

PODROBNÝ PRŮVODCE TEORIÍ

PODROBNĚ ROZEPSANÉ PŘÍKLADY S POPISEM FUNKCIONALIT OBVODŮ A PROGRAMOVÉHO KÓDU, KTERÝ JE ZAMĚŘEN NA POUŽÍVÁNÍ SENZORŮ PRO MĚŘENÍ TEPLITÝ A VLHKOSTI. PRO ZOBRAZENÍ TĚCHTO VELIČIN SE VYUŽIJÍ LCD DISPLAY. ŘEŠENÉ ÚKOLY ZOHLEDŇUJÍ NOVĚ PROBRANÉ TÉMA A STAVÝ NA PŘEDCHOZÍCH ZNALOSTECH.

OBSAH PRŮVODCE

- ① Získání dovedností při zapojování senzoru teploty.
- ② Naučení se pracovat se sériovým monitorem.
- ③ Naučit se zapojit LCD display a zobrazovat hodnoty.
- ④ Připojování externích knihoven pro snazší programování.
- ⑤ Naučit se zobrazovat hodnoty na LCD displeji.
- ⑥ Projekt skleníku.

SPRÁVA KNIHOVEN V ARDUINO

Knihovny jsou soubory kódu, které usnadňují připojení různých senzorů a dalších modulů k desce Arduino. V této části využíváme jako senzor teplotní čidlo a pro zobrazení hodnot z tohoto senzoru LCD display. Aby se nám lépe pracovalo s těmi to zařízeními, existují již hotové knihovny, které nám velmi zjednoduší jejich používání. Například pro LCD displej je již knihovna standardně nainstalovaná v prostředí Arduino IDE. Pro teplotní čidlo tomu tak není, proto si knihovnu musíme nainstalovat.



V Arduino jsou knihovny složky, které obsahují více souborů se zdrojovými kódy.

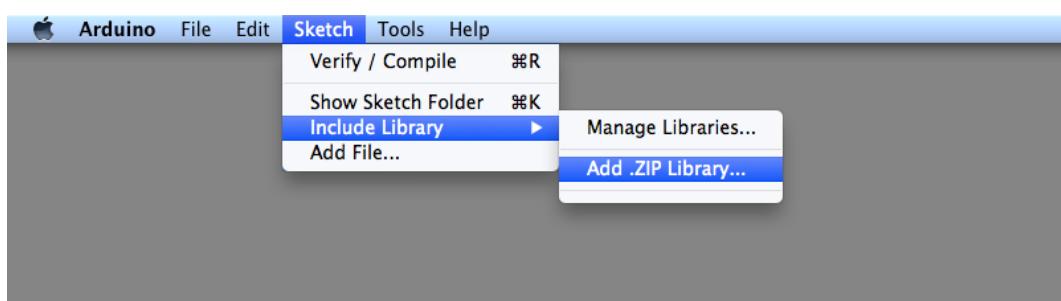
Tyto soubory jsou zpravidla napsány v jazyce C.

JAK NAINSTALOVAT KNIHOVNU

Instalaci knihoven, lze provést několika způsoby.

POMOCÍ IMPORTU KNIHOVNY ZIP

Chcete-li nainstalovat novou knihovnu do Arduino IDE, můžete využít nástroj, který je k tomu určený. Ten je dostupný přímo v IDE. Otevřete Arduino IDE a klikněte na položku menu **Projekt** (Sketch). Potom na **Přidat knihovnu** (Import Library) a zvolte **Přidat .ZIP Knihovnu...**. Otevře se okno, ve kterém vyberte ZIP archív s knihovnou a potvrďte. Pokud je knihovna v pořádku, dojde k jejímu nainstalování a objeví se v seznamu knihoven v aktuálně otevřené nabídce menu.



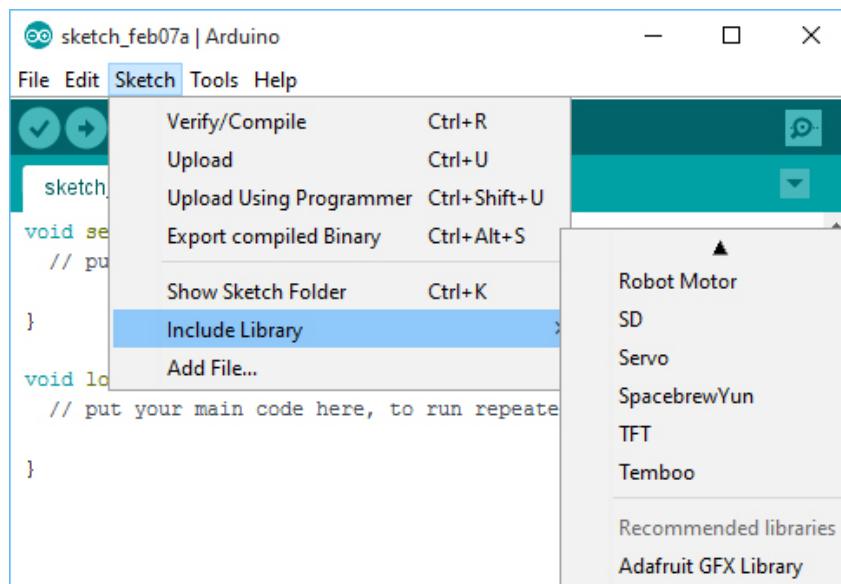
Obr. 1 - Instalace knihovny v Arduino IDE

RUČNÍ INSTALACE

Knihovnu lze přidat také ručně. Stáhněte si knihovnu jako ZIP soubor a rozbalte jej. Rozbalený adresář, který nese jméno knihovny obsahuje všechny potřebné soubory, včetně vzorových příkladů, které autor poskytnul. Tento celý adresář nakopírujte do dložky pro knihovny:

- ① V systému WINDOWS se adresář knihoven nachází: **Tento počítač > Dokumenty > Arduino > libraries.**
- ② V systému Apple OSX se adresář knihoven nachází: **MyFolder > Dokumenty > Arduino > libraries.**

Po přesunutí adresáře knihovny do správného umístění, restartujte Arduino IDE a ověřte, zda je knihovna v seznamu knihoven Obr. 2 - Seznam knihoven.



Obr. 2 - Seznam knihoven

SNÍMÁNÍ TEPLITÝ POMOCÍ TERMISTORU

Snímání teploty lze provádět několika různými senzory. Prvním takovým zástupcem je **termistor**.

TERMISTOR

Termistory jsou jednoduché, levné a relativně přesné komponenty, které lze využít pro různé meteostanice, domácí automatizační systémy a obvody řízení a ochrany zařízení. Jsou to analogové senzory, které ve srovnání s digitálními jsou velmi jednouché a nevyžadují žádné knihovny.

Termistory jsou variabilní rezistory, které mění svůj odpor v závislosti na teplotě. Jsou klasifikovány podle toho, jak jejich odpor reaguje na změnu teploty:

- ① **Termistory s negativním teplotním koeficientem** (NTC) – odpor se snižuje s nárůstem teploty.
- ② **Termistory s kladným teplotním koeficientem** (PTC) – odpor se zvyšuje s nárůstem teploty.



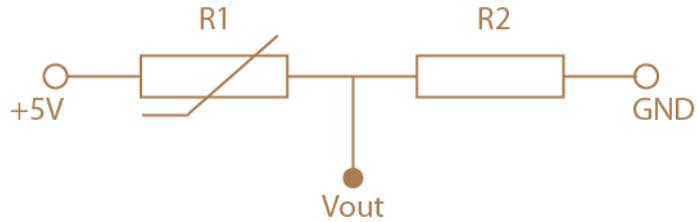
Běžnějšími termistory jsou NTC. Jsou vyrobeny z polovodičového materiálu, který je zahřát a následně stlačen, aby vytvořil teplotně citlivý vodivý materiál. Materiál obsahuje nosiče náboje, které dovolují protékání proudu. Vysoké teploty způsobují, že polovodivý materiál uvolňuje více nosičů náboje. V NTC termistorech vyrobených z oxidu železitého jsou nosičem náboje elektrony.

JAK TO FUNGUJE S ARDUINEM?

Analogové piny desky Arduino čtou hodnoty napětí od 0V do 5V, jenomže termistor nám poskytuje změnu odporu a ten přímo deskou Arduino číst nemůžeme. Jak na to? Standardní způsob, jak číst změnu odporu termistoru v závislosti na změně napětí je vytvořit obvod děliče napětí **Chyba! Nenašel se zdroj odkazu**.

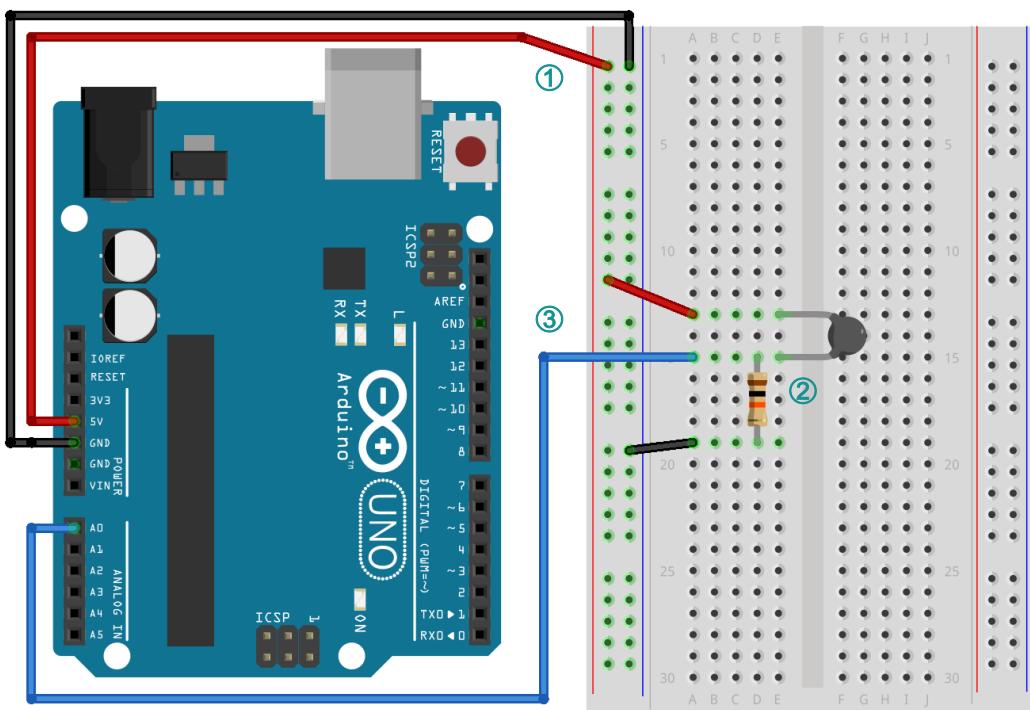
Je to velmi jednoduchý obvod. Při změně hodnoty odporu **R1** se změní hodnota **Vout**. Vzhledem k tomu, že odpor R1 je termistor a mění svůj odpor právě s teplotou, potom Vout může tvořit vstup do desky Arduino, který je pro ni čitelný.

Pro převod odporu termistoru na teplotu se využívá Steinhart-Hartova rovnice.



Obr. 3 - Dělič napětí.

ZAPOJENÍ OBVODU S TERMISTOREM



Obr. 4 - Zapojení termistoru

- ① Vodič zemnění **GND** z desky Arduino zapojte do kontaktní desky k **modré** čáře. Vodič napájení připojte z Arduina z pinu **5V** do kontaktního pole.
- ② Rezistor 10kΩ a termistor jsou zapojeny podle obvodu děliče napětí.
- ③ Signální vodič je zapojen do analogového pinu **A0**.

PROGRAMOVÝ KÓD

Programový kód obsahuje zejména matematické výpočty. Teplotu si zobrazíme v **Sériovém monitoru**.

```
1 int termistorPin = 0;          ①
2 int Vout;                     ②
3 float R2 = 10000;            ③
4 float logR1, logR2, R1, T, Tc, Tf; ④
5 float c1 = 1.009249522e-03, c2 = 2.378405444e-04, c3 = ⑤
6 2.019202697e-07;
7
8 void setup() {
9     Serial.begin(9600);          ⑥
10 }
11
12 void loop() {
13     Vout = analogRead(termistorPin);    ⑦
14     R1 = R2 * (1023.0 / (float)Vout - 1.0); ⑧
15     logR1 = log(R1);
16     T = (1.0 / (c1 + c2*logR1 + c3*logR1*logR1*logR1)); ⑨
17     Tc = T - 273.15;                ⑩
18     Tf = (Tc * 9.0) / 5.0 + 32.0;   ⑪
19
20     Serial.print("Teplota: ");
21     Serial.print(Tf);              ⑫
22     Serial.print(" F; ");
23     Serial.print(Tc);
24     Serial.println(" C");
25
26     delay(500);
27 }
```

- ① Deklarace proměnné **termistorPin** pro číslo analogového pinu desky Arduino, na který je připojen datový vodič z napěťového děliče.
- ② Deklarace proměnné **Vout**, která bude nabývat aktuální hodnoty v závislosti na teplotě.
- ③ Hodnota neměnného rezistoru **R2** u napěťového děliče, tj. $10\text{k}\Omega$.
- ④ Deklarace proměnných **logR2**, **R1**, **T**, **Tc**, **Tf**, které nabývají hodnot z výpočtů.
- ⑤ Deklarace konstant, které jsou určeny pro výpočet Steinhart-Hartovy rovnice. Tyto hodnoty jsou parametry, které se dají zjistit z katalogu součástky.

- ⑥ Nastavení rychlosti pro sériový přenos. Dílčí výsledky měření zobrazíme v sériovém monitoru.
- ⑦ Zjištění změny napětí, přečtené z analogového pinu desky Arduinoa uložení do proměnné **Vout**.
- ⑧ Výpočet aktuálního odporu **R1** v závislosti na změně **Vout**.
- ⑨ Steinhart-Hartova rovnice, která slouží k výpočtu poměru odpor-teplota. Výsledek v proměnné **T** je ve stupních Kelvinia.
- ⑩ Převod ze stupňů Kelvinia na Stupně Celsia - **Tc**.
- ⑪ Převod ze stupňů Celsia na stupně Fahreinheita - **Tf**.
- ⑫ Vypsání hodnot do sériového monitoru.



Nezapomeňte program zkompilovat a nahrát do desky Arduino. Otevřete sériový monitor. Pokud je vše v pořádku, měla by být vidět aktuální hodnota teploty naměřené termistorem.

MĚŘENÍ TEPLITÝ A VLHKOSTI

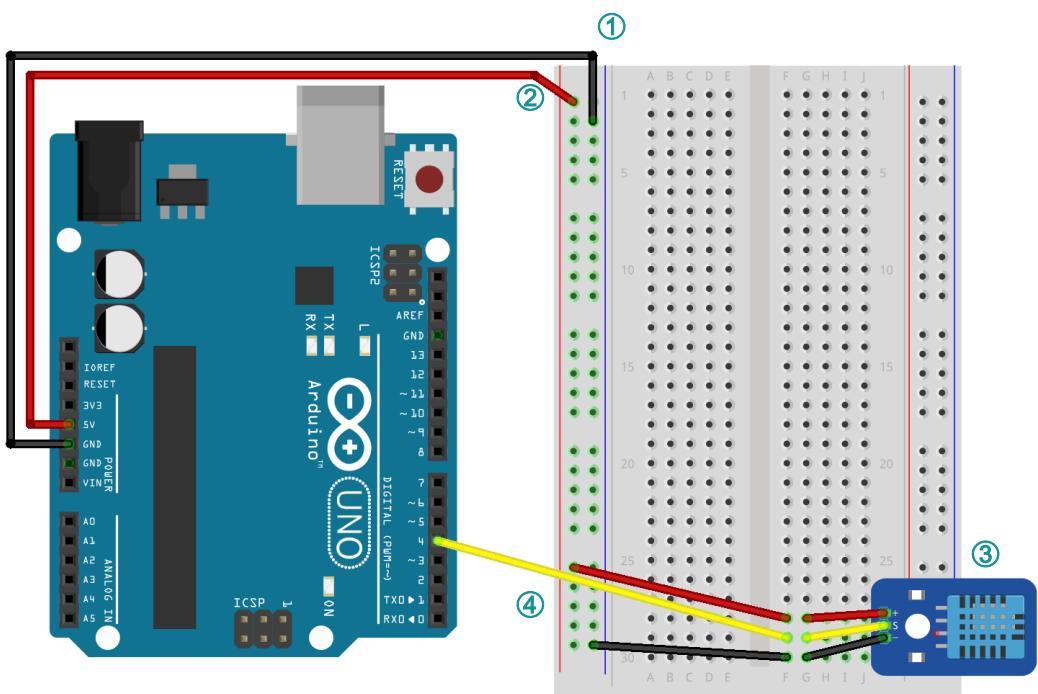
Zapojení obvodu se může jevit jako složitější, ale není tomu tak. Použité teplotní čidlo a LCD displej poskytuje rychlé řešení, které lze postavit za několik minut.

TEPLOTNÍ ČIDLO

V projektu je použito teplotní čidlo, které je součástí sady Arduino. Jeho technické označení je DHT11. Čidlo je laboratorně kalibrované, přesné, stabilní a jeho signál je **digitální**. Čidlo má tři vývody: VCC (+) pro napájení, GND (-) je zemnění a S je signální pro připojení do pinu desky Arduino. Čidlo kromě teploty, zjišťuje také vlhkost.

ZAPOJENÍ OBVODU

Nejdříve vytvoříme jednoduché zapojení pro teplotní čidlo. Naměřenou teplotu si necháme zobrazit pomocí sériového monitoru.



Obr. 5 - Zapojení teplotního čidla

- ① Vodič zemnění z desky Arduino zapojte do kontaktní desky k **modré** čáře. Tím se zpřehlední celé zapojení.
- ② Z desky Arduino, z pinu pro 5V přiveďte červený vodič do kontaktního pole k **červené** čáře.

- ③ Do kontaktní desky zapojte teplotní čidlo. Jeho konektory stačí zasunout do kontaktního pole. Teplotní čidlo má označené vývody **+**, **S**, **-**.
- ④ Jednotlivé vodiče propojte s teplotním čidlem. Vodič napájení (**červený**) přiveďte na vývod čidla „**+**“. Vodič zemnění (**černý**) přiveďte na vývod čidla „**-**“ a vodič signální (**žlutý**) přiveďte z pinu **4** desky Arduino do vývodu čidla **S**.

PROGRAMOVÝ KÓD

Programový kód je díky knihovně pro čidlo velmi jednoduchý. Teplotu si zobrazíme opět v **Sériovém monitoru**.

```

1 #include <dht11.h>                                ①
2
3 dht11 cidlo;                                       ②
4
5 int dhtpin=4;                                      ③
6
7 void setup(){                                       ④
8     Serial.begin(9600);
9 }
10
11 void loop() {
12 {
13     cidlo.read(dhtpin);                            ⑤
14     Serial.print("Teplota = ");                   ⑥
15     Serial.println(cidlo.temperature);            ⑦
16     Serial.print("Vlhkost = ");                  ⑧
17     Serial.println(cidlo.humidity);              ⑨
18     delay(1000);                                ⑩
19 }
```

- ① Příkazem **#include** se připojují knihovny k projektu. Ve špičaté závorce je název hlavičkového souboru knihovny pro teplotní čidlo. Tuto knihovnu si lze stáhnout z [adresy](https://github.com/Nowis75/PRIM/raw/master/Experiments/Arduino/06_THERMO_DISPLAY/lib/Dht11.zip)
- ② Deklarace čidla **dht11** s názvem **cidlo**.
- ③ Deklarace proměnné **dhtpin**, což je číslo pinu na který je připojen signální vývod **S** čidla.

- ④ Ve funkci **setup()** se nastavuje přenosová rychlosť v bitech za sekundu (baud) pro sériový přenos. Nastavení probíhá pomocí **Serial.begin()**. Pro komunikaci s počítačem lze požít některou z rychlostí: 200, 600, 1200, 2400, 4800, 9600, 14400, 19200, 28800, 38400, 57600 nebo 115200. Díky tomu lze na sériovém monitoru vypisovat různá hlášení.
- ⑤ Čtení dat z čidla s názvem **cidlo**. Funkce **read()** má jediný parametr a tím je číslo pinu, na který je připojen signální vývod čidla. V tomto příkladu je číslo pinu uloženo v proměnné **dhtpin**.
- ⑥ Do sériového portu se pomocí funkce **print()** vytiskne ASCII, pro lidi čitelný text. V našem případě je to text „**Teplo**ta=“.
- ⑦ Pomocí funkce **cidlo.temperature**, zjistíme aktuální teplotu na čidle a zároveň ji vytiskneme do sériového monitoru pomocí funkce **println()**.



Všimněte si, že se k výpisu používají dvě funkce **print()** a **println()**. Jaký je mezi nimi rozdíl? Pokud použijeme funkci **print()**, tak následující řetězec se vytiskne hned za řetězec vypsaný touto funkcí. Když se použije funkce **println()**, tak následující řetězec bude vytisknutý na novu řádku.

- ⑧ Vypsání řetězce „**Vlhkost**=“.
- ⑨ Zjištěn vlhkosti pomocí funkce **cidlo.humidity** a vypsání zjištěné hodnoty.
- ⑩ Zastavení běhu programu na jednu sekundu, aby bylo ve smyčce dostatek času na další zjištění aktuálních hodnot.

OTEVŘENÍ SÉRIOVÉHO MONITORU

Nahrajte program do desky Arduino. V Arduino IDE klikněte na ikonu  pro otevření sériového monitoru. Otevře se nové okno, ve kterém zobrazí aktuální hodnoty teploty a vlhkosti Obr. 6 - Okno sériového monitoru.



The screenshot shows the Arduino Serial Monitor window titled '/dev/tty.usbmodem1421'. The window displays a series of text lines: 'Vlhkost = 30', 'Teplota = 24', 'Vlhkost = 30'. Below the text area, there are three buttons: 'Autoscroll' (checked), 'Newline', and '9600 baud'.

Obr. 6 - Okno sériového monitoru



NEZOBRAZUJÍ SE HODNOTY TEPLITOY A VLHKOSTI

Zapojení čidla – zkontrolujte zapojení teplotního čidla, aby signální vývod byl opravdu připojen k digitálnímu pinu na desce Arduino.

Pozor na připojení vodičů pro napájení (5V) a zemnění (GND)

Vodiče – zkontrolujte, zda jsou vodiče opravdu dobře zasunuty do kontaktního pole desky Arduino.

Přenosová rychlosť – zkontrolujte přenosovou rychlosť sériového portu v okně sériového monitoru, že je stejná jako v kódu ve funkci

`Serial.begin(9600);`

NEJDE NAHRÁT KÓD DO DESKY

USB kabel – ujistěte se, že máte desku Arduino připojenou k počítači.

Správný port – ujistěte se, že máte vybraný správný port pro připojení k desce Arduino pomocí USB kabelu.

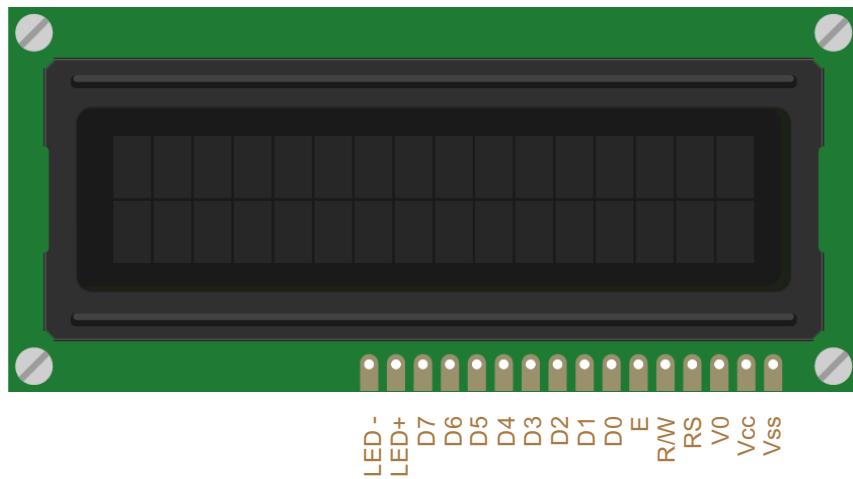


(Př. 1) Změňte programový kód předchozího příkladu tak, aby se kromě teploty ve stupních Celsia zobrazovala teplota i v Kelvinech a Fahrenheitech. Pro výpočet využijte vlastních funkcí.

ZOBRAZENÍ TEPLITY NA LCD displeji

LCD displej

Použitý displej umožňuje zobrazovat alfanumerické znaky. Má 16 sloupců a dva řádky, tzn. dokáže zobrazit 32 znaků. Obsahuje velké množství pinů pro komunikaci s deskou Arduino a k napájení. Komunikační piny, ale není třeba zapojovat všechny.



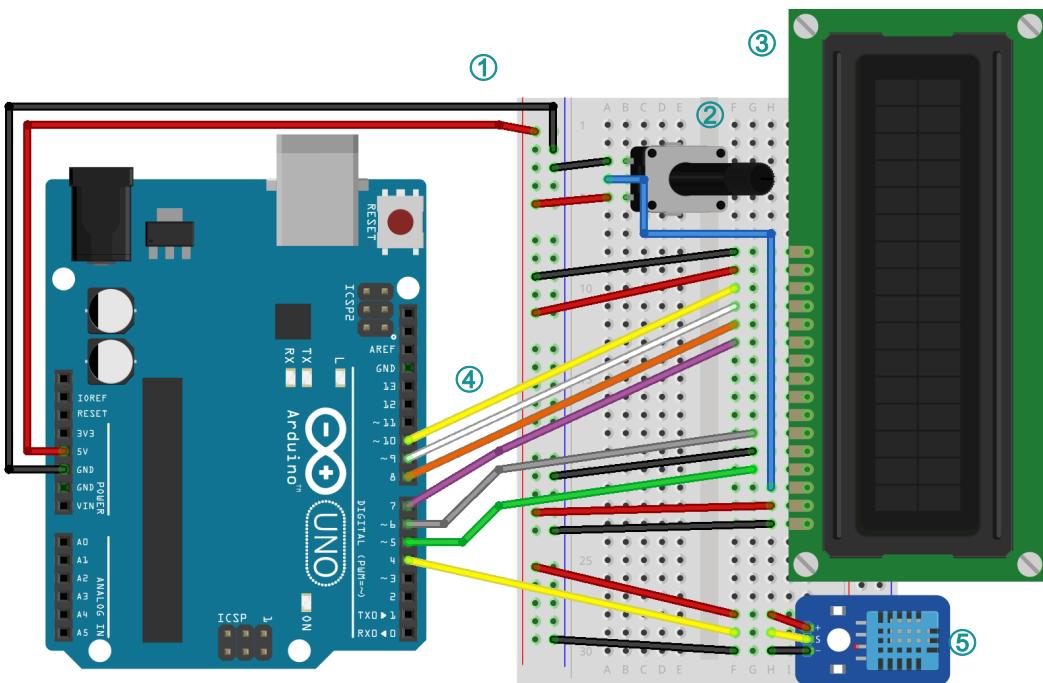
Vss	Připojení k zemi.	E	Arduino.
Vcc	Připojení k 5V.	D0 – D3	Datové piny, které se nepoužijí.
V0	Kontrast.	D4 – D7	Datové piny připojené k Arduino.
RS	Připojení k Arduinu.	A	Anoda podsvícení, 5V.
RW	Zem.	K	Katoda podsvícení, zem.

POTENCIOMETR

Jedná se o proměnný odpor a v projektu je použit pouze pro nastavení kontrastu LCD displeje.

ZAPOJENÍ OBVODU

Využijeme zapojení obvodu teplotního čidla a přidáme LCD displej pro zobrazení hodnot teploty a vlhkosti.



Obr. 7 - Zapojení teplotního čidla

- ①** Vodič zemnění **GND** z desky Arduino zapojte do kontaktní desky k **modré** čáře. Vodič napájení připojte z Arduina z pinu **5V** do kontaktního pole. Tím se zpřehlední celé zapojení.
- ②** Do kontaktního pole připojte potenciometr. Stačí zasunou vývody potenciometru přímo do pole. Jeden krajní vývod zapojte na zem a druhý krajní vývod na napájení 5V. Prostřední vývod připojte na vývod displeje **V₀**. Bude sloužit k regulaci kontrastu LCD displeje.
- ③** LCD displej zasuňte do kontaktního pole.
- ④** Vývody z LCD displeje zapojeny následně: **V_{ss}** na zem, **V_{cc}** na 5V, **V₀** na prostřední vývod potenciometru, **RS** je připojen do pinu 5 na desce Arduino, **RW** je připojen k zemi, **E** je připojen na pin 6, **D4** je připojen na pin 7, **D5** je připojen na pin 8, **D6** na pin 9, **D7** na pin 10. Vývod **A** je připojen k 5V a **K** je připojen na zem.
- ⑤** Teplotní čidlo je připojeno stejně jako v přechozím příkladu, tj. vývod (+) je na napájení 5V, vývod (-) na zem a vývod (**S**) je připojen na pin desky Arduino **4**.

PROGRAMOVÝ KÓD

Programový kód využívá knihovnu pro ovládání LCD displeje, která je v Arduino IDE již standardně nainstalovaná.

```
1 #include <dht11.h>
2 #include <LiquidCrystal.h>
3
4 int rsPin = 5;
5 int ePin = 6;
6 int d4Pin = 7;
7 int d5Pin = 8;
8 int d6Pin = 9;
9 int d7Pin = 10;
10 LiquidCrystal LCD(rsPin,ePin,d4Pin,d5Pin,d6Pin,d7Pin);
11
12 dht11 cidlo;
13 int dhtpin = 4;
14
15 void setup()
16 {
17     LCD.begin(16,2);
18     LCD.clear();
19     LCD.setCursor(0,0);
20     LCD.print("Teplota: ");
21
22     LCD.setCursor(0,1);
23     LCD.print("Vlhkost: ");
24 }
25
26 void loop()
27 {
28     cidlo.read(dhtpin);
29
30     LCD.setCursor(9,0);
31     LCD.print(cidlo.temperature);
32     LCD.setCursor(13,0);
33     LCD.print((char)223);
34     LCD.print("C");
35
36     LCD.setCursor(9,1);
37     LCD.print(cidlo.humidity);
38     delay(500);
39 }
```

- ① Příkazem `#include` se připojí knihovna pro teplotní čidlo.
- ② Stejným příkazem se připojí knihovna pro LCD displej.
- ③ Deklarace čísel pinů, které jsou využity pro připojení LCD displeje k desce Arduino. Displej můžeme připojit i na jiné piny, potom stačí změnit hodnoty těchto proměnných.
- ④ Funkce `LiquidCrystal` vytvoří objekt LCD displej o názvu LCD. Jeho parametry jsou deklarovány v proměnných viz přechozí krok. Proměnné bychom ani nemuseli používat, ale a musí být dodrženo uvedené pořadí.
- ⑤ Deklarace čidla `dht11` s názvem `cidlo`.
- ⑥ Deklarace proměnné `dhtpin` jako číslo pinu desky Arduino pro datový vývod tepelného. To je stejné jako v předchozím příkladu.
- ⑦ Spustí displej. Tento displej má 16 znaků v každém ze dvou řádků.
- ⑧ Funkce `clear()` preventivně vymaže display od starých znaků.
- ⑨ Funkce `setCursor()` nastaví kurzor na první znak a první řádek displeje. Číslování pozic LCD displeje začíná vždy od nuly.
- ⑩ Funkce `print()` napíše text „**Teplota:**“ na displej se začátkem definovaným v předchozím kroku.
- ⑪ Opět nastavíme pozici kurzoru, ale tentokrát na první pozici ve druhém řádku.
- ⑫ Na druhém řádku vypíšeme text „**Vlhkost:**“.
- ⑬ Přečteme hodnotu teploty.
- ⑭ Nastavíme kurzor na desátou pozici v prvním řádku, což je za textem „**Teplota:**“.
- ⑮ A vypíšeme naměřenou teplotu z čidla.
- ⑯ Nastavíme kurzor v prvním řádku na třináctou pozici. To je pozice pro vypsání jednotky teploty.
- ⑰ Vypíšeme znak stupňů celsia (`°`). To nám zajistí `(char)223`, což odpovídá ASCII hodnotě znaku stupňů. Pokud bychom napsali přímo znak stupňů, nemuselo by to fungovat.
- ⑱ Vypíšeme písmeno C pro. Výsledkem pak bude `°C`.
- ⑲ Nastavíme kurz v druhém řádku na devátou pozici za text „**Vlhkost:**“.
- ⑳ Vypíšeme aktuální hodnotu vlhkosti.



Nezapomeňte program zkompilovat a nahrát do desky Arduino. Pokud je vše v pořádku, na displeji se objeví teplota a vlhkost.



NA LCD displeji není nic vidět

Nastavení kontrastu – ujistěte se, že kontrast displeje je v pořádku nastaven. To provedete otočením potenciometru na jednu nebo druhou stranu.

Zapojení LCD displeje – zkontrolujte, zda jsou všechny vodiče opravdu správně zapojeny. Nezapomeňte, že piny v LCD displeji musí být v odpovídajícím pořadí.

NEZOBRAZUJÍ SE HODNOTY NA LCD displeji

Zapojení čidla – zkontrolujte zapojení teplotního čidla, aby signální vývod byl opravdu připojen k digitálnímu pinu na desce Arduino.

Pozor na připojení vodičů pro napájení (5V) a zemnění (GND).

Vodiče – zkontrolujte, zda jsou vodiče opravdu dobře zasunuty do kontaktního pole desky Arduino.

NEJDE NAHRÁT KÓD DO DESKY



(Př. 2) Změňte programový kód předchozího příkladu tak, aby se kromě teploty ve stupních Celsia, na LCD displeji, střídavě zobrazovala teplota v Kelvinech a Fahrenheitech.

ŘEŠENÍ PŘÍKLADŮ

PŘ. 1.

Řešení příkladu spočívá v přidání dvou vlastních funkcí, i když by se dal řešit i bez těchto funkcí. Pomocí nich se, ale program zpřehlední.

```
1 #include <dht11.h> // pripojení knihovny
2
3 dht11 cidlo;           // deklarace cidla
4
5 int dhtpin=4;          // deklarace signálního pinu
6
7 // funkce pro vypocet ve stupnich Fahrenheita
8 double Fahrenheit(double celsius){
9     return 1.8 * celsius + 32;
10 }
11
12 // funkce pro vypocet ve stupnich Kelvina
13 double Kelvin(double celsius){
14     return celsius - 273.15;
15 }
16
17 void setup(){
18     Serial.begin(9600);
19 }
20
21 void loop()
22 {
23     cidlo.read(dhtpin);
24
25     double celsia=cidlo.temperature; // zjistění stupnů Celsia
26     double kelviny=Kelvin(celsia); // zjistění stupnů Kelvina
27     double fahrenheity=Fahrenheit(celsia); // zjistění stupnů
28                                         // Fahrenheita
29
30     // Vypisy v seriovém monitoru
31     Serial.print("Teplota v Celsiech= ");
32     Serial.println(celsia);
33     Serial.print("Teplota v Kelvinech= ");
34     Serial.println(kelviny);
35     Serial.print("Teplota v Fahrenheitech= ");
36     Serial.println(fahrenheity);
37
38     Serial.print("Vlhkost = ");
39     Serial.println(cidlo.humidity);
40     delay(1000);
41 }
```

PŘ. 2.

K vyřešení příkladu lze využít část kódu z příkladu (Př. 1. Jsou to zejména funkce pro výpočet teploty ve stupních Kevlina a Fahrenheita.

```
1 #include <dht11.h> // pripojeni knihovny
2 #include <LiquidCrystal.h>
3
4 int rsPin = 5;
5 int ePin = 6;
6 int d4Pin = 7;
7 int d5Pin = 8;
8 int d6Pin = 9;
9 int d7Pin = 10;
10 LiquidCrystal LCD(rsPin,ePin,d4Pin,d5Pin,d6Pin,d7Pin);
11
12 dht11 cidlo;
13 int dhtpin = 4;
14
15 // funkce pro vypocet ve stupnich Fahrenheita
16 double Fahrenheit(double celsius){
17     return 1.8 * celsius + 32;
18 }
19
20 // funkce pro vypocet ve stupnich Kelvina
21 double Kelvin(double celsius){
22     return celsius + 273.15;
23 }
24
25 void setup(){
26     LCD.begin(16,2);
27     LCD.clear();
28     LCD.setCursor(0,0);
29     LCD.print("Teplota: ");
30
31     LCD.setCursor(0,1);
32     LCD.print("Vlhkost: ");
33 }
34
35 void loop(){
36     cidlo.read(dhtpin);
37
38     double celsia=cidlo.temperature; // zjisteni stupnu Celsia
39     double kelviny=Kelvin(celsia);   // zjisteni stupnu Kelvina
40
41 }
```

```
42     double fahrenheity=Fahrenheit(celsia); // zjistění stupnů  
43                                     // Fahrenheita  
44  
45     // v první rade se vypíše vlhkost  
46     LCD.setCursor(9,1);  
47     LCD.print(cidlo.humidity);  
48  
49     // vypis teploty ve stupních celsia  
50     LCD.setCursor(9,0);  
51     LCD.print(celsia);  
52     LCD.setCursor(15,0); // musí se nastavit kurzor kde se  
53     LCD.print(" ");    // zobrazuje staré hodnoty v K a F  
54     LCD.setCursor(13,0);  
55     LCD.print((char)223);  
56     LCD.print("C");  
57  
58     delay(2000);  
59  
60     // vypis teploty ve stupních kelvina  
61     LCD.setCursor(9,0);  
62     LCD.print(kelviny);  
63     LCD.setCursor(15,0);  
64     LCD.print("K");  
65  
66     delay(2000);  
67  
68     // vypis teploty ve stupních fahrenheinta  
69     LCD.setCursor(9,0);  
70     LCD.print(fahrenheity);  
71     LCD.setCursor(15,0);  
72     LCD.print("F");  
73  
74     delay(2000);  
75 }  
76
```



Proč je v programu, na řádku **53** a **54** nastavení kurzoru a tisk mezer? Musíme si uvědomit, že program probíhá v neustálé smyčce, proto pokud bychom preventivně nevymazali staré hodnoty ze stupňů Kelvina a Fahrenheita, zůstávaly by tam staré znaky, protože ve stupních Celsia je řetězec kratší. Takto se konec displeje nahradí prázdným místem a teplota ve stupních Celsia se zobrazí v pořádku.

AUTOMATICKÝ SKLENÍK

TATO ČÁST UKAZUJE, JAK SPOJIT PŘEDCHOZÍ NABYTÉ ZNALOSTI V JEDINÝ PROJEKT. TÍMTO PROJEKTEM JE VYTVOŘENÍ AUTOMATICKY OVLÁDANÉHO SKLENÍKU S VYUŽÍTM TEPLITNÍHO ČIDLA, SERVOMOTORU PRO OVLÁDÁNÍ VENTILACE A STEJNOSMĚRNÉHO MOTORU PRO VÉTRÁK.

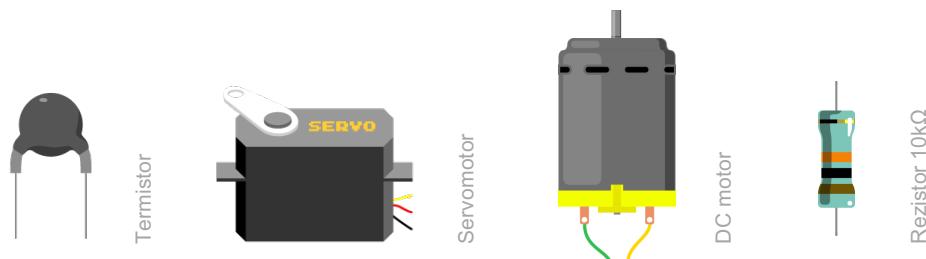
CÍLE

- ① Upevnění nabytých znalostí pro práci s motory.
- ② Upevnění znalostí pro práci se senzory.
- ③ Zvládnutí práce s více komponentami najednou.
- ④ Spojení programátorských dovedností s konstrukčními dovednostmi.

Čas: **90 min**

Úroveň:

Vychází z: **5, 6**

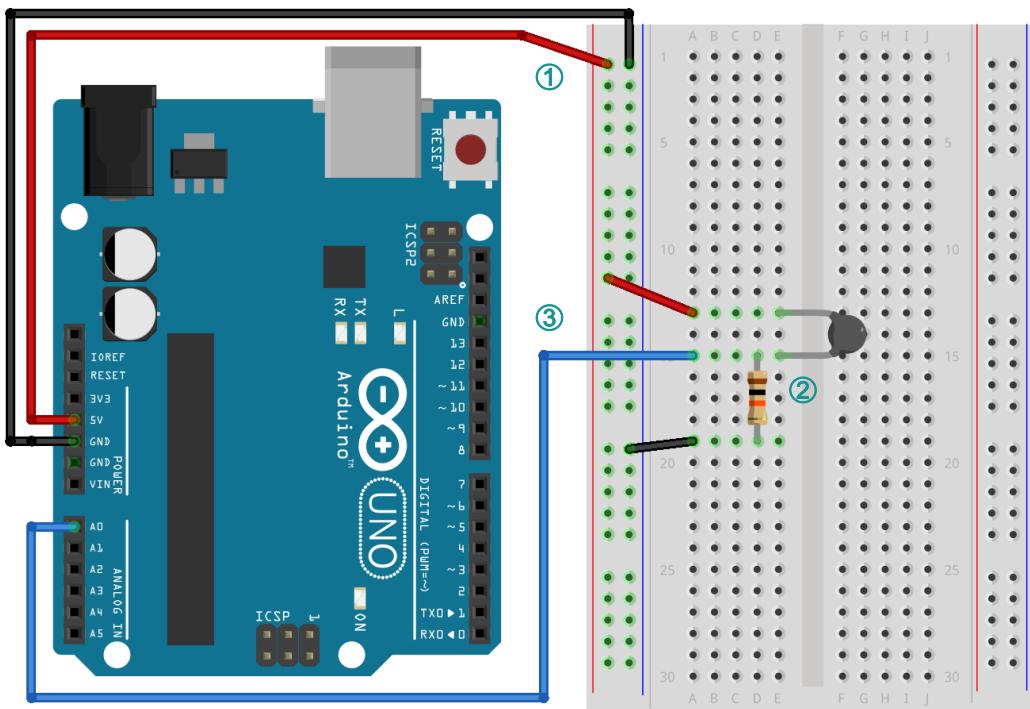


POUŽITÉ SOUČÁSTKY

Projekt automatického skleníku se skládá z několika oddělených částí:

- ① Snímání teploty pomocí termistoru.
- ② Ovládání střešního okna servomotorem.
- ③ Větrání prostřednictvím stejnosměrného motoru.

ZAPOJENÍ OBVODU S TERMISTOREM



Obr. 8 - Zapojení termistoru

- ① Vodič zemnění **GND** z desky Arduino zapojte do kontaktní desky k **modré** čáře. Vodič napájení připojte z Arduina z pinu **5V** do kontaktního pole.
- ② Rezistor $10\text{k}\Omega$ a termistor jsou zapojeny podle obvodu děliče napětí.
- ③ Signální vodič je zapojen do analogového pinu **A0**.

PROGRAMOVÝ KÓD

Programový kód je díky knihovně pro čidlo velmi jednoduchý. Teplotu si zobrazíme v [Sériovém monitoru](#).

```
1 int termistorPin = 0;          ①
2 int Vout;                     ②
3 float R2 = 10000;            ③
4 float logR1, logR2, R1, T, Tc, Tf; ④
5 float c1 = 1.009249522e-03, c2 = 2.378405444e-04, c3 = ⑤
6 2.019202697e-07;
7
8 void setup() {
9     Serial.begin(9600);          ⑥
10 }
11
12 void loop() {
13     Vout = analogRead(termistorPin);    ⑦
14     R1 = R2 * (1023.0 / (float)Vout - 1.0); ⑧
15     logR1 = log(R1);
16     T = (1.0 / (c1 + c2*logR1 + c3*logR1*logR1*logR1)); ⑨
17     Tc = T - 273.15;                ⑩
18     Tf = (Tc * 9.0) / 5.0 + 32.0;   ⑪
19
20     Serial.print("Teplota: ");
21     Serial.print(Tf);              ⑫
22     Serial.print(" F; ");
23     Serial.print(Tc);
24     Serial.println(" C");
25
26     delay(500);
27 }
```

- ① Deklarace proměnné **termistorPin** pro číslo analogového pinu desky Arduino, na který je připojen datový vodič z napěťového děliče.
- ② Deklarace proměnné **Vout**, která bude nabývat aktuální hodnoty v závislosti na teplotě.
- ③ Hodnota neměnného rezistoru **R2** u napěťového děliče, tj. $10\text{k}\Omega$.
- ④ Deklarace proměnných **logR2**, **R1**, **T**, **Tc**, **Tf**, které nabývají hodnot z výpočtů.
- ⑤ Deklarace konstant, které jsou určeny pro výpočet Steinhart-Hartovy rovnice. Tyto hodnoty jsou parametry, které se dají zjistit z katalogu součástky.

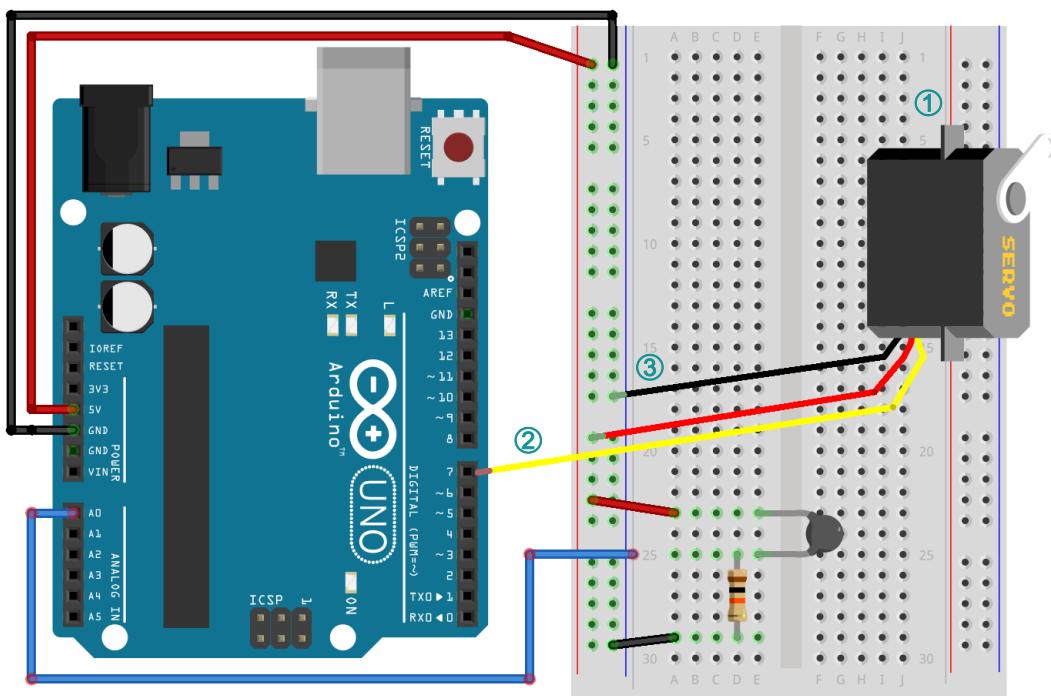
- ⑥ Nastavení rychlosti pro sériový přenos. Dílčí výsledky měření zobrazíme v sériovém monitoru.
- ⑦ Zjištění změny napětí, přečtené z analogového pinu desky Arduino a uložení do proměnné **Vout**.
- ⑧ Výpočet aktuálního odporu **R1** v závislosti na změně **Vout**.
- ⑨ Steinhart-Hartova rovnice, která slouží k výpočtu poměru odpor-teplota. Výsledek v proměnné **T** je ve stupních Kelvinia.
- ⑩ Převod ze stupňů Kelvinia na stupně Celsia - **Tc**.
- ⑪ Převod ze stupňů Celsia na stupně Fahreinheita - **Tf**.
- ⑫ Vypsání hodnot do sériového monitoru.



Nezapomeňte program zkompilovat a nahrát do desky Arduino. Otevřete sériový monitor. Pokud je vše v pořádku, měla by být vidět aktuální hodnota teploty naměřené termistorem.

ZAPOJENÍ SERVOMOTORU

Zapojení servomotoru bude pokračovat z přechozího zapojení termistoru.



Obr. 9 - Zapojení servomotoru

- ① Servomotor, jak již známe, má tři vodiče. Napájení, zem a datový vodič. Konektor těchto vodičů typu **zásvuka**, musíme nastavit vodiči, pokud možná stejné barvy, typu **zástrčka**, aby šli zapojit do kontaktního pole a desky Arduino.
- ② Datový vodič zapojíme do digitálního pinu 7 desky Arduino.
- ③ Vodiče napájení a zemnění připojíme do kontaktního pole.

PROGRAMOVÝ KÓD

Programový kód pro ovládání servomotoru je opět velmi jednoduchý. Servomotor bude pouze otevírat a zavírat střešní okno skleníku.

```
1 #include <Servo.h>                                ①
2
3 int termistorPin = 0;
4 int Vout;
5 float R2 = 10000;
6 float logR2, R1, T, Tc, Tf;
7 float c1 = 1.009249522e-03, c2 = 2.378405444e-04, c3 =
8 2.019202697e-07;
9
10 int servoPin = 7;                                 ②
11 int setTemp = 25;                                ③
12 int setTempRet=22;                               ④
13
14 Servo myservo;                                ⑤
15
16 void setup() {                                   
17     Serial.begin(9600);
18
19     myservo.attach( servoPin );                  ⑥
20     myservo.write( 0 );
21 }
22
23 void loop() {                                   
24     Vout = analogRead(termistorPin);
25     R1 = R2 * (1023.0 / (float)Vout - 1.0);
26     logR1 = log(R1);
27     T = (1.0 / (c1 + c2*logR1 + c3*logR1*logR1*logR1));
28     Tc = T - 273.15;
29     Tf = (Tc * 9.0)/ 5.0 + 32.0;
30
31     Serial.print("Teplota: ");
32     Serial.print(Tf);
33     Serial.print(" F; ");
34     Serial.print(Tc);
35     Serial.println(" C");
36     delay(500);
37
38     if(T > setTemp ){                         ⑧
39         myservo.write(180);
40     }else if(T < setTemp){                     ⑨
41         myservo.write(0);
42     }
43     delay(1000);
44 }
```

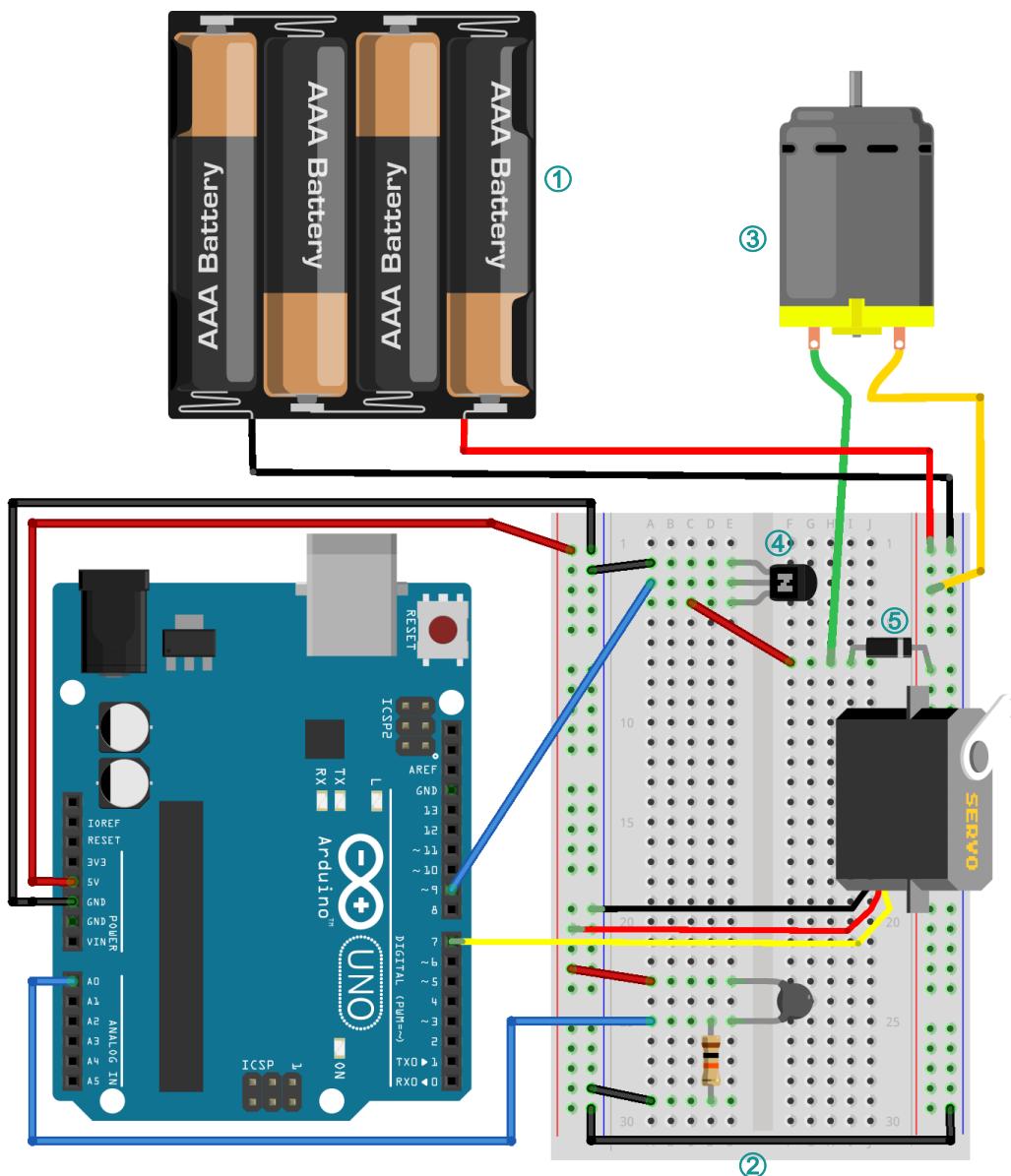
- ① Připojení knihovny pro ovládání servomotoru.
- ② Deklarace proměnné pro číslo pinu, na který je připojen datový vodič servomotoru.
- ③ Deklarace proměnné pro teplotu ve °C, při které dojde k **otevření** střešního okna.
- ④ Deklarace proměnné pro teplotu ve °C, při které dojde k **zavření** střešního okna.
- ⑤ Vytvoření instance **myservo** třídy pro servomotor.
- ⑥ Definice pinu, na který je motor připojen. Zde je na pinu **7**.
- ⑦ Nastavení výchozí pozice servomotoru. Při hodnotě **0** je střešní okno zavřené.
- ⑧ Testování pomocí podmínkového příkazu **if**, zda teplota nepřesáhla definovanou hodnotu v proměnné **setTemp**. Pokud ano, tak servomotor se nastaví do maximální pozice a střešní okno bude otevřené.
- ⑨ Druhá část podmínkového příkazu **if**, ve které se testuje, zda teplota neklesla pod stanovenou hodnotu uloženou v proměnné **setTempRet**. Pokud ano tak, se servomotor vrátí do pozice **0** a okno bude zavřené.



Opět nezapomeňte program zkompilovat a nahrát do desky Arduino. Zkuste rukou zahřívat termorezistor, nebo jej ochlazujte foukáním. V sériovém monitoru sledujte změnu teploty. Upravte případně hodnoty v proměnných **setTemp** a **setTempRet**, tak, aby došlo k aktualizaci pozice servomotoru.

ZAPOJENÍ STEJNOSMĚRNÉHO MOTORU JAKO VĚTRÁKU

Zapojení DC motoru opět zakomponováno do existujícího zapojení s termistorem a servomotorem. Otáčky motoru, který je v projektu využitý jako větrák budou závislé na aktuální teplotě.



Obr. 10 - Zapojení stejnosměrného motoru

- ① V první řadě přivedeme externí napájení do kontaktního pole. Napájení bude sloužit pouze k roztočení stejnosměrného motoru.

- ② V kontaktním poli propojíme země, které jsou přiveden z desky Arduino a z externího zdroje.
- ③ Stejnosměrný motor připojíme do kontaktního pole. Využijeme k tomu vodiče typu „zástrčka“. Vodiče k motoru připevníme buď připájením, nebo vytvořením oček a přelepením hmotou z tavné pistole nebo lepící páskou. Jeden vodič z motoru připojíme do kontaktního pole, k napájení. Druhý vodič připojíme do kontaktního pole, do střední části, ze které dále povedeme vodič k tranzistoru na **Kolektor**.
- ④ Tranzistor je vložen přímo do kontaktního pole. **Emitor** je připojený k zemnění. **Báze** je připojena k desce Arduino, do digitálního pinu 9. Kolektor je připojen k usměrňovací diodě a k motoru.
- ⑤ Usměrňovací dioda je paralelně připojena k motoru a chrání obvod proti zpětnému proudu.

PROGRAMOVÝ KÓD

Programový kód pro ovládání stejnosměrného motoru bude k regulaci otáček využívat aktuální teploty. Kód ovládání motoru je již zasazen do předchozího programu.

```

1 #include <Servo.h>
2
3 int termistorPin = 0;
4 int Vout;
5 float R2 = 10000;
6 float logR2, R1, T, Tc, Tf;
7 float c1 = 1.009249522e-03, c2 = 2.378405444e-04, c3 =
8 2.019202697e-07;
9
10 int servoPin = 7;
11 int setTemp = 25;
12 int setTempRet=22;
13
14 Servo myservo;
15
16 int dcMotorPin=9;
17 int dcMotorSpeed=0;
18
19
20 void setup() {
21   Serial.begin(9600);

```



```

22
23     myservo.attach( servoPin );
24     myservo.write( 0 );
25
26     pinMode(dcMotorPin, OUTPUT) └─③
27 }
28
29 void loop() {
30     Vout = analogRead(termistorPin);
31     R1 = R2 * (1023.0 / (float)Vout - 1.0);
32     logR1 = log(R1);
33     T = (1.0 / (c1 + c2*logR1 + c3*logR1*logR1*logR1));
34     Tc = T - 273.15;
35     Tf = (Tc * 9.0)/ 5.0 + 32.0;
36
37     Serial.print("Teplota: ");
38     Serial.print(Tf);
39     Serial.print(" F; ");
40     Serial.print(Tc);
41     Serial.println(" C");
42     delay(500);
43
44     if(T > setTemp ){
45         myservo.write(180);
46
47         dcMotorSpeed=map(T, setTemp, setTempRet, 32, 255); └─④
48         digitalWrite(dcMotorPin, dcMotorSpeed); └─⑤
49
50     }else if(T < setTempRet){
51         myservo.write(0);
52
53         digitalWrite(dcMotorPin,LOW); └─⑥
54
55     }
56     delay(1000);
57 }
```

- ① Deklarace proměnné **dcMotorPin**, která obsahuje číslo digitálního pinu pro připojení báze tranzistoru.
- ② Deklarace proměnné **dcMotorSpeed** pro uložení rychlosti otáček stejnosměrného motoru.
- ③ Definice pinu pro stejnosměrný motor.

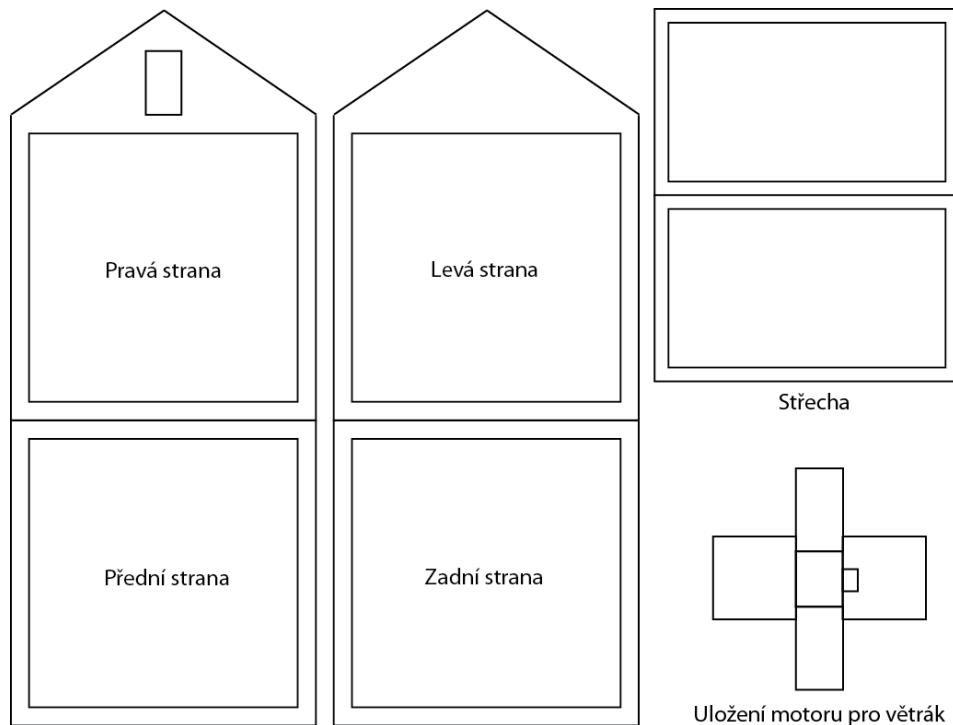
- ④ Namapování hodnot pro rychlosť motoru. Využívá se krajních hodnot teplot, pri ktorých dochází ke zmene otáček.
- ⑤ Funkcia **analogWrite()** posíla na výstup definovaný konštantou **dcMotorPin** nastavenou hodnotu rychlosťi **dcSpeedMotor**.
- ⑥ Pokud bude teplota nižšia než definovaná v konštante **setTempRet**, motor sa zastaví.



Pokud jste zvládli zapojit a naprogramovať všechny časti pro ovládání skleníku, můžete si takový model stolního skleníku vyrobit. Všechny elektronické časti pak do něj zabudovat.

MODEL SKLENÍKU

Pro vytvoření skleníku budeme potřebovat: karton (stará krabice), průsvitnou fólii, tavnou pistoli, lepidlo, kancelářskou sponku, lepící pásku, nůžky.



Obr. 11 - Šablona skleníku

PAPÍROVÁ KONSTRUKCE

V této kapitole je ukázka, jak sestrojit jednoduchý model skleníku, který lze po jeho sestavení umístit například na pracovní stůl.

V první řadě si překreslete na pevný karton model skleníku podle obrázku Obr. 11 - Šablona skleníku. Šablonu pak vystříhněte tak, aby vám vzniklo sedm samostatných dílů.



Obr. 12 – Vystřížená šablona

Nyní slepte pomocí lepící pásky vždy dvojici stěn: Pravá strana – Přední strana a Levá strana – Zadní strana, Obr. 14 – Slepění dílů stěn skleníku. Díly střechy také slepte k sobě.



Obr. 14 – Slepění dílů stěn skleníku



Obr. 13 – Nalepení fólie na stěny

Na všechny stěny, včetně střechy nalepte průhlednou fólii. Využijte k tomu obyčejného lepidla na papír, Obr. 13 – Nalepení fólie na stěny.

Dvojici dílů slepte páskou dohromady, **Chyba! Nenašel se zdroj odkazu.**

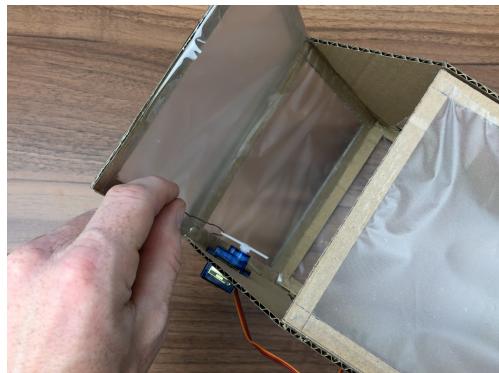


Obr. 18 – Sestavený skleník bez střechy

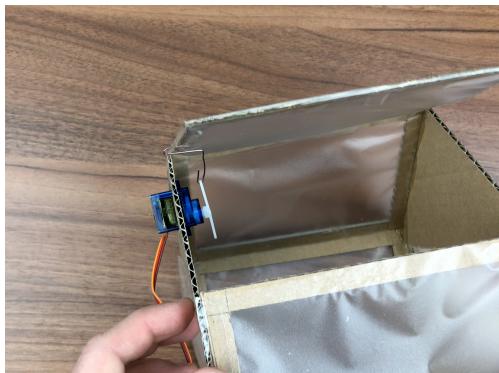


Obr. 17 – Přilepení střechy

Na stěny skleníku přilepte pomocí tavné pistole jednu polovinu střechy. Druhá polovina bude volná.



Obr. 15 – Připevnění sponky k servu

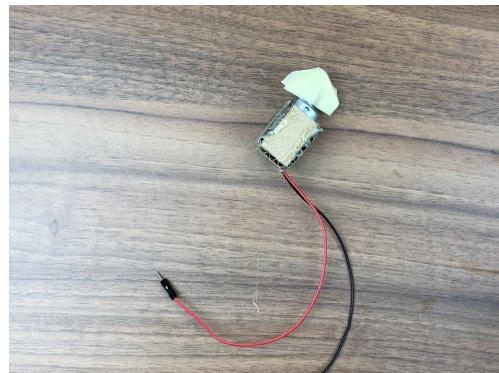


Obr. 16 – Spojení serva se střechou

Volná část střechy bude pomocí sponky spojená se servomotorem, který ji bude otevírat a zavírat. Sponku na ohýbejte podle obrázku Obr. 16 – Spojení serva se střechou.



Obr. 19 – Slepění pouzdra na motor



Obr. 20 – Motor s vrtulí

Poslední částí skleníku je větrák. Tavnou pistolí slepte úložnou kapsu, do které umístíte stejnosměrný motor. Na hřídel motoru přilepte lepící pásku jako vrtuli.