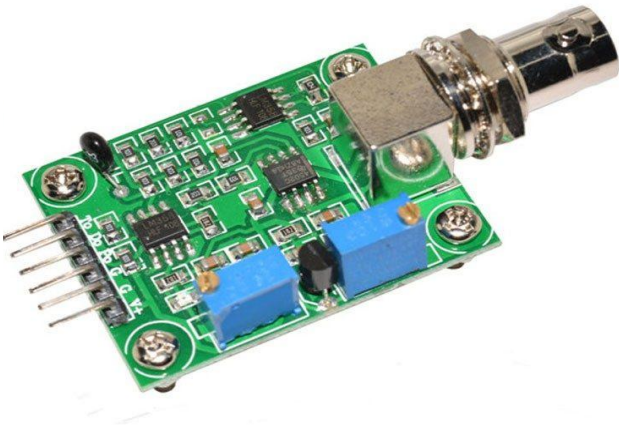
Modul PH-4502C pro měření pH

POPIS

Modul je určen pro zpracování signálu z pH senzoru. Výstup modulu lze dále zpracovávat mikrokontroléry. Zařízení lze hardwarově kalibrovat vestavěným potenciometrem. Pro optimální kalibraci je modul opatřen 10K NTC termistorem. Modul dále disponuje výstupem, který indikuje překročení nastaveného prahu kyselosti.



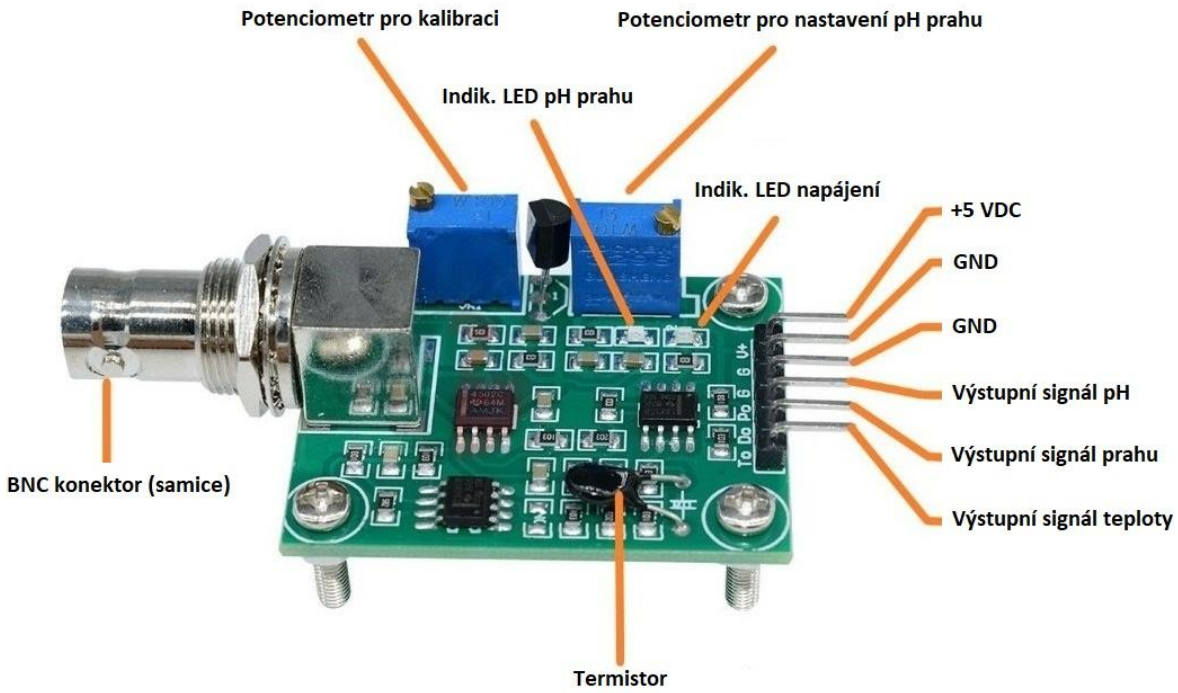
- BNC kon. (samice) pro připojení sondy
- hardwarová kalibrace
- indikace překročení pH prahu
- indikační LED diody

SPECIFIKACE

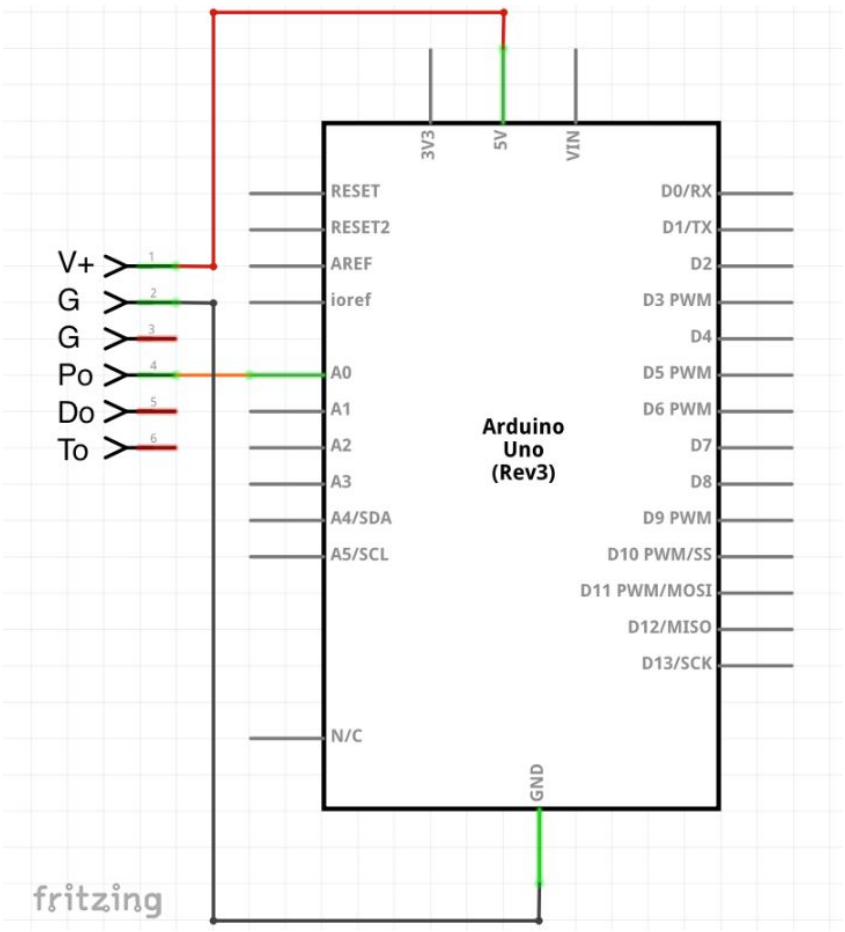
Napájecí napětí	5 VDC	**Termistor**	10K NTC
Proud	5 až 10 mA	**Detekovatelná teplota**	0 až 60 °C
Napěťová log. úroveň	5 VDC	**Operační teplota**	-10 až 50 °C
Napěťová log. – indikační výstup	1 (L) – 4 (H) VDC	**Rozměry**	66 x 32 x 16 mm
Rozsah měřitelnosti pH	0 až 14 pH	**Průměr mont. otvorů**	3 mm
Čas pro ustálení měření	< 60 s	**Rozteč mont. otvorů**	35 x 25 mm

ZAPOJENÍ

Rozložení modulu



Připojení k Arduinu



UKÁZKA PROGRAMU

Ukázkový program obsahuje možnost automatické kalibrace (bez nunosti hardwarové kalibrace). Kalibrace je nativně vypnutá a lze ji zapnout změnou proměnné *calibrationEnabled* do stavu TRUE. Výstup kalibrace upraví hodnotu proměnné *calibVal*. Daná proměnná je součástí výpočtu pH hodnoty. Pokud je kalibrace aktivována, provede se po každém startu Arduina. Je nicméně nutné, aby byla pH sonda ponořena do roztoku o hodnotě pH, která je shodná s hodnotou konstanty *PH_CALIB_REF* (uživatel ji může libovolně měnit).
Pokud se uživatel rozhodne softwarovou kalibraci nepoužít, musí vhodně zvolit hodnotu proměnné *calibVal*. Hodnota v programu je nastavena na základě kalibrace, která byla provedena během testování. Hodnota musí být přepočítána pro každý modul, aby bylo měření co nejpřesnější. Pokud uživatel nechce hodnotu měnit nebo ji ponechá nulovou, musí využít hardwarovou kalibraci.

```
/**
*********************************************************************************
* @author    Arduino development
* @version  1.0
* @date     13.02.2020
* @brief    Basic example for pH module PH-4502C
It is designed to read pH value.
Software calibration function is included.
*
*/

#include <Arduino.h>

#define PH_SENSOR_PIN          A0
#define AVG_ARRAY_SIZE         5
#define DELAY_BETWEEN_SAMPLES   20 //ms
#define VOLTAGE                5  //V
#define VOLTAGE_REF            1023
#define PH_REF                 -5.7
#define CALIB_SAMPLES          7
#define USELESS_SAMPLES        3

#define PH_CALIB_REF           6.86 //pH – used as reference for calibration

uint32_t avgValue = 0;
uint16_t analogValBuf[AVG_ARRAY_SIZE];
uint8_t actualIdx = 0;
bool firstSampling = true;
float sensorVoltage;
float phValue;

float calibVal = 33; //change this value to calibrate
bool calibrationEnabled = false; //auto calibration – change calibVal
bool firstCalibSample = true;
uint8_t numCalibSamples = 0;
uint32_t calibAvg = 0;
```

```
void calibration();
void fillBuffer();
void getFloatingAvg();
void sampling();

//function for software calibration – depends on variable calibrationEnabled and constant PH_CALIB_REF
void calibration(){
 for (numCalibSamples = 0; numCalibSamples < CALIB_SAMPLES; numCalibSamples++){
  if(firstCalibSample){
    fillBuffer();
    getFloatingAvg();
    firstCalibSample = !firstCalibSample;
  }
  else{
   if(actualIdx < AVG_ARRAY_SIZE){
    analogValBuf[actualIdx] = analogRead(PH_SENSOR_PIN);
    delay(DELAY_BETWEEN_SAMPLES);
    getFloatingAvg();
    actualIdx++;
   }
   else{
    actualIdx = 0;
    numCalibSamples--;
   }
  }
  if(numCalibSamples >= USELESS_SAMPLES && actualIdx != 0){
   calibAvg += avgValue;
  }
 }
 calibAvg = calibAvg / (CALIB_SAMPLES - USELESS_SAMPLES);
 sensorVoltage = (float)calibAvg * VOLTAGE / VOLTAGE_REF;
 calibVal = -PH_REF * sensorVoltage + PH_CALIB_REF;
 Serial.print("Calib value: ");
 Serial.println(calibVal);
}

//fill whole floating average buffer – it is used only at start (empty buffer)
void fillBuffer(){
 for (int i = 0; i < AVG_ARRAY_SIZE; i++)
 {
  analogValBuf[i] = analogRead(PH_SENSOR_PIN);
  delay(DELAY_BETWEEN_SAMPLES);
 }
}

//calculate actual value of floating average
void getFloatingAvg(){
  for (int i = 0; i < AVG_ARRAY_SIZE; i++){
   avgValue += analogValBuf[i];
  }
  avgValue = avgValue / AVG_ARRAY_SIZE;
}
```

```
//when function is called one sample is grabbed
void sampling(){
  if(firstSampling){
    fillBuffer();
    getFloatingAvg();
    firstSampling = !firstSampling;
  }
  else{
    if(actualIdx < AVG_ARRAY_SIZE){
      analogValBuf[actualIdx] = analogRead(PH_SENSOR_PIN);
      getFloatingAvg();
      actualIdx++;
    }
    else{
      actualIdx = 0;
    }
  }
}

void setup() {
  Serial.begin(9600);
  if(calibrationEnabled){
    calibration();
  }
}

void loop() {
  sampling();
  sensorVoltage = (float)avgValue * VOLTAGE / VOLTAGE_REF;
  phValue = PH_REF * sensorVoltage + calibVal;
  Serial.print("pH value: ");
  Serial.println(phValue);
  delay(1000);
}
```