Střední průmyslová škola elektrotechnická, Havířov, Czech

Srednja poklicna in tehniška šola, Murska Sobota, Slovenia

Crossroad
(CZ)

Co-funded by the
Erasmus+ Programme
of the European Union

Student: Pavel Veselka
Tutor: Ing. Milada Bajtková

Student: Aljoša Mlinaric
Tutor: Ing. Rajko Palatin

Obsah

1 Úvod

Tento projekt je o semaforech, které tvoří celou křižovatku, o problémech které mohou nastat a o nedostatku použití logických funkcí, které se vyskytují u většiny současných křižovatek. Jeden z hlavních problémů, je nedostatek pinů na Arduinu, které vysílají signál pro spuštění LED diod. Vybral jsem si zrovna tento projekt, vzhledem k jeho jednoduchému, přesto inovativnímu potenciálu.

2 Teoretická Část

2.1 Analýza Trhu

V této kategorii již existuje spousta funkčních programů, či zpracování, například na fórech Arduina či Tinkercadu. K jejím výhodám patří, že jsou jednoduché, fungující, většinou jen semafory. K jejím nevýhodám patří, že nemají přidány žádné logické funkce. Náš projekt je rozdílný právě v nich.

2.2 Technická Analýza

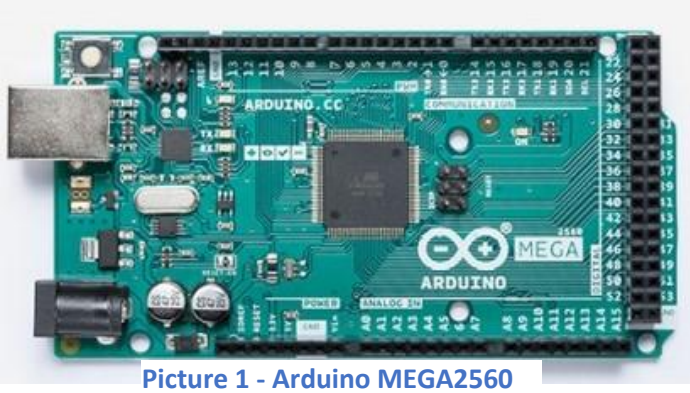
Naše křižovatka, by měla fungovat právě jako reálná křižovatka, až na to, že v naší budou přidány logické moduly, jako světelné senzory, či fungující tlačítka pro chodce a tak dále. Ovšem, musíme podotknout, že řidiči musí striktně dodržovat pravidla silničního provozu, jako na každé křižovatce. Velká část problému, je nejvíce používaná funkce v Arduinu, funkce delay(), která prakticky zastaví všechno dění ve smyčce Arduina, až na příkaz, který probíhal předtím, než se funkce delay() volala a Arduino poté není schopno vykonávat cokoli jiného. Další z problémů jsou tlačítka pro chodce, vzhledem k omezenému počtu pinů pro přerušení na Arduinu. (režimy pro přerušení jsou v referencích pod číslem 5.2)

Mega, Mega2560, MegaADK	2, 3, 18, 19, 20, 21

Picture 0 - interrupt pinypro MEGA2560

2.2.1 Použité technologie

Použili jsme světelný senzor TEMT6000, který používáme jako detektor „noci“,který - když zde není dostatek světla – pomůže rozsvítit světla na přechodu, aby byl i v noci viditelný.Používáme Arduino typ MEGA2560, protože je největší a tudíž nabízí nejvyšší počet pinů z dostupných desek. Dále jsme pomocí 3D tiskárny vytiskli ochranné obaly pro semafory.



Picture 1 - Arduino MEGA2560

2.3 Finanční analýza

Název	Číslo prodejce	Cena za kus (Kč)	Počet kusů	Cena celkem
Arduino Mega2560	A000067	1070,6	1	1070,6
LED, Yellow, 1cm	L-813YD	7,29	16	116,64
LED, Red, 1cm	L-813SRD-C	8,32	24	199,68
LED, Green, 1cm	L-813GD	7,55	24	181,2
LED, White, 3mm	VLHW4100	12,53	12	150,36
TEMT6000, phototransistor	1502184137	79	1	79
			Celkem:	1797,48

3 Praktická část

3.1 Elektronika

TEMT6000 (v referencích pod číslem 5.1), je senzor, který používáme pro skenování okolního světla, je NPN fototranzistor v malé, průhledné formě, ke snadnému pájení na DPS. Tato součást je senzitivní k viditelnému spektru. Jeho další vlastnost je adaptace vůči citlivosti lidského oka. Může být použit u mobilních telefonů, notebooků, PDA a podobně.

3.2 Programy

Zhotovil jsem program pro senzor TEMT6000 (v referencích pod číslem 5.3), který posílá hodnotu v luxech na vstup Arduina, jenž se ukazuje v konzoli. Tato funkce, kterou jsem pojmenoval reading() nevrací hodnotu (void). Kdykoliv, když naskenována úroveň světla pod nebo rovno 80, světla pro chodce na přechodech se rozsvítí.

```
void reading() {
  if(previousReadMillis + timerreading < millis()){
  Serial.print("Surrounding light: ");
  Serial.println(analogRead(sensor));
  }
  if (analogRead(sensor) <= 80) {
    digitalWrite(crossinglights, HIGH);
  }
  else {
    digitalWrite(crossinglights, LOW);
  }
}
```

Picture 2 –funkce voidreading()

Poté, přímo pro semafory jsem přišel na to, že mohou nastat jen 4 různé režimy, které jsem orientoval podle středních pruhů. První – střed má zelenou pro auta a červenou pro chodce, takže strany mají červenou pro auta a zelenou pro chodce.

```
if(lightstatus == 0){
     digitalWrite(greenm, HIGH);
     digitalWrite(greens, LOW);
     digitalWrite(redm, LOW);
     digitalWrite(reds, HIGH);
     digitalWrite(yellowm, LOW);
     digitalWrite(yellows, LOW);
     digitalWrite(pedredm, HIGH);
     digitalWrite(pedgreenm, LOW);
     digitalWrite(pedreds, LOW);
     digitalWrite(pedgreens, HIGH);
 }
```

Picture 3–první režim

Druhý – chodci na středu chtějí přejít přes cestu nebo již uběhl čas určený pro auta na středu, aby měli zelenou. Tudíž se spustí „mezi-režim“ zpomalování pro auta na středu, takže auta na středu dostanou žlutou a auta na straně dostanou červenou a žlutou. Všichni chodci mají červenou.

```
else if(lightstatus == 1){
     digitalWrite(greenm, LOW);
     digitalWrite(greens, LOW);
     digitalWrite(redm, LOW);
     digitalWrite(reds, HIGH);
     digitalWrite(yellowm, HIGH);
     digitalWrite(yellows, HIGH);
     digitalWrite(pedredm, HIGH);
     digitalWrite(pedgreenm, LOW);
     digitalWrite(pedreds, HIGH);
     digitalWrite(pedgreens, LOW);
 }
```

Picture 4–druhý režim

Třetí – „mezi-režim“ stop začal pro auta na středu a tudíž auta na stranách majíauta zelenou a chodci na středu nyní mohou projít, jelikož auta na středu mají červenou.

```
else if(lightstatus == 2){
     digitalWrite(greenm, LOW);
     digitalWrite(greens, HIGH);
     digitalWrite(redm, HIGH);
     digitalWrite(reds, LOW);
     digitalWrite(yellowm, LOW);
     digitalWrite(yellows, LOW);
     digitalWrite(pedredm, LOW);
     digitalWrite(pedgreenm, HIGH);
     digitalWrite(pedreds, HIGH);
     digitalWrite(pedgreens, LOW);
 }
```

Picture 5–třetí režim

Čtvrtý –časovač pro chodce na středu skončil, takže „mezi-režim“ zpomalování začne pro auta na straně a „mezi-režim“ příprava pro auta na středu začne. Všichni chodci v tomto režimu mají červenou.

```
else if(lightstatus == 3){
     digitalWrite(greenm, LOW);
     digitalWrite(greens, LOW);
     digitalWrite(redm, HIGH);
     digitalWrite(reds, LOW);
     digitalWrite(yellowm, HIGH);
     digitalWrite(yellows, HIGH);
     digitalWrite(pedredm, LOW);
     digitalWrite(pedgreenm, LOW);
     digitalWrite(pedredm, LOW);
     digitalWrite(pedgreenm, LOW);
 }
```

Picture 6–čtvrtý režim

Všechny tyto režimy jsou pod funkci setlight(intlightstatus), kde lightstatus je integer, který je nastaven stisknutím tlačítek pro chodce, či automaticky hlavní smyčkou. Časovače pro každý „mezi-režim“ jsou nastaveny na začátku programu.

```
#define timermainteance 1000
#define timerslow 1000
#define timerstop 5000
#define timerprepare 2000
#define timerreading 1500
```

Picture 7–časovač pro “mezi-režimy” a sensor

Konstanta timerreading udává, jak často bude světelný senzor TEMT6000 skenovat okolní světlo.

Konstanta timermainteance udává, jak často budou blikat žluté LED diody, během údržby, nebo kdykoliv vejde křižovatka do režimu „noc“.

```
void mainteancemode(bool blinkm) {
  mainteance = !mainteance;
  if(mainteance == LOW){
    mainteance = !blinkm(0);
  }
}
```

Picture 8–režim pro tlačítko “údržba”

Poslední důležitá funkce programu je funkce blinkm, která je definovaná jako bool, jelikož vrací pouze hodnotu true nebo false. Udává, jak bude režim „údržba“ fungovat.

```
bool blinkm(bool blinkstatus) {
    digitalWrite(redm, LOW);
    digitalWrite(reds LOW);
    digitalWrite(greenm, LOW);
    digitalWrite(greens, LOW);
    digitalWrite(pedredm, LOW);
    digitalWrite(pedgreenm, LOW);
    digitalWrite(pedreds, LOW);
    digitalWrite(pedgreens, LOW);
    digitalWrite(yellowm, blinkstatus);
    digitalWrite(yellows, blinkstatus);
    Serial.print(mainteance);
    Serial.println(blinkstatus);
    return !blinkstatus;
}
```

Picture 9–funkce blinkm

A ve finále, část smyčky, která ovládá režim údržby, bude vypadat takto.

```
if(mainteance == true){
  if(previousMillis + timermainteance < millis()){
    blinkstatus = blinkm(blinkstatus);
    previousMillis = millis();
  }
}
```

Picture 10–režim “udržba”

Tlačítka pro chodce musela být vyřešena přes přerušení, stejně jako tlačítko pro údržbu, jako režim pro ně jsem zvolil RISING, který spustí funkci pouze tehdy, když hodnota stoupá, tudíž, když se tlačítko stiskne, abych se vyhnul možnosti, že by se zaregistrovalo kliknutí dvakrát.

```
attachInterrupt(digitalPinToInterrupt(buttonmiddle), pedestrianlightmiddle, RISING);
attachInterrupt(digitalPinToInterrupt(buttonside), pedestrianlightside, RISING);
attachInterrupt(digitalPinToInterrupt(mainteancebutton), mainteancemode, RISING);
```

Picture 11–nastavení pinů pro přerušení

3.2.1 Náhrada funkce delay

Jak jsem řekl v teoretické části, jeden z největších problému byl s funkcí delay, kterou jsem používal pro fungování semaforů, ale pokaždé když jsem ji volal, Arduino tzv. zamrzlo a nebylo schopno vykonávat cokoliv jiného do té doby, než funkce delay skončila. Jako řešení jsem použil funkci millis(), která pro fungování místo funkce delay potřebuje definovat časovače (picture 11) a také potřebuje deklarovat pomocné proměnné (picture 12). Počet těchto funkcí závisí na počtu volání funkce millis(). (časovače v obrázku 11 jsou orientovány podle středních pruhů)

```
#define timermainteance 1000  // yellow blinking
#define timerslow 1000  // from green to yellow
#define timerstop 5000 // from yellow to red and wait
#define timerprepare 2000 // from red to red and yellow
#define timerreading 1500 // how often the sensor will read the values
```

Picture 12–časovače pro hlavní režimy

```
long previousLightMillis; // Millis function help
long previousReadMillis;  // Millis function help
long previousMillis; // timer for the crossings
```

Picture 13–pomocné proměnné pro millis()

3.3 Mechanika

Pro úplné dokončení modelu, jsme vytiskli kryty na semafory a na světla pro chodce.

3.4 Testování

Kromě testování na nepájivém poli pro základní výpočty, jsme testovali na pre-final modelu, který ještě neměl vytisklé kryty. Po prvním testování se našlo pár chyb v kódu, které bylo potřeba napravit, ale trvalo to jenom hodinu. Během testování jsme také přišli na pár vylepšení, jak upravit celý model a jak funguje.

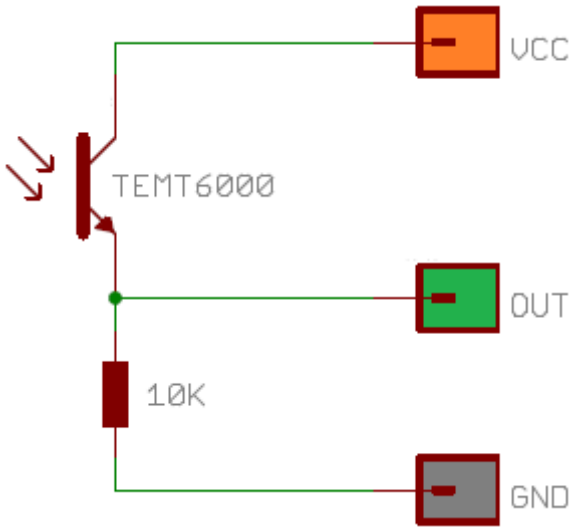
4 Závěr

Projekt byl úspěšný. Na jeho samém začátku, jsem takřkanevěděl jak programovat v Arduinu nebo cokoliv ohledně toho. Nyní jsem na pokročilé úrovni a moje angličtina se zlepšila díky spolupráci se Slovinskými studenty. Za zmínku také stojí pochopení spousty algoritmů. Spolupráce se Slovinským studentem nebyla výborná, ale viděl jsem postupem času spoustu zlepšení v jeho práci. Hodnotím naši práci kladně, jelikož jsem viděl a naučil se spoustu věcí během této spolupráce.

Plánuji pokračovat v práci na tomto projektu do té doby, než vymyslím pár dalších logických implementací pro klasickou křižovatku.

5 Reference

5.1 Světelný Senzor



5.2 Režimy Přerušení

- **LOW** to trigger the interrupt whenever the pin is low,
- **CHANGE** to trigger the interrupt whenever the pin changes value
- **RISING** to trigger when the pin goes from low to high,
- **FALLING** for when the pin goes from high to low.

5.3 Logika funkce voidreading();