Spojená škola

Červenej armády 25, 036 01 Martin

PRÍRUČKA PRÍKLADOV PRE ARDUINO

Zbierka príkladov pre programovací modul Arduino UNO

2017
Martin

autor:
Ing. Juraj Svoboda

videoprojekcia:
www.majstrissmt.eu

Obsah

Príklad č. 1: Stopky 3
Príklad č. 2: Svetelný efekt Knight Rider 8
Príklad č. 3: Pulzar (pulzujúca LED) 11
Príklad č. 4: SOS (pomocou LED) 14
Príklad č. 5: Interaktívna LED 17
Príklad č. 6: Generátor zvuku 20
Príklad č. 7: Klavír 23
Príklad č. 8: LCD displej 26
Príklad č. 9: Servomotor 28
Príklad č. 10: Servomotor 2 29
Príklad č. 11: Servomotor 3 30
Príklad č. 12: Ovládanie servomotora potenciometrom 31
Príklad č. 13: Program s LED 32
Príklad č. 14: Melódia z piezoreproduktora 34
Príklad č. 15: Užívateľsky definované funkcie 36
Príklad č. 16: Čítanie potenciometra 38
Príklad č. 17: Blikač riadený potenciometrom 40
Príklad č. 18: Pohybové čidlo HC SR 501 42
Príklad č. 19: Infračervené čidlo prekážky 45
Príklad č. 20: Snímač teploty a vlhkosti 48
Príklad č. 21: Detektor dymu a alkoholu MQ-4 51
Príklad č. 22: Snímač vzdialenosti HC SR-04 53
Príklad č. 23: Optický snímač 56
Príklad č. 24: Magnetický spínač 58
Príklad č. 25: Detektor ohňa 60
Príklad č. 26: Robotický tank 61
Príklad č. 27: Robotický tank – pohyb dopredu – dozadu 65
Príklad č. 28: Robotický tank – otáčanie namieste vpravo – vľavo 66
Príklad č. 29: Robotický tank – pohyb do štvorca 67
Príklad č. 30: Robotický tank - ovládanie pomocou Blouthoot 69
Príklad č. 31: Robotický tank – projekt snímače 70

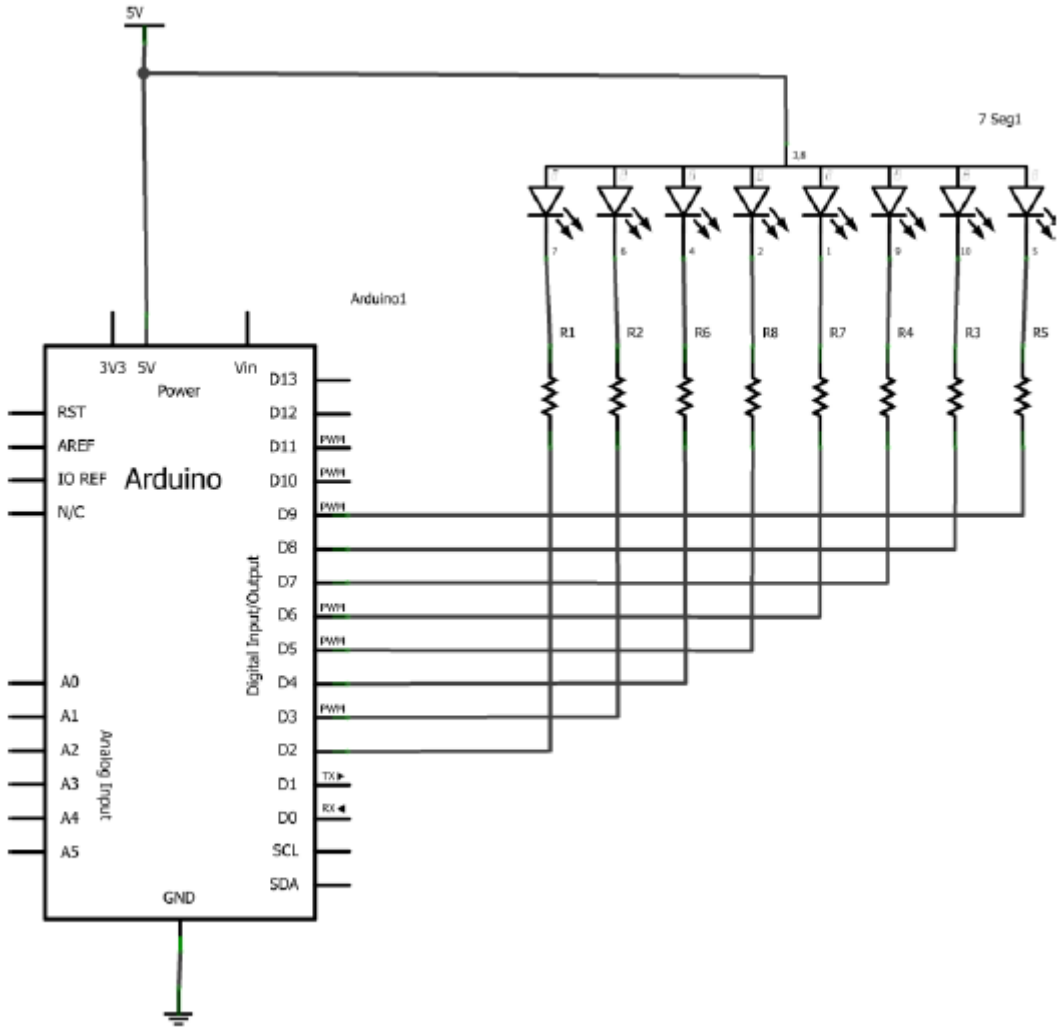
(Videoprojekcia k príkladom: www.majstrissmt.eu)

Príklad č. 1: Stopky

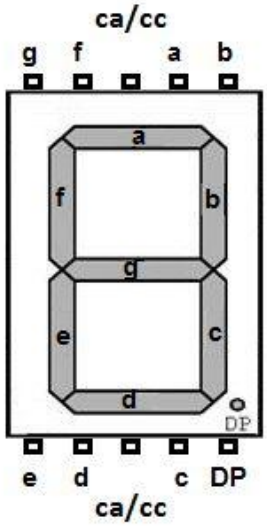
Rozpis materiálu:

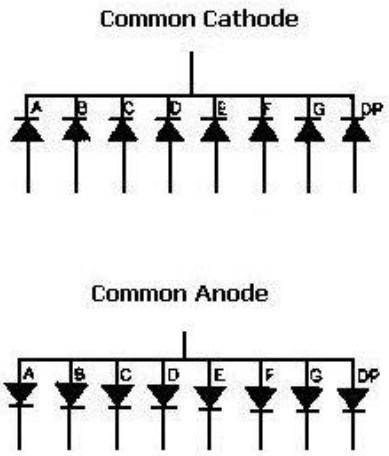
Označenie	*Hodnota*	*Počet kusov*
7 Segment1	LED display (7 segmentovka, so spoločnou anódou)	1
R1 až R8	Rezistor 150R	8
Breadboard1	Kontaktné pole	1
Arduino1	Arduino UNO R3	1
Prepojovacie káble		

Schéma zapojenia:

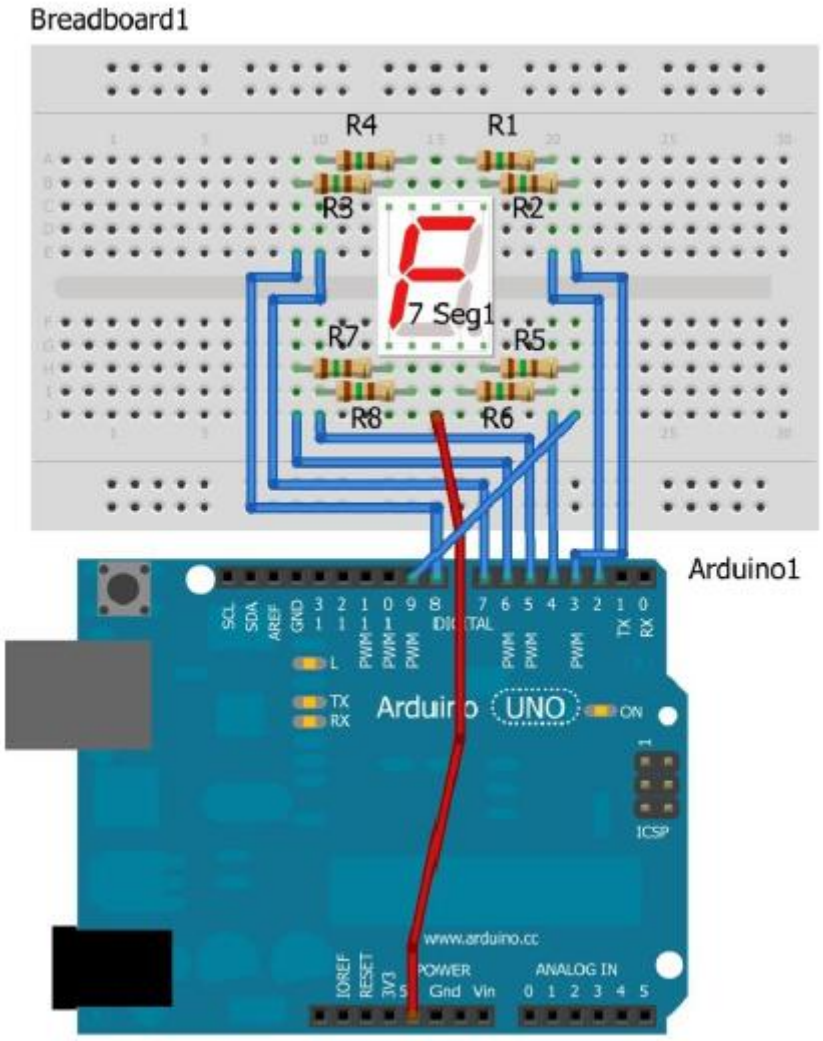


7 segmentový displej:





Zapojenie do kontaktnej plochy:



Program:

```
// čísla pinov segmentov
byte segmentPins[] = {2/*A*/,3/*B*/,4/*C*/,5/*D*/,6/*E*/,7/*F*/,8/*G*/,9/*BODKA*/};

// Pole aktívnych segmentov pre jednotlivé číslice, LOW = svieti, HIGH = nesvieti
byte segmentDigits[10][8] = {

//{ A , B , C , D , E , F , G , BODKA }

{LOW, LOW, LOW, LOW, LOW, LOW, HIGH, HIGH}, // číslica 0
{HIGH, LOW, LOW, HIGH, HIGH, HIGH, HIGH, HIGH}, // číslica 1
{LOW, LOW, HIGH, LOW, LOW, HIGH, LOW, HIGH}, // číslica 2
{LOW, LOW, LOW, LOW, HIGH, HIGH, LOW, HIGH}, // číslica 3
{HIGH, LOW, LOW, HIGH, HIGH, LOW, LOW, HIGH}, // číslica 4
{LOW, HIGH, LOW, LOW, HIGH, LOW, LOW, HIGH}, // číslica 5
{LOW, HIGH, LOW, LOW, LOW, LOW, LOW, HIGH}, // číslica 6
{LOW, LOW, LOW, HIGH, HIGH, HIGH, HIGH, HIGH}, // číslica 7
{LOW, LOW, LOW, LOW, LOW, LOW, LOW, HIGH}, // číslica 8
{LOW, LOW, LOW, LOW, HIGH, LOW, LOW, HIGH} // číslica 9
};

/* Metóda pre zobrazenie číslice */
void segmentDisplay(int digit);

void setup(){
// nastavíme porty ako vystupne

for(int i=0;i<8;i++){
pinMode(segmentPins[i], OUTPUT);
}

// nastavíme počiatočné hodnoty, nesvieti žiadna číslica
for(int i=0;i<8;i++){
digitalWrite(segmentPins[i], HIGH);
}
}

void loop(){
// Postupne zobrazíme čísla od 0 po 9

for(int n=0; n<10; n++){
segmentDisplay(n);
delay(500);
}

// Opačný postup, zobrazíme čísla od 9 po 0
for(int m=9; m>=0; m--){
segmentDisplay(m);
delay(500);
}
}

void segmentDisplay(int digit){
for(int s=0; s<8; s++){
digitalWrite(segmentPins[s], segmentDigits[digit][s]);

}
}
```

Popis programu:

Teraz si rozoberieme kód skice 8. Na začiatku máme inicializované dve premenné segmentPins a segmentDigits. Premenná segmentPins je pole typu byte a uchováva čísla výstupných pinov na ktoré máme pripojenú sedem segmentovku.
byte segmentPins[] = {2/*A*/,3/*B*/,4/*C*/,5/*D*/,6/*E*/,7/*F*/,8/*G*/,9/*BODKA*/};

V komentároch za číslom je segment ku ktorému je pin pripojený.
Druhá premenná segmentDigits je tiež pole typu byte ale dvojrozmerné, je to vlastne pole polí, v ktorých sú zadefinované hodnoty HIGH a LOW pre jednotlivé piny ku každej číslice.

```
byte segmentDigits[10][8] = {
//{ A , B , C , D , E , F , G , BODKA }
{LOW, LOW, LOW, LOW, LOW, LOW, HIGH, HIGH}, // číslica 0
{HIGH, LOW, LOW, HIGH, HIGH, HIGH, HIGH, HIGH}, // číslica 1
{LOW, LOW, HIGH, LOW, LOW, HIGH, LOW, HIGH}, // číslica 2
{LOW, LOW, LOW, LOW, HIGH, HIGH, LOW, HIGH}, // číslica 3
{HIGH, LOW, LOW, HIGH, HIGH, LOW, LOW, HIGH}, // číslica 4
{LOW, HIGH, LOW, LOW, HIGH, LOW, LOW, HIGH}, // číslica 5
{LOW, HIGH, LOW, LOW, LOW, LOW, LOW, HIGH}, // číslica 6
{LOW, LOW, LOW, HIGH, HIGH, HIGH, HIGH, HIGH}, // číslica 7
{LOW, LOW, LOW, LOW, LOW, LOW, LOW, HIGH}, // číslica 8
{LOW, LOW, LOW, LOW, HIGH, LOW, LOW, HIGH} // číslica 9
};
```

Toto pole má 10 riadkov (teda polí) pre každú jednu číslicu. Každé toto pole má zase 8 prvkov (teda toľko, koľko je pinov), prvky môžu mať hodnotu LOW alebo HIGH.

Rozoberme si prvý riadok v poli:

```
//{ A , B , C , D , E , F , G , BODKA }
{LOW, LOW, LOW, LOW, LOW, LOW, HIGH, HIGH}, // číslica 0
```

Prvý riadok určuje nastavenie pinov pre číslicu 0 (nula). Aby sa segment pripojený k pinu rozsvietil musí byť pripojený k zemi teda na 0V. Pretože máme použitý LED display so spoločnou anódou, tak vývody segmentov sú vlastne katódy LED diód v segmentoch, tzn. tieto katódy musia byť pripojené na záporný pól napätia a to docielime tým, že na výstupný pin zapíšeme hodnotu LOW. Teda bude platiť LOW = svieti, HIGH = nesvieti. Podľa tohto vidíme, že by mali svietiť segmenty A, B, C, D, E, a F. Segmenty G a BODKA majú nastavenú hodnotu HIGH teda nesvietia.

Ďalšie riadky v poli, resp. ďalšie polia určujú nastavenie pinov analogicky pre ostatné číslice 1, 2 až 9.

Ďalej máme definovanú metódu segmentDisplay:

```
void segmentDisplay(int digit);
```

Takýmto zápisom sme si zadefinovali novú metódu, ktorú môžeme neskoršie zavolať (použiť) v hocikde v programe. Ešte predtým je nutné urobiť nasledujúce:

```
void segmentDisplay(int digit){
for(int s=0; s<8; s++){
digitalWrite(segmentPins[s], segmentDigits[digit][s]);
}
}
```

A to, zapísať telo metódy tzv. implementáciu. Implementácia je zapísaná za metódou loop. Je to vlastne to, čo má metóda vlastne robiť. V krátkosti metóda nastaví piny tak aby sa rozsvietila konkrétna číslica na displeji. Metóda má jeden parameter typu int (integer). Týmto parametrom udávame, ktorá z číslic (0-9) sa má na displeji zobraziť.

A teraz sa vrátime ešte k metóde setup, kde sa nám nastavujú výstupné piny pre display a tiež nastavujeme počiatočné hodnoty pinov, tak aby nám na displeji nesvietilo žiadne číslo.

```
// nastavíme porty ako výstupné

for(int i=0;i<8;i++){
pinMode(segmentPins[i], OUTPUT);
}

// nastavíme počiatočné hodnoty, nesvieti žiadna číslica
for(int i=0;i<8;i++){
digitalWrite(segmentPins[i], HIGH);
}
```

V loop metóde máme dva cykly for. Prvý cyklus zobrazí postupne na displeji číslice od 0 po 9.

```
// Postupne zobrazíme čísla od 0 po 9

for(int n=0; n<10; n++){
segmentDisplay(n);
delay(500);
}
```

Druhý cyklus spraví analogicky to isté len opačným smerom zobrazí číslice od 9 po 0.

```
// Opačný postup, zobrazíme čísla od 9 po 0

for(int m=9; m>=0; m--){
segmentDisplay(m);
delay(500);
}
```

Zobrazenie každej číslice v oboch cykloch trvá pol sekundy, čo je spôsobené volaním delay metódy vo vnútri cyklu (delay(500)).

Alternatívne príklady:

1. Uprav program a zapojenie aby sa zobrazovali len párne čísla.
2. Uprav program a zapojenie aby sa zobrazovali len nepárne čísla.
3. Uprav program aby zobrazoval čísla len smerom nahor.
4. Uprav program aby zobrazoval čísla len smerom nadol.
5. Uprav program aby sa čísla zobrazovali po jednej sekunde.
6. Uprav program a zapojenie aby sa zobrazovali len čísla od jedna do päť.
7. Uprav program a zapojenie aby sa miesto číslic zobrazovali písmená A, B, C, D, E, F,G, H, I.

Príklad č. 2: Svetelný efekt – Knight Rider

Rozpis materiálu:

Označenie	*Hodnota*	*Počet kusov*
LED1-LED8	červená 5mm LED dióda	8
R1-R8	Rezistor 150R(150 Ohmov)	8
Breadboard1	Kontaktné pole	1
Arduino1	Arduino UNO R3	1
Prepojovacie káble		

Zapojenie do kontaktnej plochy:

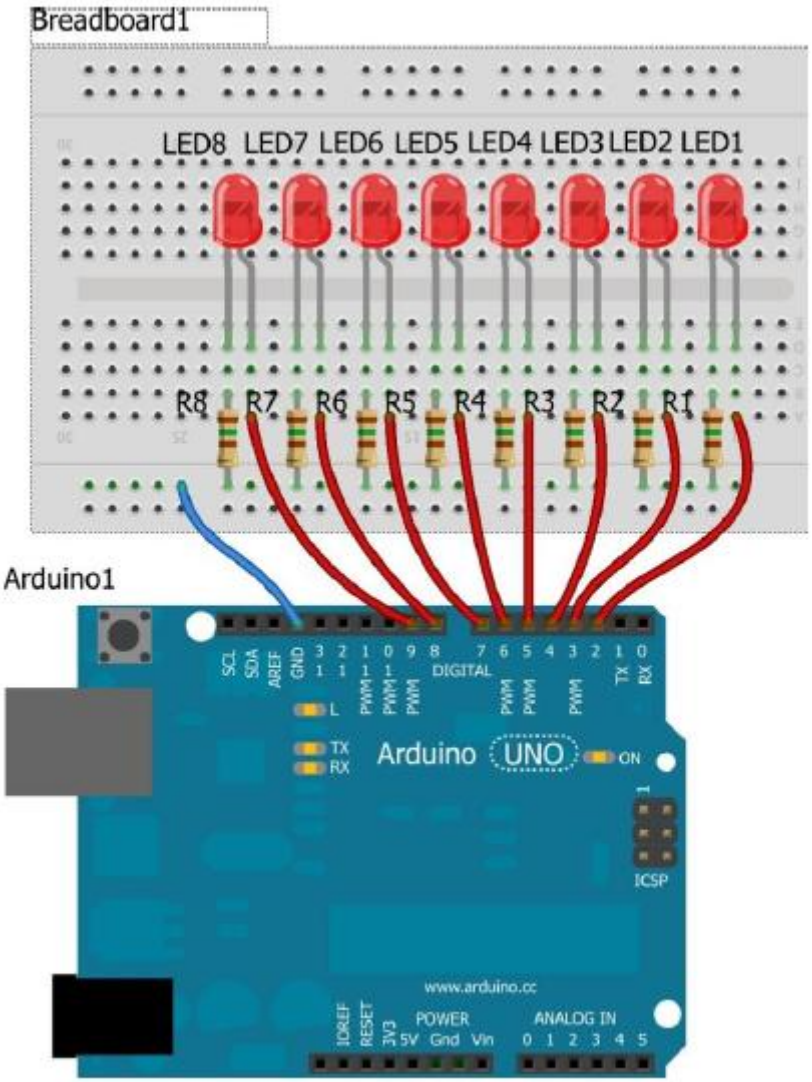
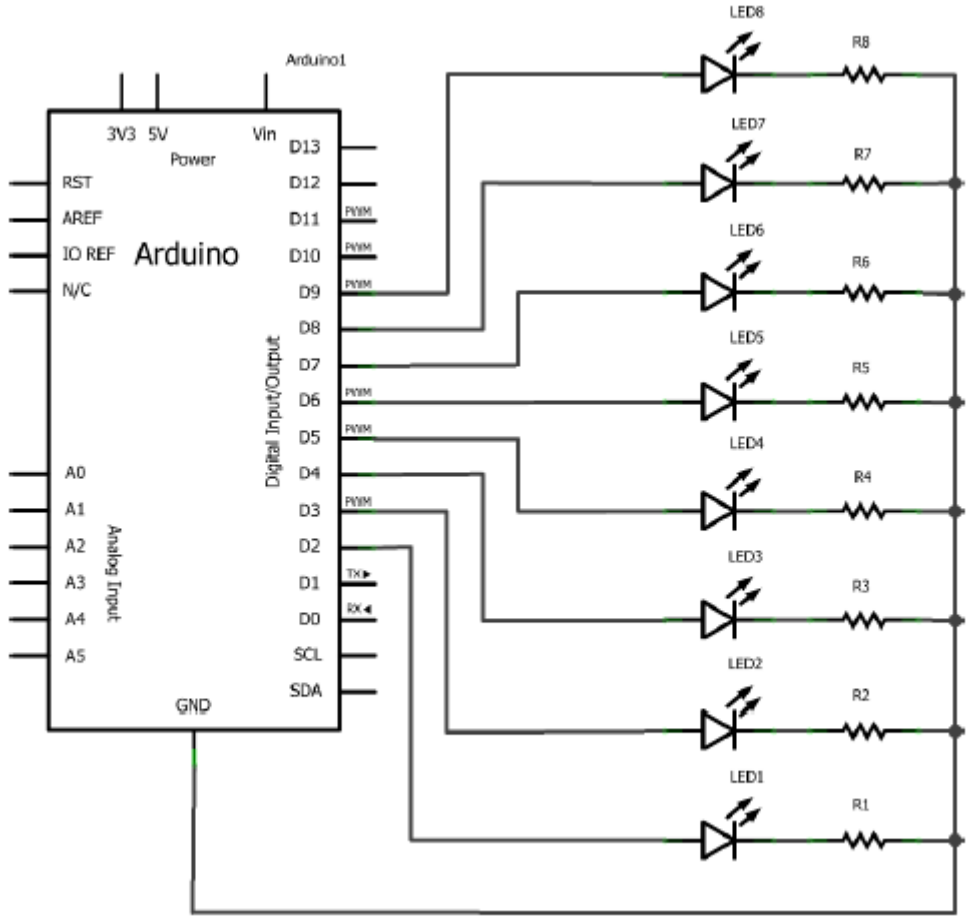


Schéma zapojenia:



Program:

```
/* čísla výstupných pinov */
byte outputPins[] = {2, 3, 4, 5, 6, 7, 8, 9};

void setup()
{
for(int i=0; i<8;i++)
{
pinMode(outputPins[i], OUTPUT);
digitalWrite(outputPins[i], LOW);
}
}

void loop()
{
for(int i=0; i<7;i++)
{
digitalWrite(outputPins[i], HIGH);
delay(75);
digitalWrite(outputPins[i], LOW);
}
for(int i=7; i>0;i--)
{
digitalWrite(outputPins[i], HIGH);
```

```
delay(75);
digitalWrite(outputPins[i], LOW);
}
}
```

Popis programu:

```
byte outputPins[] = { 2, 3, 4, 5, 6, 7, 8, 9 };
```

Tu máme inicializovanú premennú outpuPins, čo je vlastne pole typu byte. V tomto poli sú zadefinované čísla výstupných pinov 2 až 9. Polia sú indexované od nuly, čo znamená, že prvky v poli začínajú poradovým číslom 0. Napr. ak chceme získať prvý prvok, tak ho z poľa vyberieme nasledovne:

```
byte firstItem = outputPins[0];
```

Druhý prvok:

```
byte secondItem = outputPins[1];
```

. Ostatné prvky získame analogicky.

V setup metóde máme cyklus for, ktorý nám nastavuje piny na výstupné. Tiež nastavuje na výstupné piny hodnotu LOW, čo znamená, že všetky LEDky sú na začiatku programu vypnuté.

V loop metóde máme zase cyklus for a to dvakrát. Prvý cyklus spína LEDky v poradí od LED1 po LED7.

```
for(int i=0; i<7;i++)
{
digitalWrite(outputPins[i], HIGH);
delay(75);
digitalWrite(outputPins[i], LOW);
}
```

Premenná i v cykle, nadobúda hodnoty 0 až 6 a vo vnútri cyklu sa tak postupne nastavujú piny, ktorých čísla sa vyberajú z poľa outputPins podľa premennej i.
Druhý cyklus spína LEDky v opačnom poradí od LED8 po LED2. Cyklus tak pracuje analogicky ako prvý.

Alternatívne príklady:

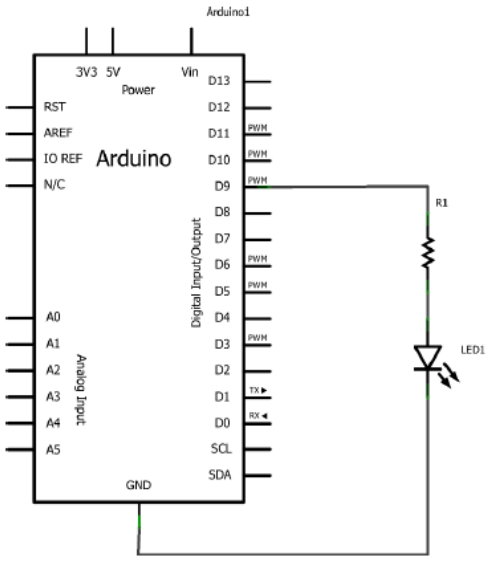
1. Uprav program a zapojenie pre 10 LED.
2. Uprav program aby sa pohybovali LEDky po dvojiciach.
3. Uprav program aby sa rozsvecovali LEDky len smerom doprava.
4. Uprav program aby sa rozsvecovali LEDky len smerom doľava.
5. Uprav program aby sa LEDky rozsvecovali smerom doprava len párne, smerom doľava nepárne.
6. Uprav program aby sa LEDky rozsvecovali s postupným časovým oneskorením.
7. Uprav zapojenie a program aby LEDky boli uložené do kružnice v počte 10.

Príklad č. 3: Pulzar (Pulzujúca LED)

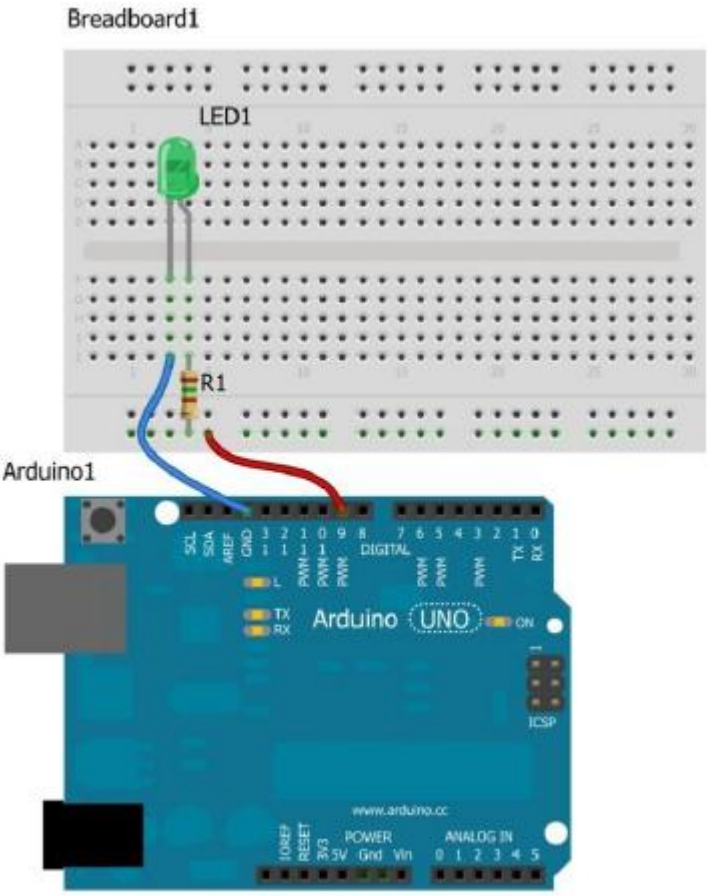
Rozpis materiálu:

Označenie	*Hodnota*	*Počet kusov*
LED1	Zelená 5mm LED dióda	1
R1	Rezistor 150R(150 Ohmov)	1
Breadboard1	Kontaktné pole	1
Arduino1	Arduino UNO R3	1
Prepojovacie káble		

Schéma zapojenia:



Zapojenie do kontaktnej plochy:



Program:

```
int ledPin = 9; // číslo pinu na ktorý je pripojená LEDka
int brightness = 0; // jas LEDky
int stepValue = 5; // veľkosť kroku na nastavenie jasu LEDky
int direction = 1; // Smer zvyšovanie alebo znižovanie jasu

void setup() {
pinMode(ledPin, OUTPUT);
}

void loop() {
// nastavenie jasu LEDky
analogWrite(ledPin, brightness);

// ak jas dosiahne maximálnej hodnoty, zmeníme smer na znižovanie
if(brightness == 255){
direction = 2;
}

// ak jas dosiahne minimálnej hodnoty, zmeníme smer na zvyšovanie
if(brightness == 0) {
direction = 1;
}

switch(direction){
case 1: // zvyšovanie jasu
brightness = brightness + stepValue;
break;
case 2: // znižovanie jasu
brightness = brightness - stepValue;
break;
}
delay(40);
}
```

Popis programu:

V skice sa nám tentokrát objavili ďalšie premenné okrem ledPin, ktorá má tentokrát hodnotu 9, pretože LEDku máme zapojenú v pine číslo 9.

```
int brightness = 0;
```

Táto premenná slúži ako aktuálna hodnota jasu. Môže nadobúdať hodnoty od 0 po 255.

```
int stepValue = 5;
```

Premenná stepValue určuje veľkosť o koľko sa zvýši alebo zníži jas každým cyklom (teda hodnota premennéj brightness) . Hodnota tejto premennej je 5 a v programe sa nemení.

```
int direction = 1;
```

A nakoniec máme premennú direction, ktorá určuje či sa hodnota jasu bude zvyšovať alebo znižovať.

Premenná direction môže mať hodnotu 1 alebo 2. Hodnota 1 určuje zvyšovanie jasu a hodnota 2 zase určuje znižovanie.

Setup metóda sa nám opäť nezmenila, preto si popíšeme metódu loop.

```
analogWrite(ledPin, brightness);
```

Metóda analogWrite má dva parametre. Prvý parameter je číslo pinu a druhý parameter je hodnota z rozsahu 0 až 255. Táto metóda urobí to, že nám na pin číslo 9 zapíše hodnotu z premennej brightness, táto hodnota sa na výstupnom pine objaví ako napätie v rozsahu 0 – 5 voltov. Napríklad ak chceme mať na výstupnom pine 2.5V, tak druhý parameter musí mať hodnotu 127. Metóda analogWrite generuje tzv. PWM signál, pomocou ktorého je možné regulovať výkon, čí už jas pri LEDkách alebo otáčky pri motoroch.

Alternatívne príklady:

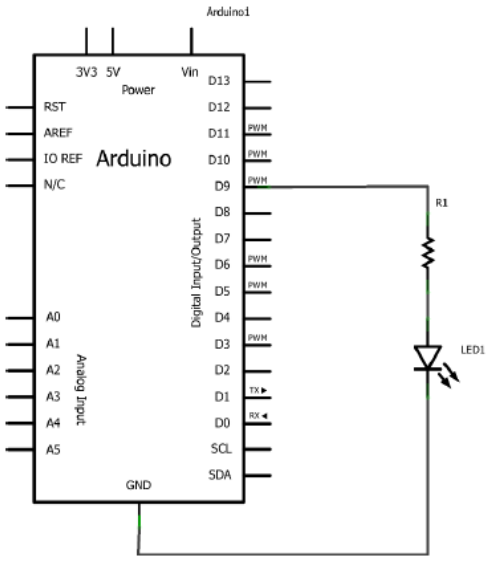
1. Uprav program a zapojenie pre 2 LED, kým prvá bude jas zvyšovať, druhá bude jas znižovať.
2. Uprav program a zapojenie pre 2 LED, druhá LED bude jas meniť dvojnásobne rýchlejšie.
3. Uprav program aby po maximálnom a minimálnom jase LED zhasla na 1 sekundu.
4. Uprav program aby sa pulz LED dvojnásobne zrýchlil.
5. Uprav program aby sa pulz LED dvojnásobne spomalil.

Príklad č. 4: SOS pomocou LED

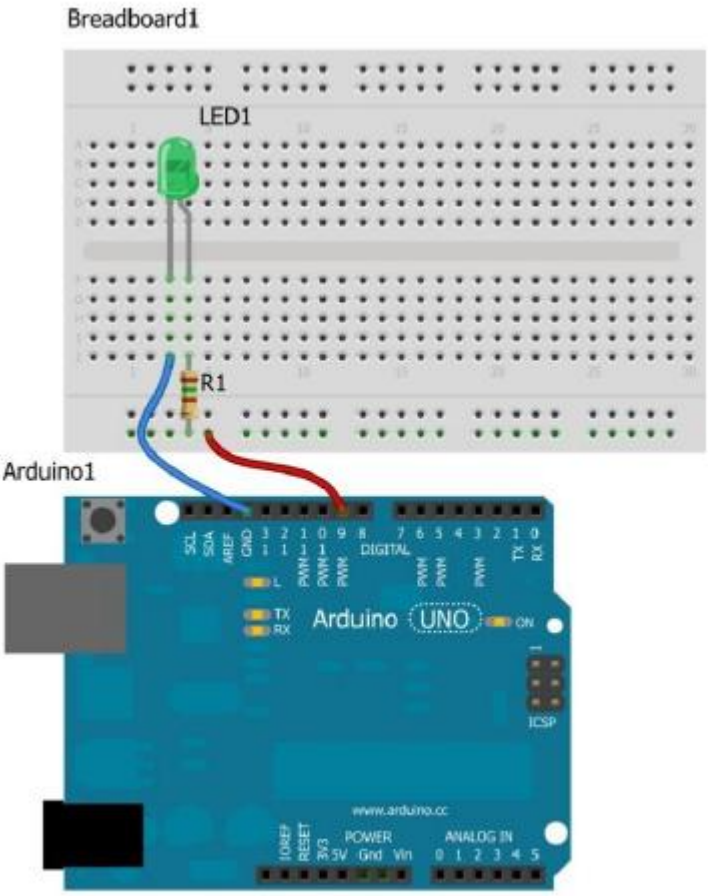
Rozpis materiálu:

Označenie	*Hodnota*	*Počet kusov*
LED1	Zelená 5mm LED dióda	1
R1	Rezistor 150R(150 Ohmov)	1
Breadboard1	Kontaktné pole	1
Arduino1	Arduino UNO R3	1
Prepojovacie káble		

Schéma zapojenia:



Zapojenie do kontaktnej plochy:



Program:

```
int ledPin = 9;
void setup() {

// nastavíme ledPin (pin číslo 13) ako výstupný
pinMode(ledPin, OUTPUT);
}

void loop() {

// 3 krátke '...' v morseovej abecede písmeno 'S'

for(int i=0; i<3; i++){
digitalWrite(ledPin, HIGH);
delay(120);
digitalWrite(ledPin, LOW);
delay(120);
}

delay(100); //čakáme 100 milisekúnd

// 3 dlhé '---' v morseovej abecede písmeno 'O'

for(int i=0; i<3; i++){
digitalWrite(ledPin, HIGH);
delay(350);
digitalWrite(ledPin, LOW);
delay(350);
}

delay(100);

//zase čakáme 100 milisekúnd
//a zase 3 krátke '...' teda 'S'

for(int i=0; i<3; i++){
digitalWrite(ledPin, HIGH);
delay(120);
digitalWrite(ledPin, LOW);
delay(120);
}
delay(5000);

//celý cyklus sa spustí znova po 5 sekundách


}
```

Popis programu:

Inicializa premennej ledPin sa nelíši od predchádzajúceho príkladu, taktiež v setup metóde sa nič nezmenilo. Zmenu môžeme vidieť v loop metóde, kde máme tri cykly **for .**

Cyklus for funguje nasledovne:

for(inicializa premennej; ***podmienka***; zmena premennej){
Kód vo vnútri sa vykonáva dovtedy, pokiaľ platí ***podmienka***.
}

V našom prípade má cyklus for v sebe premennú **i** typu **int**, s počiatočnou hodnotou **0**. Podmienka v cykle je **i < 3**, čo znamená, že cyklus sa bude vykonávať pokiaľ premenná i bude menšia ako 3. Premenná sa mení inkrementáciou **i++**, tzn. hodnota sa zvyšuje o jedna.

```
for(int i=0; i<3; i++){
...
}
```

Každým cyklom sa teda zvýši hodnota premennej o jedna a v nasledujúcom cykle sa v podmienke kontroluje. Ak hodnota vyhovuje, hovoríme, že podmienka bola splnená - cyklus pokračuje, ale ak hodnota premennej i nevyhovuje podmienke = podmienka nebola splnená = cyklus sa ukončí a beh programu pokračuje ďalej za cyklom for (výsledok podmienky môže byť **true = pravda**, alebo **false = nepravda**).

Vysvetlili sme si ako funguje cyklus for. Teraz sa pozrieme dovnútra cyklu, máme tam nasledujúci kód:

```
digitalWrite(ledPin, HIGH);
delay(120);
digitalWrite(ledPin, LOW);
delay(120);
```

Čo je vlastne rovnaký kód ako v predchádzajúcom príklade, len je tu nastavená kratšia doba svietenia LEDky na 120 ms. Podľa podmienky v cykle sa kód vo vnútri vykoná trikrát, tzn. LEDka zasvieti a zhasne trikrát za sebou. Ostatné cykly fungujú analogicky, s tým, že v strednom cykle je nastavené oneskorenie o niečo väčšie(350 ms).

Medzi cyklami je tiež nastavené oneskorenie na 100 ms. Lepšie tak uvidíme zmenu v blikaní LEDky = lepšie rozoznáme znaky Morseovej abecedy.
Na konci vidíme volanie metódy delay s hodnotou parametra 5000, teda oneskorenie 5s, tzn. celý proces blikania sa opakuje od začiatku po piatich sekundách.

Alternatívne príklady:

1. Uprav program aby LED vysielala signál v Morzeovke OSOL.
2. Uprav program aby LED vysielala signál v Morzeovke POMOC.
3. Uprav program aby LED blikala 5 x rýchlo a následne 5 x pomaly.
4. Uprav program aby LED blikala 5 x rýchlo a následne 2 x pomaly.
5. Uprav program aby LED vysielala signál v Morzeovke AHOJ.

Príklad č. 5: Interaktívna LED

Rozpis materiálu:

Označenie	*Hodnota*	*Počet kusov*
LED1	červená 5mm LED dióda	1
R1	Rezistor 150R	1
R2	Rezistor 10k	1
S1	Tlačítko	1
Breadboard1	Kontaktné pole	1
Arduino1	Arduino UNO R3	1
Prepojovacie káble		

Zapojenie do kontaktnej plochy:

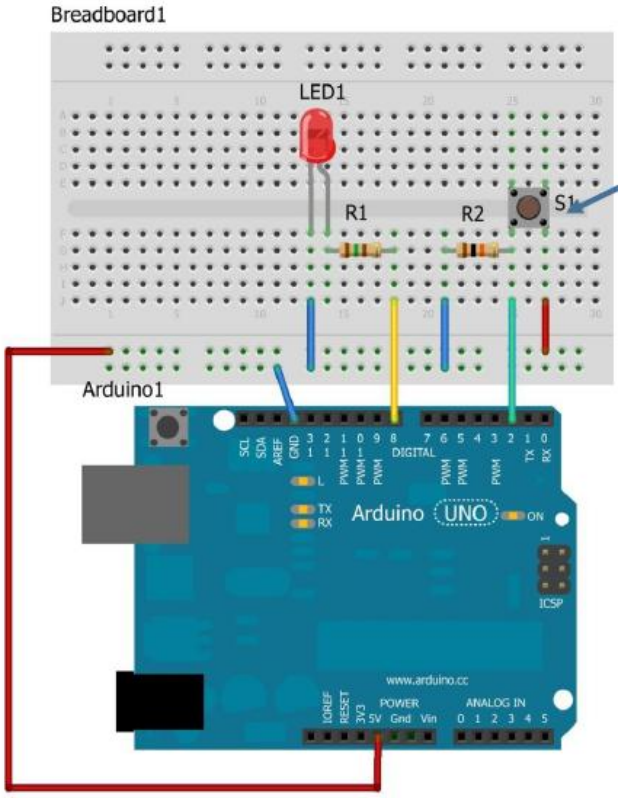
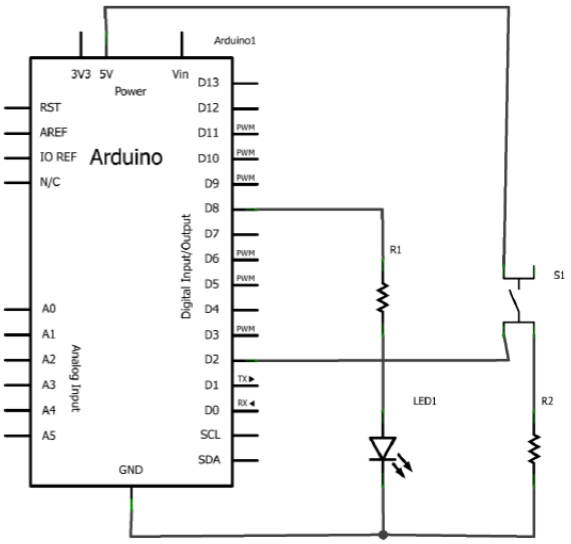


Schéma zapojenia:



Program:

```
//číslo pinu pre tlačítko
int buttonPin = 2;

//číslo pinu pre LEDku
int ledPin = 8;

//premenná uchovávajuca stav tlačítka
int buttonState = 0;

void setup() {

// nastavenie pinu pre LEDku ako výstupný
pinMode(ledPin, OUTPUT);

// nastavenie pinu pre tlačítko ako vstupný
pinMode(buttonPin, INPUT);
}

void loop(){

// načítame stav tlačítka
buttonState = digitalRead(buttonPin);

// skontrolujeme či je tlačitko stlačené alebo nie
// ak je, tzn. premenná buttonState má hodnotu HIGH
if (buttonState == HIGH) {
digitalWrite(ledPin, HIGH); // zapneme LEDku
} else {

// ak tlačítko nie je zopnuté (stlačené), LEDku vypneme
digitalWrite(ledPin, LOW);
}
}
```

Popis programu:

Na začiatku sa nám inicializujú premenné buttonPin, ledPin a buttonState.

```
//číslo pinu pre tlačítko
int buttonPin = 2;

//číslo pinu pre LEDku
int ledPin = 8;

//premenná uchovávajuca stav tlačítka
int buttonState = 0;
```

Popis premenných je možné vidieť v komentároch nad nimi. Túto časť kódu už dobre poznáte z predchádzajúcich príkladov, tak sa ďalším popisom nemusíme zaoberať.

V setup metóde ako zvyčajne nastavujeme mód jednotlivých pinov.

```
void setup() {

// nastavenie pinu pre LEDku ako výstupný
pinMode(ledPin, OUTPUT);
```

```
// nastavenie pinu pre tlačítko ako vstupný
pinMode(buttonPin, INPUT);
}
```

V tomto príklade máme malú zmenu a to tím, že nastavujeme pin 2 ako vstupný. Na tento pin je zapojené totiž tlačítko, ktorého stav budeme sledovať.

V loop metóde máme použitú novú metódu digitalRead. Táto metóda nám vracia stav na určitom pine, ktorý zadáme ako jediný parameter.

```
int pinState = digitalRead(pinNumber);
```

V našej skice máme ako parameter zadanú premennú buttonPin, teda pin číslo 2.

```
buttonState = digitalRead(buttonPin);
```

Stav pinu, ktorý nám vráti metóda digitalRead sa nám uloží do premennej buttonState.

V ďalšej časti kódu porovnávame tento stav, tzn. zisťujeme či bolo tlačítko stlačené alebo nie.

```
if (buttonState == HIGH) {
digitalWrite(ledPin, HIGH); // zapneme LEDku

} else {

// ak tlačítko nie je zopnuté (stlačené), LEDku vypneme
digitalWrite(ledPin, LOW);
}
```

Ak je premenná buttonState rovná HIGH znamená to, že tlačítko je stlačené a tak sa vykoná kód:
`digitalWrite(ledPin, HIGH);`

A vy už viete, že táto časť kódu spôsobí to že nám LEDka bude svietiť. Ale ak sa stav tlačítka nezhoduje z HIGH stavom, čo znamená, že tlačítko nie je zopnuté, vykoná sa kód v bloku **else**:
`digitalWrite(ledPin, LOW);`

Čo spôsobí, že LEDka nebude svietiť, resp. zhasne, ak sme tlačítko pustili.

Alternatívne príklady:

1. Uprav program a zapojenie pre ovládanie 2 LED rôznych farieb.
2. Uprav program a zapojenie pre ovládanie 3 LED rôznych farieb.

Príklad č. 6: Generátor zvuku

Rozpis materiálu:

Označenie	*Hodnota*	*Počet kusov*
LED1	červená 5mm LED dióda	1
R1	Rezistor 150R	1
R2	Rezistor 10k	1
S1	Tlačítko	1
J1	Piezo menič, alebo reproduktor	1
Arduino1	Arduino UNO R3	1
Prepojovacie káble		

Zapojenie do kontaktnej plochy:

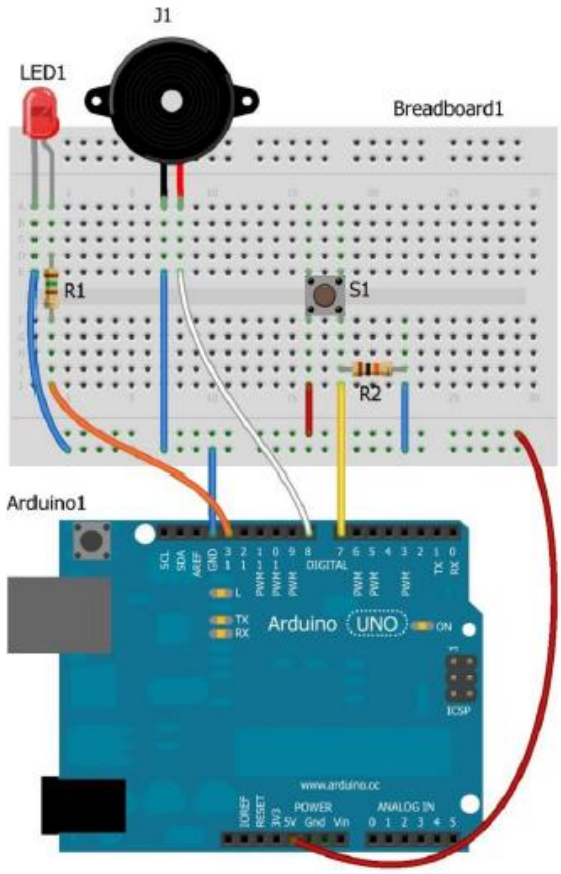
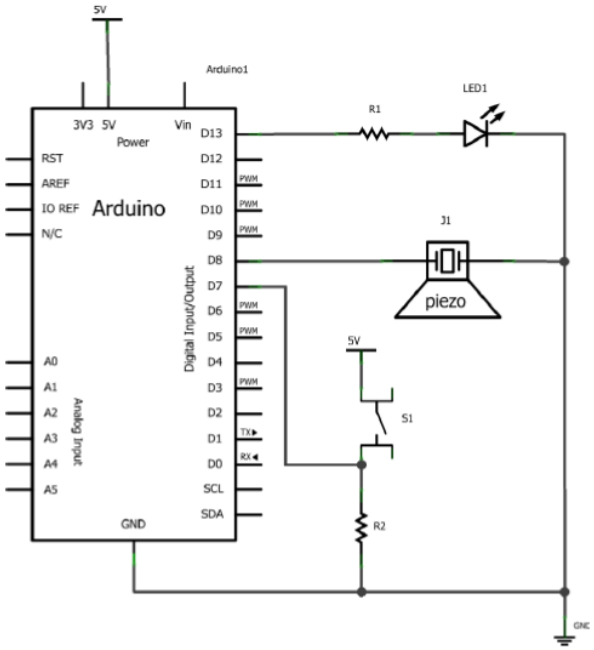


Schéma zapojenia:



Toto zapojenie je už o niečo zložitejšie, ako predchádzajúce. Zapojenie LEDky a tlačidla je v podstate rovnaké ako v predchádzajúcom priklade. Pribudlo nám tu piezogenerátor, označené J1. Piezo J1 má dva vývody, jeden je označený čiernou a druhý červenou farbou. Vývod s čiernou farbou pripojíme na zem a vývod s červenou farbou na Arduino, na pin číslo 8. Ak vývody vášho piezomeniča nie sú farebne odlíšené, zapojte vývody ľubovoľne (nebojte sa nič tým nepokazíte).

Program:

```
// číslo pinu pre LED
int ledPin = 13;

// číslo pinu pre tlačidlo
int buttonPin = 7;

// číslo pinu pre piezo
int piezoPin = 8;

// premenná uchovávajuca stav tlačidla
int buttonState = 0;

void setup() {

// nastavenie pinu pre LEDku ako výstupný
pinMode(ledPin, OUTPUT);

// nastavenie pinu pre piezo ako výstupný
pinMode(piezoPin, OUTPUT);

// nastavenie pinu pre tlačidlo ako vstupný
pinMode(buttonPin, INPUT);
}

void loop(){

// načítame stav tlačidla
buttonState = digitalRead(buttonPin);

// skontrolujeme či je tlačidlo stlačené
// ak je, tzn. premenná buttonState má hodnotu HIGH
if (buttonState == HIGH) {

// zapneme LEDku
digitalWrite(ledPin, HIGH);

// zapneme generovanie tónu
tone(piezoPin, 1000, 100);
delay(100);

// ak nie je, tzn. premenná buttonState má hodnotu LOW
} else {

// vypneme LEDku
digitalWrite(ledPin, LOW);

// vypneme generovanie tónu
noTone(piezoPin); }
}
```

Popis programu:

Kód skice sa veľmi nelíši od toho predchádzajúceho. Inicializácia premenných je skoro rovnaká, akurát nám pribudla premenná piezoPin, ktorá má hodnotu 8. V setup metóde tak následne pribudlo nastavenie pinu pre piezo:

```
// nastavenie pinu pre piezo ako výstupný
pinMode(piezoPin, OUTPUT);
```

V loop metóde je to tiež veľmi podobné, máme tu ešte dve nové metódy a to **tone** a **noTone**. Metóda tone vygeneruje jednoduchý tón za pomoci PWM. Má tri vstupné parametre z toho je posledný nepovinný.

tone(pin, frekvencia, trvanie);

alebo

tone(pin, frekvencia);

Prvý parameter **pin**, je číslo výstupného pinu na ktorom sa bude generovať tón. Druhý parameter **frekvencia,** je frekvencia tónu v Hz (Hertz), v našej skice máme frekvenciu 1000Hz (1kHz). Posledný parameter ktorý je nepovinný je **trvanie** tónu v milisekundách, v našom prípade 100ms.

tone(piezoPin, 1000, 100);

Metóda noTone ukončí generovanie tónu na pine, ktorý zadáme ako vstupný parameter do tejto metódy.

noTone(pin);

V našom projekte voláme metódu noTone, kde ako hodnotu parametra predávame hodnotu z premennej piezoPin.

noTone(piezoPin);

Čo znamená, že ak tlačítko nie je zopnuté generovanie tónu sa ukončí.

Alternatívne príklady:

1. Uprav program a zapojenie pre ovládanie 2 LED a 2 piezogenerátorov, ktoré budú nastavené na iné frekvencie .
2. Uprav program a zapojenie pre ovládanie z piezogenerátora inou frekvenciou a iným časom .
3. Uprav program a zapojenie pre ovládanie 1 LED a 1 piezogenerátora pomocou 3 tlačidiel, každé tlačidlo zopne generátor s inou frekvenciou.

Príklad č. 7: Klavír

Rozpis materiálu:

Označenie	*Hodnota*	*Počet kusov*
LED1	červená 5mm LED dióda	1
R1	Rezistor 150R	4
R2	Rezistor 10k	4
S1	Tlačítko	4
J1	Piezo menič, alebo reproduktor	1
Arduino1	Arduino UNO R3	1
Prepojovacie káble		

Zapojenie do kontaktnej plochy - podľa programu:

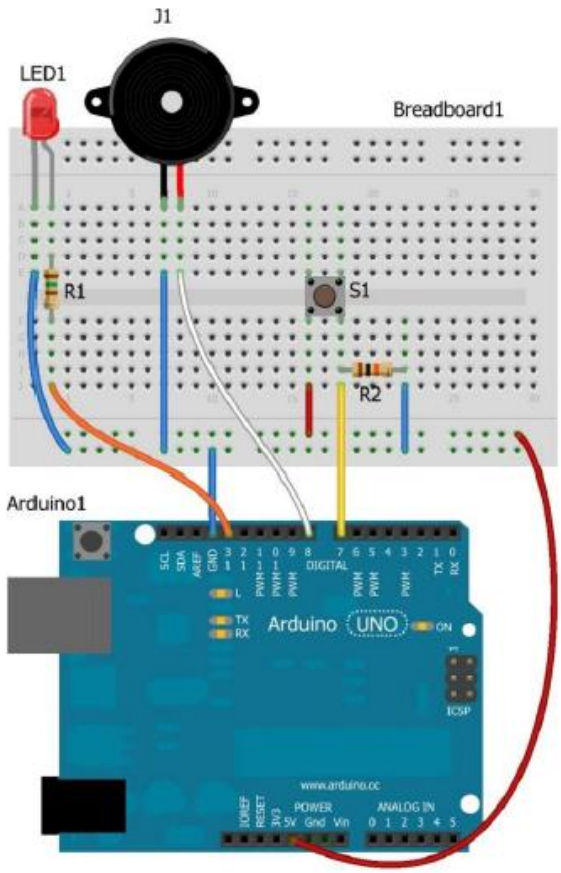
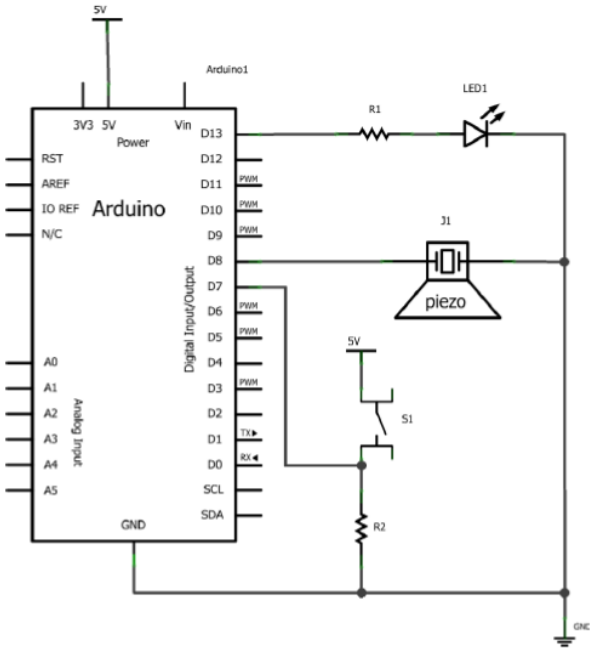


Schéma zapojenia – podľa programu:



Program:

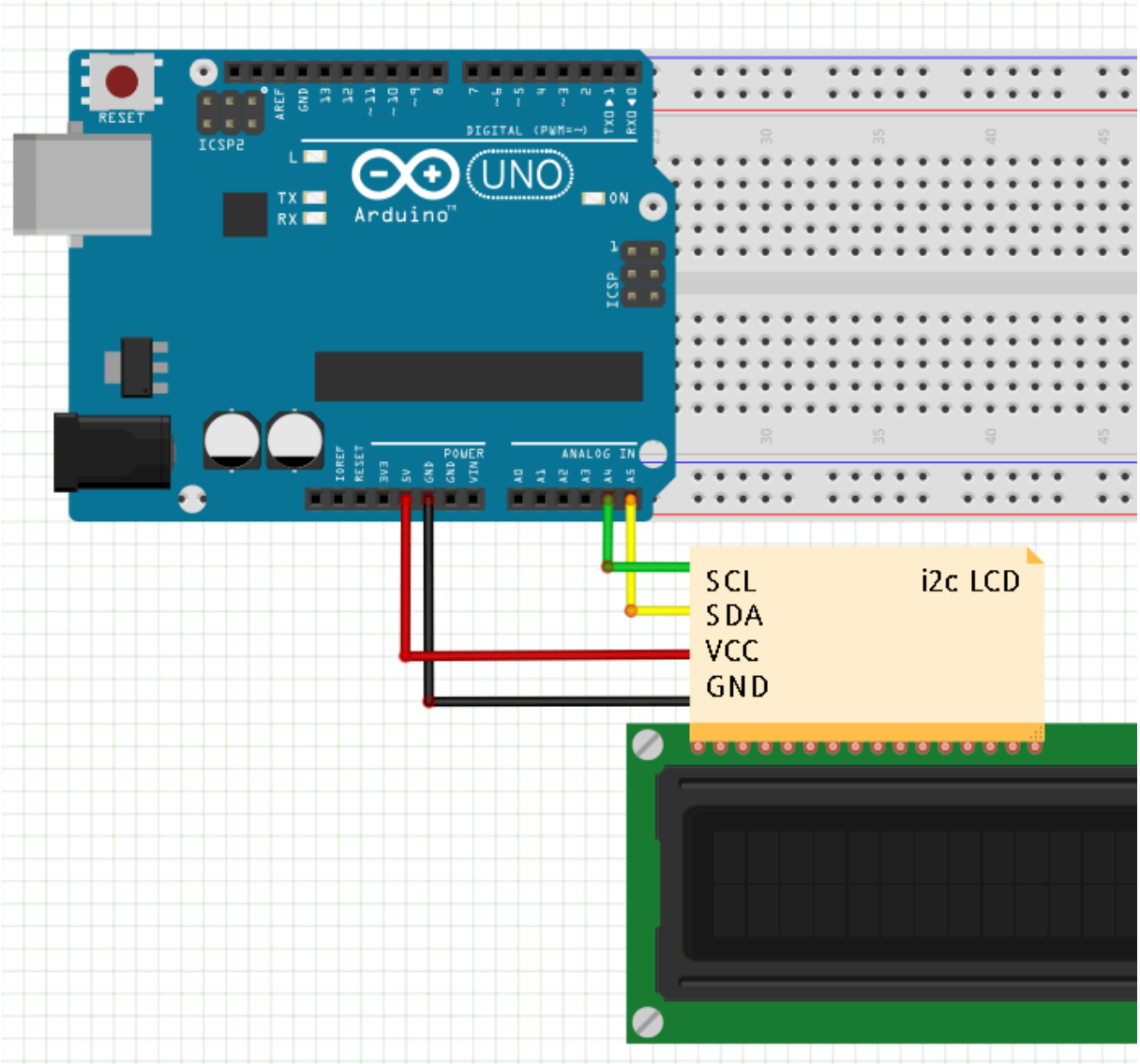
```
// číslo pinu pre LED
int ledPin = 13;

// čísla pinov pre tlačidlá
int buttonPin1 = 7;
int buttonPin2 = 6;
int buttonPin3 = 5;
int buttonPin4 = 4;

// číslo pinu pre piezo
int piezoPin = 8;

// premenné uchovávajúce stav tlačidiel
int buttonState1 = 0;
int buttonState2 = 0;
int buttonState3 = 0;
int buttonState4 = 0;


void setup() {

// nastavenie pinu pre LEDku ako výstupný
pinMode(ledPin, OUTPUT);

// nastavenie pinu pre piezo ako výstupný
pinMode(piezoPin, OUTPUT);

// nastavenie pinov pre tlačidlá ako vstupné
pinMode(buttonPin1, INPUT);
pinMode(buttonPin2, INPUT);
pinMode(buttonPin3, INPUT);
pinMode(buttonPin4, INPUT);
}

void loop(){

// načítame stavy  tlačidiel
buttonState1 = digitalRead(buttonPin1);
buttonState2 = digitalRead(buttonPin2);
buttonState3 = digitalRead(buttonPin3);
buttonState4 = digitalRead(buttonPin4);


// skontrolujeme či sú tlačidlá stlačené
// ak je, tzn. premenná buttonState má hodnotu HIGH

if (buttonState1 == HIGH) {
digitalWrite(ledPin, HIGH); // zapneme LEDku
tone(piezoPin, 1000, 100);
delay(100);
} else { // ak nie je, tzn. premenná buttonState má hodnotu LOW
digitalWrite(ledPin, LOW); // vypneme LEDku
noTone(piezoPin); // vypneme generovanie tónu
}

if (buttonState2 == HIGH) {
digitalWrite(ledPin, HIGH); // zapneme LEDku
tone(piezoPin, 2000, 100);
```

```
delay(100);
} else { // ak nie je, tzn. premenná buttonState má hodnotu LOW
digitalWrite(ledPin, LOW); // vypneme LEDku
noTone(piezoPin); // vypneme generovanie tónu
}

if (buttonState3 == HIGH) {
digitalWrite(ledPin, HIGH); // zapneme LEDku
tone(piezoPin, 3000, 100);
delay(100);
} else { // ak nie je, tzn. premenná buttonState má hodnotu LOW
digitalWrite(ledPin, LOW); // vypneme LEDku
noTone(piezoPin); // vypneme generovanie tónu
}

if (buttonState4 == HIGH) {
digitalWrite(ledPin, HIGH); // zapneme LEDku
tone(piezoPin, 4000, 100);
delay(100);
} else { // ak nie je, tzn. premenná buttonState má hodnotu LOW
digitalWrite(ledPin, LOW); // vypneme LEDku
noTone(piezoPin); // vypneme generovanie tónu
}

}
```

Alternatívne príklady:

1. Uprav program a zapojenie pre ovládanie 1 LED a 1 piezogenerátora a 3 tlačidlá .
2. Uprav program a zapojenie pre ovládanie 3 LED , 3 piezogenerátorov a 3 tlačidiel.
3. Uprav program a zapojenie pre ovládanie 1 LED , 1 piezogenerátor a 5 tlačidiel.
4. Uprav program a zapojenie pre ovládanie 1 LED a 1 piezogenerátora pomocou 3 tlačidiel.

Príklad č. 8: LCD displej




```
/*
  LCD i2c
  beziaci text
 */

#include <Wire.h>                          // zahrnutie kniznic do programu
#include <LiquidCrystal_I2C.h>

LiquidCrystal_I2C lcd(0x27,16,2);          // inicializacia kniznice (adresa, znaky, riadky)

int i=200;                                 // premenna zdrzania pri posune textu

void setup()
{
  lcd.init();                              // inicializacia LCD
  lcd.backlight();                         // zapnutie podsvietenia
  lcd.print("Toto je text, ktory bezi sprava dolava.");              // vypis retazca textu
}
```

```
void loop()
{
  lcd.scrollDisplayLeft();                  // posun textu o jeden znak vlavo
  delay(i--);                               // cakanie
  if (i<0) {i=200;}                          // znovu od pomaleho pohybu
}
```

Alternatívne príklady:

1. Uprav program pre pohyb textu zlava doprava .
2. Uprav program pre blikanie textu.
3. Uprav program pre blikanie textu a následne pohyb textu sprava dolava.

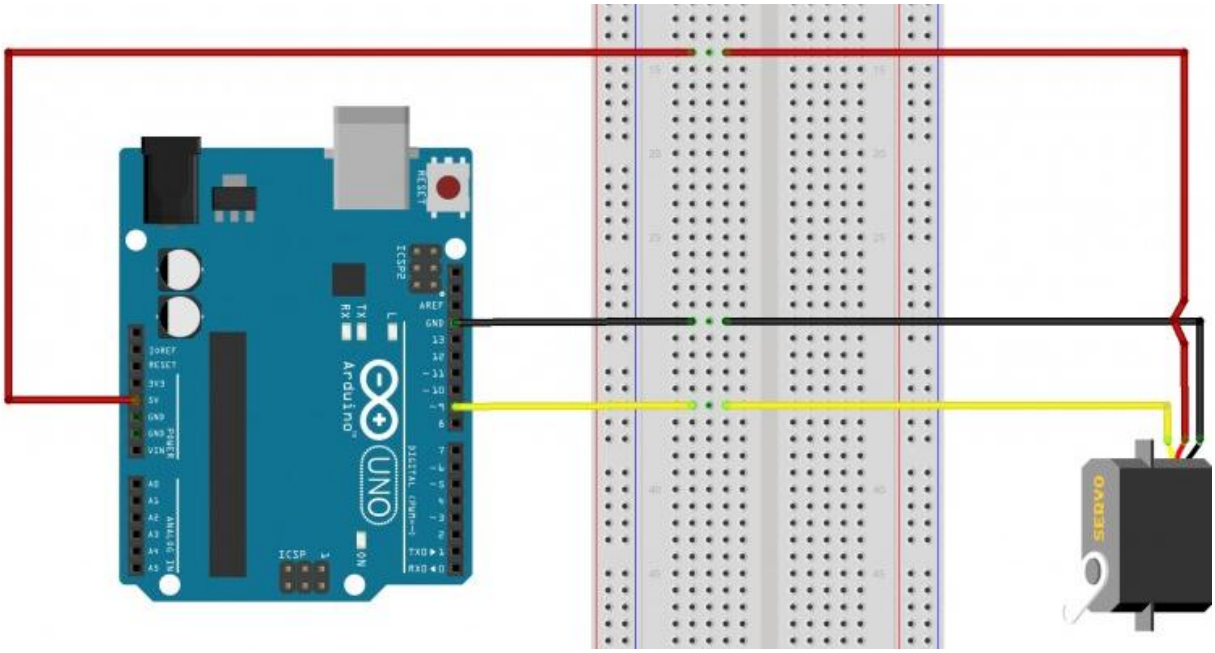
Príklad č. 9: SERVOMOTOR

Tento návod ukáže, ako ovládať smer, polohu a rýchlosť **servomotora SG90 9G Micro** s Arduino UNO doskou.

Servo motor umožňuje presné ovládanie uhlovej polohy, rýchlosti a zrýchlenia. Je to ako by ste boli za volantom svojho vozidla. Aj ty ovládaš presne rýchlosť vozidla a smer.

Profesionálne servo SG90 veža je jeden z najlacnejších servomotorov, ktoré môžete nájsť na trhu. Dokonca je to lacné. Nepokúšajte sa otáčať servomotor ručne, pretože by mohlo dôjsť k poškodeniu motora.

Zapojenie servomotora:



Ovládanie servomotora

Nižšie nájdete náčrt Arduino, ktorý riadi servo - smer, pozíciu motora a rýchlosť serva SG90. Obsahuje všetky potrebné funkcie pre riadenie serva SG90.

```
/*
Into Robotics
*/

#include <Servo.h>

int servoPin = 9;

Servo servo;

int servoAngle = 0;        // pozícia serva v uhloch

void setup()
{
  Serial.begin(9600);
  servo.attach(servoPin);
}
```

```
 void loop()
{
                          //ovládame smer servopohonu a polohu motora

   servo.write(45);       // otočí servo doľava na 50 stupňov
   delay(1000);           // čaká 1 sekundu
   servo.write(90);       // otočí servo späť na 90 stupňov (stredová poloha)
   delay(1000);           // čaká 1 sekundu
   servo.write(135);      // otočí servo doprava na 135 stupňov
   delay(1000);           // čaká 1 sekundu
   servo.write(90);       // otočí servo späť na 90 stupňov (stredová poloha)
   delay(1000);

//koniec ovládania serva a polohy motora

//ovládanie rýchlosti servopohonu

//Ak zmeníme hodnotu oneskorenia (z príkladu zmeny 50 na 10), zmení sa rýchlosť servopohonu

for(servoAngle = 0; servoAngle < 180; servoAngle++)
//posunie mikro servo z 0 stupňova na 180 stupňov
  {
    servo.write(servoAngle);
    delay(50);
  }

for(servoAngle = 180; servoAngle > 0; servoAngle--)
// teraz posunie mikro servo späť  z 0 stupňov na 180 stupňov
  {
    servo.write(servoAngle);
    delay(10);
  }
  //koniec kontroly polohy a rýchlosti servomotora
}
```

Príklad č. 10: SERVOMOTOR 2

```
#include <Servo.h>

Servo myservo;          // vytvorenie servoobjektu na ovládanie servomotora
                        // maximálne je možné vytvoriť 8 servoobjektov

int pos = 0;             // uloženie pozície servopohonu

void setup()
{
  myservo.attach(9);    // pripojí servo na kolík 9 k servoventilu
}


void loop()
{
  for(pos = 0; pos < 180; pos += 1)  // prechádza z 0 stupňov na 180 stupňov
  {                                  // v krokoch po 1 stupni
```

```
   myservo.write(pos);              // Informujte servo, aby sa dostal do pozície v premennej pos
delay(15);                          // čaká 15 ms, aby servo dosiahlo pozíciu
  }
  for(pos = 180; pos>=1; pos-=1)    // prechádza zo 180 stupňov do 0 stupňov
  {
    myservo.write(pos);             // Informuje servo, aby sa dostalo do pozície v premennej pos
    delay(15);                      // čaká 15 ms, aby servo dosiahlo pozíciu
  }
}
```

Príklad č. 11: SERVOMOTOR 3

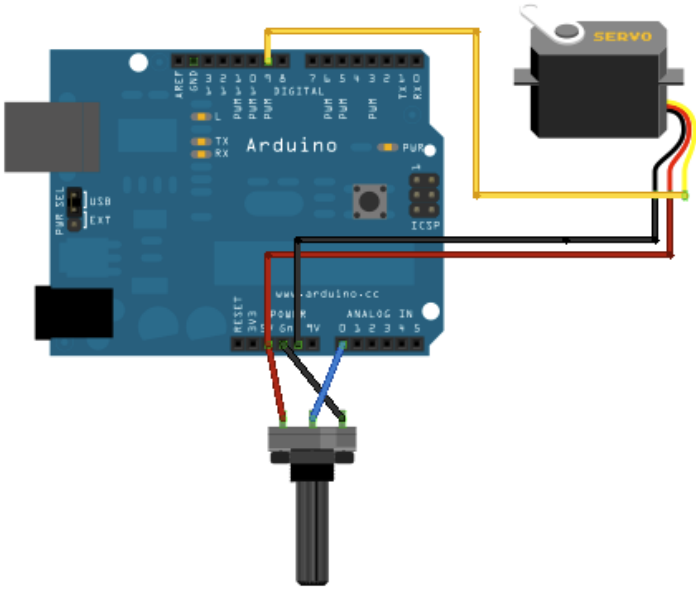
```
#include  <Servo.h>                 //Servo knižnica

Servo servo_test;                   //inicializujte servo objekt pre pripojené servo

int angle = 0;

void setup()
{
  servo_test.attach(9);             // pripojte signalizačný kolík servopohonu na pin9 Arduina
}
void loop()
{
for(angle = 0; angle < 180; angle += 1)   // príkaz na presun z 0 stupňov na 180 stupňov
  {
    servo_test.write(angle);              //príkaz otočiť servo do určeného uhla za 15ms
    delay(15);
  }
  delay(1000);

 for(angle = 180; angle>=1; angle-=5)     // príkaz na presun z 180 stupňov na 0 stupňov
  {
    servo_test.write(angle);              //   príkaz otočiť servo do určeného uhla za 15ms
    delay(15);
  }
    delay(1000);
  }
```

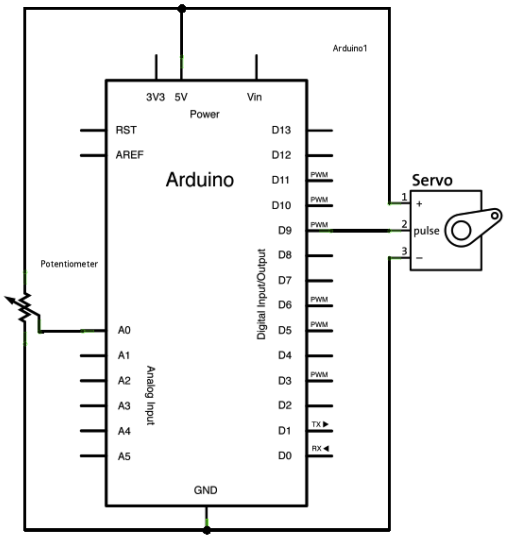
Alternatívne príklady:

1. Uprav program pre ovládanie servomotora s pohybom do 90° pomalým pohybom a späť rýchlym pohybom.
2. Uprav program pre ovládanie servomotora s rýchlosťou 1° za sekundu do uhla 45° a späť.
3. Uprav program a zapojenie pre ovládanie servomotora do 180°, pričom pohyb sa spustí zatlačením tlačidla.
4. Uprav program pre ovládanie servomotora pre pohyb do 180° a späť, pričom pohyb sa spustí zatlačením tlačidiel tam a späť.
5. Uprav program pre ovládanie a signalizáciu servomotora pomocou 2 LED a 2 tlačidiel. Pričom pohyb TAM sa spustí tlačidlom a signalizáciou červenej LED, pohyb bude do 180° a pohyb SPAŤ sa spustí tlačidlom a signalizáciou zelenej LED, pohyb bude do -180°.

Príklad č. 12: Ovládanie servomotora potenciometrom

Zapojenie servomotora:





Program:

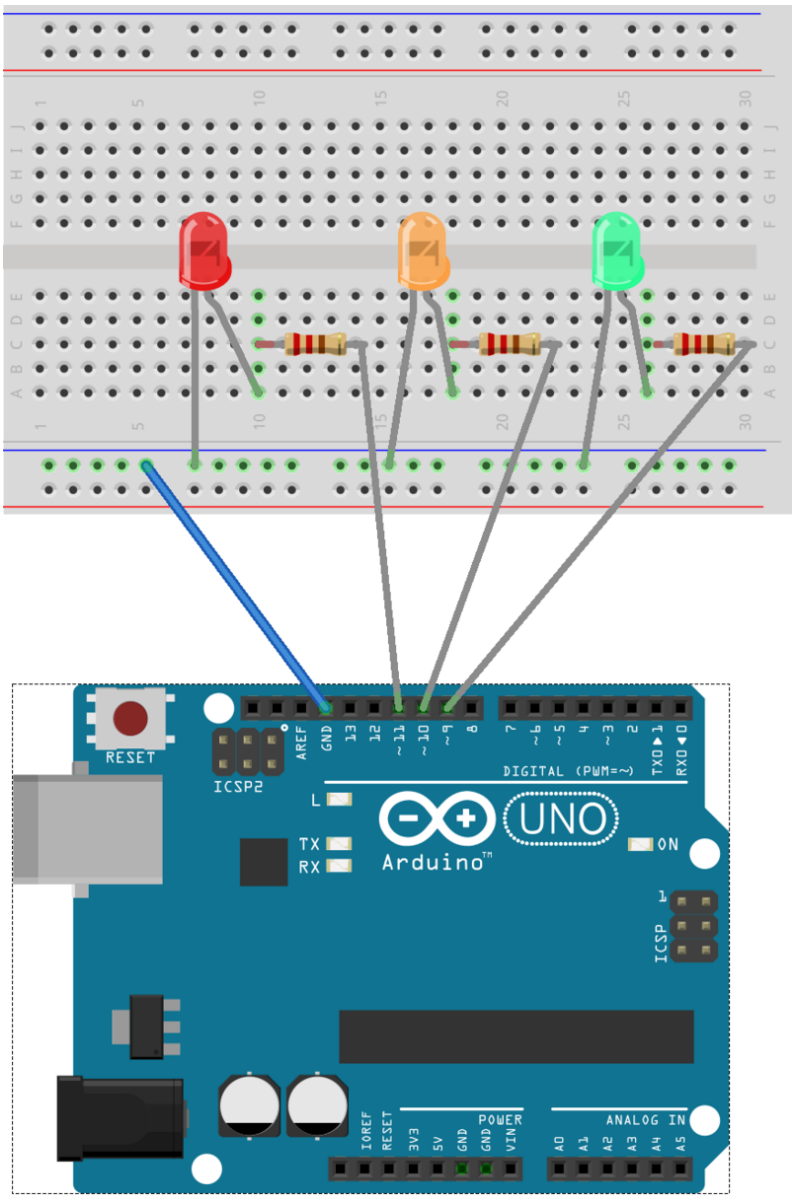
```
#include <Servo.h>

Servo myservo;                  // vytvorte servoobjekt na ovládanie servopohonu

int potpin = 0;                 // analogový kolík určený na pripojenie potenciometra
int val;                        // premenná na čítanie hodnoty z analógového kolíka

void setup() {
  myservo.attach(9);            // pripojí servo na kolík 9 k servoventilu
}

void loop() {
  val = analogRead(potpin);           // číta hodnotu potenciometra (medzi 0 a 1023)
  val = map(val, 0, 1023, 0, 180);    // upravme ho tak, aby sme ho používali so servom (hodnota
                                         medzi 0 a 180)
  myservo.write(val);                 // nastaví polohu servopohonu podľa stupnice
  delay(15);                          // čaká na servo, aby sa tam dostal
}
```

Alternatívne príklady:

1. Uprav program pre ovládanie servomotora s pohybom do 360°.
2. Uprav program pre ovládanie servomotora s rýchlosťou 1° za sekundu do uhla 90°.

Príklad č. 13: Program s LED

Zapojenie LED a rezistorov:



Rozpis materiálu:

- 3x farebné LED diody
- 3x rezistor 220Ω alebo 330Ω
- kontaktné pole
- Arduino
- Vodiče

Program:

```
//Nastavíme piny pripojených LED diod
int cervenaled = 9;

int oranzovaled = 10;

int zelenaled = 11;

void setup() {

//Nastavíme zapojené piny ako výstupné
```

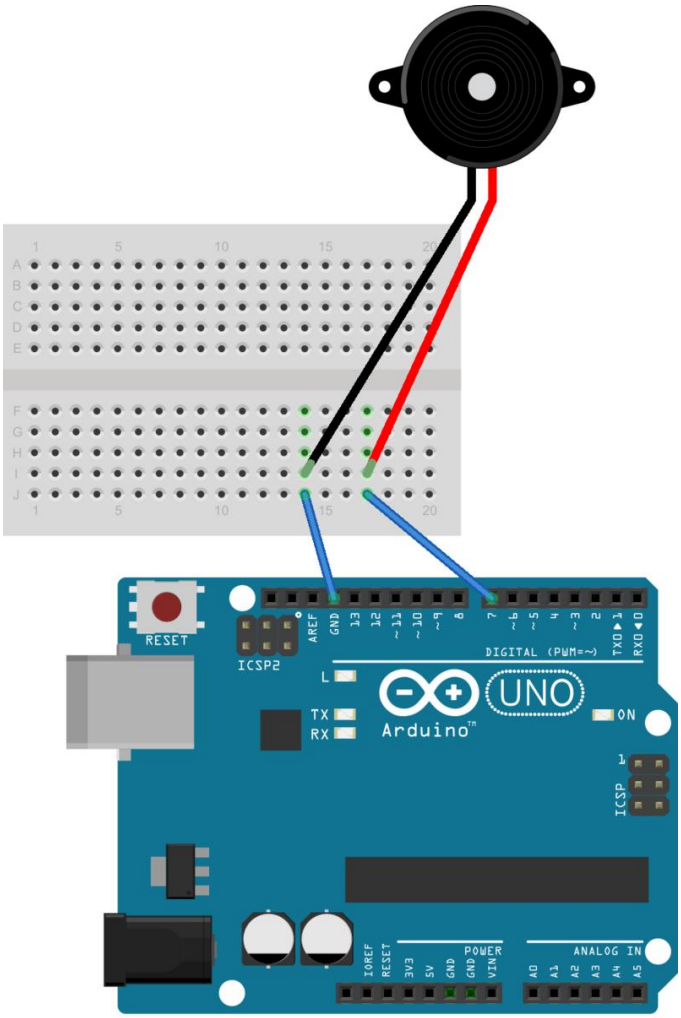
```
pinMode (cervenaled, OUTPUT);
pinMode (oranzovaled, OUTPUT);
pinMode (zelenaled, OUTPUT);
}
void loop() {
//Nastavíme v akom poradí a akú dobu budú LED diódy svietiť
digitalWrite (cervenaled, 1);
delay(9000);
digitalWrite (oranzovaled, 1);
delay (1000);
digitalWrite (cervenaled, 0);
digitalWrite (oranzovaled, 0);
digitalWrite (zelenaled, 1);
delay (9000);
digitalWrite (zelenaled, 0);
digitalWrite (oranzovaled, 1);
delay (3000);
digitalWrite (oranzovaled, 0);
}
```

Alternatívne príklady:

1. Uprav program aby diódy blikali súčasne s frekvenciou 1Hz.
2. Uprav program aby každá dióda blikala samostatne a súčasne svietila len 1 dióda.
3. Uprav program aby červená dióda blikala 2x rýchlejšie ako zelená a 2x pomalšie ako oranžová.
4. Uprav program aby diódy blikali Knight rider efektom.

Príklad č. 14: Melódia z piezoreproduktora

Zapojenie:



Príklad:

```
#include "pitches.h"
 //pridá k programu potrebný súbor, ktorý udáva hodnotu daného tónu noty, ktoré melódia obsahuje
int melodie[] = {
NOTE_C4, NOTE_G3,NOTE_G3, NOTE_A3, NOTE_G3,0, NOTE_B3, NOTE_C4};
// udává časovanie  nôt: 4 je nota štvrťová, 8 je nota osminová
int casovaniNot[] = {
4, 8, 8, 4,4,4,4,4 };
void setup() {
for (int Nota = 0; Nota < 8; Nota++) {
```

```
//výpočet dĺžky zvuku jednej  noty vypočítáme tak, že vydelíme 1000/noteDurations
//napr.1000/4 alebo 1000/8
int casovaniNoty = 1000/casovaniNot[Nota];
//prvým číslom v zátvorke určujeme digitálny pin pre výstup
tone(7, melodie[Nota],casovaniNoty);
//pre rozlišenie nôt nastavte pauzyMeziNotami * 1.30
int pauzyMeziNotami = casovaniNoty * 1.30;
delay(pauzyMeziNotami);
//zastaví melódiu
noTone(8);
}
}
void loop() {
//pokiaľ nechcete melódiu  opakovať, tak funkciu  loop nechajte prázdnu
}
```

Alternatívne príklady:

1. Uprav program aby piezogenerátor generoval nejakú známu zvučku podľa nôt.

Príklad č. 15: Užívateľsky definované funkcie

Definícia funkcie

Aby funkcia pracovala bez problémov, potrebuje mať dátový typ , názov a zátvorky.

```
dátový-typ názov-funkcie (priestor-pre-parametre)
{
        príkazy ...
}
```

U funkcií bez vstupných parametrov sa okrúhle zátvorky nechajú prázdne (ale musia tu byť).

```
void text ()
 {
        Serial.println ( "TEXT TEXT");
  }
```

S parametrami sa pracuje rovnako ako s premennými. Ak funkcia nejaké má, musíme ich nadefinovať. Definícia prebieha v okrúhlych zátvorkách. Ak má funkcia viac parametrov, oddeľujú sa čiarkami.

```
void sprava (char a [], char b [])
 {
        Serial.print (a);
        Serial.print ( '');
        Serial.println (b);
 }
```

Volanie funkcie

V tejto chvíli sa ale po spustení programu nič nestane. Funkcia je síce vytvorená, ale ešte sme ju nikde nezavolali (nepoužili). Nasledujúci kód vypíše po sériovej linke:
Ahoj Peter
TEXT TEXT

Program:

```
void setup ()
 {
        Serial.begin (9600);
        sprava ( "Ahoj", "Peter");
        text ();
}

void loop ()
{
}

void text ()
 {
        Serial.println ( "TEXT TEXT");
}

void sprava (char a [], char b [])
 {
        Serial.print (a);
        Serial.print (' ');
        Serial.println (b);
}
```

Funkcie, ktoré vracajú hodnotu

Pokiaľ má funkcia niečo vracať, musí mať iný dátový typ ako void. Pre vrátenie vybranej hodnoty sa používa príkaz return. Ak chceme vrátiť reťazec znakov, nepoužíva sa pole char [], ale dátový typ String. Tiež nie je možné jednoduchým spôsobom vrátiť poľe. Ostatné dátové typy sa používajú rovnako.

```
String slovo ()
{
        return "Ahoj";
}

// volanie
Serial.println (slovo ());   // Vypíše: Ahoj
```

Jednoduchá funkcia pre sčítanie dvoch čísel a vrátenie súčtu potom vyzerá takto:

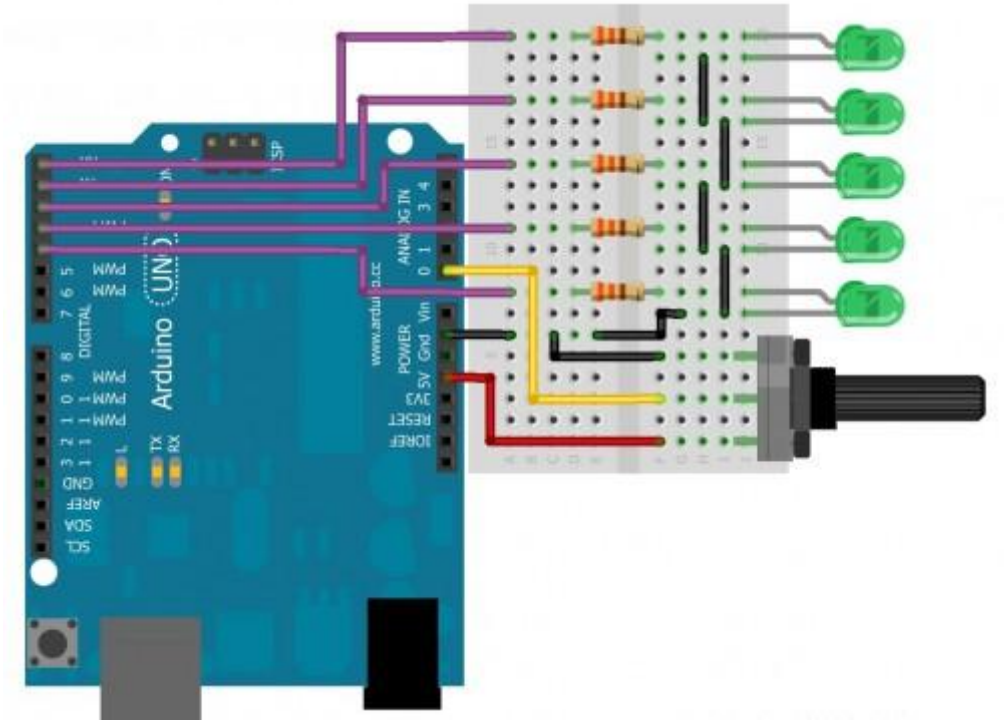
```
void setup ()
{
        Serial.begin (9600);
        Serial.println (scitaj (10,11));
}

void loop ()
 {
}

int scitaj (int a, int b)
{
        int sucet = a + b;
        return sucet;
}
```

Alternatívne príklady:

1. Uprav program aby násobil 2 číslice.
2. Uprav program aby sčitoval 3 číslice.
3. Uprav program aby našiel maximum z 3 číslic.
4. Uprav program aby našiel minimum z 5 číslic.
5. Uprav program aby vypočítal obsah štvorca.

Príklad č. 16: Čítanie potenciometra

Zapojenie do kontaktnej plochy:



Program:

```
byte led[] = {1,2,3,4,5};
//pole s pinmi  pripojených LED diód
byte pot = A0;  //pin potenciometra pripojený na analógový vstup A0

int val;

void setup() {
        pinMode(led[0], OUTPUT);
        pinMode(led[1], OUTPUT);
        pinMode(led[2], OUTPUT);
        pinMode(led[3], OUTPUT);
        pinMode(led[4], OUTPUT);
}

void loop() {
        val = analogRead(pot);

        if(val > 800){
                digitalWrite(led[0],HIGH);
        }
        else if(val > 600){
                digitalWrite(led[1],HIGH);
        }
        else if(val > 400){
                digitalWrite(led[2],HIGH);
        }
        else if(val > 200){
                digitalWrite(led[3],HIGH);
        }
```

```
        else{
                digitalWrite(led[4],HIGH);
        }

        delay(250);
        digitalWrite(led[0],LOW);
        digitalWrite(led[1],LOW);
        digitalWrite(led[2],LOW);
        digitalWrite(led[3],LOW);
        digitalWrite(led[4],LOW);
}
```

Alternatívne príklady:

1. Uprav program pre čítanie potenciometra zo 7 led.
2. Uprav program pre čítanie potenciometra tak, aby led blikali.
3. Pripojte zvukový generátor, ktorý bude generovať zvuk, podľa toho, ktorá dióda svieti.
4. Doplňte do programu kód na informáciu z potenciometra aby zobrazil na sériovom monitore.

Príklad č. 17: Blikač riadený potenciometrom

Zapojenie do kontaktnej plochy:

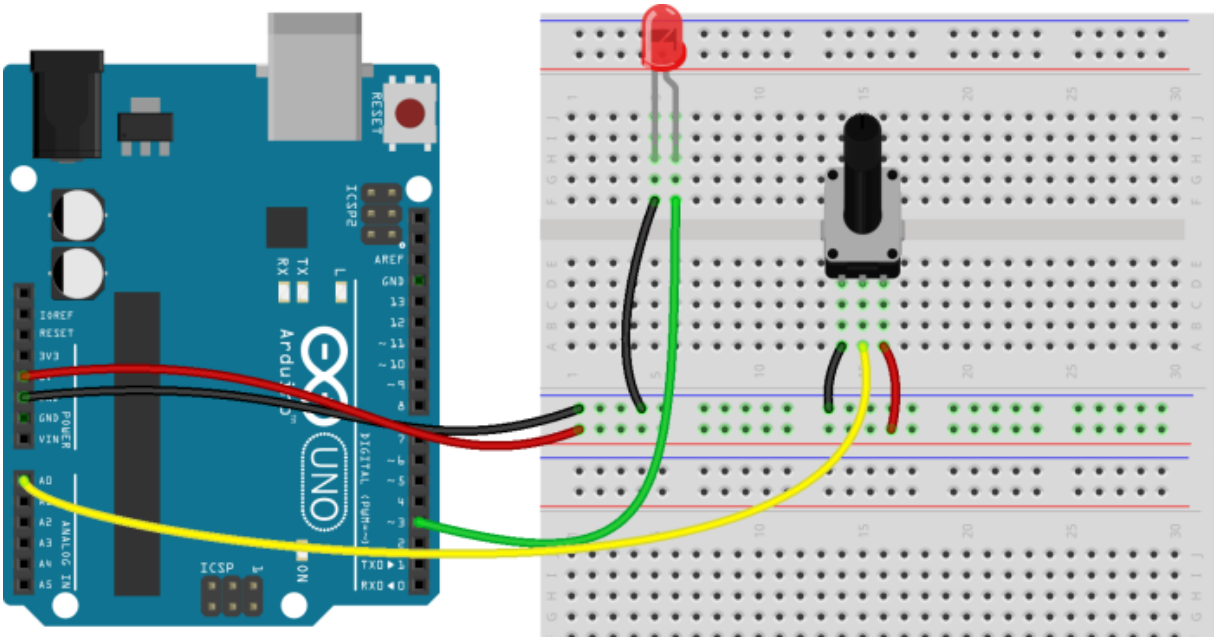
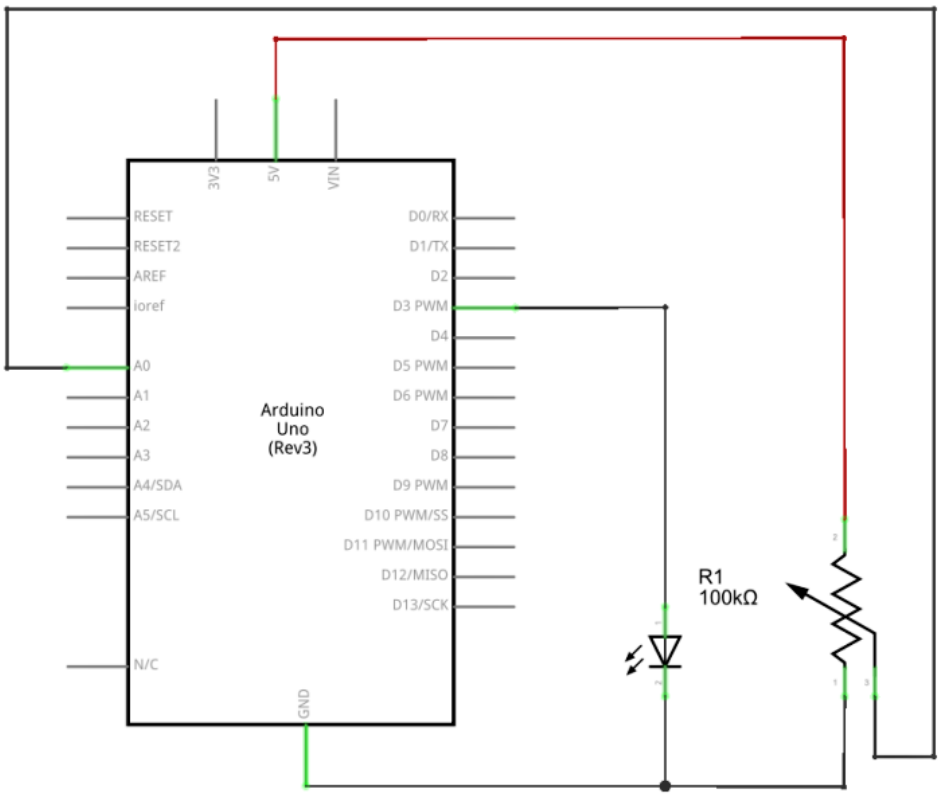


Schéma zapojenia:



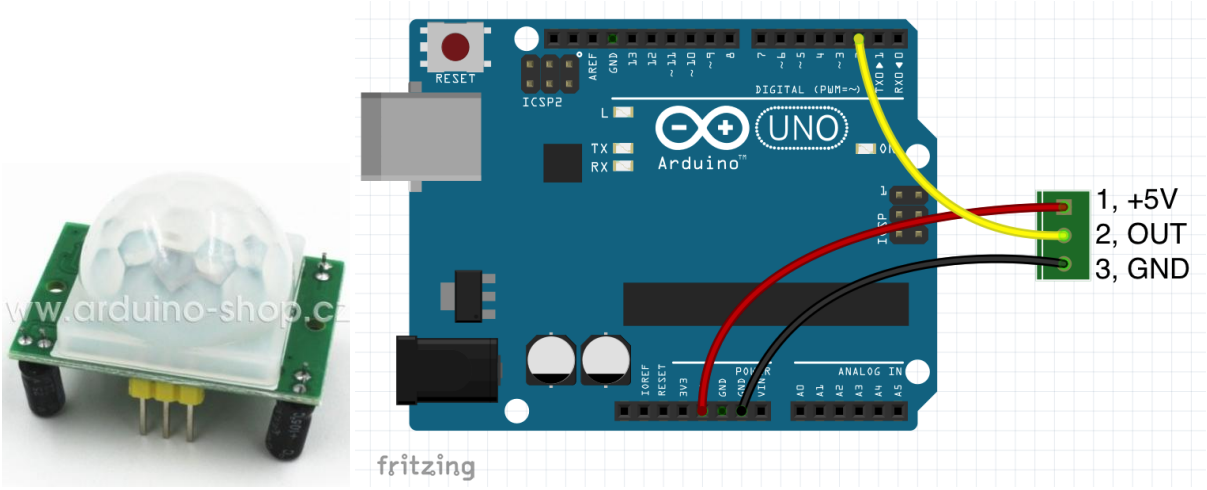
Príklad:

```
// Potenciometer

int potPin = A0;          // číslo pinu pripojeného potenciometra
int ledPin = 3;           // číslo pinu pripojenej LED diody

int potProm = 0;          // premenná pre analógovú hodnotu potenciometra

void setup() {
  // nastavenie LED ako výstup
  pinMode(ledPin, OUTPUT);
  // nastavenie potenciometra ako vstup
  pinMode(potPin, INPUT);
}

void loop() {
  // načítanie analógovej hodnoty senzora a uloženie do premennej
  potProm = analogRead(potPin);
  // zapne LED diódu
  digitalWrite(ledPin, HIGH);
  // zastaví program na čas zodpovedajúci analogovej hodnote potenciometra
  delay(potProm);
  // vypne LED diódu
  digitalWrite(ledPin, LOW);
  // zastaví program na čas zodpovedajúci analogovej hodnote potenciometra
  delay(potProm);
}
```

Alternatívne príklady:

1. Doplňte realizáciu s dvoma LED, z ktorých každá bude ovládaná iným potenciometrom.
2. Doplňte realizáciu s dvoma LED, ktoré budú ovládané jedným potenciometrom.
3. Doplňte realizáciu s dvoma LED, ktoré budú fungovať ako blikač.

Príklad č. 18: Pohybové čidlo HC-SR501



Popis programu:

Ukážkový kód je veľmi jednoduchý. Na začiatku obsahuje nastavenie čísla prepojovacieho pinu a v podprograme setup nastavenie komunikácie cez sériovú linku, nastavenie pinu cidloPin ako vstupné a tiež nastavenie tohto pinu pre využitie prerušenie číslo 0. Prerušenie ide využiť u dosky Arduino Uno iba na pinoch 2 a 3, je teda dobré zvoliť jeden z týchto pinov. Prerušenie je v tomto prípade nastavené na detekciu nábežnej (či rastúcej) hrany, teda pri zvýšení napätia z 0 na 5V. V nekonečnej slučke loop sa v tejto ukážke nachádza informácia o čase od zapnutia Arduino dosky a je to len z demonštračných účelov. V bežnom použití by teda tu mohla prebiehať akákoľvek činnosť. Na konci programu sa nachádza dôležitý podprogram detekcie, ktorý sa zavolá vždy pri detekcii prerušenia. V tomto prípade sa vypíše opäť na sériovú linku informácia o detekcii pohybu pomocou pohybového senzora HC-SR501.

Program pre použitie čidla:

```
// Pohybové čidlo HC-SR501

// nastavenie čísla vstupného pinu pre detektor, červenú led, zelenú led, tónový generátor

const int cidloPin = 2;

const int redLed = 12;

const int greenLed = 11;

const int buzzer = 10;

void setup() {

// komunikácia cez sériovú linku rýchlosťou 9600 baud

  Serial.begin(9600);

// inicializacia digitálneho vstupu

  pinMode(cidloPin, INPUT);

  pinMode(redLed, OUTPUT);
```

```
  pinMode(greenLed, OUTPUT);
  pinMode(buzzer, OUTPUT);
  // nastavenie prerušenia na pin 2 (int0)
  // pri rastúcej hrane (logO->log1) se vykoná program prerus
  attachInterrupt(0, detekce, RISING);
}

void loop() {
  // pre ukážku se každých 5 sekúnd vytlačí
  // správa o počte sekund od zapnutia Arduina
  Serial.print("Cas od zapnutia: ");
  Serial.print(millis()/1000);
  Serial.println(" sekund.");
  digitalWrite(greenLed, HIGH);
  digitalWrite(redLed, LOW);
  delay(5000);
}
void detekce() {
  // pokiaľ je aktivovaný digitálny  vstup,
  // vypiš informáciu po sériovej linke, zasvieť diódu, spusti tón
  Serial.println("Detekcia pohybu pomocou HC-SR501!");
  digitalWrite(redLed, HIGH);
  digitalWrite(greenLed, LOW);
  tone(buzzer, 1000, 500);
}
```

Po nahratí programu do Arduino dosky a detekcii pohybu v okolí čidla dostaneme napríklad tento výpis na sériovom monitore:

Cas od zapnutia: 30 sekund.

Detekcia pohybu pomocou HC-SR501!

Cas od zapnutia: 35 sekund.

Cas od zapnutia: 40 sekund.

Cas od zapnutia: 45 sekund.

Cas od zapnutia: 50 sekund.

Cas od zapnutia: 55 sekund.

Cas od zapnutia: 60 sekund.

Cas od zapnutia: 65 sekund.

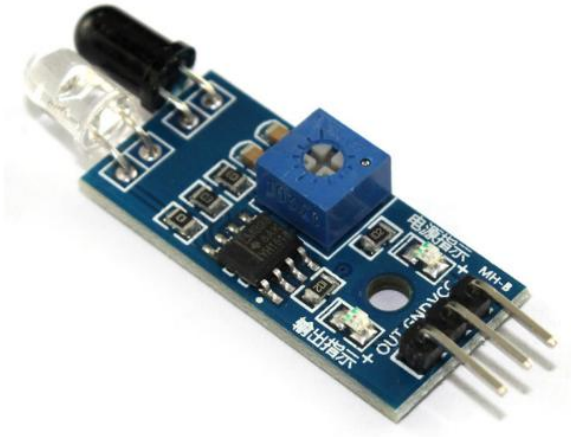
Detekcia pohybu pomocou HC-SR501!

Cas od zapnutia: 70 sekund.

Alternatívne príklady:

1. Uprav program aby vypisoval informáciu „Žiadna detekcia pohybu“, každých 10 sekúnd, pri detekcii aby vypísal „Čidlo zachytilo pohyb“.
2. Uprav program aby pri detekcii červená LED blikala a v kľude zelená LED blikala.
3. Pripoj servomotor, ktorý sa otočí o 90° pri detekcii pohybu – vytvorí sa závora.
4. Uprav program aby sa pri detekcii spustil tónový generátor s nejakou zvučkou.

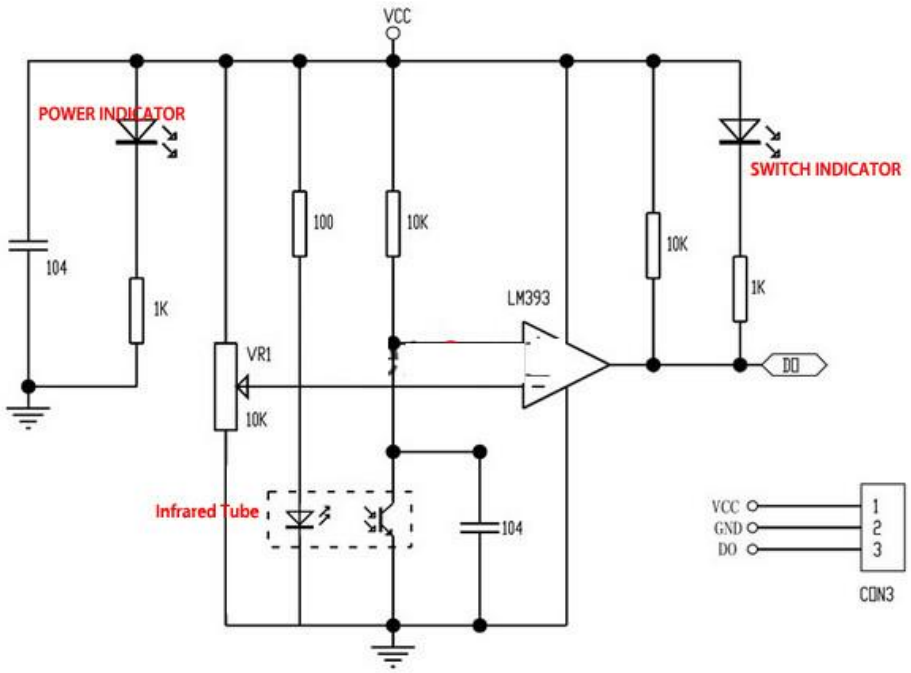
Príklad č. 19: Infračervené čidlo prekážky



Popis súčiastky:

Svetelný senzorový modul je adaptabilný k životnému prostrediu, má dvojicu infračervených rúrok, vysielaciu a prijímaciu. Vysielacia rúrka vypúšťa určitú infračervenú frekvenciu, pri detekcii v smere rúrky sa stretáva s prekážkami (odrážajúci povrch), čo sa odráža do infračervenej prijímacej rúrky, po spracovaní komparátor vyhodnotí obvod, zelená kontrolka sa rozsvieti pri zachytení odrazu, zároveň sa aktivuje výstupný signál do výstupného digitálneho signálu (úroveň signálu nízka), môže sa prostredníctvom otočného gombíka potenciometra nastaviť detekčná vzdialenosť, účinná vzdialenosť má rozsah 2 ~ 30 cm, pracovné napätie je 3,3V alebo 5V. Rozsah detekcie senzora môže byť pomocou potenciometra na nastavenie a má malú interferenciu, jednoduchá montáž, jednoduché použitie, atď, môže byť široko používaný vo vyhýbaní sa robotov prekážkam, vyhýbanie sa prekážkam pre vozidlá a black and white sledovanie čiary a tak pri iných mnohých príležitostiach.

Schéma vnútorného zapojenia:



Parametre:

1. keď modul detekuje prekážku pred návestidlom, zelená kontrolka na úrovni spojov sa rozsvieti a zároveň odošle signál nízkej hladiny OUT na výstup, detekčný modul detekuje od 2 ~ 30 cm, detekčný uhol má rozsah do 35 °, detekčnú vzdialenosť možno nastaviť pomocou potenciometra, je nastaviteľný v smere hodinových ručičiek, detekčná vzdialenosť sa v tomto smere zväčšuje; otáčaním potenciometra proti smeru hodinových ručičiek sa detekčná vzdialenosť zníži.

2. dva senzory, aktívny - infračervený, detekčný - odraz, teda odrazivosť a tvar terča je kľúčom oblasti detekcie. Biely a čierny detekčný rozsah, minimum - maximum; Malá oblasť na vzdialenosť objektu, veľký odstup.

3. Modul snímača výstupný port OUT môže byť priamo pripojený k mikrokontroléru, IO port, môže tiež riadiť 5V relé priamo; Režim pripojenia: VCC - VCC; GND - GND; OUT - IO

4. komparátor používa LM393, pracuje v stabilnom režime;

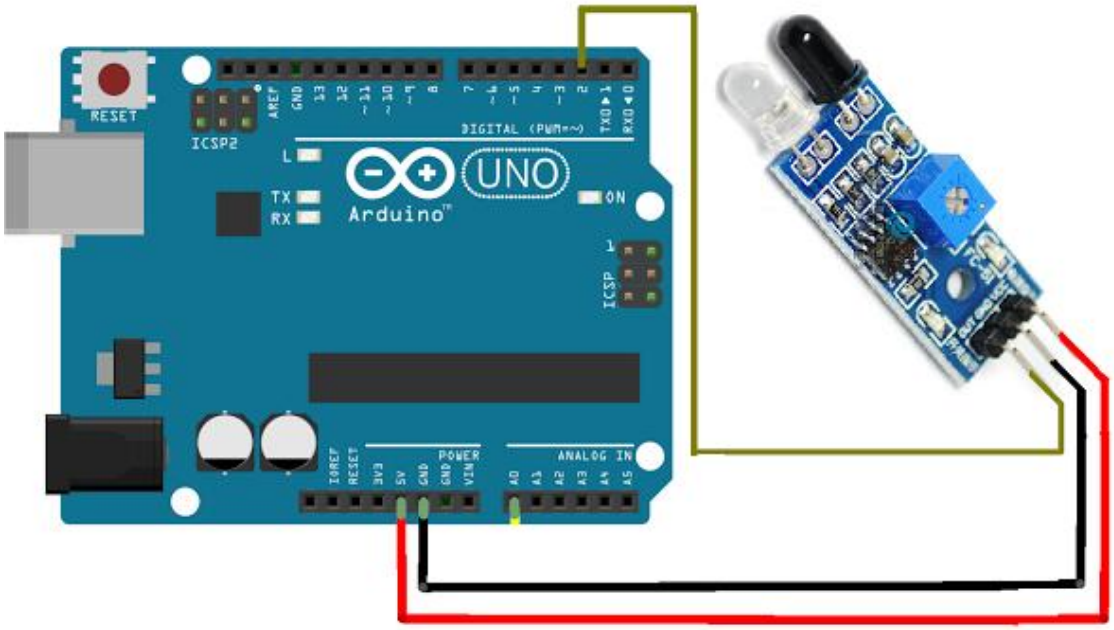
5. môžete využiť na 3 až 5V DC napájanie na napájací modul. Pri zapnutí, kontrolka napájania svieti červená;

6. 3 mm otvory na skrutky, ľahká pevná inštalácia;

7. obvod veľkosť dosky: 3.2 cm * 1.4 cm

8. Každý modul je dodávaný s potenciometrom pre nastavenie prahového napätia.

Schéma zapojenia s Arduinom:



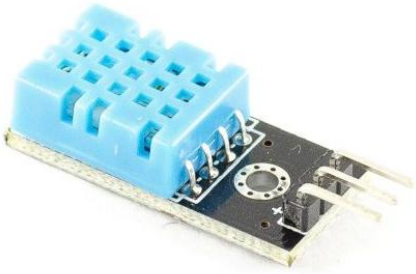
Príklad na detekciu prekážky:

```
/*
  IR Proximity Sensor interface code
  Turns on an LED on when obstacle is detected, else off.
  blog.circuits4you.com 2016
 */

const int ProxSensor=2;

void setup() {
  // initialize the digital pin as an output.
  // Pin 13 has an LED connected on most Arduino boards:
  pinMode(13, OUTPUT);
  //Pin 2 is connected to the output of proximity sensor
  pinMode(ProxSensor,INPUT);
}

void loop() {
  if(digitalRead(ProxSensor)==HIGH)      //Check the sensor output
  {
    digitalWrite(13, HIGH);   // set the LED on
  }
  else
  {
    digitalWrite(13, LOW);    // set the LED off
  }
  delay(100);              // wait for a second
}
```

Alternatívne príklady:

1. Uprav program aby pri detekcii prekážky spustil servomotor o 90° - vytvorí závoru.
2. Uprav program aby pri detekcii prekážky spustil piezogenerátor.
3. Pripoj červenú LED a zelenú LED, zelená LED svieti keď čidlo nedetekuje prekážku, červená LED sa rozsvieti pri detekcii prekážky.

Príklad č. 20: Snímač teploty a vlhkosti



Toto je lacný senzor vlhkosti a teploty. Čo to znamená? Že teplotu meria v celých stupňoch a rovnako aj vlhkosť. Teraz si možno poviete, že je to pre vás málo presné. Ale netreba hneď všetko merať s maximálnou presnosťou. Baviť vás to bude iba chvíľu, keď sa budete hrať so senzorom a na praktické použitie úplne stačí poznať teplotu len takto hrubo. Však aj na ortuťových teplomeroch za oknom len veľmi ťažko rozoznáte jemnejšie delenie.

Popis obvodu:

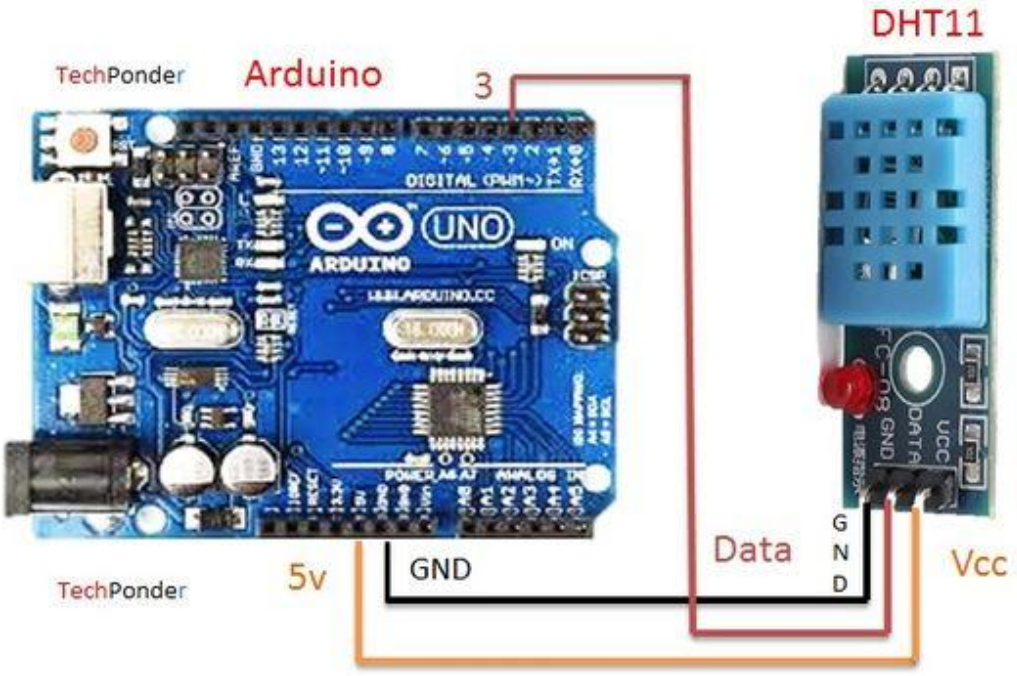
Senzor má štyri vývody:

- **VDD** - Napájanie. Napájanie je v rozsahu 3V - 5,5V, takže ho pripneme na 5V z Arduina.
- **DATA** - Prenos údajov. Údaje sa prenášajú špeciálnou knižnicou, ktorú je treba doinštalovať. Výrobca senzoru v dokumentácii píše, že sa má prepojiť zdvíhacím (pull-up) rezistorom 5k (alebo 10k) s VDD v prípade, že sú prepojovacie vodiče kratšie ako 20 metrov.
- Nepripojený
- **GND** - Sem sa pripojí zem.

Senzor meria:

- vlhkosť v rozpätí 20-90% pri teplote 25°C
- teplotu v rozpätí 0-50°C

Schéma zapojenia:



Stiahneme si knižnicu pre DH-11 , ktorá je už naprogramovaná tak prečo by sme to mali robiť ešte raz. Prácu treba efektívne využivať a kedže je to open-source netreba sa báť toho využívať.

Program:

```
#include <dht11.h>

dht11 DHT;

#define DHT11_PIN A1


void setup(){
  Serial.begin(9600);
  Serial.println("DHT TEST PROGRAM ");
  Serial.print("LIBRARY VERSION: ");
  Serial.println(DHT11LIB_VERSION);
  Serial.println();
  Serial.println("Type,\tstatus,\tHumidity (%),\tTemperature (C)");
}


void loop(){
  int chk;
  Serial.print("DHT11, \t");
  chk = DHT.read(DHT11_PIN);    // READ DATA
  switch (chk){
    case DHTLIB_OK:
                Serial.print("OK,\t");
                break;
    case DHTLIB_ERROR_CHECKSUM:
                Serial.print("Checksum error,\t");
                break;
    case DHTLIB_ERROR_TIMEOUT:
                Serial.print("Time out error,\t");
                break;
```

```
    default:
            Serial.print("Unknown error,\t");
            break;
  }
 // DISPLAT DATA
  Serial.print(DHT.humidity,1);
  Serial.print(",\t");
  Serial.println(DHT.temperature,1);

  delay(1000);
}
```

Popis programu:

V programe použijeme knižnicu DHT, ktorá v sebe zahŕňa komunikáciu so senzorom. Nemusíme teda riešiť komunikáciu a časovanie, vytvoríme len objekt dht, teda instanciu triedy DHT (viď riadok 5).

V setup metóde máme nastavenie sériového rozhrania a **nastavenie DHT** knižnice.

dht.setup(DHT_PIN);

Zavolaním setup metódy na objekte dht, máme nastavenú knižnicu DHT a môžeme s ňou pracovať ďalej. Metóda má jeden vstupný parameter, ktorým je číslo vývodu. V tomto príklade použijeme vývod číslo 8.

V loop metóde máme na začiatku oneskorenie. Toto oneskorenie sme získali z dht objektu. Je to minimálny interval, ktorým môžeme získavať údaje zo senzora. Ak by sme chceli nastaviť menší čas na dotazovanie sa o údaje, senzor by pravdepodobne neodpovedal, resp. nevracal by žiadne dáta.

Po uplynutí tohto času môžeme požiadať senzor o aktuálne údaje viď riadky 20 a 21. Trieda DHT obsahuje na to špeciálne funkcie:

```
float humidity = dht.getHumidity(); // nacitanie aktualnej vlhkosti vzduchu

float temperature = dht.getTemperature(); // nacitanie aktualnej teploty
```

Tieto údaje môžeme následne poslať cez sériové rozhranie do PC, alebo zobraziť na displeji.

V tomto príklade ich pošleme do PC sériovým portom. Spolu s tymito údajmi posielame aj teplotu vo fahrenheitoch (riadok 29) a stav (riadok 23) získaný metódou getStatusString().

Alternatívne príklady:

1. Uprav program aby výstupnú informáciu zobrazoval v slovenčine.
2. Uprav program a zapojenie tak, aby pri teplote nad 20°C zasvietila červená LED.
3. Uprav program a zapojenie tak, aby pri vlhkosti nad 50% zasvietila zelená LED.

Príklad č. 21: Detektor dymu a alkoholu MQ-4

V tomto príklade snímame výstupné napätie analógového snímača, keď dym dosiahne určitú úroveň, spustí sa zvuk bzučiaka a červená LED dióda sa rozsvieti.

Keď je výstupné napätie pod touto úrovňou, bude svietiť zelená LED dióda.

Čo je to MQ-4 dymový senzor?

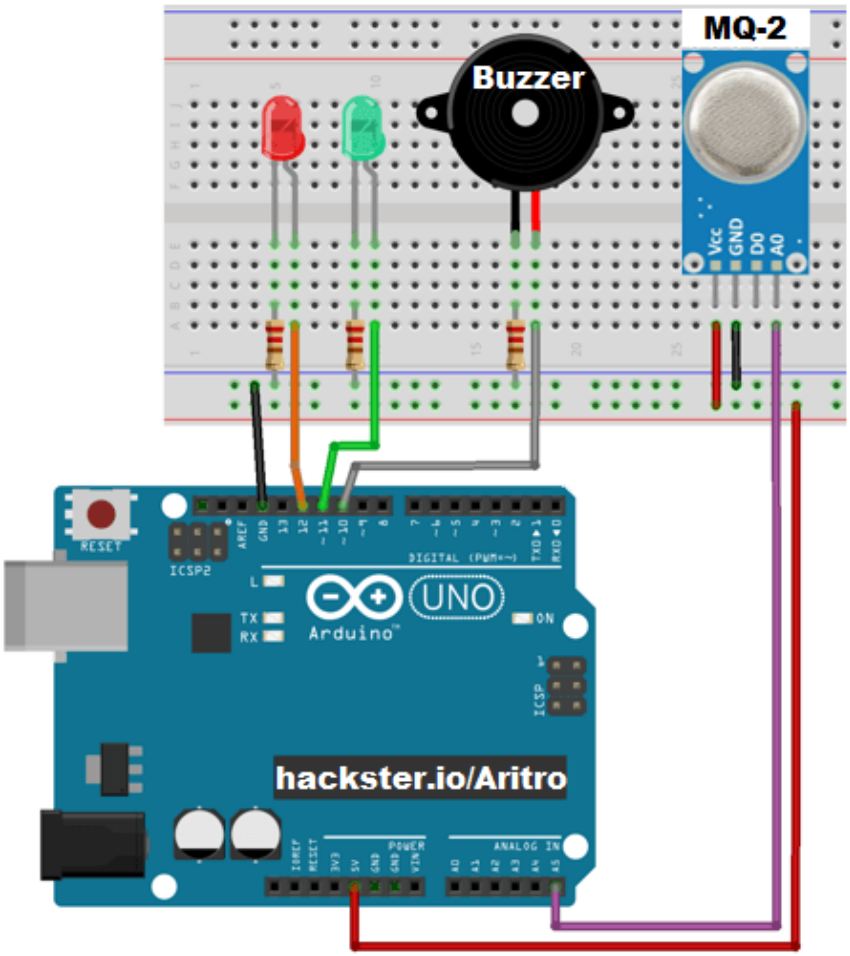
Snímač MQ-4 je citlivý na dym z cigariet a nasledujúce plyny:

- LPG
- bután
- propán
- metán
- alkohol
- vodík

Odpor snímača sa líši v závislosti na druhu plynu.

Senzor dymu má zabudovaný potenciometer, ktorý nám umožní nastaviť citlivosť snímača podľa toho, ako presne chceme zistiť plyn.

Zapojenie kontaktnej plochy:



Program:

```
int redLed = 12;
int greenLed = 11;
int buzzer = 10;
int smokeA0 = A5;

// Vaša prahová hodnota

int sensorThres = 400;

void setup() {
pinMode(redLed, OUTPUT);
pinMode(greenLed, OUTPUT);
pinMode(buzzer, OUTPUT);
pinMode(smokeA0, INPUT);
Serial.begin(9600);
}

void loop() {
int analogSensor = analogRead(smokeA0);
Serial.print("Pin A0: ");
Serial.println(analogSensor);

// Kontrola či sa dosiahne prahová hodnota

if (analogSensor > sensorThres)
{
digitalWrite(redLed, HIGH);
digitalWrite(greenLed, LOW);
tone(buzzer, 1000, 200);
}
else
{
digitalWrite(redLed, LOW);
digitalWrite(greenLed, HIGH);
noTone(buzzer);
}
delay(100);
}
```

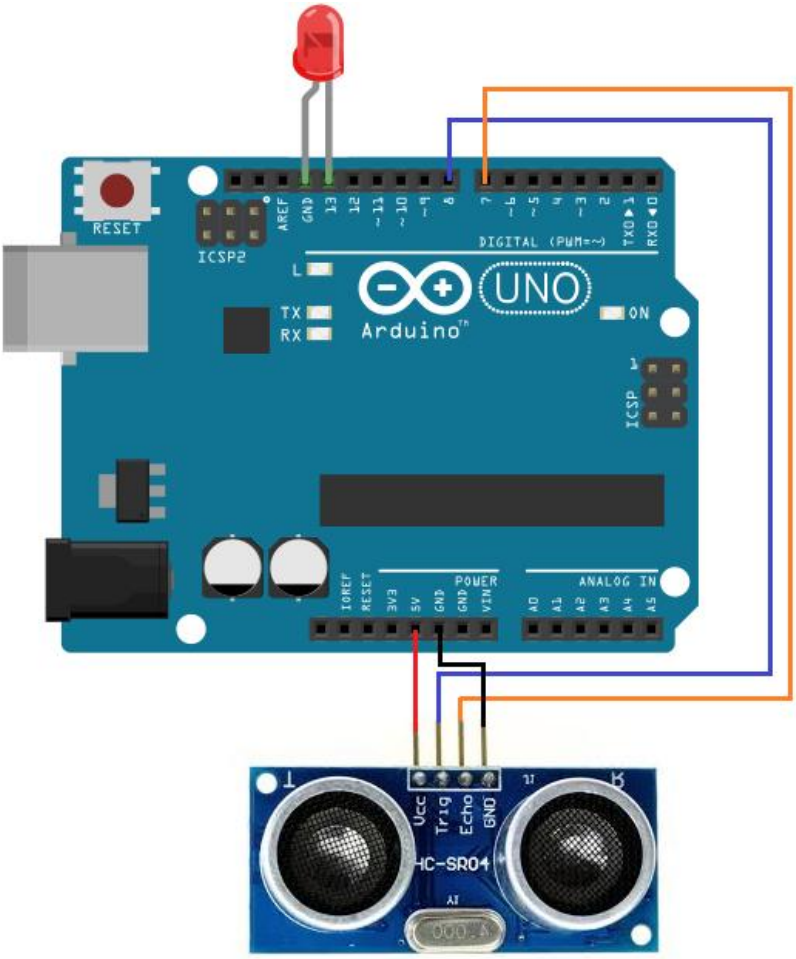
Alternatívne príklady:

1. Uprav program tak aby pri určitej prahovej hodnote sa rozblikala červená LED, a zvýšenej hodnote o 200 červená LED by svietila bez prerušenia.
2. Uprav program tak aby pri určitej prahovej hodnote bol tón z generátora 1000Hz a zvýšenej hodnote o 200 bol tón z generátora 2000Hz.

Príklad č. 22: Snímač vzdialenosti HC SR-04

Ultrazvukový senzor vzdialenosti je senzor, ktorý môže merať vzdialenosť od pevného alebo fyzického objektu. Ultrazvukový diaľkomer robí tak prostredníctvom zvukových vĺn. Vysiela vysokofrekvenčné zvukové vlny a potom čaká na odrazené zvukové vlny. Ak spätne počuje tieto zvukové vlny, potom to znamená, že zvukové vlny sa odrazili od fyzického objektu a vrátili sa k senzoru, čo znamená, že fyzický objekt je prítomný v prednej časti snímača. Ak tomu tak nie je , nepočuje tieto spätné zvukové vlny, to znamená, že nebol zistený žiadny fyzický objekt pred snímačom.

Zapojenie snímača s Arduinom:



Modul HC-SR04 nepotrebuje žiadne ďalšie knižnice. Musíte ho len pripojiť k Arduino v nadväznosti na:

5V ----> Vcc
GND ----> GND
pin 7 ----> Trigo
pin 8 ----> Echo

Program:

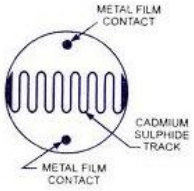
```
const int signalPin= 7;
const int triggerPin= 8;
const int LEDPin= 13;
const int LEDPin2= 12;

long signal, inches, centimeters;

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

void loop() {
pinMode (signalPin, INPUT);
pinMode (triggerPin, OUTPUT);
pinMode (LEDPin, OUTPUT);
pinMode (LEDPin2, OUTPUT);


// Posiela spúšťací impulz na spustenie senzora
digitalWrite(triggerPin, LOW);
delayMicroseconds(2);
digitalWrite(triggerPin, HIGH);
delayMicroseconds(10);
digitalWrite(triggerPin, LOW);
signal= pulseIn(signalPin, HIGH); //číta impulzy

// So snímačom HR-SRO4 trvá 148 mikrosekúnd na signál, aby odrážali späť k senzoru
// to je, ako vieme, je to palec

inches= signal/148;
centimeters= inches * 2.54; // 2.54cm je  1 palec

Serial.print(inches);
Serial.print("in, ");
Serial.print(centimeters);
Serial.print("cm");
Serial.println();
delay(500);

// Senzor HC-SR04 bude kontrolovať vzdialenosť k objektu každú pol sekundy

if (inches < 5) {
digitalWrite(LEDPin, HIGH);
}
```

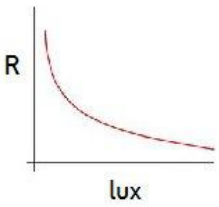
```
else
{
digitalWrite(LEDPin, LOW);
}


if (inches < 10) {
digitalWrite(LEDPin2, HIGH);
}
else
{
digitalWrite(LEDPin2, LOW);
}
delay(100);

}
```

Alternatívne príklady:

1. Uprav program tak aby signalizoval 5 vzdialeností do 50 cm, pomocou 5 LED, každých 10 cm.
2. Uprav program tak aby na vzdialenost 30cm upozornil zvukovým generátorom určitou frekvenciou a pri menšej vzdialenosti upozornil zvukovým generátorom inou frekvenciou.

Príklad č. 23: Optický snímač





Prvá vec, ktorú by sme mali vedieť, je, že LDR (svetlom závislý odpor, alebo foto odpor) je v podstate odpor, ktorý zmení odpor v závislosti na svetle. Viac svetla znamená menší odpor. Menej svetla znamená väčší odpor.

Zapojenie do kontaktnej plochy

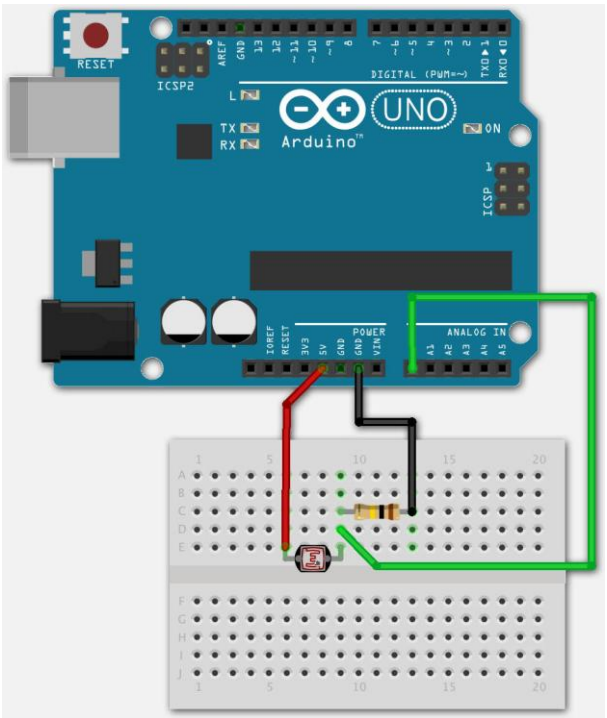
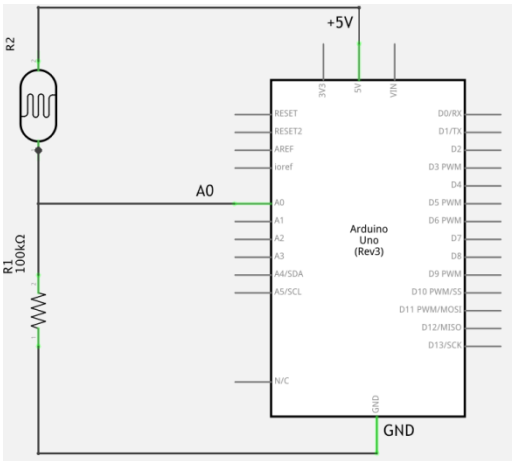


Schéma zapojenia



Program doplnený o zapojenie LED:

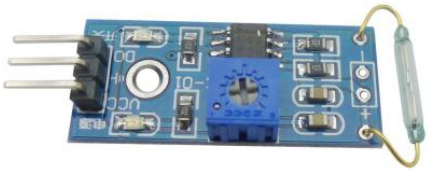
```
#define LDRpin A0 // pin where we connected the LDR and the resistor

int LDRValue = 0;     // result of reading the analog pin
int redLed = 12;
int greenLed = 11;

void setup() {
  Serial.begin(9600);                  // sets serial port for communication
```

```
  pinMode(redLed, OUTPUT);
  pinMode(greenLed, OUTPUT);

}

void loop() {
  LDRValue = analogRead(LDRpin); // read the value from the LDR
  Serial.println(LDRValue);
                                        // print the value to the serial port
  if (LDRValue > 1000)
{
digitalWrite(redLed, HIGH);
digitalWrite(greenLed, LOW);

}
else
{
digitalWrite(redLed, LOW);
digitalWrite(greenLed, HIGH);
}
   delay(100);                          // wait a little
}
```

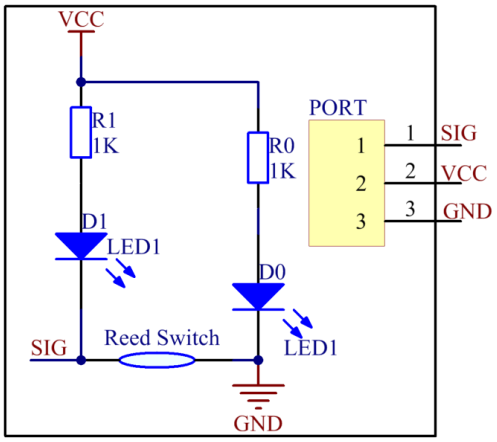
Alternatívne príklady:

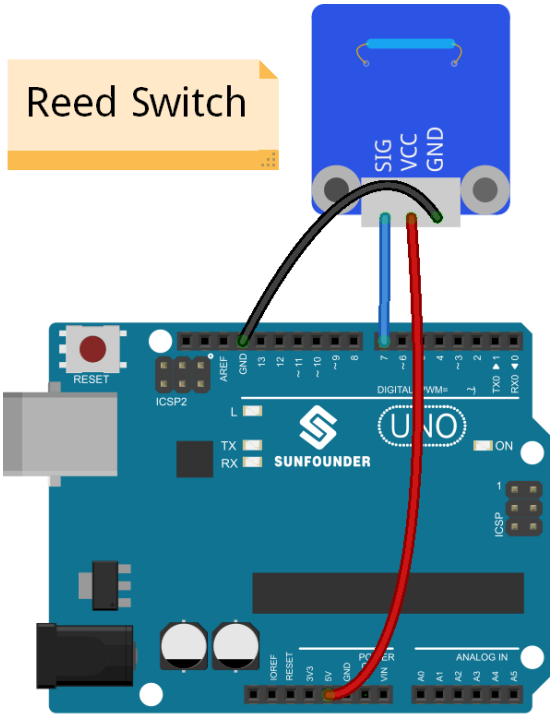
1. Uprav program, ktorý spustí zvukový generátor pri určitej hodnote osvetlenia.
2. Uprav program s LED, ktorej frekvencia blikania bude závisieť od osvetlenia.
3. Uprav realizáciu pomocou 10 LED, ktoré sa budú rozsvecovať podľa veľkosti osvetlenia, pri maximálnom osvetlení budú svietiť všetky, pri minimálnom osvetlení bude svietiť iba jedna LED.

Príklad č. 24: Magnetický spínač



Jazýčkový spínač je tiež senzor, ktorý sa používa na detekciu magnetického poľa. Reed prepínače sú často používané pre detekciu existencie magnetického poľa.





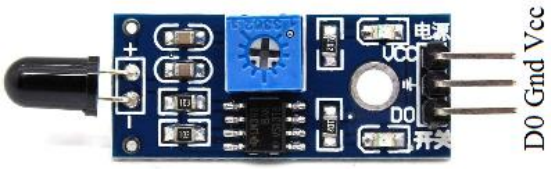
Program:

```
const int digitalInPin = 7;          // jazýčkový spínač pripojiť k pin7
const int ledPin = 13;               // pin13 vstavaný viedol
void setup ()
{
  pinMode (digitalInPin, INPUT); // nastaví digitalInPin ako vstup
  pinMode (ledPin, OUTPUT);    // nastaví ledPin ako výstup
}
void loop ()
{
  boolean stat = digitalRead (digitalInPin);        // načítanie hodnoty pin7 sa stat
  if (stat == HIGH)                                 // ak vysoko
  {
    digitalWrite (ledPin, nízka);                   // potom vypnúť viedla
  }
  else                                               // iný
  {
    digitalWrite (ledPin, HIGH);                    // zapnutie LED
  }
}
```

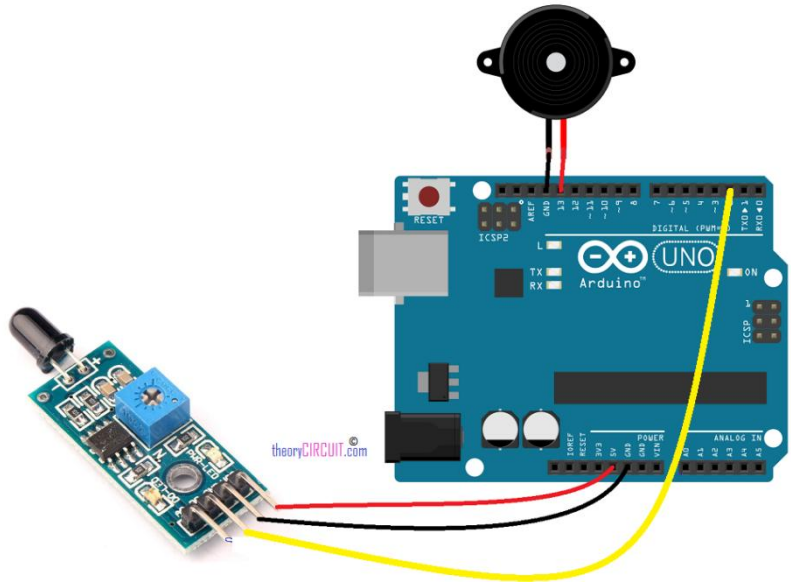
Alternatívny príklad:

1. Uprav program, kde sa spustí tónový generátor pri detekcii magnetického poľa.

Príklad č. 25: Detektor ohňa



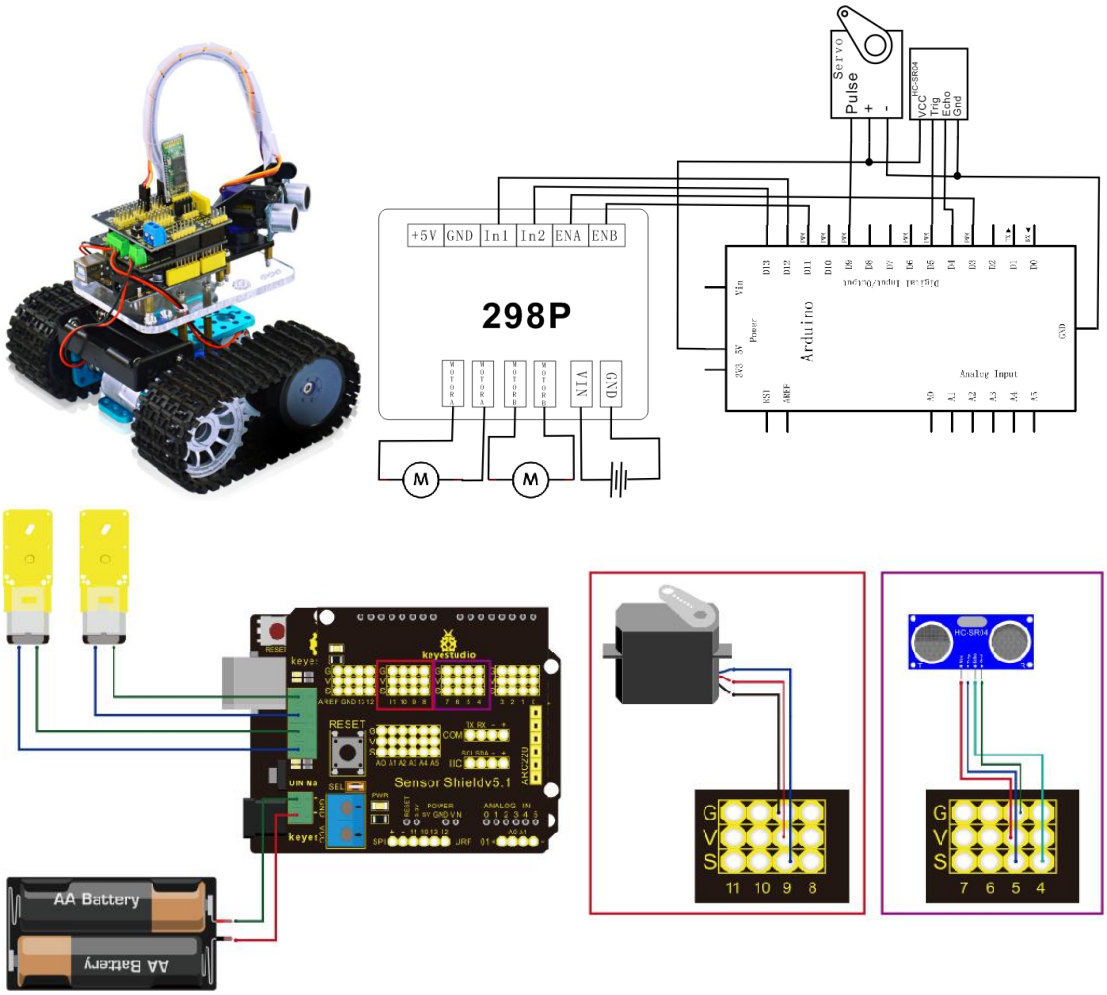
Snímače plameň dostupné na trhu, s dvoma kategóriami jeden je s tromi kolíkmi (D0, GND, Vcc) a druhý má štyri čapy (A0, d0, GND, VCC) obaja môžu byť ľahko prepojené s Arduinom a Arduino kompatibilnými doskami.



Program:

```
int Buzzer = 13; // Use buzzer for alert
int FlamePin = 2;  // This is for input pin
int Flame = LOW;  // HIGH when FLAME Exposed

void setup() {
  pinMode(Buzzer, OUTPUT);
  pinMode(FlamePin, INPUT);
  Serial.begin(9600);
}

void loop() {
  Flame = digitalRead(FlamePin);
  if (Flame== HIGH)
  {
    Serial.println("HIGH FLAME");
    digitalWrite(Buzzer, LOW);
  }
  else
  {
    Serial.println("No flame");
    digitalWrite(Buzzer, HIGH);
  }
}
```

Príklad č. 26: Robotický tank (http://www.keyestudio.cc/h-nd-127.html)




```
#include <Servo.h>
int pinLB = 12;     // definujem  pin 12 (lavy dozadu)
int pinLF = 3;      // definujem  pin 3  (lavy dopredu)
int pinRB = 13;     // definujem  pin 13 (pravy dozadu)
int pinRF = 11;     // definujem  pin 11 (pravy dopredu)
////////////////////////////////
int inputPin = 4;    // definujem pin pre senzor vzdialenosti - odpoveď
int outputPin =5;    // definujem pin pre senzor vzdialenosti  - vyslanie signálu

int Fspeedd = 0;      // vzdialenosť dopredu
int Rspeedd = 0;      // vzdialenosť vpravo
int Lspeedd = 0;      // vzdialenosť vľavo
int directionn = 0;     // prerušenie: dopredu=8 dozadu=2 vľavo=4 vpravo=6
Servo myservo;       // nastavenie servomotora so snímačom
int delay_time = 250; // doba usadzovania po pohybe servomotora

int Fgo = 8;        // pohyb dopredu
int Rgo = 6;        // pohyb vpravo
int Lgo = 4;        // pohyb vľavo
int Bgo = 2;        // pohyb dozadu
```

```
void setup()
{
 Serial.begin(9600);           // definujem piny pre motor
 pinMode(pinLB,OUTPUT); // pin 12
 pinMode(pinLF,OUTPUT); // pin 3 (PWM)
 pinMode(pinRB,OUTPUT); // pin 13
 pinMode(pinRF,OUTPUT); // pin 11 (PWM)
 pinMode(inputPin, INPUT);        // definujem vstupný pin pre senzor
 pinMode(outputPin, OUTPUT);  // definujem výstupný pin pre senzor
 myservo.attach(9);                  // definujem výstupný pin pre servomotor D9 (PWM)
}
void advance()                       // pohyb dopredu
  {
  digitalWrite(pinLB,LOW);       // pravé koleso sa posunie dopredu
  digitalWrite(pinRB, LOW);      // ľavé koleso sa posunie dopredu
  analogWrite(pinLF,255);
  analogWrite(pinRF,255);
  }
void stopp()                         // stop
  {
   digitalWrite(pinLB,HIGH);
   digitalWrite(pinRB,HIGH);
   analogWrite(pinLF,0);
   analogWrite(pinRF,0);
  }
void right()                         // odbočiť vpravo
  {

 digitalWrite(pinLB,HIGH);      // ľavé koleso sa pohybuje dopredu
 digitalWrite(pinRB,LOW);      // pravé koleso sa pohybuje dozadu
 analogWrite(pinLF, 255);
 analogWrite(pinRF,255);
  }
void left()                         // odbočiť doľava
  {
 digitalWrite(pinLB,LOW);     // l˘avé koleso sa pohybuje dozadu
 digitalWrite(pinRB,HIGH);    // pravé koleso sa pohybuje dopredu
 analogWrite(pinLF, 255);
 analogWrite(pinRF,255);
  }

void back()                          // pohyb dozadu
  {
   digitalWrite(pinLB,HIGH);  // ľavé koleso sa pohybuje dozadu
   digitalWrite(pinRB,HIGH);  // pravé koleso sa pohybuje dozadu
   analogWrite(pinLF,255);
   analogWrite(pinRF,255);
  }
void detection()                     // miera troch uhlov (0.90.179)
  {
    int delay_time = 250;        // stabilizačný čas pre servomotor po premiestnení dozadu
    ask_pin_F();                  // čítaj vzdialenosť dopredu
   if(Fspeedd < 10)               // ak je vzdialenosť dopredu <10 cm
```

```
    {
    stopp();                        // stop
    delay(100);
    back();                         // presuň sa dozadu po čas  0.2s
    delay(200);
    }
    if(Fspeedd < 25)                // ak vzdialenosť dopredu <25 cm
    {
      stopp();
      delay(100);                   // vyčistiť dáta
      ask_pin_L();                  // čítaj vzdialenosť vľavo
      delay(delay_time);            // stabilizačný čas pre servomotor
      ask_pin_R();                  // čítaj vzdialenosť vpravo
      delay(delay_time);          // stabilizačný čas pre servomotor

      if(Lspeedd > Rspeedd)   // ak je vzdialenosť vľavo> vzdialenosť vpravo
      {
        directionn = Lgo;         // pohyb doľava
      }

      if(Lspeedd <= Rspeedd)   // ak je vzdialenosť vľavo <= vzdialenosť vpravo
      {
       directionn = Rgo;            // pohyb doprava
      }

      if (Lspeedd < 10 && Rspeedd < 10)   // ak je vzdialenosť vľavo<10cm  a vzdialenosť
vpravo<10cm
      {
       directionn = Bgo;                      // pohyb dozadu
      }
    }
    else                                      // ak je vzdialenosť dopredu>25cm
    {
      directionn = Fgo;                       // pohyb dopredu
    }

  }
void ask_pin_F()                              // meraj vzdialenosť vpredu
  {
    myservo.write(90);
    digitalWrite(outputPin, LOW);   // ultrazvukový snímač prenáša nízku úroveň signálu 2µs
    delayMicroseconds(2);
    digitalWrite(outputPin, HIGH);  // ultrazvukový sen. Vysielajú vysokú úroveň signálu 10µs,
                                    //  najmenej 10µs
    delayMicroseconds(10);
    digitalWrite(outputPin, LOW);                 // udržujte vysielanie signálu nízkej úrovne
    float Fdistance = pulseIn(inputPin, HIGH);  // čítaj čas medzi tým
    Fdistance= Fdistance/5.8/10;                  // konvertuj čas na vzdialenosť (jednotka: cm)
    Fspeedd = Fdistance;                          // prečítaj vzdialenosť vpredu
  }
void ask_pin_L()                              //meranie vzdialenosti vľavo
{
    myservo.write(5);
```

```
    delay(delay_time);
    digitalWrite(outputPin, LOW);   // ultrazvukový snímač prenáša nízku úroveň signálu 2μs
    delayMicroseconds(2);
    digitalWrite(outputPin, HIGH);  //ultrazvukový senzor prenáša vysokú  úroveň signálu10μs,
                                    //najmenej 10μs
    delayMicroseconds(10);
    digitalWrite(outputPin, LOW);    // udržujte vysielanie signálu nízkej úrovne
    float Ldistance = pulseIn(inputPin, HIGH);  // čítaj čas medzitým
    Ldistance= Ldistance/5.8/10;       // konvertuj  čas na vzdialenosť (jednotka: cm)
    Lspeedd = Ldistance;                // prečítaj vzdialenosť vľavo
  }
void ask_pin_R()                        //  merajte vzdialenosť vpravo
  {
    myservo.write(177);
    delay(delay_time);
    digitalWrite(outputPin, LOW);      // ultrazvukový snímač prenáša nízku úroveň signálu 2μs
    delayMicroseconds(2);
    digitalWrite(outputPin, HIGH);  // ultrazvukový senzor prenáša vysokú  úroveň signálu10μs
    delayMicroseconds(10);
    digitalWrite(outputPin, LOW);    // udržujte vysielanie signálu nízkej úrovne
    float Rdistance = pulseIn(inputPin, HIGH);  // čítaj čas medzitým
    Rdistance= Rdistance/5.8/10;       // konvertuj čas na vzdialenosť  (jednotka: cm)
    Rspeedd = Rdistance;                // prečítaj vzdialenosť vpravo
  }

void loop()
 {
   myservo.write(90);                  // servomotor pripravený na ďalšie meranie
   detection();                        // zmerajte uhol a určite, ktorým smerom sa má pohybovať
  if(directionn == 2)                   // ak prerušenie = 2
  {
    back();
    delay(800);                        //  choď späť
    left() ;
    delay(200);              // pohybujte sa mierne doľava (aby ste zabránili uviaznutiu na konci)
  }
  if(directionn == 6)                   // ak prerušenie = 6
  {
    back();
    delay(100);
    right();
    delay(600);                        // choď vpravo
  }
  if(directionn == 4)                   // ak prerušenie = 4
  {
    back();
    delay(600);
    left();
    delay(600);                        // choď vľavo
  }
  if(directionn == 8)                   // ak prerušenie = 8
```

```
 {
  advance();                          // choď dopredu
  delay(100);
 }
 }
```

Program č. 27: Program pre pohyb dopredu-dozadu

```
#include <Servo.h>
int pinLB = 12;     // definuj  pin 12
int pinLF = 3;      // definuj  pin 3
int pinRB = 13;     // definuj  pin 13
int pinRF = 11;     // definuj  pin 11
////////////////////////////////
int inputPin = 4;    // definuj pin pre príjem senzora
int outputPin =5;    // definuj pin pre vysielanie senzora

int Fspeedd = 0;       // vzdialenosť vpredu
int Rspeedd = 0;      // vzdialenosť vpravo
int Lspeedd = 0;       // vzdialenosť vľavo
int directionn = 0;     // dopredu=8 dozadu=2 vľavo=4 vpravo=6
Servo myservo;        // nastav servomotor
int delay_time = 250; // doba usadzovania po pohybe servomotora B

int Fgo = 8;         // presun dopredu
int Rgo = 6;         // presun vpravo
int Lgo = 4;         // presun vľavo
int Bgo = 2;         // presun dozadu

void setup()
{
 Serial.begin(9600);           // definuj pin pre výstup pre motor
 pinMode(pinLB,OUTPUT); // pin 12
 pinMode(pinLF,OUTPUT); // pin 3 (PWM)
 pinMode(pinRB,OUTPUT); // pin 13
 pinMode(pinRF,OUTPUT); // pin 11 (PWM)
 pinMode(inputPin, INPUT);        // definuj vstupný pin pre senzor
 pinMode(outputPin, OUTPUT);  // definuj výstupný pin pre senzor
}

void back()                          // pohyb dozadu
  {
   digitalWrite(pinLB,HIGH);  // motor sa pohybuje doľava
   digitalWrite(pinRB,HIGH);  // motor sa pohybuje doprava
   analogWrite(pinLF,255);
   analogWrite(pinRF,255);
  }


void advance()                        // pohyb dopredu
  {
  digitalWrite(pinLB,LOW);       // pravé koleso sa posunie dopredu
  digitalWrite(pinRB, LOW);      // ľavé koleso sa posunie dopredu
```

```
  analogWrite(pinLF,255);
  analogWrite(pinRF,255);
  delay(800);                          //  pohyb späť
  }
void stopp()                           // stop
  {
   digitalWrite(pinLB,HIGH);
   digitalWrite(pinRB,HIGH);
   analogWrite(pinLF,0);
   analogWrite(pinRF,0);
  }
 void loop()
 {
   back();
   delay(8000);
   stopp();
   advance();
   delay(8000);
 }
```

Program č. 28: Program pre otáčanie namieste vľavo – vpravo

```
#include <Servo.h>
int pinLB = 12;     // definuj  pin 12
int pinLF = 3;      // definuj  pin 3
int pinRB = 13;     // definuj  pin 13
int pinRF = 11;     // definuj  pin 11
////////////////////////////////
int inputPin = 4;    // definuj pin pre príjem senzora
int outputPin =5;    // definuj pin pre vysielanie senzora

int Fspeedd = 0;       // vzdialenosť vpredu
int Rspeedd = 0;      // vzdialenosť vpravo
int Lspeedd = 0;       // vzdialenosť vľavo
int directionn = 0;     // prerušenie dopredu=8 dozadu=2 vľavo=4 vpravo=6
Servo myservo;        // nastav servomotor
int delay_time = 250; // doba usadzovania po pohybe servomotora B

int Fgo = 8;         // pohyb dopredu
int Rgo = 6;         // pohyb vpravo
int Lgo = 4;         // pohyb vľavo
int Bgo = 2;         // pohyb dozadu

void setup()
{
 Serial.begin(9600);           // definuj pin pre motor
 pinMode(pinLB,OUTPUT); // pin 12
 pinMode(pinLF,OUTPUT); // pin 3 (PWM)
 pinMode(pinRB,OUTPUT); // pin 13
 pinMode(pinRF,OUTPUT); // pin 11 (PWM)
 pinMode(inputPin, INPUT);        // definuj vstupný pin pre senzor
 pinMode(outputPin, OUTPUT);  // definuj výstupný pin pre senzor
```

```
}

void back()                        // pohyb dozadu
  {
   digitalWrite(pinLB,HIGH);  // motor sa pohybuje vľavo
   digitalWrite(pinRB,HIGH);  // motor sa pohybuje vpravo
   analogWrite(pinLF,255);
   analogWrite(pinRF,255);
  }

void advance()                      // pohyb dopredu
  {
  digitalWrite(pinLB,LOW);       // pravé koleso sa posunie dopredu
  digitalWrite(pinRB, LOW);      // ľavé koleso sa posunie dopredu
  analogWrite(pinLF,255);
  analogWrite(pinRF,255);
  delay(800);                         //  pohyb dozadu
  }
void stopp()                          // stop
  {
   digitalWrite(pinLB,HIGH);
   digitalWrite(pinRB,HIGH);
   analogWrite(pinLF,0);
   analogWrite(pinRF,0);
  }

void right()                          // pohyb vpravo
  {
 digitalWrite(pinLB,HIGH);         // ľavé koleso sa pohybuje dopredu
 digitalWrite(pinRB,LOW);          // pravé koleso sa pohybuje dozadu
 analogWrite(pinLF, 255);
 analogWrite(pinRF,255);
  }

void left()                           // pohyb vľavo
  {
 digitalWrite(pinLB,LOW);          // ľavé koleso sa pohybuje dozadu
 digitalWrite(pinRB,HIGH);         // pravé koleso sa pohybuje dopredu
 analogWrite(pinLF, 255);
 analogWrite(pinRF,255);
  }

 void loop()
 {
   left();
   delay(8000);
   stopp();
   right();
   delay(8000);
 }
```

Program č. 29: Pohyb do štvorca podľa programu

```
#include <Servo.h>
int pinLB = 12;     // definuj pin 12
int pinLF = 3;      // definuj pin 3
int pinRB = 13;    // definuj  pin 13
int pinRF = 11;    // definuj  pin 11
////////////////////////////////
int inputPin = 4;     // definuj pin pre príjem senzora
int outputPin =5;    // definuj pin pre vysielanie senzora

int Fspeedd = 0;       // vzdialenosť vpredu
int Rspeedd = 0;      // vzdialenosť vpravo
int Lspeedd = 0;       // vzdialenosť vľavo
int directionn = 0;     // prerušenie vpredu=8 vzadu=2 vľavo=4 vpravo=6
Servo myservo;        // nastav servomotor
int delay_time = 250; // doba usadzovania po pohybe servomotora B

int Fgo = 8;         // pohyb dopredu
int Rgo = 6;         // pohyb vpravo
int Lgo = 4;         // pohyb vľavo
int Bgo = 2;         // pohyb dozadu

void setup()
{
 Serial.begin(9600);           // definuj výstupný pin pre motor
 pinMode(pinLB,OUTPUT); // pin 12
 pinMode(pinLF,OUTPUT); // pin 3 (PWM)
 pinMode(pinRB,OUTPUT); // pin 13
 pinMode(pinRF,OUTPUT); // pin 11 (PWM)
 pinMode(inputPin, INPUT);        // definuj pin pre vstup do senzora
 pinMode(outputPin, OUTPUT);  // definuj výstup pre výstup zo senzora
}

void back()                          // pohyb dozadu
  {
   digitalWrite(pinLB,HIGH);  // motor sa pohybuje doľava
   digitalWrite(pinRB,HIGH);  // motor sa pohybuje doprava
   analogWrite(pinLF,255);
   analogWrite(pinRF,255);
  }

void advance()                       // pohyb dopredu
  {
  digitalWrite(pinLB,LOW);       // pravé koleso sa posunie dopredu
  digitalWrite(pinRB, LOW);      // ľavé koleso sa posunie dopredu
  analogWrite(pinLF,255);
  analogWrite(pinRF,255);
  delay(800);                          // pohyb dozadu
  }
void stopp()                           // stop
  {
```

```
   digitalWrite(pinLB,HIGH);
   digitalWrite(pinRB,HIGH);
   analogWrite(pinLF,0);
   analogWrite(pinRF,0);
  }

void right()                          // pohyb vpravo
  {
 digitalWrite(pinLB,HIGH);      // ľavé koleso dopredu
 digitalWrite(pinRB,LOW);      //  pravé koleso dozadu
 analogWrite(pinLF, 255);
 analogWrite(pinRF,255);
  }

void left()                          // pohyb vľavo
  {
 digitalWrite(pinLB,LOW);     // ľavé koleso dozadu
 digitalWrite(pinRB,HIGH);    // pravé koleso dopredu
 analogWrite(pinLF, 255);
 analogWrite(pinRF,255);
  }

 void loop()
 {
advance();
delay(1000);
stopp();
delay(1000);
left();
delay(1000);
stopp();
delay(1000);
advance();
delay(1000);
stopp();
delay(1000);
left();
delay(1000);
stopp();
delay(1000);
advance();
delay(1000);
stopp();
delay(1000);
left();
delay(1000);
advance();
delay(1000);
stopp();
delay(1000);
left();
delay(1000);
  }
```

Program č. 30: Program pre ovládanie pomocou Bloothoot

```
int pinLB = 12;     // definuj pin 12
int pinLF = 3;      // definuj pin 3
int pinRB = 13;     // definuj pin 13
int pinRF = 11;     // definuj pin 11
int val;
void setup()
{
 Serial.begin(9600);     // definuj výstupné piny pre motor
 pinMode(pinLB,OUTPUT); // pin 12
 pinMode(pinLF,OUTPUT); // pin 3 (PWM)
 pinMode(pinRB,OUTPUT); // pin 13
 pinMode(pinRF,OUTPUT); // pin 11 (PWM)
}
void advance()     // pohyb dopredu
   {
   digitalWrite(pinLB,LOW);    // pravé koleso dopredu
   digitalWrite(pinRB, LOW);  // ľavé koleso dopredu
   analogWrite(pinLF,255);
   analogWrite(pinRF,255);
   }
void stopp()        // stop
   {
    digitalWrite(pinLB,HIGH);
    digitalWrite(pinRB,HIGH);
    analogWrite(pinLF,0);
    analogWrite(pinRF,0);
   }
void right()       // pohyb vpravo
   {
  digitalWrite(pinLB,HIGH);  // ľavé koleso dopredu
  digitalWrite(pinRB,LOW); // pravé koleso dozadu
  analogWrite(pinLF, 255);
  analogWrite(pinRF,255);
   }
void left()        // pohyb vľavo
   {
  digitalWrite(pinLB,LOW);  // ľavé koleso dozadu
  digitalWrite(pinRB,HIGH); // pravé koleso dopredu
  analogWrite(pinLF, 255);
  analogWrite(pinRF,255);
   }

void back()         // pohyb dozadu
   {
    digitalWrite(pinLB,HIGH);  // motor sa pohybuje doľava
    digitalWrite(pinRB,HIGH);  // motor sa pohybuje doprava
    analogWrite(pinLF,255);
    analogWrite(pinRF,255);
   }
void loop()
```

```
  {
  val=Serial.read();
  if(val=='U') advance();
  if(val=='D') back();
  if(val=='L') left() ;
  if(val=='R') right();
if(val=='S') stopp();
}
```

Program č. 31: Projekt snímače

```
#include <dht11.h>
dht11 DHT;
#define DHT11_PIN A1

#include <Servo.h>
int pinLB = 12;     // definuj  pin 12
int pinLF = 3;      // definuj  pin 3
int pinRB = 13;     // definuj  pin 13
int pinRF = 11;     // definuj  pin 11
////////////////////////////////
int inputPin = 4;    // definuj  pin pre senzor prijatia signálu - vzdialenosť
int outputPin = 5;    // define pin pre senzor vysielania signálu - vzdialenosť

int smokeA0 = A5;  // analógový signál pre dym

int Fspeedd = 0;       // vzdialenosť vpredu
int Rspeedd = 0;      // vzdialenosť vpravo
int Lspeedd = 0;       // vzdialenosť vľavo
int directionn = 0;     // prerušenie vpredu=8 vzadu=2 vľavo=4 vpravo=6
Servo myservo;        // nastav servomotor
int delay_time = 250; // doba usadzovania po pohybe servomotora B

int Fgo = 8;         // pohyb vpred
int Rgo = 6;         // pohyb vpravo
int Lgo = 4;         // pohyb vľavo
int Bgo = 2;         // pohyb vzad

void setup(){
Serial.begin(9600);
Serial.println("DHT TEST PROGRAM ");
Serial.print("LIBRARY VERSION: ");
Serial.println(DHT11LIB_VERSION);
Serial.println();
Serial.println("Type,\tstatus,\tHumidity (%),\tTemperature (C)");

 // definuj výstupný pin pre motor
 pinMode(pinLB,OUTPUT); // pin 12
 pinMode(pinLF,OUTPUT); // pin 3 (PWM)
 pinMode(pinRB,OUTPUT); // pin 13
 pinMode(pinRF,OUTPUT); // pin 11 (PWM)
 pinMode(inputPin, INPUT);        // definuj vstupný pin pre senzor
 pinMode(outputPin, OUTPUT);
```

```
 pinMode(smokeA0, INPUT);
}

void back()                        // posun dozadu
  {
   digitalWrite(pinLB,HIGH);  // motor sa pohybuje doľava
   digitalWrite(pinRB,HIGH);  // motor sa pohybuje doprava
   analogWrite(pinLF,255);
   analogWrite(pinRF,255);
  }


void advance()                      // posun dopredu
  {
  digitalWrite(pinLB,LOW);       // pravé koleso dopredu
  digitalWrite(pinRB, LOW);      // ľavé koleso dopredu
  analogWrite(pinLF,255);
  analogWrite(pinRF,255);
  delay(800);                         //  posun dozadu
  }
void stopp()                          // stop
  {
   digitalWrite(pinLB,HIGH);
   digitalWrite(pinRB,HIGH);
   analogWrite(pinLF,0);
   analogWrite(pinRF,0);
  }

void right()                          // posun vpravo
  {
 digitalWrite(pinLB,HIGH);         // ľavé koleso dopredu
 digitalWrite(pinRB,LOW);          // pravé koleso dozadu
 analogWrite(pinLF, 255);
 analogWrite(pinRF,255);
  }

void left()                           // posun vľavo
  {
 digitalWrite(pinLB,LOW);          // ľavé koleso sa pohybuje dozadu
 digitalWrite(pinRB,HIGH);         // pravé koleso sa pohybuje dopredu
 analogWrite(pinLF, 255);
 analogWrite(pinRF,255);
  }

void ask_pin_F()                            // meranie vzdialenosti vpredu
  {
    myservo.write(90);
    digitalWrite(outputPin, LOW); // ultrazvukový snímač prenáša nízku úroveň signálu 2µs
    delayMicroseconds(2);
    digitalWrite(outputPin, HIGH); // ultrazvukový sen. Vysielajú vysokú úroveň signálu 10µs,
                                   //najmenej 10µs
    delayMicroseconds(10);
    digitalWrite(outputPin, LOW);                // udržujte vysielanie signálu nízkej úrovne
```

```
    float Fdistance = pulseIn(inputPin, HIGH);  // čítaj čas medzitým
    Fdistance= Fdistance/5.8/10;                // konvertuj čas na vzdialenosť (jednotka: cm)
    Fspeedd = Fdistance;                        // čítaj vzdialenosť vpredu
   }

void loop(){
advance();
delay(4000);
stopp();
  int chk;
  ask_pin_F();
  int analogSensor = analogRead(smokeA0);
  Serial.print("DHT11, \t");
  chk = DHT.read(DHT11_PIN);    // READ DATA
  switch (chk){
    case DHTLIB_OK:
                Serial.print("OK,\t");
                break;
    case DHTLIB_ERROR_CHECKSUM:
                Serial.print("Checksum error,\t");
                break;
    case DHTLIB_ERROR_TIMEOUT:
                Serial.print("Time out error,\t");
                break;
    default:
                Serial.print("Unknown error,\t");
                break;
  }

 // vypíš namerané hodnoty
  Serial.print("Vlhkost: ");
  Serial.print(DHT.humidity,1);
  Serial.print("Teplota: ");
  Serial.println(DHT.temperature,1);
  Serial.print("Vzdialenost: ");
  Serial.println(Fspeedd,1);
  Serial.print("Dym: ");
  Serial.println(analogSensor,1);
stopp();
delay(1000);
back();
delay(4000);
stopp();
}
```