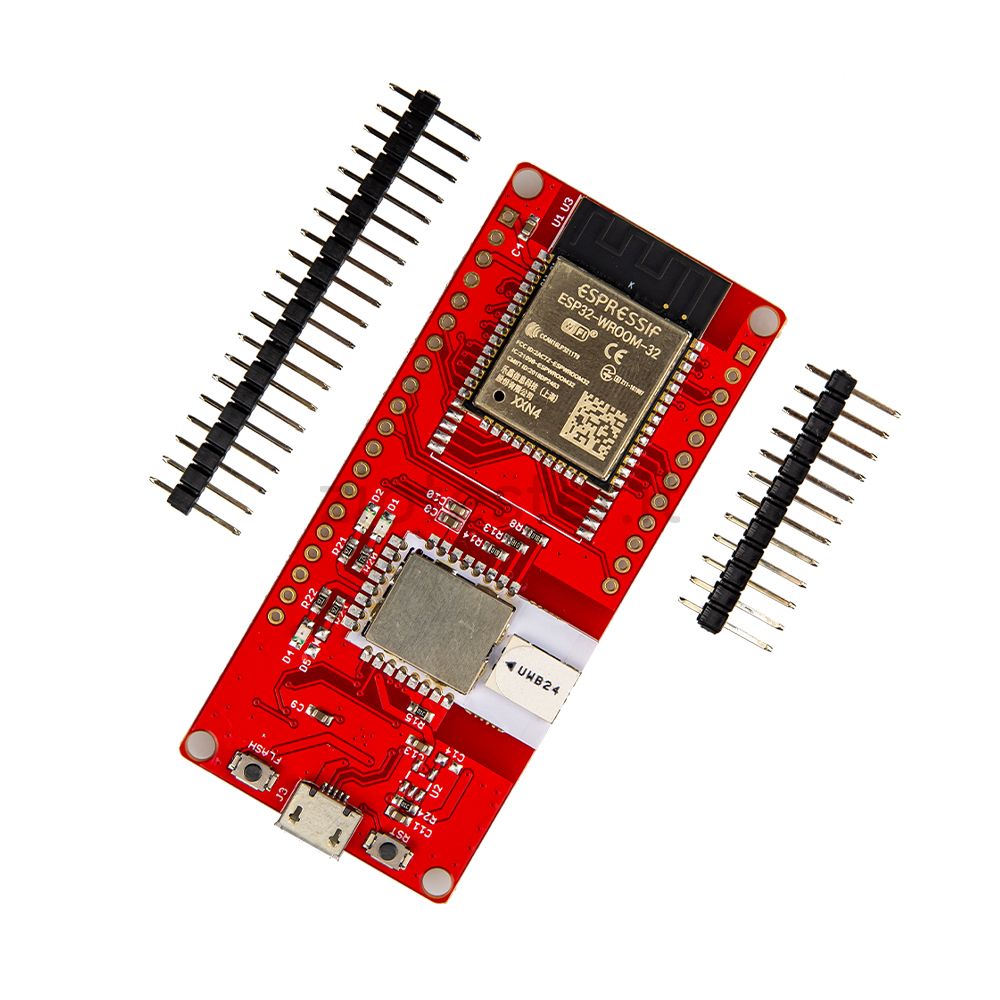


**ESP32 DW3000 UWB USER MANUAL**



**How to use, configure and control the DW3000 UWB devices.**

Sommario

[**Requisiti di sistema 3**](#_fh0c2afu68d4)

[**Installazione e configurazione 4**](#_q6hf07e8w7d5)

[**Configurazione del codice 5**](#_t5za5abi4lnp)

[Code for anchor 5](#_tqaatbj1mbwg)

[Spiegazione delle variabili e parametri nel codice Anchor scritto tramite Arduino IDE 5](#_gohoj6s5mupx)

[Code for tag 10](#_o0gygmbhkl2d)

[**Guida all’avvio del sistema 10**](#_yvat1tqe6097)

[**Risorse aggiuntive 11**](#_h61s5h9qoce3)

[**Architettura cloud 12**](#_xx2de4ymlhu8)

[Descrizione 12](#_t2k1zor6hubn)

[Utilizzo 13](#_2w8775nzxgy8)

[Introduzione 13](#_bv0vliwwu04c)

[API e Interazione con i Dispositivi 13](#_mlhxslx5auz4)

[Archiviazione dei Dati 13](#_x0rgj9bjo5eb)

[Trilaterazione e Aggiornamento delle Posizioni 13](#_kuo2n8ud3ox9)

[Comunicazione con il Front-End React 13](#_wnfmqw5cagz)

[Servizi 14](#_hmp6mjp3rilm)

[Codice 15](#_g9mr208msvkd)

[**Architettura UX 16**](#_rgr27gadd3c9)

[**Descrizione 16**](#_cfqwtepi9f2l)

[Funzionalità 17](#_9l9fy7yxvrim)

## **Requisiti di sistema**

I requisiti di sistema per l'utilizzo dei dispositivi DW3000 ESP32 UWB nel sistema RTLS si dividono in due categorie e sono i seguenti:

**Requisiti hardware:**

1. DW3000 ESP32 UWB Anchor:

* DW3000 ESP32 UWB Anchor deve essere dotato di un ESP32 come microcontrollore principale.
* Deve essere dotato di un modulo UWB (Ultra-Wide Band) DW3000 per consentire la comunicazione e la localizzazione precisa.
* L'anchor dovrebbe avere un'interfaccia di alimentazione stabile e sufficiente per il corretto funzionamento.

1. DW3000 ESP32 UWB Tag:

* DW3000 ESP32 UWB Tag deve essere dotato di un ESP32 come microcontrollore principale.
* Deve essere dotato di un modulo UWB DW3000 per consentire la comunicazione e la ricezione dei segnali dalle ancore.

1. Alimentazione:

* Le ancore e il tag devono essere alimentati con una fonte di alimentazione stabile e adeguata, come una batteria o un alimentatore.
* Assicurarsi che l'alimentazione sia compatibile con i requisiti di tensione e corrente dei dispositivi DW3000 ESP32 UWB.

**Requisiti software:**

1. Arduino IDE:

* Assicurarsi di avere l'ultima versione di Arduino IDE installata sul computer.
* Configurare correttamente Arduino IDE per lavorare con la scheda ESP32.

1. Librerie e driver:

* Installare le librerie necessarie per il corretto funzionamento dei dispositivi DW3000 ESP32 UWB nell'ambiente di sviluppo Arduino IDE.
* Verificare se è necessario installare driver aggiuntivi per consentire la comunicazione tra i dispositivi e il computer.

1. Sistema operativo e connettività:

* Assicurarsi di avere un computer con un sistema operativo compatibile (Windows, macOS, Linux) e le relative versioni supportate da Arduino IDE.
* Disporre di una porta USB funzionante o di un'interfaccia seriale per collegare le ancore e il tag al computer.

## **Installazione e configurazione**

Per permettere la configurazione del modulo ESP32 DW3000 UWB e garantire la corretta ricezione e trasmissione dei segnali di localizzazione tramite tecnologia Ultra Wide Band, è necessario seguire i seguenti passaggi:

1. Scaricare e installare il driver USB di tipo CP210x VCP, in base al sistema operativo in utilizzo.
2. Collegare il modulo DW3000 al PC utilizzando un cavo micro USB.
3. Aggiungere un url alla lista di URL dei repository delle schede installabili nell'IDE di Arduino. In particolare, l'URL “<https://dl.espressif.com/dl/package_esp32_index.json>?” permette di installare il supporto alla scheda ESP32 nel proprio ambiente di sviluppo Arduino IDE.
4. Nel menu “*Boards Manager*” cercare “*esp32*” e installare la board per renderla accessibile.
5. Selezionare la scheda Arduino corretta dal menu Tools -> Board -> ESP32 Dev Module.
6. Eseguire il comando "*pip install pyserial*" che installa il modulo PySerial per Python. PySerial è un modulo che fornisce supporto per la comunicazione seriale tra dispositivi, inclusi Arduino e Raspberry Pi, utilizzando Python. È utilizzato per creare applicazioni che richiedono l'invio e la ricezione di dati su un'interfaccia seriale.
7. Aggiungere i permessi per abilitare la scelta della porta seriale.
8. Selezionare la porta seriale corretta dal menu Tools -> Port.
9. Installare la libreria “DW3000” da Github (<https://github.com/Makerfabs/Makerfabs-ESP32-UWB-DW3000>) come zip e inserirla nell’IDE di Arduino tramite Sketch -> Include Library.
10. Installare la libreria “EEPROM-Storage” dal menu “Library Manager”.
11. Caricare il codice disponibile nella seguente documentazione, sia per tag che per ancore, che permette la comunicazione tra due moduli DW3000, nonché l’invio e la ricezione del segnale per calcolare la distanza di un dispositivo dall’altro.
12. Eseguire il comando “Upload” di Arduino IDE che permette di caricare il codice sorgente scritto in linguaggio di programmazione Arduino nel microcontrollore della scheda ESP32, che eseguirà poi il programma. Ciò consente quindi di trasferire il programma dall'IDEA alla scheda attraverso il cablaggio USB e di far eseguire questo programma dalla scheda.

**N.B.** Quando si esegue il caricamento del programma sul modulo potrebbe essere necessario premere il pulsante **“FLASH”** per forzare il modulo in modalità di caricamento (se si possiede un sistema operativo di tipo macOS).

## 

## **Configurazione del codice**

Di seguito viene fornito il codice da caricare sui dispositivi, rispettivamente per il tipo ANCORA e TAG.

### Code for anchor

Lo sketch caricato sul primo dispositivo IoT (utilizzato come ancora) è per effettuare il "ranging" tra dispositivi dotati di chip DW3000, ovvero determinare la distanza tra di essi in base alla modalità di comunicazione radio utilizzata (UWB). Il programma configura i pin di connessione, imposta le configurazioni di comunicazione e temporizzazione, attiva la ricezione e la trasmissione di frame radio e rileva gli eventi di trasmissione e reception, calcolando infine la distanza stimata tra lo stesso e il dispositivo con cui comunica (tag). In particolare, il codice presente nello sketch si riferisce ad un "SS TWR RESP" (Single-Sided Two-Way Ranging Responder) che risponde alle richieste di ranging inviate da un "SS TWR INIT" (Single-Sided Two-Way Ranging Initiator).

### Spiegazione delle variabili e parametri nel codice Anchor scritto tramite Arduino IDE

* Include la libreria `WiFi.h` all'inizio dello sketch per utilizzare la connessione Wi-Fi , la libreria ‘dw3000.h’ che consente di interagire con il modulo DW3000 ESP32 e utilizzare le sue funzionalità, la libreria ‘HTTPClient.h’ per effettuare richieste HTTP su una rete, la libreria ‘ArduinoJson.h’ che offre supporto per la manipolazione dati in formato JSON e la libreria WiFiManager.h‘’ per la gestione della connessione WiFi.

#include <dw3000.h>

#include <WiFi.h>

#include <HTTPClient.h>

#include <ArduinoJson.h>

#include <WiFiManager.h>

* Le due variabili rx\_poll\_msg e tx\_resp\_msg sono array di byte utilizzati per la gestione di messaggi tra i dispositivi. ‘rx\_poll\_msg ’ è il messaggio di richiesta inviato mentre ‘tx\_resp\_msg ’ è il messaggio di risposta atteso.

static uint8\_t rx\_poll\_msg[] = { 0x41, 0x88, 0, 0xCA, 0xDE, 'W', 'A', 'V', 'E', 0xE0, 0, 0 };

static uint8\_t tx\_resp\_msg[] = { 0x00, 0x88, 0, 0xCA, 0x00, 0x00, 0x00, 0x00, 0x00, 0xE1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 };

* Viene avviato sulla scheda un access point WiFi che consente all’utente di connettersi attraverso le credenziali (ssid: mac address dispositivo, password: dw3000esp32) e successivamente configurare il router che dovrà consentire ai dispositivi di connettersi ad Internet ed effettuare le chiamate HTTP.

WiFiManager wfm;

wfm.setDebugOutput(false);

wfm.resetSettings();

wfm.autoConnect(macRequest.c\_str(), "dw3000esp32");

* Viene effettuata una richiesta HTTP POST all'endpoint specificato per memorizzare il dispositivo nel database Cosmos DB. Il corpo della richiesta è una stringa di dati JSON contenente le chiavi 'mac Address', ‘type’ e ‘connected’.

http.begin("https://rtlswizardsfunctions.azurewebsites.net/api/anchor?");

http.addHeader("Content-Type", "application/json");

String data = "{\"macAddress\":\"" + String(macRequest) + "\",\"type\":\"anchor\",\"connected\":true}";

int response\_code\_add = http.POST(data);

if (response\_code\_add > 0) {

Serial.printf("Risposta http: %d\n", response\_code\_add);

String response = http.getString();

Serial.println(response);

} else {

Serial.println("Error in the http request");

Serial.println(response\_code\_add);

}

http.end();

### 





### Code for tag

Lo sketch caricato sul secondo dispositivo IoT (utilizzato come tag) permette di calcolare il TEMPO DI VOLO (TOF) del segnale e, quindi, la distanza sulla base dei timestamp e del ritardo di propagazione del segnale. Il codice definisce le impostazioni di comunicazione (carrier frequency, data rate, lunghezza del preambolo, ecc.), i parametri di ritardo per la sincronizzazione tra la trasmissione e la ricezione, i parametri della risposta attesa e i tempi di attesa tra le misurazioni di ranging. Inoltre gestisce la comunicazione tramite SPI e la visualizzazione dei dati sulla seriale. Il codice è il seguente:

## **Guida all’avvio del sistema**

Per avviare il sistema RTLS, bisogna seguire i seguenti passaggi:

1. **Accensione delle ancore**: Collegare le ancore a una fonte di alimentazione e assicurarsi che siano correttamente connesse alla rete elettrica. Posizionare le stesse nell'area che si desidera tracciare, considerando la copertura e la distribuzione ottimale per ottenere risultati precisi.

2. **Misurazione delle dimensioni e posizioni**: Effettuare una misurazione accurata delle dimensioni del luogo in cui verrà utilizzato il sistema RTLS. Questo consentirà di configurare correttamente l'area di tracciamento. Inoltre, registrare le posizioni esatte delle ancore nell'area.

3. **Configurazione da front end**: Accedere all'interfaccia utente front end del sistema RTLS e avviare la fase di setup. In questa fase, si dovranno fornire le informazioni sulle dimensioni del luogo e inserire le posizioni delle ancore precedentemente misurate. Seguire attentamente le istruzioni fornite nell'interfaccia per completare correttamente la configurazione.

4. **Accensione del tag**: Una volta completata la configurazione da front end, è il momento di accendere il tag. Assicurarsi che il tag sia completamente carico e pronto all'uso. Premere il pulsante di accensione o seguire le istruzioni specifiche del dispositivo per avviarlo. Il tag inizierà a inviare i segnali e a rilevare la posizione all'interno dell'area tracciata.

Una volta completati questi passaggi, il sistema RTLS sarà avviato e pronto per il tracciamento dei movimenti all'interno dell'area specifica. È possibile visualizzare i risultati sul front end, che mostrerà la mappa con il tracciamento del tag in tempo reale e fornirà ulteriori funzionalità come la visualizzazione dello storico e informazioni dettagliate sui dispositivi.

## **Risorse aggiuntive**

Le risorse a cui si è fatto riferimento per la realizzazione del sistema RTLS si dividono nelle seguenti categorie:

1. Documentazione tecnica

* <https://www.qorvo.com/products/d/da008154>
* <https://www.makerfabs.com/desfile/files/esp32-wroom-32_datasheet_en.pdf>

1. Risorse online (tutorial e video)

* <https://www.makerfabs.com/esp32-uwb-dw3000.html>
* <https://www.youtube.com/watch?v=KVPQaBgxUQg&pp=ugMICgJpdBABGAHKBQxlc3AzMiBkdzMwMDA%3D>

1. Esempi di codice

* <https://github.com/Makerfabs/Makerfabs-ESP32-UWB-DW3000>
* <https://randomnerdtutorials.com/esp32-http-get-post-arduino/>

1. Guide di riferimento

* <http://didawiki.cli.di.unipi.it/lib/exe/fetch.php/rhs/localizzazione.pdf>
* <https://how2electronics.com/ranging-localization-with-esp32-uwb-dw3000-module/#Anchor_Tag_Hardware_Setup>

## 

## 

## 

## 

## **Architettura cloud**

### Descrizione

Il cloud, in generale, si riferisce a un modello di distribuzione di servizi e risorse informatiche attraverso Internet. Esso consente di accedere a risorse, come server, archiviazione, database e software, attraverso una rete remota di server gestiti da fornitori di servizi cloud.

Nel contesto di un sistema di Localizzazione e Tracciamento in Tempo Reale (RTLS - Real-Time Location System), il cloud svolge un ruolo importante nella gestione e nell'elaborazione dei dati di localizzazione. Un sistema RTLS coinvolge l'utilizzo di dispositivi di localizzazione, come tag e ancore, che rilevano e trasmettono le informazioni di posizione. Questi dati vengono quindi inviati al cloud per l'elaborazione e l'analisi.

Il cloud offre diversi vantaggi nel contesto di un sistema RTLS, ovvero:

1. **Scalabilità**: Il cloud consente di scalare le risorse di elaborazione e archiviazione in base alle esigenze del sistema RTLS. È possibile aumentare o ridurre le risorse in modo flessibile in base alla quantità di dati di localizzazione da gestire.
2. **Affidabilità**: I fornitori di servizi cloud spesso offrono servizi affidabili con garanzie di uptime elevate. Ciò assicura che il sistema RTLS sia sempre disponibile e in grado di gestire i dati di localizzazione in tempo reale.
3. **Elaborazione** **dati**: Nel cloud, è possibile utilizzare potenti servizi di elaborazione dati per analizzare i dati di localizzazione e ottenere informazioni utili.
4. **Archiviazione** **dati**: Il cloud offre spazio di archiviazione illimitato per i dati di localizzazione. I dati possono essere archiviati a lungo termine per scopi storici, analitici o di conformità.
5. **Accessibilità**: Con il cloud, è possibile accedere ai dati di localizzazione e al sistema RTLS da qualsiasi luogo e in qualsiasi momento, utilizzando dispositivi connessi a Internet. Ciò consente una gestione centralizzata e un monitoraggio in tempo reale.
6. **Integrazione**: Il cloud facilita l'integrazione con altre applicazioni e servizi. Ad esempio, i dati di localizzazione possono essere integrati con sistemi di gestione aziendale, piattaforme di analisi dati o applicazioni mobili.

### 

### 

### Utilizzo

#### Introduzione

Azure è stata utilizzata come piattaforma cloud per implementare le funzioni serverless che forniscono le API per il sistema Real-Time Location System (RTLS). Queste funzioni consentono ai dispositivi DW3000 ESP32 UWB di comunicare con il front end React e di inviare i dati delle distanze calcolate.

#### API e Interazione con i Dispositivi

Le funzioni serverless espongono le API necessarie per il sistema RTLS. Utilizzando il metodo HTTP POST, i dispositivi possono chiamare le operazioni di creazione dell'istanza nella tabella del database Cosmos. Durante il processo di setup, i dispositivi inviano il proprio indirizzo MAC come parte dei dati inviati.

#### Archiviazione dei Dati

I dati delle distanze calcolate vengono archiviati nel database Cosmos. Dopo che i moduli DW3000 hanno completato il setup e inviato le distanze, i dati vengono salvati nel database Cosmos per l'elaborazione successiva.

#### Trilaterazione e Aggiornamento delle Posizioni

È stata implementata una funzione di trigger basata su timer su Azure. Questa funzione viene eseguita periodicamente ad intervalli di tempo specificati. Utilizzando la formula di trilaterazione, la funzione calcola e salva le posizioni basate sulle distanze registrate dalle ancore. L'operazione di calcolo delle posizioni viene eseguita ad intervalli regolari per mantenere le posizioni aggiornate in modo continuo.

#### Comunicazione con il Front-End React

Il front end React comunica con le API esposte dalle funzioni serverless su Azure utilizzando richieste HTTP. In particolare, vengono utilizzate richieste GET per recuperare i dati dei dispositivi. Questi dati vengono utilizzati per visualizzare la mappa e generare il grafico di tracciamento dei movimenti (spaghetti chart).

Questa architettura cloud basata su Azure permette una comunicazione affidabile tra il front end React e i dispositivi hardware, consentendo di visualizzare in tempo reale la posizione dei tag e generare rappresentazioni grafiche intuitive dei movimenti.

### 

### Servizi

I servizi utilizzati per quanto riguarda il cloud sono:

* **Azure**: una piattaforma cloud che fornisce una vasta gamma di servizi e risorse per la creazione, il deployment e la gestione di applicazioni e servizi scalabili.
* **Functions**: un servizio serverless che consente di eseguire codice senza preoccuparsi dell'infrastruttura sottostante, ideali per implementare logiche di business e esporre API.
* **Cosmos** **DB**: un database distribuito globale e multi-modello di Azure, che offre scalabilità, ridondanza e prestazioni elevate, adatto per archiviare e gestire grandi quantità di dati in modo efficiente.

### 

### 

### 

### 

### 

### 

### 

### 

### 

### 

### 

### 

### 

### Codice

Questo codice rappresenta una Azure Function chiamata "**AddDistance**" che viene attivata da una richiesta HTTP POST all'endpoint "/distance". La funzione riceve i dati di distanza inviati nel corpo della richiesta e li deserializza in un oggetto di tipo "**DistanceRequestModel**". Successivamente, verifica la presenza dei dispositivi di tag e ancoraggio nel database Cosmos tramite le chiamate ai metodi "**GetDeviceByMac**" dell'oggetto "**TagHttp**". Se i dispositivi esistono, crea un oggetto "**DistanceModel**" con i dati di misurazione, tag e ancoraggio e lo salva nel container "**distanceContainer**" del database Cosmos. Se tutto va bene, restituisce un risultato positivo con l'oggetto "distance" appena creato, altrimenti restituisce un errore con un messaggio appropriato.



## **Architettura UX**

## Descrizione

L'interfaccia utente (UI) è stata sviluppata utilizzando la libreria JavaScript React, al fine di garantire un'implementazione più precisa del lavoro è stato impiegato TypeScript.

Durante l'avvio dell'applicazione, l'utente ha la possibilità di selezionare la sede a cui desidera collegarsi. Successivamente, viene richiesto se desidera eseguire la configurazione dei dispositivi, e in caso di accettazione, viene guidato attraverso una fase iniziale di impostazione. In questa fase, all'utente viene richiesto di eseguire innanzitutto la configurazione manuale dei dispositivi non ancora inizializzati. Si presuppone che questi abbiano un'alimentazione elettrica, in seguito connettersi al punto di accesso da essi creato e fornire le credenziali necessarie per la connessione a una rete Wi-Fi.Successivamente, attraverso l'interfaccia, l'utente può selezionare i dispositivi associati a tale sede, nonché dispositivi non associati a nessun edificio ma attivi per poterli configurare. È possibile procedere con la configurazione delle ancore e delle rispettive posizioni. Una volta completata l'impostazione, l'utente viene riportato all'interfaccia di base.

Successivamente, l'utente viene indirizzato alla pagina della mappa, dove può visualizzare i dispositivi rappresentati, inclusi i tag e le ancore. Inoltre, è possibile visualizzare il tracciamento del tag che si muove sulla mappa, il quale viene rappresentato tramite un grafico a linee che illustra il percorso seguito dal tag nel tempo. È disponibile anche una legenda per fornire all'utente informazioni sui dati presenti nella mappa.

Nella pagina "Devices", l'utente può osservare due griglie, denominate rispettivamente "tags" e "anchors", che mostrano l'identificativo dei dispositivi insieme alla loro posizione. Da qui è possibile accedere alla pagina di dettaglio specifica del dispositivo. Le due pagine di dettaglio dei dispositivi sono simili, tuttavia la pagina relativa alle ancore consente la modifica della posizione e la possibilità di dissociare il dispositivo dalla sede, rendendolo disponibile agli altri dispositivi.

La pagina "Devices" include anche un pulsante "Start/Stop" che consente all'utente di avviare o interrompere la registrazione del tracciamento.

### Funzionalità

### 

### 