048	**SD Card Reader**	Axel Chobe

Beschreibung:

Wer seinen Arduino als Datenlogger nutzt, um z.B. einen Temperaturverlauf auf zu zeichnen, steht schnell vor dem Problem, wie die erfassten Daten gespeichert werden sollen. Am Besten speichert man seine Daten daher auf einer Speicherkarte. Hier sind die Daten auch bei Stromausfall sicher gespeichert, außerdem lässt sich die Speicherkarte direkt mit dem PC verbinden und auslesen. Wenn man die Daten im richtigen Format speichert, kann man sie auch gleich mit Excel, Matlab oder ähnlichen Programmen auswerten.

Beschaltung:

MISO	Digital 12
Ground	Masse
SCK	Digital 13
MOSI	Digital 11
CS	Digital 4
VCC 5	5 Volt
VCC 3	3,3 Volt

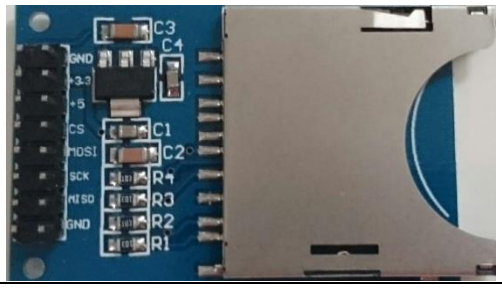
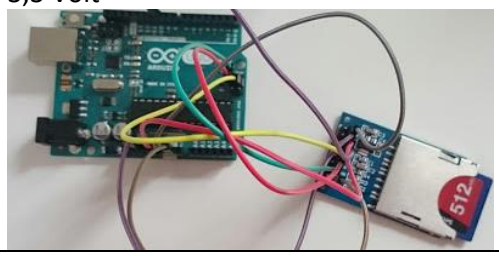


Bild:

3,3 Volt



Sketch:

```
#include <SD.h>
const int chipSelect = 4; // SELECTED_CHIP gibt an, welche Hardware beim SD-Shield verwendet wird.
                          // Den Wert erhält man vom Hersteller des SD-Shields.
void setup()
{
  Serial.begin(9600);
  Serial.print("Initialisisiere SD-Karte...");
  if (!SD.begin(chipSelect)) {   // Prüfen, ob die Karte vorhanden ist
   Serial.println("Keine Karte vorhanden");
   return; // dann tue nichts mehr
 }
 Serial.println("Karte initialisiert.");
}
void loop()
{
   String dataString = ""; // Zeichenkette für die Zusammenstellung der Daten
   for (int analogPin = 0; analogPin < 3; analogPin++) { // z.B. 3 Sensoren auslesen und an den String anhängen
   int sensor = analogRead(analogPin);
   dataString += String(sensor);
   if (analogPin < 2) {
     dataString += ",";
   } }
    File dataFile = SD.open("datalog.txt", FILE_WRITE);
// Öffnen Sie die Datei. Beachten Sie, dass immer nur eine Datei gleichzeitig geöffnet sein kann,
   if (dataFile) { //Wenn die Datei verfügbar ist, schreiben Sie in die Datei:
   dataFile.println(dataString);
   dataFile.close();
       Serial.println(dataString); // auch auf die serielle Schnittstelle drucken
 }
   else {  // Wenn die Datei nicht geöffnet ist, wird ein Fehler angezeigt:
   Serial.println("error opening datalog.txt");
 } }
```

Bibliothek: SD.h aus SD-master.zip

10 (SS) "Slave Select" (Slave-Auswahl)
11 (MOSI) "Master Out Slave In" (Master Out Slave In)
12 (MISO) "Master In Slave Out" (Master In Slave Out)
13 (SCK) "Systemuhr" (SCK)

In der setup()-Funktion wird zuerst eine serielle Verbindung aufgebaut, mit dem Befehl SD.begin(chipSelect) wird dann die SD-Karte initialisiert. Als Parameter wird hier der Pin übergeben, an dem CS angeschlossen ist. War die Initialisierung erfolgreich, geht es in den loop()-Funktion weiter. Hier werden die Werte der analogen Eingänge ausgelesen und in einer String-Variable gespeichert . Mit dem Befehl SD.open() wird dann die Datei "datalog.txt" im Schreib-Modus geöffnet. Ist das Öffnen erfolgreich, wird der Wert der String-Variable in die Datei geschrieben. Dann wird die Datei wieder geschlossen.

Ähnlich einfach ist das Auslesen von Dateien.

```
dataFile = SD.open("test.txt");
if (dataFile) {
  Serial.println("test.txt:");

  // read from the file until there's nothing else in it:
  while (dataFile.available()) {
   Serial.write(dataFile.read());
  }
  // close the file:
  data.close();
} else {
 // if the file didn't open, print an error:
  Serial.println("error opening test.txt");
}
```