

PODROBNÝ PRŮVODCE TEORIÍ

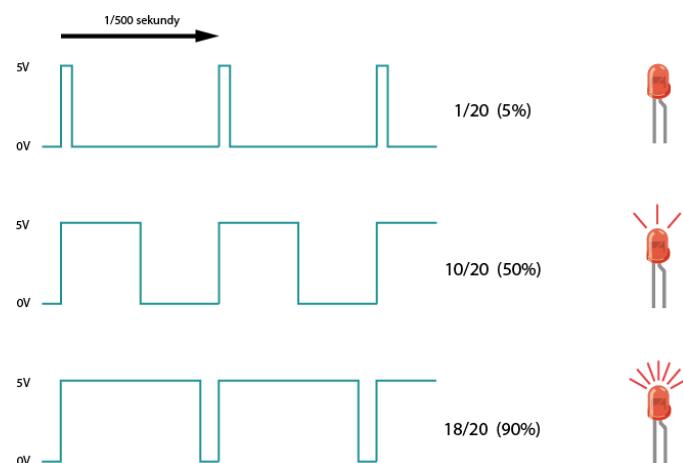
POUŽITÍ TŘÍBAREVNÉ DIODY LZE VYUŽÍT NAPŘÍKLAD K VYTVOŘENÍ LAMPY, KTERÁ BUDE PLYNULE MĚNIT BARVU.

OBSAH PRŮVODCE

- ① Seznámení se s RGB diodou a její funkčností.
- ② Princip skládání barev a PWM.
- ③ Správné zapojení RGB diody s využitím zkušeností z předchozích příkladů. Zejména se jedná o volbu rezistorů a jejich zapojení.
- ④ Zavedení pojmu a znalostí týkající se analogových výstupů a mapování hodnot.
- ⑤ Využití již zavedeného podmínkového příkazu **if**.
- ⑥ Zavedení vlastní funkce v programovém kódu.
- ⑦ Použití příkazu cyklu **for**.

PWM

Arduino má na svých výstupech napětí 5V. Toto napětí nelze souvisle měnit. Jak tedy ovládat například jas diody? K tomu se využívá technika, která se nazývá **Pulsně Šířková Modulace**. Z anglického Pulse Width Modulation se používá všeobecně známá zkratka **PWM**. Při ovládání jasu diody PWM velmi rychle přepíná na výstupu pinů hodnoty s logickou nulou (0V) a logickou jedničkou (+5V). To se děje v určitém čase. Tyto změny jsou tak rychlé, že je lidské oko nedokáže díky své setrvačnosti zachytit. Z poměru času, ve kterém je na výstupu +5V ku stavu 0V se pak odvíjí intenzita svícení LED diody nebo rychlosť otáčení motoru – Obr. 1.



Obr. 1 – Pulsně šířková modulace

Střídavé nastavování jedniček a nul pomocí programu, by poměrně hodně zaměstnalo procesor, a navíc by se nedosáhlo dostatečně vysokého frekvence spínání. Arduino ale poskytuje možnost využít modulu čítačů, které umožňují generování PWM hardwarově. Stačí vhodně nastavit čítač do režimu PWM, určit periodu a plnění. Na výstupu je pak trvale



generován signál s pulzně šířkovou modulací automaticky.

Jediným omezením je to, že nelze použít libovolného piny, ale pouze těch, na které je výstup čítačů přiveden. Arduino UNO má šest pinů určených pro PWM - **3, 5, 6, 9, 10 a 11**.

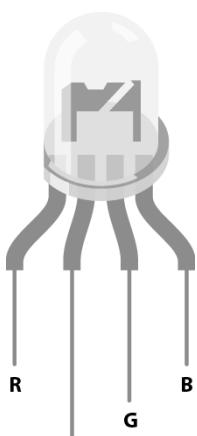
Lze je rozpoznat symbolem **~** před číslem.

V samotném programu se nastavuje pouze plnění, což je poměr doby zapnutí a celé periody. Takže rozsah 0-255, odpovídá poměru 0-100%.

LED DIODA RGB

V předchozích příkladech jste získali zkušenosti se zapojením a chováním obyčejných LED diod. RGB dioda pracuje obdobně. Její označení RGB je odvozeno od anglického pojmenování barev – červená, zelená, modrá. Tato LED dioda ve skutečnosti představuje tři LED diody v jednom balení: jedna je červená, jedna zelená a jedna modrá.

Existují dva druhy RGB diod. Dioda se **společnou anodou** a se **společnou katodou**. Anoda je pozitivní kontakt a katoda je negativní kontakt. V závislosti na druhu RGB diody se liší i její zapojení, i když pro Arduino to nemá prakticky žádný význam, protože obě využívají stejné napětí.

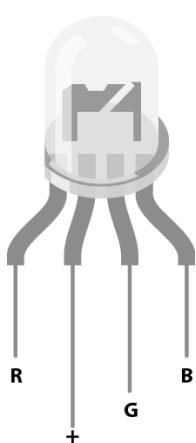


Obr. 2 – RGB dioda se společnou katodou

RGB dioda se **společnou katodou** má vlastní kladný (anodový) vývod, ale všechny sdílejí jeden záporný (katodový) vývod. RGB diody, mají vývody o různé délce. Při běžných LED diodách je krátký vývod negativní, ale u RGB diody se společnou katodou je negativní vývod ten nejdelší.

RGB dioda se **společnou anodou** má vlastní záporný (katodový) vývod, ale všechny sdílejí jeden kladný (anodový) vývod. U této diody je nejdelší vývod kladný.

Ostatní kratší vývody mají označení podle barev a u obou typů RGB diod jsou stejné. Liší se pouze jejich zapojení do obvodu. Vzhledem k tomu, že každý kratší vývod odpovídá konkrétní barvě, lze RGB diodu zapojit do obvodu, stejně jako by to byly tři samostatné LED diody.

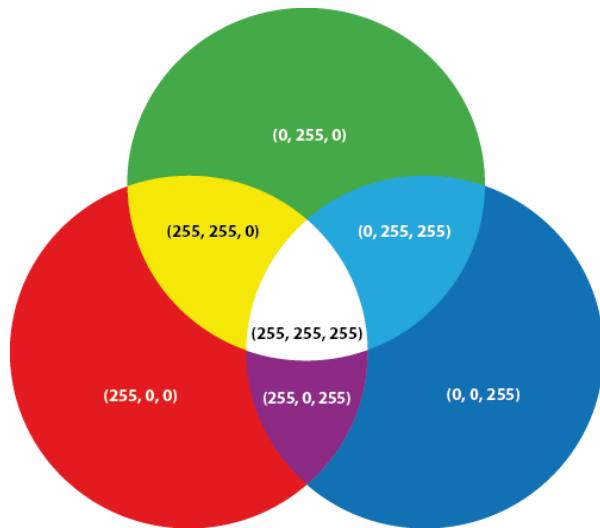


Obr. 3 - RGB dioda se společnou anodou

Pro RGB diody se **společnou katodou** se vývody R, G, B připojí k napájení nebo na výstupní pin Arduina. Před kladné vývody se ještě zapojují rezistory, které zajistí, aby nedošlo k poškození RGB diody.

Pro RGB diody se **společnou anodou** se vývody R, G, B připojí přímo k výstupním pinům Arduina a anoda se připojí na napájení s předřadným rezistorem.

RGB diody jsou skvělé, protože je lze využít k zobrazení pestrých barev. Červená, zelená a modrá jsou základní barvy a v přídavné barevné paletě je lze kombinovat a vytvářet další odstíny. Možnosti kombinace barev jsou uvedeny na obrázku Obr. 4 - Kombinace barev.



Obr. 4 - Kombinace barev

Na obrázku je také vidět, že kombinace barev je dána trojicí hodnot 0, 255. S ohledem na Arduino a PWM lze tyto limitní hodnoty přiřadit logickým úrovním 0 – **LOW** a 255 – **HIGH**. Z obrázku Obr. 4 - Kombinace barev je patrné, že pro zelenou barvu by kombinace hodnot vypadala: **[LOW, HIGH, LOW]**. To platí pro diody se společnou katodou. Pro diodu se společnou anodou by hodnoty byly opačně: **[HIGH, LOW, HIGH]**. Kombinace pro jednotlivé barvy jsou uvedeny v tabulce Tab 1 - Kombinace barev.

	Společná katoda	Společná anoda
Červená	HIGH, LOW, LOW	LOW, HIGH, HIGH
Modrá	LOW, LOW, HIGH	HIGH, HIGH , LOW
Zelená	LOW, HIGH , LOW	HIGH , LOW, HIGH

Tab 1 - Kombinace barev



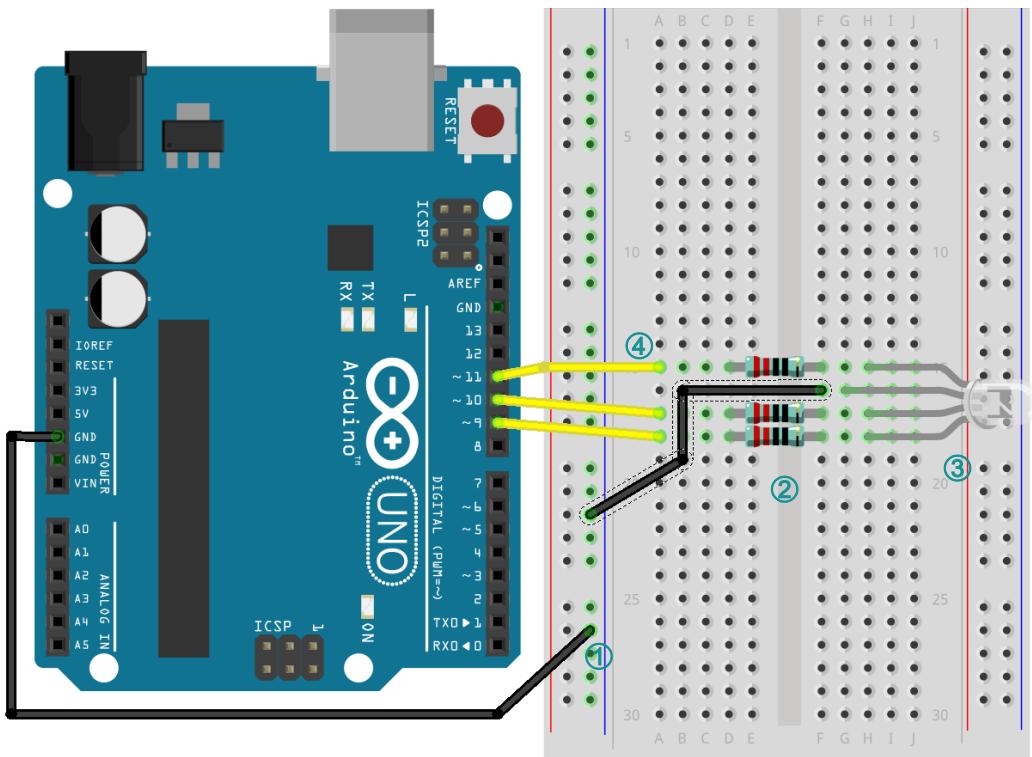
Pokud se budou požívat pouze krajní hodnoty pro definici barev, tj. 0 a 255, potom si vystačíme s funkcí **digitalWrite()**. Ovšem pokud budeme chtít řešit plynulý přechod mezi barvami s využitím PWM, musíme použít funkci **analogWrite()**.



FUNKCE ANALOGWRITE()

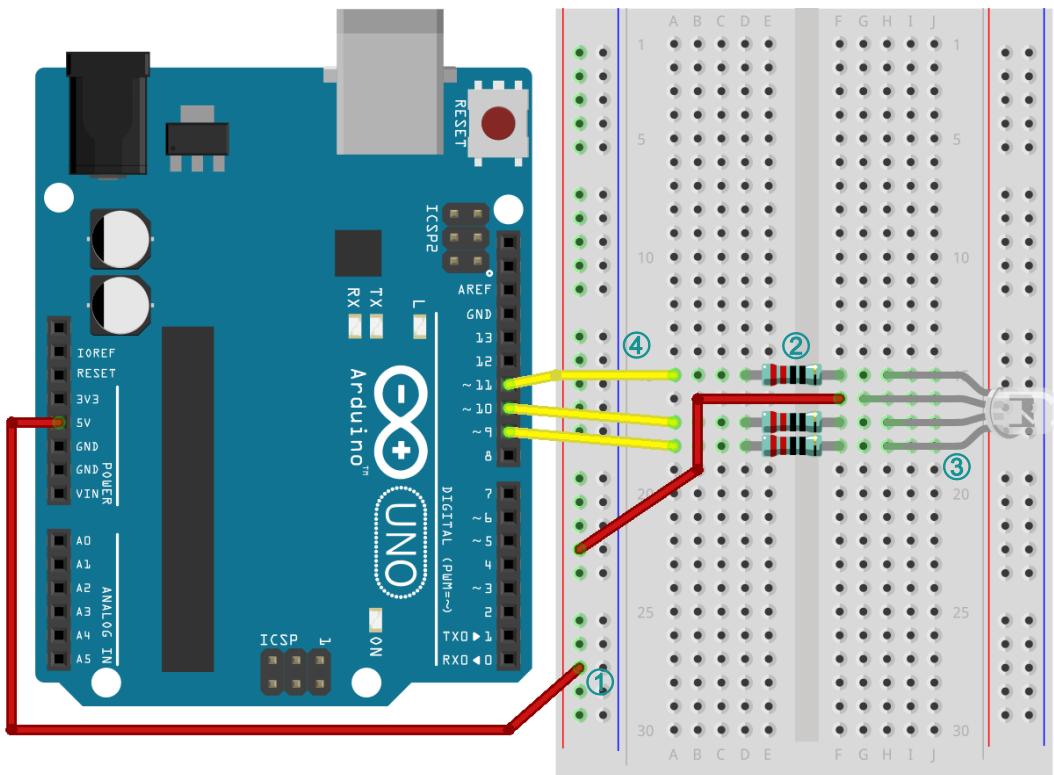
- ➔ Funkce `analogWrite(pin, hodnota)` posílá analogový signál na uvedený pin ve formě PWM. Při použití této funkce již nepracujeme pouze s krajními hodnotami, ale v plném rozsahu 0-255.

ZAPOJENÍ OBVODU



Obr. 5 - Zapojení RGB diody se společnou katodou

- ① Vodič napájení z Arduino zapojte do kontaktní desky k **modré** čáře. Tím se zpřehlední celé zapojení.
- ② Najděte tři rezistory o hodnotě 220Ω (dva oranžové proužky, modrý, černý a zlatý). Je důležité, aby tyto rezistory byly zapojeny do kontaktní desky tak, aby propojovaly její obě poloviny. Pokud by tomu tak nebylo, došlo by ke zkratování. Rezistory zajistí snížení proudu, aby nedošlo ke zničení RGB diody.
- ③ RGB diodu zapojte do kontaktní desky orientovanou tak, jak je uvedeno na obrázku Obr. 6 - Zapojení RGB diody se společnou katodou. Nejdélší vývod je propojen s **uzemněním**. Zbylé vodiče jsou propojeny s rezistory.
- ④ Druhá strana rezistorů je zapojena do digitálních pinů 9, 10 a 11.



Obr. 6 - Zapojení RGB diody se společnou anodou

- ① Vodič napájení 5V z Arduino zapojte do kontaktní desky k **červené** čáře.
- ② Najděte rezistory o hodnotě 220Ω (dva oranžové proužky, modrý, černý a zlatý). Je důležité, aby tyto rezistor byly zapojeny do kontaktní desky tak, aby propojovaly její obě poloviny. Pokud by tomu tak nebylo, došlo by ke zkratování. Rezistory opět zajistí snížení proudu, aby nedošlo ke zničení RGB diody.
- ③ RGB diodu zapojte do kontaktní desky orientovanou tak, jak je uvedeno na obrázku Obr. 6 - Zapojení RGB diody se společnou anodou. Nejdelší vývod je propojen s **napájením** z desky Arduino.
- ④ Zbylé vodiče jsou připojeny do digitálních pinů 9, 10 a 11.

Lze říci, že na vývody RGB diody – anody i katody, bude zasílán signál PWM. Podle hodnoty na jednotlivých vývodech tohoto signálu se mění barva RGB diody.

PROGRAMOVÝ KÓD

Při psaní kódu pro všechny příklady si vystačíme se základním zapojením podle schématu na Obr. 2 – RGB dioda se společnou katodou nebo Obr. 3 - RGB dioda se společnou anodou.



Příklady na sebe navazují, takže pro jejich inovace stačí provádět pouze dílčí úpravy programového kódu. Není nutné vždy psát celý program od začátku.

ZÁKLADNÍ PŘÍKLAD

Zde je uveden základní programový kód, který zobrazí na RGB diodě se společnou anodou modrou barvu.

```
1 void setup() {  
2     pinMode(11, OUTPUT);          //červená  
3     pinMode(10, OUTPUT);          //zelená  
4     pinMode(9, OUTPUT);           //modrá  
5 }  
6  
7 void loop() {  
8     digitalWrite(11, HIGH);        //červená  
9     digitalWrite(10, HIGH);        //zelená  
10    digitalWrite(9, LOW);         //modrá  
11 }
```



- ① Každý pin, který ovládá RGB diodu, je definován funkcí **pinMode()** pro nastavení výstupu. V příkladu jsou definovány piny **11, 10, 9**.
- ② Podobně jako u jednobarevné LED diody i v případě RGB se využívá funkce **digitalWrite()**. Tentokrát se používá pro každou barvu (vývod) RGB diody. Pro zobrazení modré barvy jsou zasílány signály do vývodu pro červenou a zelenou barvu. Vzhledem k tomu, že se jedná o diodu se společnou anodou jsou hodnoty **HIGH** a **LOW** zadány inverzně, ve srovnání s Obr. 4 - Kombinace barev.



(Př. 1) Změňte logické hodnoty ve funkci `digitalWrite()` tak, aby RGB dioda svítila zeleně, modře a červeně.



NESVÍTÍ DIODA

Zapojení RGB diody – zkонтrolujte její zapojení v kontaktním poli. Ujistěte se, zda se jedná o diodu se společnou anodou nebo katodou. Zapojení se tím významně liší.

Zapojení v desce – zkонтrolujte, zda jsou vývody zapojeny do odpovídajících pinů desky Arduino. Musí být využity piny, které poskytují PWM, tedy piny, které mají u svého čísla symbol `~`.

Rezistory – zkонтrolujte, zda jste použili správné rezistory.

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.

ZÁKLADNÍ PŘÍKLAD – INOVACE S PWM

Nahrazení logických hodnot **HIGH** a **LOW** konkrétními číselnými hodnotami z rozsahu PWM.

```
1 void setup() {  
2     pinMode(11, OUTPUT);          //červená  
3     pinMode(10, OUTPUT);          //zelená  
4     pinMode(9, OUTPUT);           //modrá  
5 }  
6  
7 void loop() {  
8     digitalWrite(11, 255);        //červená  
9     digitalWrite(10, 255);        //zelená  
10    digitalWrite(9, 0);           //modrá  
11 }
```



Příklad je totožný se základní variantou. Rozdíl je v nahrazení logických hodnot **HIGH** a **LOW** číselnými hodnotami **255** a **0**. Výhodou této varianty je možnost definovat celou škálu barev v závislosti na rozsahu hodnot **0-255**.



(Př. 2) Změňte hodnoty ve funkci **digitalWrite()** tak, aby RGB dioda svítila tyrkysově, žlutě a fialově. Vyzkoušejte, jak se bude měnit barva se změnou hodnot v uvedené funkci.



Pro definici barvy v hodnotách RGB lze využít některý z grafických programů pro úpravu fotografií nebo jednoduchý nástroj, který je na adrese: <http://www.colorpicker.com>. Musí být ale dodržen princip skládání barev pro RGB diodu. K pochopení lze využít animaci:

<https://milannovak.gitbooks.io/arduino/content/rgb-led-dioda/teorie.html>

ZÁKLADNÍ PŘÍKLAD – PRŮBĚŽNÁ ZMĚNA BAREV

Příklad ukazuje, jak průběžně měnit barvy RGB diody v nekonečné smyčce. K tomu se využívá funkce `delay()`. Jediným parametrem je doba, na kterou se běh programu pozastaví. Doba se uvádí v milisekundách, tzn. 1000ms = 1s.

```
1 void setup() {
2     pinMode(11, OUTPUT);          //červená
3     pinMode(10, OUTPUT);          //zelená
4     pinMode(9, OUTPUT);           //modrá
5 }
6
7 void loop() {
8     // Kód pro zelenou barvu
9     digitalWrite(11, 0);
10    digitalWrite(10, 255);
11    digitalWrite(9, 255);
12    delay(3000);                |---①
13    // Kód pro modrou barvu
14    digitalWrite(11, 255);
15    digitalWrite(10, 0);
16    digitalWrite(9, 255);
17    delay(3000);                |---①
18    // Kód pro červenou barvu
19    digitalWrite(11, 255);
20    digitalWrite(10, 0);
21    digitalWrite(9, 255);
22    delay(3000);                |---①
23 }
```

- ① Každý blok příkazů, které zajišťují rozsvícení diody v konkrétní barvě, je oddělen funkcí `delay(3000)`. Tím je zajištěno, že každá barva RGB bude svítit 3 sekundy.



(Př. 3) Přidejte další tři barvy tak, aby každá třikrát blikla a pak se zobrazila barva další. Interval blikání nastavte na 1 sekundu. Prodleva přechodu mezi barvami bude 3 sekundy.



Kombinace číselných hodnot PWM pro definici barvy, je dána tím, zda se používá dioda se společnou anodou nebo katodou. V uvedeném příkladu se barvy definují pro RGB diody se **společnou anodou**. Pro diodu se **společnou katodou** budou hodnoty opačné.

ZÁKLADNÍ PŘÍKLAD – FUNKCE PRO ZOBRAZENÍ BARVY

V předchozích příkladech jsme si vystačili se základní strukturou programového kódu. Vše fungovalo, jak mělo, ale při složitějších úlohách začíná být programový kód nepřehledný. K zpřehlednění nám opět pomůže využití vlastní **funkce**.

Zavedeme funkci **void setColor(int redC, int greenC, int blueC)**. Tato funkce má tři parametry, kterými budou čísla, která charakterizuje slovo **int**, což je opět datový typ (integer).

```
1 void setup() {  
2     pinMode(11, OUTPUT);          //červená  
3     pinMode(10, OUTPUT);          //zelená  
4     pinMode(9, OUTPUT);           //modrá  
5 }  
6  
7 void loop() {  
8     setColor(255,0,255);         //zelená barva  
9     delay(2000);  
10    setColor(255,255,0);          //modrá barva  
11    delay(2000);  
12 }  
13  
14 void setColor(int redC, int greenC, int blueC ) {  
15     digitalWrite(11, redC);  
16     digitalWrite(10, greenC);  
17     digitalWrite(9, blueC);  
18 }
```



- ② Pro složení hodnot konkrétní barvy je deklarovaná funkce **setColor** se třemi parametry.
- ③ Deklarovaná funkce **setColor** se volá ve smyčce (mj. také ve funkci **loop**) s číselnými hodnotami konkrétní barvy jako jejími parametry. Funkce se volá s pauzami 2 sekundy.



(Př. 4) Přidejte další tři barvy tak, aby každá třikrát blikla a pak se zobrazila barva další. Interval blikání nastavte na 1 sekundu. Prodleva přechodu mezi barvami bude 3 sekundy.

Na rozdíl od předchozího příkladu, tentokrát použijte funkci pro zobrazení barvy.



(Př. 5) Neustále řešíme, jak bude kombinace barev vypadat pro RGB diodu se společnou anodou a katodou. Upravte programový kód tak, aby se zadávaly hodnoty stejně pro jakýkoliv typ diody. Využijte k tomu podmínkový příkaz **if**.

PROJEKT - MAGICKÁ LAMPA

Tento projekt využije všechny poznatky z předchozích příkladů. Využije se zde i cyklu **for**, díky kterému bude magická lampa plynule měnit barvy v jednotlivých odstínech.

```
1 const int redPin = 11;          | ①
2 const int greenPin = 10;        | ②
3 const int bluePin = 9;         |
4
5 int redIntens;                | ③
6 int greenIntens;              | ④
7 int blueIntens;              |
8
9 int x;                        | ⑤
10
11 int display_time = 10;        | ⑥
12 int common_anode=1;
13
14 void setup(){                  | ⑦
15   pinMode(redPin, OUTPUT);    |
16   pinMode(greenPin, OUTPUT);  |
17   pinMode(bluePin, OUTPUT);   |
18 }
19
20 void loop(){                  | ⑧
21   for (x = 0; x < 767; x++){|
22
23     if(x <= 255){            | ⑨
24       redIntens = 255 - x;   |
25       greenIntens = x;      |
26       blueIntens = 0;        |
27     }else if (x <= 511){    | ⑩
28       redIntens = 0;        |
29       greenIntens = 255 - (x - 256); |
30       blueIntens = (x - 256); |
31     }else{                  |
32       redIntens = (x - 512); |
33       greenIntens = 0;      |
34       blueIntens = 255 - (x - 512); |
35     }
36
37   setColor(redIntens, blueIntens, greenIntens); |
38   delay(display_time);           |
39 }
```

```

43 void setColor(int redC, int greenC, int blueC){
44     if(common_anode==1){
45         redC=255-redC;
46         greenC=255-greenC;
47         blueC=255-blueC;
48     }
49     analogWrite (redPin, redC);
50     analogWrite (greenPin, greenC);
51     analogWrite (bluePin, blueC);
}

```



- ① Definice pinů pro připojení RGB diody.
- ② Deklarace proměnných **redIntens**, **greenIntens**, **blueIntens**, které budou obsahovat PWM hodnoty pro zobrazení konkrétní barvy.
- ③ Deklarace proměnné **x** pro kroky cyklu **for**.
- ④ Deklarace proměnné **display_time**, kde je uložen údaj o prodlevě při plynulé změně barev diody. Proměnná **common_anode** zajišťuje informaci, zda se používá dioda se společnou anodou (hodnota 1) nebo katodou (hodnota jiná než 1).
- ⑤ Vyhrazení použitých pinů na desce Arduino. Hodnoty jsou definovány v ①.
- ⑥ Začátek cyklu **for**. Cyklus **for** bude probíhat od 0 do 767. Hodnota 767 vychází z limitní hodnoty pro každou barvu, která je 255. Objasnění vyplýne z níže uvedeného popisu podmínky **if**.
- ⑦ První část podmínky zajistí, že výchozí barvou bude **červená**. Když si za proměnou x v prvním kroku cyklu dosadíme konkrétní hodnotu 0, získáme kombinaci PWM hodnot **redIntens=255-0; greenIntens=0; blueIntens=0;** což je červená barva.
- ⑧ Druhá část podmínky sleduje proměnnou od x=256. Opět, když dosadíme konkrétní hodnoty v prvním kroku cyklu, vyjde nám následující: **redIntens=0; greenIntens=255-(256-256); blueIntens=(256-256);** což je kombinace 0, 255, 0 a tedy zelená barva.
- ⑨ Třetí část podmínky pracuje s hodnotami vyššími jak 511. Po dosazení konkrétních hodnot: **redIntens=(512-512); greenIntens=0; blueIntens=255-(512-512);** vyjde kombinace 0, 0, 255 tj. modrá barva. Vzhledem k průběžné změně hodnot proměnné x dochází k plynulému přechodu mezi barvami.

- ⑩ Volání vlastní funkce pro zobrazení konkrétní barvy, se třemi parametry, jejichž hodnoty jsou průběžně počítány v bloku podmínkového příkazu **if**. Aby se barva stihla zobrazit, je provedena prodleva pomocí **delay**.
- ⑪ V závislosti na tom, jestli se používá dioda se společnou anodou nebo katodou, se upraví výsledné hodnoty barevných kombinací.
- ⑫ Samotné zobrazení konkrétní barvy.



Nyní, když už je připravena programová část a je plně funkční, je čas si projekt ukázat v praktickém „balení“. Podpořte svou kreativitu a vytvořte jednoduché stínidlo pro magickou lampu. Návod je uveden v následující kapitole.

STÍNÍTKO PRO MAGICKOU LAMPU

Pro vytvoření stínítka budeme potřebovat: tmavý karton (tvrdší papír), pauzák (průsvitný papír), lepidlo, nůžky.

PAPÍROVÁ KONSTRUKCE

V této kapitole je ukázka základní papírové konstrukce, kterou můžete začít. Doporučujeme, abyste si později vzhled přizpůsobili podle vlastních potřeb a nápadů. Pokud máte již svoji jasnou představu jak by lampa mohla vypadat, nemusíte využít přiložených šablon.

Na tiskárně si vytiskněte přiloženou šablonu na pevný papír, podle obrázku Obr. 5 - Šablona stínítka. Stínítko bude mít tvar krychle.

V první řadě vystříhněte jednotlivé díly z vytisknuté šablony Obr. 6 - Vystřízení dílů ze šablony. Připravte si průsvitný papír (pauzák), který později upravíte pro velikost, která bude odpovídat složené konstrukci.



Obr. 6 - Vystřízení dílů ze šablony



Obr. 5 - Šablona stínítka

SLOŽENÍ DÍLŮ

Připravte si vystřížené části konstrukce stínítka. Měli byste mít horní a dolní díl, sloupky a samozřejmě průsvitný papír Obr. 7 - Díly stínítka.

Nejprve připravte dolní díl, tj. ten s otvorem. Podle přerušovaných čar ohněte okraje do pravého úhlu. Do vnitřní části základny ohněte i spojovací západky, na které naneste trochu lepidla, jak je znázorněno na obrázku Obr. 8 - Složení základny. Stejný postup aplikujte i pro složení horního dílu.



Obr. 8 - Složení základny



Obr. 7 - Díly stínítka

Dále naohýbejte všechny postranní díly. Ohnutí je vždy provedeno podle přerušované čáry vedené středem dílů. Jakmile máte složené všechny díly, mělo by všech šest částí vypadat tak jako na obrázku Obr. 9 - Složené díly.

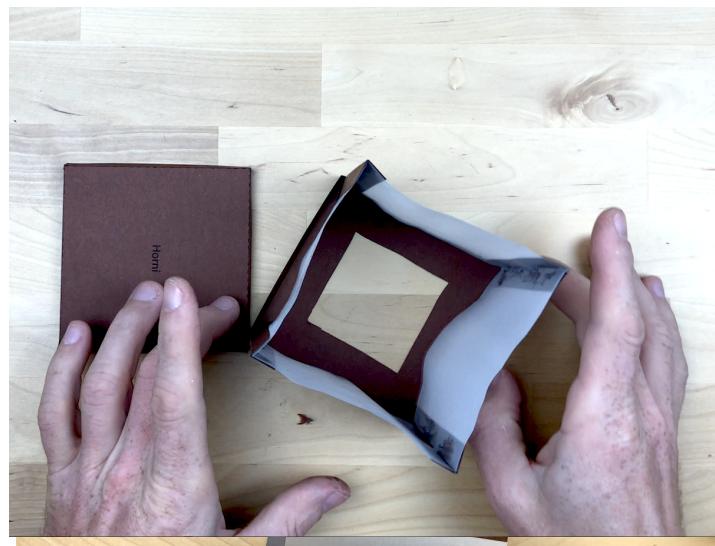
K naohýbaným rohům přilepte průhledný papír (pauzák) tak, jak je vidět na obrázku Obr. 10 - Lepení bočních rohů



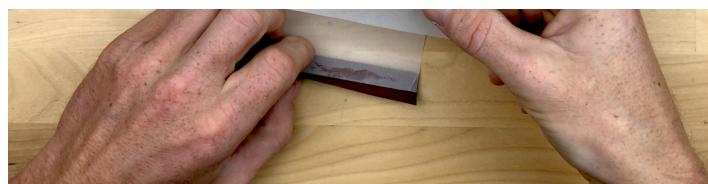
Obr. 9 - Složené díly

Když jsou nalepeny všechny boční stěny, spojte poslední díl s posledním rohem, jak je vidět na obrázku Obr. 11 - Lepení bočních stěn.

Všechny stěny vložte do podstavy. Podstavu můžete mírně namazat lepidlem a stěny k podstavě přilepte.



Obr. 12 - Přiložení bočních stěn



Obr. 11 - Lepení bočních stěn

Následně přidejte horní díl. Naneste trochu lepidla do rohů horního dílu a jednoduše jej přiložte na konstrukci stínítka viz. Obr. 13 - Přiložení horního dílu.

Kompletní stínítko je zobrazeno na obrázku Obr. 14 - Hotové stínítko. Nyní stačí stínítko přiložit na nepájivé pole s RGB diodou a spustit program.



Obr. 14 - Hotové stínítko

Obr. 13 - Přiložení horního dílu

ŘEŠENÍ ÚLOH

(PŘ. 1

Kombinace hodnot pro jednotlivé barvy pro RGB diodu se **společnou anodou**. Pro RGB diodu se **společnou katodou** budou hodnoty **HIGH** a **LOW** opačně. Jednotlivé bloky kódu pro zelenou, modrou a červenou barvu se vkládají zvlášť. Po každou barvu se kód nahraje do desky a dioda by měla svítit definovanou barvou. Následně se změní kombinace hodnot **HIGH** a **LOW** a opět se kód nahraje do desky.

```
1 void setup() {
2     pinMode(11, OUTPUT);
3     pinMode(10, OUTPUT);
4     pinMode(9, OUTPUT);
5 }
6
7 void loop() {
8     // Kód pro zelenou barvu
9     digitalWrite(11, HIGH);
10    digitalWrite(10, LOW);
11    digitalWrite(9, HIGH);
12
13    // Kód pro modrou barvu
14    digitalWrite(11, HIGH);
15    digitalWrite(10, HIGH);
16    digitalWrite(9, LOW);
17
18    // Kód pro červenou barvu
19    digitalWrite(11, LOW);
20    digitalWrite(10, HIGH);
21    digitalWrite(9, HIGH);
22 }
```

(PŘ. 2

Kombinace hodnot PWM pro jednotlivé barvy pro RGB diodu se společnou anodou. Pro každou barvu se opět nahraje kód do desky zvlášť.

```
1 void setup() {
2     pinMode(11, OUTPUT);
3     pinMode(10, OUTPUT);
4     pinMode(9, OUTPUT);
5 }
6
7 void loop() {
8     // Kód pro tyrkysovou barvu
9     digitalWrite(11, 255);
10    digitalWrite(10, 0);
11    digitalWrite(9, 0);
12
13    // Kód pro žlutou barvu
14    digitalWrite(11, 0);
15    digitalWrite(10, 0);
16    digitalWrite(9, 255);
17
18    // Kód pro fialovou barvu
19    digitalWrite(11, 0);
20    digitalWrite(10, 255);
21    digitalWrite(9, 0);
22 }
```

(PŘ. 3

Řešení je velice jednoduché a vede spíše k upevnění již naučeného, ale je ukázkou dlouhého a neefektivního programového kódu, pokud se nevyužije programových struktur, jakými jsou např. funkce.

```
1 void setup() {
2     pinMode(11, OUTPUT);
3     pinMode(10, OUTPUT);
4     pinMode(9, OUTPUT);
5 }
6
7 void loop() {
8     // Kód pro zelenou barvu
9     digitalWrite(11, 255);
10    digitalWrite(10, 0);
11    digitalWrite(9, 255);
12    delay(1000);
13    digitalWrite(11, 255);
14    digitalWrite(10, 0);
15    digitalWrite(9, 255);
16    delay(1000);
17    digitalWrite(11, 255);
18    digitalWrite(10, 0);
19    digitalWrite(9, 255);
20    delay(3000);
21
22    // Kód pro modrou barvu
23    digitalWrite(11, 255);
24    digitalWrite(10, 255);
25    digitalWrite(9, 0);
26    delay(3000);
27    digitalWrite(11, 255);
28    digitalWrite(10, 255);
29    digitalWrite(9, 0);
30    delay(3000);
31    digitalWrite(11, 255);
32    digitalWrite(10, 255);
33    digitalWrite(9, 0);
34    delay(3000);
35    // Výše uvedené řádky kódu opakujeme pro další
36    // barvy. Dioda třikrát blikne a změní
37    // se barva.
38 }
39
```

(PŘ. 4

Řešení s využitím vlastní funkce poskytuje značné zjednodušení a zpřehlednění programového kódu.

```
1 void setup() {
2     pinMode(11, OUTPUT);
3     pinMode(10, OUTPUT);
4     pinMode(9, OUTPUT);
5 }
6
7 void loop() {
8     //zelená barva
9     setColor(255,0,255);
10    delay(1000);
11    setColor(255,0,255);
12    delay(1000);
13    setColor(255,0,255);
14    delay(3000);
15    //modrá barva
16    setColor(255,255,0);
17    delay(1000);
18    setColor(255,255,0);
19    delay(1000);
20    setColor(255,255,0);
21    delay(3000);
22    //červená barva
23    setColor(0,255,255);
24    delay(1000);
25    setColor(0,255,255);
26    delay(1000);
27    setColor(0,255,255);
28    delay(3000);
29
30    // Výše uvedené řádky kódu opakujeme pro další
31    // barvy. Dioda třikrát blikne a změní
32    // se barva.
33 }
34
35 void setColor(int redC, int greenC, int blueC ) {
36     digitalWrite(11, redC);
37     digitalWrite(10, greenC);
38     digitalWrite(9, blueC);
39 }
```

(PŘ. 5

Příklad využívá podmínkového příkazu **if**, který testuje hodnotu proměnné **common_anode**. Pokud bude hodnota proměnné **1**, bude se jednat o zapojení RGB diody se společnou anodou a tím se zadávané hodnoty pro definici barvy odečtou od čísla **255**. V opačném případě, pokud bude hodnota proměnné rovna jiné hodnotě než **1**, budou zadávané hodnoty ponechány.

```
1 int common_anode=1;           //pokud se bude jednat o
2                                         //diodu se společnou
3                                         //anodou, hodnota proměnné
4                                         //bude rovna 1, v opačném
5                                         //případě změňte na jinou
6
7 void setup() {
8     pinMode(11, OUTPUT);
9     pinMode(10, OUTPUT);
10    pinMode(9, OUTPUT);
11 }
12
13 void loop() {
14     setColor(255,0,0);          //červená barva
15     delay(2000);
16     setColor(0,0,255);         //modrá barva
17     delay(2000);
18 }
19
20 void setColor(int redC, int greenC, int blueC ){
21     if(common_anode==1){
22         redC=255-redC;
23         greenC=255-redC;
24         blueC=255-redC;
25     }
26
27     digitalWrite(11, redC);
28     digitalWrite(10, greenC);
29     digitalWrite(9, blueC);
30 }
```