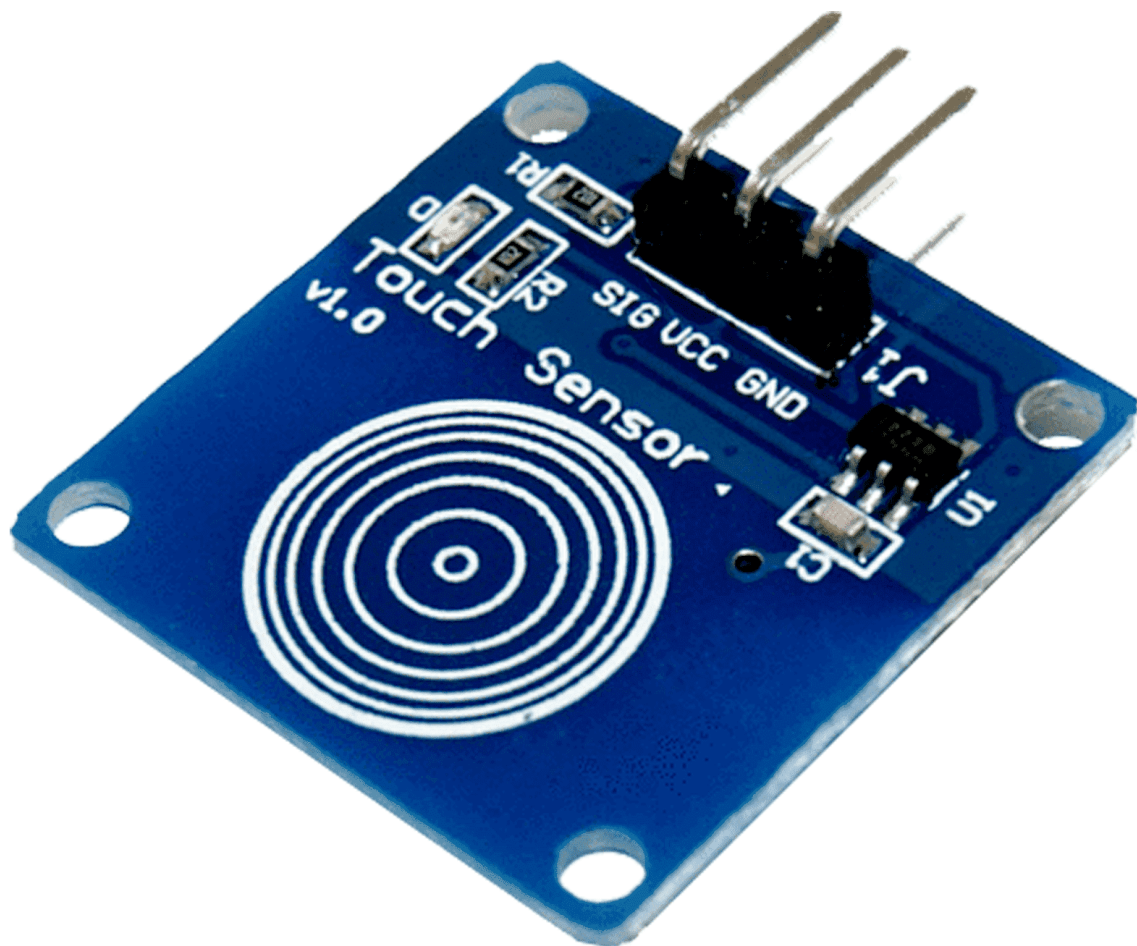


AZ-Delivery

Willkommen!

Vielen Dank, dass sie sich für unseren *TTP223B digitalen kapazitiven Touchsensor* von *AZ-Delivery* entschieden haben. In den nachfolgenden Seiten werden wir Ihnen erklären wie Sie das Gerät einrichten und nutzen können.

Viel Spaß!



Az-Delivery

Zur Steuerung elektronischer Geräte ist es besser, digitale Schalter statt elektrischer zu verwenden (elektronische Schalter verursachen bei Berührung mit nassen Händen einen elektronischen Schlag). TTP223B IC-basierte digitale kapazitive Sensoren sind sehr ökonomisch und geben bei Berührung ein gutes Feedback. Dieses Sensor-Breakout kann mit jeder Art von Mikrocontrollern verbunden werden und besitzt nur drei Terminals als externe Schnittstellen.

Ein kapazitiver Touchsensor basiert auf dem dedizierten IC-Berührungssensor TTP223B. Das Modul bietet eine integrierte Sensorfläche von 11 x 11 mm. Eine integrierte LED zeigt visuell an, dass der Sensor ausgelöst wird. Wenn er ausgelöst wird, wechselt der Modulausgang vom LOW-Zustand in den HIGH-Zustand.

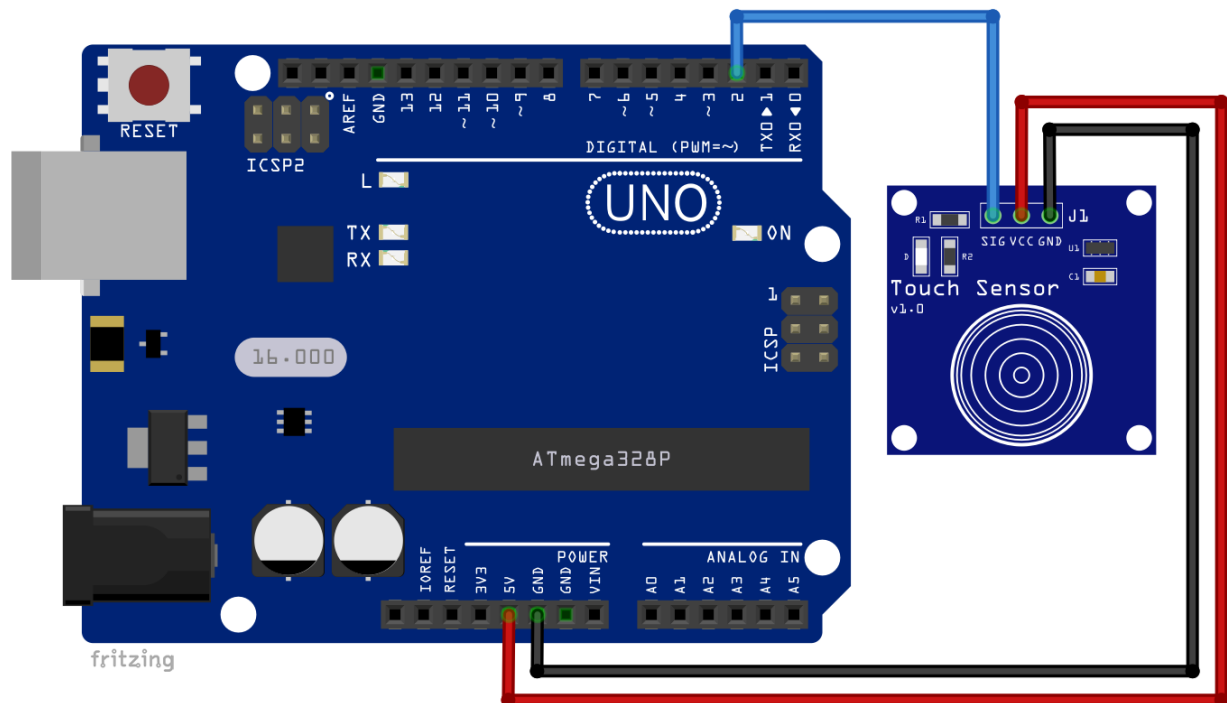
Der TTP223 ist ein Tasten-Touchpad-Detektor-IC und eignet sich zur Erkennung von einer Variation kapazitiver Elementen. Er verbraucht sehr wenig Strom und die Betriebsspannung liegt zwischen 2,0V und 5,5V.

Technische Daten:

- » Stromversorgungs- und Logikspannungsbereich:: 2.0V bis 5.5V DC
- » Betriebstemperatur: -20°C bis 70°C
- » Dimensionen: 24 x 24mm
- » Stabile Berührungserkennung des menschlichen Körpers
- » Low-Power-Modus
- » Automatische Kalibrierung für die Lebensdauer

Az-Delivery

Verbindung des Moduls mit dem Arduino Uno



Sensor Pin > Uno Pin

SIG > D2

GND > GND

VCC > 5V

Blauer Draht

Schwarzer Draht

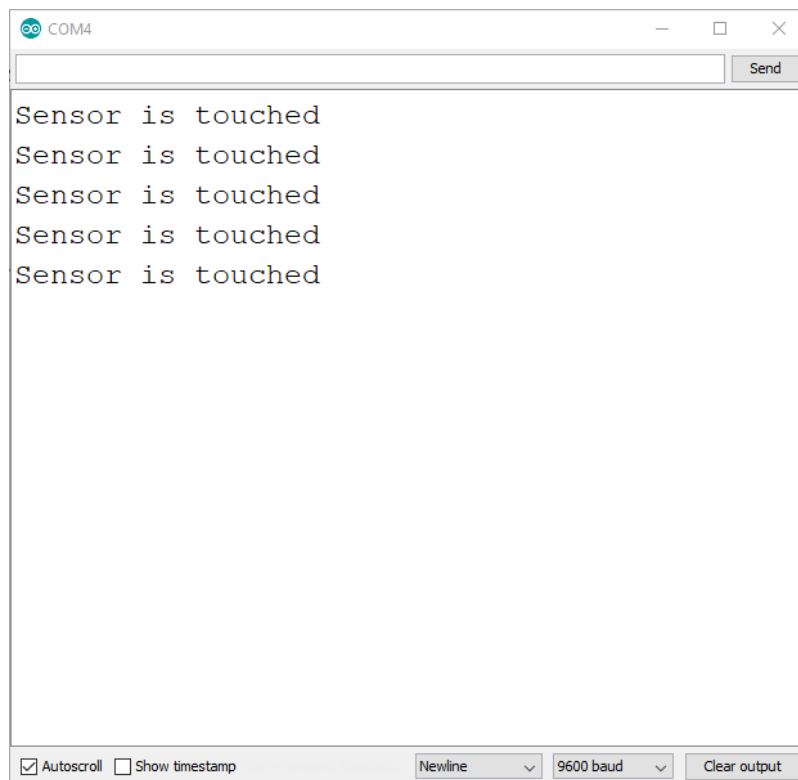
Roter Draht

AZ-Delivery

Sketch-Code:

```
void setup() {  
  pinMode(2, INPUT);  
  Serial.begin(9600);  
}  
void loop() {  
  if (digitalRead(2) == HIGH) {  
    Serial.println("Sensor is touched");  
  }  
  delay(500);  
}
```

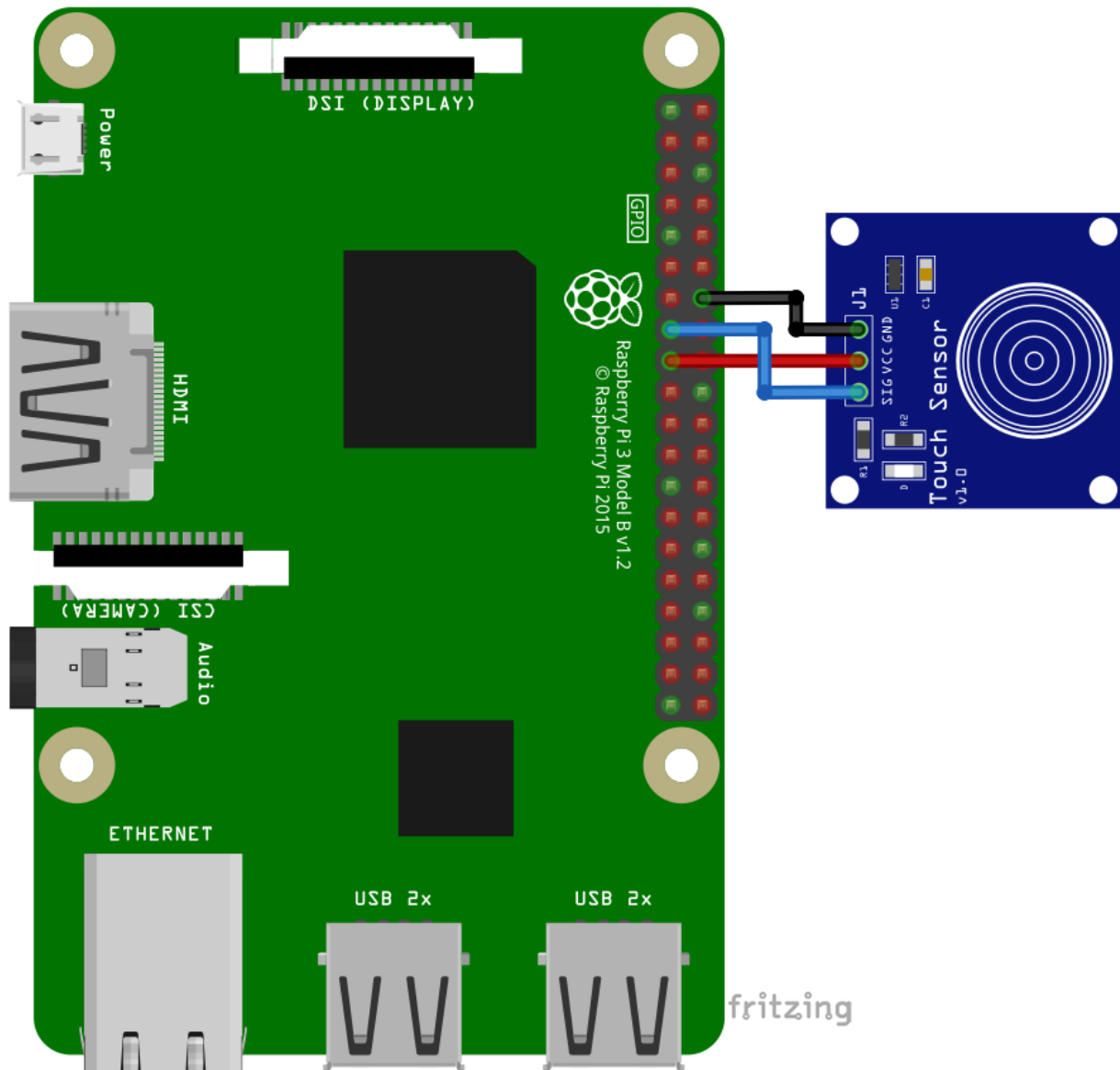
Laden Sie den Sketch auf den Arduino Uno und öffnen Sie den Serial Monitor (*Tools > Serial Monitor*). Die Ausgabe sollte wie folgt aussehen:



The screenshot shows the Serial Monitor window for COM4. The output consists of five lines of text: "Sensor is touched". The window has a "Send" button at the top right and a status bar at the bottom with the following options: Autoscroll, Show timestamp, Newline (dropdown), 9600 baud (dropdown), and Clear output.

AZ-Delivery

Verbindung des Moduls mit dem Raspberry Pi



Sensor Pin > Raspberry Pi Pin

GND	>	GND	[pin 14]
SIG	>	GPIO22	[pin 15]
VCC	>	3V3	[pin 17]

Schwarzer Draht

Blauer Draht

Roter Draht

Az-Delivery

Skript-Code:

```
import RPi.GPIO as GPIO
import time

GPIO.setmode(GPIO.BCM)
GPIO.setwarnings(False)

TOUCH = 22
GPIO.setup(TOUCH, GPIO.IN)

def touch(channel):
    print('Sensor is touched')

GPIO.add_event_detect(TOUCH, GPIO.RISING, callback=touch, bouncetime=200)

print('[press ctrl+c to stop the script]')
try:
    while True:
        time.sleep(0.0001)

except KeyboardInterrupt:
    print('Script end!')

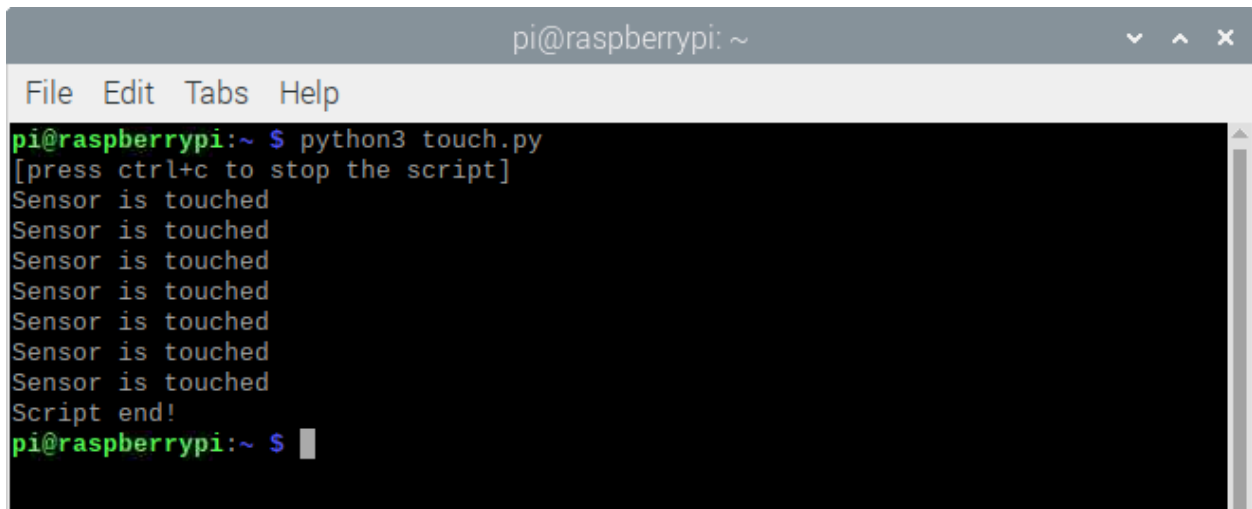
finally:
    GPIO.cleanup()
```

Az-Delivery

Speichern Sie dieses Skript unter dem Namen "*touch.py*" und starten Sie die Terminal App im selben Verzeichnis und führen diesen Befehl aus:

```
python3 touch.py
```

Die Ausgabe sollte wie folgt aussehen:



```
pi@raspberrypi: ~  
File Edit Tabs Help  
pi@raspberrypi:~ $ python3 touch.py  
[press ctrl+c to stop the script]  
Sensor is touched  
Sensor is touched  
Sensor is touched  
Sensor is touched  
Sensor is touched  
Sensor is touched  
Sensor is touched  
Sensor is touched  
Script end!  
pi@raspberrypi:~ $
```

Interrupts werden im Skript verwendet. Um einen Interrupt zu erzeugen, müssen wir die Funktion `add_event_detect()` verwenden. Die Funktion akzeptiert vier Argumente.

Als erstes wählen wir die GPIO-Pin-Nummer, für die wir den Interrupt verwenden wollen.

Das zweite Argument ist das Ereignis selbst, `RISING`, `FALLING` oder `BOTH`. Dies ist die Flanke des digitalen Signals. Wenn das Digitalsignal seinen Zustand von `LOW` auf `HIGH` ändert, haben wir eine `RISING`-Flanke, und bei `HIGH` auf `LOW`, haben wir eine `FALLING`-Flanke. Der Ausgang des Berührungssensors befindet sich auf `LOW`, wenn er nicht berührt wird, und der Sensor zeigt `HIGH` an, wenn er berührt wird. Dies ist die `RISING`-Flanke des digitalen Signals, die es zu einem Interrupt-Ereignis macht.

Az-Delivery

Das dritte Argument der Funktion `add_event_detect()` ist die Callback Funktion. Wenn die Unterbrechung (Interrupt) stattfindet, wird die Callback Funktion ausgeführt. In unserem Beispiel gibt die Callback Funktion die Meldung "*Sensor is touched*".

Das vierte Argument ist die *bouncetime*, die die Verzögerungszeit (Zeitintervall in Millisekunden) anzeigt. Manchmal geschehen Unterbrechungen zu schnell. Da kann es passieren, dass wir den Signalwechsel an einem bestimmten Pin verpassen. Deshalb müssen wir ein Zeitintervall, z.B. *100* oder *200* Millisekunden, abwarten, damit der Raspberry Pi die Signaländerung erkennen kann.

Das Skript beginnt mit dem Import von zwei Libraries, eine für GPIO-Pin-Namen und eine zweite für das Timing.

Dann stellen wir den Modus der GPIO-Suite ein und deaktivieren alle Warnungen bezüglich der GPIO-Schnittstellen.

Danach erstellen wir die Variable "*TOUCH*" und initialisieren sie mit der Nummer *22*. Diese Nummer steht für den GPIO-Pin-Namen, an dem der Ausgangspin des Berührungssensors angeschlossen ist. Dann setzen wir diesen Pin-Modus als *INPUT*.

Als nächstes erstellen wir eine Interrupt-Funktion, die ausgeführt wird, wenn ein Interrupt (Unterbrechung) auftritt.

Schließlich richten wir die Funktion `add_event_detect()` ein und erstellen einen Infinite loop Block (`while True:`).

Az-Delivery

Im Infinite Loop-Block des Codes unterbrechen wir den Code für 100 Mikrosekunden. Wenn wir dies nicht tun, blockiert der Code die CPU des Raspberry Pi, weil sie zu schnell laufen wird. Im Infinite Loop-Block brauchen wir nichts weiter zu tun, weil das ganze Skript auf den Signalwechsel am GPIO-Pin 22 wartet.

Um das Skript zu beenden, drücken Sie STRG + C. Dies wird Tastaturunterbrechung genannt, und wir warten darauf im Code-Ausnahmeblock:

```
except KeyboardInterrupt:  
    print('Script end!')
```

Wenn das Skript endet, wird der *finally* Codeblock ausgeführt. Dies ist der Zeitpunkt, an dem das Skript alle GPIO-Pins ausschaltet und alle mit GPIO-Pins verbundenen Schnittstellen trennt. Wir tun dies mit der folgenden Codezeile: `GPIO.cleanup()`

Sie haben es geschafft. Sie können jetzt unser Modul für Ihre Projekte nutzen.

Az-Delivery

Jetzt sind Sie dran! Entwickeln Sie Ihre eigenen Projekte und Smart-Home Installationen. Wie Sie das bewerkstelligen können, zeigen wir Ihnen unkompliziert und verständlich auf unserem Blog. Dort bieten wir Ihnen Beispielskripte und Tutorials mit interessanten kleinen Projekten an, um schnell in die Welt der Mikroelektronik einzusteigen. Zusätzlich bietet Ihnen auch das Internet unzählige Möglichkeiten, um sich in Sachen Mikroelektronik weiterzubilden.

Falls Sie nach weiteren hochwertigen Produkten für Arduino und Raspberry Pi suchen, sind Sie bei AZ-Delivery Vertriebs GmbH goldrichtig. Wir bieten Ihnen zahlreiche Anwendungsbeispiele, ausführliche Installationsanleitungen, E-Books, Bibliotheken und natürlich die Unterstützung unserer technischen Experten.

<https://az-delivery.de>

Viel Spaß!

Impressum

<https://az-delivery.de/pages/about-us>