|  |  |  |
| --- | --- | --- |
|  | **Guia de montagem iModBot@ipleiria.pt** |  |

# Tutorial de como carregar programas para o ESP32 via WiFi

O robô iModBot é um veículo elétrico de pequenas proporções controlado pelo microcontrolador ESP32. Este microcontrolador pode ser programado usando a linguagem C/C++ e o software gratuito Arduino IDE.

Este tutorial vai focar-se na funcionalidade OTA (Over the Air), que permite carregar programas para o ESP32 usando uma conexão WiFi em vez de se utilizar um cabo entre o computador e o microcontrolador. Este tipo de funcionalidade é essencial para este projeto porque a estrutura do iModBot não permite uma conexão física entre o computador e o ESP32, visto que a sua entrada USB está bloqueada pela caixa das baterias, e assim não é necessário estar a tirar e pôr o ESP32 na placa sempre que seja preciso inserir um novo programa.

1º Passo: Conecte o ESP32 ao seu computador através de um cabo micro-usb.

2º Passo: Vá a “Ferramentas” e selecione o modelo da placa ESP32 e a respetiva porta COM.

Uma imagem com texto, eletrónica, captura de ecrã, software

Descrição gerada automaticamente

3º Passo: Copie o seguinte código para o Arduino IDE e nele mude o “ssid” e a “password” pelo nome da rede e a sua palavra-passe, respetivamente:

|  |
| --- |
| #include <WiFi.h>  #include <ESPmDNS.h>  #include <WiFiUdp.h>  #include <ArduinoOTA.h>  // Replace with your network credentials  const char\* ssid = "YOUR\_SSID";  const char\* password = "YOUR\_PASSWORD";  void setup() {  Serial.begin(115200);  Serial.println("Booting");  WiFi.mode(WIFI\_STA);  WiFi.begin(ssid, password);  while (WiFi.waitForConnectResult() != WL\_CONNECTED) {  Serial.println("Connection Failed! Rebooting...");  delay(5000);  ESP.restart();  }  // Port defaults to 3232  // ArduinoOTA.setPort(3232);  // Hostname defaults to esp3232-[MAC]  // ArduinoOTA.setHostname("myesp32");  // No authentication by default  // ArduinoOTA.setPassword("admin");  // Password can be set with it's md5 value as well  // MD5(admin) = 21232f297a57a5a743894a0e4a801fc3  // ArduinoOTA.setPasswordHash("21232f297a57a5a743894a0e4a801fc3");  ArduinoOTA  .onStart([]() {  String type;  if (ArduinoOTA.getCommand() == U\_FLASH)  type = "sketch";  else // U\_SPIFFS  type = "filesystem";  // NOTE: if updating SPIFFS this would be the place to unmount SPIFFS using SPIFFS.end()  Serial.println("Start updating " + type);  })  .onEnd([]() {  Serial.println("\nEnd");  })  .onProgress([](unsigned int progress, unsigned int total) {  Serial.printf("Progress: %u%%\r", (progress / (total / 100)));  })  .onError([](ota\_error\_t error) {  Serial.printf("Error[%u]: ", error);  if (error == OTA\_AUTH\_ERROR) Serial.println("Auth Failed");  else if (error == OTA\_BEGIN\_ERROR) Serial.println("Begin Failed");  else if (error == OTA\_CONNECT\_ERROR) Serial.println("Connect Failed");  else if (error == OTA\_RECEIVE\_ERROR) Serial.println("Receive Failed");  else if (error == OTA\_END\_ERROR) Serial.println("End Failed");  });    ArduinoOTA.begin();  Serial.println("Ready");  Serial.print("IP address: ");  Serial.println(WiFi.localIP());  }  void loop() {  ArduinoOTA.handle();  } |

4º Passo: Pressione em “Envio”, que é o butão com uma seta a apontar para a direira no canto superior esquerdo, e ao mesmo tempo que está a enviar o programa pressione o botão “Boot” do ESP32 para que ele permitir carregar o programa.

Uma imagem com texto, eletrónica, captura de ecrã, software

Descrição gerada automaticamente

5º Passo: Abra o “Monitor série” do Arduino IDE, que é o botão com uma lupa no canto superior direito, pressione o botão “EN” do ESP32 para fazer reset e se inseriu as credenciais certa da rede deverá aparecer o endereço IP do ESP.

Uma imagem com texto, captura de ecrã, software, número

Descrição gerada automaticamente

6º Passo: Agora que o ESP32 está preparado para receber programas OTA, desconecte o microcontrolador do computador, insira-o na placa do iModBot e ligue o robô.

7º Passo: Abra o Arduino IDE, selecione “Ferramentas” e selecione a Porta de rede do ESP32, que deverá ter um aspeto semelhante ao da imagem abaixo:

Uma imagem com texto, eletrónica, captura de ecrã, software

Descrição gerada automaticamente

8º Passo: Com o ESP32 conectado via Wifi basta inserir o código. Para que o código seja compatível com o OTA é necessário inserir o código em si no código básico do OTA. Em baixo está o código do tutorial de controlo de movimentos simples do iModBot sem e com as propriedades do OTA, respetivamente.

|  |
| --- |
| *Exemplo controlo de movimentos simples* #include <**iModBot**.h>  **iModBot** robot;  void setup()  {   robot.begin();  }  void loop()  {   // Velocidade   // Pode variar de 1 a 255   byte speed = 255;     // Para seguir em frente   robot.forward(speed);   delay(500);   // Para recuar   robot.reverse(speed);   delay(500);   // Rodar para a direita   robot.rotateRight(speed);   delay(500);   // Rodar para a esquerda   robot.rotateLeft(speed);   delay(500);   // Parar os motores   robot.stopMotors();     delay(1000);  } |

|  |
| --- |
| *Exemplo controlo de movimentos simples com OTA* #include <WiFi.h>  #include <ESPmDNS.h>  #include <WiFiUdp.h>  #include <ArduinoOTA.h>  #include <**iModBot**.h>  **iModBot** robot;  // Replace with your network credentials  const char\* ssid = "YOUR\_SSID";  const char\* password = "YOUR\_PASSWORD";  void setup() {  Serial.begin(115200);  Serial.println("Booting");  WiFi.mode(WIFI\_STA);  WiFi.begin(ssid, password);  while (WiFi.waitForConnectResult() != WL\_CONNECTED) {  Serial.println("Connection Failed! Rebooting...");  delay(5000);  ESP.restart();  }  // Port defaults to 3232  // ArduinoOTA.setPort(3232);  // Hostname defaults to esp3232-[MAC]  // ArduinoOTA.setHostname("myesp32");  // No authentication by default  // ArduinoOTA.setPassword("admin");  // Password can be set with it's md5 value as well  // MD5(admin) = 21232f297a57a5a743894a0e4a801fc3  // ArduinoOTA.setPasswordHash("21232f297a57a5a743894a0e4a801fc3");  ArduinoOTA  .onStart([]() {  String type;  if (ArduinoOTA.getCommand() == U\_FLASH)  type = "sketch";  else // U\_SPIFFS  type = "filesystem";  // NOTE: if updating SPIFFS this would be the place to unmount SPIFFS using SPIFFS.end()  Serial.println("Start updating " + type);  })  .onEnd([]() {  Serial.println("\nEnd");  })  .onProgress([](unsigned int progress, unsigned int total) {  Serial.printf("Progress: %u%%\r", (progress / (total / 100)));  })  .onError([](ota\_error\_t error) {  Serial.printf("Error[%u]: ", error);  if (error == OTA\_AUTH\_ERROR) Serial.println("Auth Failed");  else if (error == OTA\_BEGIN\_ERROR) Serial.println("Begin Failed");  else if (error == OTA\_CONNECT\_ERROR) Serial.println("Connect Failed");  else if (error == OTA\_RECEIVE\_ERROR) Serial.println("Receive Failed");  else if (error == OTA\_END\_ERROR) Serial.println("End Failed");  });    ArduinoOTA.begin();  Serial.println("Ready");  Serial.print("IP address: ");  Serial.println(WiFi.localIP());  robot.begin();  }  void loop() {  ArduinoOTA.handle();   // Velocidade   // Pode variar de 1 a 255   byte speed = 255;     // Para seguir em frente   robot.forward(speed);   delay(500);   // Para recuar   robot.reverse(speed);   delay(500);   // Rodar para a direita   robot.rotateRight(speed);   delay(500);   // Rodar para a esquerda   robot.rotateLeft(speed);   delay(500);   // Parar os motores   robot.stopMotors();     delay(1000); } |