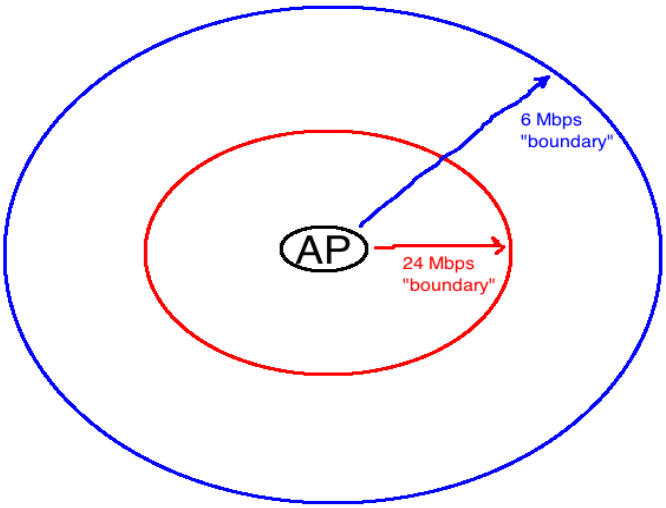
Programme Longue distance

Sur module ESP32 dans l'Ide Arduino

Après avoir installé le module ESP32 dans l'Ide Arduino, j'ai compilé le code ci-dessous :

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

const char* ssid     = "yourssid";
const char* password = "yourpasswd";

WiFiServer server(80);
int buff_size = -1;
uint8_t client_buffer[1024];

uint8_t current_protocol;
 esp_interface_t current_esp_interface;
wifi_interface_t current_wifi_interface;

void setup()
 {
 Serial.begin(115200);
  pinMode(5, OUTPUT);      // set the LED pin mode

    delay(10);

    // We start by connecting to a WiFi network

    Serial.println();
    Serial.println();
    Serial.print("Connecting to ");
    Serial.println(ssid);

    //WiFi.begin(ssid, password);
   WiFi.mode(WIFI_AP_STA);
 //tcpip_adapter_init();
 //wifi_init_config_t cfg = WIFI_INIT_CONFIG_DEFAULT();
   //esp_wifi_init(&cfg);
  //esp_wifi_set_storage(WIFI_STORAGE_RAM);
//esp_wifi_set_mode(WIFI_MODE_AP);     check_protocol();
```

esp_wifi_set_protocol(current_wifi_interface, WIFI_PROTOCOL_LR); // la fonction permet d'activer la wifi Low Rate qui peut être capté à une plus longue distance comme illustré dans la figure suivante :




```
check_protocol();
 WiFi.softAP(ssid, password);
  check_protocol();
  Serial.println("");
 Serial.println("WiFi connected.");
   Serial.println("IP address: ");
   Serial.println(WiFi.localIP());
  server.begin();

}

int value = 0;

void loop(){
WiFiClient client = server.available();   // listen for incoming clients

  //
 if (client) {                             // if you get a client,
Serial.println("*********************************************");
 Serial.print("New Client in ");          // print a message out the serial port
Serial.println(client.remoteIP());
 String currentLine = "";                // make a String to hold incoming data from the
client
  if (client.connected()) {            // loop while the client's connected
  buff_size = client.available();
```

```
   if (buff_size) {             // if there's bytes to read from the client,
Serial.print("Payload size: ");
  Serial.println(buff_size);
   Serial.println("Data is:");
  Serial.println();
  client.read(client_buffer,buff_size);             // read a byte, then
Serial.write(client_buffer,buff_size);                    // print it out the serial monitor
Serial.println();         Serial.println();
      }     }
   client.flush();     // close the connection:
 client.stop();     Serial.println("Client Disconnected.");
Serial.println("************************************************");
    Serial.println();
}
}

esp_interface_t check_protocol() {
char error_buf1[100];
   tcpip_adapter_get_esp_if(&current_esp_interface);
   if (current_esp_interface == ESP_IF_WIFI_STA)
   Serial.println("Interface is ESP_IF_WIFI_STA");
   else if (current_esp_interface == ESP_IF_WIFI_AP)
 Serial.println("Interface is ESP_IF_WIFI_AP");
  else       Serial.println("Unknown interface!!");
  current_wifi_interface = current_esp_interface;
  if (current_wifi_interface == WIFI_IF_STA)
   Serial.println("Interface is WIFI_IF_STA");
  else if (current_wifi_interface == WIFI_IF_AP)
   Serial.println("Interface is WIFI_IF_AP");
  else       Serial.println("Unknown interface!!");
 esp_err_t error_code = esp_wifi_get_protocol(current_wifi_interface,
&current_protocol);
esp_err_to_name_r(error_code,error_buf1,100);
Serial.print("esp_wifi_get_protocol error code: ");
   Serial.println(error_buf1);
  Serial.print("Current protocol code is ");
   Serial.println(current_protocol);
  if ((current_protocol&WIFI_PROTOCOL_11B) == WIFI_PROTOCOL_11B)
Serial.println("Protocol is WIFI_PROTOCOL_11B");
   if ((current_protocol&WIFI_PROTOCOL_11G) == WIFI_PROTOCOL_11G)
Serial.println("Protocol is WIFI_PROTOCOL_11G");
    if ((current_protocol&WIFI_PROTOCOL_11N) == WIFI_PROTOCOL_11N)
Serial.println("Protocol is WIFI_PROTOCOL_11N");
  if ((current_protocol&WIFI_PROTOCOL_LR) == WIFI_PROTOCOL_LR)
Serial.println("Protocol is WIFI_PROTOCOL_LR");

    return current_esp_interface;
```

```
}
```