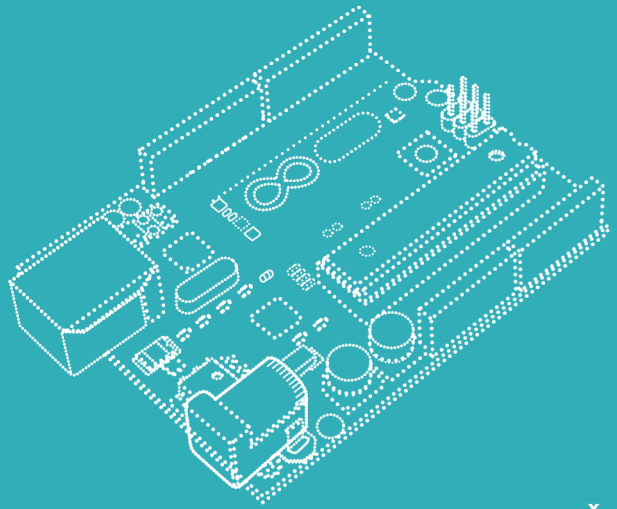


OVLÁDÁNÍ SVĚTELNÉ KŘIŽOVATKY POMOCÍ ARDUINA – SEMAFOR 1

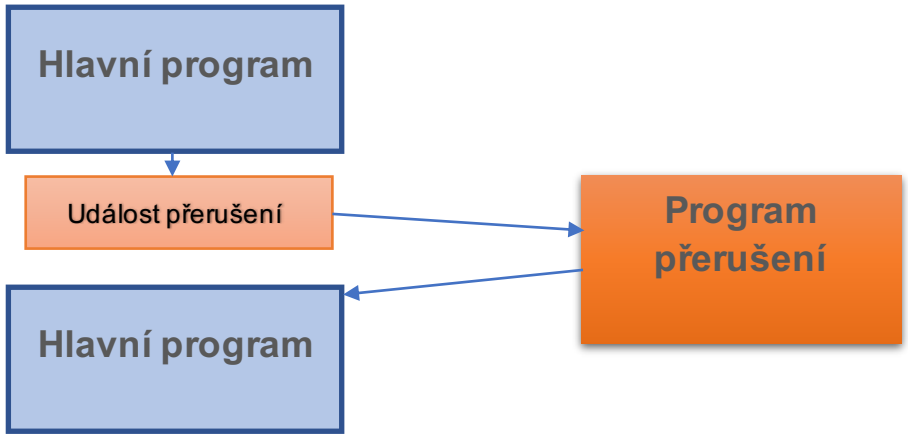
VĚTŠINA Z VÁS DENNĚ PŘI CESTĚ DO ŠKOLY, NA NÁKUPY ATD. POTKÁVÁ SVĚTELNÉ KŘIŽOVATKY. JISTĚ ČASTO MÁTE POCIT, ŽE INTERVALY SVITU ČERVENÉ JSOU DLOUHÉ A NAOPAK INTERVALY SVITU ZELENÉ ABNORMÁLNĚ KRÁTKÉ. V TÉTO KAPITOLE SI SESTROJÍTE MODEL KŘIŽOVATKY, KTEROU BUDETE ŘÍDIT POMOCÍ ARDUINA.

Co se naučíte

1. Princip semaforu
2. Jak fungují světelné křižovatky
3. Co je to přerušení a jak jej použít

Co je to přerušení

Přerušení si můžeme představit nezávisle běžící program, čekající na nějakou událost. Zde je to stisk tlačítka. Pokud tato situace nastane, pak hlavní program, přenechá výpočetní prostředky pro nezbytnou dobu programu pro obsluhu přerušení, ten vykoná, co potřebuje a opět vše vrátí hlavnímu programu.



Arduino zná tyto typy přerušení:

Typ	Význam
LOW	Přerušení je voláno, pokud na daném pinu není signál (logická 0).
CHANGE	Přerušení je voláno, pokud na daném pinu dojde ke změně signálu (změna logické 0 na 1 nebo naopak).
RISING	Přerušení je voláno, pokud na daný pin přijde signál (změna z logické 0 na 1).
FALLING	Přerušení je voláno, pokud na daném pinu je signál ukončen (změna z logické 1 na 0).

Pár otázek

- **Jak se mění stavy semaforu pro řidiče?**
  Jednotlivé stavy jsou: červená, červená a oranžová (žlutá), zelená, oranžová, žlutá. Doba, po kterou se semafor nachází v konkrétním stavu se může význa mně měnit, dle důležitosti směru, hustoty provozu atd.
- **Uveďte příklady, kde se dá použít přerušení?**
- **Který typ přerušení podporovaného Arduinem, zde můžeme použít?**

  De facto kterýkoliv, rozdíl by byl pouze v zapojení tlačítka. V našich příkladech po užíváme RISING – hlídající stisk tlačítka.

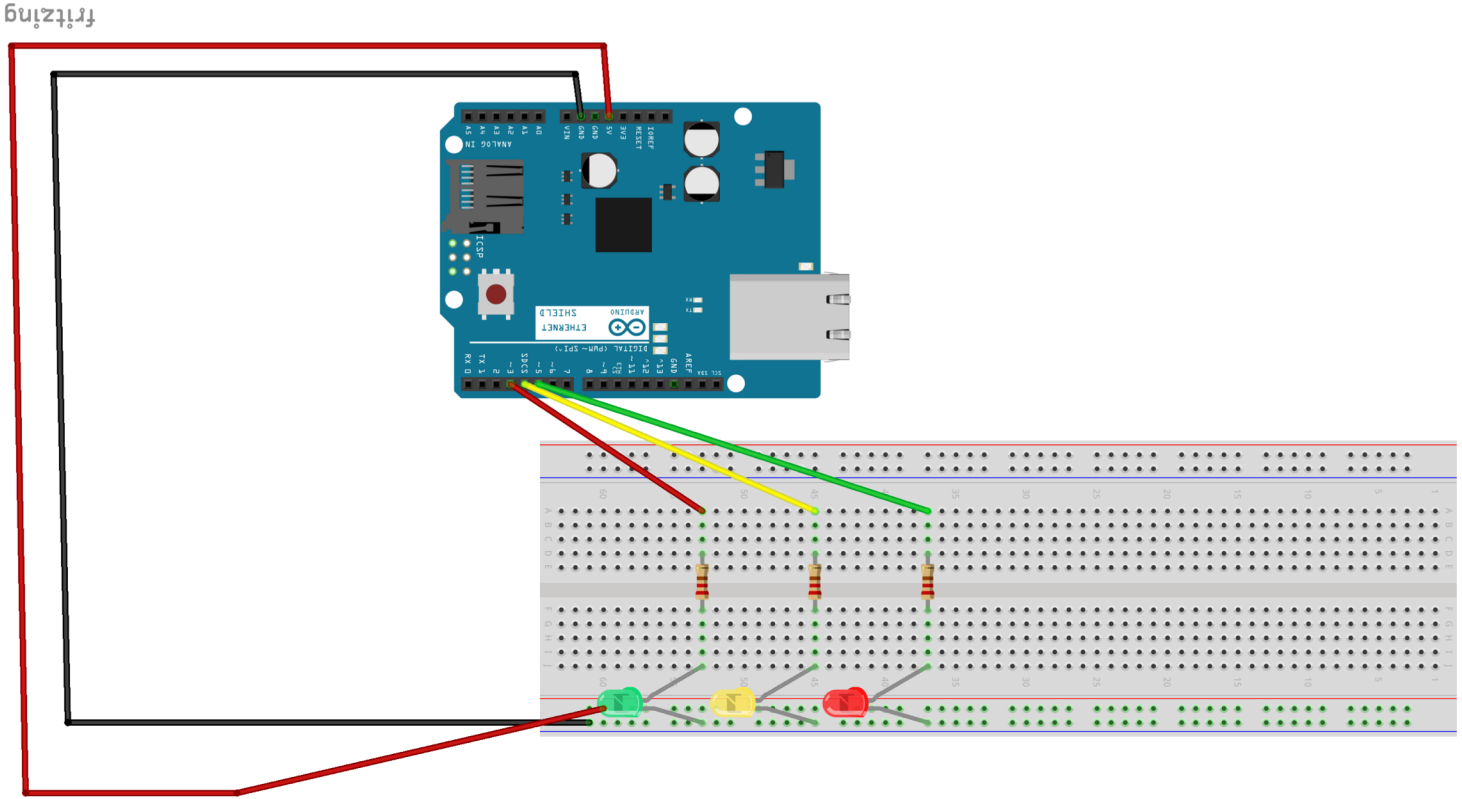
Sestavení obvodu

Co budeme potřebovat?

- LED diody (2 x červenou, 2 x zelenou, 1x žlutou, 1 x modrou).
- Tlačítko
- Arduino
- Kontaktní pole
- Odpory 220 Ω (6x) a 10 kΩ (1x)
- Vodiče typu samec-samec

Elektronický obvod

Schéma zapojení



Programový kód

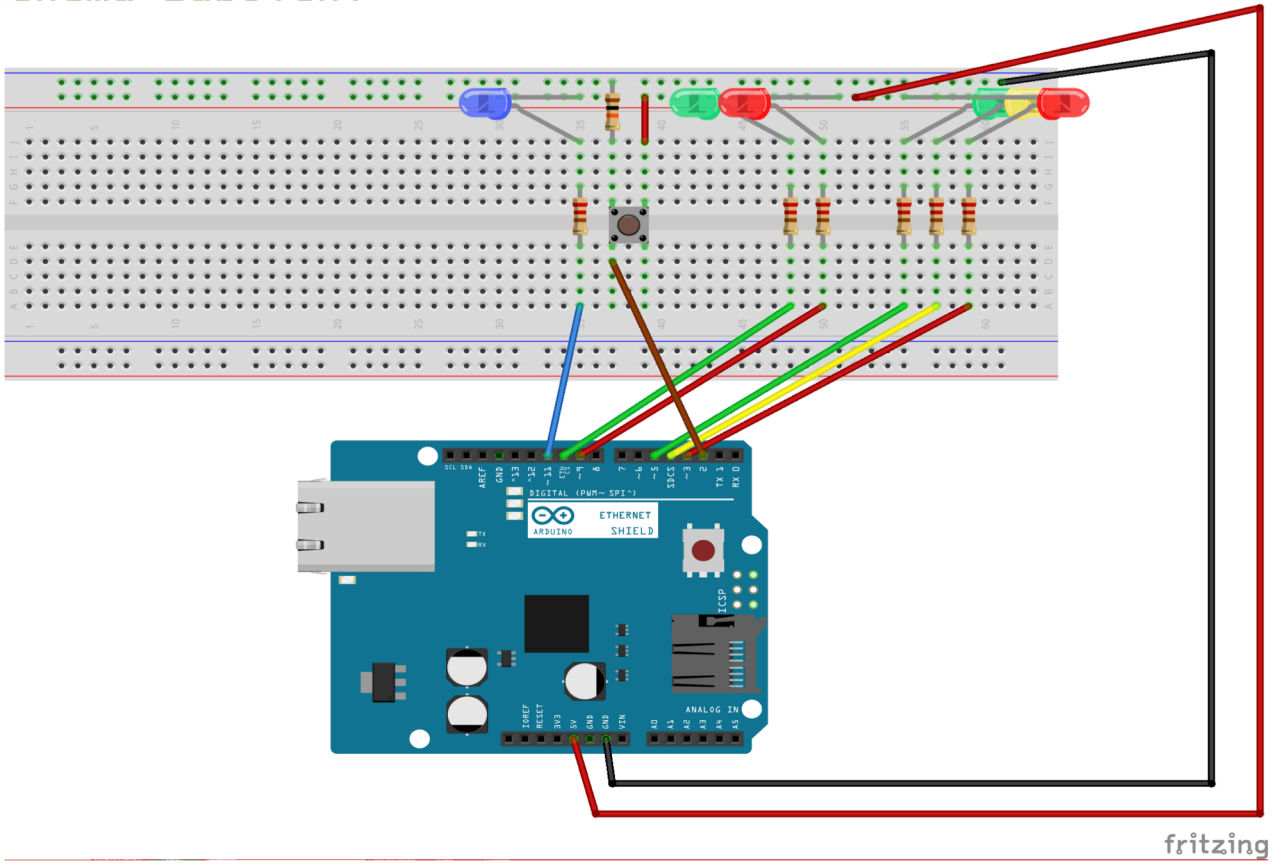
```
int cervena1=3;
int oranzova1=4;
int zelena1=5;

void setup() {
  pinMode(cervena1, OUTPUT);
  pinMode(oranzova1, OUTPUT);
  pinMode(zelena1, OUTPUT);
}

void loop() {
  digitalWrite(cervena1,HIGH);
  delay(1000);
  digitalWrite(oranzova1,HIGH);
  delay(1000);
  digitalWrite(cervena1,LOW);
  digitalWrite(oranzova1,LOW);
  digitalWrite(zelena1,HIGH);
  delay(2000);
  digitalWrite(zelena1,LOW);
  digitalWrite(oranzova1,HIGH);
  delay(1000);
  digitalWrite(oranzova1,LOW);
  digitalWrite(cervena1,HIGH);
  delay(1000);
}
```

Úkoly pro vás

1. Pokud vše funguje měli byste před sebou mít fungující semafor.
2. Můžete experimentovat s dobou svícení jednotlivých světel.
3. Až bude vše fungovat, jak si představujete, postupujte dál.
4. Upravte schéma zapojení:

Elektronický obvod

Schéma zapojení



Programový kód 1

```
int prepinac=2;
int tlacitko = 0;
int cervena1=3;
int oranzova1=4;
int zelena1=5;
int cervena3=9;
int zelena3=10;
int modra=11; //kontrolni dioda pro chodce

void setup() {
  pinMode(prepinac, INPUT);
  pinMode(cervena1, OUTPUT);
  pinMode(oranzova1, OUTPUT);
  pinMode(zelena1, OUTPUT);
  pinMode(cervena3, OUTPUT);
  pinMode(zelena3, OUTPUT);
  pinMode(modra, OUTPUT);
  digitalWrite(zelena1, HIGH);
  digitalWrite(cervena3, HIGH);
  attachInterrupt(digitalPinToInterrupt(prepinac),     zmena, RISING);
}
```

Programový kód 2

```
void loop() {
  delay(2000);
  if (tlacitko)
    {
      digitalWrite(zelena1, LOW);
      digitalWrite(oranzova1, HIGH);
      delay(1000);
      digitalWrite(oranzova1, LOW);
      digitalWrite(cervena1, HIGH);
      delay(500);
      digitalWrite(zelena3, HIGH);
      digitalWrite(cervena3, LOW);
      digitalWrite(modra,LOW);
      tlacitko=0;
      delay(2000);
      digitalWrite(zelena3, LOW);
digitalWrite(oranzova1, HIGH);
      digitalWrite(cervena3, HIGH);
      delay(1000);
      digitalWrite(cervena1, LOW);
      digitalWrite(oranzova1, LOW);
      digitalWrite(zelena1, HIGH);
    }
}
void zmena(){
  tlacitko=1;
  digitalWrite(modra, HIGH);
}
```

Vysvětlení
A úkoly

- Asi nejdůležitější (a nové) pro vás v tomto případě je přerušení a jeho obsluha.
- Přerušení se nastavuje pomocí funkce *attachInterrupt* v části setup.
- Samotná obsluha přerušení je ve funkci *zmena*. Všimněte si, že jedinné co tato funkce udělá, je že při stisku tlačítka změní hodnotu proměnné. Dle její hodnot y pak program pozná, zda tlačítko bylo od minulého průchodu stisklé.

ÚKOLY PRO VÁS

- A) Přemýšlejte, jak by bylo možné naprogramovat tuto úlohu bez použití přer ušení.
- Která možnost je jednodušší
- Zkuste vymyslet další případy, kde lze s úspěchem použít přerušení.

Programový kód

```
void loop() {
  delay(2000);
  if (tlacitko)
    {
      digitalWrite(zelena1, LOW);
      digitalWrite(oranzova1, HIGH);
      delay(1000);
      digitalWrite(oranzova1, LOW);
      digitalWrite(cervena1, HIGH);
      delay(500);
      digitalWrite(zelena3, HIGH);
      digitalWrite(cervena3, LOW);
      digitalWrite(modra,LOW);
      tlacitko=0;
      delay(2000);
      digitalWrite(zelena3, LOW);
digitalWrite(oranzova1, HIGH);
      digitalWrite(cervena3, HIGH);
      delay(1000);
      digitalWrite(cervena1, LOW);
      digitalWrite(oranzova1, LOW);
      digitalWrite(zelena1, HIGH);
    }
}
void zmena(){
  tlacitko=1;
  digitalWrite(modra, HIGH);
}
```