_address,
           bitRead(value, 7),
           bitRead(value, 6),
           bitRead(value, 5),
           bitRead(value, 4),
           bitRead(value, 3),
           bitRead(value, 2),
           bitRead(value, 1),
           bitRead(value, 0),
           value
          );
  Serial.println(snprintf_buffer);
}

void readAndPrintRegister(uint8_t register_address) {
  Serial.print("Read: ");
  Serial.println(register_address, HEX);

  uint8_t value = fm.read(register_address);
  printErrorIfExists();
  Serial.println("Register address, BIN value, HEX value:");
  printRegister(register_address, value);
}

void readAndPrintAllRegisters() {
  Serial.println("Read all: ");
  Serial.println("Register address, BIN value, HEX value:");
  for (uint8_t i = 0; i < fm.getRegistersCount(); i++) {
    uint8_t value = fm.read(fm.getRegisters()[i]);
    printErrorIfExists();
    printRegister(fm.getRegisters()[i], value);
  }
  Serial.println("Done");
}

void writeAndPrintRegister(uint8_t register_address, uint8_t value) {
  Serial.print("Write: ");
  Serial.print(register_address, HEX);
  Serial.print(" ");
  Serial.println(value, HEX);

  value = fm.write(register_address, value);
```

```
  printErrorIfExists();
  Serial.println("Register address, BIN value, HEX value:");
  printRegister(register_address, value);
  value = fm.read(register_address);
  printErrorIfExists();
  Serial.println("Register address, BIN value, HEX value:");
  printRegister(register_address, value);
}

void setFrequency(uint16_t frequency) {
  Serial.print("Set frequency: ");
  Serial.print(frequency / 10.0);
  Serial.println("MHz");
  fm.setFrequency(frequency);
}

void loop() {
      audio.loop();
}

void audio_info(const char *info){
    Serial.print("info        "); Serial.println(info);
}

void audio_id3data(const char *info){  //id3 metadata
    Serial.print("id3data     ");Serial.println(info);}

void audio_eof_mp3(const char *info){  //end of file
    Serial.print("eof_mp3     ");Serial.println(info);}

void audio_showstation(const char *info){
    Serial.print("station     ");Serial.println(info);}

void audio_showstreaminfo(const char *info){
    Serial.print("streaminfo  ");Serial.println(info);}

void audio_showstreamtitle(const char *info){
    Serial.print("streamtitle ");Serial.println(info);}

void audio_bitrate(const char *info){
    Serial.print("bitrate     ");Serial.println(info);}

void audio_commercial(const char *info){  //duration in sec
    Serial.print("commercial  ");Serial.println(info);}

void audio_icyurl(const char *info){  //homepage
    Serial.print("icyurl      ");Serial.println(info);}

void audio_lasthost(const char *info){  //stream URL played
    Serial.print("lasthost    ");Serial.println(info);}

void audio_eof_speech(const char *info){
    Serial.print("eof_speech  ");Serial.println(info);}
```

24.4.3 Simplified code for WebRadio receiver and RF transmitter for ESP32 mini board

In the following example we set just one FM parameter – the frequency. The complete setting task presented above is no more needed.

```
#include <XantoI2C.h>
#include "XantoKT0803L.h"
#include "Arduino.h"
#include "Audio.h"
#include <WiFi.h>
char *ssid = "Livebox-08B0";
char *pass = "G79ji6dtE..VTPWmZP";

#define I2S_DOUT      12 //mini // D2
#define I2S_BCLK      16 //mini // D3
#define I2S_LRC       14  //mini // 2 D1

Audio audio;

const uint8_t PIN_SCL = 14;  // configuration in setup - may be shared
const uint8_t PIN_SDA = 12;
```

```
XantoKT0803L fm(PIN_SCL, PIN_SDA);
char snprintf_buffer[19] = "0x00 76543210 0x00";

void setFrequency(uint16_t frequency) {
  Serial.print("Set frequency: ");
  Serial.print(frequency / 10.0);
  Serial.println("MHz");
  fm.setFrequency(frequency);
}

uint16_t freq=992;

void setup() {
  Serial.begin(9600);
  WiFi.mode(WIFI_STA);
  WiFi.begin(ssid,pass);
  if(WiFi.status() != WL_CONNECTED){
  delay(500); Serial.print(".");  }
  Serial.println("WiFi connected");
  delay(3000);
  setFrequency(freq);  // transmission frequency - to be modified !
  audio.setPinout(I2S_BCLK, I2S_LRC, I2S_DOUT);
  audio.setVolume(8); // 0...21
  audio.connecttohost("http://cast3.hoost.com.br:9071/live");
}

void loop() {
      audio.loop();
}

void audio_info(const char *info){
    Serial.print("info        "); Serial.println(info);}

void audio_id3data(const char *info){  //id3 metadata
    Serial.print("id3data     ");Serial.println(info);}

void audio_eof_mp3(const char *info){  //end of file
    Serial.print("eof_mp3     ");Serial.println(info);}

void audio_showstation(const char *info){
    Serial.print("station     ");Serial.println(info);}

void audio_showstreaminfo(const char *info){
    Serial.print("streaminfo  ");Serial.println(info);}

void audio_showstreamtitle(const char *info){
    Serial.print("streamtitle ");Serial.println(info);}

void audio_bitrate(const char *info){
    Serial.print("bitrate     ");Serial.println(info);}

void audio_commercial(const char *info){  //duration in sec
    Serial.print("commercial  ");Serial.println(info);}

void audio_icyurl(const char *info){  //homepage
    Serial.print("icyurl      ");Serial.println(info);}

void audio_lasthost(const char *info){  //stream URL played
    Serial.print("lasthost    ");Serial.println(info);}

void audio_eof_speech(const char *info){
    Serial.print("eof_speech  ");Serial.println(info);}
```

24.5 BT audio link – music receiver

The audio extension board may also be used with the Bluetooth connection to play the music from a Smartphone.
Below we give the code to implement on our DevKit with ESP32 and PCM5102 extension board.

```
#include <Arduino.h>
#include "Audio.h"
#include "esp32_bt_music_receiver.h"
#define I2S_DOUT      32
#define I2S_BCLK      14
#define I2S_LRC       2
#define buttonPin     0 //17 - depending on the control line and button

#include <U8x8lib.h>
U8X8_SSD1306_128X64_NONAME_SW_I2C u8x8(15, 4, 16);

Audio audio;

BlootoothA2DSink a2d_sink;

int getvol()
{
int count=0, max_count=30;
int  buttonState;
char dbuff[32];
u8x8.clear();
u8x8.drawString(0, 0,"Set Volume");
u8x8.drawString(0, 1,"Min:1, Max:21");
while(1)
  {
  buttonState = digitalRead(buttonPin);

  if(buttonState)
    {
      if(count==21)count=1;else count++;
      Serial.println(count);
      sprintf(dbuff,"%d",count);
      u8x8.drawString(4, 3,dbuff);
      max_count--;
      if(!max_count)
        { u8x8.drawString(4, 6,"Set to: 10");return 10; }
    }
  else
    {
    sprintf(dbuff,"Set to: %d",count);
    u8x8.drawString(4, 6,dbuff);
    delay(3000);
    return count;

    }
  delay(2000);
  }
}

void setup() {
  u8x8.begin();  // initialize OLED
  u8x8.setFont(u8x8_font_chroma48medium8_r);
  u8x8.clear();
  u8x8.drawString(0,0,"BT - MyMusic");
  pinMode(buttonPin,INPUT_PULLUP);
  audio.setPinout(I2S_BCLK, I2S_LRC, I2S_DOUT);
  audio.setVolume(getvol()); // 0...21
  u8x8.clear();
  u8x8.drawString(0,2,"BT - MyMusic");
  a2d_sink.start("MyMusic");
}

void loop() {
}
```

To do:

Test the above codes with modified parameters:

- Web sources for webradio streaming
- Frequency values for the RF transmitter
- Add display to show the parameters and info

Table des matières