1. **NO. JOBSHEET** : 1
2. **JUDUL** : JARINGAN SENSOR NIRKABEL MENGGUNAKAN ESP-NOW
3. **TUJUAN**
    1) Mahasiswa dapat memahami konsep topologi jaringan sensor nirkabel berbasis ESP-NOW.
    2) Mahasiswa dapat melakukan konfigurasi berbagai topologi ESP-NOW.
    3) Mahasiswa dapat menganalisis dan menentukan topologi ESP-NOW,sesuai dengan studi kasus proyek.
4. **ALAT DAN BAHAN**
    1) ESP32
    2) Breadboard
    3) Kabel jumper
    4) Resistor 10K Ohm
5. **TEORI SINGKAT**

ESP-NOW adalah protokol yang dikembangkan oleh Espressif, yang memungkinkan banyak perangkat untuk berkomunikasi satu sama lain tanpa menggunakan Wi-Fi. Protokol ini mirip dengan konektivitas nirkabel 2.4GHz berdaya rendah. Pendifinisian alamat (MAC Address) pada masing-masing ESP32 diperlukan pada awal konfigurasi. Setelah konfigurasi alamat selesai dilakukan, jaringan *peer-to-peer* akan terbentuk dan perangkat tidak perlu melakukan *handshaking* kembali ketika akan berkomunikasi. Hal ini memunjukkan bahwa setelah perangkat ESP32 saling terpasang satu sama lain, koneksi akan tetap ada. Dengan kata lain, jika tiba-tiba salah satu ESP32 kehilangan daya atau diatur ulang, ketika *restart*, secara otomatis akan terhubung ke pasangan ESP32 yang telah terdefinisi alamatnya untuk melanjutkan komunikasi.

ESP-NOW mempunyai fitur sebagai berikut.

a. Komunikasi unicast yang terenkripsi maupun tidak terenkripsi.
b. Perpaduan komunikasi data yang terenkripsi maupun yang tidak terenkripsi pada perangkat yang berada pada topologi peer-to-peer.
c. Payload (ukuran) data yang dapat dikirm mencapai 250 byte.
d. Terdapat fungsi *callback* yang dapat menginformasikan data berhasil terkirim maupun gagal dikirim.

Selain itu, ESP-NOW mempunyai batasan sebagai berikut.

a. Jumlah maksimal perangkat yang dapat berkomunikasi dalam mode station dengan data terenkripsi adalah 10 unit (6 dalam mode SoftAP atau SoftAP+Station).
b. Untuk komunikasi tidak terenkripsi, jumlah maksimal perangkat adalah 20 unit, termasuk dengan yang terenkripsi.

ESP-NOW mempunyai 2 tipe jaringan, yaitu One-Way Communication dan Two-Way Communication. One-Way Communication terbagi menjadi Point-to-Point, One-to-Many Communication dan Many-to-One Communication. Sementara Two-Way Communication terbagi menjadi Point-to-Point dan Mesh Communication.

5.1. One-Way Communication

Gambar 1. Point-to-Point pada One Way Communication

Gambar 2. One-to-Many Communication

Gambar 3. Many-to-One Communication

Konfigurasi point-to-point seperti Gambar 1 sangat mudah diterapkan dan sangat baik untuk mengirim data dari satu ESP32 ke ESP32 lainnya seperti pembacaan sensor atau perintah ON/OFF untuk mengontrol GPIO. Sementara pada Gambar 2, satu board ESP32 (master) mengirimkan perintah yang sama atau berbeda ke papan ESP32 yang berbeda (slave). Konfigurasi ini sangat ideal untuk membangun suatu sistem seperti remote control. Pada Gambar 3, konfigurasi tersebut sangat ideal untuk sistem pengumpulan data dari beberapa node sensor ke dalam satu board ESP32. Topologi ini mirip seperti topologi star, yang mana ESP32 yang berperan sebagai Slave dapat dikonfigurasi sebagai Web Server.

5.2. Two-Way Communication

Gambar 4. Point-to-Point pada Two-Way Communication

Gambar 5. Topologi Mesh

Konfigurasi point-to-point pada mode Two-Way Communication seperti yang ditunjukkan pada Gambar 4, masing-masing board ESP32 dapat saling berkomunikasi secara simultan untuk bertukar informasi. Sementara itu, Gambar 5 merupakan konfigurasi ESP-NOW dengan topologi Mesh. Dengan topologi tersebut, setiap board ESP32 akan dapat saling berkomunikasi dengan semua anggota jaringan. Hal tersebut dapat mencegah terjadinya kegagalan transmisi data, karena terdapat jalur cadangan. Namun, konsumsi energi pada masing-masing ESP32 akan menjadi semakin besar.

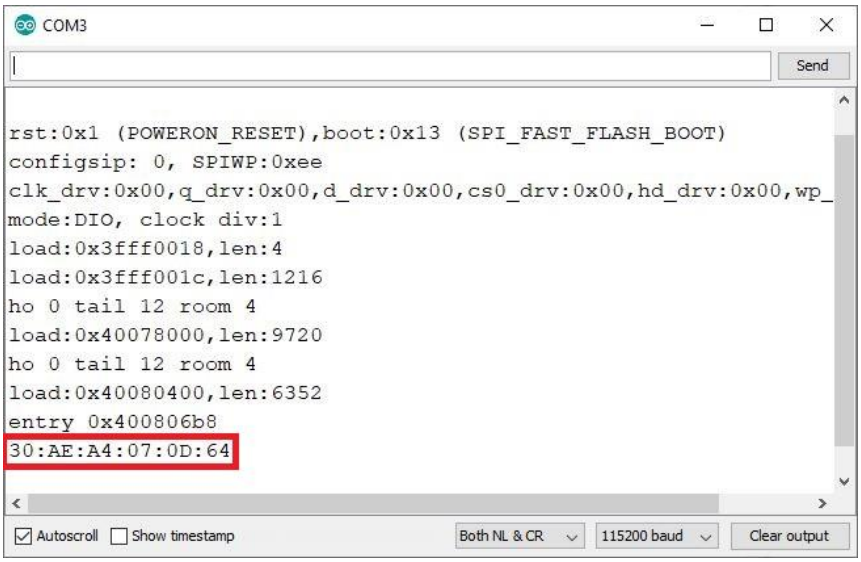
6. LANGKAH PERCOBAAN

A. Memperoleh MAC Address ESP32 Receiver

1. Buka Arduino IDE
2. Kemudian ketikkan script program berikut di Arduino IDE.

```
#include "WiFi.h"

void setup(){
  Serial.begin(115200);
  WiFi.mode(WIFI_MODE_STA);
  Serial.println(WiFi.macAddress());
}

void loop(){

}
```

3. Upload program tersebut ke ESP32.
4. Setelah program berhasil diupload, buka serial monitor.
5. Catat Mac Address ESP32.



B. ESP-NOW One-Way Point-to-Point Communication

1. Siapkan board ESP32 sejumlah 2 unit yang akan diprogram sebagai Sender dan Receiver.
2. Kemudian ketikkan script program berikut untuk mengkonfigurasi ESP32 sebagai sender.
3. Isi array broadcastAddress [] dengan MAC Address ESP32 Receiver yang didapatkan pada praktikum poin A.
4. Upload program pada ESP32 Sender.

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

// Ganti dengan Mac Address ESP32 Receiver
uint8_t broadcastAddress[] = {0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF};

// Struktur pesan sender dan receiver harus sama
typedef struct struct_message {
  char a[32];
  int b;
  float c;
  bool d;
} struct_message;

// Driver untuk struktur pesan
struct_message myData;

esp_now_peer_info_t peerInfo;

// fungsi callback
void OnDataSent(const uint8_t *mac_addr, esp_now_send_status_t status) {
  Serial.print("\r\nStatus Paket Terakhir :\t");
  Serial.println(status == ESP_NOW_SEND_SUCCESS ? "Sukses Terkirim" : "Gagal
Terkirim");
}

void setup() {
  // Init Serial Monitor
  Serial.begin(115200);

  // Set ESP32 sebagai station
  WiFi.mode(WIFI_STA);

  // Init ESP-NOW
  if (esp_now_init() != ESP_OK) {
    Serial.println("Gagal menginisialisasi ESP-NOW");
    return;
  }

  // Fungsi akses register cb untuk perintah mengirim data
  esp_now_register_send_cb(OnDataSent);

  // Register peer
  memcpy(peerInfo.peer_addr, broadcastAddress, 6);
  peerInfo.channel = 0;
  peerInfo.encrypt = false;

  // Add peer
  if (esp_now_add_peer(&peerInfo) != ESP_OK){
    Serial.println("Gagal menambahkan peer");
    return;
  }
}

void loop() {
  // Set values to send
  strcpy(myData.a, "INI ADALAH CHAR");
  myData.b = random(1,20);
  myData.c = 1.2;
```

```
  myData.d = false;

  // Send message via ESP-NOW
  esp_err_t result = esp_now_send(broadcastAddress, (uint8_t *) &myData,
sizeof(myData));

  if (result == ESP_OK) {
    Serial.println("Data berhasil terkirim");
  }
  else {
    Serial.println("Gagal mengirim data");
  }
  delay(2000);
}
```

5. Setelah ESP32 sender dikonfigurasi, kemudian konfigurasikan ESP32 yang lain sebagai Receiver. Ketikkan script program berikut untuk mengkonfigurasi ESP32 sebagai receiver.
6. Upload program, kemudian dokumentasikan hasilnya.
7. Buatlah analisis atau flow chart program untuk menjelaskan fungsi per-bagian pada program.

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

// Struktur pesan sender dan receiver harus sama
typedef struct struct_message {
    char a[32];
    int b;
    float c;
    bool d;
} struct_message;

// Driver struktur pesan
struct_message myData;

// fungsi callback yang akan dieksekusi ketika ada pesan diterima
void OnDataRecv(const uint8_t * mac, const uint8_t *incomingData, int len) {
  memcpy(&myData, incomingData, sizeof(myData));
  Serial.print("Bytes received: ");
  Serial.println(len);
  Serial.print("Char: ");
  Serial.println(myData.a);
  Serial.print("Int: ");
  Serial.println(myData.b);
  Serial.print("Float: ");
  Serial.println(myData.c);
  Serial.print("Bool: ");
  Serial.println(myData.d);
  Serial.println();
}

void setup() {
  // Initialize Serial Monitor
  Serial.begin(115200);
```

```
  // Set ESP32 sebagai station
  WiFi.mode(WIFI_STA);

  // Init ESP-NOW
  if (esp_now_init() != ESP_OK) {
    Serial.println("Error initializing ESP-NOW");
    return;
  }

  // Fungsi akses register cb untuk proses penerimaan data
  esp_now_register_recv_cb(OnDataRecv);
}

void loop() {

}
```

8. Buatlah data dummy dengan ukuran yang terbaca oleh receiver ± 250 byte.
9. Aturlah jarak awal komunikasi antara sender dan receiver yaitu 1 meter. Kemudian ubah jarak komunikasi dengan penambahan 1 meter. Data yang dikirim pada tiap iterasi pengujian adalah 10 data.
10. Aturlah tinggi antena atau peletakan ESP32 pada ground level (menempel tanah), 30 cm di atas tanah, dan 1 meter di atas tanah.
11. Masukkan hasil pengukuran pada kolom berikut dan buatlah analisisnya dengan pendekatan teori freshnel zone.

Tinggi Antena (m)	Jarak Transmisi (m)	Jumlah Pesan yang Dikirim [Dt]	Jumlah Pesan yang Diterima [Dr]	Packet Loss (%) $\frac{Dr-Dt}{Dt} x\ 100$
Ground	1			
	2			
	Dst …			
0.3	1			
	2			
	Dst …			
1	1			
	2			
	Dst …			

C. One-Way, One-to-Many Communication

a) Mengirim Pesan yang Sama Ke Beberapa Board ESP32

1. Siapkan 4 unit board ESP32.
2. Unggah program untuk mendapatkan Mac Address, kemudian catat masing-masing Mac Address pada setiap board ESP32 yang akan diatur sebagai Receiver.
3. Siapkan 1 unit ESP-32 yang akan dikonfigurasi sebagai Master/Sender.
4. Ketik program berikut ini dan tambahkan semua Mac Address ESP32 Receiver pada bagian broadcastAddress[].

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

// REPLACE WITH YOUR ESP RECEIVER'S MAC ADDRESS
uint8_t broadcastAddress1[] = {0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF};
uint8_t broadcastAddress2[] = {0xFF, , , , , };
uint8_t broadcastAddress3[] = {0xFF, , , , , };

typedef struct test_struct {
  int x;
  int y;
} test_struct;

test_struct test;

esp_now_peer_info_t peerInfo;

// callback when data is sent
void OnDataSent(const uint8_t *mac_addr, esp_now_send_status_t status) {
  char macStr[18];
  Serial.print("Packet to: ");
  // Copies the sender mac address to a string
  snprintf(macStr, sizeof(macStr), "%02x:%02x:%02x:%02x:%02x:%02x",
           mac_addr[0], mac_addr[1], mac_addr[2], mac_addr[3], mac_addr[4],
mac_addr[5]);
  Serial.print(macStr);
  Serial.print(" send status:\t");
  Serial.println(status == ESP_NOW_SEND_SUCCESS ? "Delivery Success" : "Delivery
Fail");
}

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

  WiFi.mode(WIFI_STA);

  if (esp_now_init() != ESP_OK) {
    Serial.println("Error initializing ESP-NOW");
    return;
  }

  esp_now_register_send_cb(OnDataSent);

  // register peer
  peerInfo.channel = 0;
  peerInfo.encrypt = false;
  // register first peer
  memcpy(peerInfo.peer_addr, broadcastAddress1, 6);
  if (esp_now_add_peer(&peerInfo) != ESP_OK){
    Serial.println("Failed to add peer");
    return;
  }
  // register second peer
  memcpy(peerInfo.peer_addr, broadcastAddress2, 6);
  if (esp_now_add_peer(&peerInfo) != ESP_OK){
    Serial.println("Failed to add peer");
    return;
  }
  /// register third peer
```

```
  memcpy(peerInfo.peer_addr, broadcastAddress3, 6);
  if (esp_now_add_peer(&peerInfo) != ESP_OK){
    Serial.println("Failed to add peer");
    return;
  }
}

void loop() {
  test.x = random(0,20);
  test.y = random(0,20);

  esp_err_t result = esp_now_send(0, (uint8_t *) &test, sizeof(test_struct));

  if (result == ESP_OK) {
    Serial.println("Sent with success");
  }
  else {
    Serial.println("Error sending the data");
  }
  delay(2000);
}
```

5. Upload program tersebut pada board ESP32 Sender.
6. Siapkan board Receiver, kemudian ketik script berikut ini pada Arduino IDE.

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

//Structure example to receive data
//Must match the sender structure
typedef struct test_struct {
  int x;
  int y;
} test_struct;

//Create a struct_message called myData
test_struct myData;

//callback function that will be executed when data is received
void OnDataRecv(const uint8_t * mac, const uint8_t *incomingData, int len) {
  memcpy(&myData, incomingData, sizeof(myData));
  Serial.print("Bytes received: ");
  Serial.println(len);
  Serial.print("x: ");
  Serial.println(myData.x);
  Serial.print("y: ");
  Serial.println(myData.y);
  Serial.println();
}

void setup() {
  //Initialize Serial Monitor
  Serial.begin(115200);

  //Set device as a Wi-Fi Station
  WiFi.mode(WIFI_STA);

  //Init ESP-NOW
  if (esp_now_init() != ESP_OK) {
```

```
    Serial.println("Error initializing ESP-NOW");
    return;
  }

  // Once ESPNow is successfully Init, we will register for recv CB to
  // get recv packer info
  esp_now_register_recv_cb(OnDataRecv);
}

void loop() {

}
```

7. Upload program tersebut pada Receiver.
8. Dokumentasikan output dari program tersebut secara lengkap pada masing-masing board.
9. Matikan salah satu board Receiver, dokumentasikasikan hasilnya, dan buatlah analisisnya.
10. Buatlah koneksi menggunakan semua board ESP32 yang ada dikelas, dengan menambahkan Receiver ke dalam jaringan secara bertahap,
11. Dokumentasikan hasilnya secara lengkap. Catat dan analisis jumlah board maksimal yang dapat membentuk jaringan.

b) Mengirim Pesan yang Berbeda Ke Beberapa Board ESP32

1. Siapkan 4 board ESP32, 1 board sebagai Sender dan 3 board sebagai Receiver.
2. Buatlah program pada Sender agar dapat mengirim pesan yang berbeda pada 3 Receiver.
3. Tips : Buat 3 buat struktur data.

   Buat 3 random data dummy generator pada masing-masing variabel x dan y.

   Gunakan fungsi esp_now_send() pada masing-masing broadcastAddress dengan script yang terpisah

D. One-Way, Many-to-One Communication

Di dalam mode ini, Receiver harus dapat mengidentifikasi setiap MAC Address unik dari Sender. Namun, untuk dapat membaca MAC Address yang berbeda cukup rumit dan butuh sedikit trik. Sehingga, untuk membuatnya lebih mudah, masing-masing Sender akan diberikan ID unik, agar Receiver dapat lebih mudah mengidentifikasi Sender.

1. Siapkan 4 board ESP32. 3 board diatur sebagai Sender dan 1 board diatur sebagai Receiver.
2. Unggah program untuk menemukan MAC Address pada board Receiver, kemudian catat MAC Address-nya.
3. Ketikkan program berikut pada Arduino IDE untuk mengkonfigurasi board Sender.

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

// REPLACE WITH THE RECEIVER'S MAC Address
uint8_t broadcastAddress[] = {0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF};

// Structure example to send data
// Must match the receiver structure
typedef struct struct_message {
    int id; // must be unique for each sender board
    int x;
    int y;
} struct_message;

// Create a struct_message called myData
struct_message myData;

// Create peer interface
esp_now_peer_info_t peerInfo;

// callback when data is sent
void OnDataSent(const uint8_t *mac_addr, esp_now_send_status_t status) {
  Serial.print("\r\nLast Packet Send Status:\t");
  Serial.println(status == ESP_NOW_SEND_SUCCESS ? "Delivery Success" : "Delivery
Fail");
}

void setup() {
  // Init Serial Monitor
  Serial.begin(115200);

  // Set device as a Wi-Fi Station
  WiFi.mode(WIFI_STA);

  // Init ESP-NOW
  if (esp_now_init() != ESP_OK) {
    Serial.println("Error initializing ESP-NOW");
    return;
  }

  // Once ESPNow is successfully Init, we will register for Send CB to
  // get the status of Trasnmitted packet
  esp_now_register_send_cb(OnDataSent);

  // Register peer
  memcpy(peerInfo.peer_addr, broadcastAddress, 6);
  peerInfo.channel = 0;
  peerInfo.encrypt = false;

  // Add peer
  if (esp_now_add_peer(&peerInfo) != ESP_OK){
    Serial.println("Failed to add peer");
    return;
  }
}

void loop() {
  // Set values to send
  myData.id = 1;
```

```
  myData.x = random(0,50);
  myData.y = random(0,50);

  // Send message via ESP-NOW
  esp_err_t result = esp_now_send(broadcastAddress, (uint8_t *) &myData,
sizeof(myData));

  if (result == ESP_OK) {
    Serial.println("Sent with success");
  }
  else {
    Serial.println("Error sending the data");
  }
  delay(10000);
}
```

4. Upload program tersebut pada board Sender.
5. Siapkan board Receiver, ketikkan script berikut di Arduino IDE, kemudian upload program tersebut.

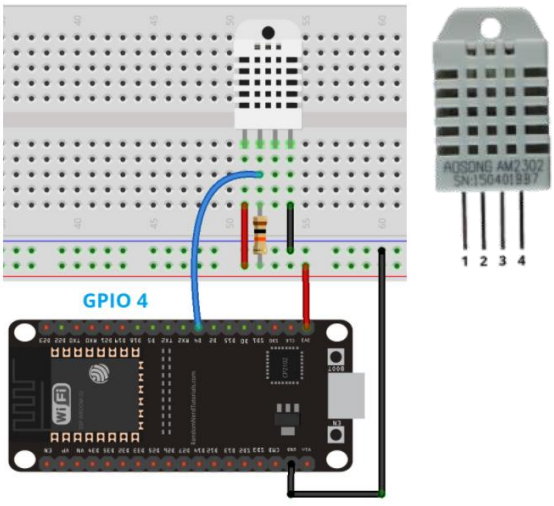
```
#include <esp_now.h>
#include <WiFi.h>

// Structure example to receive data
// Must match the sender structure
typedef struct struct_message {
  int id;
  int x;
  int y;
}struct_message;

// Create a struct_message called myData
struct_message myData;

// Create a structure to hold the readings from each board
struct_message board1;
struct_message board2;
struct_message board3;

// Create an array with all the structures
struct_message boardsStruct[3] = {board1, board2, board3};

// callback function that will be executed when data is received
void OnDataRecv(const uint8_t * mac_addr, const uint8_t *incomingData, int len) {
  char macStr[18];
  Serial.print("Packet received from: ");
  snprintf(macStr, sizeof(macStr), "%02x:%02x:%02x:%02x:%02x:%02x",
           mac_addr[0], mac_addr[1], mac_addr[2], mac_addr[3], mac_addr[4],
mac_addr[5]);
  Serial.println(macStr);
  memcpy(&myData, incomingData, sizeof(myData));
  Serial.printf("Board ID %u: %u bytes\n", myData.id, len);
  // Update the structures with the new incoming data
  boardsStruct[myData.id-1].x = myData.x;
  boardsStruct[myData.id-1].y = myData.y;
  Serial.printf("x value: %d \n", boardsStruct[myData.id-1].x);
  Serial.printf("y value: %d \n", boardsStruct[myData.id-1].y);
  Serial.println();
```

```
}

void setup() {
  //Initialize Serial Monitor
  Serial.begin(115200);

  //Set device as a Wi-Fi Station
  WiFi.mode(WIFI_STA);

  //Init ESP-NOW
  if (esp_now_init() != ESP_OK) {
    Serial.println("Error initializing ESP-NOW");
    return;
  }

  // Once ESPNow is successfully Init, we will register for recv CB to
  // get recv packer info
  esp_now_register_recv_cb(OnDataRecv);
}

void loop() {
  // Acess the variables for each board
  /*int board1X = boardsStruct[0].x;
  int board1Y = boardsStruct[0].y;
  int board2X = boardsStruct[1].x;
  int board2Y = boardsStruct[1].y;
  int board3X = boardsStruct[2].x;
  int board3Y = boardsStruct[2].y;*/

  delay(10000);
}
```

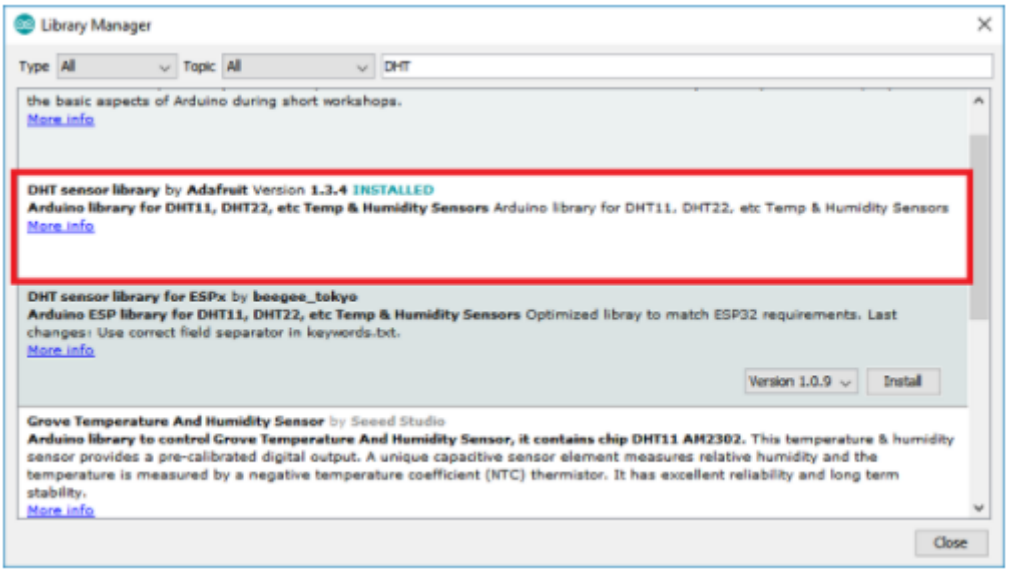
6. Buka serial monitor dan dokumentasikan output program.

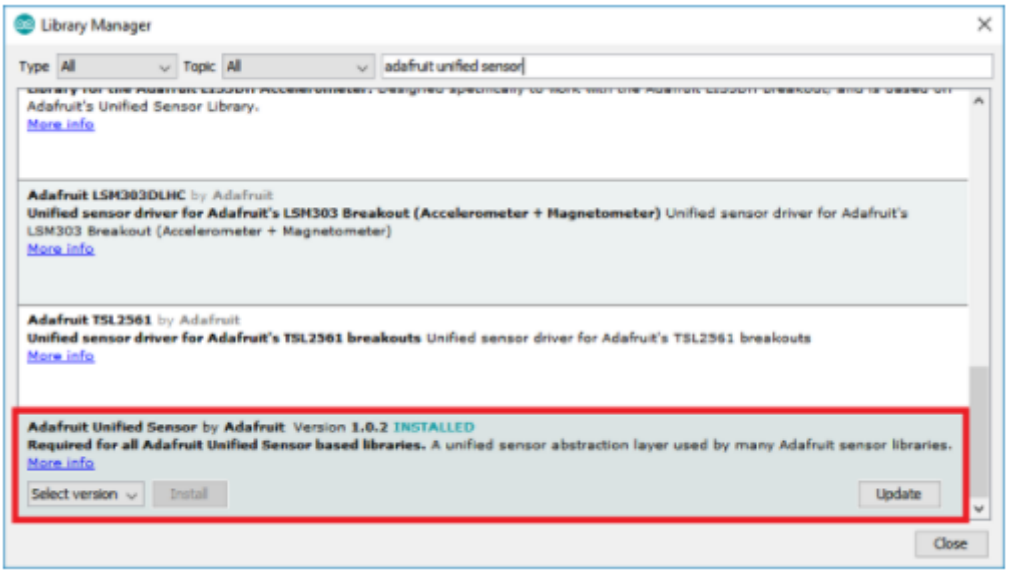
E. Two-Way Communication

1. Siapkan 2 unit ESP32 dan 2 unit sensor DHT11.
2. Buatlah rangkaian seperti pada Gambar di bawah ini.



3. Install library sensor DHT 11 melalui **Sketch > Include Library > Manage Libraries**. Ketikkan **DHT** pada kolom pencarian, pilih library yang akan diinstall seperti pada Gambar berikut ini. Kemudian install juga **Adafruit Unified Sensor** menggunakan cara yang sama.





4. Upload program berikut ini untuk melakukan pengecekan sensor DHT11.

```
#include "DHT.h"

#define DHTPIN 4     // Digital pin connected to the DHT sensor

// Uncomment whatever type you're using!
#define DHTTYPE DHT11   // DHT 11
//#define DHTTYPE DHT22   // DHT 22  (AM2302), AM2321
//#define DHTTYPE DHT21   // DHT 21 (AM2301)

DHT dht(DHTPIN, DHTTYPE);
```

```
void setup() {
 Serial.begin(9600);
 Serial.println(F("DHT11 Embedded System Test!"));

 dht.begin();
}

void loop() {
 // Wait a few seconds between measurements.
 delay(2000);

 // Reading temperature or humidity takes about 250 milliseconds!
 // Sensor readings may also be up to 2 seconds 'old' (its a very slow sensor)
 float h = dht.readHumidity();
 // Read temperature as Celsius (the default)
 float t = dht.readTemperature();
 // Read temperature as Fahrenheit (isFahrenheit = true)
 float f = dht.readTemperature(true);

 // Check if any reads failed and exit early (to try again).
 if (isnan(h) || isnan(t) || isnan(f)) {
  Serial.println(F("Failed to read from DHT sensor!"));
  return;
 }

 // Compute heat index in Fahrenheit (the default)
 float hif = dht.computeHeatIndex(f, h);
 // Compute heat index in Celsius (isFahreheit = false)
 float hic = dht.computeHeatIndex(t, h, false);

 Serial.print(F("Humidity: "));
 Serial.print(h);
 Serial.print(F("%  Temperature: "));
 Serial.print(t);
 Serial.print(F("°C "));
 Serial.print(f);
 Serial.print(F("°F  Heat index: "));
 Serial.print(hic);
 Serial.print(F("°C "));
 Serial.print(hif);
 Serial.println(F("°F"));
}
```

5. Dokumentasikan output program yang ditampilkan pada serial monitor Arduino IDE.
6. Unggah program menemukan MAC Address pada masing-masing board, kemudian catat MAC Address-nya.
7. Ketikkan script berikut, kemudian masukkan MAC Address receiver pada masing-masing board.
8. Upload program pada masing-masing board.

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

#include <DHT.h>

#define DHTPIN 4
#define DHTTYPE DHT11

DHT dht(DHTPIN, DHTTYPE);

// REPLACE WITH THE MAC Address of your receiver
uint8_t broadcastAddress[] = {0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF};

// Define variables
float temperature;
float humidity;

// Define variables to store incoming readings
float incomingTemp;
float incomingHum;

// Variable to store if sending data was successful
String success;

//Structure example to send data
//Must match the receiver structure
typedef struct struct_message {
    float temp;
    float hum;
} struct_message;

// Create a struct_message sensors reading
struct_message DHTReadings;

// Create a struct_message to hold incoming sensor readings
struct_message incomingReadings;

esp_now_peer_info_t peerInfo;

// Callback when data is sent
void OnDataSent(const uint8_t *mac_addr, esp_now_send_status_t status) {
  Serial.print("\r\nLast Packet Send Status:\t");
  Serial.println(status == ESP_NOW_SEND_SUCCESS ? "Delivery Success" : "Delivery
Fail");
  if (status ==0){
    success = "Delivery Success :)";
  }
  else{
    success = "Delivery Fail :(";
  }
}

// Callback when data is received
void OnDataRecv(const uint8_t * mac, const uint8_t *incomingData, int len) {
  memcpy(&incomingReadings, incomingData, sizeof(incomingReadings));
  Serial.print("Bytes received: ");
  Serial.println(len);
  incomingTemp = incomingReadings.temp;
  incomingHum = incomingReadings.hum;
```

```
}

void setup() {
  // Init Serial Monitor
  Serial.begin(115200);

  // Init DHT
  dht.begin();

  // Set device as a Wi-Fi Station
  WiFi.mode(WIFI_STA);

  // Init ESP-NOW
  if (esp_now_init() != ESP_OK) {
    Serial.println("Error initializing ESP-NOW");
    return;
  }

  // Once ESPNow is successfully Init, we will register for Send CB to
  // get the status of Trasnmitted packet
  esp_now_register_send_cb(OnDataSent);

  // Register peer
  memcpy(peerInfo.peer_addr, broadcastAddress, 6);
  peerInfo.channel = 0;
  peerInfo.encrypt = false;

  // Add peer
  if (esp_now_add_peer(&peerInfo) != ESP_OK){
    Serial.println("Failed to add peer");
    return;
  }
  // Register for a callback function that will be called when data is received
  esp_now_register_recv_cb(OnDataRecv);
}

void loop() {
  delay(2000);
  getReadings();

  // Set values to send
  DHTReadings.temp = temperature;
  DHTReadings.hum = humidity;

  // Send message via ESP-NOW
  esp_err_t result = esp_now_send(broadcastAddress, (uint8_t *) &DHTReadings,
sizeof(DHTReadings));

  if (result == ESP_OK) {
    Serial.println("Sent with success");
  }
  else {
    Serial.println("Error sending the data");
  }

  delay(1000);
}

void getReadings(){
```

```
  temperature=dht.readTemperature();
  humidity=dht.readHumidity();

  if (isnan(h) || isnan(t) || isnan(f)) {
    Serial.println(F("Failed to read from DHT sensor!"));
    return;
  }
  Serial.print(F("Humidity: "));
  Serial.print(h);
  Serial.print(F("%  Temperature: "));
  Serial.print(t);
  Serial.print(F("°C "));
}
```

9. Buka serial monitor pada Arduino IDE, kemudian dokumentasikan outputnya.
10. Buatlah jaringan sensor nirkabel ESPNow topologi MESH berlandaskan Two-Way Communication. Diagram blok jaringan dapat dilihat pada Gambar 5.

7. PERTANYAAN DAN TUGAS