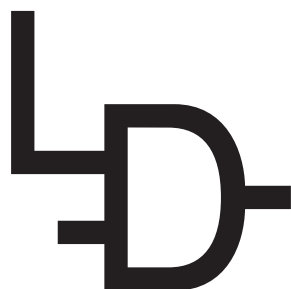


PCL Workshop

Kapazitive Berührungssensoren



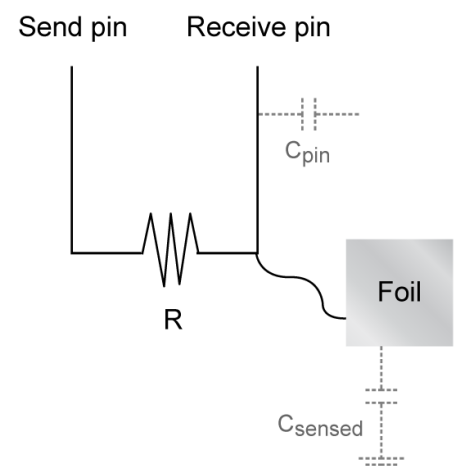
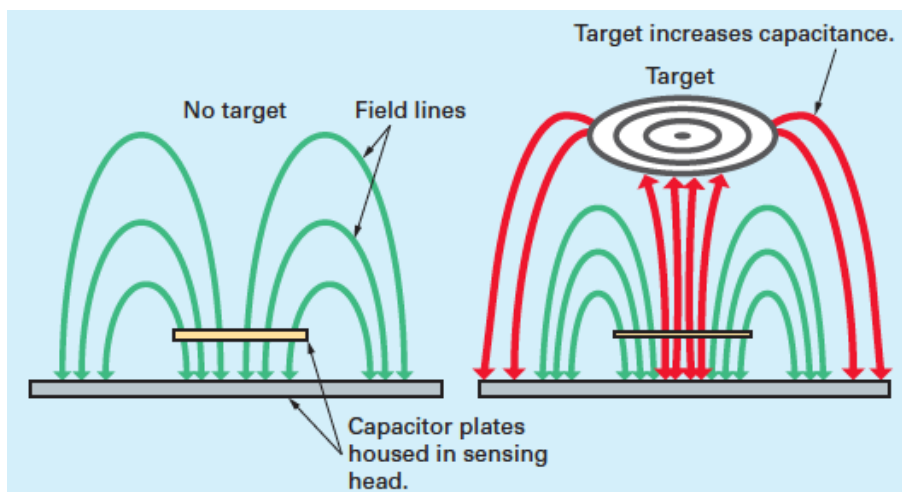
Kapazitive Berührungssensoren

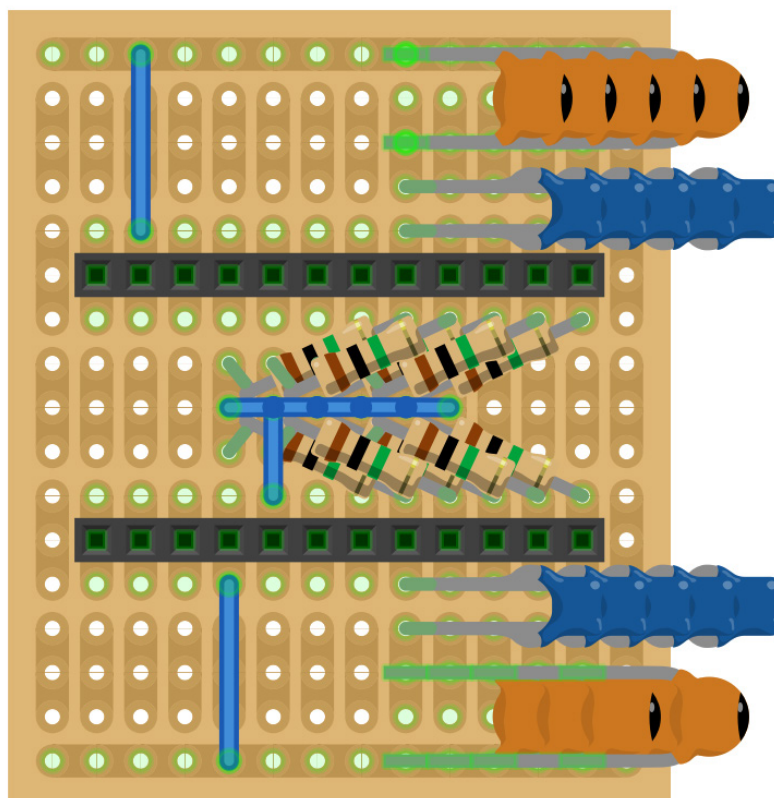
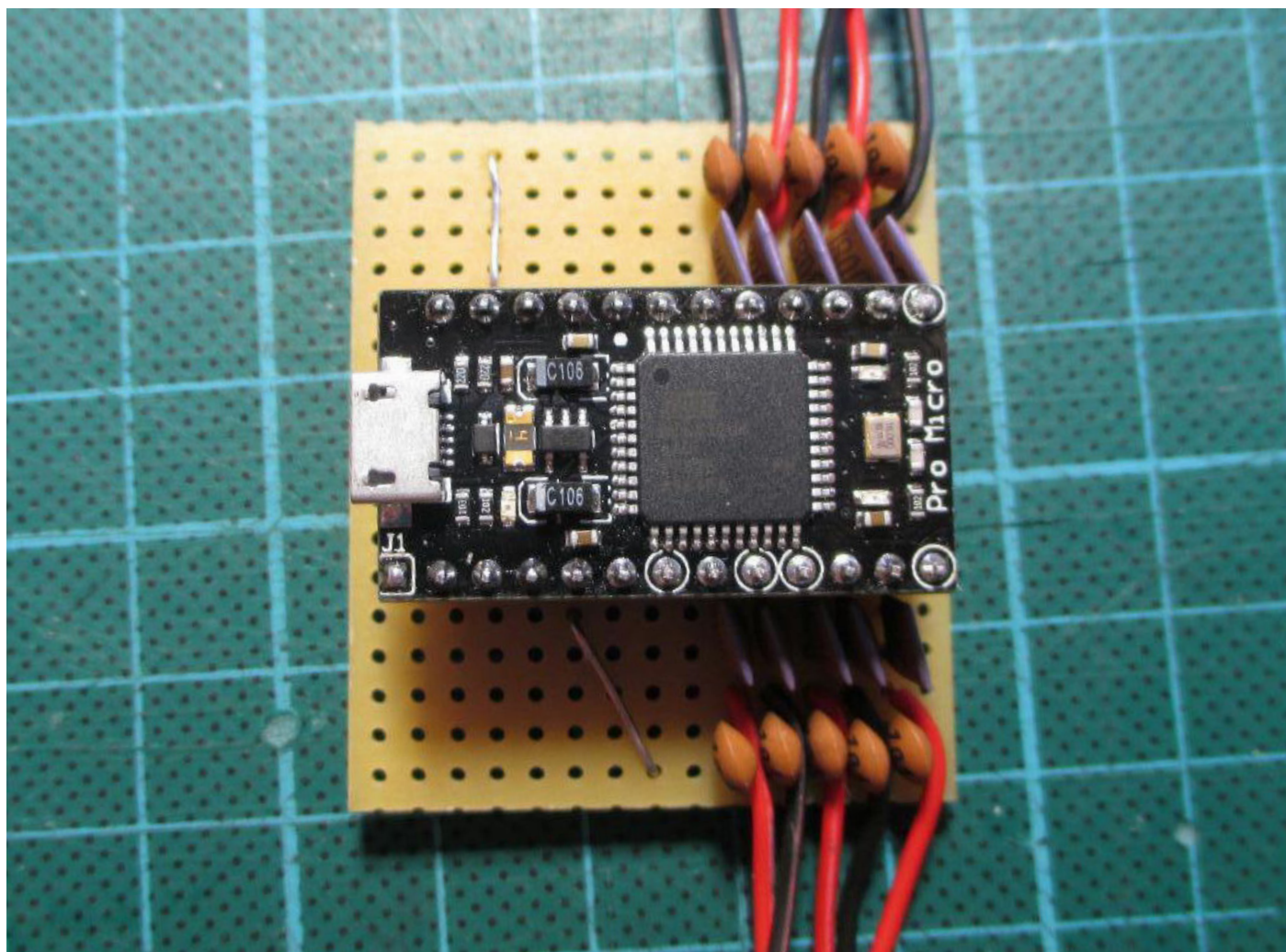
Beim kapazitiven Berührungssensor wird eine Ladungsänderung der Grundkapazität des Sensor-Pads erfasst, sobald der Benutzer seine Fingerspitze annähert. Im selbst-abtastenden Detektor fügt der angenäherte Finger mehr Kapazität zum Sensor hinzu. Diese Änderung erhöht die Zeitkonstante des Berührungssensor-Schaltkreises, die über einen Ladezeit-Messkreis erfasst wird, der in den Sensor-Controller integriert ist.

Gemessen wird die Zeit, bis der Kondensator geladen ist. Je größer die Kapazität, umso größer die Ladezeit. Darüber hinaus lässt sich die Sensitivität anpassen, indem der Ladestrom über den Speisewiderstand begrenzt wird. Je höher der Widerstand, um so geringer der Ladestrom und umso größer ist die neitliche Änderung bei Veränderung der Kapazität.

In der Beispielschaltung lgh ein vergleichsweise kleiner Widerstand von $1\text{M}\Omega$ verbaut, wodurch der Sensor weniger sensibel reagiert und erst bei Berührung eine größere Änderung zeigt. Zur Stabilisierung wird ein 180pF Kondensator in Serie und ein 100pF parallel zum Berührungssensor verbaut. Der Kondensator in Serie ist nicht zwingend notwendig, sorgt aber für verlässlichere Sensorwerte.

Um die Schaltung sensibler zu machen, kann der Speisewiderstand erhöht ($10\text{-}50\text{M}\Omega$), eine größere leitende Fläche am Sensorkontakt und zusätzlich eine geerdete Fläche hinter der Sensorfläche (elektrisch isoliert) verbaut werden.





Arduino Sketch Beispiel, Sensorpins zu Tastaturausgabe

```
#include <CapacitiveSensor.h>
#include <Keyboard.h>
// #include <Mouse.h>

/*
 * Simple capacitive touch board
 * Utilises »CapacitiveSensor« library by Paul Badger and Paul Stoffregen
 * by: Frederik Brückner
 * date: 2018-01-14
 * Atmega 32u4 (Arduino Leonardo, Pro Micro) based HID
 * Send keystrokes with touch electrodes
 */

boolean debug = false;           // debug mode enable/disable
const int threshold = 500;       // sensitivity threshold for touch detection
const int numSense = 10;         // number of sensors
const int sampleLength = 80;     // sample length in bytes
volatile long senseArray[numSense]; // sensor value array
unsigned long debounceDelay = 20; // software debounce time
unsigned long lastTouch = 0;      // when did the last touch register

CapacitiveSensor CS[numSense] = // sensor object array 1M resistor between
    pins 2 & sensor pin, add a wire and or foil
{
    CapacitiveSensor(2,5),
    CapacitiveSensor(2,6),
    CapacitiveSensor(2,7),
    CapacitiveSensor(2,8),
    CapacitiveSensor(2,9),
    CapacitiveSensor(2,10),
    CapacitiveSensor(2,16),
    CapacitiveSensor(2,14),
    CapacitiveSensor(2,15),
    CapacitiveSensor(2,18)
};

char keys[numSense] =           // char array to hold key stroke values
{
    '1',
    '2',
    '3',
    '4',
    '5',
    '6',
    '7',
    '8',
    '9',
    '0'
};
```

```

void setup()
{
  if (debug){
    Serial.begin(9600);
  }
  Keyboard.begin();
  //Mouse.begin();
  lastTouch = millis();
} // end setup

void loop()
{
  unsigned long start = millis();
  for (int i=0; i<numSense; i++){
    senseArray[i] = CS[i].capacitiveSensor(sampleLength);
  }
  if (debug){
    Serial.print(millis() - start); // check on performance in milliseconds
    Serial.println("\t"); // tab character for debug window spacing
    for (int i=0; i<numSense; i++){
      Serial.print("Sensor ");
      Serial.print(i);
      Serial.print("= ");
      Serial.println(senseArray[i]); // print sensor output
    }
    delay(10); // arbitrary delay to limit data to serial
               port
  } // end debug

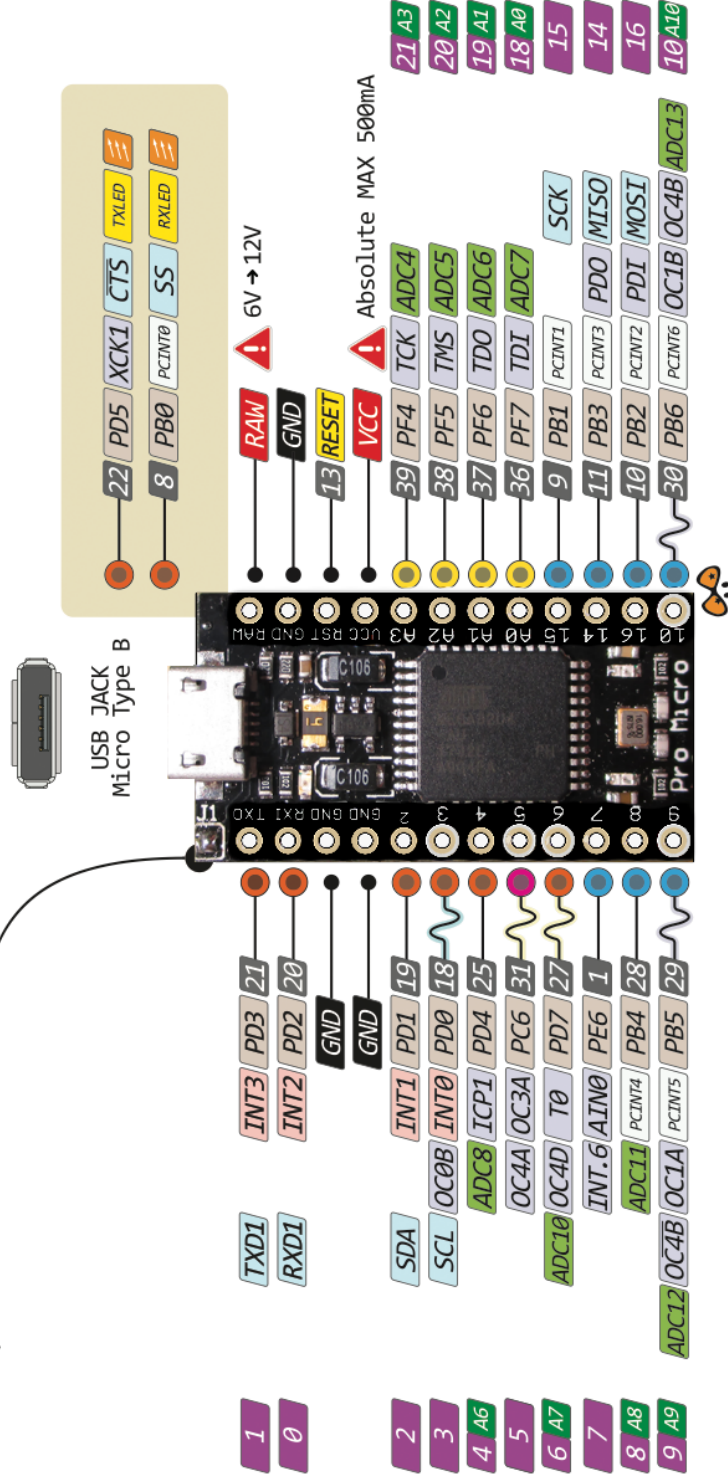
  for (int i=0; i<numSense; i++){
    if (senseArray[i] > threshold && (millis() - lastTouch) > debounceDelay){
      lastTouch = millis();
      Keyboard.print(keys[i]);
    }
  }
} // end loop

```


PROMICRO

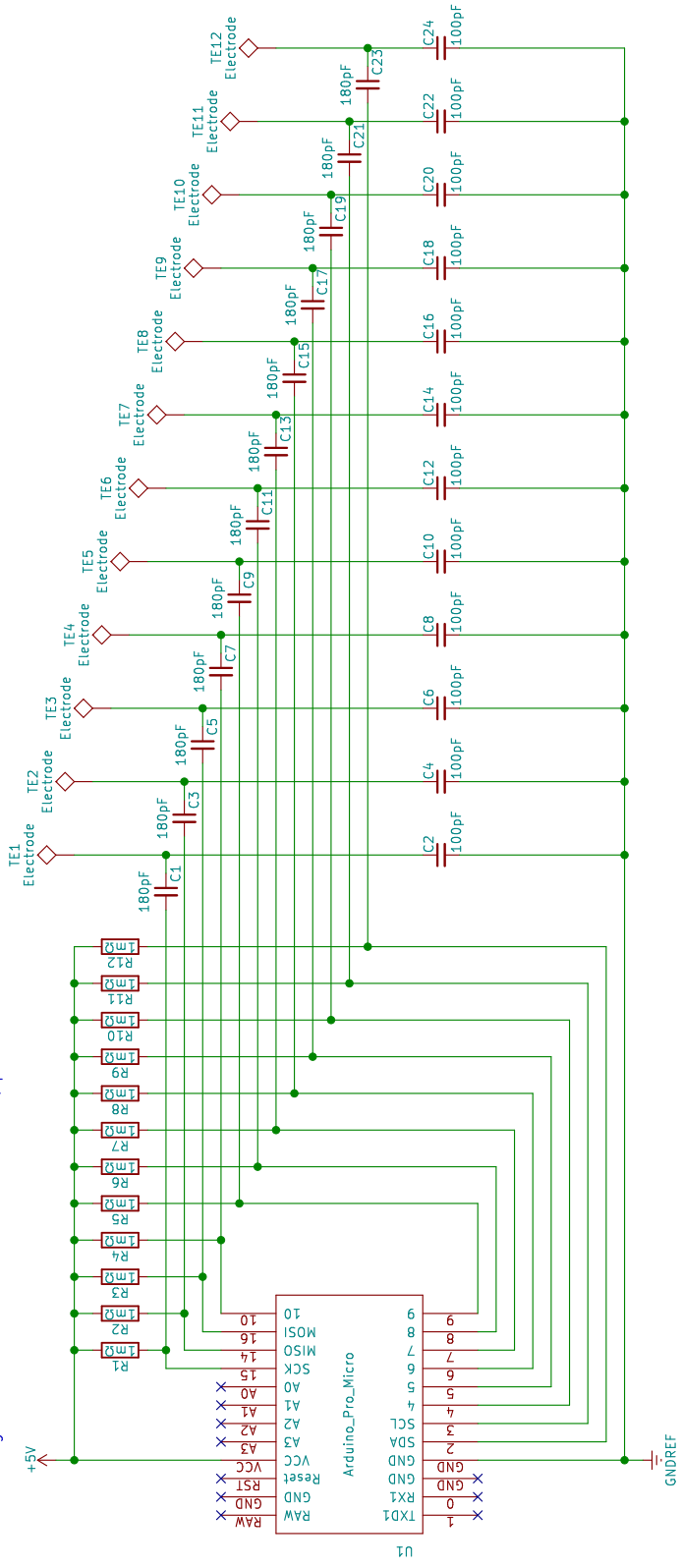
⚠ The power sum for each pin's group should not exceed 100mA

Closed for 5V boards
Open for 3.3V boards



- Power
- GND
- Serial Pin
- Analog Pin
- Control
- INT
- Physical Pin
- Port Pin
- Pin function
- Interrupt Pin
- PWM Pin
- Port Power

Change resistors R1 .. R12 to increase sensitivity.
 Higher values equal higher sense distance.
 100kΩ .. 1mΩ = measure physical touch
 1mΩ .. 10mΩ = touchless sensing up to 10cm away
 10mΩ .. 50mΩ = touchless sensing up to 30cm away
 Higher values make the sensors act slower, possible timeout!



by Frederik Brückner
 FH Aachen, FB Gestaltung, Physical Computing Labor

Sheet: /
 File: DIY_Capacitive_Touch.sch

Title: Arduino Pro Micro simple capacitive touch shield

Size: A4

Date: 2018-01-12

KiCad E.D.A. KiCad 4.0.6

Rev: 1

Id: 1/1

1. Fassung

13. Januar 2018

Frederik Brückner

brueckner@fh-aachen.de