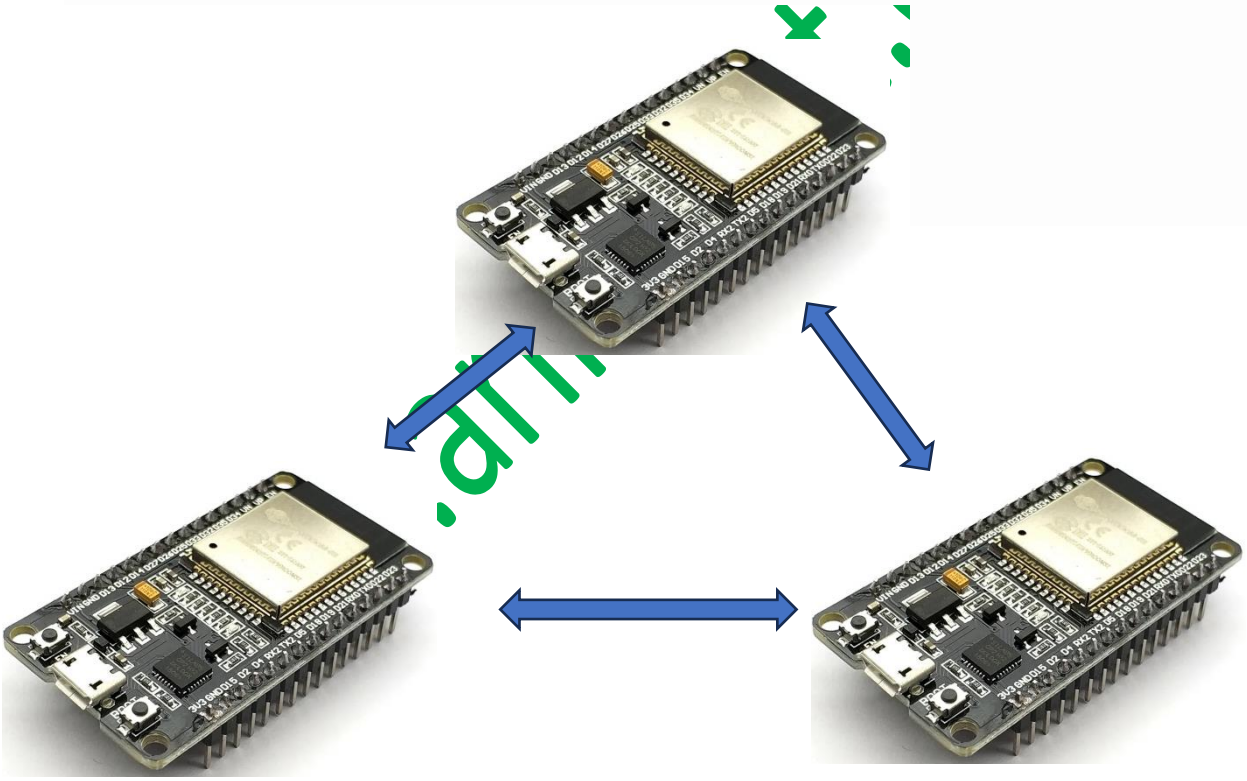
תקשורת ESP-NOW WiFi

א. מבוא

- ESP-NOW הוא פרוטוקול תקשורת אלחוטית שפותח על ידי Espressif Systems עבור רכיבי ה-Wi-Fi שלה.
- הוא מאפשר למכשירים לתקשר ישירות זה עם זה ללא צורך בנקודת גישה ל- Wi-Fi.
- פרוטוקול זה מבוסס על תקן Wi-Fi 802.11, אך הוא משתמש במערך רשת שונה מזה של Wi-Fi מסורתי.
- כמות הנתונים המרבי המותר בחבילה הוא 250 בתים. עם זאת, הוא מתקשר מהר יותר עם גודל חבילה קטן.
- אם יש צורך להעביר כמות גדולה יותר של נתונים, השימוש בפרוטוקול זה אינו שימושי.
- ESP-NOW פועל בתחום התדרים 2.4 GHz זהה ל- Wi-Fi, אך הוא אינו צריך להתחבר לחיבור רשת ה- WiFi.
- המרחק שאליו יכולה להגיע התקשורת הוא עד כ 220 מטר בשטח פתוח ללא הפרעות.

האיור הבא מתאר תקשורת : ESP NOW

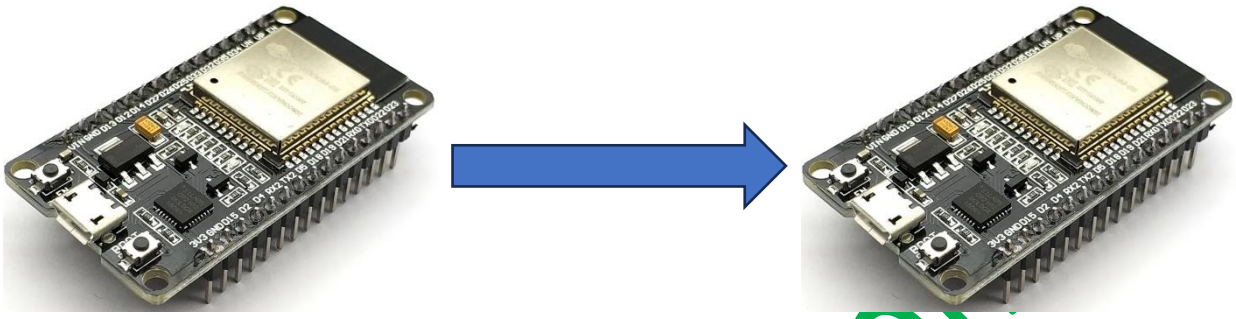


איור 1 : תקשורת ESP NOW בין 3 כרטיסי ESP

ישנן מספר תצורות (קונפיגורציות) לחבר בין כרטיסי ESP . נתאר את האפשרויות השונות:

ב. תקשורת חד כיוונית ESP-NOW בין כרטיס אחד השולח נתונים לכרטיס אחר

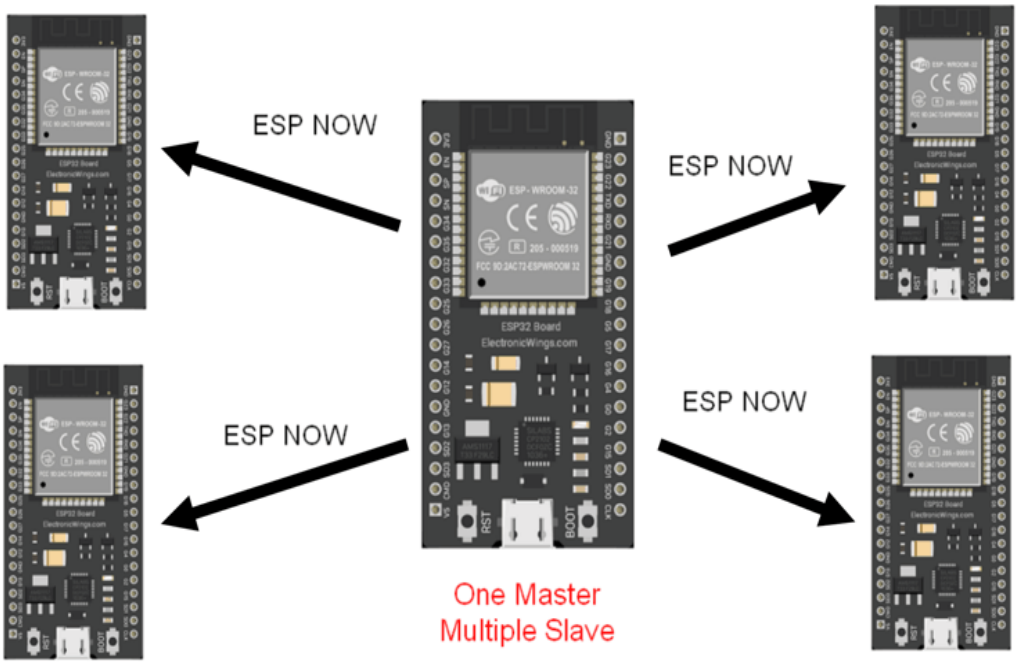
זוהי תצורה פשוטה להגדרה והיא אידיאלית לשליחת נתונים בין לוחות, כגון קריאות חיישנים או פקודות להפעלה או כיבוי של GPIO . האיור הבא מראה תצורה של תקשורת כזו :



איור 2 : תקשורת ESP-Now חד כיוונית

ג. תקשורת ESP-NOW בין כרטיס MASTER השולח נתונים למספר SLAVES

לוח ESP32 יחיד יכול לשלוח פקודות זהות או שונות לכרטיסי ESP32 מרובים. תצורה זו אידיאלית לבניית שלט רחוק, שבו לוח ראשי יחיד יכול לשלוט במספר מכשירים ברחבי הבית. האיור הבא מתאר צורת תקשורת כזו:



איור 3 : תקשורת בין כרטיס אחד למספר כרטיסים

ד. תקשורת ESP-NOW בין מספר כרטיסים אל כרטיס אחד

הגדרה זו אידיאלית לאיסוף נתונים ממספר חיישנים והצגתם בשרת אינטרנט יחיד. לדוגמה, נוכל להשתמש בזה כדי ליצור מערכת ניטור ביתית שאוספת נתונים מחיישני טמפרטורה, לחות ותנועה .

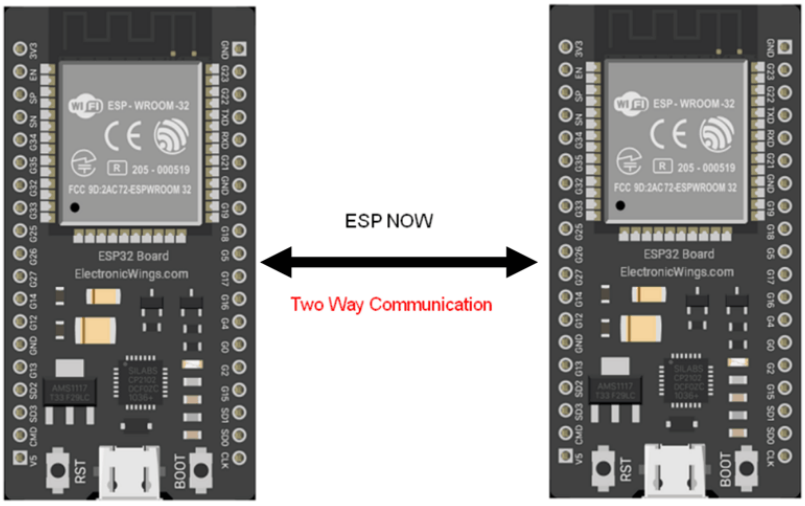
האיור הבא מתאר צורת חיבור כזו:

איור 4 : חיבור מספר SLAVES השולחים נתונים אל MASTER

ה. תקשורת ESP-NOW דו כיוונית

באמצעות ESP-NOW כל לוח יכול לשדר ולקבל את הנתונים בו זמנית. כלומר ליצור תקשורת דו-כיוונית בין הכרטיסים.

האיור הבא מתאר צורת חיבור כזו:

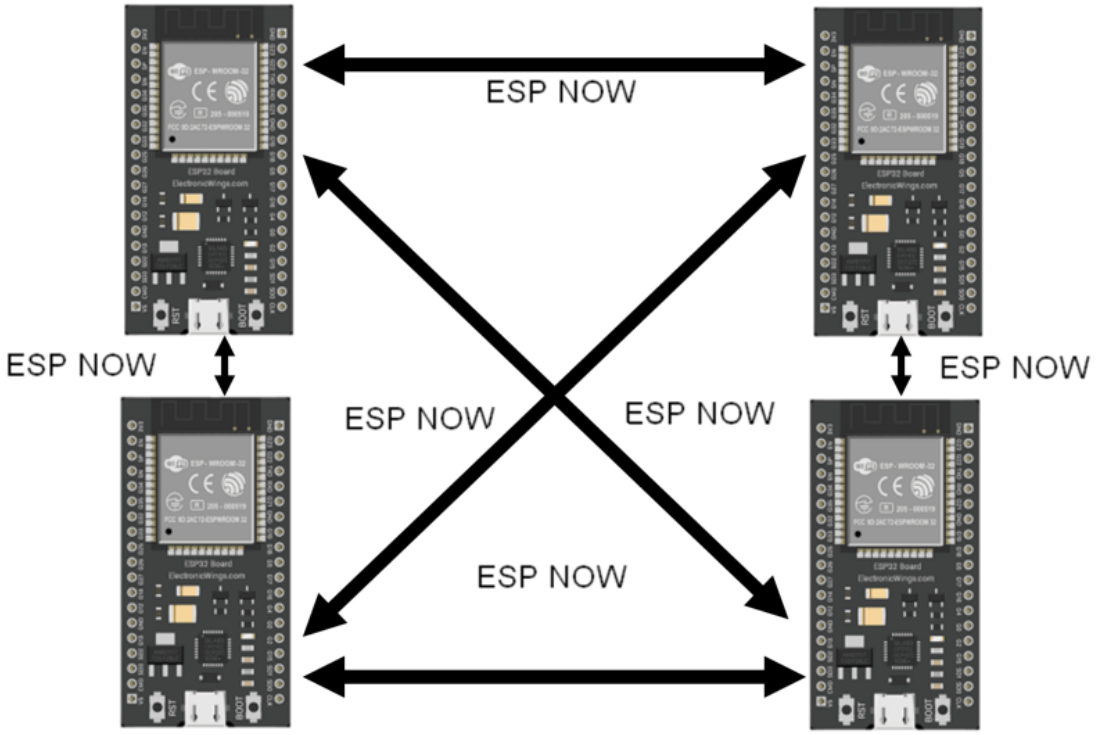


איור 5 : תקשורת דו כיוונית בין 2 כרטיסים

ו. תקשורת ESP-NOW דו כיוונית מרובה (בין יותר מ 2 כרטיסים).

ניתן גם להוסיף לוחות ESP32 נוספים לתצורה זו, אשר תיצור רשת של התקנים כמו שרואים באיור הבא :

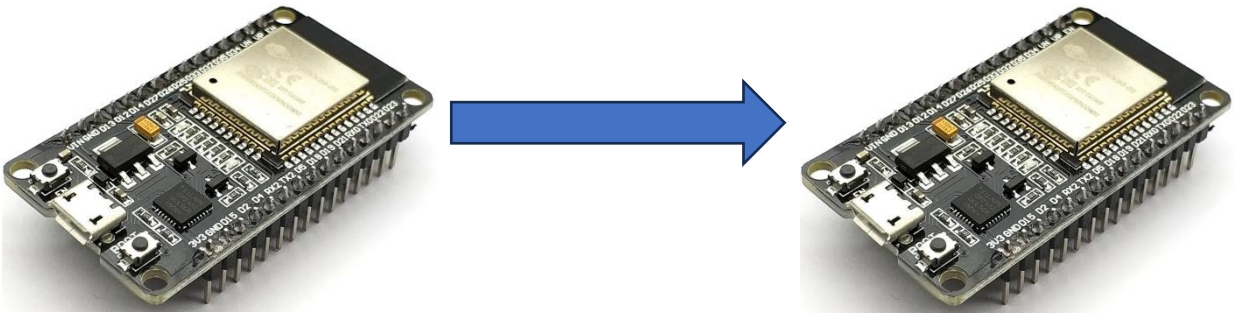
.



איור 6 : תקשורת דו כיוונית מרובה .

ז. תקשורת חד כיוונית ביתר פירוט

האיור הבא מתאר שוב את תצורת התקשורת החד כיוונית בין 2 כרטיסים:



איור 7 : תקשורת חד כיוונית בין 2 כרטיסי ESP

נראה תוכנית דוגמה של ESP-Now בתקשורת חד כיוונית. תחילה, עלינו לדעת מי המסטר ומי ה SLAVE מבין 2 הלוחות ואז לדעת את כתובת ה MAC של ה ESP32 ה SLAVE. כתובת **MAC** - **M**edia **A**ccess **C**ontrol address - כתובת בקרת גישה למדיה - היא מזהה ייחודי המוטבע על כל רכיב תקשורת נתונים בעת הייצור שלו .

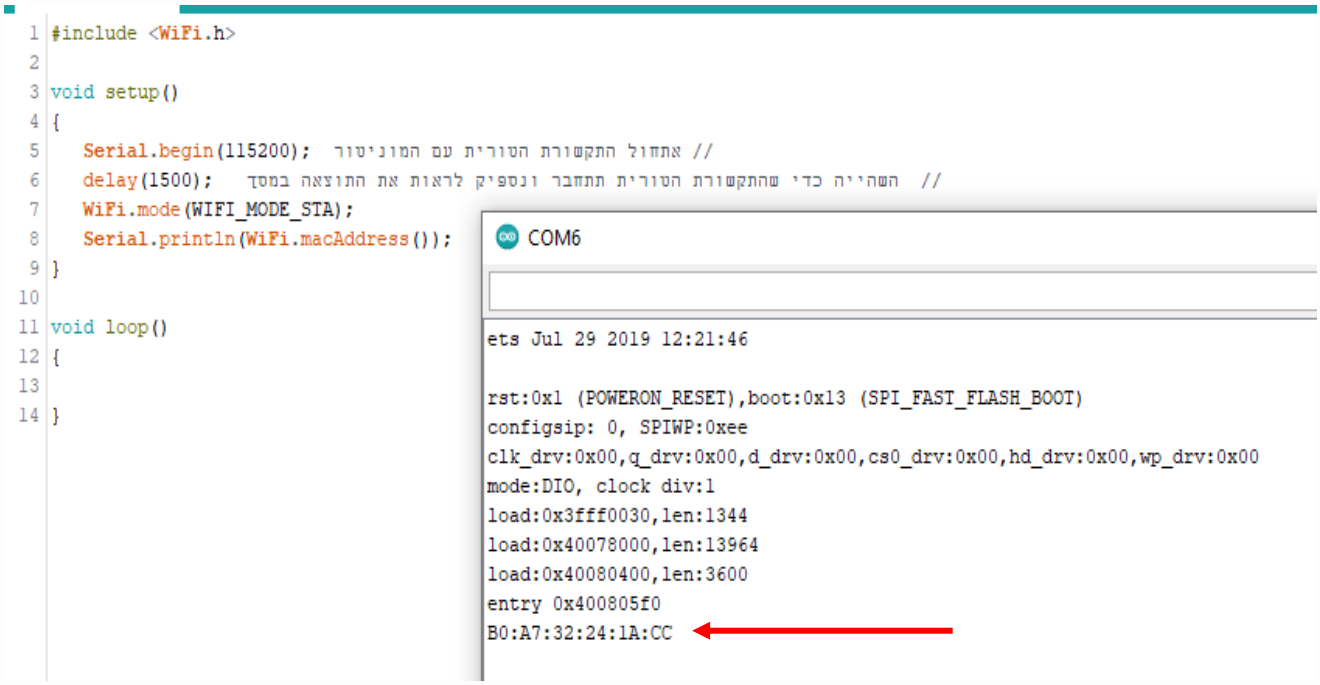
אם התוכנית לא עובדת ומקבלים כתובת 00:00:00:00:00:00 יש לרשום את התוכנית שנמצאת אחרי התוכנית הבאה.

ז.1 נרשום תוכנית לקבלת כתובת MAC של ה SLAVE :

```
#include <WiFi.h>

void setup()
{
  Serial.begin(115200);  // אתחול התקשורת הטורית עם המוניטור
  delay(1500);
  WiFi.mode(WIFI_MODE_STA);
  Serial.println(WiFi.macAddress());
}
void loop()
{
}
```

אחרי העלאת התוכנית נלחץ על מפסק EN והתוצאה שנקבל עבור ה ESP32 שלי שישמש כמקלט - Receiver (בכל ESP32 תוצאה אחרת !!) :



איור 8 : כתובת ה MAC ב ESP32 שלי

בחלק מכרטיסי ה ESP32 שקניתי קיבלתי את כתובת ה MAC כ 00:00:00:00:00:00 וזה כמובן **לא תקין**. יש לזה כמה סיבות אפשריות – חלקן פשוטות מאוד לפתרון.

ז.1.א נתחיל עם האפשרות שה WiFi **לא מאותחל כראוי לפני בקשת כתובת ה MAC** :

ב־ ESP32 הפונקציה WiFi.macAddress() מחזירה את ה־ MAC **רק אם ה WiFi-מאותחל כראוי.**

נכון שקריאת WiFi.mode(WIFI_MODE_STA) אמורה להספיק **אבל לפעמים צריך לוודא שהרשת ממש התחילה לפעול.** לשם כך נרשום את התוכנית הבאה :

```
#include <WiFi.h>

void setup()
{
  Serial.begin(115200);
  delay(1500);

  WiFi.mode(WIFI_MODE_STA);      // מצב תחנה
  WiFi.begin();                  // התחלה "רכה" של ה-WiFi (למרות שאין SSID)

  delay(100);                    // המתן רגע

  Serial.print("STA MAC: ");
  Serial.println(WiFi.macAddress());

  WiFi.mode(WIFI_MODE_AP);       // מצב נקודת גישה
  Serial.print("AP MAC: ");
  Serial.println(WiFi.softAPmacAddress());
}

void loop()
{
}
```

ואז קיבלתי את הכתובת שבאיור הבא :

APmacAddress | Arduino IDE 2.3.3

ESP32 Dev Module

APmacAddress.ino

```
#include <WiFi.h>

void setup()
{
  Serial.begin(115200);
  delay(1500);

  WiFi.mode(WIFI_MODE_STA);       // מצב תחנה
  WiFi.begin();                   // SSID התחלה "רכה" של ה WiFi למרות שאין

  delay(100);                     // המתן רגע

  Serial.print("STA MAC: ");
  Serial.println(WiFi.macAddress());

  WiFi.mode(WIFI_MODE_AP);        // מצב נקודת גישה
  Serial.print("AP MAC: ");
  Serial.println(WiFi.softAPmacAddress());
}

void loop()
{
}
```

Output Serial Monitor ×

Message (Enter to send message to 'ESP32 Dev Module' on 'COM5')

```
13:36:14.972 -> ets Jul 29 2019 12:21:46
13:36:14.972 ->
13:36:14.972 -> rst:0x1 (POWERON_RESET),boot:0x13 (SPI_FAST_FLASH_BOOT)
13:36:14.972 -> configsip: 0, SPIWP:0xee
13:36:14.972 -> clk_drv:0x00,q_drv:0x00,d_drv:0x00,cs0_drv:0x00,hd_drv:0x00,wp_drv:0x00
13:36:14.972 -> mode:DIO, clock div:1
13:36:14.972 -> load:0x3fff0030,len:4888
13:36:14.972 -> load:0x40078000,len:16516
13:36:14.972 -> load:0x40080400,len:4
13:36:14.972 -> load:0x40080404,len:3476
13:36:14.972 -> entry 0x400805b4
13:36:16.973 -> STA MAC: 88:13:BF:69:33:08
13:36:16.973 -> AP MAC: 88:13:BF:69:33:09
```

2 כתובות MAC שקיבלנו

איור 9 : כתובת ה STA MAC וה AP MAC

הקריאה ל WiFi.begin( ) בשורה 9 באה בדרך כלל בצורה הבאה : WiFi.begin(ssid, password) ; כלומר : WiFi.begin("סיסמה","שם-רשת"); . כשכותבים רק WiFi.begin(); ללא פרמטרים היא מנסה להתחבר לרשת האחרונה שה־ ESP32 זוכר. זה שימושי כשרוצים שה־ ESP32 יתחבר אוטומטית אחרי אתחול ויפעל רק אם בעבר השתמשנו ב WiFi.begin(ssid, password); והמידע נשמר ב NVS - הזיכרון הלא נדיף של ה ESP32 .

במקרה של הקוד שלנו שלא רשמנו SSID הקריאה ל־ WiFi.begin(); **לא באמת מחברת לרשת** , אלא **מעירה את ממשק הרשת** ומבצעת אתחול מסוים למודול WiFi . זה מאפשר ל־ WiFi.macAddress() להחזיר כתובת תקינה, אפילו אם אין רשת להתחבר אליה. במילים אחרות: זו דרך "לבעוט" ל־ WiFi שיתחיל לפעול, גם אם לא סיפקנו לו רשת.

מה ההבדל בין STA MAC ל AP MAC ?

STA MAC (Station Mode MAC) זו כתובת ה MAC של ה־ ESP32 כאשר הוא פועל **כ"תחנה** – "כלומר, כשהוא **מתחבר לרשת WiFi קיימת** (כמו לנתב שלנו או לנקודת גישה).

שימושים: התחברות לאינטרנט , תקשורת עם שרתים אחרים ברשת, שימוש רגיל בפרוטוקולי IoT כמו MQTT , HTTP וכו'.

AP MAC (Access Point Mode MAC) זו כתובת ה MAC של ה־ ESP32 כאשר הוא פועל **כנקודת גישה** – כלומר, כשהוא **יוצר רשת WiFi משלו** ,והטלפון או המחשב שלנו יכולים להתחבר אליו.

שימושים: מצב תכנות או קונפיגורציה , פרויקטים שבהם ה ESP32 יוצר רשת מקומית (ללא ראוטר), חיבור ישיר לטלפון/ טאבלט.

למה הן שונות?

כתובת MAC היא ייחודית לכל ממשק רשת.

ל־ ESP32 יש **שני ממשקי WiFi נפרדים לוגית**:

- אחד למצב STA
- אחד למצב AP

לכן, יש לו **שתי כתובות MAC שונות** ,לרוב הן עוקבות (כמו באיור 9 – ההבדל הוא רק בספרה האחרונה).

אם עובדים עם **שליטה מרחוק** דרך – WiFi נצטרך בדרך כלל את **STA MAC** כשה ESP32 מתחבר לראוטר.

אם נרצה שהטלפון שלנו יתחבר **ישירות ל ESP32** נשתמש ב **AP MAC** כאשר נפעיל WiFi.softAP().

ז.2 התוכנית של ה MASTER השולח את ההודעות :

כעת נרשום את הקוד הבא ונעלה אותו ל MASTER . לפני ההעלאה נכניס את כתובת ה- MAC של מכשיר העבד. נשלח את ההודעה "שלום לכול העולם!" בכל שנייה דרך ESPNOW. האיור הבא מראה את התוכנית :

WWW.

esp_now_sender

```
עבד  ESP32 שלום לכל העולם !! " אל רכיב"  ESP NOW תוכנית המשדרת ב  //
#include <esp_now.h>
#include <WiFi.h>

 SLAVE נכניס את כתובת ה MAC של ה //
uint8_t broadcastAddress[] = {0xB0, 0xA7, 0x32, 0x24, 0x1A, 0xCC};

char msg[] = "!! שלום לכול העולם";

esp_now_peer_info_t peerInfo;

void OnDataSent(const uint8_t *mac_addr, esp_now_send_status_t status)
{
  Serial.print("\r\n מצב המשלוח :");
  Serial.println(status == ESP_NOW_SEND_SUCCESS ? "נמסר בהצלחה" : "מסירה נכשלה");
}

void setup()
{
  Serial.begin(115200);
  WiFi.mode(WIFI_STA);
  if (esp_now_init() != ESP_OK)
  {
    Serial.println("ESP NOW שגיאה באתחול");
    while(1);
  }

  esp_now_register_send_cb(OnDataSent);
  memcpy(peerInfo.peer_addr, broadcastAddress, 6);
  peerInfo.channel = 0;
  peerInfo.encrypt = false;

  if (esp_now_add_peer(&peerInfo) != ESP_OK)
  {
    Serial.println("כשלון בהוספת עמית/חבר");
    while(1);
  }
}

void loop()
{
  esp_err_t result = esp_now_send(broadcastAddress, (uint8_t *) &msg, sizeof(msg));

  if (result == ESP_OK)
  {
    Serial.println(" נשלח בהצלחה");
  }
  else
  {
    Serial.println("שגיאה בזמן שליחת נתונים");
  }
  delay(1000);
}
```

כתובת ה MAC
שהתקבלה באיור 8

איור 10 : תוכנית esp_now של המשדר

נרשום את התוכנית שוב כדי שיהיה קל לעשות copy – paste ולאחריה נסביר את התוכנית:

// תוכנית המשדרת ב ESP NOW "אל רכיב " !! שלום לכל העולם ESP32 עבד

#include <esp_now.h>

#include <WiFi.h>

```
// נכניס את כתובת ה MAC של ה SLAVE
uint8_t broadcastAddress[] = {0xB0, 0xA7, 0x32, 0x24, 0x1A, 0xCC};

char msg[] = "!! שלום לכול העולם ";

esp_now_peer_info_t peerInfo;

void OnDataSent(const uint8_t *mac_addr, esp_now_send_status_t status)
{
  Serial.print("\r\n מצב המשלוח :");
  Serial.println(status == ESP_NOW_SEND_SUCCESS ? "מסירה נכשלה" : "נמסר בהצלחה");
}

void setup()
{
  Serial.begin(115200);
  WiFi.mode(WIFI_STA);
  if (esp_now_init() != ESP_OK)
  {
    Serial.println("שגיאה באתחול ESP NOW");
    while(1);
  }

  esp_now_register_send_cb(OnDataSent);
  memcpy(peerInfo.peer_addr, broadcastAddress, 6);
  peerInfo.channel = 0;
  peerInfo.encrypt = false;

  if (esp_now_add_peer(&peerInfo) != ESP_OK)
  {
    Serial.println("כשלון בהוספת עמית/חבר");
    while(1);
  }
```

```
}

void loop()
{
  esp_err_t result = esp_now_send(broadcastAddress, (uint8_t *) &msg, sizeof(msg));

  if (result == ESP_OK)
  {
    Serial.println("נשלח בהצלחה ");
  }
  else
  {
    Serial.println("שגיאה בזמן שליחת נתונים");
  }
  delay(1000);
}
```

- לאחר העלאת הקוד, נפתח את המוניטור הטורי ונגדיר את קצב השידור ל- 115200, לאחר מכן נלחץ על BOOT בלוח ה ESP32 .

ז.3 הסבר התוכנית

השורה הראשונה בתוכנית מסבירה מה עושה התוכנית . השורה לא ברורה בגלל העתקה מתוכנת הארדואינו לתוכנת ה WORD שגורמת להיפוך שורות שבהן יש גם עברית וגם אנגלית. צריך להיות בשורה זו המשפט :

// תוכנית המשדרת ב ESP NOW "שלום לכל העולם !! " אל רכיב ESP32 עבד .

- שתי השורות הבאות :

```
#include <esp_now.h>
#include <WiFi.h>
```

אומרות לקומפיילר לכלול את 2 קבצי הכותרת הרשומים . בקבצים אלו יש הצהרות על קבועים ופונקציות שמשתמשים בתוכנית.

- השורה

//נכניס את כתובת ה MAC של ה SLAVE

```
uint8_t broadcastAddress[] = {0xB0, 0xA7, 0x32, 0x24, 0x1A, 0xCC};
```

מכניסה את כתובת ה MAC של כרטיס ה ESP32 עבד שמצאנו בתוכנית הקודמת (ניתן לראות את הכתובת באיור קודם).

- ההודעה שנשדר מה MASTER אל ה SLAVE כל שנייה היא : "שלום לכול העולם !! "

```
char msg[] = "שלום לכול העולם !!";
```

- השורה הבאה מגדירה משתנה לאחסון מידע של עמיתים (מידע של כרטיסים אחרים).

```
esp_now_peer_info_t peerInfo;
```

- השורה הבאה מגדירה פונקציה ששמה OnDataSent . היא לא מחזירה ערך והיא מקבלת מצביע/כתובת של כתובת ה MAC וכמו כן משתנה בשם status שהוא אובייקט של המחלקה esp_now_send_status_t . זוהי פונקציית התקשרות חוזרת שתבוצע בעת שליחת הודעה. אנו נדפיס את סטטוס המסירה - "נשלח בהצלחה" או "שליחה נכשלה" באמצעות פונקציה זו.

```
void OnDataSent(const uint8_t *mac_addr, esp_now_send_status_t status)
```

- השורות הבאות הן שורות הפונקציה. בשורה הראשונה אנחנו מדפיסים : "מצב המשלוח : " ובשורה אחריה מופיע המשפט עם סימן השאלה (הנקרא משפט טרינארי) . במשפט בודקים האם הייתה הצלחה בשליחת חבילת הנתונים ? אם כן מודפס "נמסר בהצלחה" ואם לא אז מודפס "מסירה נכשלה" . **הפונקציה מחזירה 0 אם השליחה הצליחה ו 1 אם נכשלה**. גם פה יש היפוך במיקום של "מסירה נכשלה" ו "נמסר בהצלחה" בגלל ההבדל בין העורך של הארדואינו ושל הוורד במעבר מאנגלית לעברית (ראה איור קודם איך צריכה להיראות השורה).

```
{
Serial.print("\r\n מצב המשלוח :");
Serial.println(status == ESP_NOW_SEND_SUCCESS ? "מסירה נכשלה" : "נמסר בהצלחה");
}
```

- בפונקציית ה ( )setup מאתחלים בשורה הראשונה את קצב התקשורת הטורית עם המוניטור של הארדואינו ל 115200 ביטים בשנייה. בשורה השנייה מגדירים את ה ESP32 כתחנת WIFI :

```
Serial.begin(115200);
WiFi.mode(WIFI_STA);
```

- בשורות הבאות של פונקציית ה setup בודקים האם האתחול של esp_now לא הצליח ? כלומר הייתה שגיאה באתחול . במקרה כזה מדפיסים למוניטור "שגיאה באתחול ESP_NOW " ומסיימים את התוכנית על ידי לולאה אין סופית של while(1);.

```
if (esp_now_init() != ESP_OK)
  {
   Serial.println("שגיאה באתחול ESP NOW");
   while(1);;
  }
```

- במידה והאתחול הצליח עוברים לשורות הבאות ונרשום את פונקציית ההתקשרות החוזרת שתיקרא בעת שליחת הודעה.

```
esp_now_register_send_cb(OnDataSent);
```

- בשורות הבאות לאחר מכן, נעשה תאום למכשיר מקלט ESP-NOW כדי לשלוח נתונים. מבצעים תאום עם הכתובת של העמית . אם התאום לא הצליח מדפיסים למוניטור "כשלון בהוספת עמית/חבר" ומסיימים את התוכנית. אם נמצא מכשיר עמית עוברים לפונקציית ה loop( ) .

```
memcpy(peerInfo.peer_addr, broadcastAddress, 6);
peerInfo.channel = 0;
peerInfo.encrypt = false;
```

```
if (esp_now_add_peer(&peerInfo) != ESP_OK)
 {
  Serial.println("כשלון בהוספת עמית/חבר");
  while(1);
 }
```

- בפונקציית ה loop( ) נשלח בכל שנייה את ההודעה " שלום לכול העולם !! " .
- תחילה נגדיר משתנה ששמו result. בשורה הבאה קוראים לפונקציה / מתודה esp_now_send( ) לה שולחים את הכתובת של העמית ואת ההודעה להדפסה וגם את כמות התווים בהודעה. הפונקציה/מתודה תחזיר למשתנה result האם השליחה הצליחה או נכשלה.

```
esp_err_t result = esp_now_send(broadcastAddress, (uint8_t *) &msg, sizeof(msg));
```

- אם השליחה הצליחה מדפיסים למסך המוניטור " נשלח בהצלחה " . אם השליחה לא הצליחה מדפיסים "שגיאה בזמן שליחת נתונים" .

```
 if (result == ESP_OK)
 {
  Serial.println("נשלח בהצלחה");
 }
 else
 {
  Serial.println("שגיאה בזמן שליחת נתונים");
 }
```

- בשורה הבאה עושים השהייה של שנייה ומתחילים את פונקציית ה loop( ) פעם נוספת.

```
delay(1000);
```

ח. התוכנית של הקליטה ב SLAVE המקבל את ההודעות :

```
 /*
  *  ומדפיסה את הנתונים למסך המוניטור ESP-NOW התוכנית קולטת נתונים בתקשורת
  */
 #include <esp_now.h>
 #include <WiFi.h>

 // הגדרה של מבנה שבו משתנה שהוא מערך של 32 תווים ( מחרוזת).
 // כמו כן נגדיר משתנה מטיפוס המבנה הזה
```

```
typedef struct struct
{
    char a[32];
} struct_message;

// נגדיר משתנה מטיפוס המבנה הזה
struct_message myData;

// פונקציה המקבלת את הכתובת שלה , מצביע (כתובת)  ואת אורך המחרוזת המתקבלת.
void OnDataRecv(const uint8_t * mac, const uint8_t *incomingData, int len)
{
  memcpy(&myData, incomingData, sizeof(myData));
  Serial.println(myData.a);
}
void setup()
{
  Serial.begin(115200);
  delay(1500);
  WiFi.mode(WIFI_STA);
  if (esp_now_init() != ESP_OK)   //  אם יש שגיאה בתקשורת ?
  {
    Serial.println("שגיאה באתחול ESP-NOW");
    while(1);
  }
  esp_now_register_recv_cb(OnDataRecv);
}
 void loop()
{
}
```

ח.1 הסבר תוכנית הקליטה

- שתי השורות הבאות :

```
#include <esp_now.h>
#include <WiFi.h>
```

בדומה לשולח MASTER אומרות לקומפיילר לכלול את 2 קבצי הכותרת הרשומים . בקבצים אלו יש הצהרות על קבועים ופונקציות שמשתמשים בתוכנית.

- בשורות הבאות מצהירים על מבנה שנקרא struct_message שבו יש שדה/איבר של מערך בן 32 תווים (מחרוזת) ששמו a.

```
typedef struct struct
{
  char a[32];
} struct_message;
```

- נגדיר משתנה בשם myData מטיפוס המבנה struct_message שהצהרנו עליו בשורות מעל שורה זו.
- struct_message myData;
- בשורה הבאה נגדיר פונקציה בשם OnDataRecv( ) . הפונקציה איננה מחזירה ערך והיא מקבלת 3 ערכים : מצביע/כתובת לקבוע שבו נמצא קוד ה MAC שלנו , מצביע לכתובת שבו תיכנס המחרוזת שנקלוט ואורך המחרוזת . זוהי פונקציית התקשרות חוזרת שניקרא לה כאשר ESP32 יקבל את הנתונים דרך ESP-NOW. בפונקציה הזו אנחנו מדפיסים את הנתונים שהתקבלו.
- void OnDataRecv(const uint8_t * mac, const uint8_t *incomingData, int len)

השורות הבאות הן משפטי הפונקציה. במשפט הראשון קוראים לפונקציה memcpy ונשלח לה 3 פרמטרים. קבוע המציין את כתובת ה MAC של הכתובת של כרטיס ה ESP32 שלנו , את המשתנה incomingData שהוא מצביע והערך שבו הוא הכתובת של המחרוזת המתקבלת ואת כמות הבתים שיש במבנה myData (32 בדוגמה שלנו). הפונקציה memcpy מעתיקה את התוכן של המשתנה incomingData (לכאן נכנסו הנתונים) למבנה myData (וליתר דיוק למחרוזת a שבמבנה) . המשפט לאחר מכן מדפיס למסך את הנתון שנקלט באיבר/שדה a שבמבנה myData .

```
{
  memcpy(&myData, incomingData, sizeof(myData));
  Serial.println(myData.a);
}
```

- בפונקציית ה setup( ) , בדומה לזו שבמשדר , מאתחלים בשורה הראשונה את קצב התקשורת הטורית עם המוניטור של הארדואינו ל 115200 ביטים בשנייה. בשורה השנייה מגדירים את ה ESP32 כתחנת WIFI :

```
  Serial.begin(115200);
  WiFi.mode(WIFI_STA);
```

- בשורות הבאות של פונקציית ה setup בודקים האם האתחול של esp_now לא הצליח ? כלומר הייתה שגיאה באתחול . במקרה כזה מדפיסים למוניטור "שגיאה באתחול ESP_NOW " ומסיימים את התוכנית על ידי לולאה אין סופית של while(1);.

```
if (esp_now_init() != ESP_OK)
  {
    Serial.println("שגיאה באתחול ESP NOW");
    while(1);;
  }
```

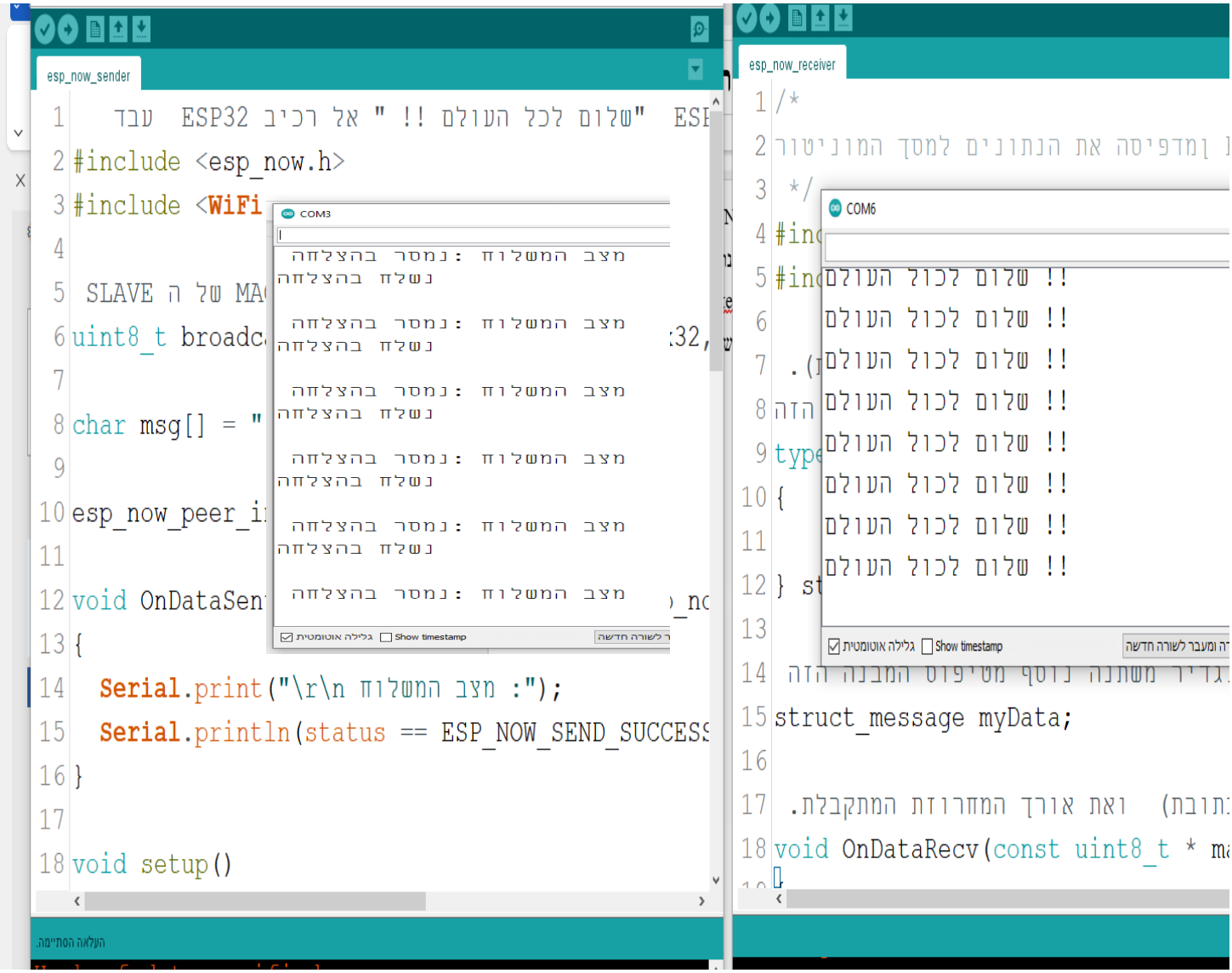
- במשפט הבא היא פונקציית התקשרות חוזרת המופעלת עם קבלת נתונים. כאשר נתונים מתקבלים דרך ESP-NOW, מתבצעת קריאה לפונקציה הזו . כך שלמרות שהיא איננה רשומה בפונקציית ה loop היא תתבצע בכל פעם שיתקבלו נתונים מהמשדר.

esp_now_register_recv_cb(OnDataRecv);

- פונקציית ה loop( ) היא ללא משפטים כי הפונקציה esp_now_register_recv_cb(OnDataRecv) שרשמנו ב setup( ) מתבצעת בכל פעם שיש קליטת נתונים.

ט. הפעלת המשדר והמקלט יחד

בהפעלת 2 המערכות יחד נקבל את המצב המתואר באיור הבא:



איור 11 : תוצאת השידור והקליטה . מצד שמאל תוכנית המשדר וההדפסות שלה ומימין תוכנית המקלט וההדפסה המתקבלת.

י. ביבליוגרפיה

1. chatgpt
2. ESP-NOW Introduction | ESP32 (electronicwings.com)
3. ESP-NOW Two-Way Communication Between ESP32 Boards | Random Nerd Tutorials
4. ESP32 With ESP-Now Protocol : 16 Steps - Instructables

www.arikporat.com