🔔 ESP32 Alarm System - WAV Version

✨ Tentang Versi WAV

Versi WAV menggunakan format audio **lossless** dengan kualitas terbaik, cocok untuk:

- ✅ Sistem yang membutuhkan kualitas audio maksimal
- ✅ SD Card dengan kapasitas besar (8GB+)
- ✅ Audio alarm yang jernih tanpa kompresi
- ✅ Processing CPU minimal (WAV tidak perlu decode kompleks)

📋 Spesifikasi Format WAV

Rekomendasi Setting WAV

Parameter	Recommended	Keterangan
Format	PCM (Uncompressed)	Standard WAV
Sample Rate	16000 Hz / 22050 Hz	Cukup untuk suara alarm
Bit Depth	16-bit	Balance quality & size
Channels	Mono	Hemat storage 50%
Duration	3-10 detik	Optimal untuk alarm
File Size	500KB - 2MB	Tergantung durasi

Perbandingan Sample Rate

Sample Rate	Quality	File Size (5s)	Rekomendasi
8000 Hz	Phone quality	~80 KB	❌ Terlalu rendah
16000 Hz	Voice quality	~160 KB	✅ **Recommended**
22050 Hz	FM Radio	~220 KB	✅ Good
44100 Hz	CD Quality	~440 KB	⚠️ Overkill
48000 Hz	Studio	~480 KB	❌ Terlalu besar

💡 **Tip:** Gunakan **16000 Hz Mono 16-bit** untuk alarm - hemat storage tapi tetap jernih!

🗂️ Struktur File System

SD Card / LittleFS Structure

```
/
├── index.html        (Web interface)
├── wifi.html         (WiFi setup page - optional)
└── audio/
    ├── alarm.wav      (Default alarm - 16kHz mono)
    ├── warning.wav    (Default warning - 16kHz mono)
    ├── alarm1.wav     (Lapangan 1 alarm)
    ├── warning1.wav   (Lapangan 1 warning)
    ├── alarm2.wav     (Lapangan 2 alarm)
    ├── warning2.wav   (Lapangan 2 warning)
    ├── alarm3.wav     (Lapangan 3 alarm)
    ├── warning3.wav   (Lapangan 3 warning)
    ├── alarm4.wav     (Lapangan 4 alarm)
    └── warning4.wav   (Lapangan 4 warning)
```

🎵 Membuat File WAV yang Optimal

Menggunakan Audacity (Recommended)

1. Buka/Record Audio

- File → Open (untuk file existing)
- Atau Transport → Record untuk merekam baru

2. Optimasi Audio

```
Effect → Normalize
  - Normalize peak amplitude to: -3.0 dB
  - ✅ Remove DC offset

Effect → Compressor (untuk konsistensi volume)
  - Threshold: -20 dB
  - Ratio: 3:1

Effect → Fade Out (0.5s di akhir untuk smooth ending)
```

3. Konversi ke Mono (Hemat 50% size!)

```
Tracks → Mix → Mix Stereo Down to Mono
```

4. Resample (Kurangi sample rate)

```
Tracks → Resample
  - New sample rate: 16000 Hz ✅ Recommended
```

5. Export WAV

```
File → Export → Export as WAV

Settings:
- Format: WAV (Microsoft)
- Encoding: Signed 16-bit PCM
- Sample Rate: 16000 Hz
- Channels: 1 (Mono)
```

💾 **Hasil:** File 5 detik = ~160KB (vs ~440KB jika 44.1kHz stereo)

Menggunakan FFmpeg (Command Line)

Install FFmpeg

```
bash

# Windows
# Download dari https://ffmpeg.org/download.html

# Linux
sudo apt install ffmpeg

# Mac
brew install ffmpeg
```

Konversi MP3/Audio lain ke WAV Optimized

```
bash
```

```
# Basic conversion (16kHz mono)
ffmpeg -i input.mp3 -ar 16000 -ac 1 -sample_fmt s16 output.wav

# Dengan normalisasi audio
ffmpeg -i input.mp3 -ar 16000 -ac 1 -af "loudnorm" output.wav

# Trim + Convert (potong 5 detik pertama)
ffmpeg -i input.mp3 -ss 0 -t 5 -ar 16000 -ac 1 output.wav
```

Batch Convert Multiple Files

```bash
# Windows (PowerShell)
Get-ChildItem *.mp3 | ForEach-Object {
    ffmpeg -i $_.FullName -ar 16000 -ac 1 "$($_.BaseName).wav"
}

# Linux/Mac (Bash)
for f in *.mp3; do
    ffmpeg -i "$f" -ar 16000 -ac 1 "${f%.mp3}.wav"
done
```

Cek Info File WAV

```bash
ffmpeg -i alarm.wav
# Output akan menampilkan: sample rate, channels, duration, size
```

📦 Library yang Dibutuhkan

Arduino Libraries

```
ESP32 (by Espressif) - v2.0.0+
WiFi (Built-in)
WebServer (Built-in)
Preferences (Built-in)
ArduinoJson by Benoit Blanchon - v6.x
LittleFS (Built-in ESP32)
SD (Built-in)
```

```
SPI (Built-in)
ESPmDNS (Built-in)
ESP8266Audio by Earle Philhower - Latest
```

Install ESP8266Audio Library

1. Arduino IDE → **Tools** → **Manage Libraries**

2. Search: **"ESP8266Audio"**

3. Install versi terbaru (support WAV by default)

🚀 Upload File ke ESP32

Metode 1: Via Web Interface ⭐ Recommended

```
1. Akses: http://system-alarm.local atau http://192.168.1.100
2. Buka tab "Control Alarm"
3. Klik "⚙️ Pengaturan" pada lapangan
4. Browse file WAV (max 10MB)
5. Klik upload
6. Progress bar akan muncul
7. Tunggu "Upload successful"
```

Metode 2: Via Arduino IDE (LittleFS Upload)

```
1. Install plugin: ESP32 Sketch Data Upload
   - Download dari: https://github.com/me-no-dev/arduino-esp32fs-plugin
   - Extract ke: Arduino/tools/ESP32FS/tool/

2. Buat struktur folder:
   YourSketch/
   ├── ESP32_Alarm_System.ino
   └── data/
       ├── index.html
       └── audio/
           ├── alarm.wav
           ├── warning.wav
           └── ...

3. Arduino IDE → Tools → ESP32 Sketch Data Upload
4. Tunggu upload selesai (~2-5 menit)
```

Metode 3: Via SD Card (Paling Mudah)

```
1. Format SD Card → FAT32
2. Buat folder: audio/
3. Copy semua file WAV ke: SD_CARD/audio/
4. Eject safely
5. Masukkan SD Card ke ESP32
6. Pastikan code: config.useSDCard = true
7. Restart ESP32
```

🎛️ Konfigurasi Hardware

Pin Configuration

```cpp
// LED Output
LED 1: GPIO 14
LED 2: GPIO 13
LED 3: GPIO 2
LED 4: GPIO 15

// SD Card (SPI)
CS:   GPIO 5
MOSI: GPIO 23
MISO: GPIO 19
SCK:  GPIO 18

// MAX98357 I2S Audio
BCLK: GPIO 25
LRC:  GPIO 27
DOUT: GPIO 26
```

Wiring MAX98357A

```
MAX98357A   →   ESP32
─────────────────────────
VIN        →   5V
GND        →   GND
DIN        →   GPIO 26
BCLK       →   GPIO 25
```

```
LRC       →  GPIO 27
SD       →  (NC or GND for normal mode)

Speaker OUT →  4-8Ω Speaker (3-5W)
```

Wiring SD Card Module

```
SD Module  →  ESP32
────────────────────────
VCC       →  3.3V (PENTING: jangan 5V!)
GND       →  GND
CS       →  GPIO 5
MOSI      →  GPIO 23
MISO      →  GPIO 19
SCK       →  GPIO 18
```

⚙️ Upload ke ESP32

1. Persiapan

- Install Arduino IDE
- Install ESP32 board support
- Install libraries yang diperlukan
- Sambungkan ESP32 via USB

2. Upload Code

```
1. Buka: ESP32_Alarm_System.ino
2. Tools → Board → ESP32 Dev Module
3. Tools → Port → (pilih COM port ESP32)
4. Tools → Flash Size → 4MB (jika ada opsi)
5. Sketch → Upload
6. Tunggu "Done uploading"
```

3. Upload File System (Pilih salah satu)

Option A: LittleFS (Internal)

- Kapasitas: ~1.5MB

- Cocok untuk: File WAV optimized (16kHz)
- Upload via: ESP32 Sketch Data Upload

Option B: SD Card (External)

- Kapasitas: 2GB - 32GB
- Cocok untuk: Banyak file WAV atau high quality
- Upload via: Copy files ke SD Card

4. Upload HTML

- Copy `index.html` ke folder `/data/` atau SD Card root
- Upload via LittleFS atau SD Card

🔍 Testing & Troubleshooting

Checklist Testing

✅ Hardware Test

```
[ ] LED menyala saat alarm trigger
[ ] Speaker mengeluarkan suara
[ ] SD Card terdeteksi (cek Serial Monitor)
[ ] WiFi connect (LED board berkedip)
```

✅ Audio Test

```
[ ] Upload file WAV berhasil
[ ] File muncul di Serial Monitor
[ ] Audio play saat timer habis
[ ] Audio play saat warning
[ ] Volume control berfungsi
[ ] Loop alarm berfungsi (jika enabled)
```

✅ Web Interface Test

```
[ ] Bisa akses via browser
[ ] Timer start/pause/stop
[ ] Upload file via web
[ ] Settings tersimpan
[ ] WiFi setup berfungsi
```

Common Issues & Solutions

1. Audio Tidak Keluar

```
Serial Monitor shows: "Audio file not found"

✅ Solusi:
- Cek nama file: alarm1.wav (bukan Alarm1.wav atau alarm1.WAV)
- Cek path: harus di /audio/alarm1.wav
- Cek SD Card mounted: "SD Card mounted successfully"
- Cek file exists di SD/LittleFS
```

2. Audio Terpotong/Distorsi

```
Suara keluar tapi pecah atau tidak jelas

✅ Solusi:
- Kurangi sample rate: 44.1kHz → 16kHz
- Power supply kurang: gunakan 5V 2A
- Kabel speaker terlalu panjang: max 30cm
- Speaker impedance: gunakan 4-8Ω, bukan 2Ω
```

3. Upload WAV Gagal

```
"Upload failed" atau "File too large"

✅ Solusi:
- Cek ukuran: max 10MB
- Cek format: harus WAV (PCM)
- LittleFS penuh: gunakan SD Card
- Folder /audio/ belum ada: akan dibuat otomatis
```

4. ESP32 Restart Saat Play Audio

```
ESP32 reboot sendiri saat audio play

✅ Solusi:
- Power supply kurang: MIN 5V 2A
- Jangan pakai USB laptop (max 500mA)
- Tambah kapasitor 1000uF di VIN-GND
- Kurangi volume global
```

5. SD Card Tidak Terdeteksi

```
Serial Monitor: "SD Card Mount Failed"

✅ Solusi:
- Gunakan 3.3V bukan 5V (bisa rusak SD!)
- Cek wiring SPI
- Format SD: FAT32 (bukan exFAT/NTFS)
- Ukuran SD: max 32GB
- Coba SD Card lain (kadang tidak kompatibel)
```

📊 Perbandingan WAV vs MP3

Aspek	WAV ✅	MP3
Kualitas	Lossless (100%)	Lossy (~90-95%)
File Size (5s)	160KB (16kHz) / 440KB (44kHz)	50-80KB
CPU Usage	Very Low	Medium
Decode Speed	Instant	Slower
Compatibility	Universal	Need decoder
Best For	Quality priority	Storage priority

Kapan Menggunakan WAV?

✅ Gunakan WAV jika:

- Punya SD Card besar (8GB+)
- Butuh kualitas audio terbaik
- Ingin CPU load minimal
- Tidak masalah dengan file size besar

❌ **Jangan gunakan WAV jika:**

- Storage terbatas (LittleFS only)
- Ingin upload cepat via WiFi
- File audio banyak (>20 files)

💡 Tips & Tricks

1. Optimasi File Size WAV

```bash
# Dari 44.1kHz stereo ke 16kHz mono = 87% lebih kecil!
ffmpeg -i input.wav -ar 16000 -ac 1 output.wav

# Potong bagian diam di awal/akhir
ffmpeg -i input.wav -af "silenceremove=start_periods=1:start_silence=0.1:start_threshold=-50dB" output.wav
```

2. Batch Optimize WAV Files

```bash
# Optimize semua WAV di folder
for f in *.wav; do
    ffmpeg -i "$f" -ar 16000 -ac 1 -sample_fmt s16 "opt_$f"
done
```

3. Record Audio Langsung dari Windows

```
1. Buka: Voice Recorder (bawaan Windows)
2. Record alarm (3-10 detik)
3. Save as .m4a
4. Convert via Audacity:
   - Import M4A
   - Resample 16000 Hz
   - Mix Stereo to Mono
   - Export WAV 16-bit PCM
```

4. Test Audio via Serial Monitor

```cpp
```

```
// Tambahkan di playAudio() untuk debug
Serial.print("File size: ");
Serial.println(SD.open(audioPath).size());
Serial.print("Sample rate: ");
// Check dengan oscilloscope di I2S pins
```

🔐 Security & Optimization

Ganti Default Password

```
cpp

// Di ESP32_Alarm_System.ino
const char* ap_password = "12345678";  // ❌ GANTI INI!

// Ganti jadi password kuat:
const char* ap_password = "MyStr0ng_P@ss!";  // ✅
```

Optimasi Memory

```
cpp

// Jika crash atau restart:
// 1. Kurangi JsonDocument size
StaticJsonDocument<1024> doc;  // → 512 jika tidak perlu besar

// 2. Disable debug Serial
// Serial.println(...);  // Comment semua debug print

// 3. Gunakan SD Card daripada LittleFS
config.useSDCard = true;
```

📚 Resources

Audio Tools

- **Audacity:** https://www.audacityteam.org/
- **FFmpeg:** https://ffmpeg.org/
- **Online WAV Converter:** https://audio.online-convert.com/convert-to-wav

ESP32 Resources

- **ESP8266Audio Library:** https://github.com/earlephilhower/ESP8266Audio
- **ESP32 Audio Examples:** https://github.com/atomic14/esp32-i2s-examples
- **MAX98357A Datasheet:** https://www.maximintegrated.com/en/products/analog/audio/MAX98357A.html

Sound Effects (Free)

- **Freesound:** https://freesound.org/ (alarm sounds)
- **Zapsplat:** https://www.zapsplat.com/ (alarm, buzzer)
- **Notification Sounds:** https://notificationsounds.com/

🎉 Quick Start Guide

5 Langkah Mulai Cepat

1. **Upload Code ke ESP32**

```
Arduino IDE → Upload ESP32_Alarm_System.ino
```

2. **Siapkan File Audio WAV**

```
- Buat: alarm.wav (3-5 detik, 16kHz mono)
- Buat: warning.wav (3-5 detik, 16kHz mono)
```

3. **Upload ke SD Card**

```
SD_CARD/
└── audio/
    ├── alarm.wav
    └── warning.wav
```

4. **Connect ke WiFi ESP32**

```
SSID: Alarm_Timer_Setup
Password: 12345678
URL: http://192.168.4.1
```

5. **Test Alarm!**

```
- Set timer 1 menit
- Set warning 0.5 menit
- Start timer
- Tunggu alarm bunyi!
```

✅ Checklist Sebelum Deploy

Pre-deployment

- □ Code ter-upload tanpa error
- □ Serial Monitor menunjukkan "Audio system ready"
- □ SD Card terdeteksi (atau LittleFS ready)
- □ File alarm.wav & warning.wav exist
- □ Speaker connect ke MAX98357
- □ Power supply 5V 2A (bukan USB)

Testing

- □ Upload WAV via web berhasil
- □ Timer countdown bekerja
- □ LED menyala saat alarm
- □ Audio play dengan jelas
- □ Volume control berfungsi
- □ Pause/Resume/Stop bekerja

Production

- □ Ganti AP password
- □ Connect ke WiFi lokal
- □ Test access via mDNS (.local)
- □ Test dari device lain (HP/laptop)
- □ Label semua kabel
- □ Pasang di enclosure

🆘 Support

Jika masih ada masalah:

1. Cek Serial Monitor untuk error message
2. Test hardware satu-satu (LED → Audio → SD)
3. Gunakan audio sample pendek (3 detik) untuk test
4. Coba WAV file dari sumber lain (sample online)

Happy Building! 🎉