7. MATICOVÝ LED DISPLEJ

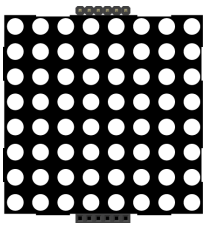
V TÉTO LEKCI SE ZAMĚŘÍME NA PRAKTICKÉ VYUŽITÍ MATICOVÉHO LED DISPLEJE. SEZNÁMÍME SE JAK TENTO DISPLEJ ZAPOJIT A JEJ TAKÉ PROGRAMOVAT. TYTO ZDÁNLIVĚ JEDNODUCHÉ DISPLEJE MAJÍ I V DNEŠNÍ DOBĚ SVÉ VYUŽITÍ PRO SVOU ČITELNOST, TECHNICKOU NENÁROČNOST A POMĚRNĚ SNADNÉ PROGRAMOVÁNÍ.

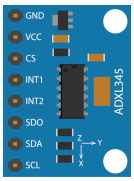
CÍLE

① Zapojení maticového displeje 8x8.
② Programování maticového displeje ro zobrazení jednoduchých symbolů.
③ Zapojení a programování maticového displeje ve spojení s potenciometry.
④ Pro zručné studenty je určen úkol pro naprogramování hry Ping-Pong.
⑤ Seznámení s akcelerometrem.
⑥ Programování akcelerometru ve spojení s maticovým displejem.

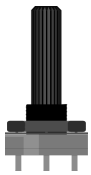
Čas: **4x45 min**
Úroveň: ■■■□□

Vychází z: **3, 4, 5**


Maticový displej 8x8



Akcelerometr


Potenciometr 2x

POUŽITÉ SOUČÁSTKY

PRŮVODCE HODINOU I

Studenti sestaví prakticky jediný obvod, do kterého budou v dalších hodinách pouze přidávat další komponenty. V této hodině bude hlavním úkolem pochopit princip maticového displeje. V programu využijí již získané vědomosti týkající se polí a jejich procházení. Součástí jsou jednoduché samostatné úkoly.

PŘÍPRAVA

Co bude v této hodině potřeba?

⑦ Součásti obvodu – deska Arduino s USB kabelem, kontaktní pole, maticový LED displej 8x8, vodiče.
⑧ Osobní počítač pro studenty s nainstalovaným Arduino IDE.
⑨ Pokud je k dispozici, tak dataprojektor.
⑩ Prezentace k lekci 7.
⑪ Pracovní listy pro studenty.

1. KROK 5 minut

Na úvod rozdejte studentům sady Arduino. Řekněte, že náplní vašeho kurzu bude si ukázat praktické možnosti využití maticového LED displeje.

ZEPTEJTE SE STUDENTŮ

➔ **Kde jste se setkali s maticovým displejem?**
Např. ve veřejné dopravě, venkovní reklamě, na stadionu při zobrazení výsledků.

➔ **V čem byste spatřovali výhody maticového displeje?**
Jednoduchost, čitelnost, cena.

Studenti ať nejdříve zapojí displej pro jeho otestování. Zapojení je velice jednoduché. Ať využijí zobrazeného schématu, které je součástí pracovních listů nebo přiložené prezentace, kterou lze promítat pomocí dataprojektoru.

2. KROK 10 minut

Nyní studentům ukažte prostřednictvím dataprojektoru nebo pracovního listu základní kód, který zajistí blikání jediné diody na displeji.

RYCHLÝ TIP

➔ **Vysvětlete, princip maticového displeje tak, že se jedná o samostatné LED diody, které jsou vzájemně propojeny. Využijte k tomu přiložené schéma a tabulku zapojených pinů.**

```
int pinA=2;
int pinB=6;

void setup() {
  pinMode(pinA,OUTPUT);
  pinMode(pinB,OUTPUT);
  digitalWrite(pinA,HIGH);
  digitalWrite(pinB,HIGH);
}

void loop() {
  digitalWrite(pinB,LOW);
  delay(200);
  digitalWrite(pinB,HIGH);
  delay(200);
}
```

Studenti ať program nahrají do desky a odzkouší, zda se dioda v horním levém rohu rozbliká.

RYCHLÝ TIP

➔ **Nezapomeňte, že při každé změně se musí program opět nahrát do desky.**

3. KROK 20 minut

K objasnění principu maticového displeje ať studenti zkusí vyřešit následující úkol.

ÚKOL PRO STUDENTY

➔ **A) Upravte obvod zapojení displeje a programový kód předchozího příkladu tak, aby blikaly i diody ve všech rozích stejně jako dioda první.** Řešení je tohoto úkolu spočívá v zapojení odpovídajících výstupů displeje do desky Arduino.

4. KROK 15 minut

Pro stejné zapojení displeje z předchozího příkladu ať studenti vyřeší následující úkol.

ÚKOL PRO STUDENTY

➔ **B) Změňte programový kód předchozího příkladu tak, aby diody v protilehlých rozích blikali střídavě.**

PRACOVNÍ LIST – MATICOVÝ DISPLEJ

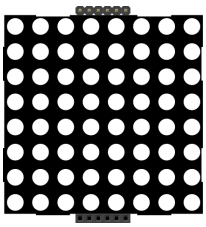
LED MATICOVÝ DISPLEJ JE SOUČÁSTKA, KTERÁ OBSAHUJE DIODY USPOŘÁDANÉ DO SLOUPCŮ A ŘÁDKŮ. TVOŘÍ DVOJROZMĚRNÉ POLE A VELIKOST ZÁVISÍ NA POČTU DIOD. NEJČASTĚJI JSOU POUŽÍVANÉ DISPLEJE O VELIKOSTI 8X8.

CO SE NAUČÍTE

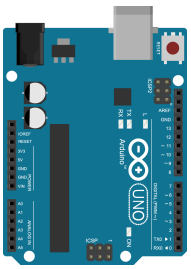
① Zapojit maticový displej.
② Zopakujete si zapojování LED diod.
③ Programovat maticový displej.

CO BUDETE POTŘEBOVAT

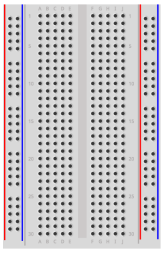
① Maticový displej.
② Desku Arduino.
③ Kontaktní pole.
④ Vodiče typu zásuvka-zásuvka.



Maticový displej 8x8



Deska Arduino



Kontaktní pole

POUŽITÉ SOUČÁSTKY

OTÁZKY PRO VÁS

➔ **Kde jste se setkali s maticovým displejem?**

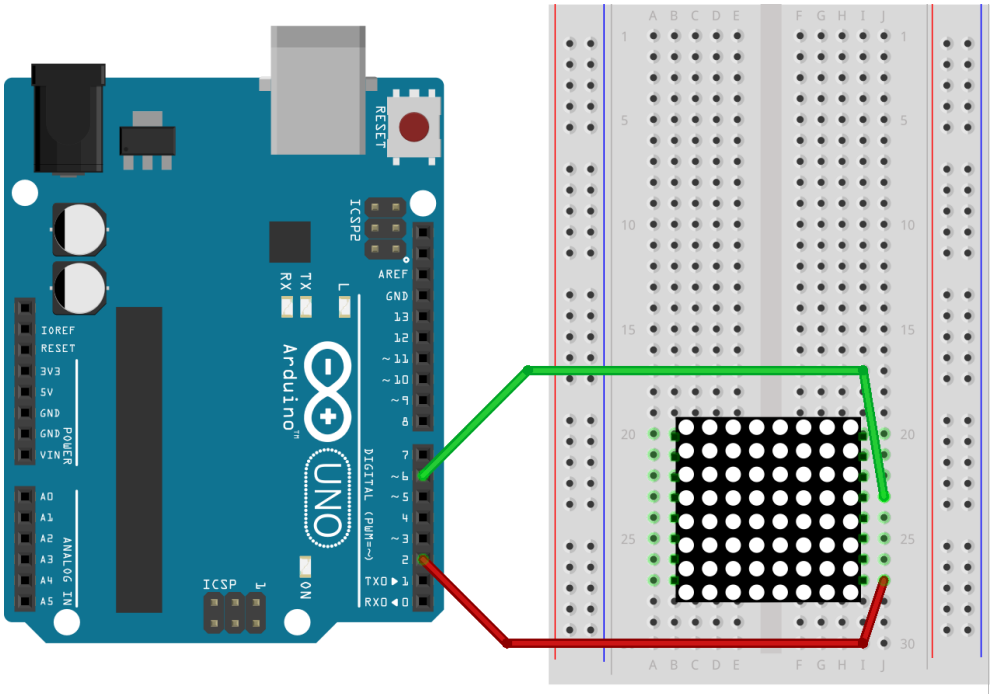
➔ **V čem byste spatřovali výhody maticového displeje?**

DEJTE SI POZOR

➔ **Pozor si dejte při vkládání displeje do kontaktního pole. Dejte pozor, abyste nožičky displeje zbytečně neohnuli. Všimněte si, že displej je na středu pole. Tím jsou jeho kontakty odděleny.**

A JDĚTE NA TO ...

① Podle přiloženého schématu zapojte obvod s maticovým displejem.



② Napište program, který rozbliká první diodu displeje.

```
int pinA=2;
int pinB=6;

void setup() {
  pinMode(pinA,OUTPUT);
  pinMode(pinB,OUTPUT);
  digitalWrite(pinA,HIGH);
  digitalWrite(pinB,HIGH);
}

void loop() {
  digitalWrite(pinB,LOW);
  delay(200);
  digitalWrite(pinB,HIGH);
  delay(200);
}
```

ÚKOL PRO VÁS

➔ **A) Upravte obvod zapojení displeje a programový kód předchozího příkladu tak, aby blikaly i diody ve všech rozích stejně jako dioda první.**

③ Pokud jste úkoly splnili a vše funguje, jak má zkuste si v rámci dalšího úkolu zapojit a naprogramovat ještě jeden obvod.

ÚKOL PRO VÁS

➔ **B) Změňte programový kód předchozího příkladu tak, aby diody v protilehlých rozích blikali střídavě.**

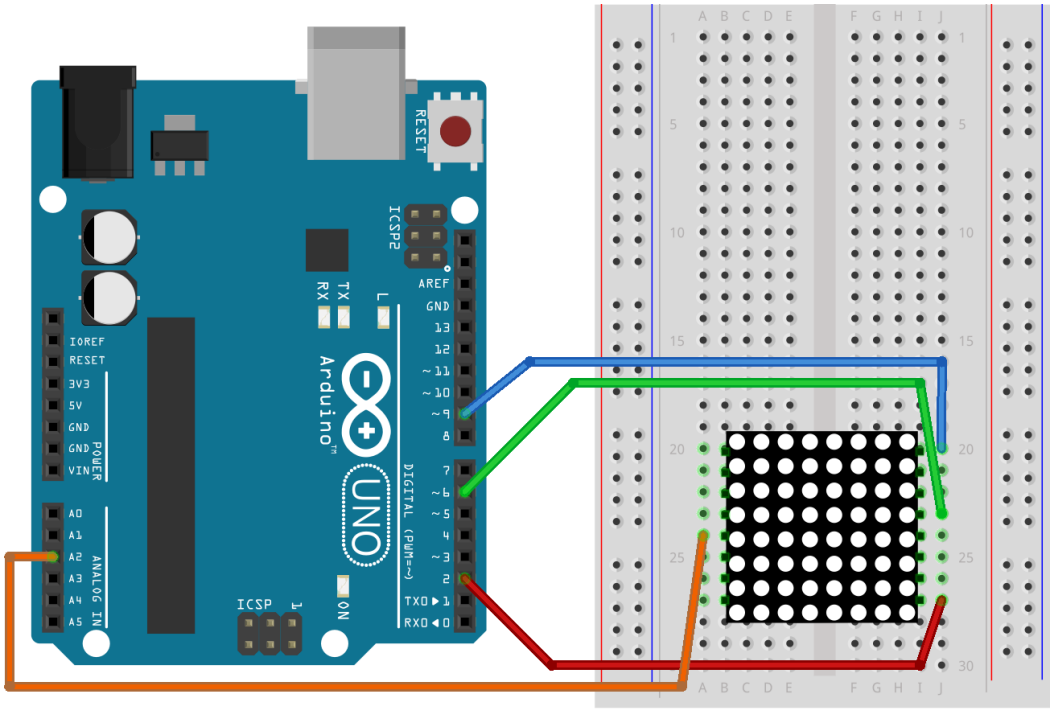
➔

NEZAPOMEŇTE

Nahrajte program do desky Arduino, kliknutím na ikonu 

ŘEŠENÍ ÚLOH

Úkol A)



```
int pinA=2;
int pinB=6;

int pinC=9;
int pinD=A2;

void setup() {
  pinMode(pinA,OUTPUT);
  pinMode(pinB,OUTPUT);
  digitalWrite(pinA,HIGH);
  digitalWrite(pinB,HIGH);

  pinMode(pinC,OUTPUT);
  pinMode(pinD,OUTPUT);
  digitalWrite(pinC,HIGH);
  digitalWrite(pinD,HIGH);

}

void loop() {
  digitalWrite(pinB,LOW);
  digitalWrite(pinD,HIGH);
  delay(200);

  digitalWrite(pinB,LOW);
  digitalWrite(pinD,HIGH);
  delay(200);
}
```

Úkol B)

```
int pinA=2;
int pinB=6;

int pinC=9;
int pinD=A2;

void setup() {
  pinMode(pinA,OUTPUT);
  pinMode(pinB,OUTPUT);
  digitalWrite(pinA,HIGH);
  digitalWrite(pinB,HIGH);

  pinMode(pinC,OUTPUT);
  pinMode(pinD,OUTPUT);
  digitalWrite(pinC,HIGH);
  digitalWrite(pinD,HIGH);

}

void loop() {
  digitalWrite(pinB,HIGH);  // změna na HIGH
  digitalWrite(pinD,LOW);   // změna na LOW
  delay(200);

  digitalWrite(pinB,LOW);
  digitalWrite(pinD,HIGH);
  delay(200);
}
```

PRŮVODCE HODINOU II

Tentokrát studenti budou pracovat s kompletně zapojeným maticovým displejem. Toto zapojení budou používat pro řešení několika příkladů. Naučí se zejména pracovat s vícerozměrným polem, jak jím procházet a přistupovat k hodnotám.

PŘÍPRAVA

Co bude v této hodině potřeba?

④ Součásti obvodu – deska Arduino s USB kabelem, kontaktní pole, maticový LED displej 8x8, vodiče.
⑤ Osobní počítač pro studenty s nainstalovaným Arduino IDE.
⑥ Pokud je k dispozici, tak dataprojektor.
⑦ Prezentace k lekci 7.
⑧ Pracovní listy pro studenty.

1. KROK 5 minut

Na úvod rozdejte studentům sady Arduino. Řekněte, že náplní hodiny bude si ukázat další možnosti ve využití maticového LED displeje.

RYCHLÝ TIP

➔ **Pro připomenutí ukažte studentům tabulku zapojení pinů a zeptejte se, zda by byli schopní maticový displej zapojit sami.**

2. KROK 10 minut

Ať studenti zapojí displej pro plnou funkcionalitu displeje podle přiloženého schématu v pracovním listu nebo promítaném prostřednictvím dataprojektoru.

POZOR NA ZAPOJENÍ DISPLEJE

➔ **Při zapojování displeje s větším množstvím vodičů upozorněte studenty, aby zapojení prováděli obzvláště pečlivě.**

3. KROK 10 minut

Po zapojení obvodu mohou studenti začít psát programový kód. Uvedený kód postupně rozsvěcí v každém sloupci diody. Tím dojde ke kompletnímu otestování displeje.

```
const int row[8] = {2, 7, 19, 5, 13, 18, 12, 16};
const int col[8] = {6, 11, 10, 3, 17, 4, 8, 9};

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

void loop(){
    for(int j = 0; j<8;j++) {
      digitalWrite(col[j],LOW);
      for(int k = 0;k<8;k++){
        digitalWrite(row[k],HIGH);
        delay(200);
      }
      for(int i = 0;i<8;i++){
        digitalWrite(row[i],LOW);
        digitalWrite(col[i],HIGH);
      }
    }
}
```

OTÁZKY PRO STUDENTY

- **Ať studenti po nahrátí programu do desky, jak se chovají diody na displeji.**
- **Zeptejte se, při jaké kombinaci hodnot ve funkci `digitalWrite()` dioda na displeji svítí nebo je zhasnutá?**

4. KROK 10 minut

Následující příklady upevňují znalosti týkající se principu programování maticového displeje.

ÚKOL PRO STUDENTY

- **A) Upravte (optimalizujte) programový kód tak, aby se aktualizace a mazání displeje prováděla ve dvou vámi deklarovaných funkcích.**

5. KROK 10 minut

V návaznosti na předchozí úkol, kdy by studenti měli vytvořit dvě funkce a tím tak optimalizovat kód i pro pozdější použití, stačí v následujícím úkolu provést změny v pořadí zapínání diod displeje.

ÚKOL PRO STUDENTY

- **B) Upravte programový kód tak, aby se v celém, rozsvíceném displeji postupně posouval vypnutý sloupec a při tomto vypnutém sloupci projížděl vypnutý řádek.**

PRACOVNÍ LIST – MATICOVÝ DISPLEJ - II

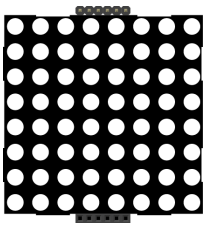
V TÉTO ČÁSTI BUDETE POKRAČOVAT V ZAPOJOVÁNÍ A PROGRAMOVÁNÍ MATICOVÉHO DISPLEJE. TENTOKRÁT SE JIŽ NAUČÍTE OVLÁDAT CELÝ DISPLEJ A VYZKOUŠÍTE SI, JAK PRACOVAT S JEDNOTLIVÝMI DIODAMI.

CO SE NAUČÍTE

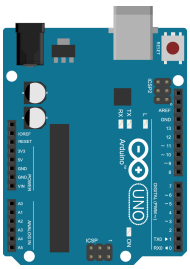
① Zapojit celý maticový displej.
② Zopakujete cyklus **FOR**.
③ Programovat průchod polem pro rozsvícení diod maticového displeje.

CO BUDETE POTŘEBOVAT

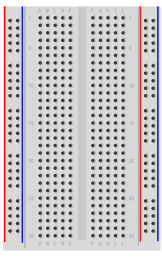
① Maticový displej.
② Desku Arduino.
③ Kontaktní pole.
④ Vodiče typu zásuvka-zásuvka.



Maticový displej 8x8



Deska Arduino



Kontaktní pole

POUŽITÉ SOUČÁSTKY

ZOPAKUJTE SI …

① Podívejte se na níže uvedenou tabulku *Tab. 1* a promyslete si, jak zapojit celý maticový displej.

Matice pin	Řádek	Sloupec	Arduino pin
1	5	-	13
2	7	-	12
3	-	2	11
4	-	3	10
5	8	-	A2
6	-	5	A3
7	6	-	A4
8	3	-	A5
9	1	-	2
10	-	4	3
11	-	6	4
12	4	-	5
13	-	1	6
14	2	-	7
15	-	7	8
16	-	8	9

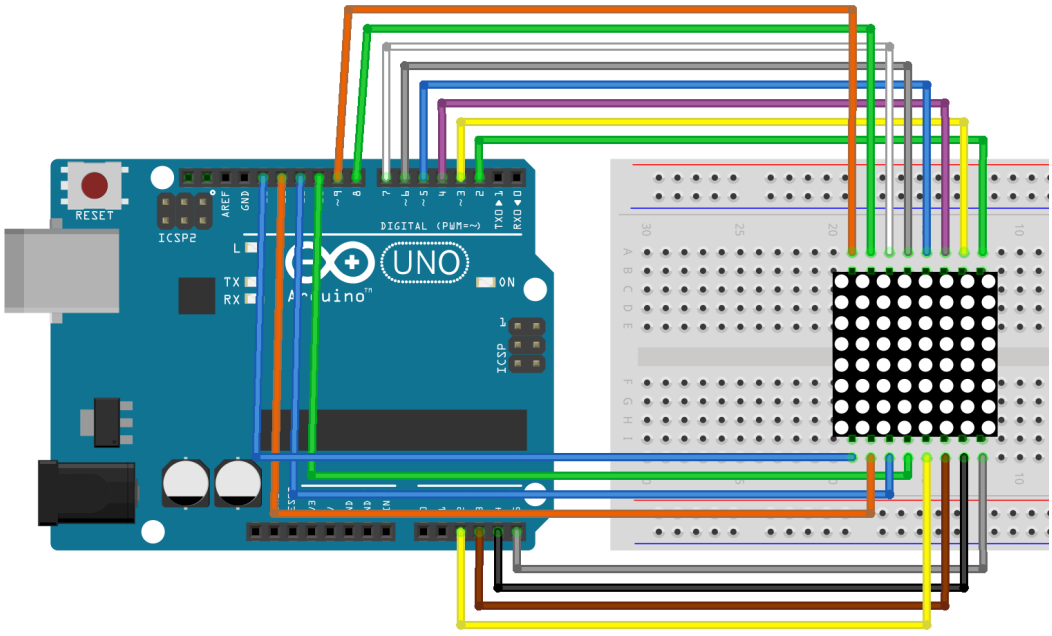
Tab. 1 - Rozložení pinů

DEJTE SI POZOR

➔ **Všimněte si pořadí pinů ve sloupci Arduino pin. Přestože se používá označení analogových vstupů A2 – A5, lze je definovat jako číselné hodnoty 14-17.**

A JDĚTE NA TO …

② Pokud si netroufnete zapojit displej podle tabulky pinů *Tab. 1*, využijte následující schéma.



DEJTE SI POZOR

➔ **Pozor si dejte při vkládání displeje do kontaktního pole. Dejte pozor, abyste nožičky displeje zbytečně neohnuli. Všimněte si, že displej je na středu pole. Tím jsou jeho kontakty odděleny.**

③ Napište a nahrajte následující program do desky Arduino.

```
const int row[8] = {
  2, 7, 19, 5, 13, 18, 12, 16
};

const int col[8] = {
```

```
  6, 11, 10, 3, 17, 4, 8, 9
};

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

void loop(){
    for(int j = 0; j<8;j++) {
      digitalWrite(col[j],LOW);
      for(int k = 0;k<8;k++){
        digitalWrite(row[k],HIGH);
        delay(200);
      }
      for(int i = 0;i<8;i++){
        digitalWrite(row[i],LOW);
        digitalWrite(col[i],HIGH);
      }
    }
}
```

OTÁZKA PRO VÁS

- **Pokud jste v pořádku nahráli program do desky, popište, jak se chovají diody na displeji.**
- **Při jaké kombinaci hodnot ve funkci `digitalWrite`() dioda na displeji svítí nebo je zhasnutá?**

④ Pokud se vám podařilo otestovat displej podle předchozího základního programu, vyřešte následující úkoly. Úkoly se týkají pouze úpravy programového kódu, není nutné měnit zapojení displeje.

ÚKOL PRO VÁS

- **A) Upravte (optimalizujte) programový kód tak, aby se aktualizace a mazání displeje prováděla ve dvou vámi deklarovaných funkcích.**
- **B) Upravte programový kód tak, aby se v celém, rozsvíceném displeji postupně posouval vypnutý sloupec a při tomto vypnutém sloupci projížděl vypnutý řádek.**

ŘEŠENÍ ÚLOH

Úkol A)

```
const int row[8] = {
  2, 7, 19, 5, 13, 18, 12, 16
};

const int col[8] = {
  6, 11, 10, 3, 17, 4, 8, 9
};

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

void loop(){
    refreshScreen();
}

void refreshScreen(){
  for(int j = 0; j<8;j++){
    digitalWrite(col[j], LOW);
    for(int k = 0; k<8; k++){
      digitalWrite(row[k], HIGH);
    }
    Clear();
  }
}

void Clear(){
  for(int i = 0; i<8; i++){
    digitalWrite(row[i],LOW);
    digitalWrite(col[i],HIGH);
  }
}
```

Úkol B)

```
const int row[8] = {
  2, 7, 19, 5, 13, 18, 12, 16
};

const int col[8] = {
  6, 11, 10, 3, 17, 4, 8, 9
};

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

void loop(){
    refreshScreen();
}

void refreshScreen(){
  for(int j = 0; j<8;j++){
    digitalWrite(row[j], LOW);
    for(int k = 0; k<8; k++){
      digitalWrite(col[k], HIGH);
      delay(100);
      digitalWrite(col[k], LOW);
    }
    digitalWrite(row[j], HIGH);
  }
}
```

PRŮVODCE HODINOU III

Studenti opět budou pracovat s kompletně zapojeným maticovým displejem. Tentokrát se naučí pracovat s vícerozměrným polem, kdy budou na displeji zobrazovat jednoduché symboly.

PŘÍPRAVA

Co bude v této hodině potřeba?

1. Součásti obvodu – deska Arduino s USB kabelem, kontaktní pole, maticový LED displej 8x8, vodiče.
2. Osobní počítač pro studenty s nainstalovaným Arduino IDE.
3. Pokud je k dispozici, tak dataprojektor.
4. Prezentace k lekci 7.
5. Pracovní listy pro studenty.

1. KROK 10 minut

Na úvod rozdejte studentům sady Arduino. Řekněte, že náplní hodiny bude si ukázat, jak se dají na maticovém displeji zobrazit jednoduché symboly.

RYCHLÝ TIP

- **Pokud mají studenti zapojený obvod s displejem z předchozí hodiny, můžete přistoupit rovnou k programovému kódu.**
- **V opačném případě si studenti v rámci opakování obvod sestaví podle schématu, který mají v pracovním listu nebo promítaného pomocí dataprojektoru.**

2. KROK 10 minut

Studenti ať napíší následující programový kód. Nevysvětlujte jim jeho princip, na to se zeptáte až jej spustí.

RYCHLÝ TIP

➔ **Ať studenti použijí některý z předchozích příkladů. Tzn. otevřou si jej v prostředí IDE a následně uloží pod novým jménem. Programový kód stačí pouze jednoduše inovovat.**

3. KROK 5 minut

Věnujte chvilku času pro vysvětlení programového kódu formou otázek.

OTÁZKY PRO STUDENTY

➔ **V čem se liší programový kód pro zobrazení symbolu od kódu z předchozích kapitol?**

➔ **Jak si myslíte že vznikl tvar srdce na displeji. Kde je nadefinován?**

4. KROK 5 minut

Studenti budou řešit následující úkol. Je velmi jednoduchý, spočívá pouze v úpravě pole `image`.

ÚKOL PRO STUDENTY

➔ **A) Upravte programový kód tak, aby se na displeji zobrazil symbol smajlíku.**

Definice tvaru symbolů je velmi snadné. Studenti mohou využít nástroj, pomocí něhož si symbol „naklikají“ a následně použijí vygenerované dvourozměrné pole vypnutých/zapnutých diod, které vloží do programového kódu.

Odkaz: https://www.prf.jcu.cz/generator-led-matrix/index.htm

5. KROK 15 minut

ÚKOL PRO STUDENTY

➔ **B) Změňte programový kód tak, aby se střídavě zobrazoval symbol velkého a malého srdce.**

PRACOVNÍ LIST – MATICOVÝ DISPLEJ - III

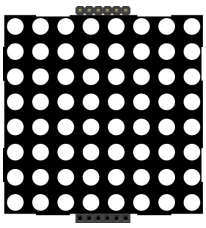
V TÉTO ČÁSTI BUDETE POKRAČOVAT ZEJMÉNA V PROGRAMOVÁNÍ MATICOVÉHO DISPLEJE. TENTOKRÁT SE NAUČÍTE PRACOVAT S VÍCEROZMĚRNÝM POLEM, POMOCÍ KTERÉHO SI ZOBRAZÍTE JEDNODUCHÉ SYMBOLY.

CO SE NAUČÍTE

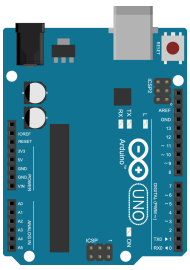
① Zopakujete si cyklus **FOR**.
② Pracovat s vícerozměrným polem.
③ Naučíte se princip zobrazování symbolů na maticovém displeji.

CO BUDETE POTŘEBOVAT

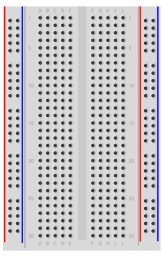
① Maticový displej.
② Desku Arduino.
③ Kontaktní pole.
④ Vodiče typu zásuvka-zásuvka.



Maticový displej 8x8



Deska Arduino

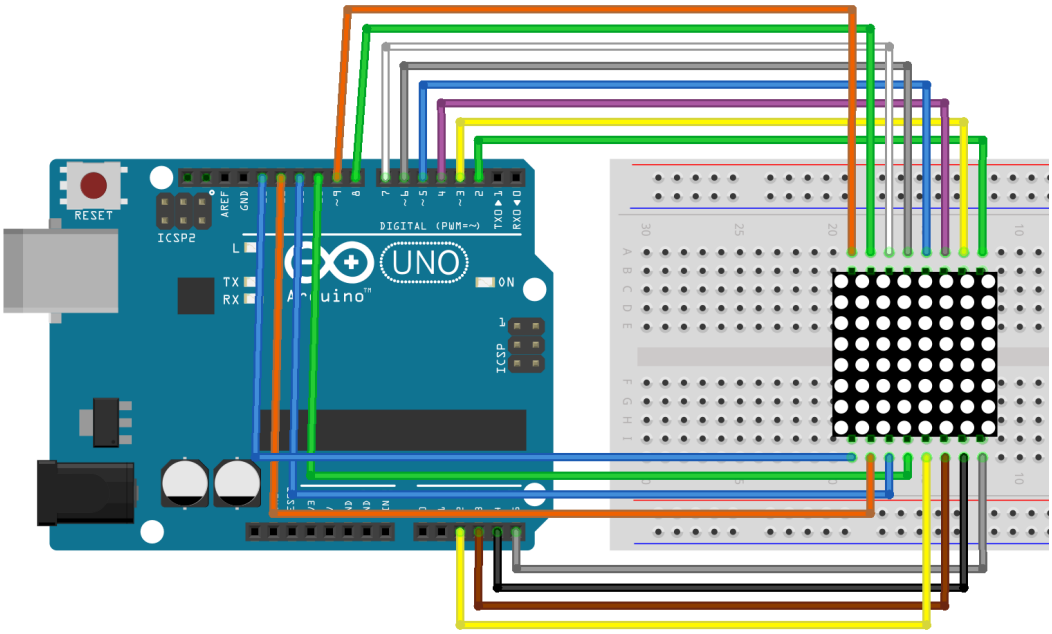


Kontaktní pole

POUŽITÉ SOUČÁSTKY

RYCHLÝ ÚVOD …

① Pokud nemáte sestavený obvod s maticovým displejem, tak v rámci opakování jej zapojte podle níže uvedeného schématu.



A JDE SE PROGRAMOVAT …

② Napište a nahrajte do desky Arduino následující programový kód.

RYCHLÝ TIP

➔ **Použijte některý z předchozích příkladů. Tzn. otevřete si jej v prostředí IDE a následně uložte pod novým jménem. Programový kód stačí pouze jednoduše inovovat.**

```
const int row[8] = {
  2, 7, 19, 5, 13, 18, 12, 16
};

const int col[8] = {
  6, 11, 10, 3, 17, 4, 8, 9
};

byte image[8][8] = {
  {0,0,0,0,0,0,0,0},
  {0,1,1,0,0,1,1,0},
  {1,0,0,1,1,0,0,1},
  {1,0,0,0,0,0,0,1},
  {1,0,0,0,0,0,0,1},
  {0,1,0,0,0,0,1,0},
  {0,0,1,0,0,1,0,0},
  {0,0,0,1,1,0,0,0}};

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

void loop(){
    refreshScreen();
}

void refreshScreen(){
  for(int j = 0; j<8;j++){
    digitalWrite(col[j], LOW);
    for(int k = 0; k<8; k++){
      digitalWrite(row[k], image[k][j]);
    }
    Clear();
  }
}
```

③ Pokud se vám podařilo nahrát do desky Arduino programový kód, zkuste si odpovědět na následující otázky.

OTÁZKY PRO VÁS

- ➔ **V čem se liší programový kód pro zobrazení symbolu od kódu z předchozích kapitol?**
- ➔ **Jak si myslíte že vznikl tvar srdce na displeji. Kde je nadefinován?**

VÝBORNĚ A JDE SE NA ÚKOLY

④ Pokud již chápete, jak se zobrazuje symbol srdce na displeji, vyřešíte následující úkol velmi rychle.

Definice tvaru symbolů je velmi snadné. Studenti mohou využít nástroj, pomocí něhož si symbol „naklikají“ a následně použijí vygenerované dvourozměrné pole vypnutých/zapnutých diod, které vloží do programového kódu.

Odkaz: https://www.prf.jcu.cz/generator-led-matrix/index.htm

ÚKOL PRO VÁS

- ➔ **A) Upravte programový kód tak, aby se na displeji zobrazil symbol smajlíku.**

A JEŠTĚ JEDEN ÚKOL

ÚKOL PRO VÁS

➔ **B) Změňte programový kód tak, aby se střídavě zobrazoval symbol velkého a malého srdce.**

ŘEŠENÍ ÚLOH

Úkol A)

```
// Srdce
byte image[8][8] = {
  {0,0,0,0,0,0,0,0},
  {0,1,1,0,0,1,1,0},
  {1,0,0,1,1,0,0,1},
  {1,0,0,0,0,0,0,1},
  {1,0,0,0,0,0,0,1},
  {0,1,0,0,0,0,1,0},
  {0,0,1,0,0,1,0,0},
  {0,0,0,1,1,0,0,0}};

// Smajlík
byte image[8][8] = {
  {0,0,1,1,1,1,0,0},
  {0,1,0,0,0,0,1,0},
  {1,0,1,0,0,1,0,1},
  {1,0,0,0,0,0,0,1},
  {1,0,1,0,0,1,0,1},
  {1,0,0,1,1,0,0,1},
  {0,1,0,0,0,0,1,0},
  {0,0,1,1,1,1,0,0}};
```

Úkol B)

```
const int row[8] = {
  2, 7, 19, 5, 13, 18, 12, 16
};

const int col[8] = {
  6, 11, 10, 3, 17, 4, 8, 9
};

// Velke srdce
byte image[8][8] = {
  {0,0,0,0,0,0,0,0},
  {0,1,1,0,0,1,1,0},
  {1,0,0,1,1,0,0,1},
  {1,0,0,0,0,0,0,1},
  {1,0,0,0,0,0,0,1},
  {0,1,0,0,0,0,1,0},
  {0,0,1,0,0,1,0,0},
  {0,0,0,1,1,0,0,0}};

// Male srdce
byte imageS[8][8] = {
  {0,0,0,0,0,0,0,0},
  {0,0,0,0,0,0,0,0},
  {0,0,0,1,0,1,0,0},
  {0,0,1,0,1,0,1,0},
  {0,0,1,0,0,0,1,0},
  {0,0,0,1,0,1,0,0},
  {0,0,0,0,1,0,0,0},
  {0,0,0,0,0,0,0,0}};

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

void loop(){
    // Zobrazeni vždy po dobu 100 iteraci
    for(int i = 0; i < 100; i++){
```

```
43          refreshScreen(image);
45       }
46
47       for(int i = 0; i < 100; i++){
48          refreshScreen(imageS);
49       }
50    }
51
52    void refreshScreen(unsigned char dat[8][8]){
53      for(int j = 0; j<8;j++){
54        digitalWrite(col[j], LOW);
55        for(int k = 0; k<8; k++){
56          digitalWrite(row[k], dat[k][j]);
57        }
58        delay(1);
59        Clear();
60      }
61    }
62
63    void Clear(){
64      for(int i = 0; i<8; i++){
65        digitalWrite(row[i],LOW);
66        digitalWrite(col[i],HIGH);
67      }
68    }
69
```

PRŮVODCE HODINOU IV

Studenti opět budou pracovat s kompletně zapojeným maticovým displejem. Tentokrát se do obvodu přidají dva potenciometry. Tím získáme dva zdroje pro analogové vstupy, které využijme pro ovládání diod na maticovém displeji.

PŘÍPRAVA

Co bude v této hodině potřeba?

① Součásti obvodu – deska Arduino s USB kabelem, kontaktní pole, maticový LED displej 8x8, vodiče, potenciometr (2x).
② Osobní počítač pro studenty s nainstalovaným Arduino IDE.
③ Pokud je k dispozici, tak dataprojektor.
④ Prezentace k lekci 7.
⑤ Pracovní listy pro studenty.

1. KROK 15 minut

Na úvod rozdejte studentům sady Arduino. Řekněte, že náplní hodiny bude si ukázat, jak spojit displej s potenciometry tak, aby se dali jeho diody ovládat. Vytvoří se ovladač displeje.

RYCHLÝ TIP

- **Pokud mají studenti zapojený obvod s displejem z předchozí hodiny, tak je dobré jej využít. Nové schéma spočívá pouze v přidání dvou potenciometrů.**
- **V opačném případě studenti musí zapojit obvod celý.**

OTÁZKA PRO STUDENTY

➔ **Zapojování potenciometrů byste již měli znát. Jak tedy zapojit potenciometry do obvodu s maticovým displeje?**
Řešení je zcela standardní a studenti se pro nápovědu mohou podívat na řeše problému do kapitoly týkající se servomotorů.

ÚKOL PRO STUDENTY

➔ **A) Zapojte oba potenciometry do obvodu s maticovým displejem. Použijte analogové vstupy na desce Arduino A0 a 11.**

2. KROK 15 minut

Ať si studenti otevřou programový kód naposledy realizované úlohy, týkající se zobrazování symbolů.

OTÁZKY PRO STUDENTY

➔ **Jak byste upravili kód, aby docházelo pomocí potenciometrů k posunu svíticí diody na displeji? Jak se čtou data z potenciometru a jakých nabývají hodnot?**
Otázky by je měli přivést k řešení. Tzn. měli by již vědět, že použijí funkci `analogRead()`. Hodnoty získané z potenciometru jsou 0-1024. Toto rozmezí hodnot se musí rozložit do 8-mi diod v ose x a y.

➔ **Jak se tyto hodnoty rozloží do 8-mi diod na displeji?**
Použije se funkce `map()`.

➔ **Když už víte, jak se čtou hodnoty z potenciometru a jak se dají rozložit do hodnot pro displej, jak byste řešili rozsvícení diody v závislosti na otočení potenciometru?**
Nejefektivnější je vytvořit funkci určenou pro čtení hodnot z potenciometru.

3. KROK 10 minut

Ukažte studentům celý programový kód, ať jej porovnají s kódem vlastním. Případně ať jej upraví a otestují.

Ve zbytku hodiny, můžete studentům ukázat ještě další aplikaci na spojení potenciometrů a maticového displeje. Je to hra pro dva PING-PONG.

Pro zručnější studenty by mohlo být naprogramování této hry i samostatným úkolem.

Programový kód hry je dostupný na GitHUB.

PRACOVNÍ LIST – MATICOVÝ DISPLEJ - IV

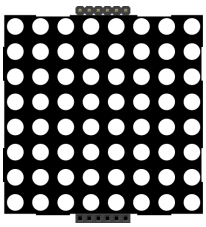
PŮVODNÍ ZAPOJENI MATICOVÉHO LED DISPLEJE ROZŠÍŘÍTE O DVA POTENCIOMETRY. TĚMITO POTENCIOMETRY BUDETE OVLÁDAT DIODY DISPLEJE NA KONKRÉTNÍCH POZICÍCH.

CO SE NAUČÍTE

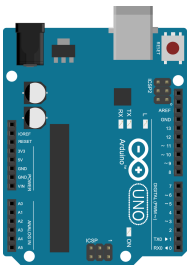
① Zopakujete si zapojení potenciometru.
② Spojení potenciometru a maticového displeje.
③ Zpracovávat hodnoty z potenciometru pro displej.

CO BUDETE POTŘEBOVAT

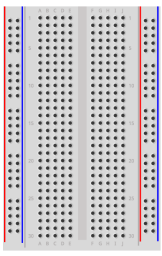
① Maticový displej.
② Desku Arduino.
③ Potenciometr – 2x
④ Kontaktní pole.
⑤ Vodiče typu zásuvka-zásuvka.



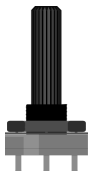
Maticový displej 8x8



Deska Arduino



Kontaktní pole

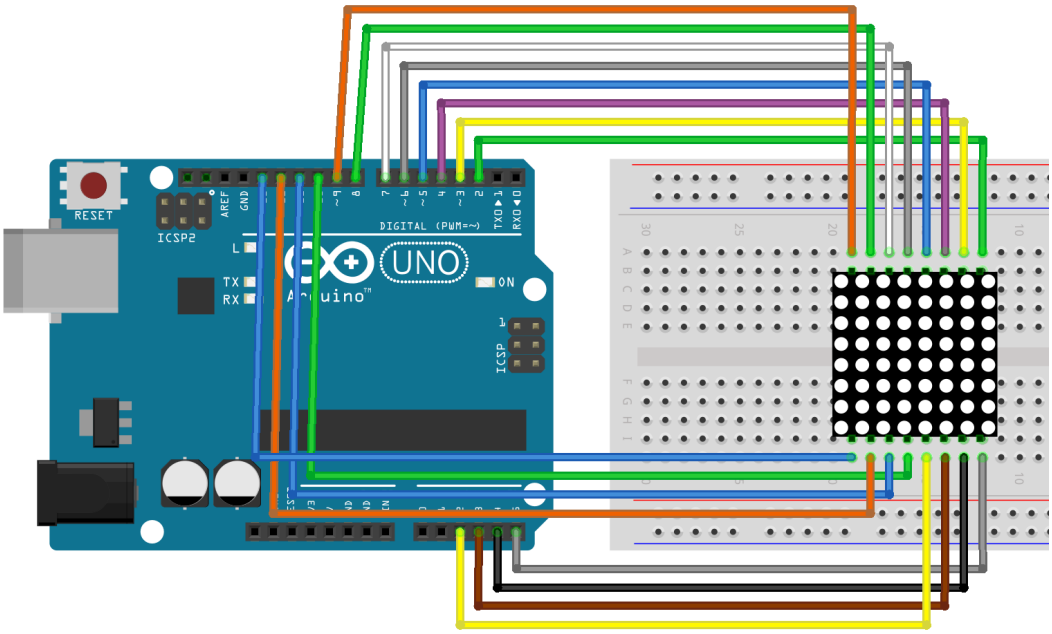


Potenciometr 2x

POUŽITÉ SOUČÁSTKY

RYCHLÝ ÚVOD ...

① Pokud nemáte sestavený obvod s maticovým displejem, tak v rámci opakování jej zapojte podle níže uvedeného schématu.



OTÁZKA PRO VÁS

➔ **Zapojování potenciometrů byste již měli znát. Jak tedy zapojit potenciometry do obvodu s maticovým displeje?**

ÚKOL PRO VÁS

➔ **A) Zapojte oba potenciometry do obvodu s maticovým displejem. Použijte analogové vstupy na desce Arduino A0 a 11.**

② Po zapojení potenciometrů, přistupte k programovému kódu. Otevřete si některý z předchozích příkladů, týkajících zobrazování symbolů. Tento příklad uložte pod novým jménem.

OTÁZKY PRO VÁS

- **Jak byste upravili kód, aby docházelo pomocí potenciometrů k posunu svíticí diody na displeji? Jak se čtou data z potenciometru a jakých nabývají hodnot?**
- **Jak se tyto hodnoty rozloží do 8-mi diod na displeji?**
- **Když už víte, jak se čtou hodnoty z potenciometru a jak se dají rozložit do hodnot pro displej, jak byste řešili rozsvícení diody v závislosti na otočení potenciometru?**

A JDE SE PROGRAMOVAT …

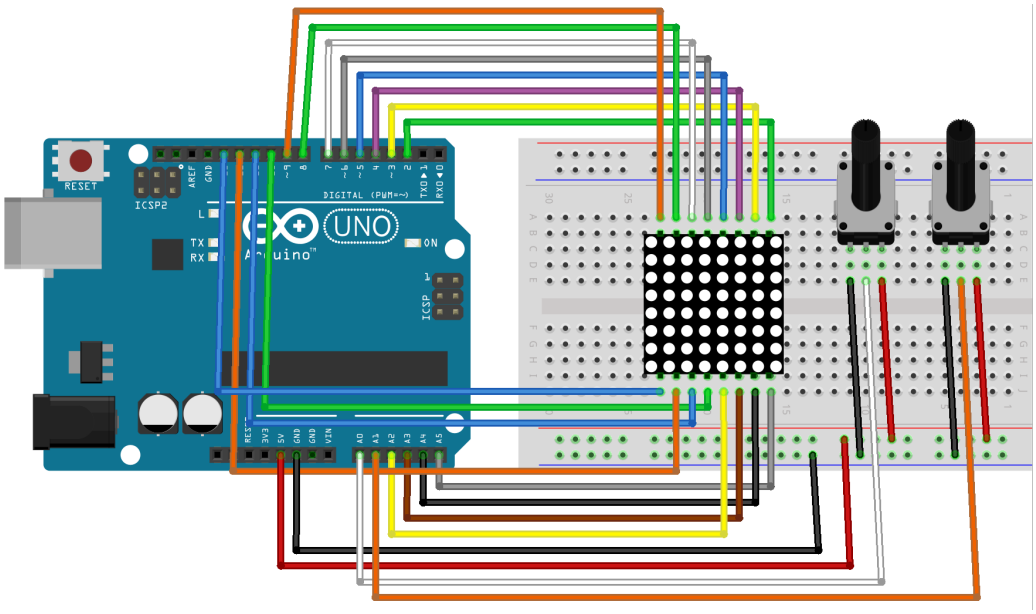
③ Zkuste porovnat vaše odpovědi z předchozích otázek, týkajících se programového kódu s přiloženým programem.

```
1  const int row[8] = {
2    2, 7, 19, 5, 13, 18, 12, 16
3  };
4
5  const int col[8] = {
6    6, 11, 10, 3, 17, 4, 8, 9
7  };
8
9  int pixels[8][8];
10
11 int x = 5;
12 int y = 5;
13
14 void setup(){
15     for(int i = 0; i < 8; i++){
16         pinMode(col[i], OUTPUT);
17         pinMode(row[i], OUTPUT);
18         digitalWrite(row[i], LOW);
```

```
    }

    for(int x = 0; x < 8; x++) {
      for(int y = 0; y < 8; y++) {
        pixels[x][y] = HIGH;
      }
    }
}

void loop(){
    readSensors();
    refreshScreen();
}

void readSensors(){
  pixels[x][y] = HIGH;
  x = 7 - map(analogRead(A0), 0, 1023, 0, 7);
  y = map(analogRead(A1), 0, 1023, 0, 7);
  pixels[x][y] = LOW;
}


void refreshScreen(){
  for(int j = 0; j<8;j++){
    digitalWrite(row[j], HIGH);
    for(int k = 0; k<8; k++){
      int thisPixel = pixels[j][k];
      digitalWrite(col[k], thisPixel);
      if (thisPixel == LOW) {
        digitalWrite(col[k], HIGH);
      }
    }
   digitalWrite(row[j], LOW);
  }
}
```

ŘEŠENÍ ÚLOH

Úkol A)



PRŮVODCE HODINOU V

Studenti opět budou pracovat s kompletně zapojeným maticovým displejem. Tentokrát se do obvodu přidá akcelerometr. Ten bude poskytovat vstupní hodnoty pro maticový displej.

PŘÍPRAVA

Co bude v této hodině potřeba?

④ Součásti obvodu – deska Arduino s USB kabelem, kontaktní pole, maticový LED displej 8x8, vodiče, akcelerometr.
⑤ Osobní počítač pro studenty s nainstalovaným Arduino IDE.
⑥ Pokud je k dispozici, tak dataprojektor.
⑦ Prezentace k lekci 7.
⑧ Pracovní listy pro studenty.

1. KROK 10 minut

Na úvod rozdejte studentům sady Arduino. Řekněte, že náplní hodiny bude si ukázat, jak spojit maticový displej s akcelerometrem.

OTÁZKY PRO STUDENTY

➔ **Víte kde se můžete setkat se zařízením akcelerometr?**
V dnešní době má akcelerometr takřka každý mobilní telefon. Dále jej nalezneme v automobilech, letadlech apod.

➔ **Víte, co akcelerometr měří?**
Měří pohybové zrychlení, a to nejlépe ve všech třech osách.

2. KROK 5 minut

Vysvětlete podrobněji princip akcelerometru.

3. KROK 10 minut

Ať studenti zapojí akcelerometr podle přiloženého nebo promítaného schématu.

RYCHLÝ TIP

- **Pokud mají studenti zapojený obvod s displejem z předchozí hodiny, tak je dobré jej využít. Nové schéma spočívá pouze v přidání dvou potenciometrů.**
- **V opačném případě studenti musí zapojit obvod celý.**

NA CO SI DÁT POZOR

- **Zaměřte se zejména na správné zapojení napájení akcelerometru a datových pinů SDA a SCL.**

4. KROK 15 minut

Studenti by měli přistoupit k programování. Opět mohou použít předchozí programový kód vztahující se k potenciometrům. Ať si studenti otevřou předchozí program a uloží jej pod novým názvem.

Pro co možná nejjednodušší programování akcelerometru ADXL 345 je vhodné použít některou z knihoven. Proto ji studenti musí na začátku programového kódu připojit. Použitá knihovna pro ADXL 345 je k dispozici na GitHub.

Ukažte studentům výpočet úhlů **roll** a **pitch**.

```
roll  = (atan2(-Yg, Zg)*180.0)/M_PI;
pitch = (atan2(Xg, sqrt(Yg*Yg + Zg*Zg))*180.0)/M_PI;
```

ÚKOL PRO STUDENTY

➔ **A) Inovujte programový kód otevřeného programu tak, abyste aplikovali uvedený vzorec pro výpočet úhlů roll a pitch.**
Tento úkol je velmi jednoduchý. Stačí upravit funkci `readSensors()`. Pro správné namapování hodnot z akcelerometru by měli studenti ověřit, jaké hodnoty poskytuje. K tomu mohou využít sériový monitor. Následně podle získaných maxim upraví funkci `map()`.

PRACOVNÍ LIST – MATICOVÝ DISPLEJ - V

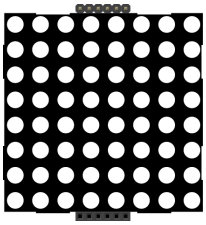
PŮVODNÍ ZAPOJENI MATICOVÉHO LED DISPLEJE ROZŠÍŘÍTE O PŘIPOJENÍ AKCELEROMETRU. V ZÁVISLOSTI NA POLOZE BUDE POSKYTOVAT DATA PRO POZICI ROZSVÍCENÉ DIODY NA MATICOVÉM DISPLEJI.

CO SE NAUČÍTE

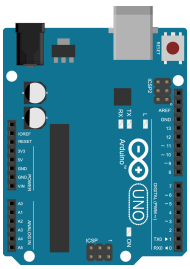
① Zapojovat akcelerometr.
② Spojení akcelerometru a maticového displeje.
③ Zpracovávat hodnoty z akcelerometru pro displej.

CO BUDETE POTŘEBOVAT

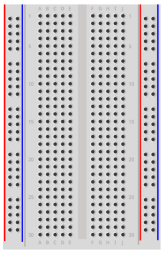
① Maticový displej.
② Desku Arduino.
③ Akcelerometr.
④ Kontaktní pole.
⑤ Vodiče typu zásuvka-zásuvka.



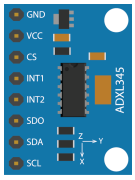
Maticový displej 8x8



Deska Arduino



Kontaktní pole

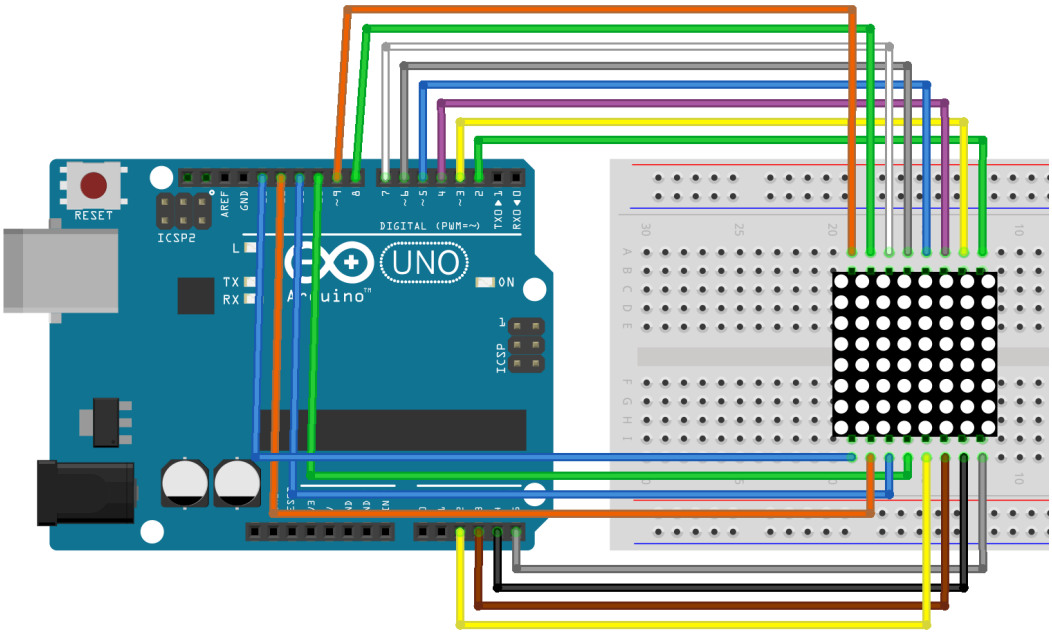


Akcelerometr

POUŽITÉ SOUČÁSTKY

RYCHLÝ ÚVOD ...

① Pokud nemáte sestavený obvod s maticovým displejem, tak v rámci opakování jej zapojte podle níže uvedeného schématu.



OTÁZKY PRO VÁS

➔ **Víte kde se můžete setkat se zařízením akcelerometr?**

➔ **Víte, co akcelerometr měří?**

KRÁTCE O AKCELEROMETRU

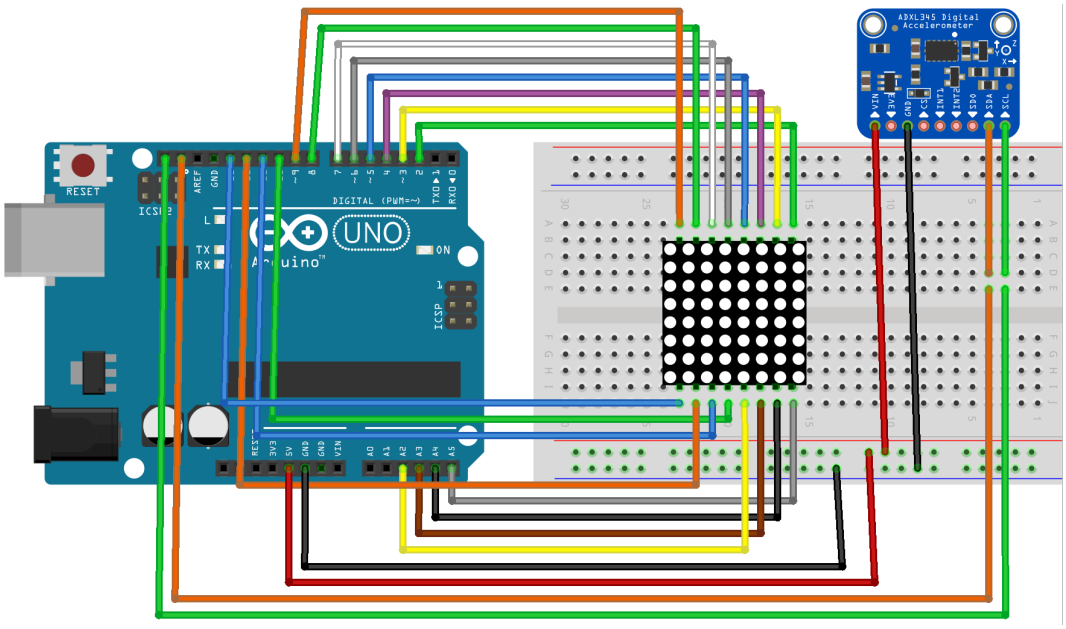
Akcelerometr je malé pohybové čidlo, které měří pohybové zrychlení a to nejlépe ve všech třech osách. Ze znalosti zrychlení a hmotnosti lze zjistit sílu působící na těleso.

Akcelerometry jsou vhodné nejen pro měření odstředivých a setrvačných sil, ale i pro určování pozice tělesa, jeho náklon nebo vibrace. Akcelerometry jsou dnes i v mobilních telefonech a využívají se v leteckém a automobilovém průmyslu.

Aby bylo možné definovat úhly akcelerometru ve třech rozměrech **pitch**, **roll** a **theta**, využívají se všechny tři výstupy akcelerometru. **Pitch** (ró), je definováno jako úhel vzhledem k ose X a země. **Roll** (fí) je definováno jako úhel vzhledem k ose Y a země. **Theta** je úhel vzhledem k ose Z - gravitace.

JAK ZAPOJIT AKCELEROMETR

② Pokud máte zapojený maticový displej, připojte podle přiloženého schématu akcelerometr.



DEJTE SI POZOR

➔ **Zaměřte se zejména na správné zapojení napájení akcelerometru a datových pinů SDA a SCL.**

PROGRAMOVÁNÍ

③ Otevřete si předchozí programový kód a uložte jej jako nový soubor. Tím si ušetříte práci a čas.

Pro co možná nejjednodušší programování akcelerometru ADXL 345 je vhodné použít některou z knihoven. Proto si ji na začátku programového kódu připojte. Použitá knihovna pro ADXL 345 je k dispozici na GitHub ke stažení.

④ Připojení knihoven pro práci s akcelerometrem je následující:

```
#include <Wire.h>
#include <ADXL345.h>

acc.read(&Xg, &Yg, &Zg);
```

⑤ Pro výpočet úhlů **roll** a **pitch** využijte následující programový zápis.

```
roll  = (atan2(-Yg, Zg)*180.0)/M_PI;
pitch = (atan2(Xg, sqrt(Yg*Yg + Zg*Zg))*180.0)/M_PI;
```

ÚKOL PRO VÁS

➔ **A) Inovujte programový kód otevřeného programu tak, abyste aplikovali uvedený vzorec pro výpočet úhlů roll a pitch. Nezapomeňte definovat všechny proměnné.**

⑥ Hotový program nahrajte do desky Arduino. Pokud je vše v pořádku, tak na pohyb akcelerometru je znázorněn na maticovém displeji pohybujícím se světlem.

ŘEŠENÍ ÚLOH

Úkol A)

```
#include <Wire.h>
#include <ADXL345.h>

ADXL345 acc;

const int row[8] = {2, 7, 19, 5, 13, 18, 12, 16};
const int col[8] = {6, 11, 10, 3, 17, 4, 8, 9};

int pixels[8][8];

int x = 5;
int y = 5;

void setup(){
    acc.begin();

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


for(int x = 0; x < 8; x++) {
      for(int y = 0; y < 8; y++) {
        pixels[x][y] = HIGH;
      }
    }
}

void loop(){
    readSensors();
    refreshScreen();
}

void readSensors(){
  double pitch, roll, Xg, Yg, Zg;
  acc.read(&Xg, &Yg, &Zg);

  roll  = (atan2(-Yg, Zg)*180.0)/M_PI;
  pitch = (atan2(Xg, sqrt(Yg*Yg + Zg*Zg))*180.0)/M_PI;
```

```
42
43    pixels[x][y] = HIGH;
44    x = 7 - map(roll, -20, 20, 0, 7);
45    y = map(pitch, -20, 20, 0, 7);
46    pixels[x][y] = LOW;
47  }
48
49  void refreshScreen(){
50    for(int j = 0; j<8;j++){
51      digitalWrite(row[j], HIGH);
52      for(int k = 0; k<8; k++){
53        int thisPixel = pixels[j][k];
54        digitalWrite(col[k], thisPixel);
55        if (thisPixel == LOW) {
56          digitalWrite(col[k], HIGH);
57        }
58      }
59     digitalWrite(row[j], LOW);
60    }
61  }
66
```

PODROBNÝ PRŮVODCE TEORIÍ

PODROBNĚ ROZEPSANÉ PŘÍKLADY S POPISEM FUNKCIONALIT OBVODŮ A PROGRAMOVÉHO KÓDU, KTERÝ JE ZAMĚŘEN NA POUŽÍVÁNÍ MATICOVÉHO LED DISPLEJE A AKCELEROMETRU. ŘEŠENÉ ÚKOLY ZOHLEDŇUJÍ NOVĚ PROBRANÉ TÉMA A STAVÝ NA PŘEDCHOZÍCH ZNALOSTECH.

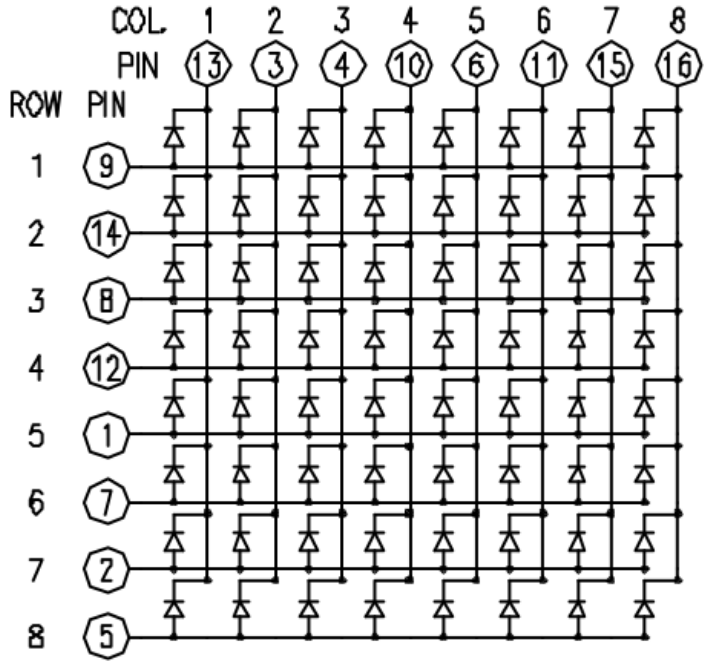
OBSAH PRŮVODCE

ZOBRAZOVÁNÍ SYMBOLŮ NA MATICOVÉM LED DISPLEJI

MATICOVÝ LED DISPLEJ

Maticové LED displeje, i když mají jen malé rozlišení, se používají hlavě proto, že lze s nimi dosáhnout efektního vzhledu a zobrazení jednoduché grafiky, jsou relativně velké a čitelné z větší vzdálenosti, v šeru aktivním svitem poutají pozornost.

LED displeje jsou nejčastěji reprezentovány jako matice LED diod, uspořádaných v řadách a sloupcích. Řady představují běžné anody a sloupce společné katody nebo naopak.



Obr. 1 - Schéma zapojení diod maticového displeje

Chceme-li ovládat matici, musí se propojit její řady i sloupce s mikrokontrolérem. Sloupce jsou připojeny ke katodám LED (viz Obr. 1), takže pokud mají být zapnuty LED diody v konkrétním sloupci, musí být hodnota pro každou z diod nastavena na **LOW**. Řádky jsou připojeny k anodám LED diod, takže pro jejich zapnutí musí být nastavena hodnota na **HIGH**. Pokud jsou hodnoty pro řádky i sloupce nastaveny stejně na **LOW** nebo **HIGH**, dioda se na displeji nerozsvítí.

Chcete-li ovládat jednotlivé LED diody displeje, musí se nastavit ve jejím sloupci hodnota **LOW** a řádek na **HIGH**. Má-li být ovládáno několik LED diod za sebou, musí se nastavit

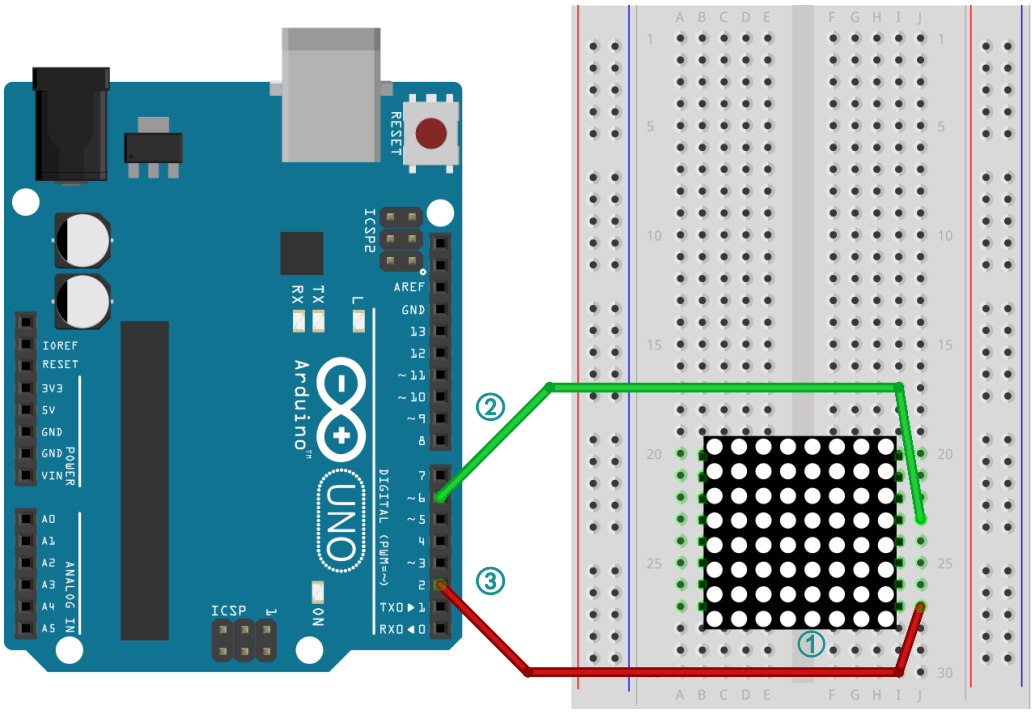
řádek na **HIGH**, poté konkrétní sloupec na **LOW** nebo **HIGH**; sloupec **LOW** zapne odpovídající LED a ve sloupci **HIGH** jej vypne.

Pokud chceme rozsvítit LED na pozici [1,1], musíme připojit pin displeje 13 na + a 9 na GND. Pokud bychom ale chtěli rozsvítit zároveň bod [1,1] a [2,2], je situace trošku komplikovanější. Kdybychom totiž připojili 13 a 3 na + a 14 i 9 na GND současně, rozsvítil by se celý čtverec ([1,1], [2,1], [1,2], [2,2]). Z tohoto důvodu se vždy pracuje jen s jednou řadou (ať už jde o řádek, nebo o sloupec), rozsvítí se všechny body, které se mají zobrazit, poté se napájení řady vypne, a to samé se opakuje se všemi dalšími řadami. Pokud toto „překreslování“ probíhá dostatečně rychle, lidské oko si ničeho nevšimne (kvůli jeho setrvačnosti). Anody jsou vypnuté, pokud je na jejich pinu stav LOW, u katod je tomu naopak – vypnuté jsou při stavu HIGH. Kombinace pinů a zapojení displeje je v následující tabulce.

Matice pin	Řádek	Sloupec	Arduino pin
1	5	-	13
2	7	-	12
3	-	2	11
4	-	3	10
5	8	-	A2
6	-	5	A3
7	6	-	A4
8	3	-	A5
9	1	-	2
10	-	4	3
11	-	6	4
12	4	-	5
13	-	1	6
14	2	-	7
15	-	7	8
16	-	8	9

Tab. 2 - Rozložení pinů

ZAPOJENÍ DISPLEJE PRO JEHO OTESTOVÁNÍ



Obr. 2 – Základní zapojení LED matice

① Maticový displej je umístěn v kontaktním poli, tím se bude lépe propojovat s deskou Arduino.

② Zelený vodič připojte k pinu displeje číslo 13 a druhý konec vodiče k desce Arduino na pin 6.

③ Červený vodič připojte k pinu displeje číslo 9 a druhý konec vodiče k desce Arduino na pin 2.

Princip zapojení maticového displeje je patrný z výše uvedené tabulky kombinace pinů.

PROGRAMOVÝ KÓD

Programový kód je velmi jednoduchý a je podobný jako při zapojení obyčejné LED diody.

```
1   int pinA=2;                                  ├──①
2   int pinB=6;                                  ├──②
3
4   void setup() {
5     pinMode(pinA,OUTPUT);                      ├──③
6     pinMode(pinB,OUTPUT);                      ├──④
7     digitalWrite(pinA,HIGH);                   ├──⑤
8     digitalWrite(pinB,HIGH);                   ├──⑥
9   }
10
11  void loop() {
12    digitalWrite(pinB,LOW);                    ├──⑦
13    delay(200);                                ├──⑧
14    digitalWrite(pinB,HIGH);                   ├──⑨
15    delay(200);                                ├──⑩
16  }
```

① Deklarace proměnné **pinA** definuje číslo pinu na desce Arduino. V maticovém displeji je připojen na katodu diody.

② Deklarace proměnné **pinB** definuje číslo pinu na desce Arduino. V maticovém displeji je připojen na anodu diody.

③ Vyhrazení pinu pro katodu.

④ Vyhrazení pinu pro anodu.

⑤ Nastavení hodnoty na **HIGH**.

⑥ Nastavení hodnoty na **HIGH**. Pokud je na anodě a katodě stejná hodnota, tak dioda nesvítí.

⑦ Na anodu je přivedena hodnota **LOW**), tím dojde k rozsvícení diody.

⑧ Dioda svítí 200ms.

⑨ Na diodu je přivedena hodnota **HIGH**, tím dioda opět zhasne.

⑩ Dioda zhasne na 200ms.

Nezapomeňte program zkompilovat a nahrát do desky Arduino. Pokud je vše v pořádku, měla by blikat první dioda, tj. v levém horním rohu.

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.Správný port – ujistěte se, že máte vybraný správný port pro připojení k desce Arduino pomocí USB kabelu.

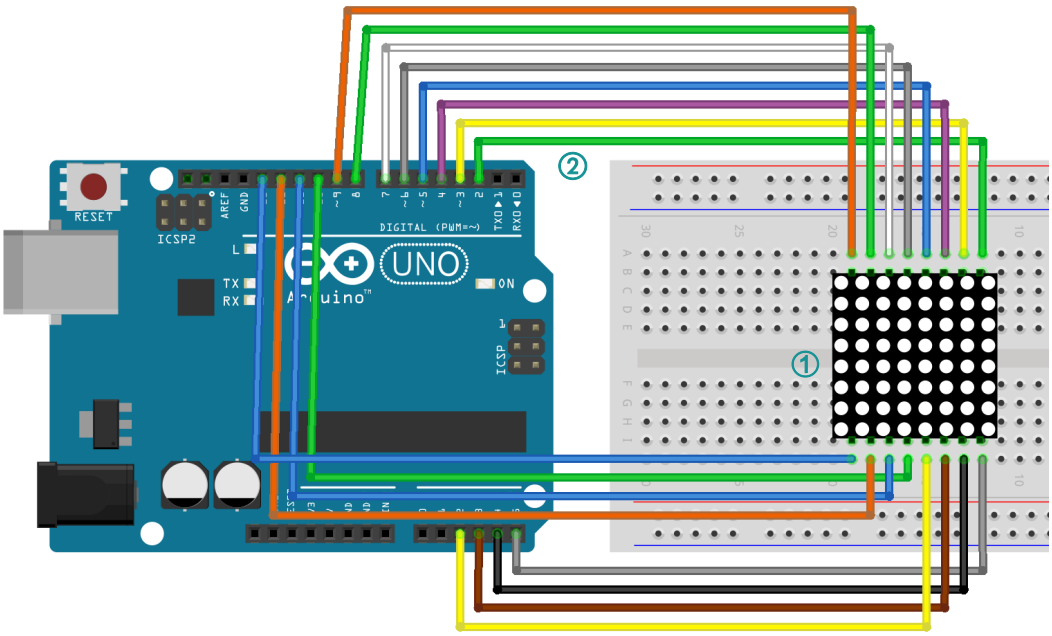
DIODA NA DISPLEJI NEBLIKÁ

Kontrola zapojení – ujistěte se, že jsou vodiče opravdu zapojeny do správných pinů na kontaktním poli a desce Arduino.

(Př. 1) Upravte obvod zapojení displeje a programový kód předchozího příkladu tak, aby blikaly i diody ve všech rozích stejně jako dioda první.

(Př. 2) Změňte programový kód předchozího příkladu tak, aby diody v protilehlých rozích blikali střídavě.

ÚPLNÉ ZAPOJENÍ DISPLEJE



Obr. 3 – Základní zapojení LED matice

① Maticový displej je umístěn v kontaktním poli, tím se bude lépe propojovat s deskou Arduino.

② Pro propojení displeje s deskou Arduino je potřeba 16 vodičů. Zapojení je velmi jednoduché. Při zapojení si lze vzít na pomoc rozložení pinů v tabulce Tab. 2 - Rozložení pinů.

Na uvedeném zapojení lze provést celou řadu příkladů. Proto by bylo dobré, aby tento obvod mohl být složen i do příštích hodin a vy jste se věnovali pouze programování.

PROGRAMOVÝ KÓD – POSTUPNÉ ROZSVĚCOVÁNÍ DIOD DISPLEJE

Rozsvěcování jednotlivých diod na maticovém displeji, lze udělat několika různými způsoby. Resp. lze vytvořit mnoho různých světelných kombinací. Zde je uvedena varianta, kdy se v každém řádku postupně rozsvěcují diody.

```
1   const int row[8] = {                                          ①
2     2, 7, 19, 5, 13, 18, 12, 16
3   };
4
5   const int col[8] = {                                          ②
6     6, 11, 10, 3, 17, 4, 8, 9
7   };
8
9   void setup(){
10      for(int i = 0; i < 8; i++){                               ③
11          pinMode(col[i], OUTPUT);
12          pinMode(row[i], OUTPUT);
13          digitalWrite(col[i], HIGH);                           ④
14          digitalWrite(row[i], LOW);                            ⑤
15      }
16  }
17
18  void loop(){
19      for(int j = 0; j<8;j++) {
20        digitalWrite(col[j],LOW);                               ⑥
21        for(int k = 0;k<8;k++){
22          digitalWrite(row[k],HIGH);                            ⑦
23          delay(200);                                           ⑧
24        }
25        for(int i = 0;i<8;i++){                                 ⑨
26          digitalWrite(row[i],LOW);
27          digitalWrite(col[i],HIGH);
28        }
29      }
30  }
```

① Deklarace pole **`row[8]`**, ve kterém jsou definovány čísla pinů na desce Arduino pro řádky displeje.

② Deklarace pole **`col[8]`**, ve kterém jsou definovány čísla pinů na desce Arduino pro sloupce displeje.

③ Využití cyklu **`for`** k nastavení pinů pro sloupce a řádky jako výstupních.
④ Vymazání (zhasnutí) sloupců displeje.
⑤ Vymazání (zhasnutí) řádků displeje.
⑥ V cyklu **`for`** se postupně nastavují jednotlivé sloupce na hodnotu **`LOW`**.
⑦ V následujícím cyklu **`for`** se pro každý řádek v aktuálním sloupci nastavuje hodnota **`HIGH`**, čímž dojde k rozsvícení diody.
⑧ Aby bylo vidět postupné rozsvěcování diod je nastavena pauza na 200ms.
⑨ Ostatní sloupce a řádky se vypnou (zhasnou).

Nezapomeňte program zkompilovat a nahrát do desky Arduino. Pokud je vše v pořádku, měla by blikat první dioda, tj. v levém horním rohu.

(Př. 3) Upravte (optimalizujte) programový kód tak, aby se aktualizace a mazání displeje prováděla ve dvou vámi deklarovaných funkcích.

(Př. 4) Upravte programový kód tak, aby se v celém, rozsvíceném displeji postupně posouval vypnutý sloupec a při tomto vypnutém sloupci projížděl vypnutý řádek.

PROGRAMOVÝ KÓD – ZOBRAZOVÁNÍ SYMBOLŮ

Maticový displej, při vhodné kombinaci rozsvícených diod, může zobrazovat jednoduché symboly.

```
1   const int row[8] = {                                          ①
2     2, 7, 19, 5, 13, 18, 12, 16
3   };
4
5   const int col[8] = {                                          ②
6     6, 11, 10, 3, 17, 4, 8, 9
7   };
8
9   byte image[8][8] = {                                          ③
10    {0,0,0,0,0,0,0,0},
11    {0,1,1,0,0,1,1,0},
12    {1,0,0,1,1,0,0,1},
13    {1,0,0,0,0,0,0,1},
14    {1,0,0,0,0,0,0,1},
15    {0,1,0,0,0,0,1,0},
16    {0,0,1,0,0,1,0,0},
17    {0,0,0,1,1,0,0,0}};
18
19  void setup(){
20      for(int i = 0; i < 8; i++){                               ④
21          pinMode(col[i], OUTPUT);
22          pinMode(row[i], OUTPUT);
23          digitalWrite(col[i], HIGH);                           ⑤
24          digitalWrite(row[i], LOW);
25      }
26  }
27
28  void loop(){
29      refreshScreen();                                          ⑥
30  }
31
32  void refreshScreen(){
33    for(int j = 0; j<8;j++){                                    ⑦
34      digitalWrite(col[j], LOW);
35      for(int k = 0; k<8; k++){                                 ⑧
36        digitalWrite(row[k], image[k][j]);                      ⑨
37      }
38      Clear();                                                  ⑩
39    }
40  }
```

```
41
42  void Clear(){                                                    ⑪
43    for(int i = 0; i<8; i++){
44      digitalWrite(row[i],LOW);
45      digitalWrite(col[i],HIGH);
46    }
47  }
48
```

① Deklarace pole **`row[8]`**, ve kterém jsou definovány čísla pinů na desce Arduino pro řádky displeje.

② Deklarace pole **`col[8]`**, ve kterém jsou definovány čísla pinů na desce Arduino pro sloupce displeje.

③ Deklarace dvourozměrného pole **`image[8][8]`**, které obsahuje definici řádků a sloupců displeje. Rozsvícené diody jsou reprezentovány hodnotou 1 a zhasnuté hodnotou 0.

④ Využití cyklu **`for`** k nastavení pinů pro sloupce a řádky jako výstupních.

⑤ Vymazání (zhasnutí) sloupců a řádků displeje.

⑥ Volání funkce **`refreshScreen`**, která vykresluje symbol na displej.

⑦ V cyklu **`for`** se postupně nastavují jednotlivé sloupce na hodnotu **`LOW`**.

⑧ V následujícím cyklu **`for`** se pro každý řádek v aktuálním sloupci.

⑨ Nastavuje se hodnota z pole **`image[k][j]`**, kde hodnota **`HIGH`** = **`1`** a **`LOW`** = **`0`**. Tím dojde k rozsvícení/zhasnutí aktuální diody.

⑩ Volání funkce **`Clear`**, která zhasne diody mimo symbol.

⑪ Deklarace funkce **`Clear`**.

Definice tvaru symbolů je velmi snadné. Můžete využít nástroj, pomocí něhož si symbol „naklikáte“ a následně použijete vygenerované dvourozměrné pole vypnutých/zapnutých diod, které vložíte do programového kódu.

Odkaz: https://www.prf.jcu.cz/generator-led-matrix/index.htm

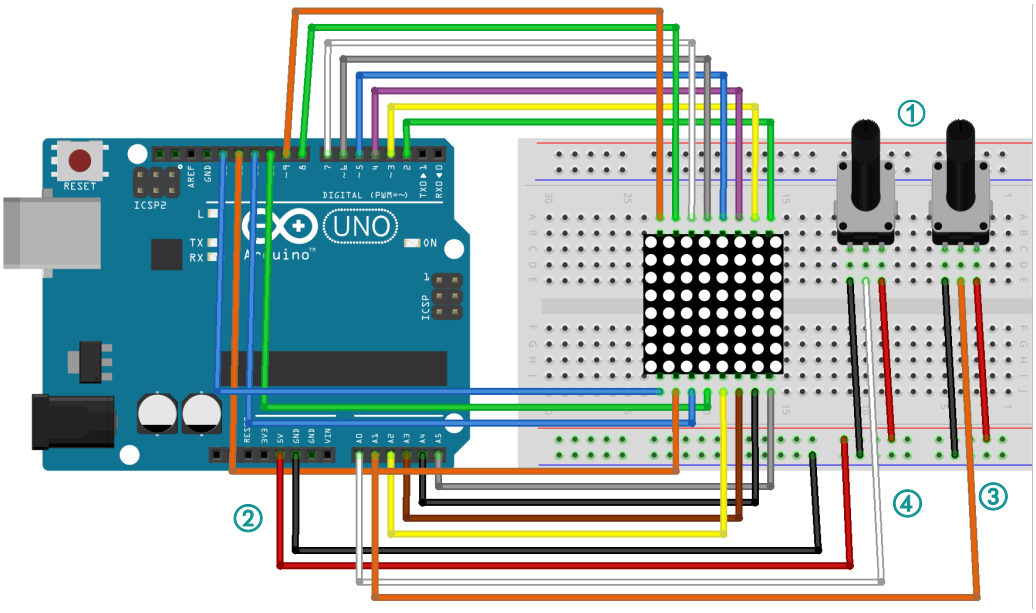
Nezapomeňte program zkompilovat a nahrát do desky Arduino. Pokud je vše v pořádku, zobrazí se na displeji symbol srdce.

(Př. 5) Vyzkoušejte si zobrazit další symboly. Může to být smajlík, trojúhelník, dva kruhy v sobě atd.

(Př. 6) Změňte programový kód tak, aby se střídavě zobrazovalo velké a malé srdce.

OVLÁDÁNÍ DIOD MATICOVÉHO DISPLEJE POMOCÍ HODNOT Z ANALOGOVÉHO VSTUPU

Pro vstup se použijí dva potenciometry. Jeden bude ovládat pohyb rozsvícené diody v řádcích a druhý ve sloupcích.



Obr. 4 – Základní zapojení LED matice

① Potenciometry jsou připojeny přímo do kontaktní desky.

② Na kontaktní desku je přivedeno napájení **5V** a zemnící vodič **GND** přímo z desky Arduino.

③ Signál z prvního potenciometru je přiveden na analogový vstup desky Arduino – **A1**.

④ Signál z druhého potenciometru je přiveden na analogový vstup desky Arduino – **A0**.

Zapojení maticového displeje je stejné, jako u předchozích příkladů, proto lze využít již zapojený obvod maticového displeje a přidat pouze potenciometry, což je již opakování. Studenti zapojení mohou provést v rámci samostatné úlohy.

PROGRAMOVÝ KÓD – ANALOGOVÉ VSTUPY

Při programování ovládání maticového displeje pomocí analogových vstupů, lze opět vyjít z předchozích programů. Samotné doprogramování ovládání je pak velice jednoduché.

```
1   const int row[8] = {                                          ①
2     2, 7, 19, 5, 13, 18, 12, 16
3   };
4
5   const int col[8] = {                                          ②
6     6, 11, 10, 3, 17, 4, 8, 9
7   };
8
9   int pixels[8][8];                                             ③
10
11  int x = 5;                                                    ④
12  int y = 5;
13
14  void setup(){
15      for(int i = 0; i < 8; i++){
16          pinMode(col[i], OUTPUT);
17          pinMode(row[i], OUTPUT);
18          digitalWrite(row[i], LOW);
19      }
20
21      for(int x = 0; x < 8; x++) {
22        for(int y = 0; y < 8; y++) {
23          pixels[x][y] = HIGH;                                  ⑤
24        }
25      }
26  }
27
28  void loop(){
29      readSensors();                                            ⑥
30      refreshScreen();
31  }
32
33  void readSensors(){                                           ⑦
34    pixels[x][y] = HIGH;
35    x = 7 - map(analogRead(A0), 0, 1023, 0, 7);                 ⑧
36    y = map(analogRead(A1), 0, 1023, 0, 7);                     ⑨
37    pixels[x][y] = LOW;                                         ⑩
38  }
39
40
```

```
41  void refreshScreen(){
42    for(int j = 0; j<8;j++){                          ⑪
43      digitalWrite(row[j], HIGH);                     ⑫
44      for(int k = 0; k<8; k++){
45        int thisPixel = pixels[j][k];                 ⑬
46        digitalWrite(col[k], thisPixel);              ⑭
47        if (thisPixel == LOW) {
48          digitalWrite(col[k], HIGH);                 ⑮
49        }
50      }
51     digitalWrite(row[j], LOW);                       ⑯
52    }
53  }
```

① Deklarace pole **`row[8]`**, ve kterém jsou definovány čísla pinů na desce Arduino pro řádky displeje.

② Deklarace pole **`col[8]`**, ve kterém jsou definovány čísla pinů na desce Arduino pro sloupce displeje.

③ Deklarace dvourozměrného pole **`pixels[8][8]`**, které obsahuje pozici rozsvícené diody.

④ Výchozí pozice rozsvícené diody při spuštění programu. Následuje již známá inicializace pinů pro výstup a vypnutí všech diod.

⑤ Inicializace matice **`pixels`**.

⑥ Volání funkce **`readSensors()`**, která čte hodnoty z potenciometrů.

⑦ Deklarace funkce **`readSensors()`**.

⑧ Namapování hodnoty z potenciometru pro souřadnici **`x`**.

⑨ Namapování hodnoty z potenciometru pro souřadnici **`y`**.

⑩ Nastavení nové pozice bodu tak, aby se LED dioda rozsvítila.

⑪ Projít přes řádky displeje.

⑫ Nastavení konkrétního bodu v řádku na hodnotu **HIGH**.

⑬ Získat hodnotu aktuálního bodu.

⑭ Pokud je hodnota aktuálního bodu pro řádek **HIGH** a pro sloupec **LOW**, dioda se rozsvítí.

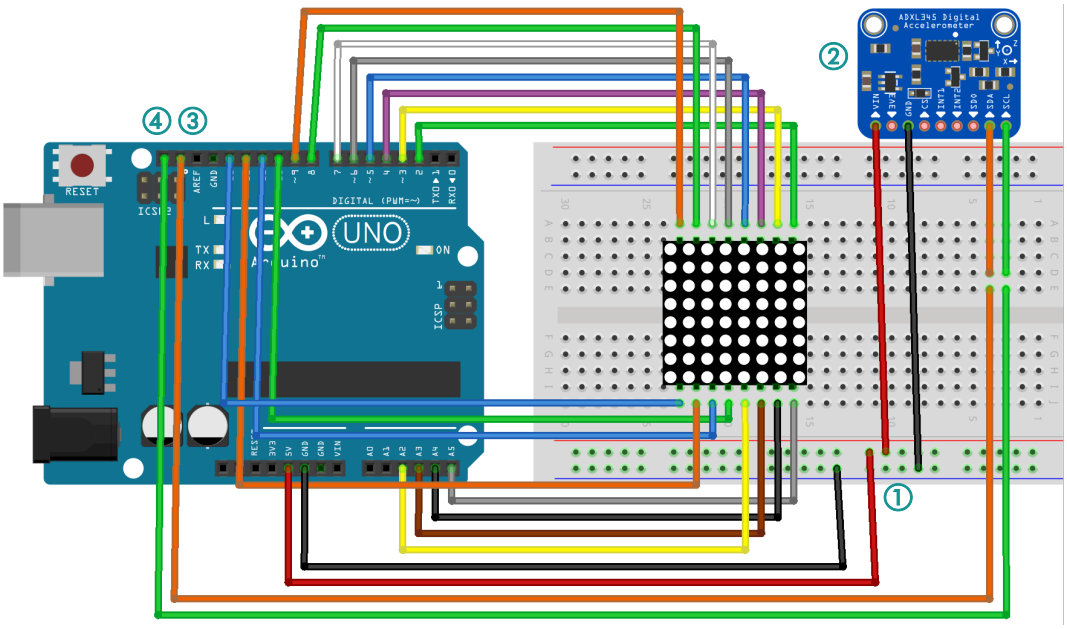
⑮ Vypnutí bodu.

⑯ Vypnutí celého řádku.

(Př. 7) Dokázali byste pomocí potenciometrů a maticového displeje vytvořit klasickou hru Ping Pong? Pokud byste si nevěděli rady, tak vzorové řešení je k dispozici na GitHub.

SPOJENÍ MATICOVÉHO DISPLEJE A AKCELEROTMETRU

Pro spojení maticového displeje a akcelerometru lze vyjít z předchozího příkladu. Na základě předchozích kapitol již víte, jak maticový displej používat. Největším problémem je tak zapojení a programování akcelerometru, jako zdroje signálu.



Obr. 5 – Zapojení LED matice a akcelerometru

① Na kontaktní desku je přivedeno napájení **5V** a zemnící vodič **GND** přímo z desky Arduino.

② Akcelerometr je připojen pouze na vodiče, aby se sním dalo snadno pohybovat. Z akcelerometru se připojí pin **VIN** (tento pin může být také značen jako VCC, VCC_IN) do kontaktního pole k napájení. Pin **GND** se připojí do kontaktního pole na zem.

③ Z akcelerometru se dále připojí pin **SDA** do desky Arduino. Na desce Arduino tento pin bývá označen stejným názvem, na spodní straně desky.

④ Stejně se do desky Arduino připojí i druhý datový pin **SCL**.

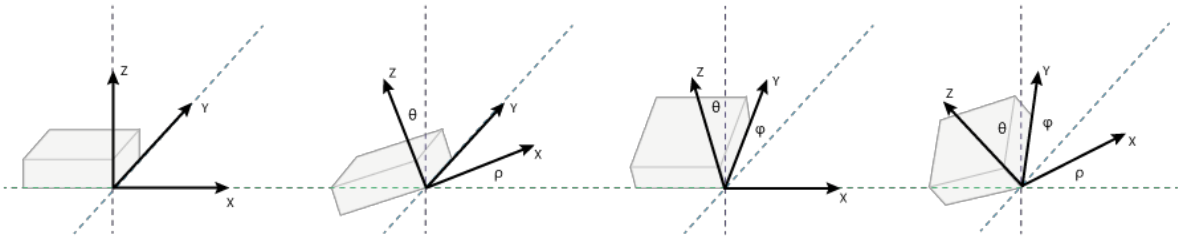
Zapojení maticového displeje je stejné, jako u předchozích příkladů, proto lze využít již zapojený obvod maticového displeje a přidat pouze akcelerometr.

CO TO JE AKCELEROMETR A JAK PRACUJE

Akcelerometr je malé pohybové čidlo, které měří pohybové zrychlení a to nejlépe ve všech třech osách. Ze znalosti zrychlení a hmotnosti lze zjistit sílu působící na těleso.

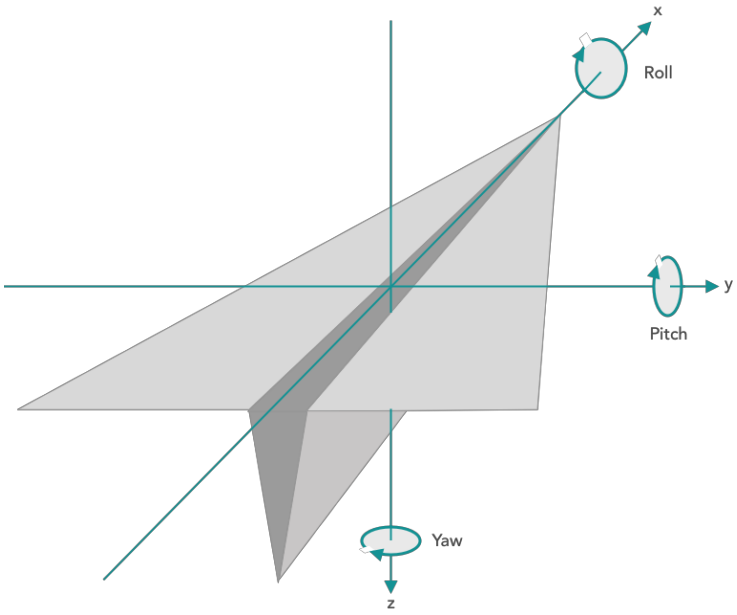
Akcelerometry jsou vhodné nejen pro měření odstředivých a setrvačných sil, ale i pro určování pozice tělesa, jeho náklon nebo vibrace. Akcelerometry jsou dnes i v mobilních telefonech a využívají se v leteckém a automobilovém průmyslu.

Aby bylo možné definovat úhly akcelerometru ve třech rozměrech **pitch**, **roll** a **theta**, využívají se všechny tři výstupy akcelerometru. **Pitch** (ró), je definováno jako úhel vzhledem k ose X a země. **Roll** (fí) je definováno jako úhel vzhledem k ose Y a země. **Theta** je úhel vzhledem k ose Z - gravitace.



$$\rho = arctan\left(\frac{Ax}{\sqrt{Ay^2 + Az^2}}\right) \qquad \varphi = arctan\left(\frac{Ay}{\sqrt{Ax^2 + Az^2}}\right) \qquad \theta = arctan\left(\frac{\sqrt{Ax^2 + Ay^2}}{Az^2}\right)$$

Je důležité poznamenat, že akcelerometr poskytuje poměrně přesné údaje úhlové orientace za předpokladu, že gravitace je jediná síla působící na snímač. Nicméně, při pohybu a otáčení senzoru, mohou působit jiné síly a dochází ke kolísání přesnosti. Výsledkem potom jsou údaje obsahující šum, který způsobuje sice krátkodobé, ale významné odchylky.



Pro uvedený příklad nás budou zajímat úhly **Pitch** a **Roll**. Zobecněný vzorec pro zrychlení z naměřených hodnot akcelerometru je:

$$G_{Accel} = Raw_{Accel} \frac{Range}{2^{Resolution-1}}$$

Jakmile jsou k dispozici odpovídající hodnoty z akcelerometru, lze pokračovat ve výpočtu úhlů pomocí následujících rovnic:[1]

$$pitch = arctan\left(\frac{G_y}{\sqrt{G_x^2 + G_x^2}}\right) \qquad roll = arctan\left(\frac{-G_x}{G_z}\right)$$

Tyto vzorce lze v Arduino kódu dají přepsat v následujícím tvaru:

```
1  roll  = (atan2(-Yg, Zg)*180.0)/M_PI;
2  pitch = (atan2(Xg, sqrt(Yg*Yg + Zg*Zg))*180.0)/M_PI;
```

[1] *NXP: Freescale Semiconductor* [online]. 2013, 2013(6) [cit. 2018-11-15]. Dostupné z: https://cache.freescale.com/files/sensors/doc/app_note/AN3461.pdf

PROGRAMOVÝ KÓD – ČTENÍ HODNOT Z AKCELEROMETRU

Programové propojení akcelerometru a maticového displeje je velmi podobné jako při zapojení s potenciometry. Složitější je pouze v tom, že se musí provést čtení dat z akcelerometru. Složitost tohoto programování je závislá na zvolené knihovně, která se použije pro spojení akcelerometru a desky Arduino.

KNIHOVNA ADXL345

➔ **Pro jednoduší práci a správnou funkcionalitu akcelerometru, musí být nainstalovaná podpůrná knihovna, kterou naleznete na GitHub.**

```
1   #include <Wire.h>                                    ①
2   #include <ADXL345.h>                                ②
3
4   ADXL345 acc;                                        ③
5
6   const int row[8] = {
7     2, 7, 19, 5, 13, 18, 12, 16
8   };
9
10  const int col[8] = {
11    6, 11, 10, 3, 17, 4, 8, 9
12  };
13
14  int pixels[8][8];
15
16  int x = 5;
17  int y = 5;
18
19  void setup(){                                       ④
20      acc.begin();
21
22      for(int i = 0; i < 8; i++){
23          pinMode(col[i], OUTPUT);
24          pinMode(row[i], OUTPUT);
25          digitalWrite(row[i], LOW);
26      }
27
28
```

```
29  for(int x = 0; x < 8; x++) {
30        for(int y = 0; y < 8; y++) {
31          pixels[x][y] = HIGH;
32        }
33      }
34  }
35
36  void loop(){
37      readSensors();
38      refreshScreen();
39  }
40
41  void readSensors(){                                              ├──⑤
42    double pitch, roll, Xg, Yg, Zg;                                ├──⑥
43    acc.read(&Xg, &Yg, &Zg);                                       ├──⑦
44
45    roll  = (atan2(-Yg, Zg)*180.0)/M_PI;                           ├──⑧
46    pitch = (atan2(Xg, sqrt(Yg*Yg + Zg*Zg))*180.0)/M_PI;           ├──⑨
47
48    pixels[x][y] = HIGH;
49    x = 7 - map(roll, -20, 20, 0, 7);                              ├──⑩
50    y = map(pitch, -20, 20, 0, 7);                                 ├──⑪
51    pixels[x][y] = LOW;
52  }
53
54  void refreshScreen(){
55    for(int j = 0; j<8;j++){
56      digitalWrite(row[j], HIGH);
57      for(int k = 0; k<8; k++){
58        int thisPixel = pixels[j][k];
59        digitalWrite(col[k], thisPixel);
60        if (thisPixel == LOW) {
61          digitalWrite(col[k], HIGH);
62        }
63      }
64     digitalWrite(row[j], LOW);
65    }
66  }
```

① Připojení knihovny **Wire.h**, která umožňuje komunikaci se zařízeními s I2C/TWI.

② Připojení knihovny **ADXL345.h**, zajišťuje jednodušší komunikaci mezi deskou Arduino a akcelerometrem.

③ Vytvoření instance třídy ADXL 345 pro práci s akcelerometrem.

④ Zahájení komunikace akcelerometru s deskou Arduino.

⑤ Deklarace funkce **readSensor**, která provádí čtení a výpočet dat z akcelerometru.

⑥ Deklarace proměnných pro výpočet klopení a klonění.
⑦ Čtení hodnot z akcelerometru.
⑧ Výpočet klopení.
⑨ Výpočet klonění.
⑩ Namapování hodnoty z akcelerometru pro **`klopení`**.
⑪ Namapování hodnoty z akcelerometru pro **`klonění`**.

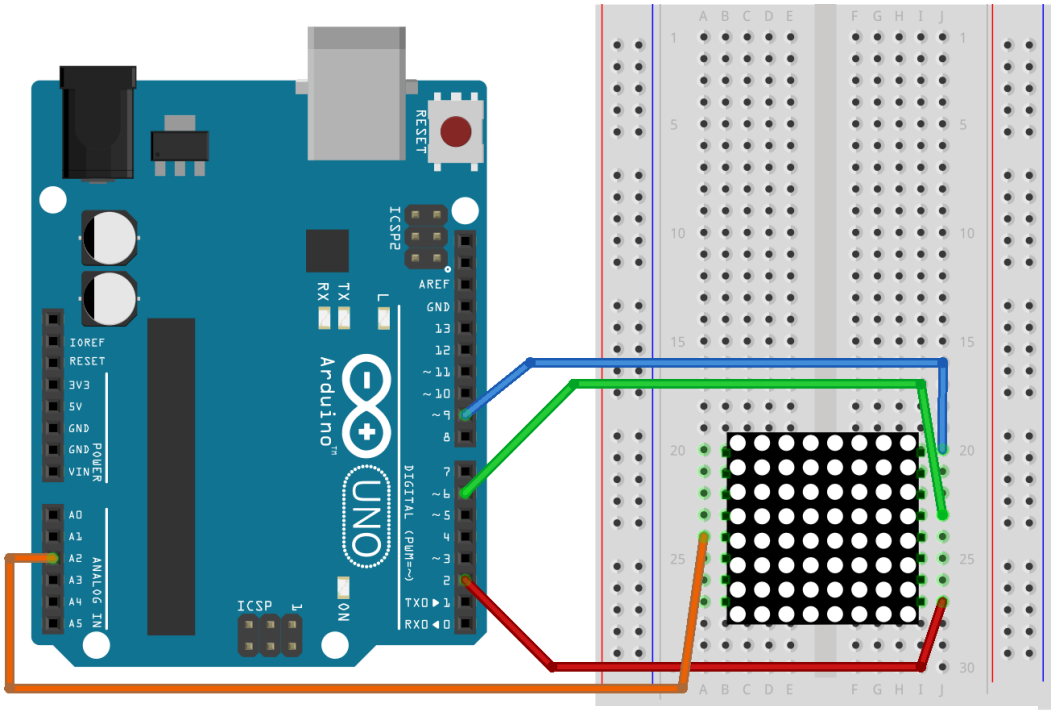
Z programového kódu je patrné, že při využití funkcí je změna kódu minimální. Prakticky došlo pouze ke změně ve funkci pro čtení ze senzorů, která spočívala v převedení získaných hodnot z akcelerometru na hodnoty použitelné k namapování pro maticový displej.

(Př. 8) Předchozí schéma a programový kód rozšiřte o zapojení dvou servomotorů, které budou reagovat na polohu akcelerometru.

ŘEŠENÍ PŘÍKLADŮ

CHYBA! NENAŠEL SE ZDROJ ODKAZU.

Změna zapojení obvodu má vést k ujasnění si principu zapojení maticového displeje. Při řešení příkladu stačí využít tabulku pinů Tab. 2 - Rozložení pinů.



Obr. 6 – Základní zapojení LED matice

```
1  int pinA=2;
2  int pinB=6;
3
4  int pinC=9;
5  int pinD=A2;
6
7  void setup() {
8    pinMode(pinA,OUTPUT);
9    pinMode(pinB,OUTPUT);
10   digitalWrite(pinA,HIGH);
11   digitalWrite(pinB,HIGH);
12
13   pinMode(pinC,OUTPUT);
```

```
14    pinMode(pinD,OUTPUT);
15    digitalWrite(pinC,HIGH);
16    digitalWrite(pinD,HIGH);
17
18  }
19
20  void loop() {
21    digitalWrite(pinB,LOW);
22    digitalWrite(pinD,HIGH);
23    delay(200);
24
25    digitalWrite(pinB,LOW);
26    digitalWrite(pinD,HIGH);
27    delay(200);
28  }
```

(PŘ. 1

K vyřešení příkladu se využije stejné zapojení jako u příkladu 1. Programový kód bude takřka stejný. Pouze se změní pořadí zapínání diod.

```
int pinA=2;
int pinB=6;

int pinC=9;
int pinD=A2;

void setup() {
  pinMode(pinA,OUTPUT);
  pinMode(pinB,OUTPUT);
  digitalWrite(pinA,HIGH);
  digitalWrite(pinB,HIGH);

  pinMode(pinC,OUTPUT);
  pinMode(pinD,OUTPUT);
  digitalWrite(pinC,HIGH);
  digitalWrite(pinD,HIGH);

}

void loop() {
  digitalWrite(pinB,HIGH);  // změna na HIGH
  digitalWrite(pinD,LOW);   // změna na LOW
  delay(200);

  digitalWrite(pinB,LOW);
  digitalWrite(pinD,HIGH);
  delay(200);
}
```

(PŘ. 3

Původní programový kód stačí rozdělit do dvou funkcí. Např. **`refreshScreen`** a **`Clear`**.

```
const int row[8] = {
  2, 7, 19, 5, 13, 18, 12, 16
};

const int col[8] = {
  6, 11, 10, 3, 17, 4, 8, 9
};

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

void loop(){
    refreshScreen();
}

void refreshScreen(){
  for(int j = 0; j<8;j++){
    digitalWrite(col[j], LOW);
    for(int k = 0; k<8; k++){
      digitalWrite(row[k], HIGH);
    }
    Clear();
  }
}

void Clear(){
  for(int i = 0; i<8; i++){
    digitalWrite(row[i],LOW);
    digitalWrite(col[i],HIGH);
  }
}
```

(PŘ. 4

Postačí upravit pořadí zapínání diod ve funkci **`refreshScreen`**.

```

const int row[8] = {
  2, 7, 19, 5, 13, 18, 12, 16
};

const int col[8] = {
  6, 11, 10, 3, 17, 4, 8, 9
};

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

void loop(){
    refreshScreen();
}

void refreshScreen(){
  for(int j = 0; j<8;j++){
    digitalWrite(row[j], LOW);
    for(int k = 0; k<8; k++){
      digitalWrite(col[k], HIGH);
      delay(100);
      digitalWrite(col[k], LOW);
    }
    digitalWrite(row[j], HIGH);
  }
}
```

(PŘ. 5

V původním programovém kódu postačí změnit dvourozměrné pole image.

Definice tvaru symbolů je velmi snadné. Můžete využít nástroj, pomocí něhož si symbol „naklikáte“ a následně použijete vygenerované dvourozměrné pole vypnutých/zapnutých diod, které vložíte do programového kódu.

Odkaz: https://www.prf.jcu.cz/generator-led-matrix/index.htm

```
// Srdce
byte image[8][8] = {
  {0,0,0,0,0,0,0,0},
  {0,1,1,0,0,1,1,0},
  {1,0,0,1,1,0,0,1},
  {1,0,0,0,0,0,0,1},
  {1,0,0,0,0,0,0,1},
  {0,1,0,0,0,0,1,0},
  {0,0,1,0,0,1,0,0},
  {0,0,0,1,1,0,0,0}};

// Smajlík
byte image[8][8] = {
  {0,0,1,1,1,1,0,0},
  {0,1,0,0,0,0,1,0},
  {1,0,1,0,0,1,0,1},
  {1,0,0,0,0,0,0,1},
  {1,0,1,0,0,1,0,1},
  {1,0,0,1,1,0,0,1},
  {0,1,0,0,0,0,1,0},
  {0,0,1,1,1,1,0,0}};
```

(PŘ. 6

V příkladu je jednoduchá inovace, a to v podobě předávání parametru funkce **`refreshScreen`**. Tímto parametrem je pole s různým obrazcem.

```

const int row[8] = {
  2, 7, 19, 5, 13, 18, 12, 16
};

const int col[8] = {
  6, 11, 10, 3, 17, 4, 8, 9
};

// Velke srdce
byte image[8][8] = {
  {0,0,0,0,0,0,0,0},
  {0,1,1,0,0,1,1,0},
  {1,0,0,1,1,0,0,1},
  {1,0,0,0,0,0,0,1},
  {1,0,0,0,0,0,0,1},
  {0,1,0,0,0,0,1,0},
  {0,0,1,0,0,1,0,0},
  {0,0,0,1,1,0,0,0}};

// Male srdce
byte imageS[8][8] = {
  {0,0,0,0,0,0,0,0},
  {0,0,0,0,0,0,0,0},
  {0,0,0,1,0,1,0,0},
  {0,0,1,0,1,0,1,0},
  {0,0,1,0,0,0,1,0},
  {0,0,0,1,0,1,0,0},
  {0,0,0,0,1,0,0,0},
  {0,0,0,0,0,0,0,0}};

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

```

```
41  void loop(){
42      // Zobrazeni vždy po dobu 100 iteraci
43      for(int i = 0; i < 100; i++){
45        refreshScreen(image);
46     }
47
48     for(int i = 0; i < 100; i++){
49        refreshScreen(imageS);
50     }
51  }
52
53  void refreshScreen(unsigned char dat[8][8]){
54    for(int j = 0; j<8;j++){
55      digitalWrite(col[j], LOW);
56      for(int k = 0; k<8; k++){
57        digitalWrite(row[k], dat[k][j]);
58      }
59      delay(1);
60      Clear();
61    }
62  }
63
64  void Clear(){
65    for(int i = 0; i<8; i++){
66      digitalWrite(row[i],LOW);
67      digitalWrite(col[i],HIGH);
68    }
69  }
```