PRÁCTICA 7.2: REPRODUCIR UN ARCHIVO WAVE EN ESP32 DESDE UNA TARJETA SD EXTERNA

CÓDIGO

```
#include "Audio.h"
#include "SD.h"
#include "FS.h"

// Digital I/O used
#define SD_CS          5
#define SPI_MOSI      23
#define SPI_MISO      19
#define SPI_SCK       18
#define I2S_DOUT      25
#define I2S_BCLK      27
#define I2S_LRC       26

Audio audio;

void setup(){

    Serial.begin(115200);

    pinMode(SD_CS, OUTPUT);
    digitalWrite(SD_CS, HIGH);
    SPI.begin(SPI_SCK, SPI_MISO, SPI_MOSI);
    SD.begin(SD_CS);
    audio.setPinout(I2S_BCLK, I2S_LRC, I2S_DOUT);
    audio.setVolume(15); // 0...21
    audio.connecttoFS(SD, "Ensoniq-ZR-76-01-Dope-77.wav");
}

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

// 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);
}
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);
}
```

FUNCIONAMENTO

Empezamos añadiendo tres librerías:

```
#include "Audio.h"
#include "SD.h"
#include "FS.h"
```

A continuación definimos los diferentes puertos que usaremos para la recepción y transmisión de datos. Los pines `SPI_MOSI` , `SPI_MISO` , `SPI_SCK` y `SPI_CS` són los de la tarjeta SD y los pines `I2S_DOUT` , `I2S_BCLK` y `I2S_LRC` los vamos a conectar al MAX98357A.

```
#define SD_CS          5
#define SPI_MOSI      23
#define SPI_MISO      19
#define SPI_SCK       18
#define I2S_DOUT      25
#define I2S_BCLK      27
#define I2S_LRC       26
```

Ahora antes del `void setup()` , crearemos un objeto de la clase *Audio* que vamos a usar para enviar la configuración de pines, de volumen, etc. Ya dentro del `void setup()` en la primera línea inicializa una comunicación en serie a una velocidad de 115200 bauds, después con la función `pinMode()` establecemos *SD_CS* como salida y con `digitalWrite()` le asignamos el valor **HIGH**. A continuación los pines *SPI_SCK*, *SPI_MISO* y *SPI_MOSI* los configuramos para la recepción de datos desde la SD con la función `SPI.begin()` . En la siguiente línea con `SD.begin()` inicializa la biblioteca y la tarjeta SD para el pni *SD_CS*. Ahora seleccionamos los pines de salida del I2S, y en la siguiente línea

configuramos el volumen con `audio.setVolume(15)` , en este caso ponemos el valor de 15 este puede variar entre 0 y 21. Y finalmente tenemos que saber que fichero leer de la SD, por eso usamos la función `audio.connecttoFS()` , a esta le pasamos como parametros el objeto SD y el nombre del fichero que vamos a reproducir

```
void setup(){

    Serial.begin(115200);

    pinMode(SD_CS, OUTPUT);
    digitalWrite(SD_CS, HIGH);
    SPI.begin(SPI_SCK, SPI_MISO, SPI_MOSI);
    SD.begin(SD_CS);
    audio.setPinout(I2S_BCLK, I2S_LRC, I2S_DOUT);
    audio.setVolume(15);
    audio.connecttoFS(SD, "Ensoniq-ZR-76-01-Dope-77.wav");
}
```

Para finalizar hacemos el `void loop()` en el cual usaremos la función `audio.loop()` de la librería *Audio* que nos va a hacer un *loop* del fichero de audio.

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