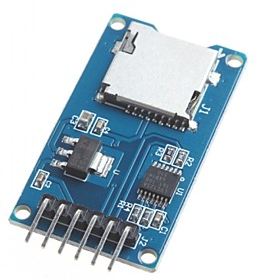
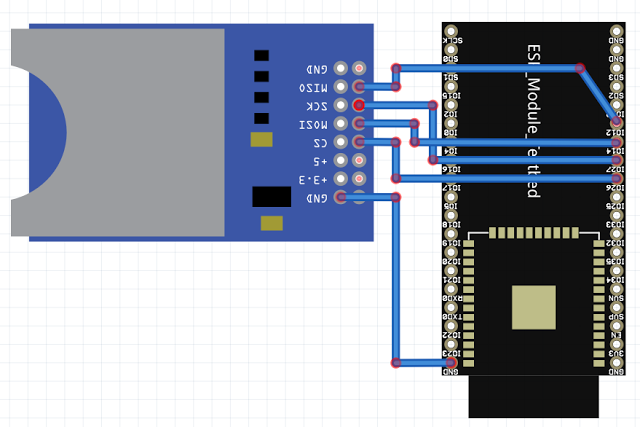
# **[Demo 7: How to use Arduino ESP32 to store data to sdcard](http://www.iotsharing.com/2017/05/how-to-use-arduino-esp32-to-store-data-to-sdcard.html)**

**[1.Introduction](http://www.iotsharing.com/2017/05/how-to-use-arduino-esp32-to-display-oled.html)** [The use of sdcard is to store the data. Because sdcard has big capacity so it can store more data comparing to EEPROM. Why sdcard is useful for ESP32? As you know ESP8266 and ESP32 are more popular with IoT application. They can keep the roles such as:  
 - A client to collect data from sensors such as temperature or humidity or log of an event, and save these data into sdcard.  
- A web server using sdcard to store web pages, images, javascript, … that will be returned to the client requests.  
Since micro sdcard (microSD) is popular so we will use micro sdcard for our demo. Here is the requirements:  
1. ESP32 connect to micro sdcard  
2. Traverse the micro sdcard to list all files and folders in it and print to Terminal.   
3. Open a file is named “test.txt” to write “Hello world!” text to it and close it  
4. Open “test .txt” again for reading the text then print to Terminal and close it.](http://www.iotsharing.com/2017/05/how-to-use-arduino-esp32-to-display-oled.html)

**[2. Hardware](http://www.iotsharing.com/2017/05/how-to-use-arduino-esp32-to-display-oled.html)** [In order to connect with sdcard we use the sdcard read-write module :](http://www.iotsharing.com/2017/05/how-to-use-arduino-esp32-to-display-oled.html)

[[](https://1.bp.blogspot.com/-vCx2UoxAUPw/WRo-qBsU1CI/AAAAAAAAD1Y/md4J8L1gu040DvOib9H8GYzcMgec-8w2ACLcB/s1600/sdcard1.png)](http://www.iotsharing.com/2017/05/how-to-use-arduino-esp32-to-display-oled.html)

**[Figure: module micro sdcard with SPI connection](http://www.iotsharing.com/2017/05/how-to-use-arduino-esp32-to-display-oled.html)**

[[](https://1.bp.blogspot.com/-zYdaT6b1p1g/WTF6o284m9I/AAAAAAAAD_s/8hrgs8ERMO0S3KjFxOm-LprdVgNDGFM-gCLcB/s1600/esp32sdcard.png)](http://www.iotsharing.com/2017/05/how-to-use-arduino-esp32-to-display-oled.html)

**[Figure: ESP32 connect microSD module](http://www.iotsharing.com/2017/05/how-to-use-arduino-esp32-to-display-oled.html)**

[Here we connect:](http://www.iotsharing.com/2017/05/how-to-use-arduino-esp32-to-display-oled.html)

[[ESP32 IO26 – CS MICROSD]  
[ESP32 IO14 – MOSI MICROSD]  
[ESP32 IO12 – MISO MICROSD]  
[ESP32 IO27 – SCK MICROSD]  
[ESP32 GND – GND MICROSD]  
[3.3V – VCC MICROSD]](http://www.iotsharing.com/2017/05/how-to-use-arduino-esp32-to-display-oled.html)**[3. Software](http://www.iotsharing.com/2017/05/how-to-use-arduino-esp32-to-display-oled.html)** [We will use library sdcard for communication. You can download it here:](http://www.iotsharing.com/2017/05/how-to-use-arduino-esp32-to-display-oled.html)<https://github.com/nhatuan84/esp32-micro-sdcard> [After downloading, unzip it and copy unzipped folder to libraries folder under Arduino folder:](http://www.iotsharing.com/2017/05/how-to-use-arduino-esp32-to-display-oled.html)**[C:/Users/[YOUR\_USER\_NAME]/Documents/Arduino/libraries](http://www.iotsharing.com/2017/05/how-to-use-arduino-esp32-to-display-oled.html)** [This library supplies some classes and interfaces:](http://www.iotsharing.com/2017/05/how-to-use-arduino-esp32-to-display-oled.html)**[Class SD:](http://www.iotsharing.com/2017/05/how-to-use-arduino-esp32-to-display-oled.html)** [- SD.begin(uint8\_t cs , int8\_t mosi , int8\_t miso , int8\_t sck): initialize library with SPI pins  
- SD.open(filename, FILE\_WRITE): open file for writing  
- SD.open(filename): open file for reading   
- SD.open("/"): open sdcard at root “/”](http://www.iotsharing.com/2017/05/how-to-use-arduino-esp32-to-display-oled.html)**[Class File:](http://www.iotsharing.com/2017/05/how-to-use-arduino-esp32-to-display-oled.html)** [- openNextFile(): traverse the directory  
- name(): get the name of file or directory  
- isDirectory(): check if entry is directory  
- size(): get size of file  
- close(): close the opened entry  
- println(text): write text to opened file  
- available(): check data available for reading  
- read(): if data is available then read data  
- close(): close the opened file   
Create an Arduino project and Save as esp32msdinfo with code:](http://www.iotsharing.com/2017/05/how-to-use-arduino-esp32-to-display-oled.html)

|  |
| --- |
| #include <mySD.h>  File root;  void setup()  {  Serial.begin(115200);  Serial.print("Initializing SD card...");  /\* initialize SD library with SPI pins \*/  if (!SD.begin(26, 14, 12, 27)) {  Serial.println("initialization failed!");  return;  }  Serial.println("initialization done.");  /\* Begin at the root "/" \*/  root = SD.open("/");  if (root) {  printDirectory(root, 0);  root.close();  } else {  Serial.println("error opening test.txt");  }  /\* open "test.txt" for writing \*/  root = SD.open("test.txt", FILE\_WRITE);  /\* if open succesfully -> root != NULL  then write string "Hello world!" to it  \*/  if (root) {  root.println("Hello world!");  root.flush();  /\* close the file \*/  root.close();  } else {  /\* if the file open error, print an error \*/  Serial.println("error opening test.txt");  }  delay(1000);  /\* after writing then reopen the file and read it \*/  root = SD.open("test.txt");  if (root) {  /\* read from the file until there's nothing else in it \*/  while (root.available()) {  /\* read the file and print to Terminal \*/  Serial.write(root.read());  }  root.close();  } else {  Serial.println("error opening test.txt");  }    Serial.println("done!");  }  void loop()  {  }  void printDirectory(File dir, int numTabs) {    while(true) {  File entry = dir.openNextFile();  if (! entry) {  break;  }  for (uint8\_t i=0; i<numTabs; i++) {  Serial.print('\t'); // we'll have a nice indentation  }  // Print the name  Serial.print(entry.name());  /\* Recurse for directories, otherwise print the file size \*/  if (entry.isDirectory()) {  Serial.println("/");  printDirectory(entry, numTabs+1);  } else {  /\* files have sizes, directories do not \*/  Serial.print("\t\t");  Serial.println(entry.size());  }  entry.close();  }  } |

1. **Result**

