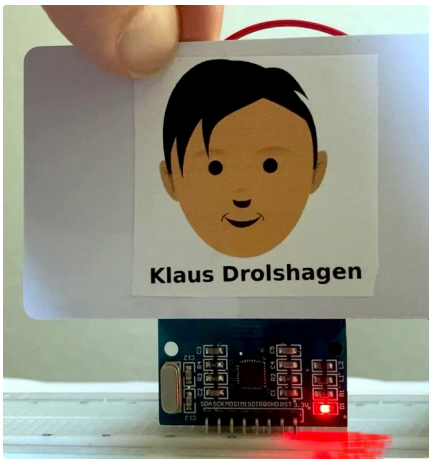
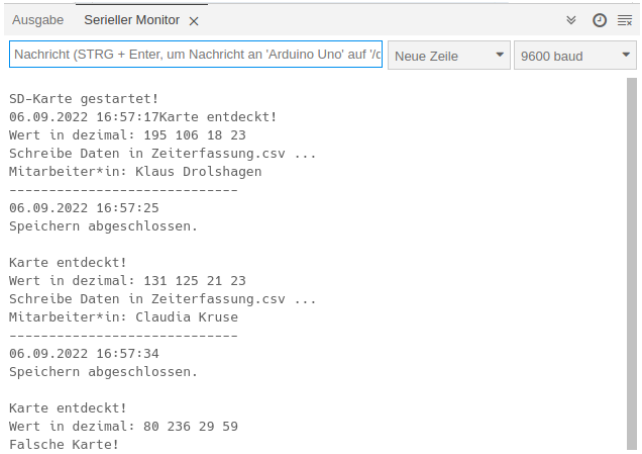
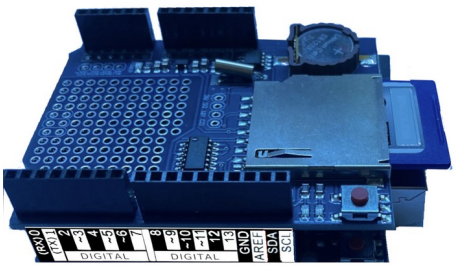
Mit einem RFID-Modul, einem RTC-Modul und einer SD-Karte sollen Datum, Zeit und der Name der Mitarbeiterin/des Mitarbeiters erfasst werden und auf einer SD-Karte gespeichert werden. Außerdem werden die Daten im Seriellen Monitor angezeigt.





Ein RFID-Modul (radio-frequency identification) besteht aus sogenannten „Tags“, und einem Lesegerät.

Tags

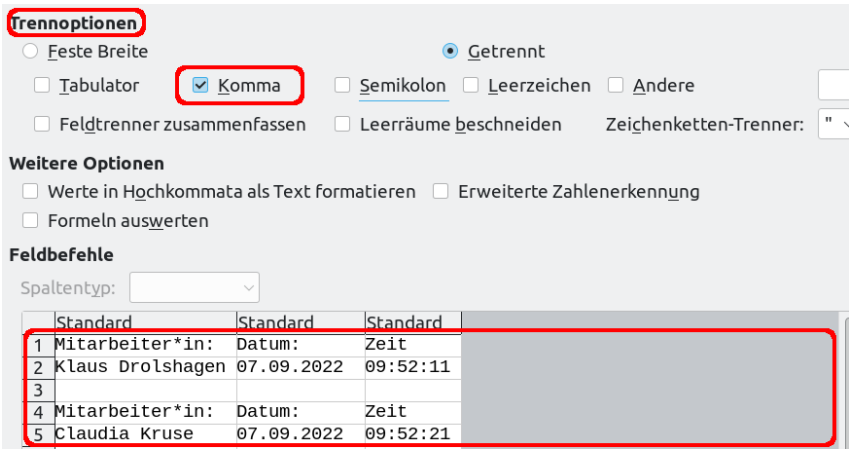


Ein Datenlogger-Shield vereinfacht die Verkabelung. Das Shield bietet ein Modul zur Zeiterfassung (RTC-Modul: Real Time Clock) und eine SD-Karte. Es wird einfach auf den Arduino aufgesteckt.

Die Daten werden im CSV-Format (Comma-separated values) gespeichert. Die Daten – hier der Name der Mitarbeiterin/des Mitarbeiters, das Datum und die Uhrzeit werden durch Kommas getrennt, anschließend folgt ein Zeilenvorschub (println).

Jede Tabellenkalkulation kann die Daten lesen. Beim Öffnen der Datei wird gefragt, welches Trennzeichen verwendet werden soll:

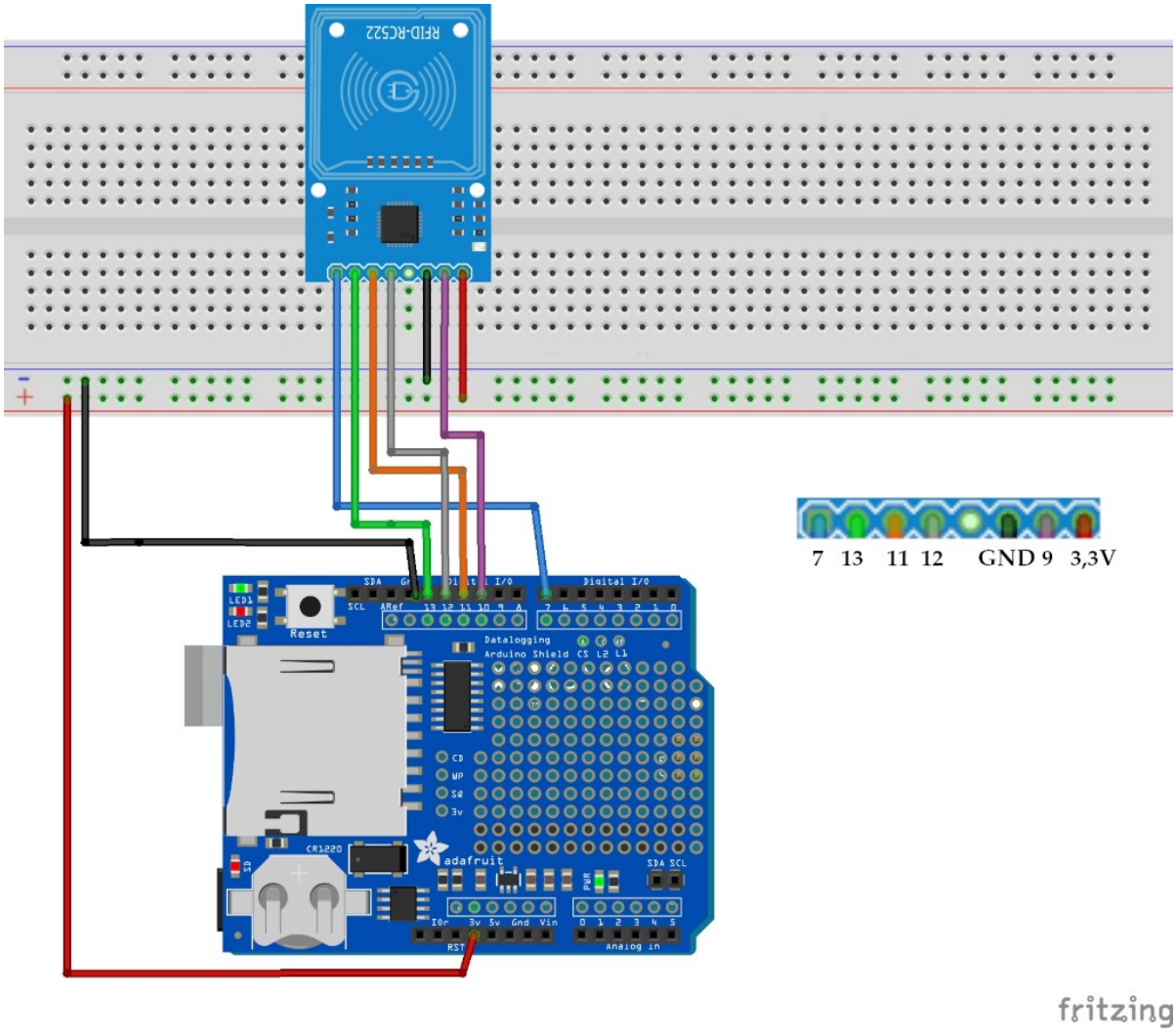
(Beispiel: LibreOffice Calc)



Benötigte Bauteile:

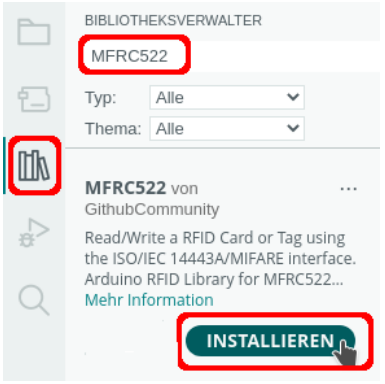
- RFID-Modul
- kodierte Karten
- Datenlogger-Shield
- Leitungsdrähte

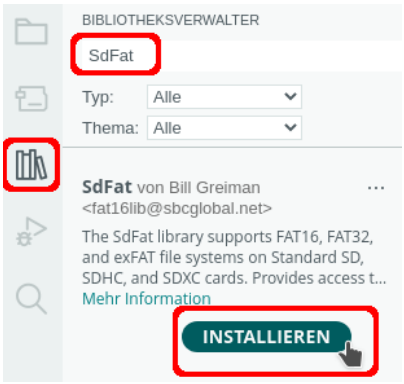
Baue die Schaltung auf.

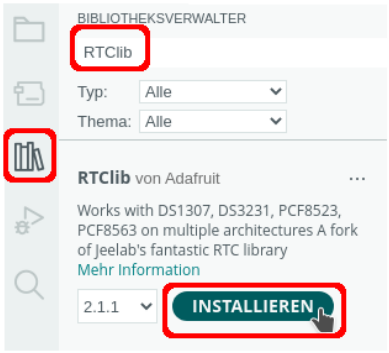


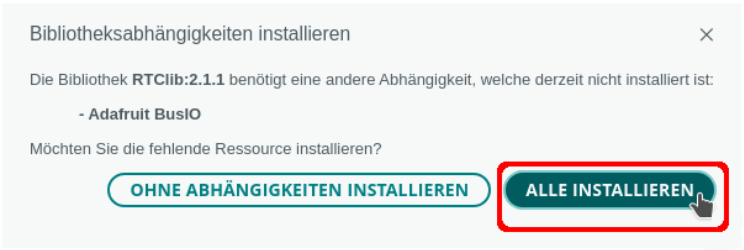
Benötigte Bibliotheken:

Sketch → Bibliothek einbinden → Bibliotheken verwalten









Zunächst musst du die individuelle Kennung der Karte mit diesem Programm herausfinden. Die Kennung wird im Seriellen Monitor angezeigt.

```
// RFID-Bibiothek hinzufügen
# include <MFRC522.h>

// Anschlüsse definieren
# define SDA 7
# define RST 9

// RFID-Empfänger benennen
MFRC522 mfrc522(SDA, RST);
```

```
void setup()
{
  Serial.begin(9600);

  // Initialisierung des RFID-Empfängers
  mfrc522.PCD_Init();
}

void loop()
{
  String WertDEZ;

  // Wenn keine Karte in Reichweite ist ..
  if (!mfrc522.PICC_IsNewCardPresent())
  {
    // .. wird die Abfrage wiederholt.
    return;
  }

  // Wenn kein RFID-Sender ausgewählt wurde ..
  if (!mfrc522.PICC_ReadCardSerial())
  {
    // .. wird die Abfrage wiederholt.
    return;
  }

  Serial.println(F("Karte entdeckt!"));

  // Dezimal-Werte in String schreiben
  for (byte i = 0; i < mfrc522.uid.size; i++)
  {
    WertDEZ = WertDEZ + String(mfrc522.uid.uidByte[i], DEC) + " ";
  }

  // Kennung dezimal anzeigen
  Serial.println("Wert in dezimal: " + WertDEZ);

  delay(1000);
}
```

Das eigentliche Programm

Um den Speicherplatz des Arduino UNO zu sparen, wird jede print() oder println() Anweisung, die normalen Text enthält, mit dem F-Makro ausgeführt. Das F-Makro sorgt dafür, dass der Text nicht im Programmspeicher verbleibt und so Speicherplatz im Programmspeicher zur Verfügung stellt.

Binde die Bibliotheken ein und definiere die Variablen:

```
# include <MFRC522.h>
# include <SPI.h>
# include <SdFat.h>
# include <RTClib.h>
```

```
// Name des RTC-Moduls (rtc)
RTC_DS3231 rtc;

// Bezeichnung der SD-Karte
SdFat SD;

// Bezeichnung der CSV-Datei
File Zeiterfassung;

// Anschlüsse RFID definieren
# define SDA 7
# define RST 9

// RFID-Empfänger benennen und Datenpins zuordnen
MFRC522 mfrc522(SDA, RST);

// Daten-Pin SD-Karte
int DatenPin = 10;

// Trennzeichen für die CSV-Datei
const String TrennZeichen = ",";

String Mitarbeiter;
```

Der setup-Teil. Beachte die Kommentare:

```
void setup()
{
  Serial.begin(9600);
  SPI.begin();

  // Initialisierung des RFID-Empfängers
  mfrc522.PCD_Init();

  /*
    wenn Datum und Zeit nicht korrekt -> Datum/Zeit setzen
    Jahr, Monat, Tag, Stunde, Minute, Sekunde
    Beispiel:  2022 August 31. 10 Uhr 30 Minuten 30 Sekunden
    rtc.adjust(DateTime(2022, 8, 31, 10, 31, 30));
 */

  // RTC-Modul starten
  rtc.begin();

  // SD-Karte starten
  if (!SD.begin(DatenPin))
  {
    Serial.println(F("Initialisierung fehlgeschlagen!"));
  }
  else Serial.println("SD-Karte gestartet!");

  // rtc.now() -> aktuelle Zeit holen
  DateTime aktuell = rtc.now();
```

```
  // Format des Datums festlegen
  char Datum[] = "DD.MM.YYYY";

  // Datum in Zeichenkette (String) umwandeln und anzeigen
  Serial.print(aktuell.toString(Datum));

  // Format der Zeitangabe festlegen
  char Zeit[] = "hh:mm:ss";

  // Zeitangabe in Zeichenkette (String) umwandeln und anzeigen
  Serial.print(aktuell.toString(Zeit));
}
```

Der loop-Teil. Beachte die Kommentare.

```
void loop()
{
  // vom RFID ermittelter Wert der Karte als Folge von Dezimalzahlen
  String WertDEZ;

  // String Mitarbeiter löschen, damit falsche Karte erkannt wird
  Mitarbeiter = "";

  // Wenn keine Karte in Reichweite ist ..
  if (!mfrc522.PICC_IsNewCardPresent())
  {
    // .. wird die Abfrage wiederholt.
    return;
  }

  // Wenn kein RFID-Sender ausgewählt wurde ..
  if (!mfrc522.PICC_ReadCardSerial())
  {
    // .. wird die Abfrage wiederholt.
    return;
  }
  Serial.println(F("Karte entdeckt!"));

  // Dezimal-Werte in String WertDEZ schreiben
  for (byte i = 0; i < mfrc522.uid.size; i++)
  {
    WertDEZ = WertDEZ + String(mfrc522.uid.uidByte[i], DEC) + " ";
  }

  // dezimalen Wert anzeigen
  Serial.println("Wert in dezimal: " + WertDEZ);

  // Leerzeichen am Ende entfernen
  WertDEZ.trim();

  /*
    Karten abfragen und Namen der Mitarbeiter*innen zuordnen
    Dezimal-Werte durch die eigenen Werte ersetzen
  */
```

```
  if (WertDEZ == "195 106 18 23") Mitarbeiter = "Klaus Drolshagen";
  if (WertDEZ == "227 77 233 22") Mitarbeiter = "Eva Hilger";
  if (WertDEZ == "131 125 21 23") Mitarbeiter = "Claudia Kruse";
  /*
    Namen abfragen
    wenn (if) Mitarbeiter == Klaus Drolshagen oder (||) Mitarbeiter == Eva Hilger || . .
    werden Die Daten angezeigt und in die Datei geschrieben
  */
  if (Mitarbeiter == "Klaus Drolshagen" || Mitarbeiter == "Eva Hilger" || Mitarbeiter == "Claudia Kruse")
  {
    // Daten in die Datei schreiben
    DatenSchreiben();
  }

  // falsche Karte
  else
  {
    Serial.println(F("Falsche Karte!"));
  }

  // Pause, damit die Daten nicht mehrfach in die Datei geschrieben werden
  delay(1000);
}
```

Jetzt fehlt noch die Methode DatenSchreiben(). Beachte die Kommentare.

```
void DatenSchreiben()
{
  // Meldungen im Seriellen Monitor
  Serial.println(F("Schreibe Daten in Zeiterfassung.csv ... "));

  // Name der Mitarbeiterin/des Mitarbeiters
  Serial.println("Mitarbeiter*in: " + Mitarbeiter);
  Serial.println(F("-----------------------------"));

  // rtc.now() -> aktuelle Zeit holen
  DateTime aktuell = rtc.now();
  char Datum[] = "DD.MM.YYYY ";

  // Datum in Zeichenkette (String) umwandeln und anzeigen
  Serial.print(aktuell.toString(Datum));

  // Format der Zeitangabe festlegen
  char Zeit[] = "hh:mm:ss";

  // Zeitangabe in Zeichenkette (String) umwandeln und anzeigen
  Serial.println(aktuell.toString(Zeit));

  /*
    Zeiterfassung ist das Datei-Objekt
    FILE_WRITE -> Datei zum Schreiben öffnen
```

```
    wenn die Datei noch nicht existiert, wird sie neu erstellt
    ansonsten werden die neuen Daten ans Ende angehängt
  */
  Zeiterfassung = SD.open("Zeiterfassung.csv", FILE_WRITE);
  if (Zeiterfassung)
  {
    // Überschrift nach jedem Datensatzwiederholen
    Zeiterfassung.print(F("Mitarbeiter*in:"));
    Zeiterfassung.print(TrennZeichen);
    Zeiterfassung.print(F("Datum:"));
    Zeiterfassung.print(TrennZeichen);
    Zeiterfassung.print(F("Zeit"));
    Zeiterfassung.println();

    // Daten in die Datei schreiben: Mitarbeiter*in, Datum und Zeit
    Zeiterfassung.print(Mitarbeiter);
    Zeiterfassung.print(TrennZeichen);
    Zeiterfassung.print(Datum);
    Zeiterfassung.print(TrennZeichen);
    Zeiterfassung.print(Zeit);
    Zeiterfassung.println();
    Zeiterfassung.println();
  }

  // Datei schließen
  Zeiterfassung.close();
  Serial.println(F("Speichern abgeschlossen."));
  Serial.println();
}
```

Anhang:

Quelle: https://openclipart.org/detail/20698/smooth-faces