# 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: 5x45 min  Úroveň: | Vychází z: 3, 4, 5 |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | Maticový displej 8x8 |  | Akcelerometr | Potenciometr 2x |  |  | |
| Použité součástky | | | | | | |

|  |  |
| --- | --- |
|  | 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?   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. |

PrŮvodce hodinou I

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.

1. 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.



|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16 | 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.



1. 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.



1. 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 |  | ../../../_source/el-components/breadbords.pngDeska 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.

#### 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.



|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16 | 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);  } |

1. 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.



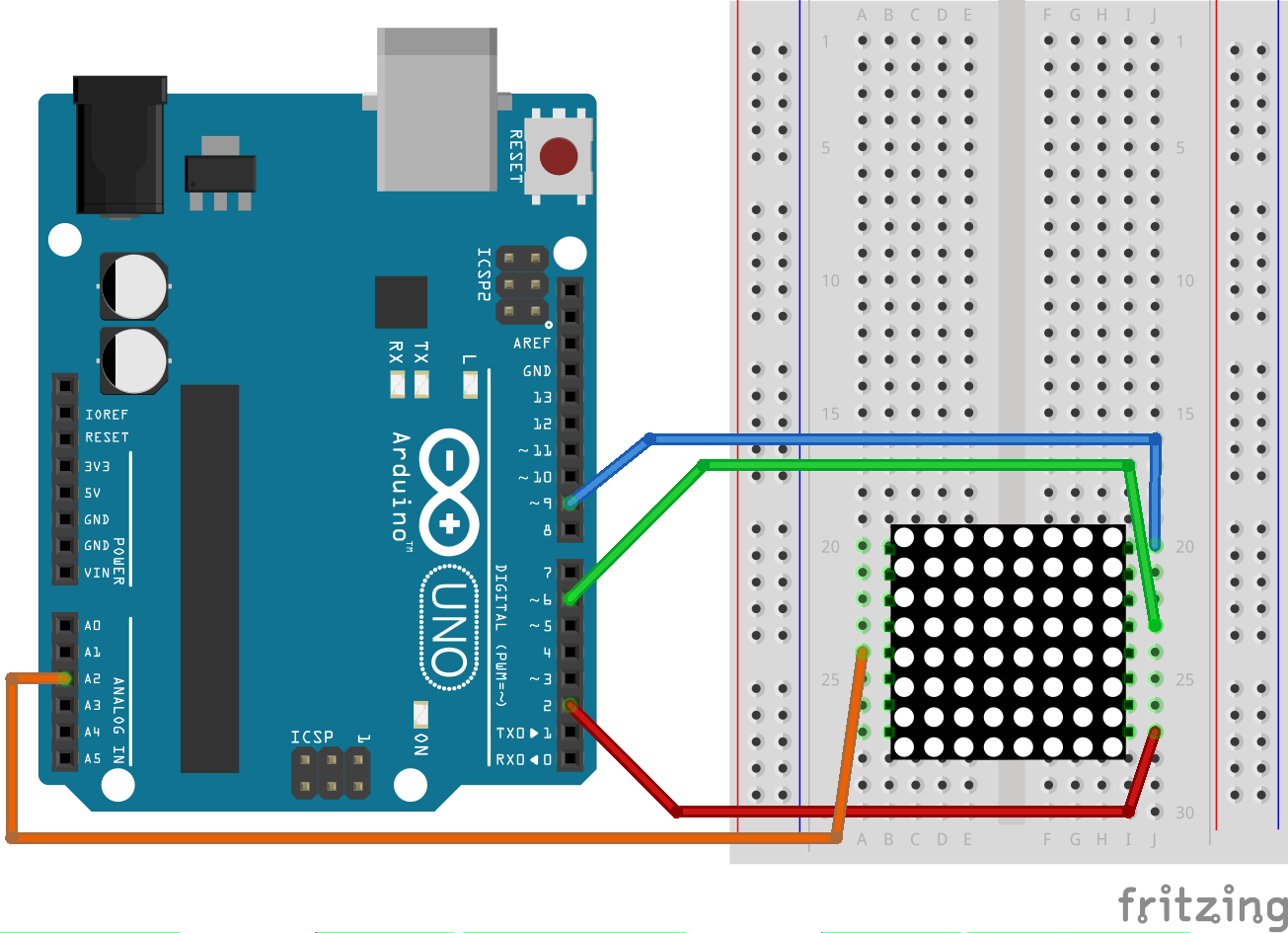
#### **Nezapomeňte**

Nahrajte program do desky Arduino, kliknutím na ikonu



Řešení úloh

**Úkol A)**



|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28 | 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)**

|  |  |  |
| --- | --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28 | 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);  } |  |

|  |  |
| --- | --- |
|  | 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?   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. |

PrŮvodce hodinou II

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.



1. 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.



1. 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.

|  |  |  |
| --- | --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25 | 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?



1. 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.



1. 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 |  | ../../../_source/el-components/breadbords.pngDeska Arduino |  | Kontaktní pole |  | |
| Použité součástky | | | | | | |

## Zopakujte si …

|  |  |  |  |
| --- | --- | --- | --- |
| 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 |

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

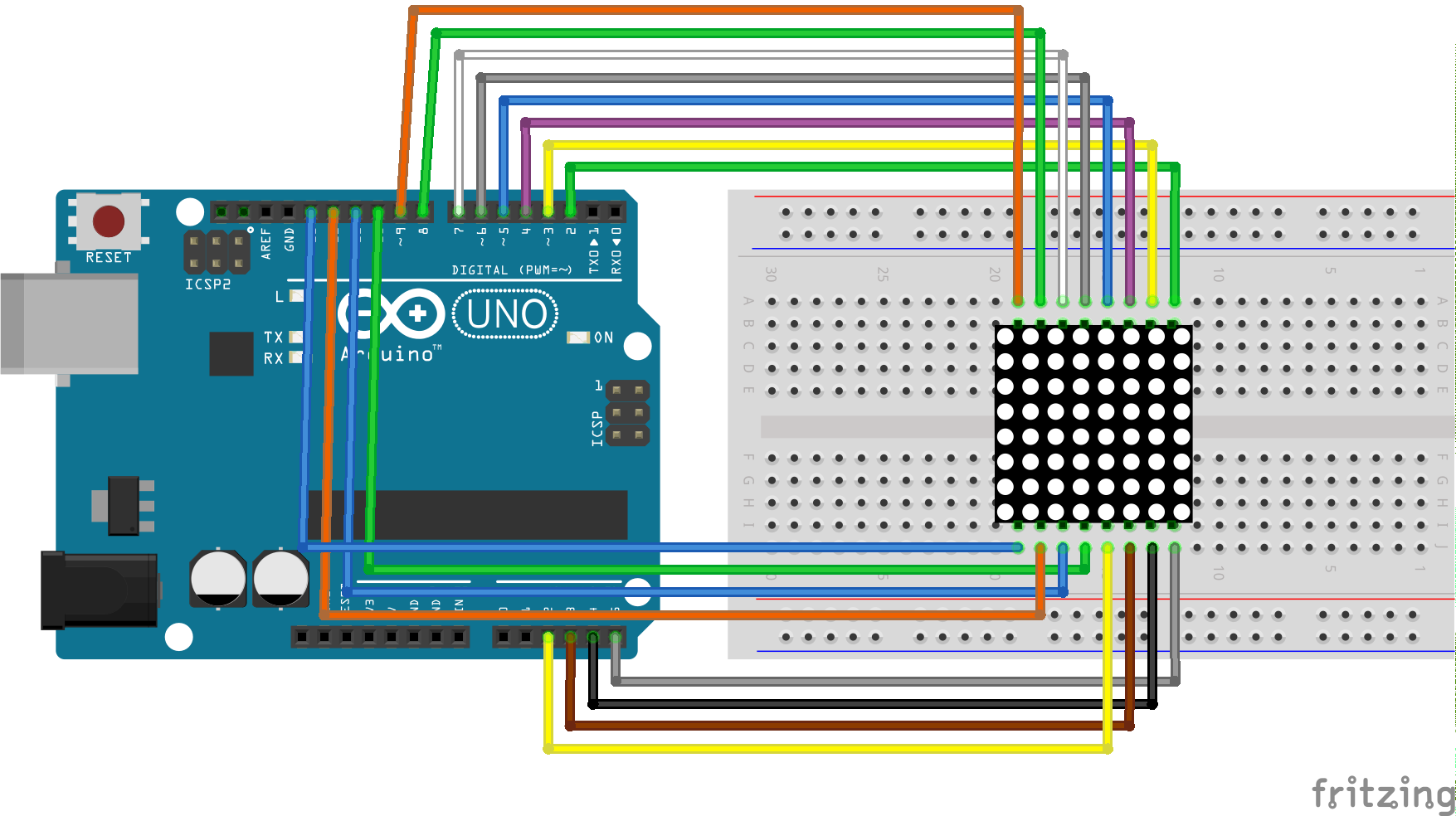
*Tab. 1 - Rozložení pinů*

#### **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 …

1. 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.



1. Napište a nahrajte následující program do desky Arduino.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30 | 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?



1. 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

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37 | 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 A)**

**Úkol B)**

|  |  |  |
| --- | --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32 | 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);  }  } |  |

|  |  |
| --- | --- |
|  | 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. |

PrŮvodce hodinou III

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.



1. 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.



1. 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?



1. 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



1. 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

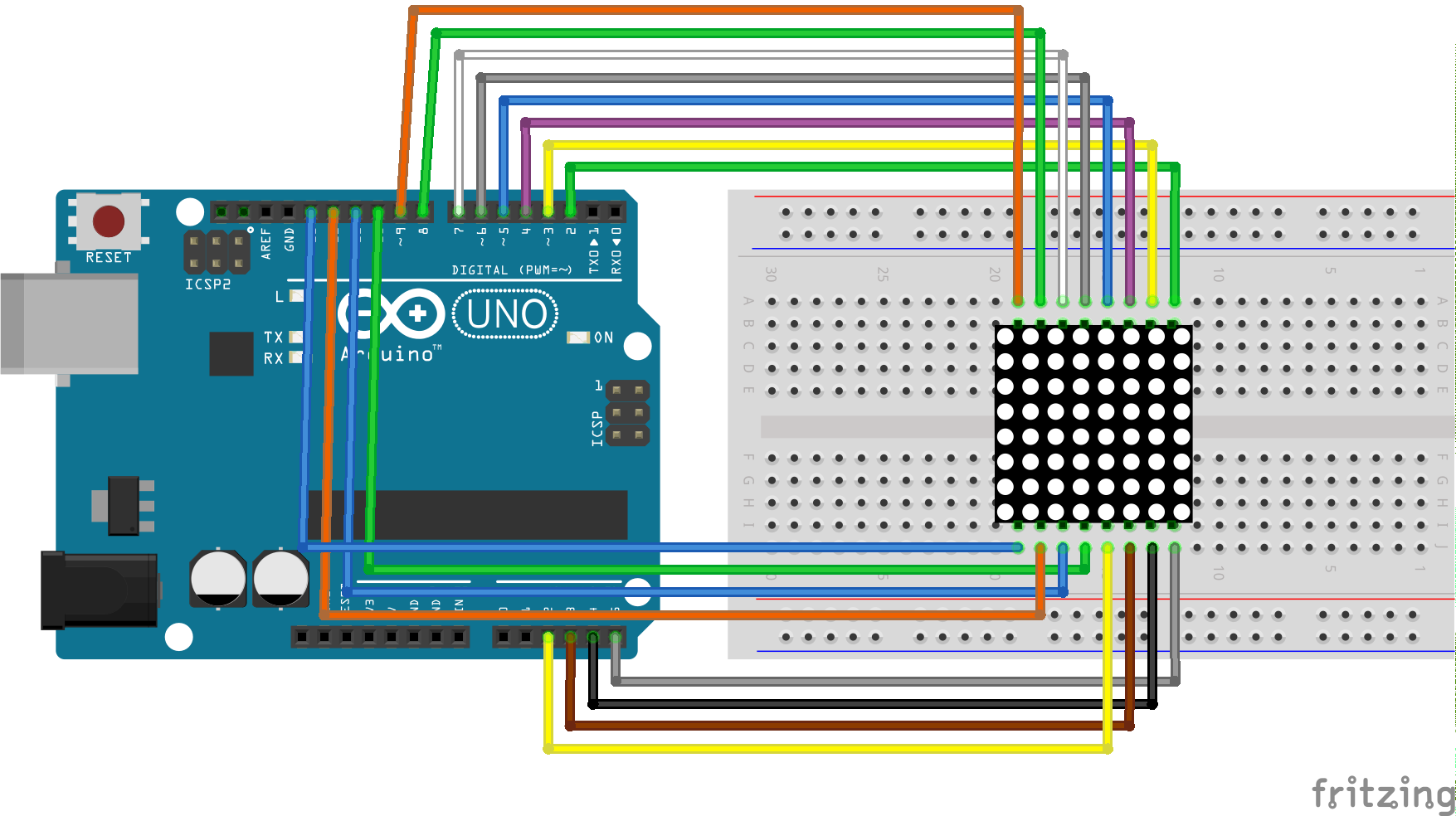
1. Zopakujete si cyklus FOR.
2. Pracovat s vícerozměrným polem.
3. Naučíte se princip zobrazování symbolů na maticovém displeji.

## Co budete potřebovat

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

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | Maticový displej 8x8 |  | ../../../_source/el-components/breadbords.pngDeska 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 …

1. 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.



|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40 | 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();  }  } |

1. 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

1. 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

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21 | // 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 A)**

**Úkol B)**

|  |  |  |
| --- | --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69 | 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++){  refreshScreen(image);  }  for(int i = 0; i < 100; i++){  refreshScreen(imageS);  }  }  void refreshScreen(unsigned char dat[8][8]){  for(int j = 0; j<8;j++){  digitalWrite(col[j], LOW);  for(int k = 0; k<8; k++){  digitalWrite(row[k], dat[k][j]);  }  delay(1);  Clear();  }  }  void Clear(){  for(int i = 0; i<8; i++){  digitalWrite(row[i],LOW);  digitalWrite(col[i],HIGH);  }  } |  |

|  |  |
| --- | --- |
|  | 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. |

PrŮvodce hodinou IV

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.



#### Úkol pro studenty

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



#### **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 řešení problému do kapitoly týkající se servomotorů.



1. 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.



1. 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

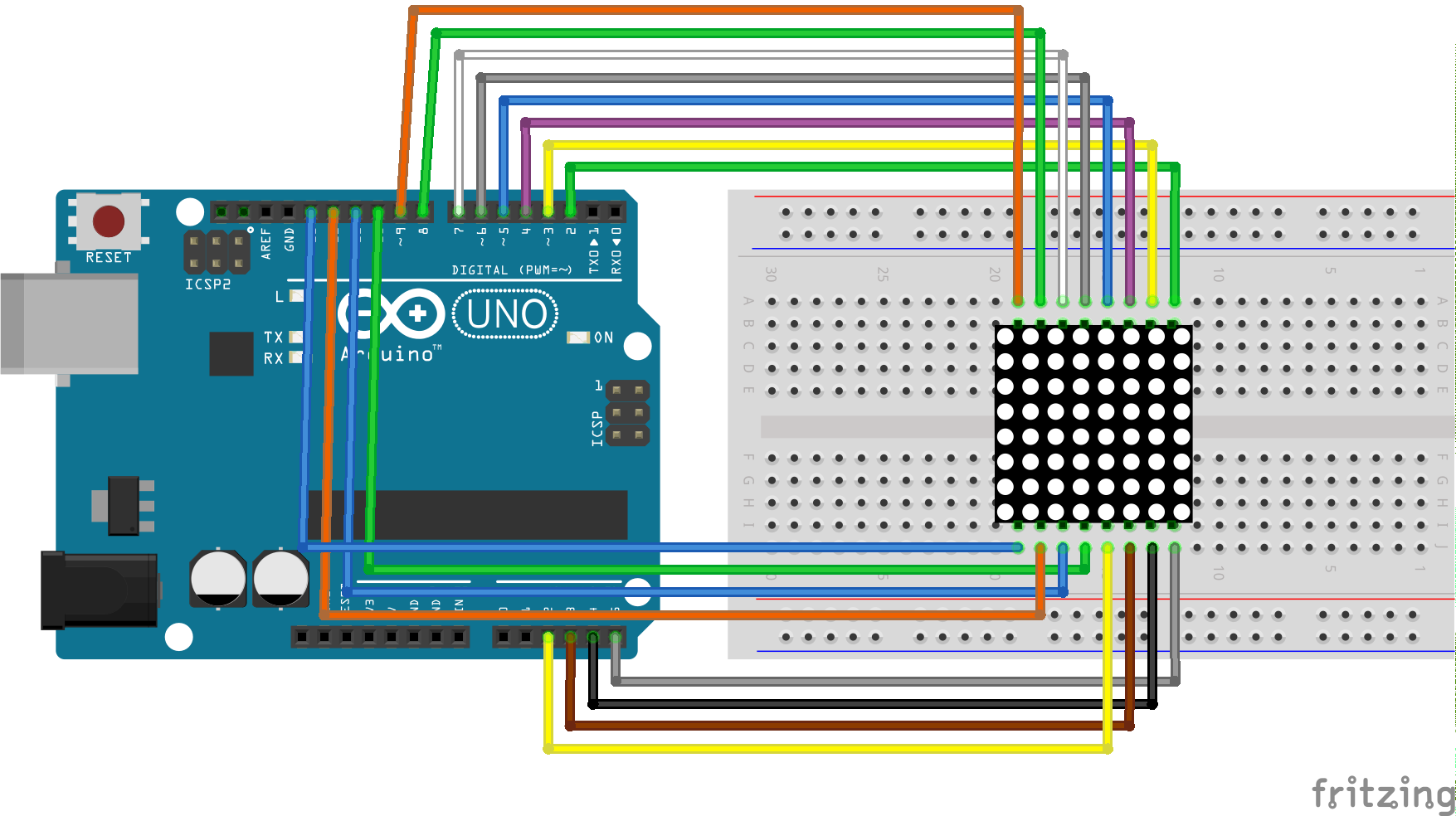
1. Zopakujete si zapojení potenciometru.
2. Spojení potenciometru a maticového displeje.
3. Zpracovávat hodnoty z potenciometru pro displej.

## Co budete potřebovat

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

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | Maticový displej 8x8 |  | ../../../_source/el-components/breadbords.pngDeska 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.



1. 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?



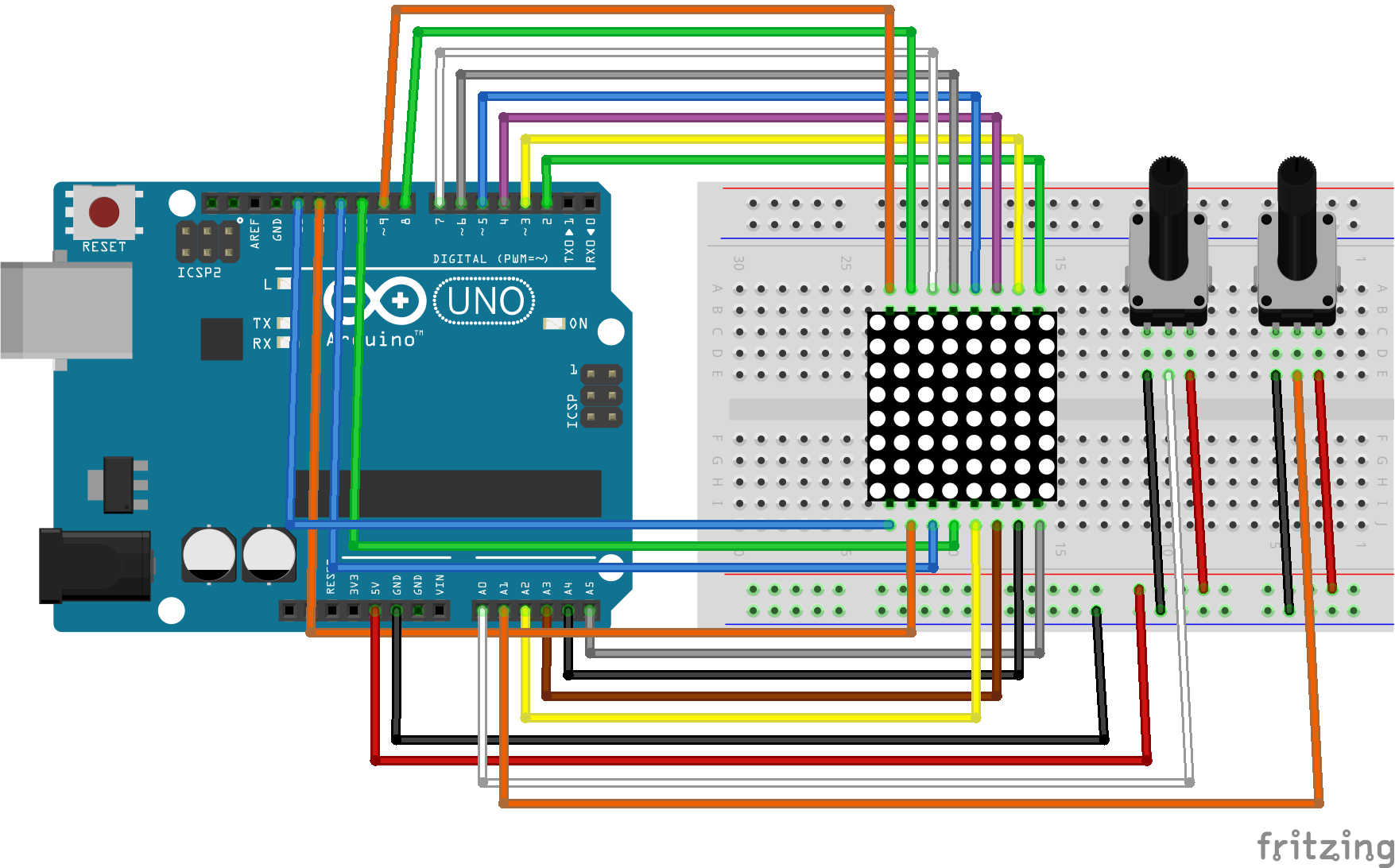
## A jde se programovat …

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

|  |  |  |
| --- | --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53 | 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(){  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(){  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)**



|  |  |
| --- | --- |
|  | 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?   1. Součásti obvodu – deska Arduino s USB kabelem, kontaktní pole, maticový LED displej 8x8, vodiče, akcelerometr. 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. |

PrŮvodce hodinou V

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.



1. KROK 5 minut

Vysvětlete podrobněji princip akcelerometru.

1. KROK 10 minut

Ať studenti zapojí akcelerometr podle přiloženého nebo promítaného schématu.

#### **NA co si dát pozor**

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



#### 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.



1. 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.

#### 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().



|  |  |  |
| --- | --- | --- |
| 1  2 | roll = (atan2(-Yg, Zg)\*180.0)/M\_PI;  pitch = (atan2(Xg, sqrt(Yg\*Yg + Zg\*Zg))\*180.0)/M\_PI; |  |

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

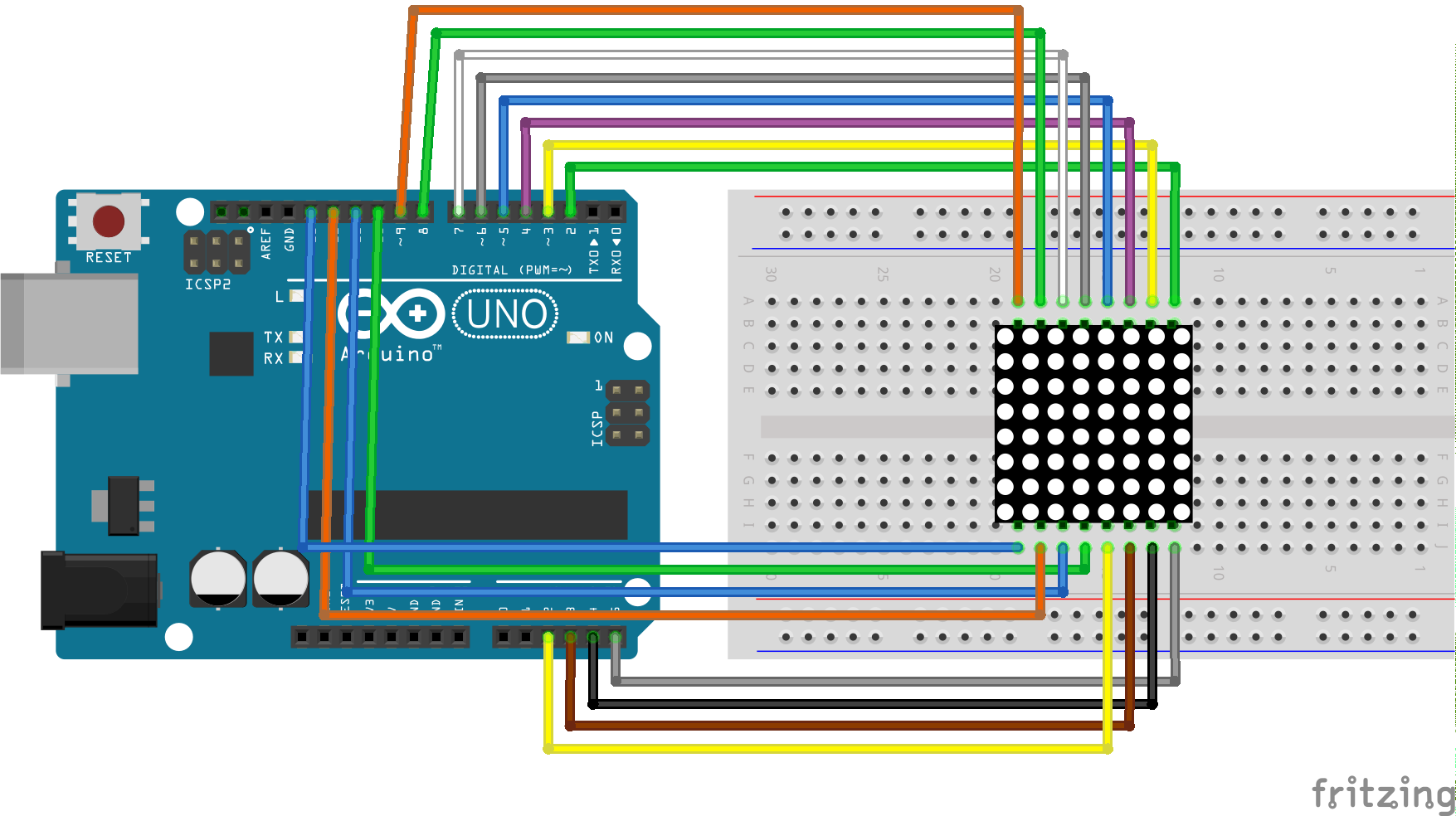
1. Zapojovat akcelerometr.
2. Spojení akcelerometru a maticového displeje.
3. Zpracovávat hodnoty z akcelerometru pro displej.

## Co budete potřebovat

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

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | Maticový displej 8x8 |  | ../../../_source/el-components/breadbords.pngDeska 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

1. 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

1. 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.



1. Připojení knihoven pro práci s akcelerometrem je následující:

|  |  |  |
| --- | --- | --- |
| 1  2  3  4 | #include <Wire.h>  #include <ADXL345.h>  acc.read(&Xg, &Yg, &Zg); |  |

|  |  |  |
| --- | --- | --- |
| 1  2 | roll = (atan2(-Yg, Zg)\*180.0)/M\_PI;  pitch = (atan2(Xg, sqrt(Yg\*Yg + Zg\*Zg))\*180.0)/M\_PI; |  |

1. Pro výpočet úhlů roll a pitch využijte následující programový zápis.

#### Úkol pro vás

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



1. 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)**

|  |  |  |
| --- | --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60  61  66 | #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;  pixels[x][y] = HIGH;  x = 7 - map(roll, -20, 20, 0, 7);  y = map(pitch, -20, 20, 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);  }  } |  |

# 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

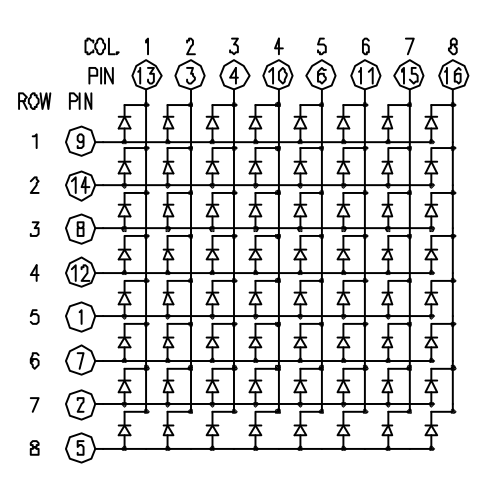
1. Získání dovedností při zapojování maticového LED displeje.
2. Naučení se zobrazovat symbolů na maticovém LED displeji.
3. Využití analogových hodnot pro ovládání diod maticového displeje.
4. Získání dovedností při zapojování akcelerometru.
5. Programování ovládání akcelerometru.
6. Získání dovedností při propojení maticového LED displeje a akcelerometru.

## 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. - 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. - Rozložení pinů

### Zapojení displeje pro jeho otestování



Obr. – Základní zapojení LED matice

b

c

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  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16 | 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);  } | h  c  d  e  f  a  b  g  j  i |

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.

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

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



#### 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

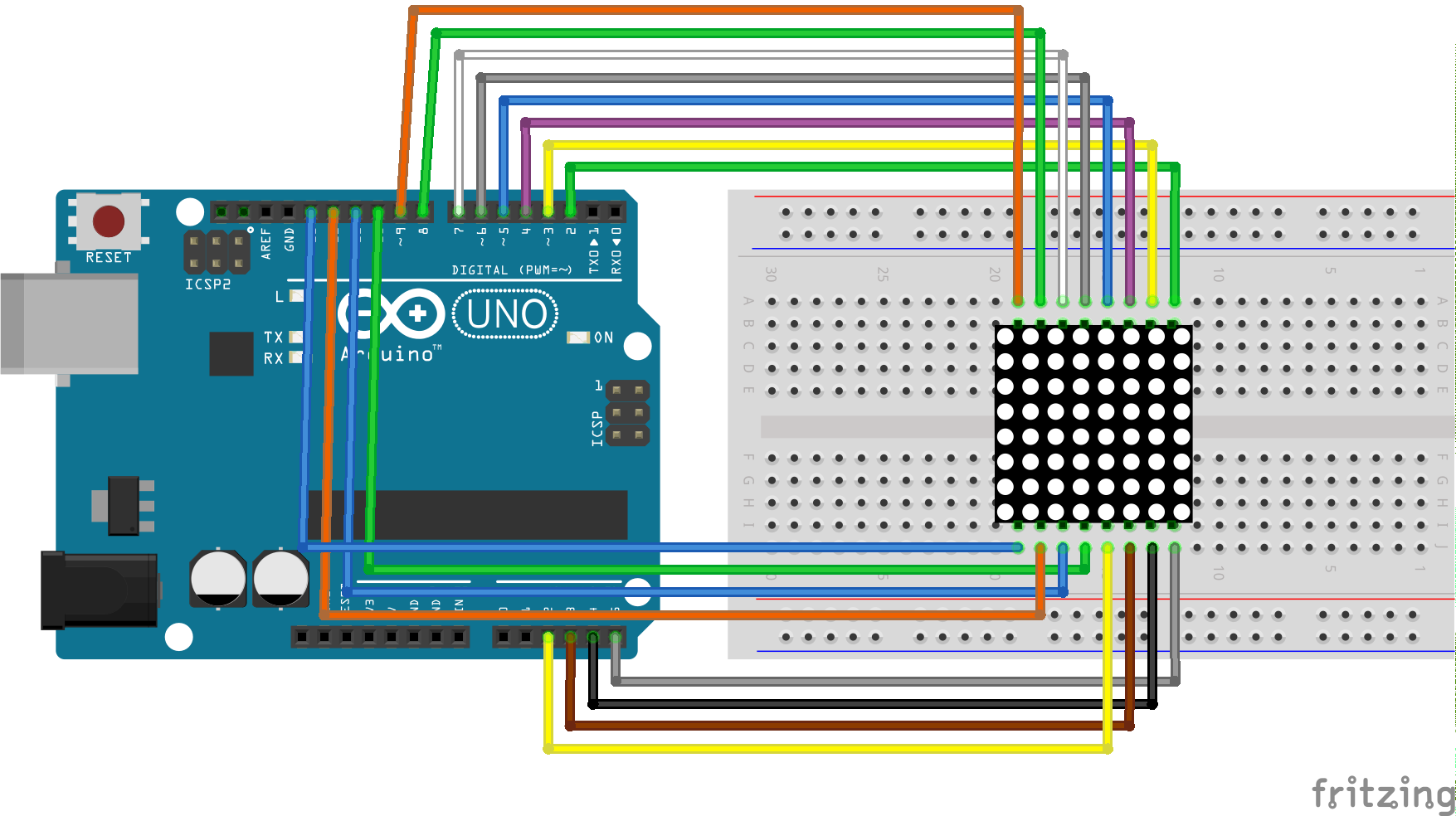
Kontrola zapojení – ujistěte se, že jsou vodiče opravdu zapojeny do správných pinů na kontaktním poli a desce Arduino.



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



### Úplné Zapojení displeje



Obr. – Základní zapojení LED matice

b

b

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  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30 | 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);  }  }  } | a  g  h  i  f  e  d  c  b |

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

(Př. ) 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  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48 | 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();  }  }  void Clear(){  for(int i = 0; i<8; i++){  digitalWrite(row[i],LOW);  digitalWrite(col[i],HIGH);  }  } | f  e  d  c  b  a  j  h  i  g  k |

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.



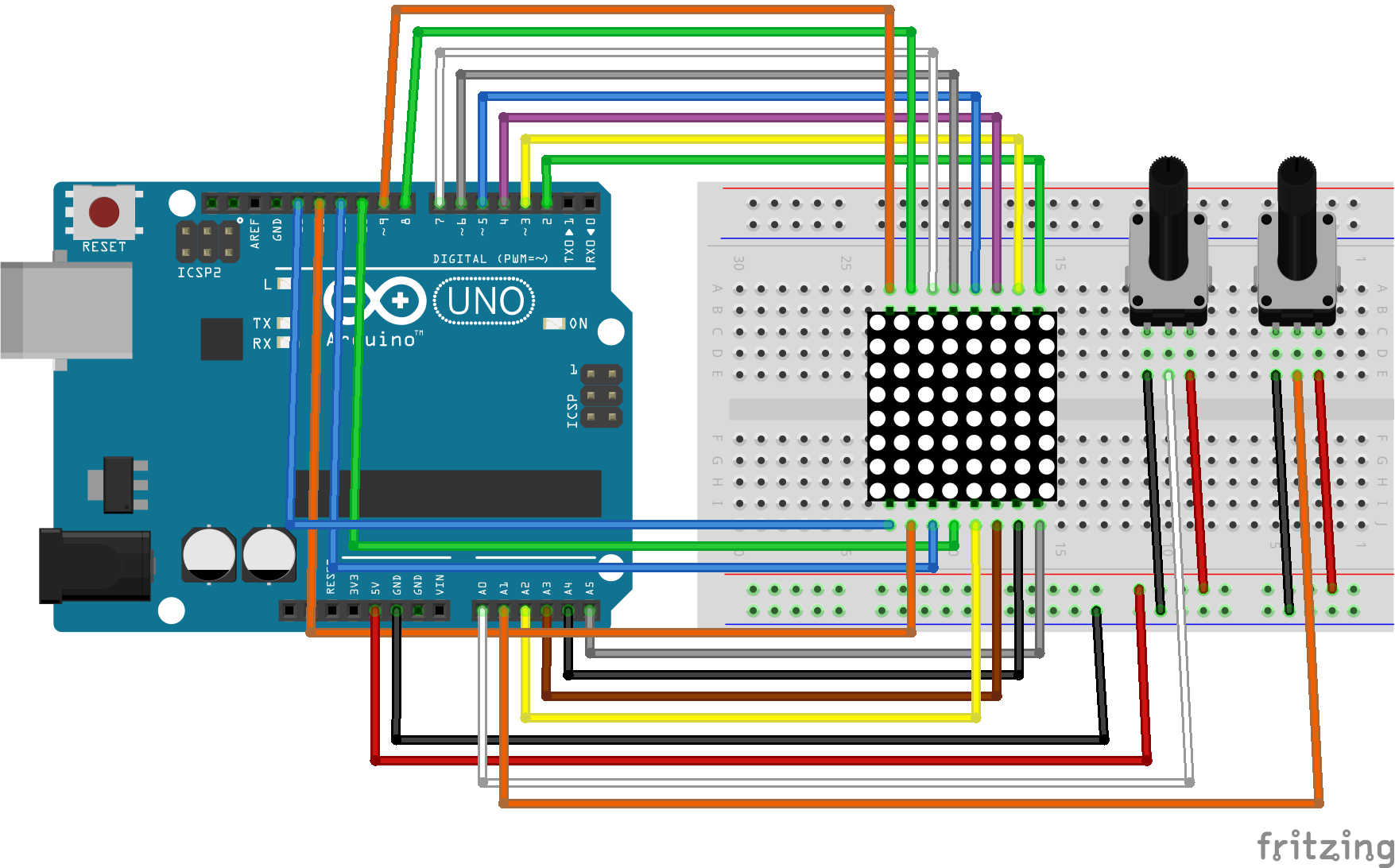
(Př. ) Vyzkoušejte si zobrazit další symboly. Může to být smajlík, trojúhelník, dva kruhy v sobě atd.

(Př. ) 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. – Základní zapojení LED matice

b

c

d

b

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  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53 | 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(){  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(){  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);  }  } | c  b  a  d  e  f  j  i  h  g  k  p  o  n  m  l |

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. ) 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. – Zapojení LED matice a akcelerometru

b

c

a

d

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.

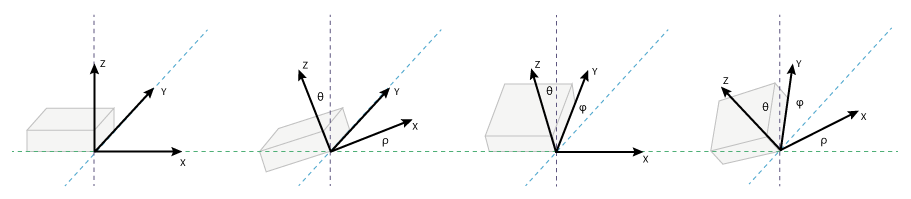
Zapojení maticového displeje je stejné, jako u předchozích příkladů, proto lze využít již zapojený obvod maticového displeje a přidat pouze akcelerometr.



### Co to je akcelerometr a jak pracuje

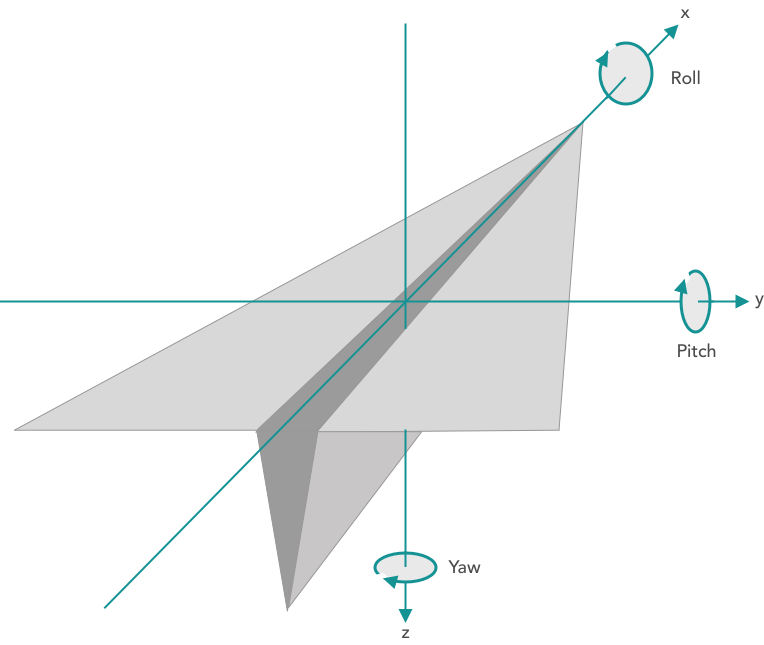
Akcelerometr je malé pohybové čidlo, které měří pohybové zrychlení a to nejlépe ve všech třech osách. Ze znalosti zrychlení a hmotnosti lze zjistit sílu působící na těleso.

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.

|  |  |  |
| --- | --- | --- |
|  |  |  |

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:

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

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

|  |  |  |
| --- | --- | --- |
| 1  2 | roll = (atan2(-Yg, Zg)\*180.0)/M\_PI;  pitch = (atan2(Xg, sqrt(Yg\*Yg + Zg\*Zg))\*180.0)/M\_PI; |  |

### 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  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66 | #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;  pixels[x][y] = HIGH;  x = 7 - map(roll, -20, 20, 0, 7);  y = map(pitch, -20, 20, 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);  }  } | c  b  a  d  e  f  g  i  h  j  k |

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.

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



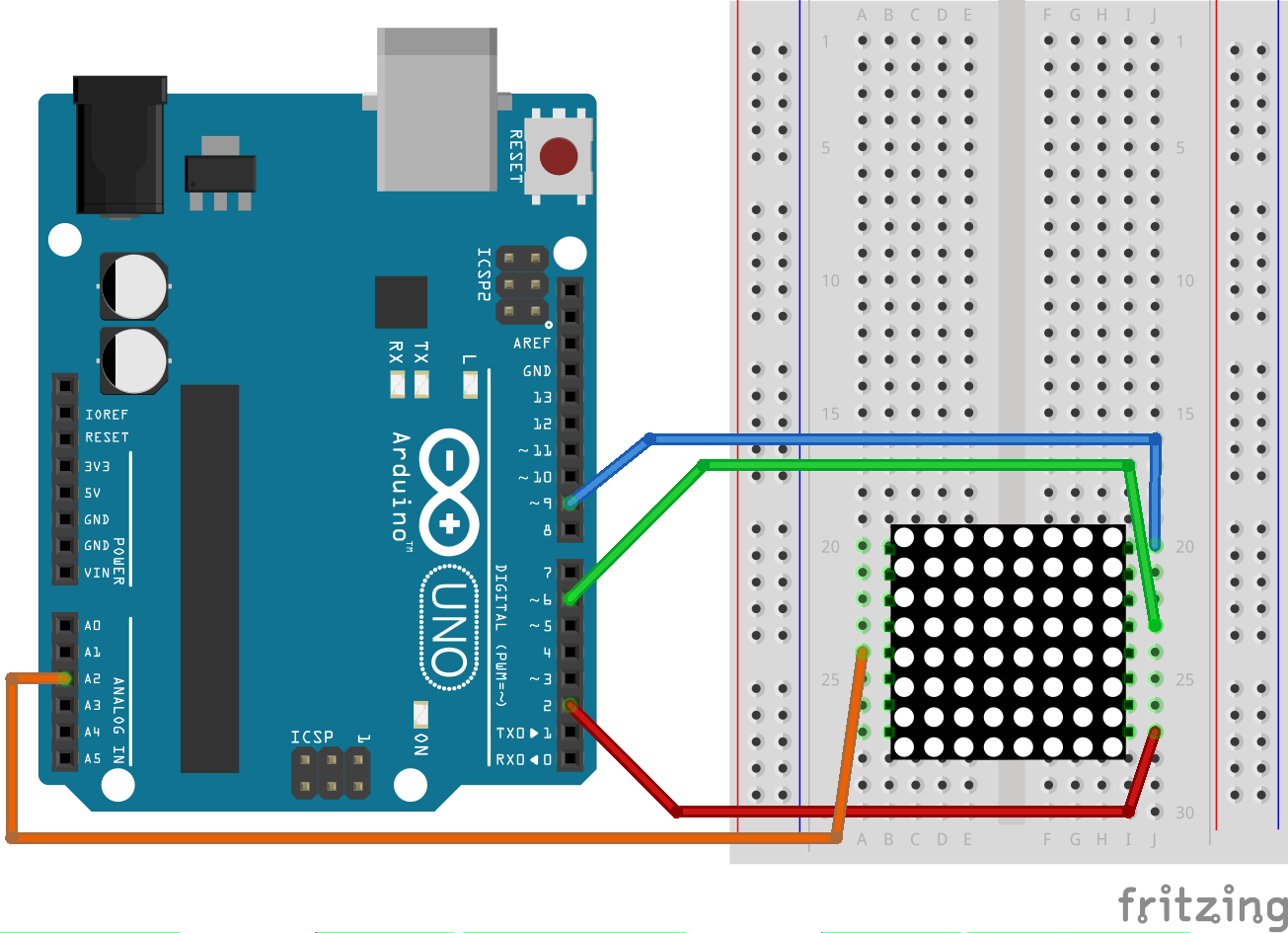
Z programového kódu je patrné, že při využití funkcí je změna kódu minimální. Prakticky došlo pouze ke změně ve funkci pro čtení ze senzorů, která spočívala v převedení získaných hodnot z akcelerometru na hodnoty použitelné k namapování pro maticový displej.



## 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. – Základní zapojení LED matice

|  |  |  |
| --- | --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28 | 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);  } |  |

### (Př. 1

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

|  |  |  |
| --- | --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28 | 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.

|  |  |  |
| --- | --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37 | 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.

|  |  |  |
| --- | --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33 | 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



|  |  |  |
| --- | --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21 | // 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.

|  |  |  |
| --- | --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69 | 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++){  refreshScreen(image);  }  for(int i = 0; i < 100; i++){  refreshScreen(imageS);  }  }  void refreshScreen(unsigned char dat[8][8]){  for(int j = 0; j<8;j++){  digitalWrite(col[j], LOW);  for(int k = 0; k<8; k++){  digitalWrite(row[k], dat[k][j]);  }  delay(1);  Clear();  }  }  void Clear(){  for(int i = 0; i<8; i++){  digitalWrite(row[i],LOW);  digitalWrite(col[i],HIGH);  }  } |  |

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 [↑](#footnote-ref-1)