# **[Demo 19: How to use UDP/IP with Arduino ESP32](http://www.iotsharing.com/2017/06/how-to-use-udpip-with-arduino-esp32.html)**

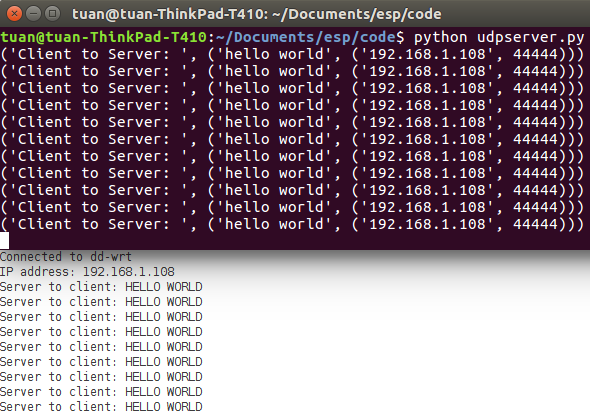
1. **Introduction**  
   - I made a TCP/IP demo in聽[Demo 8: How to use TCP/IP with Arduino ESP32](http://www.iotsharing.com/2017/05/tcp-udp-ip-with-esp32.html" \t "http://www.iotsharing.com/2017/06/_blank)聽so in this tutorial I will make a demo with UDP and apply tutorial聽[How to make IoT testing/debugging application (TCP, UDP, HTTP, MQTT) using Python](http://www.iotsharing.com/2017/06/how-to-make-iot-testing-application-using-python.html" \t "http://www.iotsharing.com/2017/06/_blank)聽for testing.  
   - Create a UDP server using Python and Arduino ESP32 UDP client. Client will send the data to server, server convert to upper case and respond to client.  
   **2. Hardware**  
   You do not need any extra hardware.  
   **3. Software**  
   - In order to make a UDp client in Arduino ESP32, we will use class聽**WiFiUDP**. It has some interfaces:  
   + begin(port): initialize UDP protocol and transfer buffer.  
   + beginPacket(udpAddress, udpPort): prepare sending data to server at IP and port.  
   + write(buffer, size of buffer): send data buffer with size of it.  
   + endPacket(): end of sending.  
   + parsePacket(): processing incoming packet, must be called before reading the buffer.  
   + read(buffer, size of buffer): read incoming data into buffer.  
   **3.1 Python server**

|  |
| --- |
| import socket  # bind all IP  HOST = '0.0.0.0'  # Listen on Port  PORT = 44444  #Size of receive buffer  BUFFER\_SIZE = 1024  # Create a TCP/IP socket  s = socket.socket(socket.AF\_INET, socket.SOCK\_DGRAM)  # Bind the socket to the host and port  s.bind((HOST, PORT))  while True:  # Receive BUFFER\_SIZE bytes data  # data is a list with 2 elements  # first is data  #second is client address  data = s.recvfrom(BUFFER\_SIZE)  if data:  #print received data  print('Client to Server: ' , data)  # Convert to upper case and send back to Client  s.sendto(data[0].upper(), data[1])  # Close connection  s.close() |

**3.2 Arduino ESP32 UDP client**

|  |
| --- |
| #include <WiFi.h>  #include <WiFiUdp.h>  /\* WiFi network name and password \*/  const char \* ssid = "dd-wrt";  const char \* pwd = "0000000000";  // IP address to send UDP data to.  // it can be ip address of the server or  // a network broadcast address  // here is broadcast address  const char \* udpAddress = "192.168.1.100";  const int udpPort = 44444;  //create UDP instance  WiFiUDP udp;  void setup(){  Serial.begin(115200);    //Connect to the WiFi network  WiFi.begin(ssid, pwd);  Serial.println("");  // Wait for connection  while (WiFi.status() != WL\_CONNECTED) {  delay(500);  Serial.print(".");  }  Serial.println("");  Serial.print("Connected to ");  Serial.println(ssid);  Serial.print("IP address: ");  Serial.println(WiFi.localIP());  //This initializes udp and transfer buffer  udp.begin(udpPort);  }  void loop(){  //data will be sent to server  uint8\_t buffer[50] = "hello world";  //send hello world to server  udp.beginPacket(udpAddress, udpPort);  udp.write(buffer, 11);  udp.endPacket();  memset(buffer, 0, 50);  //processing incoming packet, must be called before reading the buffer  udp.parsePacket();  //receive response from server, it will be HELLO WORLD  if(udp.read(buffer, 50) > 0){  Serial.print("Server to client: ");  Serial.println((char \*)buffer);  }  //Wait for 1 second  delay(1000);  } |

**4. Result**

[](https://2.bp.blogspot.com/-S_jqgSELaBQ/WTgt2-Rqd-I/AAAAAAAAEB8/WDNyoPyJMBIIwRmyipTf_V5PbW6V5cCQgCEw/s1600/esp32udp.png)

**Figure: UDP/IP with Arduino ESP32**