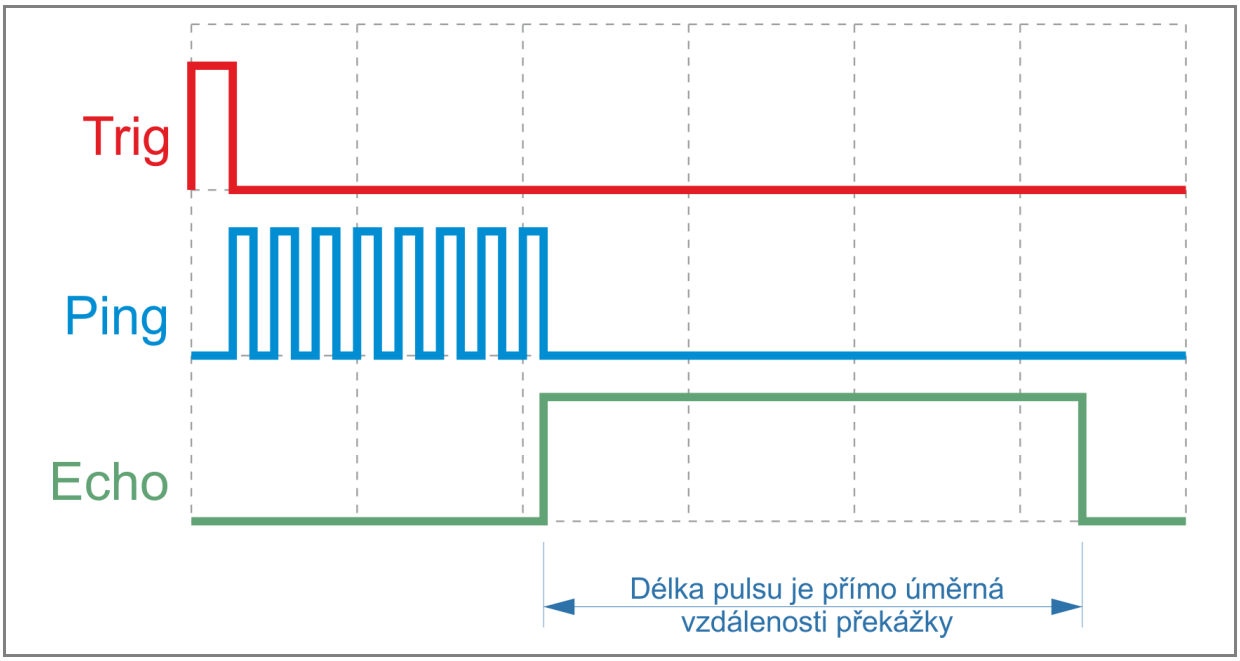
Měření vzdálenosti ultrazvukem

Změřit bezkontaktně vzdálenost tak, aby výsledek měření nebyl příliš závislý na okolních podmínkách není úplně snadný úkol. K takovému účelu se používají – mimo jiné – také ultrazvukové dálkoměry. Ostatně většina moderních aut je vybavena několika těmito senzory, které používá jejich parkovací asistent. Bohužel tyhle dálkoměry jsou přizpůsobeny konstrukci vozidla, takže komunikují po speciálních sběrnicích a v amatérské praxi se s nimi pracuje celkem obtížně.

Sonar HC-SR04

Naštěstí je na trhu sonarový modul HC-SR04 a jeho různé klony, který je snadno použitelný ve spolupráci s jednoduchými mikrokontroléry typu Arduina a obecně i se všemi dalšími mikrokontrolery s napájením napětím 5 V. Modul se skládá z ultrazvukového vysílače, přijímače a vyhodnocovacích obvodů, které zajišťují funkčnost a jednoduchou obsluhu.
HC-SR04 se používá tam, kde potřebujeme s průměrnou přesností zkoumat prostor před senzorem, což bývá často například u amatérských konstrukcí mobilních robotů.

Ovládání sonaru



Ovládání sonaru

Sonar se spouští sestupnou hranou impulzu, přivedeného na pin Trig. Po skončení tohoto pulzu vyšle sonar ultrazvukový impulz (Ping), nastaví pin Echo z log.0 na log.1 a začne „naslouchat“ odraženému signálu. Jakmile je ozvěna (echo) vyslaného impulzu přijata, je pin Echo nastaven zpět na log.0. Doba trvání tohoto impulzu v čase je tedy přímo úměrná vzdálenosti překážky.

Výpočet vzdálenosti

Převod doby trvání výstupního impulzu sonaru na vzdálenost je už matematickou záležitostí.

Rychlost šíření zvukové vlny ve vzduchu teplém 20° C je 343 m/s. Vynásobením doby trvání impulzu v mikrosekundách (µs) konstantou 0,0343 (343 / 10 000) dostaneme vzdálenost v centimetrech, kterou zvuková vlna urazila k překážce a zpět.

A právě kvůli tomu „a zpět“ musí být výsledek výpočtu ještě dělen dvěma, protože nás zajímá jen doba, která trvala zvukové vlně, než dorazila k překážce.

Vzorec pro výpočet vzdálenosti překážky zní:

```
vzdálenost = délka_impulzu * 0,0343 / 2
```

tedy:

```
vzdálenost = délka_impulzu * 0,01715
```

Převrácenou hodnotou čísla 0,01715 je po zaokrouhlení číslo 58,31, takže vzorec můžeme zjednodušit na:

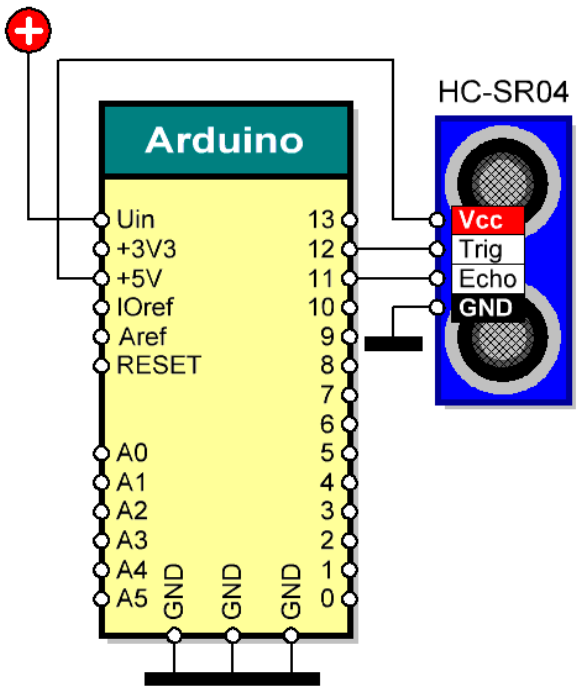
vzdálenost = délka_impulzu / 58,31

Pokud ovšem potřebujete měřit ultrazvukem vzdálenost při výrazně jiné teplotě či dokonce v jiné atmosféře, budete si muset ověřit rychlost šíření zvuku pro daný případ ve fyzikálních tabulkách a výpočet vzdálenosti podle toho patřičně upravit.

Praktická část

V praktická části si ukážeme, jak připojit laciný ultrazvukový dálkoměr HC-SR04 k Arduinu a programově ho obsloužit.

Zapojení



Sonar HC-SR04 je s Arduinem spojen čtyřmi vodiči. Na pin VCC se připojuje napájecí napětí +5V, na pin GND nulový potenciál zdroj (zem), pin Trig je spojen s pinem D12 a pin Echo s D11[1].

[1] *Vycházíme z defaultního nastavení knihovny NewPing (viz dále). Použít však můžete samozřejmě kterékoli datové piny.*

Ovládací programy

Níže uvedený ukázkový kód pro mikrokontroler Arduino nejprve určí, které piny se budou používat pro ovládání senzoru a pak inicializuje proměné.
Funkcí `setup()` nastaví potřebné parametry.
Vlastní program se skládá ze tří funkcí:

- funkce `start()` zajistí korektní obsluhu pinu Trig.
- ve funkci `vypocti()` se nejprve změří doba trvání pulzu na pinu Echo, pak se tento údaj přepočte na vzdálenost překážky v centimetrech a uloží do proměnné vzdalenost.
- funkce `zobraz()` odesílá naměřený údaj po sériové lince a je ho možno zobrazit libovolným sériovým monitorem.

Ptáte se asi, proč je tak jednoduchý program rozdělen do několika funkcí.
Má to několik důvodů:

- didaktický. Má začínajícího programátora naučit správným návykům.
- zjednodušení pochopení programu. Začátečník není nucen zkoumat, které části kódu k sobě patří, protože ve funkci jsou jednoznačně vymezeny složenými závorkami.
- možnost „růstu“ programu, kdy je možno se na už odladěné části kódu spolehnout a použít je v jiných programech.

Základní program

```
// UZ dálkoměr

// piny pro připojení signálů Trig a Echo
const int pTrig = 12;
const int pEcho = 11;
// inicializace proměnných, do kterých se uloží data
long odezva, vzdalenost;

void setup()
{
  // nastavení pinu pTrig na výstup
  pinMode(pTrig, OUTPUT);

  // nastavení pinu pEcho na vstup
  pinMode(pEcho, INPUT);

  // výstup pTrig nastavíme na log.0 (pro jistotu)
  digitalWrite(pTrig, LOW);
  // povolení komunikace sériovou linkou rychlostí 115200 baud
  Serial.begin(115200);
```

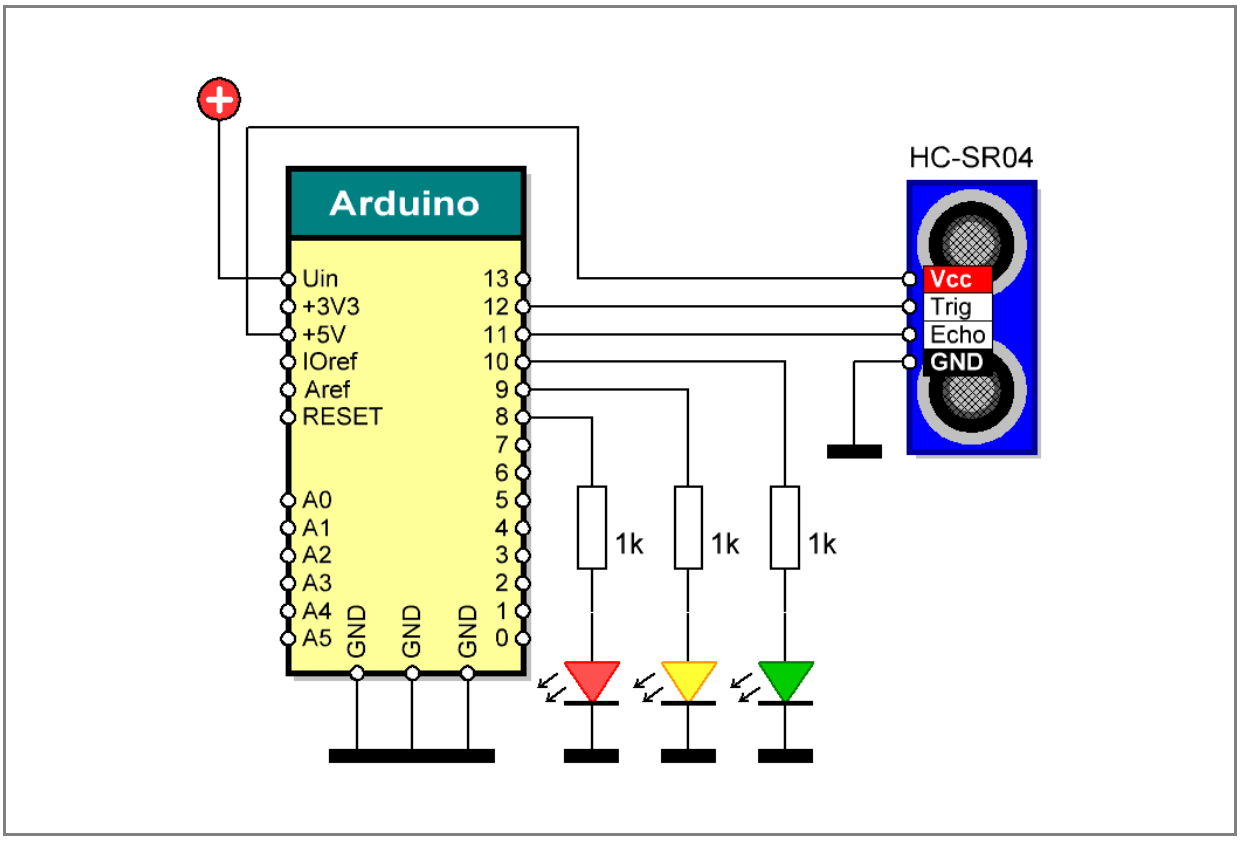
```
}

void loop()
{
  start();
  vypocti();
  zobraz();
}

void start()
// funkce generuje spouštěcí (TRIG) signál pro ultrazvukový senzor
{
  digitalWrite(pTrig, HIGH);  // výstup nastavíme na log.1
  delayMicroseconds(10);  // počkáme 10 mikrosekund
  digitalWrite(pTrig, LOW);  // a pak ho vrátíme ho zpět na log.0
}

void vypocti()
// funkce přepočítává délku přijatého pulzu na vzdálenost v centimetrech
{
  // funkce pulseIn vrátí délku přijatého pulzu v mikrosekundách (us)
  odezva = pulseIn(pEcho, HIGH);
  // přepočet délky pulzu na vzdálenost v cm
  vzdalenost = odezva / 58.31;
}

void zobraz()
// funkce odesílá naměřené hodnoty sériovou linkou
{
  Serial.print("Vzdálenost překážky je ");
  Serial.print(vzdalenost);
  Serial.println(" cm");
  // pauza 0,5 sekundy zajistí přehledné čtení
  delay(500);
}
```

A takto vypadají data, odeslaná po sériové lince:

```
Vzdálenost překážky je 8 cm
Vzdálenost překážky je 55 cm
Vzdálenost překážky je 58 cm
Vzdálenost překážky je 18 cm
Vzdálenost překážky je 556 cm
Vzdálenost překážky je 3 cm
Vzdálenost překážky je 813 cm
Vzdálenost překážky je 4 cm
```

Sonar HC-SR04 neměří příliš přesně na vzdálenost větší než 2 metry a tak se může stát, že občas dostanememe nesmyslná data (viz výpis). S touto skutečností je tedy nutno počítat a případně ji programově ošetřit.

Tři LED - meze



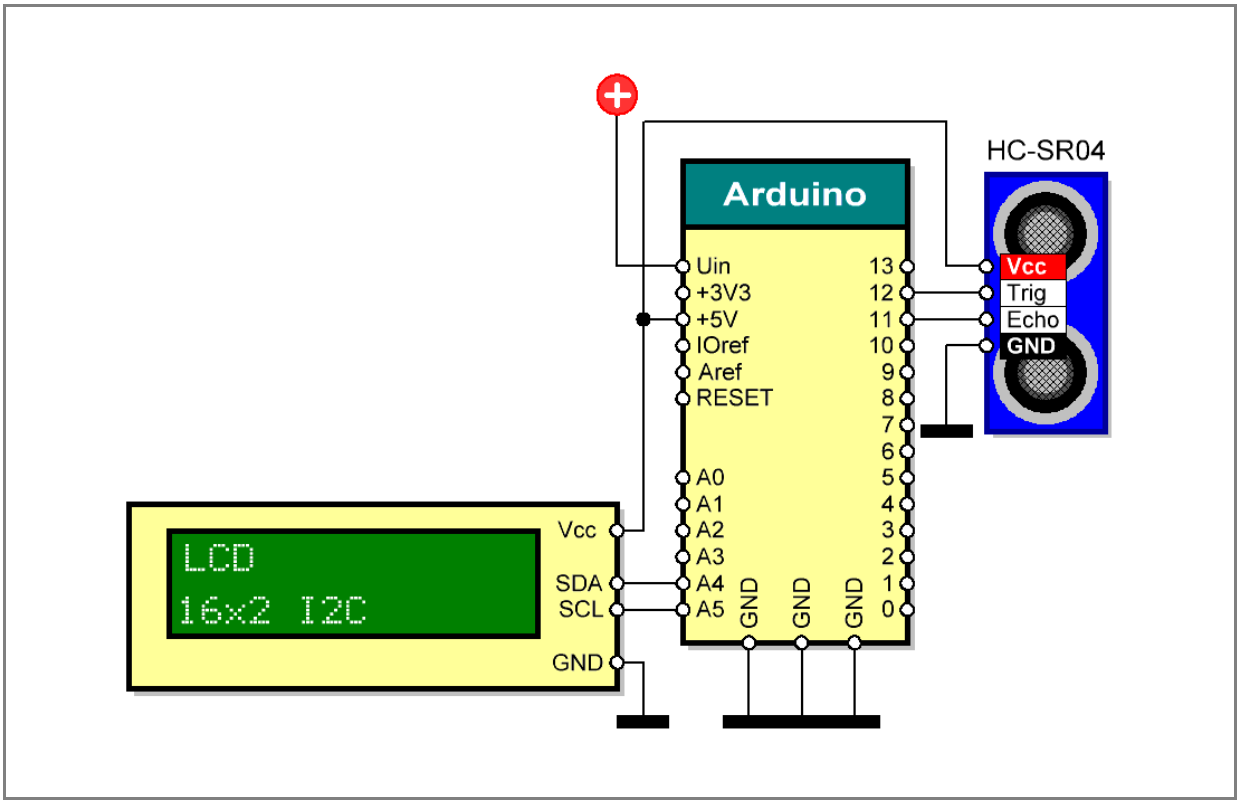
V tomto zapojení přibyly tři LED s předřadnými rezistory, které znázorňují, jak vzdálená je překážka. Význam je stejný jako na dopravním semaforu: zelená – volno, žlutá – pozor, červená - překážka

```
/*
 HC-SR04 ULTRAZVUKOVÝ SENZOR
*/
// piny pro připojení signálů Trig a Echo
const int pTrig = 12;
const int pEcho = 11;

// piny pro připojení LED
const int ledMin = 8;  // červena LED
const int ledMid = 9;  // žlutá LED
const int ledMax = 10;  // zelena LED

// konstanty
const int maximal = 50;  // maximální vzdálenost v centimetrech
const int minimal = 6;  // minimální vzdálenost v centimetrech
```

```
// inicializace proměnných, do kterých se uloží data
long odezva, vzdalenost;

void setup()
{
  // nastavení pinu pTrig na výstup
  pinMode(pTrig, OUTPUT);

  // nastavení pinu pEcho na vstup
  pinMode(pEcho, INPUT);

  // pin pTrig na log.0 (pro jistotu)
  digitalWrite(pTrig, LOW);

  pinMode(ledMin, OUTPUT);
  pinMode(ledMid, OUTPUT);
  pinMode(ledMax, OUTPUT);
  zhasniVse();
}

void loop()
{
  start();
  vypocti();
  rozhodni();
}

void start()
// funkce generuje spouštěcí (TRIG) signál pro ultrazvukový senzor
{
  digitalWrite(pTrig, HIGH);  // výstup nastavíme na log.1
  delayMicroseconds(10);  // počkáme 10 mikrosekund
  digitalWrite(pTrig, LOW);  // a pak ho vrátíme ho zpět na log.0
}

void vypocti()
// funkce přepočítává délku přijatého pulzu na vzdálenost v centimetrech
{
  // funkce pulseIn vrátí délku přijatého pulzu v mikrosekundách (us)
  odezva = pulseIn(pEcho, HIGH);
  // přepočet délky pulzu na vzdálenost v cm
  vzdalenost = odezva / 58.31;
}

void rozhodni()
// funkce rozhoduje, která LED bude svítit
{
```

```
  if(vzdalenost <= minimal)
  {
    zhasniVse();
    digitalWrite(ledMin,HIGH);
  }
  else if(vzdalenost >= minimal && vzdalenost <= maximal)
  {
    zhasniVse();
    digitalWrite(ledMid,HIGH);
  }
  else if(vzdalenost >= minimal)
  {
    zhasniVse();
    digitalWrite(ledMax,HIGH);
  }
  delay(500);
}

void zhasniVse()
{
  digitalWrite(ledMin,LOW);
  digitalWrite(ledMid,LOW);
  digitalWrite(ledMax,LOW);
}
```

Zobrazení vzdálenosti na LCD displeji



Toto zapojení používá k zobrazení vzdálenosti překážky dvouřádkový LCD displej se 16 znaky na řádku, připojený k Arduinu přes komunikační linku I2C.

Program bez použití knihovny NewPing:

```
//import knihovny LiquidCrystal_I2C
#include <LiquidCrystal_I2C.h>

// nastavena adresa 0x3F a pořadí pinů displeje na převodníku:
// en,rw,rs,d4,d5,d6,d7,backlight,backlight polarity
LiquidCrystal_I2C lcd_1(0x3F, 2, 1, 0, 4, 5, 6, 7, 3, POSITIVE);

// piny pro připojení signálů Trig a Echo
const int pTrig = 12;
const int pEcho = 11;

// inicializace proměnných, do kterých se uloží data
long odezva, vzdalenost;

void setup()
{
  // pin pTrig nastaven jako výstupní
  pinMode(pTrig, OUTPUT);

  // pin pEcho nastaven jako vstupní
  pinMode(pEcho, INPUT);

  // výstup pTrig nastaven na log.0 (pro jistotu)
  digitalWrite(pTrig, LOW);

  // formát displeje nastaven na 16x2
  lcd.begin(16, 2);

  // podsviceni zapnuto
  lcd.backlight();
}

void loop()
{
  start();
  vypocti();
  if (vzdalenost < 250)
  {
    zobrazNaLCD();
  }
  else
  {
    Err();
```

```
  }
}

void start()
// funkce generuje spouštěcí (TRIG) signál pro ultrazvukový senzor
{
  digitalWrite(pTrig, HIGH);  // výstup nastavíme na log.1
  delayMicroseconds(10);  // počkáme 10 mikrosekund
  digitalWrite(pTrig, LOW);  // a pak ho vrátíme ho zpět na log.0
}

void vypocti()
// funkce přepočítává délku přijatého pulzu na vzdálenost v centimetrech
{
  // funkce pulseIn vrátí délku přijatého pulzu v mikrosekundách (us)
  odezva = pulseIn(pEcho, HIGH);
  // přepočet délky pulzu na vzdálenost v cm
  vzdalenost = odezva / 58.31;
}

void zobrazNaLCD()
// tato funkce zajistí odeslání naměřené hodnoty na LCD displej
{
  lcd_1.print("Distance: ");
  lcd_1.print(vzdalenost);
  lcd_1.print(" cm");
  // pauza 1 sekunda pro přehledné čtení
  delay(1000);
  // smazání displeje
  lcd_1.clear();
}

void Err()
// tato funkce zobrazí na LCD displeji chybovou hlášku
{
  lcd_1.print("MIMO ROZSAH!");
  // pauza 1 sekunda pro přehledné čtení
  delay(1000);
  // smazání displeje
  lcd_1.clear();
}
```

Totéž s použitím knihovny NewPing[2]:

```
#include <NewPing.h>
```

[2] *Viz příručka knihovny NewPing*

```
const int TRIGGER_PIN = 12;
const int ECHO_PIN = 11;
const int MAX_DISTANCE = 200;

NewPing sonar1(TRIGGER_PIN, ECHO_PIN, MAX_DISTANCE);

void setup()
{
  Serial.begin(115200);
}

void loop()
{
  delay(500);
  Serial.print("Vzdálenost překážky je ");
  Serial.print(sonar1.ping_cm());
  Serial.println(" cm");
}
```