# ESP32-S3-Touch-LCD-4.3

## Panoramica

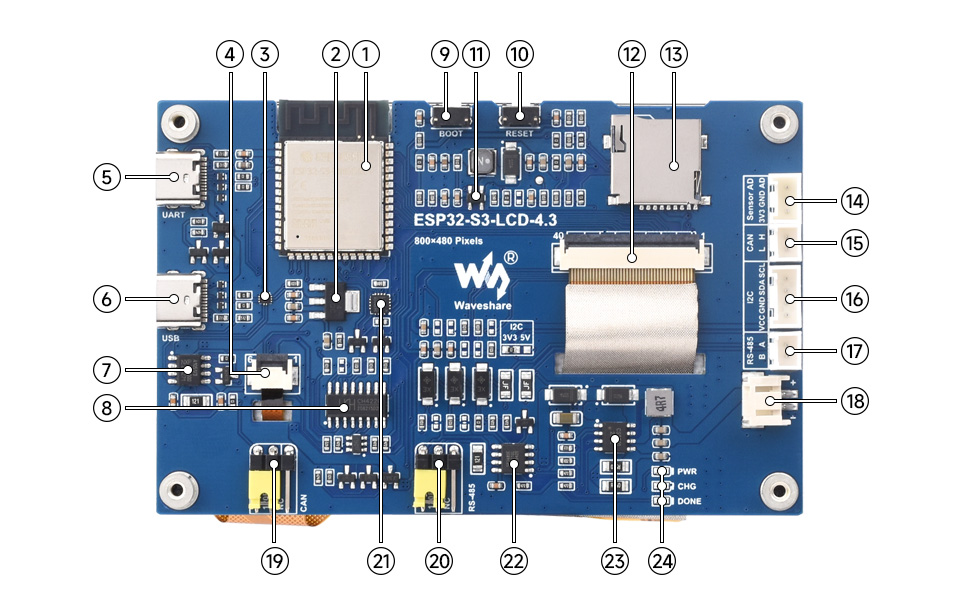
### Introduzione

Questa è una scheda di sviluppo MCU a basso costo e ad alte prestazioni progettata da Waveshare. Supporta WiFi a 2,4 GHz e BLE 5, integra Flash e PSRAM ad alta capacità e dispone di un ampio display LCD touch capacitivo da 4,3 pollici per eseguire senza problemi programmi con interfaccia grafica come LVGL. Combina una varietà di interfacce periferiche (ad esempio CAN, I2C, RS485, ecc.) per sviluppare rapidamente applicazioni come HMI per ESP32-S3. Grazie a un'ampia gamma di funzioni e interfacce, è in grado di soddisfare i requisiti di consumo energetico di applicazioni quali Internet of Things (IoT), dispositivi mobili, smart home e altre applicazioni.

### Funzionalità

* Dotato di processore dual-core Xtensa LX7 a 32 bit ad alte prestazioni, con frequenza principale fino a 240 MHz
* Supporta Wi-Fi a 2,4 GHz (802.11 b/g/n) e Bluetooth 5 (BLE), con antenna integrata
* 512 KB di SRAM e 384 KB di ROM integrate, con 16 MB di Flash e 8 MB di PSRAM integrati
* Display LCD integrato da 4,3 pollici, risoluzione 800x480, 65.000 colori
* Supporta il controllo touch capacitivo tramite interfaccia I2C (opzionale), con touch a 5 punti e supporto per interrupt
* Interfaccia CAN, RS485, I2C e slot per schede TF integrati, integra USB full-speed
* Supporta clock flessibile, impostazione indipendente dell'alimentazione del modulo e altri controlli per ottenere bassi consumi energetici in diversi scenari

### Risorse Integrate

[](https://www.waveshare.com/wiki/File:ESP32-S3-Touch-LCD-4.3-details-intro.jpg)

**1. ESP32-S3-WROOM-1-N16R8**   
Modulo SoC WiFi Bluetooth con frequenza operativa di 240 MHz   
Package 8MB PSRAM e 16MB Flash

**2. SGM2212-3.3**   
LDO a basso rumore da 800 mA

**3. FSUSB42UMX**   
Per il multiplexing dei pin USB

**4. Connettore per pannello touch**   
Per i dispositivi non touch, non è collegato

**5. Porta USB-UART Type-C**

**6. Interfaccia USB Type-C**

**7. TJA1051T/3/1J**   
Chip di interfaccia CAN

**8. CH422G**   
Chip di espansione IO

**9. Pulsante BOOT**   
Tenere premuto per accendere il programma e farlo lampeggiare

**10. Pulsante RESET**   
Premere per riavviare il controller

**11. MP3302DJ-LF-Z**   
Chip di boost della retroilluminazione dello schermo **12. Connettore del pannello display da 4,3 pollici**

**13. Slot per scheda TF**

**14. Terminale a vite del sensore**

**15. Interfaccia CAN**

**16. Terminale I2C**

**17. Terminale RS485**

**18. Interfaccia PH2.0 per batteria al litio singola da 3,7 V**

**19. Interfaccia di selezione della resistenza terminale CAN**

**20. Interfaccia di selezione della resistenza terminale RS485**

**21. CH343P**   
chip da USB a UART

**22. SP3485**   
Chip transceiver RS485

**23. CS8501**   
Chip di gestione della carica della batteria al litio

**24. Indicatore di stato**   
PWR: Indicatore di alimentazione   
CHG: Indicatore di carica della batteria al litio   
DONE: Indicatore di carica della batteria al litio completata

## Pinout

* **LCD interface**: Interfaccia per collegare il cavo LCD

|  |  |  |
| --- | --- | --- |
| **ESP32-S3** | **LCD** | **Descrizione** |
| GPIO0 | G3 | Verde dati 3° posto |
| GPIO1 | R3 | Rosso dati 3° posto |
| GPIO2 | R4 | Rosso dati 4° posto |
| GPIO3 | VSYNC | Segnale di sincronizzazione verticale |
| GPIO5 | DE | Segnale di abilitazione dati |
| GPIO7 | PCLK | Segnale di clock |
| GPIO10 | B7 | Blu 7° posto |
| GPIO14 | B3 | Blu 3° posto |
| GPIO17 | B6 | Blue 6° posto |
| GPIO18 | B5 | Blu 5° posto |
| GPIO21 | G7 | Verde 7° posto |
| GPIO38 | B4 | Blu 4° posto |
| GPIO39 | G2 | Verde 2° posto |
| GPIO40 | R7 | Rosso 7° posto |
| GPIO41 | R6 | Rosso 6° posto |
| GPIO42 | R5 | Rosso 5° posto |
| GPIO45 | G4 | Verde 4° posto |
| GPIO46 | HSYNC | Segnale di sincronizzazione orizzontale |
| GPIO47 | G6 | Verde 6° posto |
| GPIO48 | G5 | Verde 5° posto |
| CH422G | LCD | - |
| EXIO2 | DISP | Pin di abilitazione retroilluminazione |

* **Touch screen interface**: L'interfaccia utilizzata per collegare il cavo touch

|  |  |  |
| --- | --- | --- |
| **ESP32-S3** | **Touch** | **Descrizione** |
| GPIO4 | TP\_IRQ | Pin di interrupt touch |
| GPIO8 | TP\_SDA | Pin dati touch |
| GPIO9 | TP\_SCL | Pin di clock touch |
| CH422G | Touch | - |
| EXIO1 | TP\_RST | Pin di reset touch |

* **USB interface**: Utilizzata per l'alimentazione e il flashing

|  |  |  |
| --- | --- | --- |
| **ESP32-S3** | **USB** | **Descrizione** |
| GPIO19 | USB\_DN | Cavo dati D- |
| GPIO20 | USB\_DP | Cavo dati D+ |
| CH422G | USB | - |
| EXIO5 | USB\_SEL | Pull down in modalità USB, altrimenti in modalità CAN |

* **TF card interface**: Interfaccia utilizzata per collegare la scheda TF. I collegamenti dei pin sono mostrati nella tabella seguente.

|  |  |  |
| --- | --- | --- |
| **ESP32-S3** | **TF** | **Descrizione** |
| GPIO11 | MOSI | Pin di ingresso della scheda TF |
| GPIO12 | SCK | Pin di clock della scheda TF |
| GPIO13 | MISO | Pin di uscita della scheda TF |
| CH422G | TF | - |
| EXIO4 | SD\_CS | Pin di abilitazione della scheda TF, attivo basso |

* **RS485 interface**: La scheda di sviluppo è dotata di un'interfaccia RS485, che consente la connessione diretta per la comunicazione con il dispositivo, con commutazione automatica delle modalità di trasmissione e ricezione del circuito.

|  |  |  |
| --- | --- | --- |
| **ESP32-S3** | **RS485** | **Descrizione** |
| GPIO16 | RS485\_RXD | Ingresso dati |
| GPIO15 | RS485\_TXD | Uscita dati |

* **CAN interface**: Implementa il controllo di trasmissione e ricezione, l'analisi dei dati, l'acquisizione e il monitoraggio della rete bus CAN.

|  |  |  |
| --- | --- | --- |
| **ESP32-S3** | **CAN** | **Descrizione** |
| GPIO15 | CANTX | Uscita dati |
| GPIO16 | CANRX | Ingresso dati |
| CH422G | CAN | - |
| EXIO5 | CAN\_SEL | Pull-up in modalità CAN, altrimenti in modalità USB |

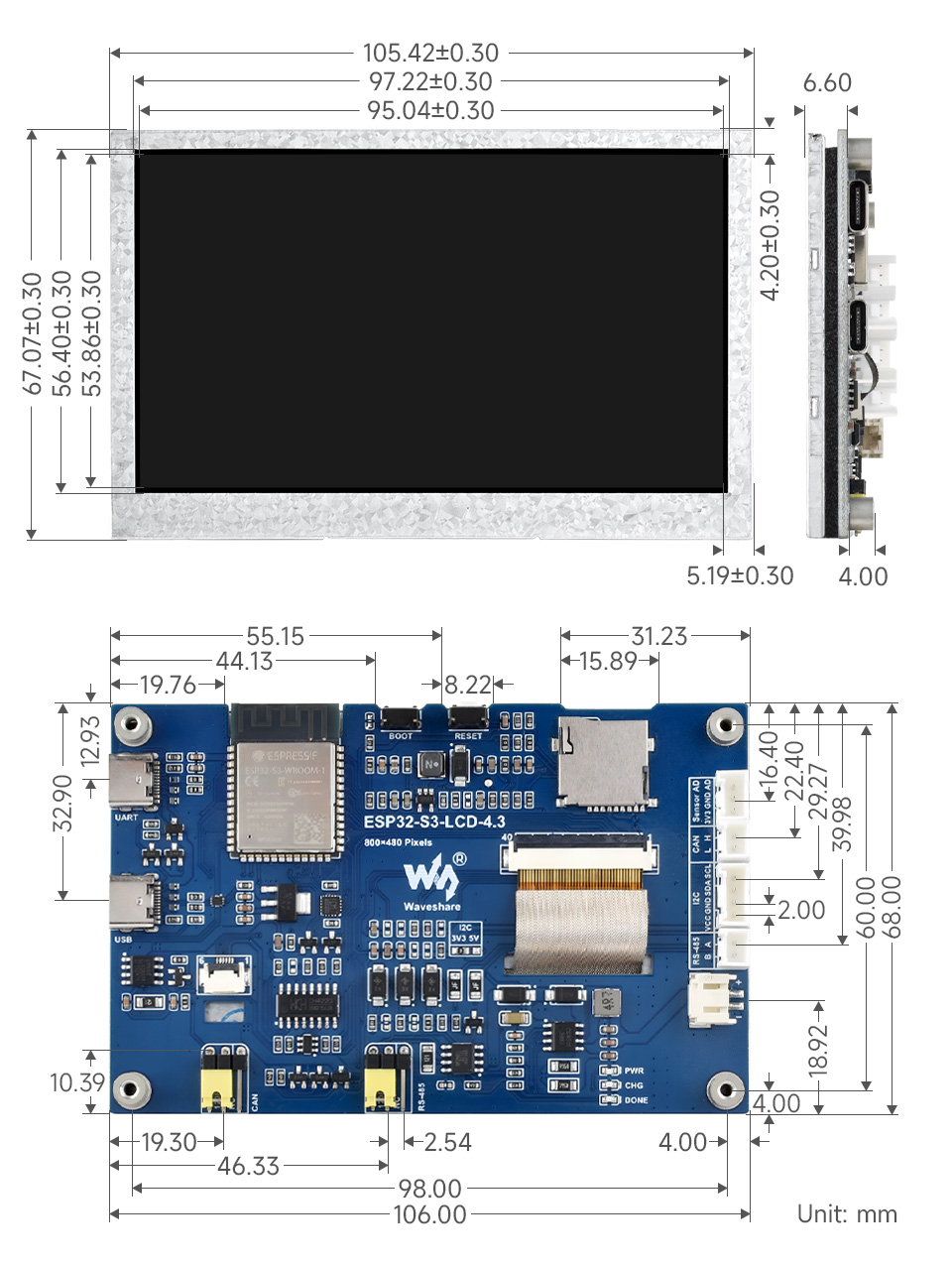
* **I2C interface**: ESP32-S3 fornisce hardware I2C multicanale, attualmente utilizzando pin GPIO8 (SDA) e GPIO9 (SCL) per il bus I2C.   
  È destinato al montaggio del chip di espansione IO, dell'interfaccia touch e dell'interfaccia esterna I2C.

|  |  |  |
| --- | --- | --- |
| **ESP32-S3** | **I2C** | **Descrizione** |
| GPIO8 | SDA | Pin dati I2C |
| GPIO9 | SCL | Pin clock I2C |

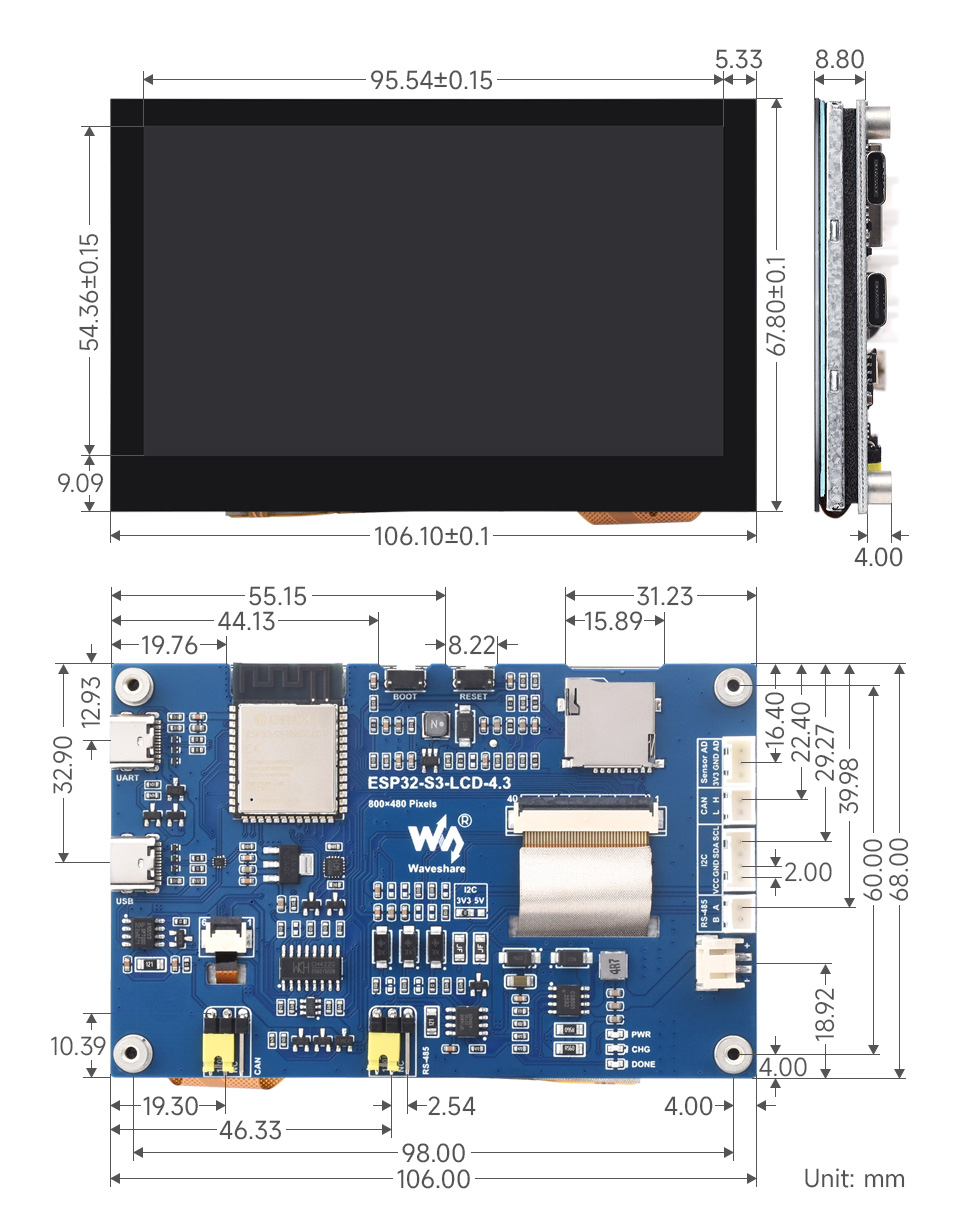
* **PH2.0 battery interface**: La scheda di sviluppo utilizza un chip di gestione carica/scarica ad alta efficienza CS8501, in grado di caricare una singola batteria al litio fino a 5 V. La corrente di carica attuale è di 580 mA e l'utente può modificare la corrente di carica sostituendo la resistenza R45. Per i dettagli, fare riferimento allo [schema ESP32-S3-Touch-LCD-4.3](https://files.waveshare.com/wiki/ESP32-S3-Touch-LCD-4.3/manual/ESP32-S3-Touch-LCD-4.3-Sch.pdf).

### Dimensioni

#### Versione Senza Touch

[](https://www.waveshare.com/wiki/File:ESP32-S3-LCD-4.3-details-size.jpg)

#### Versione Con Touch

[](https://www.waveshare.com/wiki/File:Esp32-s3-touch-lcd-4.3-003.jpg)

### Specifiche

|  |  |
| --- | --- |
| **Parametri di base** | |
| Processore | Processore dual-core Xtensa LX7 a 32 bit ad alte prestazioni con frequenza fino a 240 MHz |
| Wifi/Bluetooth | Supporta Wi-Fi a 2,4 GHz (802.11 b/g/n) e Bluetooth 5 (LE) con antenne integrate |
| Flash | 16MB Flash |
| PSRAM | 8MB PSRAM |
| Power supply range | TypeC 5V |
| **Parametri dello schermo** | |
| Risoluzione | 800 x 480 |
| Interfaccia display | RGB |
| Pannello display | IPS |
| Angolo di visione | 160° |
| Luminosità schermo | 270 Cd/m² |
| Tipo touch | Capacitivo |
| Pannello touch | Vetro temprato |
| **Interfaccia periferica** | |
| Interfaccia di comunicazione | CAN, RS485, I2C, USB |
| **Altro** | |
| Consumo energetico | 5V 450mA |
| Temperatura di esercizio | 0℃ ~ 65℃ |
| Dimensioni del prodotto (L×W) | Versione non-touch: 105.4×67.1mm  Versione touch: 106.1×67.8mm |

## Istruzioni per l'uso

Attualmente sono disponibili due strumenti e framework di sviluppo, **Arduino IDE** e **ESP-IDF**, che offrono opzioni di sviluppo flessibili. È possibile scegliere lo strumento di sviluppo più adatto alle esigenze del progetto e alle proprie abitudini.

### Strumenti di Sviluppo

|  |  |
| --- | --- |
|  | Arduino IDE Arduino IDE è una piattaforma di prototipazione elettronica open source, pratica e flessibile, facile da usare. Dopo un semplice apprendimento, è possibile iniziare a sviluppare rapidamente. Allo stesso tempo, Arduino vanta un'ampia comunità di utenti globale, che fornisce un'abbondanza di codice open source, esempi di progetti e tutorial, oltre a una ricca libreria di risorse che incapsula funzioni complesse, consentendo agli sviluppatori di implementare rapidamente diverse funzionalità. |
|  | ESP-IDF ESP-IDF, o nome completo Espressif IDE, è un framework di sviluppo professionale introdotto da Espressif Technology per i chip della serie ESP. È sviluppato in linguaggio C, includendo un compilatore, un debugger, uno strumento di flashing, ecc. e può essere sviluppato tramite riga di comando o tramite un ambiente di sviluppo integrato (come Visual Studio Code con il plugin Espressif IDF). Il plugin offre funzionalità come la navigazione del codice, la gestione dei progetti e il debug, ecc. |

Ognuno di questi due approcci di sviluppo ha i suoi vantaggi e gli sviluppatori possono scegliere in base alle proprie esigenze e al proprio livello di competenza. Arduino è adatto a principianti e non professionisti perché è facile da imparare e veloce da usare. ESP-IDF è una scelta migliore per gli sviluppatori con un background professionale o che richiedono elevate prestazioni, poiché fornisce strumenti di sviluppo più avanzati e maggiori capacità di controllo per lo sviluppo di progetti complessi.

### Preparazione dei Componenti

* ESP32-S3-Touch-LCD-4.3 x1
* Scheda TF x 1
* Cavo USB (da Tipo A a Tipo C) x 1
* Convertitore bidirezionale da USB a RS485 x 1
* Analizzatore adattatore da USB a CAN x 1

[](https://www.waveshare.com/wiki/File:ESP32-Arduino-47.png)

## Precauzioni

* La scheda di sviluppo è dotata di un circuito di download automatico integrato; la porta Tipo C sulla serigrafia UART viene utilizzata per il download del programma e la stampa del log. Dopo aver scaricato il programma, premere il pulsante RESET per eseguirlo.
* Prestare attenzione all'area dell'antenna sul PCB durante l'utilizzo ed evitare di fissarvi altri componenti metallici o in plastica.
* La scheda TF utilizza SPI/MMC per comunicare; si noti che il pin SD\_CS deve essere pilotato dall'EXIO4 del CH422G.
* La scheda di sviluppo utilizza un connettore PH2.0 per collegare i pin delle periferiche ADC, CAN, I2C e RS485 e collega i componenti del sensore tramite un connettore maschio da 2,54 mm.
* Le periferiche CAN e RS485 utilizzano jumper per collegare la resistenza da 120 ohm di default, mentre NC è opzionale per annullare la connessione della resistenza terminale.
* Lo schermo da 4,3 pollici occupa la maggior parte del GPIO e la scheda di sviluppo utilizza il chip [CH422G](https://files.waveshare.com/wiki/common/CH422DS1_EN.pdf) per espandere l'IO per il reset, lo spegnimento e l'accensione della retroilluminazione, ecc.
* **La scheda di sviluppo utilizza la porta USB per scaricare la demo. Se la porta non viene riconosciuta, accedere alla modalità di boot (tenere premuto il pulsante di boot, quindi connettersi al computer e rilasciare il pulsante di boot). Dopo aver scaricato la demo, premere il pulsante RESET per eseguirla.**
* Attualmente, con ESP-IDF v5.3, il limite medio di frame rate per l'esecuzione della demo del benchmark LVGL con un singolo core è di 26 frame al secondo, corrispondenti a un frame rate dell'interfaccia di 41 (PCLK 21 MHz). Prima della compilazione, ESP32 e LVGL devono essere configurati tramite menuconfig:

CONFIG\_FREERTOS\_HZ=1000

CONFIG\_ESP\_DEFAULT\_CPU\_FREQ\_MHZ\_240=y

CONFIG\_ESPTOOLPY\_FLASHMODE\_QIO=y

CONFIG\_ESPTOOLPY\_FLASHFREQ\_120M=y [Deve essere coerente con PSRAM]

CONFIG\_SPIRAM\_MODE\_OCT=y

CONFIG\_IDF\_EXPERIMENTAL\_FEATURES=y e CONFIG\_SPIRAM\_SPEED\_120M=y [Deve essere coerente con FLASH]

CONFIG\_SPIRAM\_FETCH\_INSTRUCTIONS=y

CONFIG\_SPIRAM\_RODATA=y

CONFIG\_ESP32S3\_DATA\_CACHE\_LINE\_64B=y

CONFIG\_COMPILER\_OPTIMIZATION\_PERF=y

#I seguenti elementi di configurazione LVGL sono utili per migliorare il frame rate (LVGL v8.3):

#define LV\_MEM\_CUSTOM 1 o CONFIG\_LV\_MEM\_CUSTOM=y

#define LV\_MEMCPY\_MEMSET\_STD 1 o CONFIG\_LV\_MEMCPY\_MEMSET\_STD=y

#define LV\_ATTRIBUTE\_FAST\_MEM IRAM\_ATTR o CONFIG\_LV\_ATTRIBUTE\_FAST\_MEM=y

* Per descrizioni dettagliate delle prestazioni di LCD e LVGL, fare riferimento alla [Documentazione](https://files.waveshare.com/wiki/common/Performance.pdf)
* Il supporto per batterie al litio PH1.25 supporta solo una singola batteria al litio da 3,7 V; non utilizzare più set di batterie da collegare per caricare e scaricare contemporaneamente; si consiglia che la capacità di una singola batteria sia inferiore a 2000 mAH.
* Il CH422G e il touch della scheda occupano il seguente indirizzo slave; non utilizzare dispositivi I2C con lo stesso indirizzo:

0 1 2 3 4 5 6 7 8 9 a b c d e f

00: - - - - - - - - - - - - - - - -

10: - - - - - - - - - - - - - - - -

20: 20 21 22 23 24 25 26 27 - - - - - - - -

30: 30 31 32 33 34 35 36 37 38 39 3a 3b 3c 3d 3e 3f

40: - - - - - - - - - - - - - - - -

50: - - - - - - - - - - - - - - 5d -

60: - - - - - - - - - - - - - - - -

70: - - - - - - - - - - - - - - - -

Prima di iniziare, si consiglia di consultare l'indice per comprendere rapidamente la struttura del documento. Per un funzionamento ottimale, si prega di leggere attentamente le [FAQ](https://www.waveshare.com/wiki/ESP32-S3-Touch-LCD-4.3#FAQ) per comprendere in anticipo eventuali problemi. Tutte le risorse presenti nel documento sono fornite di link per un facile download.

# Lavorare con Arduino

Questo capitolo introduce la configurazione dell'ambiente Arduino, incluso l'IDE Arduino, la gestione delle schede ESP32, l'installazione delle librerie correlate, la compilazione e il download dei programmi, nonché il test delle demo. L'obiettivo è aiutare gli utenti a padroneggiare la scheda di sviluppo e facilitare lo sviluppo secondario.

## Configurazione dell'Ambiente

### Download e Installazione di Arduino IDE

Cliccare per visitare il [sito web ufficiale di Arduino](https://www.arduino.cc/en/software), selezionare il sistema e i bit corrispondenti da scaricare La versione dell'IDE Arduino deve essere ≥ 1.8 e il path di installazione non deve essere in cinese, altrimenti si verificherà un errore durante la compilazione.   
[](https://www.waveshare.com/wiki/File:ESP32-S3-AMOLED-1.91-Ar-software-01.png)

* Eseguire il programma di installazione e installare tutto per default

La configurazione dell'ambiente viene eseguita sul sistema Windows 10; gli utenti Linux e Mac possono accedere a [Configurazione dell'ambiente Arduino-esp32](https://docs.espressif.com/projects/arduino-esp32/en/latest/installing.html) come riferimento.

### Installazione di Arduino-ESP32

* Per quanto riguarda le schede madri ESP32 utilizzate con l'IDE Arduino, è necessario installare prima la libreria **esp32 di Espressif Systems**.
* In genere, si consiglia di utilizzare l'**Installazione online**. Se l'installazione online non riesce, utilizzare l'**Installazione offline**.
* Per installare il tutorial Arduino-ESP32, fare riferimento al [tutorial Arduino Board Manager.](https://www.waveshare.com/wiki/Arduino_Board_Managers_Tutorial)
* La scheda di sviluppo ESP32-S3-Touch-LCD-4.3 viene fornita con un pacchetto offline. Cliccare qui per scaricare: [esp32\_package\_3.0.7\_arduino offline package](https://drive.google.com/drive/folders/1Pcs_A4FKWvdSHnz9lEBYqOpr-noTMbIv?usp=sharing)
* Istruzioni per l'installazione della scheda di sviluppo ESP32-S3-Touch-LCD-4.3

|  |  |  |
| --- | --- | --- |
| **Nome della Board** | **Requisiti per l'installazione della board** | **Numero di versione richiesto** |
| ESP32-S3-Touch-LCD-4.3 | "Install Offline" / "Install Online" | 3.0.0 e versioni successive |

### Installazione delle Librerie

* Quando si installano le librerie Arduino, di solito ci sono due modi tra cui scegliere: **Installazione online** e **Installazione offline**. **Se l'installazione della libreria richiede l'installazione offline, è necessario utilizzare il file di libreria fornito.**   
  Per la maggior parte delle librerie, gli utenti possono facilmente cercarle e installarle tramite il gestore di librerie online del software Arduino. Tuttavia, alcune librerie open source o personalizzate non sono sincronizzate con Arduino Library Manager, quindi non possono essere acquisite tramite ricerche online. In questo caso, gli utenti possono installare manualmente queste librerie solo offline.
* Per il tutorial sull'installazione delle librerie, fare riferimento al [Tutorial sul gestore librerie Arduino](https://www.waveshare.com/wiki/Arduino_Library_Manager_Tutorial)
* Il file della libreria ESP32-S3-Touch-LCD-4.3 è memorizzato nel programma di esempio, cliccare qui per passare a: [Demo di ESP32-S3-Touch-LCD-4.3](https://www.waveshare.com/wiki/ESP32-S3-Touch-LCD-4.3#Resources)
* Istruzioni per l'installazione del file della libreria ESP32-S3-Touch-LCD-4.3

|  |  |  |  |
| --- | --- | --- | --- |
| **Nome della Libreria** | **Descrizione** | **Versione** | **Requisiti per l'Installazione della Libreria** |
| ESP32\_Display\_Panel | Libreria di controllo del pannello di visualizzazione specifica del microcontrollore ESP32 | v0.1.4 e successive | "Installazione online" o "Installazione offline" |
| ESP32\_IO\_Expander | Libreria di espansione I/O di ESP32 | v0.0.4 o successive | "Installazione online" o "Installazione offline" |
| lvgl | Libreria grafica LVGL | v8.4.0 | "Installazione Offline" |
| lv\_conf.h | File di configurazione di LVGL | —— | "Installazione Offline" |

## Eseguire la Prima Demo di Arduino

Se si è appena iniziato a usare ESP32 e Arduino e non si sa come creare, compilare, flashare ed eseguire programmi Arduino ESP32, preghiamo di approfondire e dare un'occhiata. Si spera che possa essere utile!

#### Nuovo Progetto

Eseguire l'IDE Arduino e selezionare **File** -> **New Sketch**

* Immettere il codice:

void setup() {

// put your setup code here, to run once:

Serial.begin(115200);

}

void loop() {

// put your main code here, to run repeatedly:

Serial.println("Hello, World!");

delay(2000);

}

* Salvare il progetto e selezionare **File** -> **Save As...**. Nel menu pop-up, selezionare il path in cui salvare il progetto e inserire un nome per il progetto, ad esempio **Hello\_World**, cliccare su **Save**

### Compilazione e Flash dei Demo

* Selezionare la board di sviluppo corrispondente, prendendo come esempio la scheda madre ESP32S3:

①. Cliccare per selezionare l'opzione del menù a discesa **Select Other Board and Port**;   
②. Cercare il modello di board di sviluppo desiderato **esp32s3 dev module** e selezionare;   
③. Selezionare **COM Port**;   
④. **Salvare** la selezione.

* Se la scheda madre ESP32S3 ha solo una porta USB, è necessario abilitare **USB CDC**, come mostrato nel diagramma seguente:
* Compilare e caricare il programma:

①. Compilare il programma; ②. Compilare e scaricare il programma; ③. Download completato.

* Aprire la finestra **Serial Monitor** e la demo stamperà "Hello World!" ogni 2 secondi; il funzionamento è il seguente:

### Demo

|  |  |  |
| --- | --- | --- |
| * Demo di ESP32-S3-Touch-LCD-4.3 | | |
| **Demo** | **Descrizione di Base** | **Dipendenze da Librarie** |
| 01\_I2C\_Test | Test header I2C | - |
| 02\_RS485\_Test | Test header RS485 | - |
| 03\_SD\_Test | Test slot per scheda TF | - |
| 04\_Sensor\_AD | Test header ADC | - |
| 05\_UART\_Test | Test UART | - |
| 06\_TWAItransmit | Test header CAN | - |
| 07\_TWAIreceive | Test header CAN | - |
| 08\_DrawColorBar | Test schermo RGB | ESP32\_Display\_Panel |
| 09\_lvgl\_Porting | Test porting LVGL | LVGL, ESP32\_Display\_Panel |

* **Se il numero di versione dell'ESP32 è 3.0.6 o superiore, l'ESP32-S3-Touch-LCD-4.3 supporta il modello diretto selezione. Dopo aver selezionato direttamente il modello, alcuni parametri non devono essere modificati per default**
  + Prendiamo come esempio ESP32-S3-LCD-1.69

[](https://www.waveshare.com/wiki/File:ESP32-S3-LCD-1.69-demo-02.png)

#### 01\_I2C\_Test

##### Collegamento hardware

* Collegare la board al computer tramite un cavo USB.

##### Analisi del codice

* **loop()**:

loop()La funzione è la parte principale del programma, ovvero il loop, e la sua funzione principale è la scansione dei dispositivi sul bus I2C.   
Innanzitutto, sono state definite delle variabili per memorizzare i codici di errore, gli indirizzi dei dispositivi e per registrare il numero di dispositivi rilevati.   
Quindi, si itera attraverso i possibili indirizzi dei dispositivi I2C dalla posizione 0x01 alla 0x7f utilizzando un loop. Per ogni indirizzo, utilizza Wire.beginTransmission(address) per avviare la trasmissione al dispositivo all'indirizzo specificato, quindi utilizza Wire.endTransmission() per terminare la trasmissione e ottenere il codice di errore.   
Se il codice di errore è 0, è stato trovato un dispositivo I2C a quell'indirizzo, ne visualizza l'indirizzo e incrementa il conteggio dei dispositivi. Se il codice di errore non è 2 (indicando che il dispositivo non risponde), vengono visualizzati il ​​codice di errore e l'indirizzo corrispondente.   
Infine, se non vengono trovati dispositivi I2C, visualizza il messaggio appropriato e utilizza delay(5000) per mettere in pausa il programma per 5 secondi e ripetere la scansione.

##### Demo flashing

* Selezionare "ESP32S3 Dev Module" e la porta
* Impostare i parametri della scheda di sviluppo

[](https://www.waveshare.com/wiki/File:ESP32-Arduino-41.png)

* Flashare la demo

##### Dimostrazione dei risultati

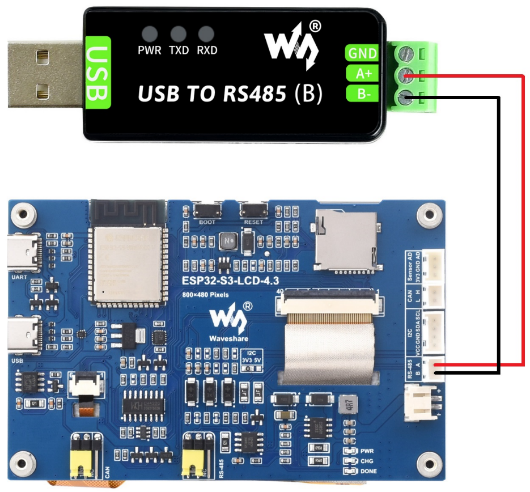
* Il monitor seriale visualizza l'indirizzo del dispositivo sul bus I2C

[](https://www.waveshare.com/wiki/File:ESP32-Arduino-6.png)

#### 02\_RS485\_Test

##### Collegamento hardware

* Collegare la board al computer tramite un cavo USB.
* Collegare la scheda di sviluppo al [convertitore USB-RS485](https://www.waveshare.com/USB-TO-RS485-B.htm), come mostrato in figura

[](https://www.waveshare.com/wiki/File:ESP32-Arduino-43.png)

##### Analisi del codice

* **setup()**:

setup viene utilizzata principalmente per inizializzare la comunicazione seriale.   
Utilizzare la funzione RS485.begin per inizializzare la porta seriale Serial1, impostare la velocità in baud, il formato dei dati e specificare i pin di ricezione e trasmissione. Quindi, tramite un ciclo, assicurarsi che l'inizializzazione della porta seriale abbia esito positivo.

* **loop()** :

loop è la parte principale del programma, e la sua funzione principale è implementare un semplice ritorno di dati di comunicazione 485.   
Verificando la presenza di dati sulla porta seriale, se presenti, legge un byte e lo invia immediatamente, in modo che i dati 485 ricevuti possano essere restituiti intatti.

##### Demo flashing

* Selezionare "ESP32S3 Dev Module" e la porta
* Impostare i parametri della scheda di sviluppo

[](https://www.waveshare.com/wiki/File:ESP32-Arduino-41.png)

* Flashare la demo

##### Dimostrazione dei risultati

* Aprire l'assistente di debug della porta seriale per inviare un messaggio al dispositivo ESP32-S3-Touch-LCD-4.3, che restituirà il messaggio ricevuto all'assistente per il debug della porta seriale

[](https://www.waveshare.com/wiki/File:ESP32-Arduino-32.png)

#### 03\_SD\_Test

##### Collegamento hardware

* Collegare la board al computer tramite un cavo USB.
* Inserire la scheda TF nella board

##### Analisi del codice

* **setup()**:
* setup esegue principalmente una serie di operazioni di inizializzazione e test sulla scheda TF
  + Innanzitutto, inizializza la comunicazione tramite porta seriale e imposta la velocità in baud a 115200. Quindi, crea un oggetto ESP\_IOExpander\_CH422G per gestire i pin IO estesi, inizializzare e impostare più pin come modalità di output e controllare lo stato di pin come reset del touchscreen (TP\_RST), retroilluminazione LCD (LCD\_BL), reset LCD (LCD\_RST), selezione della scheda TF (SD\_CS) e selezione USB (USB\_SEL).
  + Poi, utilizza i pin GPIO estesi per gestire la scheda TF, inizializzare la comunicazione SPI e tentare di montarla. Se il montaggio fallisce, verrà visualizzato un messaggio di errore. Se il montaggio ha esito positivo, verrà rilevato e visualizzato il tipo di scheda TF, insieme alle informazioni sulla sua dimensione.
  + Successivamente, si eseguono una serie di test sulle operazioni del file system, tra cui l'elenco del contenuto delle directory, la creazione di directory, l'eliminazione di directory, la scrittura su file, l'aggiunta di contenuto ai file, la lettura di file, l'eliminazione di file, la ridenominazione di file e il test di input e output dei file, nonché la visualizzazione dello spazio totale e dello spazio utilizzato della scheda TF.

##### Demo flashing

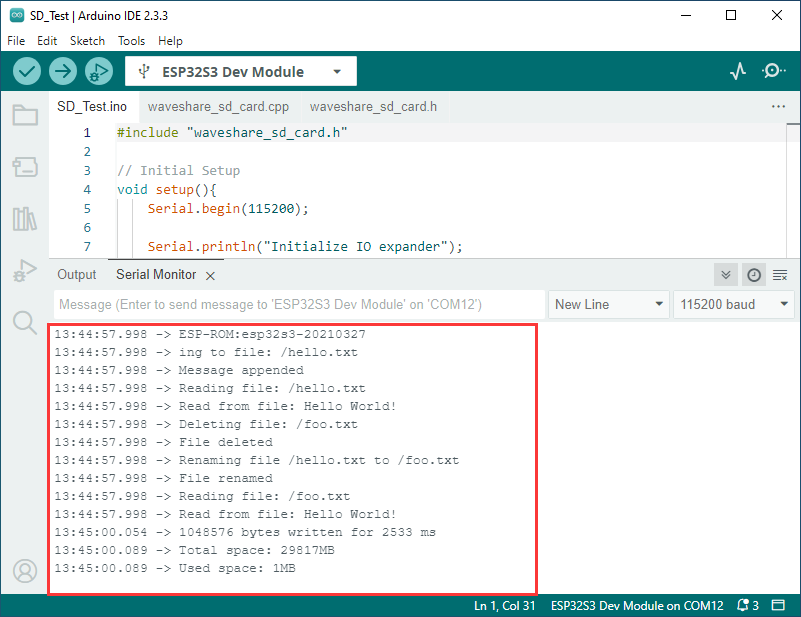
* Selezionare "ESP32S3 Dev Module" e la porta
* Impostare i parametri della scheda di sviluppo

[](https://www.waveshare.com/wiki/File:ESP32-Arduino-41.png)

* Flashare la demo

##### Dimostrazione dei risultati

* L'ESP32-S3-Touch-LCD-4.3 è in grado di identificare il tipo e la dimensione di una scheda TF, per poi eseguire operazioni sui file quali aggiunta, eliminazione, modifica e query.

[](https://www.waveshare.com/wiki/File:ESP32-Arduino-12.png)

#### 04\_Sensor\_AD\_Test

##### Collegamento hardware

* Collegare la board al computer tramite un cavo USB.
* Collegare il connettore maschio PH2.0-2,54 mm all'interfaccia Sensor AD della board.

##### Analisi del codice

* **setup()**:

La funzione setup inizializza prima la comunicazione seriale, poi inizializza l'ADC e imposta la risoluzione dell'ADC a 12 bit (0-4096).

* **loop()** :

La funzione loop legge il valore AD corrente ogni 100 ms e lo stampa.

##### Demo flashing

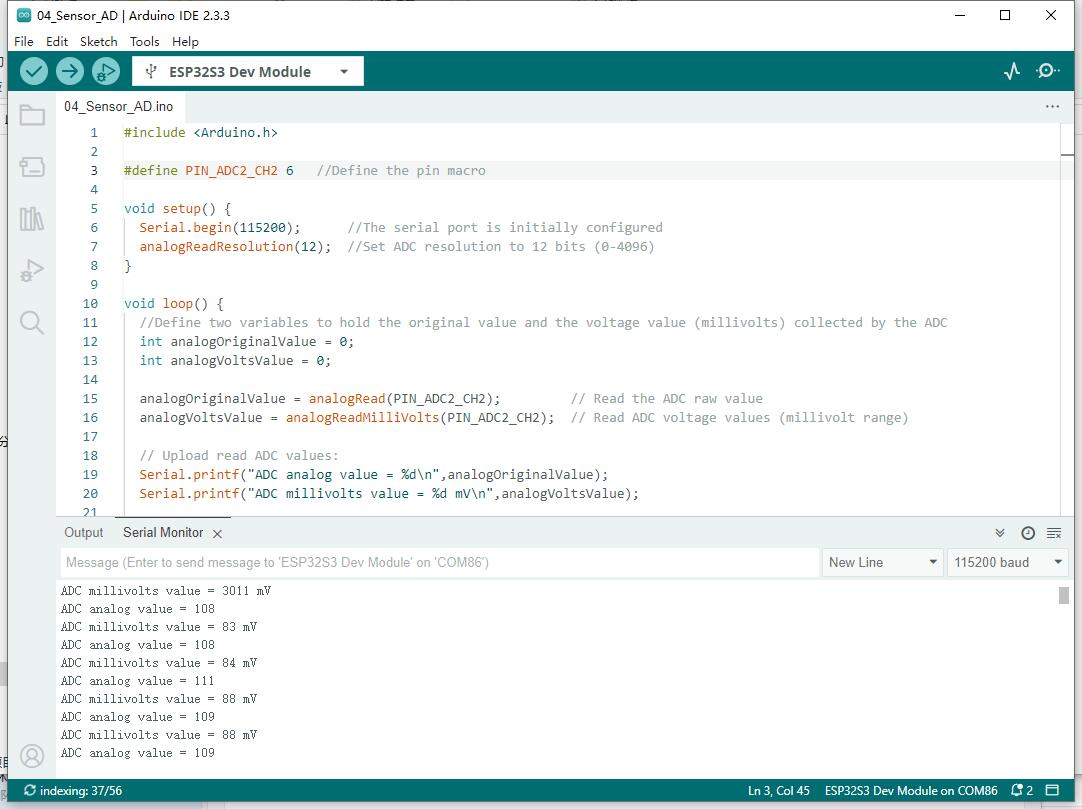
* Selezionare "ESP32S3 Dev Module" e la porta
* Impostare i parametri della scheda di sviluppo

[](https://www.waveshare.com/wiki/File:ESP32-Arduino-41.png)

* Flashare la demo

##### Dimostrazione dei risultati

* Dimostrazione dei risultati: ESP32-S3-Touch-LCD-4.3 imposterà la risoluzione ADC, leggerà il valore AD corrente e stamperà sul terminale della porta seriale

[](https://www.waveshare.com/wiki/File:ESP32-Arduino-40.jpg)

#### 05\_UART\_Test

##### Collegamento hardware

* Collegare la porta UART della scheda al computer tramite un cavo USB

##### Analisi del codice

* **setup()**:

setup viene utilizzata principalmente per inizializzare la comunicazione seriale.   
Utilizzare la funzione UART.begin per inizializzare la porta seriale Serial, impostare la velocità in baud, il formato dei dati e specificare i pin di ricezione e trasmissione. Quindi, tramite un ciclo, assicurarsi che l'inizializzazione della porta seriale abbia esito positivo.

* **loop()** :

loop è la parte principale del programma e la sua funzione principale è implementare una semplice comunicazione UART.   
Verificando se ci sono dati disponibili sulla porta seriale, legge un byte e lo invia immediatamente, in modo che i dati UART ricevuti possano essere inviati intatti.

##### Demo flashing

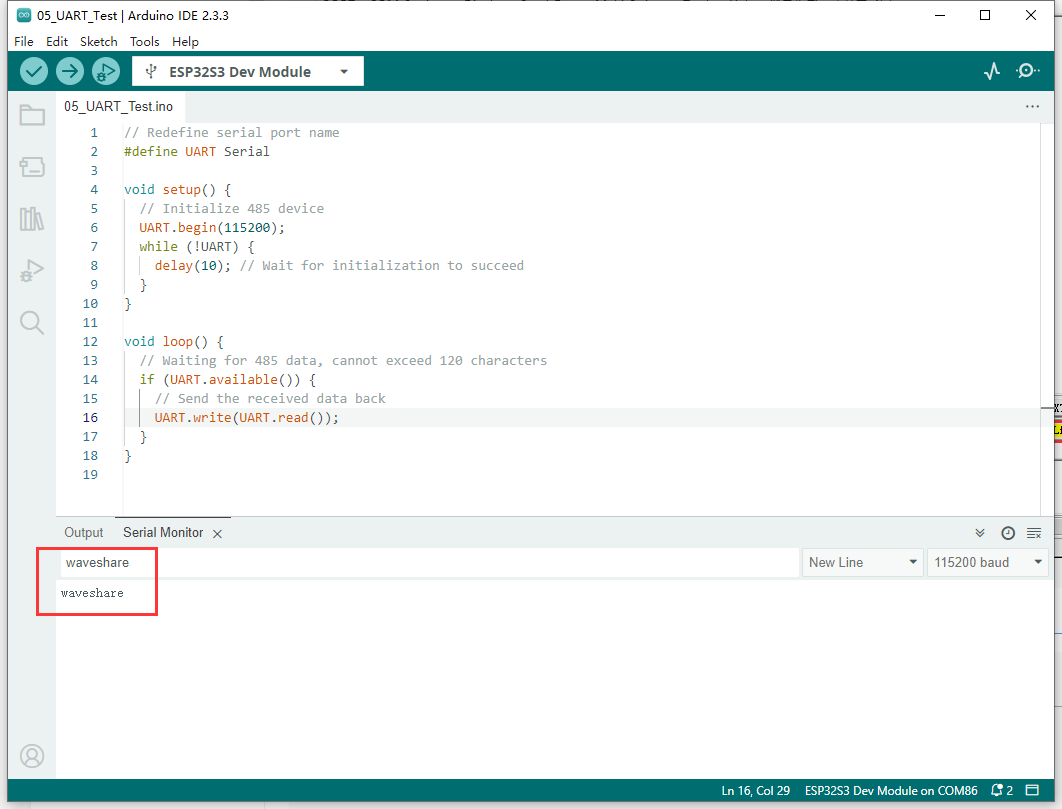
* Selezionare "ESP32S3 Dev Module" e la porta
* Impostare i parametri della scheda di sviluppo

[](https://www.waveshare.com/wiki/File:ESP32-Arduino-41.png)

* Flashare la demo

##### Dimostrazione dei risultati

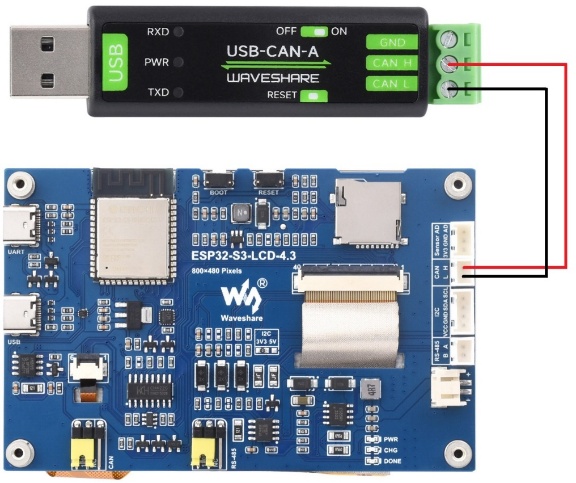
* Aprire l'assistente di debug della porta seriale per inviare un messaggio al dispositivo ESP32-S3-Touch-LCD-4.3, che restituirà il messaggio ricevuto all'assistente per il debug della porta seriale

[](https://www.waveshare.com/wiki/File:ESP32-Arduino-45.png)

#### 06\_TWAItransmit

##### Collegamento hardware

* Collegare la board al computer tramite un cavo USB.
* Collegare la scheda di sviluppo a [USB-CAN-A](https://www.waveshare.com/USB-CAN-A.htm), come mostrato in figura

[](https://www.waveshare.com/wiki/File:ESP32-Arduino-44.png)

##### Analisi del codice

* **waveshare\_twai\_transmit()**:
* waveshare\_twai\_transmit() gestisce principalmente la trasmissione e l'elaborazione degli allarmi per TWAI (un'interfaccia simile al CAN bus)
  + Innanzitutto, verifica se ci sono allarmi. Legge gli allarmi attivati ​​chiamando twai\_read\_alerts e ottiene le informazioni sullo stato TWAI in una struttura twai\_status\_info\_t. Poi, in base ai diversi allarmi attivati, esegue l'elaborazione corrispondente. Ad esempio, se viene attivato un allarme passivo di errore, un allarme di errore del bus, un allarme di errore di trasmissione o un allarme di trasmissione riuscita, stampa il messaggio corrispondente e restituisce alcune informazioni sullo stato, come il conteggio degli errori del bus, il numero di messaggi da inviare, il contatore degli errori di trasmissione e il contatore degli errori di trasmissione.
  + Poi, rileva l'ora corrente (in millisecondi) e verifica se è il momento di inviare il messaggio. Se la differenza tra l'ora corrente e l'ultima volta che un messaggio è stato inviato è maggiore o uguale all'intervallo di tempo di trasmissione impostato TRANSMIT\_RATE\_MS, aggiorna l'ora dell'ultimo invio all'ora corrente e chiama la funzione send\_message per inviare un messaggio. send\_message configura e mette in coda un messaggio contenente un identificatore specifico, una lunghezza dei dati e il contenuto dei dati per la trasmissione. Se l'inserimento in coda ha esito positivo, stampa un messaggio di successo; in caso contrario, stampa un messaggio di errore. Dopo l'invio, cancella l'array di dati del messaggio.

##### Demo flashing

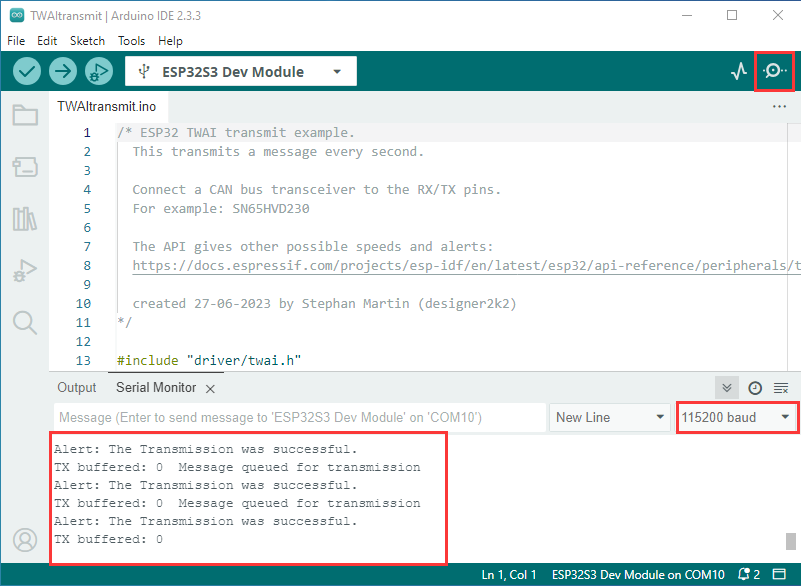
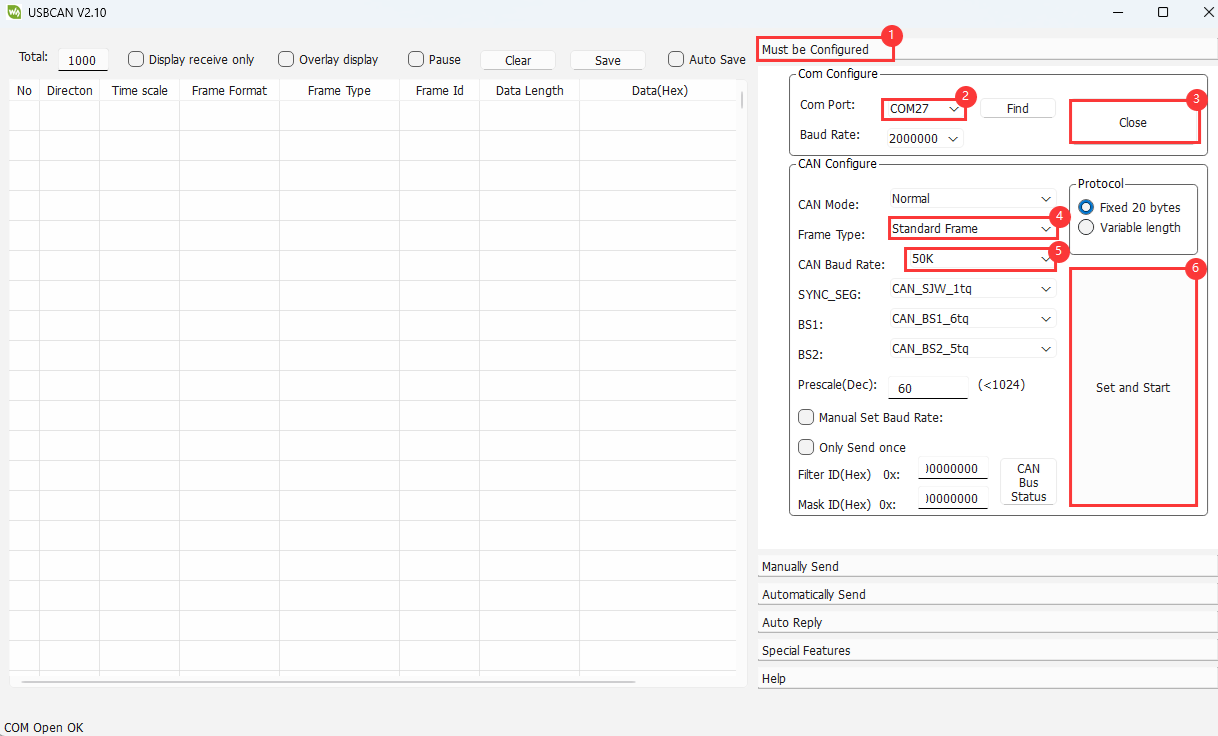
* Selezionare "ESP32S3 Dev Module" e la porta
* Impostare i parametri della scheda di sviluppo

[](https://www.waveshare.com/wiki/File:ESP32-Arduino-41.png)

* Flashare la demo

##### Dimostrazione dei risultati

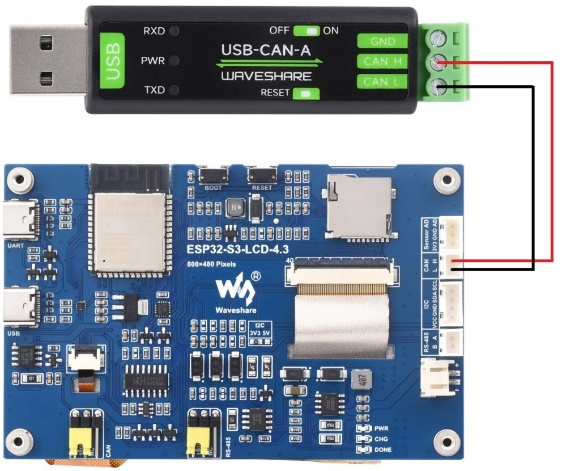
* La stampa sulla porta seriale indica la corretta trasmissione del messaggio CAN. Dopo aver configurato [USB-CAN-A\_TOOL](https://www.waveshare.com/wiki/ESP32-S3-Touch-LCD-4.3#Resources), è possibile osservare i messaggi CAN inviati dall'ESP32-S3-Touch-LCD-4.3 all'avvio.

[](https://www.waveshare.com/wiki/File:ESP32-Arduino-20.png)[](https://www.waveshare.com/wiki/File:ESP32-Arduino-CN-1.png)

#### 07\_TWAIreceive

##### Collegamento hardware

* Collegare la board al computer tramite un cavo USB.
* Collegare la scheda di sviluppo a [USB-CAN-A](https://www.waveshare.com/USB-CAN-A.htm), come mostrato in figura

[](https://www.waveshare.com/wiki/File:ESP32-Arduino-44.png)

##### Analisi del codice

* **waveshare\_twai\_receive()**:
  + Innanzitutto, l'allarme attivato viene letto e vengono ottenute le informazioni sullo stato, quindi viene eseguita l'elaborazione corrispondente in base a diverse condizioni di allarme, come errore di stampa passivo, errore del bus, coda di ricezione piena e altre informazioni di allarme e relativi conteggi.
  + Quando viene attivato un avviso sui dati, il ciclo riceve i messaggi e richiama la funzione handle\_rx\_message per elaborarli. Questa funzione determina il formato del messaggio e ne stampa l'identificativo e il contenuto dei dati (escluse le richieste di trasmissione remota), gestendo efficacemente i messaggi ricevuti sul bus TWAI e rispondendo agli avvisi.

##### Demo flashing

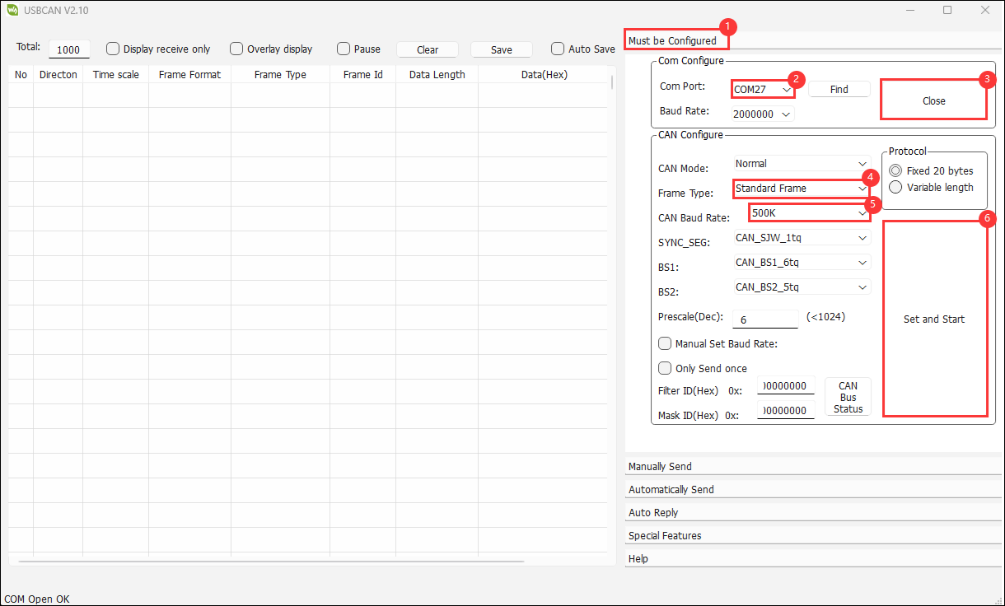
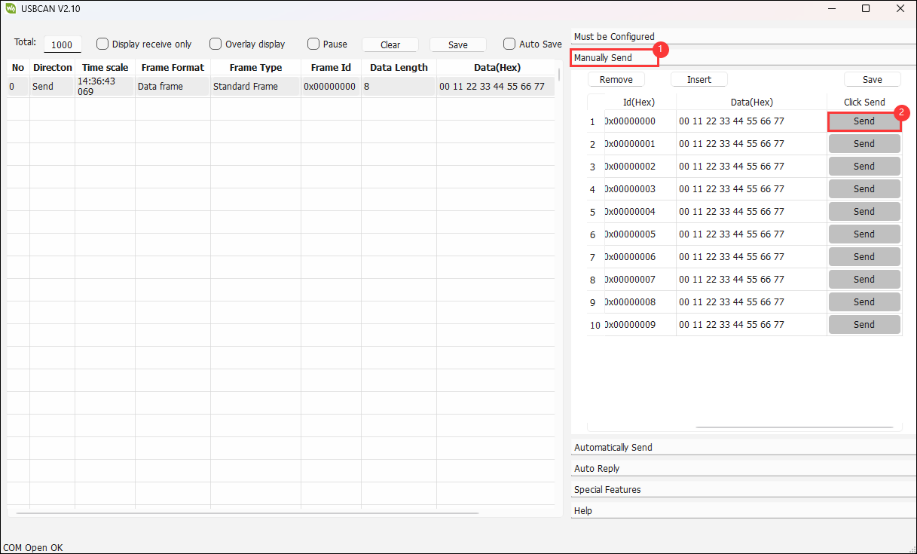
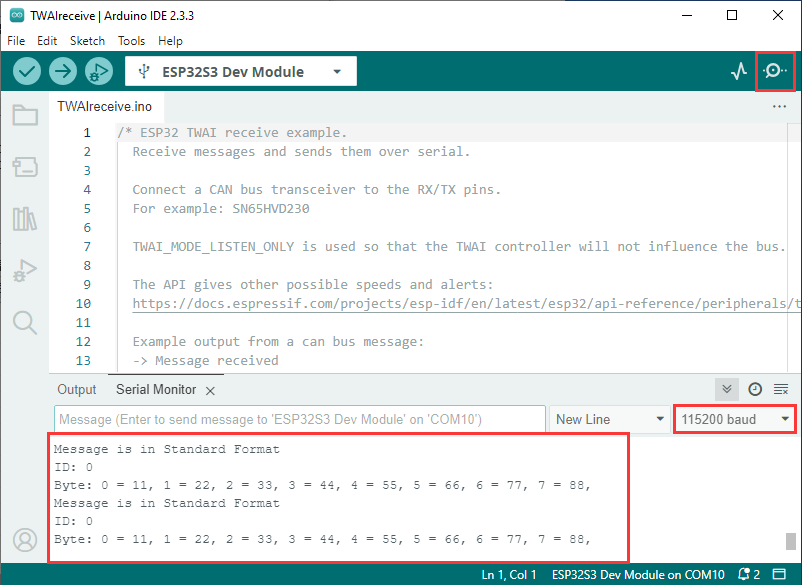
* Selezionare "ESP32S3 Dev Module" e la porta
* Impostare i parametri della scheda di sviluppo

[](https://www.waveshare.com/wiki/File:ESP32-Arduino-41.png)

* Flashare la demo

##### Dimostrazione dei risultati

* ESP32-S3-Touch-LCD-4.3 attende [USB-CAN-A\_TOOL](https://www.waveshare.com/wiki/ESP32-S3-Touch-LCD-4.3#Resources) per inviare un messaggio. Se il messaggio viene ricevuto correttamente, verrà stampato sulla porta seriale.

[](https://www.waveshare.com/wiki/File:ESP32-Arduino-CN-2.png)[](https://www.waveshare.com/wiki/File:ESP32-Arduino-23.png)[](https://www.waveshare.com/wiki/File:ESP32-Arduino-22.png)

#### 08\_DrawColorBar

##### Collegamento hardware

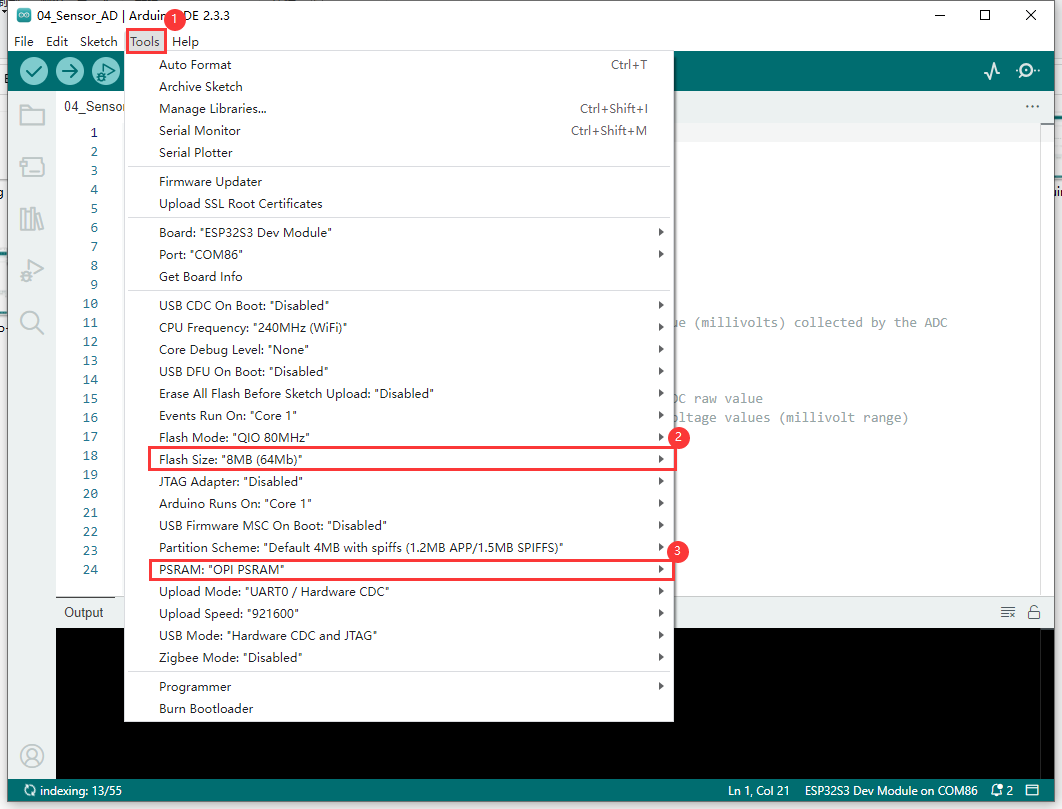
* Collegare la board al computer tramite un cavo USB.

##### Analisi del codice

* **waveshare\_lcd\_init()**:
  + Innanzitutto, stampa "Initialize IO expander" per indicare l'avvio dell'inizializzazione dell'espansore IO. Poi crea un'istanza di ESP\_IOExpander\_CH422G, la inizializza e ne avvia il funzionamento. Impostare i pin IO0 - IO7 in modalità output e imposta il pin di reset del touchscreen (TP\_RST) e il pin di reset del LCD (LCD\_RST) a livello alto, spegnendo la retroilluminazione LCD (LCD\_BL) e attendendo 100 ms.
  + Successivamente, stampa "Create RGB LCD bus", crea un oggetto bus pannello RGB ESP\_PanelBus\_RGB, configura i suoi pin, larghezza, altezza, frequenza di temporizzazione RGB e parametri di temporizzazione, ecc., imposta la dimensione del buffer di rimbalzo e visualizza il flag di livello basso attivo, quindi avvia il funzionamento del bus pannello.
  + Poi, stampa "Create LCD device", crea un oggetto LCD ESP\_PanelLcd, passa parametri come l'oggetto bus del pannello, il numero di bit di colore e i pin di reset per inizializzare, resettare e avviare l'operazione. Se EXAMPLE\_ENABLE\_PRINT\_LCD\_FPS è definito, la funzione di callback che termina con VSync viene associata all'oggetto LCD.
  + Infine, stampa "Draw color bar from top left to bottom right, the order is B - G - R", chiama la funzione colorBarTest per disegnare la barra dei colori sull'LCD e accende la retroilluminazione dell'LCD.

##### Demo flashing

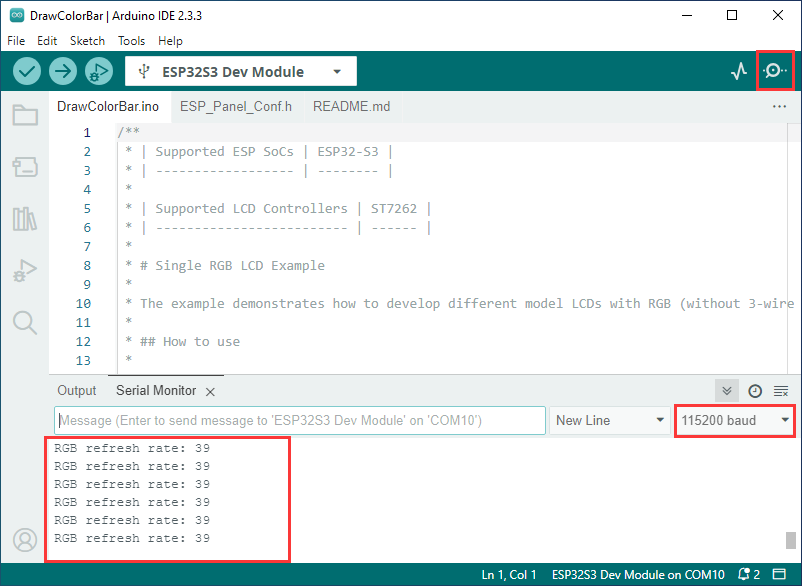
* Selezionare "ESP32S3 Dev Module" e la porta
* Impostare i parametri della scheda di sviluppo

[](https://www.waveshare.com/wiki/File:ESP32-Arduino-42.png)

* Flashare la demo

##### Dimostrazione dei risultati

* La porta seriale stampa il log e lo schermo si illumina

[](https://www.waveshare.com/wiki/File:ESP32-Arduino-24.png)[](https://www.waveshare.com/wiki/File:800px-ESP32-Arduino-25.png)

#### 09\_lvgl\_Porting

##### Collegamento hardware

* Collegare la board al computer tramite un cavo USB.

##### Analisi del codice

* **setup()**:

Inizializza la comunicazione seriale a una velocità di 115200 baud. Successivamente, crea e inizializza l'espansore IO, impostare la modalità e lo stato dei pin e inizializza il touchscreen GT911. Quindi, crea e inizializza il dispositivo del pannello, configura il bus RGB secondo necessità. Infine, inizializza LVGL, crea un'etichetta semplice o una funzione di esempio o demo che opzionalmente richiama LVGL e rilascia il blocco mutex alla fine.

* **loop()** :

Stampa solo "IDLE loop" e attende 1 secondo senza altre azioni sostanziali. L'obiettivo generale è creare un ambiente di interfaccia utente basato su LVGL.

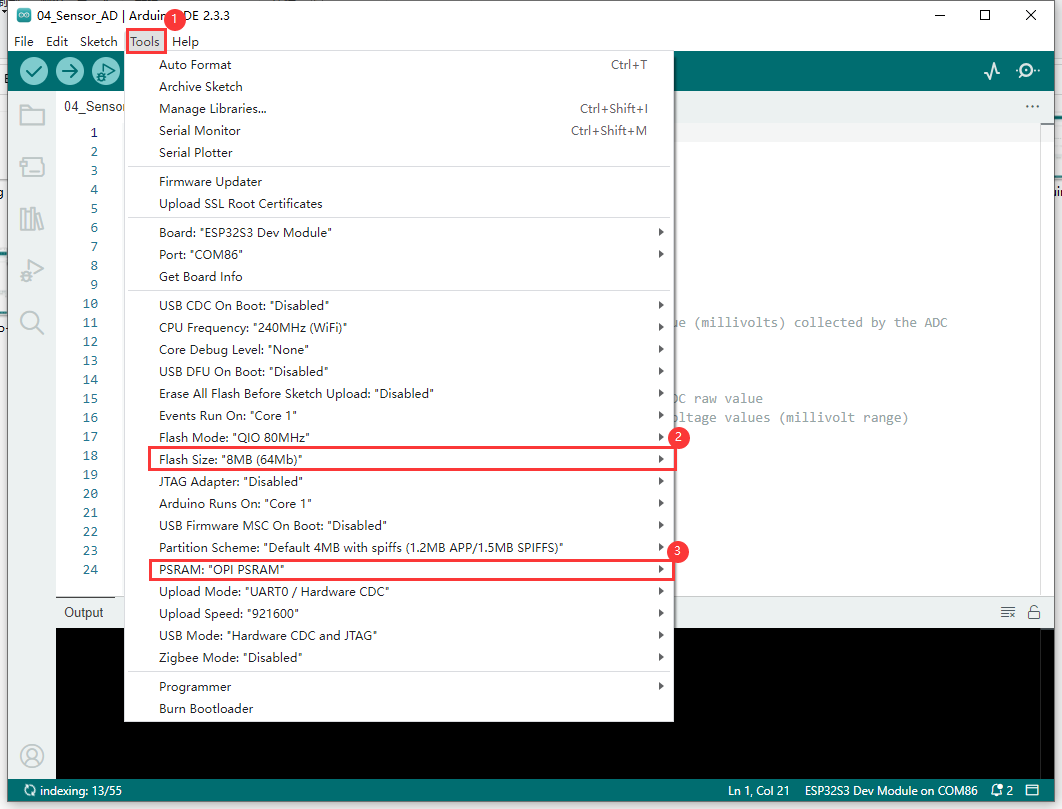
##### Modifica del codice

* In ESP\_Panel\_Board\_Custom.h è presente una definizione macro per selezionare se attivare o meno la funzione touch: il valore 0 corrisponde alla disattivazione del touch, mentre il valore 1 corrisponde all'attivazione del touch, e può essere selezionato in base al modello acquistato.

#define ESP\_OPEN\_TOUCH 0 // 1 attiva il touch, 0 lo disattiva.

##### Demo flashing

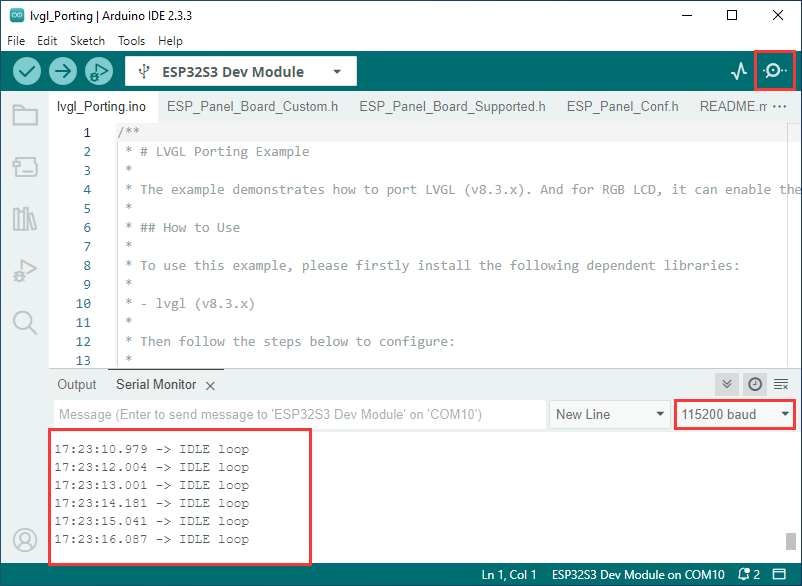
* Selezionare "ESP32S3 Dev Module" e la porta
* Impostare i parametri della scheda di sviluppo

[](https://www.waveshare.com/wiki/File:ESP32-Arduino-42.png)

* Flashare la demo

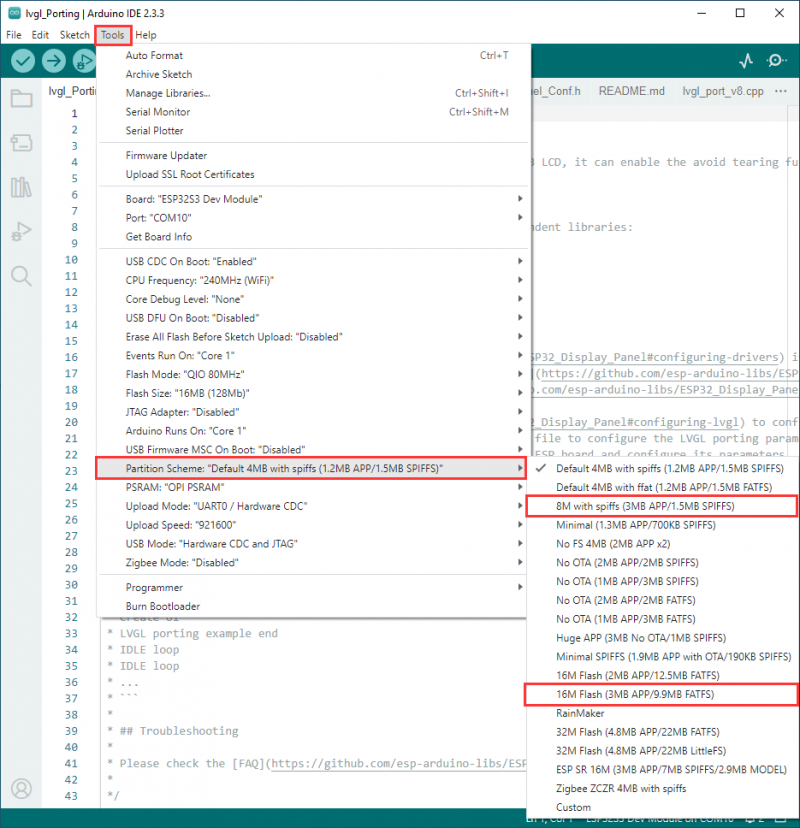
##### Dimostrazione dei risultati

* La porta seriale stampa la frequenza di aggiornamento dello schermo e poi si illumina.

[](https://www.waveshare.com/wiki/File:ESP32-Arduino-26.png)   
[Screen lights up\_video](https://www.youtube.com/watch?v=luICguq9cjo)

### Altre istruzioni

* Se si verifica un drifting dello schermo durante l'utilizzo, consultare le [FAQ ufficiali di ESP](https://docs.espressif.com/projects/esp-faq/en/latest/software-framework/peripherals/lcd.html#why-do-i-get-drift-overall-drift-of-the-display-when-esp32-s3-is-driving-an-rgb-lcd-screen)

Se si utilizza il proprio programma di interfaccia utente e si verifica una mancanza di memoria, è possibile cliccare su Tools per selezionare una tabella delle partizioni più grande   
[](https://www.waveshare.com/wiki/File:800px-ESP32-Arduino-28.png)

* La versione di lvgl utilizzata è la 8.4 ed è possibile interrogare e utilizzare l'API LVGL tramite la seguente documentazione:
  + [Documentazione LVGL](https://docs.lvgl.io/8.4/)

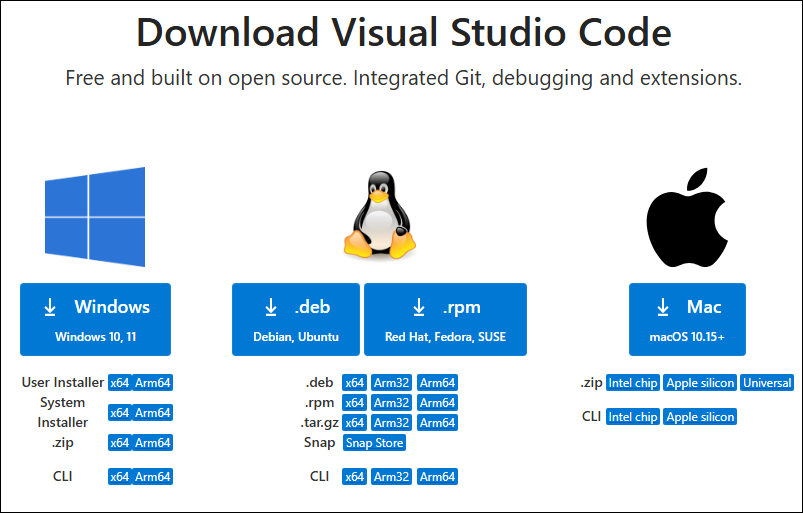
# Utilizzo di ESP-IDF

Questo capitolo introduce la configurazione dell'ambiente ESP-IDF, inclusa l'installazione di Visual Studio e del plugin Espressif IDF, la compilazione del programma, il download e il test delle demo, per aiutare gli utenti a padroneggiare la scheda di sviluppo e facilitare lo sviluppo secondario.

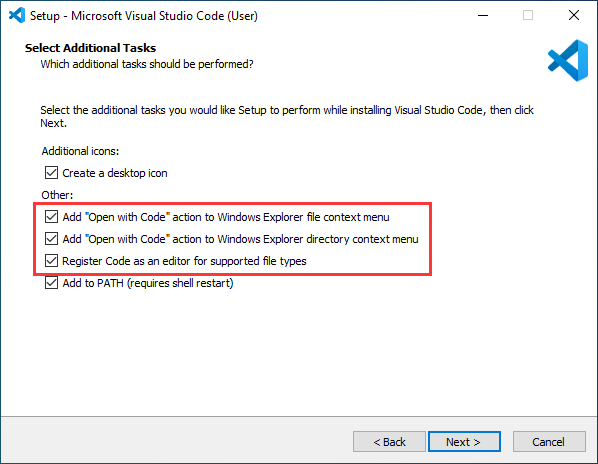
## Configurazione dell'Ambiente

### Download e Installazione di Visual Studio

Aprire la pagina di download del [sito web ufficiale di VScode](https://code.visualstudio.com/download), scegliere il sistema e il bit di sistema corrispondenti da scaricare

[](https://www.waveshare.com/wiki/File:ESP32-S3-AMOLED-1.91-VScode-01.png)

Dopo aver eseguito il pacchetto di installazione, il resto può essere installato per default, ma per un'esperienza utente successiva, si consiglia di selezionare le caselle 1, 2 e 3

[](https://www.waveshare.com/wiki/File:ESP32-S3-AMOLED-1.91-VScode-02.png)

* + Dopo aver abilitato i primi due elementi, è possibile aprire VSCode direttamente cliccando con il pulsante destro del mouse su file o directory, il che può migliorare l'esperienza utente successiva.
  + Dopo aver abilitato la terza voce, si può selezionare VSCode direttamente quando si sceglie come aprirlo.

La configurazione dell'ambiente viene eseguita sul sistema Windows 10; gli utenti Linux e Mac possono accedere a [Configurazione dell'ambiente ESP-IDF](https://docs.espressif.com/projects/esp-idf/en/v5.1.4/esp32s3/get-started/windows-setup.html) come riferimento.

### Installazione del Plugin Espressif IDF

* In genere, si consiglia di utilizzare l'**Installazione online**. Se l'installazione online non riesce a causa di problemi di rete, utilizzare **Installazione Offline**.
* Per ulteriori informazioni su come installare il plugin Espressif IDF, consultare [Installazione del plugin Espressif IDF](https://www.waveshare.com/wiki/Install_Espressif_IDF_Plugin_Tutorial)

## Eseguire la prima demo di ESP-IDF

Se si è appena iniziato a usare ESP32 ed ESP-IDF e non si sa come creare, compilare, flashare ed eseguire programmi ESP-IDF ESP32, espandere e dare un'occhiata. Si spera che possa essere utile!

### Nuovo Progetto

### Creazione di Demo

* Utilizzando la scorciatoia F1, immettere esp-idf:show examples projects
* Selezionare la versione IDF attuale
* Si prende la demo Hello World come esempio

① Selezionare la demo corrispondente

② Il suo readme indicherà a quale chip si applica la demo (le modalità di utilizzo della demo e la struttura dei file sono descritte di seguito, omesse qui)

③ Cliccare per creare la demo

* Selezionare il percorso in cui salvare la demo e impostare che le demo non possano utilizzare cartelle con lo stesso nome

### Modifica della Porta COM

* Le porte COM corrispondenti sono mostrate qui, cliccare per modificarle.
* Selezionare le porte COM in base al dispositivo (si può visualizzarle da "device manager")
* In caso di errore del download, premere il pulsante Reset per più di 1 secondo o entrare in modalità download e attendere che il PC riconosca nuovamente il dispositivo prima di eseguire un altro download.

### Modifica dell'Oggetto Driver

* Si seleziona l'oggetto da pilotare, ovvero il chip principale ESP32S3.
* Si sceglie il percorso openocd, non ci riguarda qui, quindi scegliamone uno.

### Altre Funzioni della Barra di Stato

①.Gestore delle versioni dell'ambiente di sviluppo ESP-IDF: quando il progetto richiede la differenziazione delle versioni dell'ambiente di sviluppo, è possibile gestirlo installando diverse versioni di ESP-IDF. Quando il progetto utilizza una versione specifica, è possibile passare a questa utilizzandola

②.Porta COM flash del dispositivo, selezionare per flashare il programma compilato nel chip

③.Selezionare il modello di chip target impostato, selezionare il modello di chip corrispondente, ad esempio, ESP32-P4-NANO deve scegliere esp32p4 come chip target

④.menuconfig, cliccarci sopra per modificare il file di configurazione sdkconfig [Dettagli di configurazione del progetto](https://docs.espressif.com/projects/esp-idf/en/latest/esp32s3/api-reference/kconfig.html)

⑤.Pulsante fullclean: quando un errore di compilazione del progetto o altre operazioni inquinano il contenuto compilato, è possibile ripulirlo completamente cliccandoci sopra.

⑥.Build project, quando un progetto soddisfa i requisiti di build, fare clic su questo pulsante per compilarlo

⑦.Modalità di download corrente: l'impostazione predefinita è UART.

⑧.Pulsante flash: quando la build di un progetto è completata, selezionare la porta COM della scheda di sviluppo corrispondente e fare clic su questo pulsante per flashare il firmware compilato sul chip.

⑨.monitor abilita il monitoraggio della porta flash. Quando un progetto passa attraverso Build --> Flash, fare clic su questo pulsante per visualizzare il log dell'output dalla porta flash e dalla porta di debug, in modo da verificare il corretto funzionamento dell'applicazione.

⑩.Debug

⑪.Pulsante Build Flash Monitor: con un clic, viene utilizzato per eseguire in modo continuo Build --> Flash --> Monitor, spesso definito "little flame" (piccola fiamma).

### Compilazione, Flash e Monitor della Porta Seriale

* Cliccare sul pulsante multifunzione descritto in precedenza per compilare, flashare e aprire il monitor della porta seriale.
* La compilazione potrebbe richiedere molto tempo, soprattutto la prima volta.
* Durante questo processo, l'ESP-IDF potrebbe occupare molte risorse della CPU, causando rallentamenti del sistema.
* Se è la prima volta che si esegue il flashing del programma per un nuovo progetto, è necessario selezionare il metodo di download e selezionare UART.
* Questa impostazione può essere modificata in seguito nella sezione dei Metodi di Download (cliccare per visualizzare le opzioni).
* Poiché è dotato di un circuito di download automatico integrato, può essere scaricato automaticamente senza operazioni manuali.
* Al termine del download, il dispositivo accederà automaticamente al monitor seriale, dove sarà possibile visualizzare le informazioni corrispondenti in uscita dal chip e ricevere la richiesta di riavvio dopo 10 secondi.

### Utilizzo delle Demo IDF

Quanto segue prende come esempio la **ESP32-S3-LCD-1.47-Demo** per introdurre i due metodi di apertura del progetto, le fasi generali di utilizzo e la spiegazione dettagliata del progetto ESP-IDF. Se si utilizzano altri progetti, le fasi operative possono essere applicate in modo simile.

### Aprire il Software

* Aprire il software VScode e selezionare la cartella in cui aprire la demo
* Selezionare l'esempio ESP-IDF fornito e cliccare per selezionare il file (che si trova nel percorso /Demo/ESP-IDF sotto demo)

#### Aprire dall'Esterno del Software

* Selezionare correttamente la directory del progetto e aprire il progetto, altrimenti ciò influirà sulla compilazione e sull'installazione dei programmi successivi.
* Dopo aver collegato il dispositivo, selezionare la porta COM e il modello, quindi cliccare di seguito per compilare e flashare per ottenere il controllo del programma.

#### Dettagli del Progetto ESP-IDF

* Componente: I componenti in ESP-IDF sono i moduli di base per la creazione di applicazioni. Ogni componente è solitamente una base di codice o libreria relativamente indipendente, che può implementare funzioni o servizi specifici e può essere riutilizzata da applicazioni o altri componenti, in modo simile alla definizione di librerie nello sviluppo Python.
  + Riferimento ai componenti: L'importazione di librerie nell'ambiente di sviluppo Python richiede solo "importare il nome o il percorso della libreria", mentre ESP-IDF si basa sul linguaggio C e l'importazione delle librerie viene configurata e definita tramite CMakeLists.txt.
  + Scopo di CmakeLists.txt: durante la compilazione di ESP-IDF, lo strumento di compilazione CMake legge prima il contenuto del file CMakeLists.txt di primo livello nella directory del progetto per leggere le regole di compilazione e identificare il contenuto da compilare. Quando i componenti e le demo richiesti vengono importati in CMakeLists.txt, lo strumento di compilazione CMake importerà ogni contenuto che deve essere compilato in base all'indice. Il processo di compilazione è il seguente:

## Demo

|  |  |
| --- | --- |
| * Demo di ESP32-S3-Touch-LCD-4.3 | |
| **Demo** | **Descrizione di Base** |
| 01\_I2C\_Test | Test header I2C |
| 02\_RS485\_Test | Test header RS485 |
| 03\_SD\_Test | Test slot per scheda TF |
| 04\_Sensor\_AD | Test header ADC |
| 05\_UART\_Test | Test UART |
| 06\_TWAItransmit | Test CAN seat |
| 07\_TWAIreceive | Test CAN seat |
| 08\_lvgl\_Porting | Test del porting UART |

* Le librerie dipendenti vengono scaricate automaticamente in fase di compilazione tramite IDF component.yml
  + Fare riferimento a [Gestore Componenti IDF](https://docs.espressif.com/projects/esp-idf/en/latest/esp32s3/api-guides/tools/idf-component-manager.html) per ulteriori link

### 01\_I2C\_Test

#### Collegamento hardware

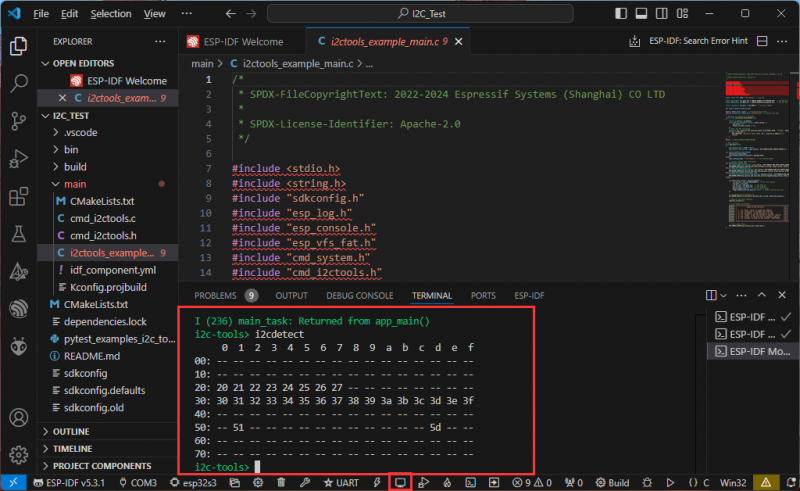
* Collegare la board al computer tramite un cavo USB.

#### Analisi del codice

* **app\_main()**:
  + Innanzitutto, vengono definite le costanti e le variabili relative a I2C, come i tag di log, i pin SDA e SCL per I2C e i numeri di porta.
  + In seguito, si installa l'ambiente REPL della console per l'interazione dell'utente in base alle diverse opzioni di configurazione. Quindi, si configura il bus I2C, inclusi la sorgente di clock, le porte, i pin e si abilitano le resistenze di pull-up interne, e si inizializza il bus master I2C.
  + Viene quindi registrata una serie di comandi di utilità I2C, come i comandi per il rilevamento dei dispositivi, le letture e le scritture dei registri e così via. Vengono inoltre stampate istruzioni per l'uso che spiegano all'utente come utilizzare questi comandi.
  + Infine, si avvia il REPL della console, che consente agli utenti di interagire con l'applicazione tramite la riga di comando ed eseguire diverse operazioni I2C, offrendo loro un modo pratico per gestire il bus I2C tramite la riga di comando.

#### Dimostrazione dei risultati

* Dopo il flashing, aprire il terminale seriale, inserire il comando e premere Enter per eseguirlo:

[](https://www.waveshare.com/wiki/File:800px-ESP32-IDF-14.png)

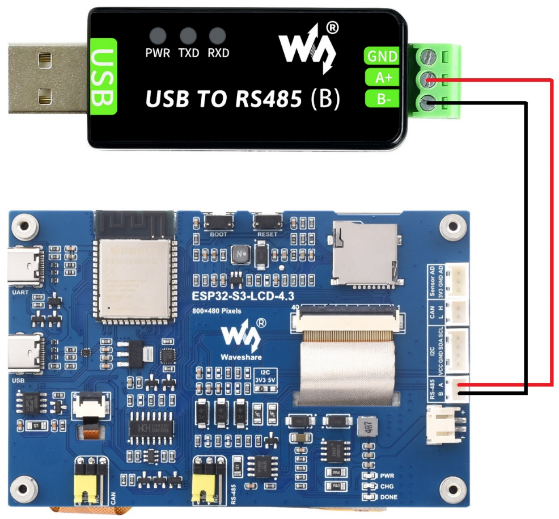
* I passaggi sono i seguenti:

1. Usare "help" per verificare tutti i comandi supportati
2. Usare "i2cconfig" per configurare il bus I2C
3. Usare "i2cdetect" per scansionare i dispositivi sul bus
4. Usare "i2cget" per recuperare il contenuto di un registro specifico
5. Usare "i2cset" per impostare il valore di un registro specifico
6. Usare "i2cdump" per eseguire il dump di tutti i registri (esperimento)

### 02\_RS485\_Test

#### Collegamento hardware

* Collegare la board al computer tramite un cavo USB.
* Collegare la scheda di sviluppo al [convertitore USB-RS485](https://www.waveshare.com/USB-TO-RS485-B.htm), come mostrato in figura

[](https://www.waveshare.com/wiki/File:ESP32-Arduino-43.png)

#### Analisi del codice

* **echo\_task()**:
  + Innanzitutto, sono stati configurati i parametri UART, tra cui baud rate, bit di dati, bit di parità, bit di stop e controllo di flusso hardware, ecc.
  + Quindi, è stato installato il driver UART, sono stati impostati i pin UART e è stato allocato un buffer temporaneo per la ricezione dei dati.
  + In un ciclo infinito, i dati vengono letti dall'UART, i dati letti vengono riscritti sull'UART e le informazioni di log vengono registrate al momento della ricezione.

#### Dimostrazione dei risultati

* Aprire l'assistente di debug della porta seriale per inviare un messaggio al dispositivo ESP32-S3-Touch-LCD-4.3, che restituirà il messaggio ricevuto all'assistente per il debug della porta seriale

[](https://www.waveshare.com/wiki/File:ESP32-Arduino-32.png)

### 03\_SD\_Test

#### Collegamento hardware

* Collegare la board al computer tramite un cavo USB.

#### Analisi del codice

* **waveshare\_sd\_card\_init()**:

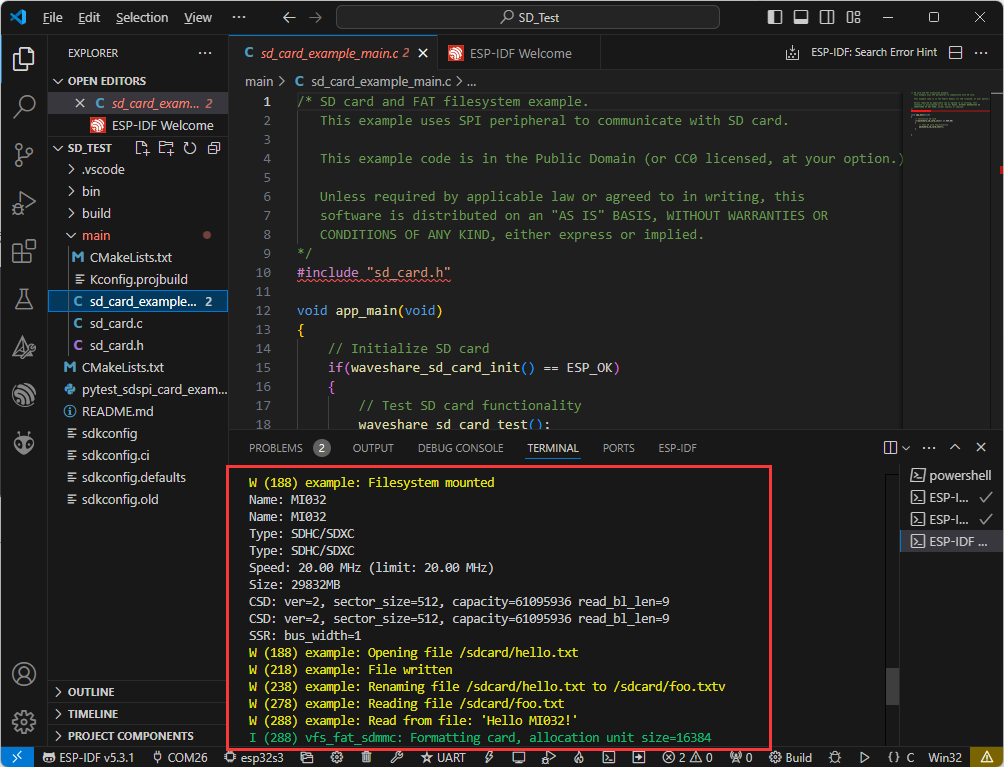
Questa funzione viene utilizzata principalmente per inizializzare la scheda TF. Per prima cosa, inizializza l'I2C, abbassando il pin CS della scheda TF tramite il chip di controllo I2C. Quindi, configura le opzioni di montaggio della scheda TF, tra cui se formattare in caso di errore di montaggio, il numero massimo di file, la dimensione dell'unità di allocazione, ecc. Successivamente, inizializza il bus SPI e monta il file system della scheda TF utilizzando il bus SPI configurato e le opzioni di montaggio. Se il montaggio ha esito positivo, restituisce ESP\_OK, a indicare che l'inizializzazione della scheda TYF è completa.

* **waveshare\_sd\_card\_test()**:

Questa funzione viene utilizzata per testare la funzionalità della scheda TF. Per prima cosa, stampa le informazioni della scheda TF inizializzata. Poi crea un file, ci scrive i dati, lo rinomina e ne legge il contenuto. Quindi, formatta il file system e verifica se il file è stato eliminato dopo la formattazione. Infine, crea un nuovo file e ne legge il contenuto, smonta la scheda TF e libera le risorse del bus SPI al termine del test.

#### Dimostrazione dei risultati

* Dopo aver completato la programmazione, la porta seriale visualizzerà informazioni sulla scheda di memoria, come nome, tipo, capacità e frequenza massima supportata. Quindi, creerà un file, scriverà al suo interno, lo rinominerà e leggerà il file rinominato:

[](https://www.waveshare.com/wiki/File:ESP32-IDF-18.png)

### 04\_Sensor\_AD\_Test

#### Collegamento hardware

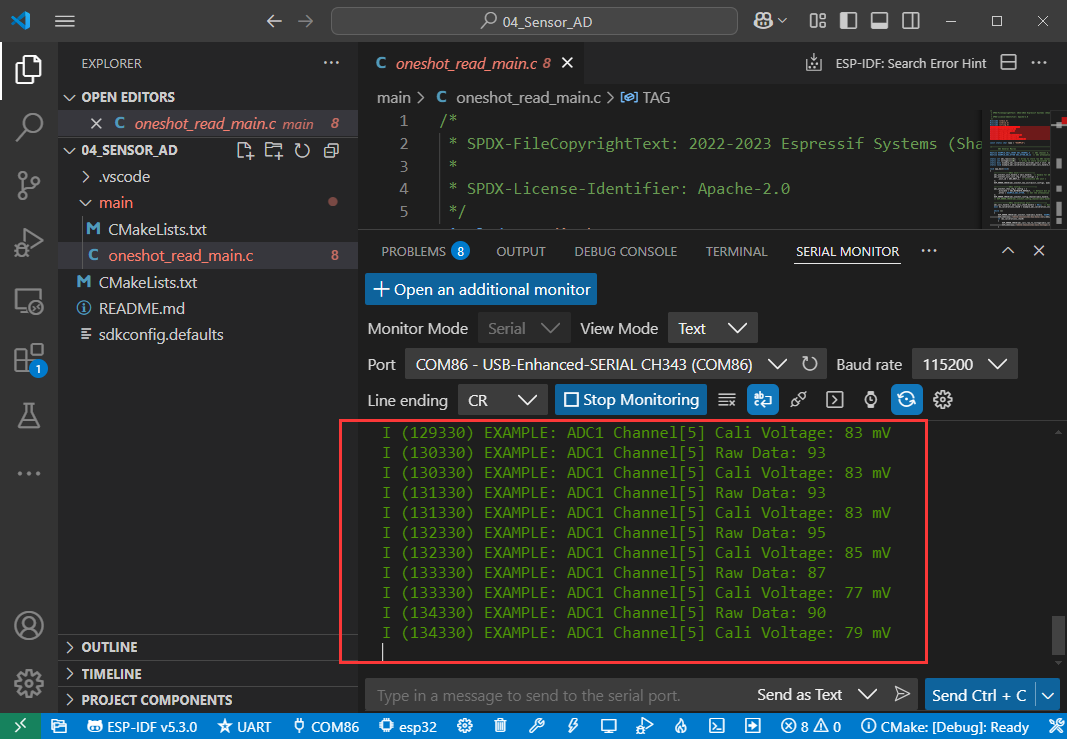
* Collegare la board al computer tramite un cavo USB.
* Collegare il connettore maschio PH2.0-2,54 mm all'interfaccia Sensor AD della board.

#### Analisi del codice

* **app\_main()**:
  + Innanzitutto, vengono definite alcune variabili utilizzate per memorizzare il valore corrente dell'ADC e viene dichiarata la funzione di calibrazione.
  + Successivamente, si inizializza l'ADC, se ne impostano la risoluzione e attenuazione, quindi si crea l'ADC. In un ciclo infinito, c'è anche un ritardo di 1 secondo nel ciclo di stampa dopo la lettura del valore corrente dell'ADC.

#### Dimostrazione dei risultati

* Dimostrazione dei risultati: ESP32-S3-Touch-LCD-4.3 imposterà la risoluzione ADC, leggerà il valore AD corrente e stamperà sul terminale della porta seriale

[](https://www.waveshare.com/wiki/File:ESP32-Arduino-46.png)

### 05\_UART\_Test

#### Collegamento hardware

* Collegare la porta UART della scheda al computer tramite un cavo USB

#### Analisi del codice

* **echo\_task()**:
  + Innanzitutto, sono stati configurati i parametri UART, tra cui baud rate, bit di dati, bit di parità, bit di stop e controllo di flusso hardware, ecc.
  + Quindi, è stato installato il driver UART, sono stati impostati i pin UART e è stato allocato un buffer temporaneo per la ricezione dei dati.
  + In un ciclo infinito, i dati vengono letti dall'UART, i dati letti vengono riscritti sull'UART e le informazioni di log vengono registrate al momento della ricezione.

#### Dimostrazione dei risultati

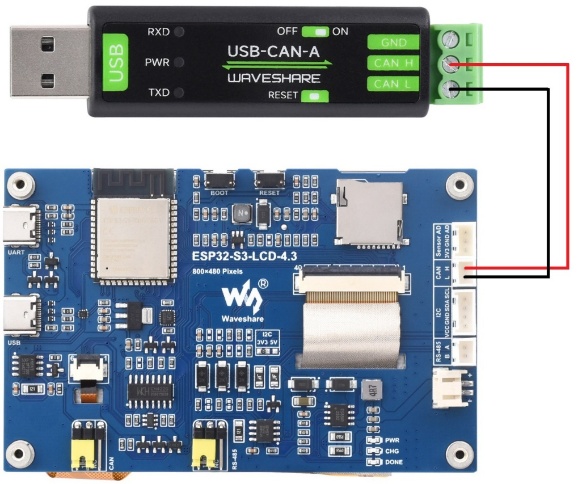
* Aprire l'assistente di debug della porta seriale per inviare un messaggio al dispositivo ESP32-S3-Touch-LCD-4.3, che restituirà il messaggio ricevuto all'assistente per il debug della porta seriale

[](https://www.waveshare.com/wiki/File:ESP32-Arduino-32.png)

### 06\_TWAItransmit

#### Collegamento hardware

* Collegare la board al computer tramite un cavo USB.
* Collegare la scheda di sviluppo a [USB-CAN-A](https://www.waveshare.com/USB-CAN-A.htm), come mostrato in figura

[](https://www.waveshare.com/wiki/File:ESP32-Arduino-44.png)

#### Analisi del codice

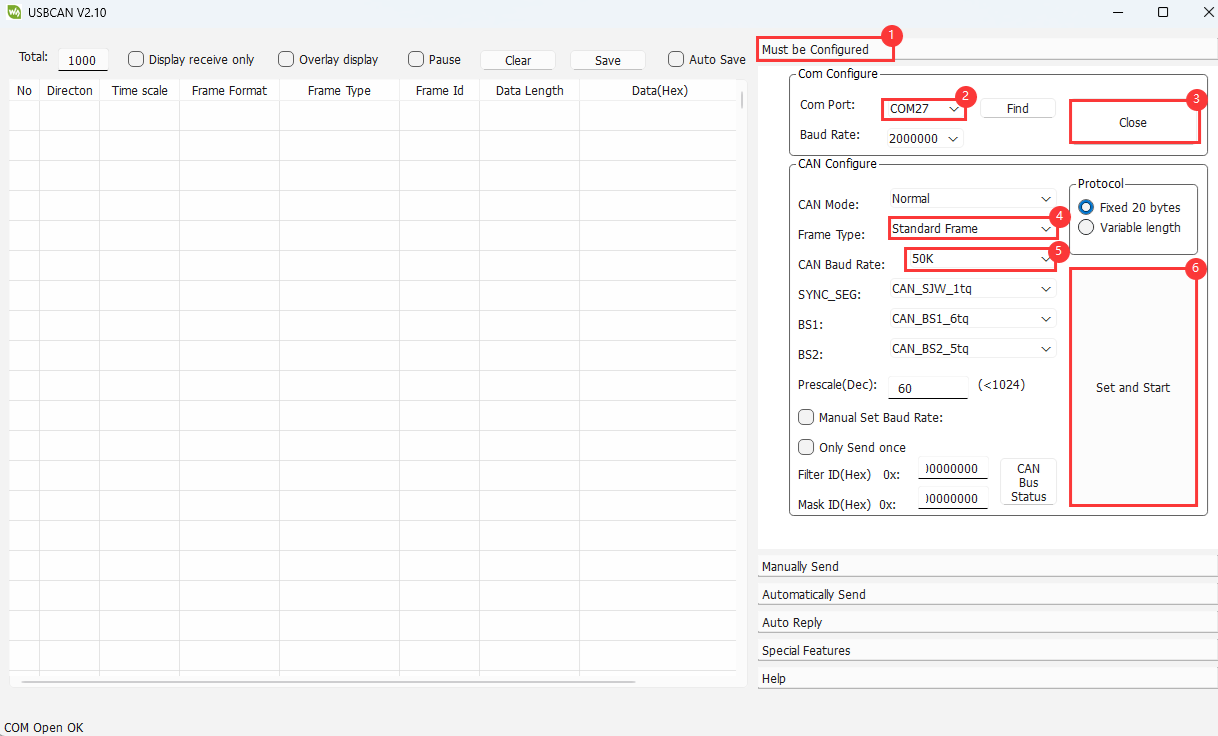
* **waveshare\_twai\_transmit()**:
  + Se il driver non è installato, tornerà allo stato di errore dopo un certo periodo di tempo.
  + Legge gli "alert" attivati ​​e ottien informazioni sullo stato TWAI.
  + Stampa le informazioni dei log corrispondenti in base ai diversi tipi di allarme, inclusi allarmi di errore passivo, allarmi di errore bus, allarmi di errore di trasmissione e allarmi di trasmissione riuscita, e stampa le relative informazioni dello stato.
  + Determina se è il momento di inviare un messaggio e, in tal caso, lo invia e aggiorna l'ultima volta che è stato inviato.

#### Dimostrazione dei risultati

* La stampa sulla porta seriale indica la corretta trasmissione del messaggio CAN. Dopo aver configurato [USB-CAN-A\_TOOL](https://www.waveshare.com/wiki/ESP32-S3-Touch-LCD-4.3#Resources), è possibile osservare i messaggi CAN inviati dall'ESP32-S3-Touch-LCD-4.3 all'avvio.

[](https://www.waveshare.com/wiki/File:ESP32-IDF-20.png)

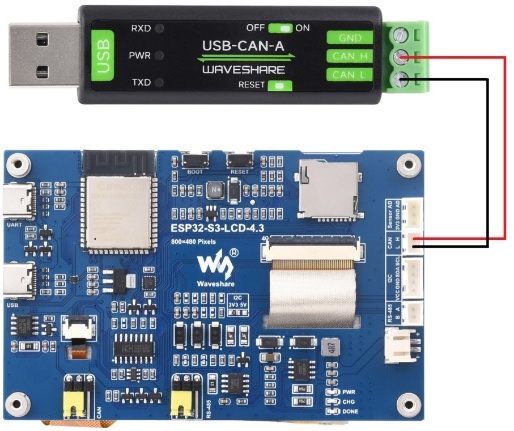
* Osservare ulteriormente USB-CAN-A\_TOOL e si vedranno i messaggi CAN inviati da ESP32-S3-Touch-LCD-4.3

[](https://www.waveshare.com/wiki/File:ESP32-Arduino-CN-1.png)

### 07\_TWAIreceive

#### Collegamento hardware

* Collegare la board al computer tramite un cavo USB.
* Collegare la scheda di sviluppo a [USB-CAN-A](https://www.waveshare.com/USB-CAN-A.htm), come mostrato in figura

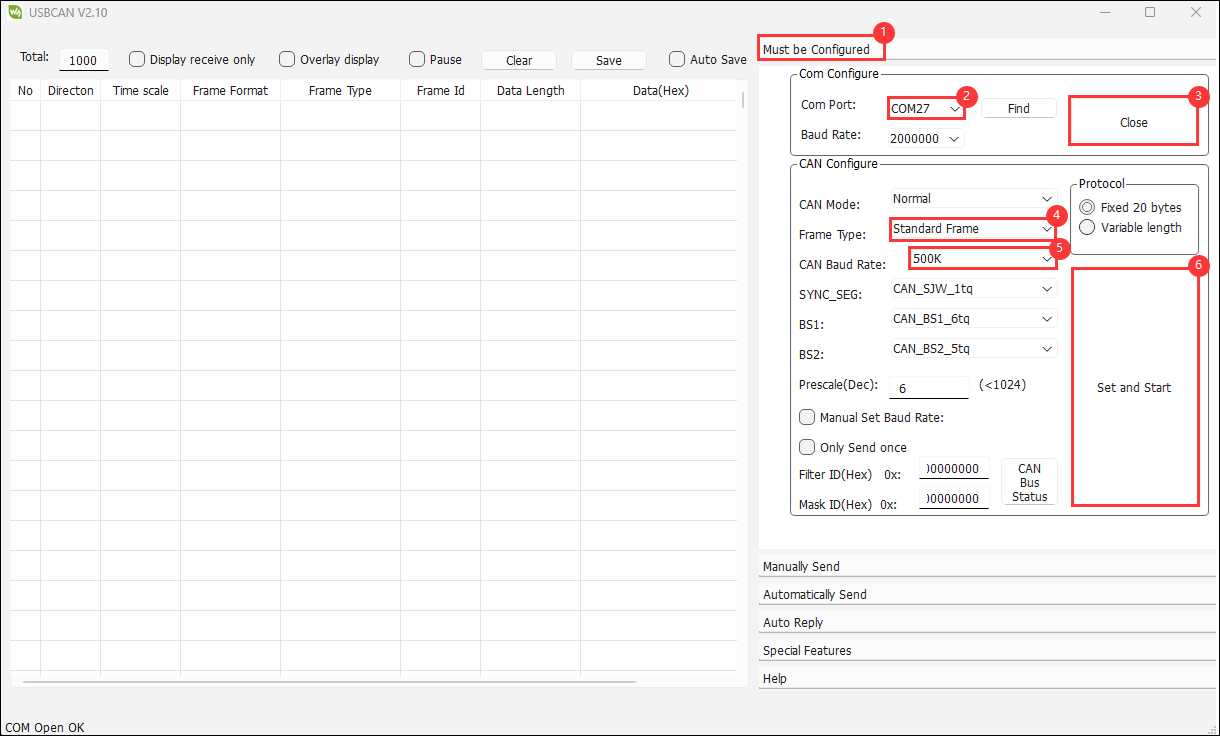
[](https://www.waveshare.com/wiki/File:ESP32-Arduino-44.png)

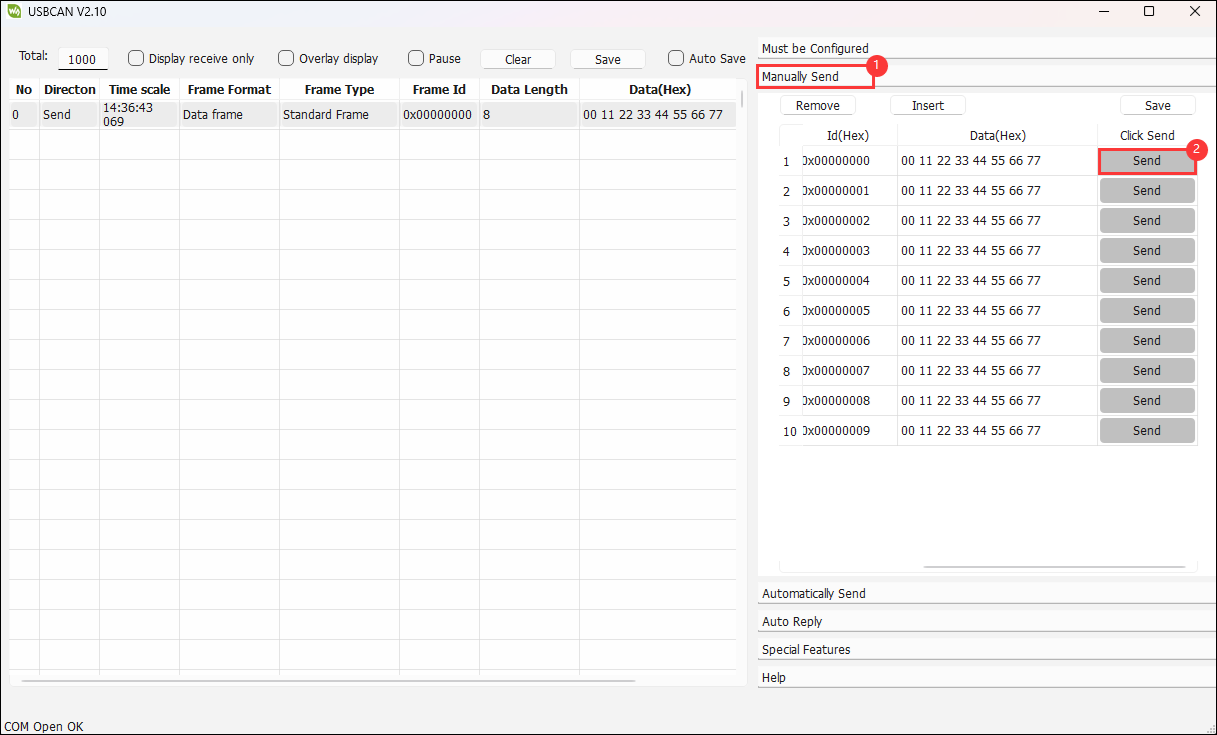
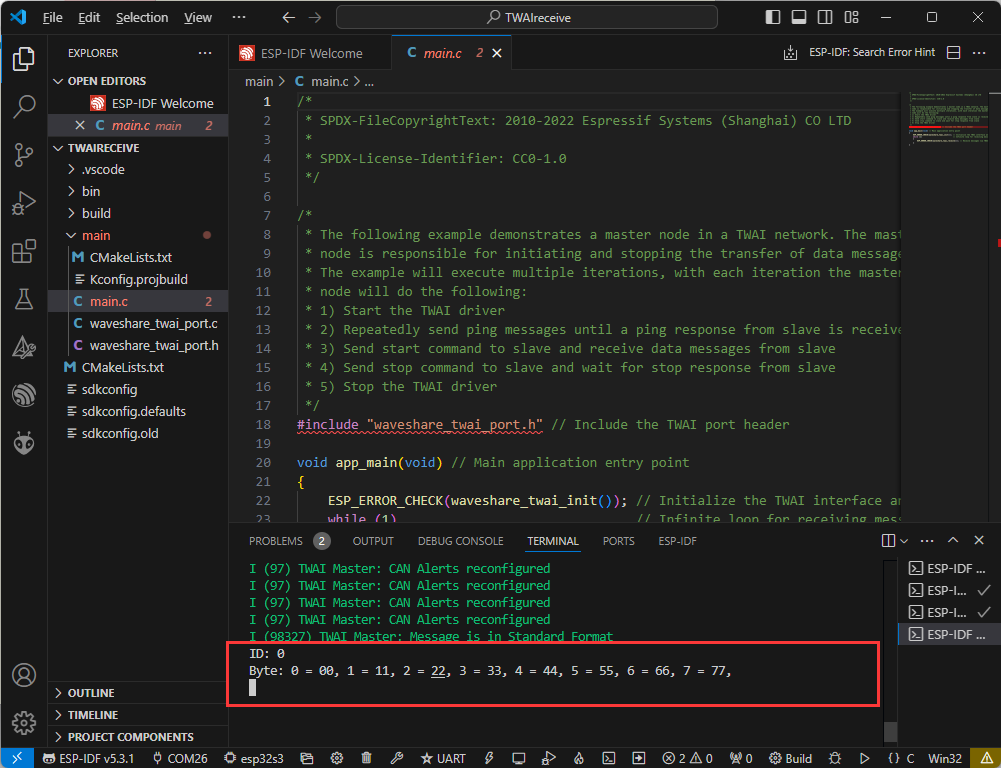
#### Analisi del codice

* **waveshare\_twai\_receive()**:
  + Se il driver non è installato, tornerà allo stato di errore dopo un certo periodo di tempo.
  + Legge gli "alert" attivati ​​e ottien informazioni sullo stato TWAI.
  + Stampa le informazioni di log corrispondenti in base ai diversi tipi di allarme attivati, inclusi allarmi passivi di errore, allarmi di errore del bus e allarmi di overflow della coda di ricezione, e stampa le relative informazioni di stato.
  + Se viene attivato un avviso di dati ricevuti, i messaggi vengono ricevuti in un ciclo e la funzione handle\_rx\_message viene chiamata per elaborare ciascun messaggio ricevuto. Infine, viene restituito lo stato di successo.

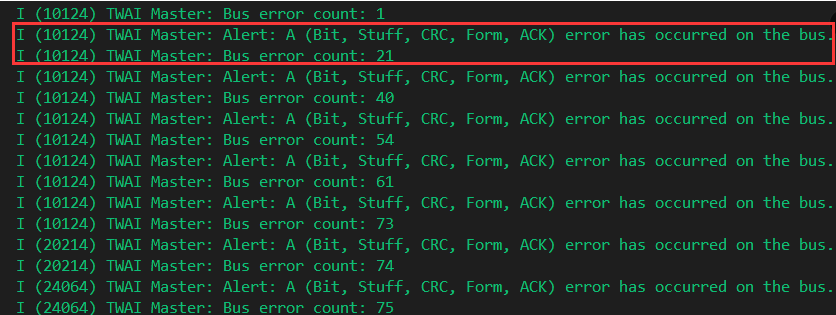
#### Dimostrazione dei risultati

* ESP32-S3-Touch-LCD-4.3 attende [USB-CAN-A\_TOOL](https://www.waveshare.com/wiki/ESP32-S3-Touch-LCD-4.3#Resources) per inviare un messaggio. Se il messaggio viene ricevuto correttamente, verrà stampato sulla porta seriale.

[](https://www.waveshare.com/wiki/File:ESP32-Arduino-CN-2.png)

[](https://www.waveshare.com/wiki/File:ESP32-Arduino-23.png)[](https://www.waveshare.com/wiki/File:ESP32-IDF-21.png)

* Se si verifica il seguente errore, cliccare sul monitor seriale e utilizzare lo strumento di debug per inviare nuovamente i dati. (Se si preme Reset, a volte è necessario cliccare nuovamente sul monitor seriale):

[](https://www.waveshare.com/wiki/File:ESP32-IDF-22.png)

### 08\_lvgl\_Porting

#### Collegamento hardware

* Collegare la board al computer tramite un cavo USB.

#### Analisi del codice

* **app\_main()**:
  + Inizializza il display LCD RGB Waveshare ESP32-S3, quindi si puà scegliere se accendere o spegnere la retroilluminazione dello schermo.
  + Quindi, visualizza un messaggio che indica che si vuole visualizzare il contenuto dimostrativo di LVGL. Poiché l'API LVGL non è thread-safe, il mutex viene bloccato per primo.
  + Si può quindi scegliere di eseguire diverse demo LVGL come lv\_demo\_stress, lv\_demo\_benchmark, lv\_demo\_music, lv\_demo\_widgets o example\_lvgl\_demo\_ui ecc.
  + Infine, rilascia il lock di mutua esclusione.

#### Modifica del codice

* In ESP\_Panel\_Board\_Custom.h è presente una definizione macro per selezionare se attivare o meno la funzione touch: il valore 0 corrisponde alla disattivazione del touch, mentre il valore 1 corrisponde all'attivazione del touch, e può essere selezionato in base al modello acquistato.

#define CONFIG\_EXAMPLE\_LCD\_TOUCH\_CONTROLLER\_GT911 0 // 1 avvia il touch, 0 lo chiude.

Dimostrazione dei risultati

* Dopo aver completato il flashing, premere il pulsante di reset per avviare la demo.

[**Screen lights up\_video**](https://www.youtube.com/watch?v=luICguq9cjo)

* Per aumentare ulteriormente il frame rate, è possibile fare riferimento a questo [link](https://docs.espressif.com/projects/esp-faq/en/latest/software-framework/peripherals/lcd.html#how-can-i-improve-the-display-frame-rate-of-lcd-screens) per la configurazione.
* Per i driver LCD RGB, fare riferimento a questo [link](https://docs.espressif.com/projects/esp-idf/en/release-v5.3/esp32s3/api-reference/peripherals/lcd/rgb_lcd.html)
* Per i driver GT911, fare riferimento a questo [link](https://github.com/espressif/esp-bsp/tree/master/components/lcd_touch/esp_lcd_touch_gt911)
* La versione di lvgl utilizzata è la 8.3 ed è possibile interrogare e utilizzare l'API LVGL tramite la seguente documentazione
  + [Documentazione LVGL](https://docs.lvgl.io/8.3/)

# Flash e Cancellazione del Firmware

* **La demo corrente fornisce un firmware di prova, utilizzabile per verificare il corretto funzionamento del dispositivo integrato, flashando direttamente il firmware di prova.**
* **Path del file bin:**

...\ESP32-S3-Touch-LCD-4.3-Demo\Firmware

[Flash e Cancellazione del Firmware](https://www.waveshare.com/wiki/Flash_Firmware_Flashing_and_Erasing) per riferimento

## Risorse

### Schema Elettrico

* [Schema](https://files.waveshare.com/wiki/ESP32-S3-Touch-LCD-4.3/manual/ESP32-S3-Touch-LCD-4.3-Sch.pdf)

### Diagramma del Progetto

* [Disegno 3D](https://files.waveshare.com/wiki/ESP32-S3-Touch-LCD-4.3/ESP32-S3-Touch-LCD-4in3_3D_Drawing.zip)

### Demo

* [Demo di ESP32-S3-Touch-LCD-4.3](https://files.waveshare.com/wiki/ESP32-S3-Touch-LCD-4.3/ESP32-S3-Touch-LCD-4.3-Demo.zip)

## Datasheet

### ESP32-S3

* [Datasheet di ESP32-S3](https://files.waveshare.com/wiki/common/Esp32-s3_datasheet_en.pdf)
* [Manuale tecnico di riferimento ESP32-S3](https://files.waveshare.com/wiki/common/Esp32-s3_technical_reference_manual_en.pdf)
* [datasheet di ESP32-S3-WROOM-1](https://files.waveshare.com/wiki/common/Esp32-s3-wroom-1_wroom-1u_datasheet_en.pdf)

### Altri Componenti

* [Datasheet di ST7262](https://files.waveshare.com/wiki/common/ST7262.pdf)
* [Datasheet di ST7262](https://files.waveshare.com/wiki/common/GT911_EN_Datasheet.pdf)
* [Datasheet di ST7262](https://files.waveshare.com/wiki/common/CH343DS1-en.pdf)
* [Datasheet di ST7262](https://files.waveshare.com/wiki/common/TJA1051.pdf)
* [Datasheet di ST7262](https://files.waveshare.com/wiki/common/CH422DS1_EN.pdf)

## Strumenti Software

### Arduino

* [Link ufficiale per il download dell'IDE Arduino](https://www.arduino.cc/en/software/)

### VScode

* [Sito web ufficiale di VScode](https://code.visualstudio.com/download)

### Strumento di debug

* [Assistente per il debug di porte seriali e reti SSCOM](https://files.waveshare.com/wiki/common/Sscom5.13.1.zip)
* [USB-CAN-A\_TOOL](https://files.waveshare.com/wiki/USB-CAN-A/Tool/USBCANV2.10.zip)

# Link ad Altre Risorse

* [Documentazione ufficiale di ESP32-Arduino](https://docs.espressif.com/projects/arduino-esp32/en/latest/index.html)
* [Risorse ufficiali di ESP32-Arduino](https://github.com/espressif/arduino-esp32)
* [Risorse ufficiali di ESP-IDF](https://github.com/espressif/esp-idf)
* [Documentazione ufficiale di LVGL](https://docs.lvgl.io/master/intro/introduction.html)

# FAQ

[Domanda: Perché la ricezione CAN dell'ESP32-S3-Touch-LCD-4.3 non funziona?](https://www.waveshare.com/wiki/ESP32-S3-Touch-LCD-4.3#accordion1)

**Risposta:**

* Riavviare la porta COM in UCANV2.0.exe e premere più volte il pulsante Reset dell'ESP32-S3-Touch-LCD-4.3.
* Rimuovere la spunta per "DTR" nell'assistente di debug della porta seriale.

[Domanda: L'ESP32-S3-Touch-LCD-4.3 non risponde dopo aver flashato una demo Arduino per la visualizzazione dello schermo RGB?](https://www.waveshare.com/wiki/ESP32-S3-Touch-LCD-4.3#accordion2)

**Risposta:**

Se non si verifica alcuna risposta sullo schermo dopo aver programmato il codice, verificare che le configurazioni corrette siano impostate in Arduino IDE -> Tools, selezionare la Flash corrispondente (16 MB) e abilitare PSRAM (8 MB OPI).

[Domanda: Il programma di flashing dello schermo RGB di Arduino ESP32-S3-Touch-LCD-4.3 non riesce a compilare e segnala un errore?](https://www.waveshare.com/wiki/ESP32-S3-Touch-LCD-4.3#accordion3)

**Risposta:**

Per verificare se la libreria è installata, fare riferimento alla [procedura di installazione delle librerie](https://www.waveshare.com/wiki/Arduino_Library_Manager_Tutorial)

[Domanda: Perché sono installate tutte le librerie e il programma di flashing LVGL segnala un errore di lv\_cong.h mancante?](https://www.waveshare.com/wiki/ESP32-S3-Touch-LCD-4.3#accordion4)

**Risposta:**

Il path per installare la libreria contiene caratteri cinesi, con conseguente impossibilità di trovare il file della libreria.

[Domanda: Perché lo schermo non viene visualizzato dopo aver flashato correttamente con Arduino?](https://www.waveshare.com/wiki/ESP32-S3-Touch-LCD-4.3#accordion5)

**Risposta:**

Per eseguire la demo a scopo di confronto, è possibile fare riferimento ai seguenti passaggi:

1. Prima di eseguire il programma, consultare il [tutorial sul library manager Arduino](https://www.waveshare.com/wiki/Arduino_Library_Manager_Tutorial) per l'installazione della libreria.
2. Per installare la libreria, consultarere il [video di riferimento](https://files.waveshare.com/wiki/common/ESP32-S3-7-lib.zip)
3. Eseguire e flashare la[Demo](https://www.waveshare.com/wiki/ESP32-S3-Touch-LCD-4.3#Resources)

[Domanda: Perché si verifica l'errore "fatal error: esp\_ memory\_ utils.h: No such file or directory" durante la compilazione di demo utilizzando l'IDE Arduino?](https://www.waveshare.com/wiki/ESP32-S3-Touch-LCD-4.3#accordion6)

**Risposta:**

Installare Arduino esp32 versione ≥v3.0.2, questo potrebbe risolvere il problema.

[Domanda: La comunicazione CAN può utilizzare strumenti di debug per inviare frame continui?](https://www.waveshare.com/wiki/ESP32-S3-Touch-LCD-4.3#accordion7)

**Risposta:**

Sì, è anche possibile personalizzare la frequenza dei frame consecutivi. Quando la frequenza è troppo alta e il computer si blocca, potrebbero verificarsi errori del bus.

[Domanda: Cosa devo fare se il flashing dell'ESP-IDF non riesce?](https://www.waveshare.com/wiki/ESP32-S3-Touch-LCD-4.3#accordion8)

**Risposta:**

1. È possibile impostare la scheda in modalità download per risolvere questo problema. Spegnere completamente la scheda, tenere premuto il pulsante di Boot e riaccenderla, quindi rilasciarlo, entrare in modalità download, rieseguire il flashing del programma, resettare ed eseguire.
2. Provare a premere il pulsante fullclean nella barra di stato e ricompilare il flashing. Questa funzione serve a pulire tutto il contenuto compilato, cliccando quando si verifica un errore di compilazione del progetto o altre operazioni che inquinano il contenuto compilato.

[Domanda: Cosa devo fare se non riesco a trovare la cartella AppData?](https://www.waveshare.com/wiki/ESP32-S3-Touch-LCD-4.3#accordion9)

**Risposta:**

Alcune cartelle AppData sono nascoste per default e possono essere impostate per essere visualizzate.

* + Sistema inglese: Explorer->View->Check "Hidden items"
  + Sistema cinese: File Explorer -> View -> Display -> Check "Hidden Items"

[Domanda: Come controllare la porta COM?](https://www.waveshare.com/wiki/ESP32-S3-Touch-LCD-4.3#accordion10)

**Risposta:**

Premere i tasti Windows + R per aprire la finestra di dialogo "Run", inserire devmgmt.msc e premere Enter per aprire Device Manager; espandere la sezione "Port (COM and LPT)", dove verranno elencate tutte le porte COM e il loro stato attuale.

[Domanda: Come risolvere il problema della lenta compilazione iniziale del programma?](https://www.waveshare.com/wiki/ESP32-S3-Touch-LCD-4.3#accordion11)

**Risposta:**

È normale che la prima compilazione sia lenta, basta avere pazienza.

[Domanda: Come risolvere il problema per cui il programma viene flashato correttamente ma non viene visualizzato nulla sul display LCD?](https://www.waveshare.com/wiki/ESP32-S3-Touch-LCD-4.3#accordion12)

**Risposta:**

Se sulla scheda di sviluppo è presente un pulsante di reset, premerlo; in caso contrario, riaccendere la scheda.

[Domanda: Perché il flashing del programma non riesce quando si utilizza un dispositivo MAC?](https://www.waveshare.com/wiki/ESP32-S3-Touch-LCD-4.3#accordion13)

**Risposta:**

Installare il [MAC Driver](https://files.waveshare.com/wiki/common/CH34XSER_MAC.7z) e flashare nuovamente.

[Domanda: Come utilizzare SquareLine Studio per progettare interfacce?](https://www.waveshare.com/wiki/ESP32-S3-Touch-LCD-4.3#accordion14)

**Risposta:**

* Consultare il [tutorial di SquareLine Studio](https://www.waveshare.com/wiki/Waveshare_SquareLine_Studio)

# Supporto

## Supporto Tecnico

Se c'è bisogno di supporto tecnico o si hanno feedback/recensioni, cliccare sul pulsante **Invia Ora** per inviare un ticket. Il nostro team di supporto verificherà la situazione e risponderà entro 1 o 2 giorni lavorativi. Preghiamo di avere pazienza: faremo tutto il possibile per aiutare a risolvere il problema.

Orario di lavoro: 9:00 - 18:00 GMT+8 (dal lunedì al venerdì)