

Radio-réveil ESP32-S3

Radar de trafic aérien • Webradio • Horloge • Météo

Guide de construction

Matériel, câblage et bibliothèques Arduino

1. Liste du matériel

Composant	Référence / modèle	Rôle
Carte principale	ESP32-S3-DevKitC-1, N16R8 (16 Mo flash, 8 Mo PSRAM)	Carte principale
Écran	GC9A01, rond, 1.28" (240x240), interface SPI	Affichage horloge/radar/radio/météo
Amplificateur audio	MAX98357A, ampli I2S mono, classe D	Sortie audio de la webradio
Haut-parleur	Petit haut-parleur, 3-4W, 4 ou 8 ohms	Restitution sonore
Capteur de luminosité	BH1750, I2C	Détection du mode nuit (horloge)
Encodeur rotatif	EC11, avec bouton poussoir intégré	Réglages contextuels selon le mode
Boutons	3x mini-switch à LED intégrée	MODE / SET-ALARME / SNOOZE-MUTE
Résistances	3x 220-330 Ω ;	Limitation de courant des LED des boutons
Condensateur	1x électrolytique 470 μ F / 16-25V	Découplage alimentation de l'ampli
Condensateur	1x céramique 100 nF	Filtrage haute fréquence (bruit ampli classe D)
Alimentation	Chargeur USB-C, 5V / 3A minimum	Alimentation générale (voir section dédiée)

il n'est pas obligatoire d'utiliser des boutons avec led, celui de l'alarme est sympa avec, il clignote quand la radio s'allume avec l'alarme

2. Câblage

Toutes les broches indiquées sont celles de l'ESP32-S3 (numérotation GPIO, pas les numéros de broche physique du connecteur).

Écran GC9A01 (bus SPI, 3.3V)

GC9A01	ESP32-S3
VCC	3.3V
GND	GND
SCL	GPIO12
SDA	GPIO11
RST	GPIO8
DC	GPIO9
CS	GPIO10

Amplificateur MAX98357A (bus I2S, 5V)

MAX98357A	ESP32-S3
Vin	5V
GND	GND
BCLK	GPIO4
LRC	GPIO5
DIN	GPIO6
SD	GPIO7 (mute logiciel)
GAIN	GND (fixe le gain à 15 dB)

Ajouter le condensateur 470 μ F + le 100 nF entre Vin et GND, au plus près des pattes du module -- indispensable pour éviter les parasites de commutation de l'ampli classe D sur le reste du circuit (voir section 5).

Capteur BH1750 (bus I2C, 3.3V)

BH1750	ESP32-S3
VCC	3.3V
GND	GND
SCL	GPIO2
SDA	GPIO1
ADDR	GND (adresse I2C fixe 0x23)

Encodeur rotatif EC11 (3.3V)

EC11	ESP32-S3
------	----------

GND	GND
S1	GPIO14
S2	GPIO13
KEY	GPIO17
5V (silkscreen)	3.3V -- voir note ci-dessous

Le module EC11 est souvent sérigraphié "5V", mais il fonctionne très bien en 3.3V -- à privilégier pour rester au même niveau logique que les GPIO de l'ESP32. Si le sens de rotation perçu est inversé à l'usage, échanger simplement les valeurs de ENC_S1 et ENC_S2 dans le code (pas besoin de ressouder).

Les 3 boutons à LED

Bouton	Contact	LED (+ résistance)
MODE	GPIO38	GPIO39
SET / ALARME	GPIO40	GPIO41
SNOOZE / MUTE	GPIO18	GPIO21

Chaque contact utilise la résistance de tirage interne de l'ESP32 (pull-up), pas besoin de résistance externe sur le contact -- seulement sur la LED. GPIO40/41 sont physiquement adjacentes sur le connecteur : vérifier l'absence de pont de soudure entre les deux si un bouton se comporte de façon incohérente.

Broches à éviter absolument

- **GPIO0** : bouton BOOT déjà présent sur la carte.
- **GPIO19 / GPIO20** : réservées à l'USB natif (D-/D+).
- **GPIO33 à GPIO37** : réservées en interne à la PSRAM octale sur ce module N16R8, inutilisables même si présentes sur le connecteur.
- **GPIO45 / GPIO46** : broches de strapping (niveau au boot), à éviter en sortie.

3. Alimentation

■ Utiliser un chargeur USB-C **5V / 3A minimum**. L'ampli et le WiFi créent des pics de consommation ponctuels qui peuvent dépasser 1,5A -- un simple port USB d'ordinateur (limité à ~500mA) provoque des redémarrages en boucle.

Un chargeur avec Quick Charge (QC 2.0/3.0) convient d'office. Un chargeur USB-C "PD pur" fonctionne dans l'immense majorité des cas grâce au repli 5V standard, mais en cas de redémarrages instables au premier essai, tester un chargeur 5V/2-3A classique en priorité.

Pour déboguer via le moniteur série tout en restant sur l'alimentation externe (nécessaire pour un fonctionnement stable), utiliser un câble USB-C "data only" côté PC, ou un câble Y dédié au débogage -- brancher les deux alimentations (PC + externe) simultanément sur les deux ports USB-C de la carte n'est pas recommandé, le circuit de sélection d'alimentation n'étant pas garanti sur tous les exemplaires de cette carte.

4. Configuration de la carte (menu Outils)

Réglage	Valeur
Carte	ESP32S3 Dev Module
USB CDC On Boot	Enabled
CPU Frequency	240MHz (WiFi/BT)
Flash Mode	QIO 80MHz
Flash Size	16MB (128Mb)
Partition Scheme	16M Flash (3MB APP/9.9MB FATFS) ou équivalent 16MB
PSRAM	OPI PSRAM
Upload Speed	921600
USB Mode	Hardware CDC and JTAG

■ Le réglage le plus souvent oublié : **PSRAM doit être sur "OPI PSRAM"**, pas "QSPI PSRAM" ni "Disabled". Le module N16R8 a une PSRAM octale ; un mauvais réglage ici provoque une erreur de lecture de la PSRAM au démarrage, ou un plantage silencieux.

Ces réglages peuvent revenir à leur valeur par défaut après une mise à jour du core ESP32 ou un changement de carte dans l'IDE -- les vérifier si un comportement déjà résolu (reboot, port série muet) réapparaît sans raison apparente.

5. Bibliothèques à installer

Toutes s'installent depuis le **Gestionnaire de bibliothèques** d'Arduino IDE (menu Croquis → Inclure une bibliothèque → Gérer les bibliothèques), en cherchant le nom indiqué.

Bibliothèque	Auteur	Usage
GFX Library for Arduino	moononournation	Pilotage de l'écran GC9A01
ESP32-audioI2S	schreibfaul1	Décodage et streaming des flux webradio
BH1750	claws	Lecture du capteur de luminosité
ArduinoJson	bblanchon	Analyse des réponses JSON (radar ADS-B, météo)
WiFiManager	tzapu	Portail web de configuration (WiFi, position, stations)

Déjà incluses avec le core ESP32 (rien à installer)

- `WiFi.h`, `HTTPClient.h` -- réseau et requêtes web
- `Wire.h` -- bus I2C
- `Preferences.h` -- mémoire non-volatile (NVS)
- `time.h` -- horloge et synchronisation NTP

Core ESP32 (Arduino-ESP32)

Installer via le gestionnaire de cartes (Fichier → Préférences → URL de gestionnaire de cartes supplémentaire, puis Outils → Type de carte → Gestionnaire de cartes → rechercher "esp32"). **Version 3.x recommandée** -- l'API utilisée pour piloter les LED (`ledcAttach`) est celle de cette génération du core.