Otrais solis Arduino programmēšanā

Ivars Driķis

2023. gada 27. oktobrī

1 Analogā signāla ievads

Arduino spēj vienlaicīgi nolasīt 6 analogos signālus, kas pieslēgti ieejām A0, A1, A2, A3, A4, A5 un A6. Signālam jābūt robežās no 0 līdz 5 voltiem. ACP izšķirtspēja ir 10 biti, tātad 5 volti atbilst nolasītajai vērtībai 1023. Attiecīgi viens solis ir apmēram 5 milivolti.

Kursa pirmsākumos kā mainīga analogā signāla avots tika izmantots džoistiks, zimējums 1. Tas sastāv no diviem potenciometriem un vienas spiedpogas ar pull-up rezistoru. Nodarbību laikā izrādījās, ka ar džoistiku uzstādīt nepieciešamo signālu ir neērti, tāpēc komplektiem tika iegādāti parasti potenciometri, zimējums 1. Komplektos, kas tiek izmantoti nodarbibās, ir gan potenciometri gan arī džoistiki.

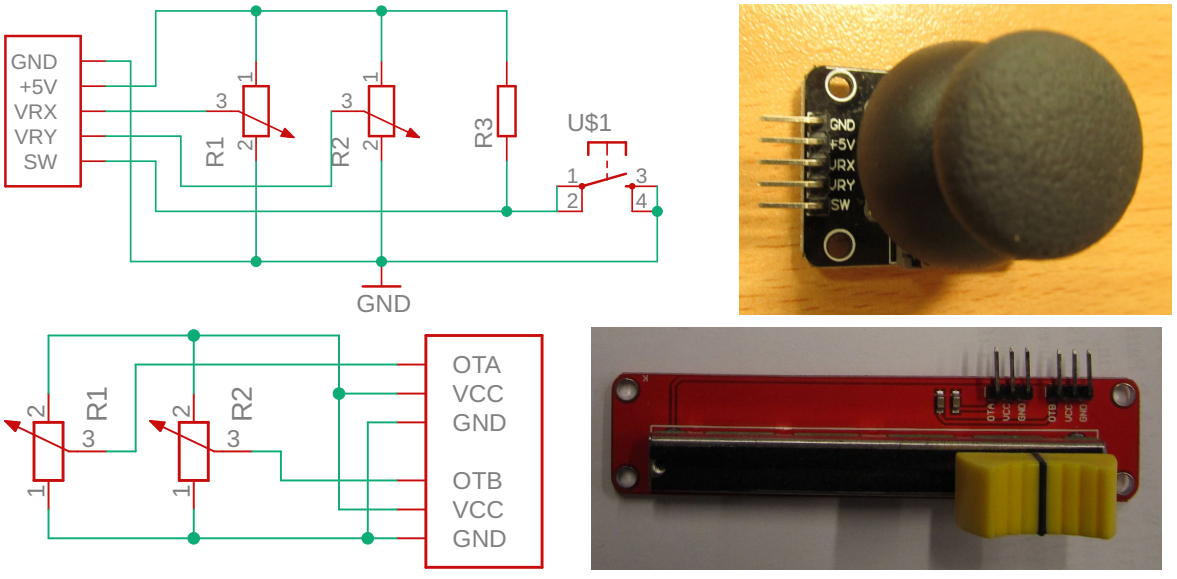
1.1 Vienkāršs analogā signāla ievads ar potenciometru

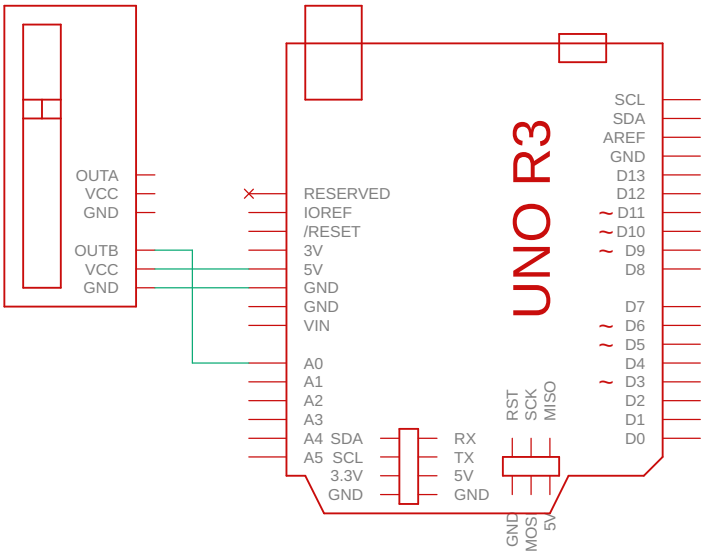
Saslēdzam shēmu atbilstoši zīmējumam 2. Šeit analogo signālu veidojam izmantojot potenciometru. Bīdām potenciometru un novērojam

Listing 1: AnalogPrint.ino

```
1 const int pinAnalog = A0;
2
3 void setup( void )
4 {
5   Serial.begin(9600);
```

Att. 1: Džoistika panelis un Potenciometra panelis, to shēmas





Att. 2: Vienkāršs analogā signāla ievads izmantojot potenciometru

```
6 }
7
8 void loop( void )
9 {
10   int valInt = analogRead(pinAnalog);
11   double valU = 5./1024.*valInt;
12
13   Serial.print( valInt );
14   Serial.print( " " );
15   Serial.println(valU, 2);
16   delay( 100 );
17 }
```

Šeit izmantota analogRead funkcija

analogRead(pin) - lasa analogā signāla līmeni no norādītās ieejas, [1]. Atgriež skaitli robežas no 0 līdz 1023, kas nozīmē spriegumu uz ieejas no 0 līdz 5 voltiem.

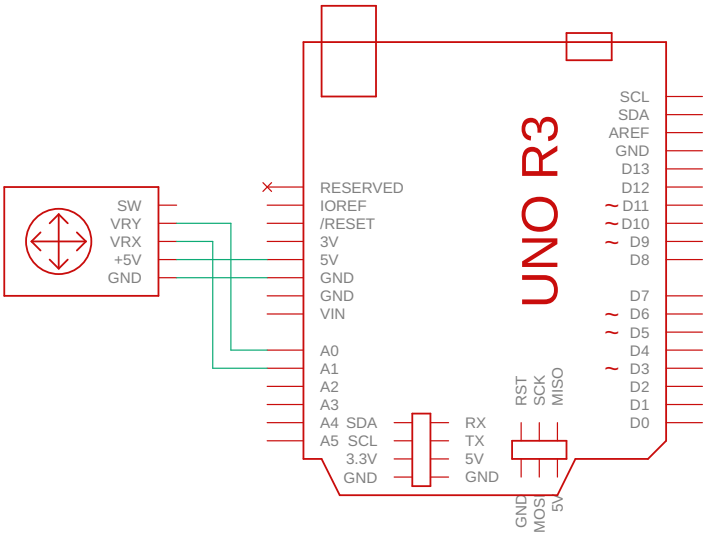
pin: analogās ieejas numurs

1.2 Divkāršs analogā signāla ievads ar džoistiku

Izamtojot džoistiku, var ielasīt uzreiz divus analogos signālus. Saslēdzam shēmu atbilstoši zīmējumam 3 un darbinām programmu

Listing 2: AnalogPrint2.ino

```
1 const int pinAnalog0 = A0;
2 const int pinAnalog1 = A1;
3
4 void setup( void )
5 {
6   Serial.begin(9600);
7 }
8
9 void loop( void )
10 {
```



Att. 3: Divkāršs analogā signāla ievads ar džoistiku

```
11    int valInt0 = analogRead(pinAnalog0);
12    int valInt1 = analogRead(pinAnalog1);
13    double valU0 = 5./1024.*valInt0;
14    double valU1 = 5./1024.*valInt1;
15
16    Serial.print( valInt0 );
17    Serial.print( " " );
18    Serial.println(valU0, 2);
19    Serial.print( valInt1 );
20    Serial.print( " " );
21    Serial.println(valU1, 2);
22    delay( 100 );
23 }
```

2 Ļoti ļoti daudz mirdzdiožu

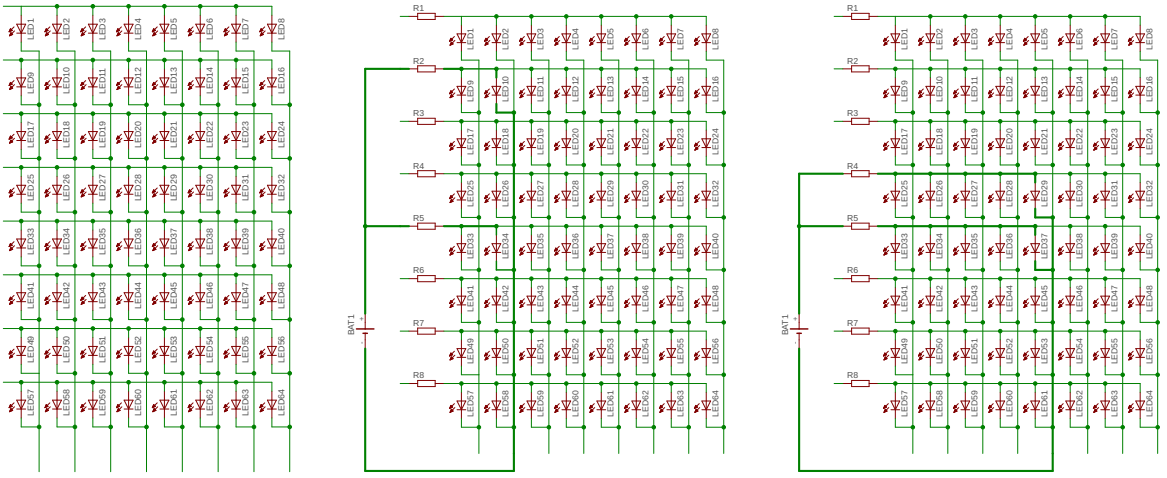
2.1 Kā pieslēgt ļoti daudz mirdzdiožu?

Ja nepieciešams vadīt ar Arduino vairākus 7 segmentu indikatorus vai arī vairākus desmitus mirdzdiožu[1], tad viegli izrēķināt, ka visu Arduino izeju nepietiek lai pieslēgtu nepieciešamo skaitu mirdzdiožu. Ne mazāk nozīmīga problēma ir arī milzīgais savienotājvadu skaits, kas šādai konstrukcijai nepieciešams. Risinājums ir asprātīgs un vienkāršs, pie kam atrisina abas problēmas (sk. 4 attēlā pa kreisi): ir "horizontālie" vadi un "vertikālie" vadi un to krustpunktos ievietotas mirdzdiodes kā tas redzams. Tas strādā, redzams blakus attēlos. Ja pie barošanas avota pieslēdz 2 "vertikālo" un 2 un 5 "horizontālo" līnijas, tad spīdēs tikai 10 un 34 mirdzdiodes. Savukārt, ja pie barošanas avota pieslēdz 5 "vertikālo" un 4 un 5 "horizontālo" līnijas, tad spīdēs tikai 29 un 37 mirdzdiodes. Tātad, lai vadītu 64 mirdzdiodes, pietiek ar 16 Arduino izejām kā arī ar tikai 16 vadiem! Atliek tikai pietiekoši ātri pārslēgt "vertikālās" līnijas, lai cilvēka acs nejustu pārtraukumus mirdzdiodes spīdēšanā.

Izskatīsim iespēju šādu matricu pieslēgt tieši Arduino izejām. Pieredze rāda, ka laba mirdzdiode pietiekami spilgti spīd patērējot 2 mA strāvu. Tā kā strāva diodei tiek padota

[1] Tādi bija mans pirmais un otrais Arduino projekti

Att. 4: 8x8 mirdzdiožu matrica un tās darbība.



tikai 1/8 no tās darbības laika, tad mirdzdiodei ir īslaicīgi jāpadod 16 mA. Diodes ir konstruētas, lai spētu ilgu laiku izturēt šādu darba režīmu. Tā kā vienai "vertikālajai" līnijai var nākties strādāt 8 diodēm, kas tai pieslēgtas, tad maksimālā strāva, kura jāiztur Arduino līnijai ir 128 mA. Dokumentācijā atrodam, ka maksimālā strāva, kura nedrīkst pārsniegt Arduino līnijā pieslēdzot to GND (sink current) ir 40 mA, tātad stipri par maz un tāpēc Arduino izejām ir japieslēdz ārējie tranzistori, kas šo strāvu var izturēt. Tas aizņem vietu un ir ķēpīgi montāžā. Tāpēc lietderīgāk ir izmantot specializētas mikroshēmas, piemēram MAX7219 [2], kas ne tikai nodrošina nepieciešamo strāvu "vertikālajās" un "horizontālajās" līnijas, bet arī patstāvīgi veic nepieciešamās līniju pārslēgšanas, lai ieslēgtu tikai tās mirdzdiodes, kuras norādītas šās mikroshēmas iekšējā atmiņā. Savukārt informāciju par mirdzdiodēm uz dotās mikroshēmas atmiņu nogādā izmantojot tikai 3 datu līnijas. Rūpnieciski ražotu paneļu piemēri redzami 5 attēlā. Pa kreisi panelis, kas vada astoņus 7-segmentu indikatorus, pa labi viena no versijām 8x8 diožu matricas vadīšanai.

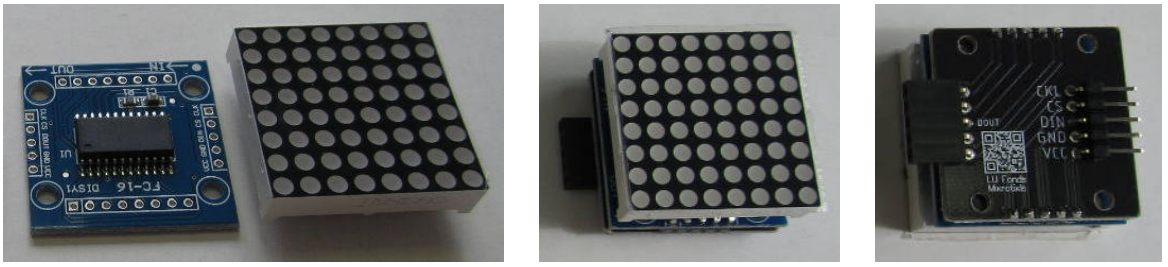
Ļoti iespaidīgi izskatās skrejošo uzrakstu lenta, kas izveidota no 8x8 diožu matricām. Tāpēc tika nolemts 5 attēlā redzamo 8x8 diožu matricas paneli papildināt ar papildus platīti, kas ļauj ērti un ātri izveidot diožu matricu lentu.

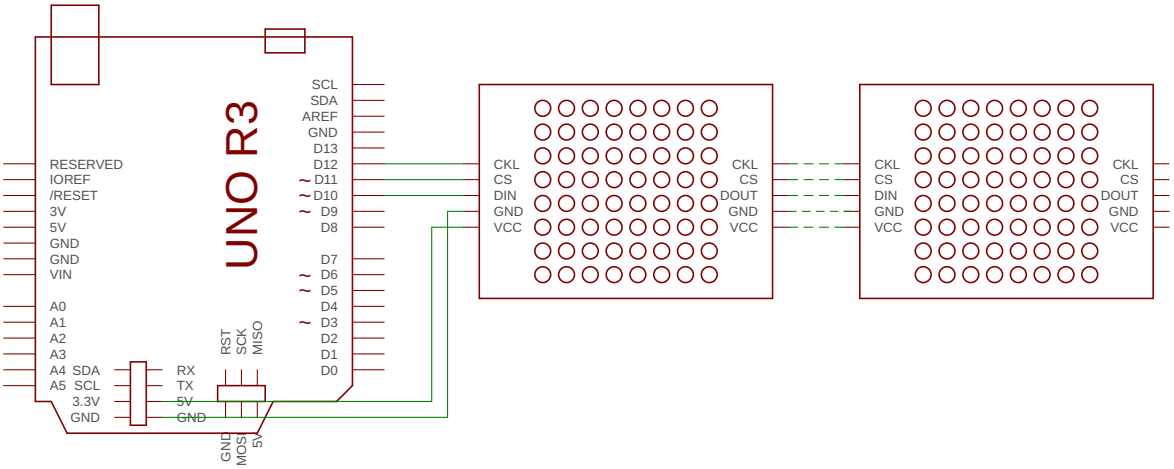
2.2 Vienkāršs 8x8 mirdzdiožu ekrāns

Lai vadītu 8x8 mirdzdiožu ekrānu izmantot MAX7219 mikroshēmu lietderīgi izmantot LedControl bibliotēku [3]. Saslēdzam shēmu kā redzams zīmējumā 6 un apskatīsim pašu vienkāršāko piemēru. Šeit soli pa solim tiek ieslēgtas katra no matricas mirdzdiodēm, pēc tam tādā pašā secībā tās tiek izslēgtas.

Tā kā lietosim LedControl bibliotēku, programmas pirmajā rindā iekļaujam failu LedControl.h

Att. 5: Augšējā attēlā rūpnieciski ražots MAX7219 panelis 8x8 diožu matricas vadīšanai





Att. 6: 8x8 mirdzdiožu ekrāna 8x8 pieslēgšana pie Arduino

no sistēmas bibliotēku kataloga.

Listing 3: led8x8.ino

```
1 #include <LedControl.h>
```

Pēc tam tipa dokumentācija kur un kā pieslēgt indikatoru.

```
3 const int pinDIN = 10;
4 const int pinCLK = 12;
5 const int pinCS = 11;
```

Tālāk kā globālo mainīgo definējam 8x8 mirdzdiožu ekrānu vadības objektu. Konstruktora pēdējais parametrs nozīmē cik indikatoru rindiņā mums būs saslēgts. Lasiet bibliotēkas aprakstu [3].

```
7 LedControl lc=LedControl(pinDIN, pinCLK, pinCS, 1);
```

Tālāk ērtības laban ieviesīsim mainīgo, kas visās funkcijās norādīs ar kuru no 8x8 mirdzdiožu ekrāniem strādāsim.

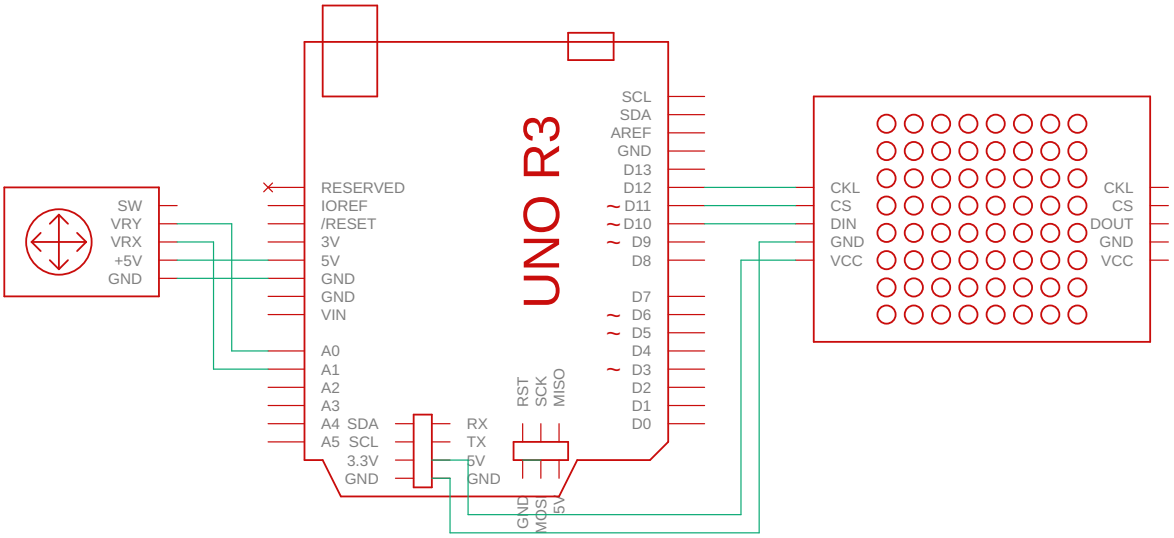
```
8 const int displ = 0;
```

Tālāk veicam virkni cermoniju. Vispirms restartējam indikatoru. Pēc tam uzstādam indikatora spīdēšanas intensitāti. Pamēģiniet uzminēt, kā to regulē! Pēc tam nodzēš visas spīddiodes.

```
10 void setup()
11 {
12   lc.shutdown(displ, false);
13   lc.setIntensity(displ, 8);
14   lc.clearDisplay( displ );
15 }
```

Pēc tam pakāpeniski ieslēdzam spīddiodes vienu pēc otras. Katru diodi slēdz individuāli norādot indikatora numuru, kolonnu un rindu. Pēdējais parametrs parāda ieslēdzam mēs vai izslēdzam.

```
17 void loop()
18 {
```



Att. 7: Džoistiks un 8x8 mirdzdiožu ekrāns

```
19    for(int row=0; row<8; row++) {
20      for(int col=0; col<8; col++) {
21        lc.setLed(displ,col,row,true);
22        delay(25);
23      }
24    }
```

Pēc tam tieši tādā paša veidā izslēdzam diodes vienu pēc otras.

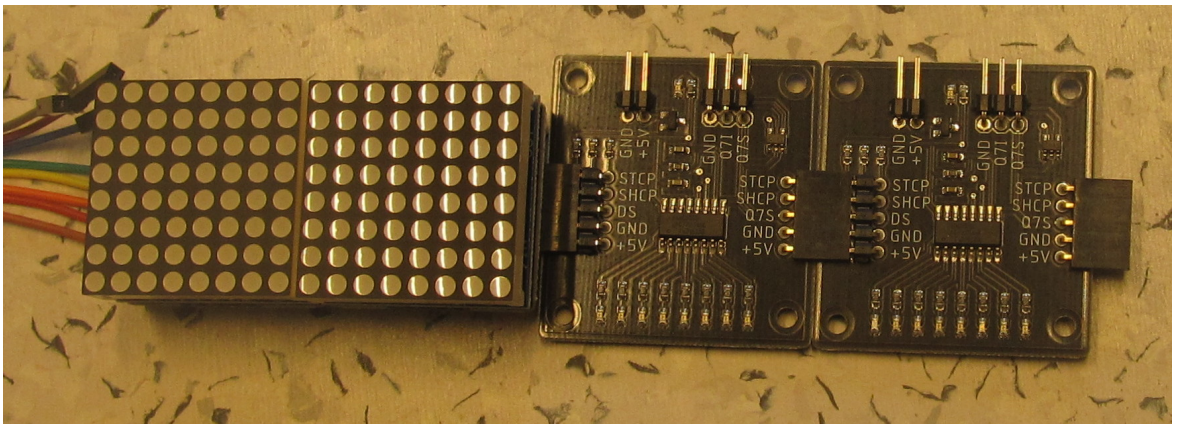
```
26    for(int row=0; row<8; row++) {
27      for(int col=0; col<8; col++) {
28        lc.setLed(displ,col,row,false);
29        delay(25);
30      }
31    }
32  }
```

3 Punkts uz diožu ekrāna

Pieslēdzam pie Arduino džoistiku un 8x8 ekrānu, kā tas darīts zīmējumā 7. Izveidosim programmu, ar kuras palīdzību var ”vadīt” spīdošu diodi pa 8x8 mirdzdiožu ekrānu.

Listing 4: you8x8.ino

```
1 #include <LedControl.h>
2
3 const int pinAnalog0 = A0;
4 const int pinAnalog1 = A1;
5 const int pinDIN = 10;
6 const int pinCLK = 12;
7 const int pinCS = 11;
8
9 LedControl lc=LedControl(pinDIN, pinCLK, pinCS, 1);
10 const int displ = 0;
11
12 void setup()
```



Att. 8: Divu 8x8 mirdzdiožu ekrāni un divi nobīdes reģistri

```
13 {
14   lc.shutdown(displ, false);
15   lc.setIntensity(displ, 8);
16   lc.clearDisplay( displ );
17 }
18
19 void loop()
20 {
21   int vi0 = analogRead(pinAnalog0);
22   int vi1 = analogRead(pinAnalog1);
23   lc.setLed(displ, vi0/128, vi1/128, true);
24   delay(50);
25 }
```

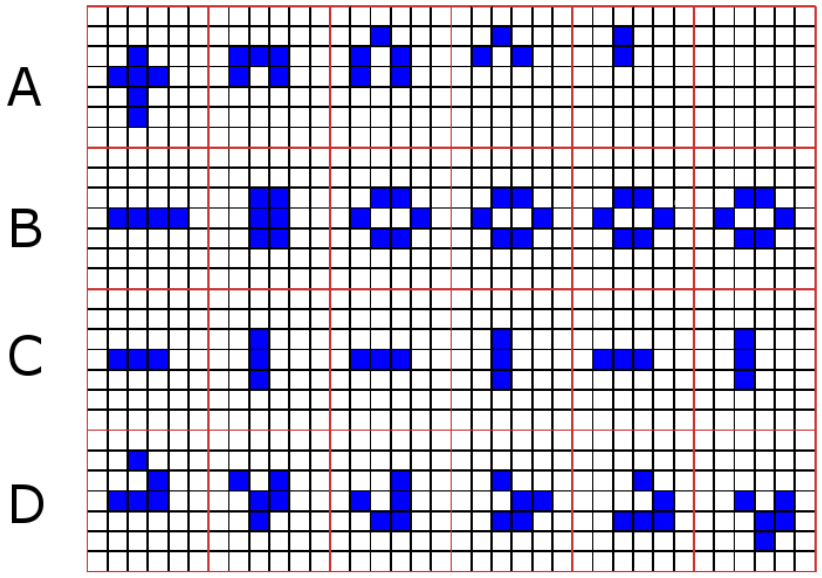
4 Uzdevums patstāvīgajam darbam

1. Pievienojam ortu 8x8 mirdzdiožu ekrānu kā tas redzams zīmējumā 6.
   (a) Veicot izmaiņas tikai divās no programmas 3 rindiņām, iedarbiniet šo otro 8x8 mirdzdiožu ekrānu.
   (b) Kas tajā laikā notiek ar pirmo ekrānu?
   (c) Pārveidojiet programmu 3 tā, lai pakāpeniski tiktu aizpildīti un notīrīti abi 8x8 mirdzdiožu ekrāni.
   (d) Pievienot divu 8x8 mirdzdiožu ekrānu galā divus nobīdes reģistrus kā tas redzams zīmējumā 8 Novērojam!

5 Konveja dzīves spēle

Izaicinājums ir izrakstīt *Konveja dzīves spēli* priekš vairākiem šādiem 8x8 mirdzdiožu ekrāniem. Dzīve jeb Konveja dzīves spēle (Conway's Game of Life) ir angļu matemātiķa Džona Konveja 1970. gadā izveidots šūnu automāts [4]. Spēles noteikumus pārrakstu no Wikipēdijas,

Spēles lauciņš jeb "visums" ir bezgalīga rūtiņās sadalīta virsma (tā var būt arī noslēgta; spēles datorversijās visbiežāk izmanto tora virsmu). Katra virsmas rūtiņa jeb šūna var būt vai nu dzīva (iekrāsota), vai mirusi. Katrai šūnai ir astoņi kaimiņi (gan tās šūnas, kas



Att. 9: Dažas vienkāršas figūras Konveja dzīves spēlē, [4]

saskaras ar malām, gan tās, kas saskaras ar stūriem). Uz lauciņa esošās dzīvās šūnas spēles sākumā sauc par pirmo paaudzi. Katra nākamā paaudze tiek aprēķināta pēc vienkāršiem noteikumiem:

- tukša (mirusi) šūna, kam ir tieši 3 dzīvi kaimiņi, atdzīvojas;
- ja dzīvai šūnai ir 2 vai 3 dzīvi kaimiņi, tā turpina dzīvot, pretējā gadījumā tā mirst no "vientulības" vai "pārapdzīvotības".

Spēlētājs pats nepiedalās spēlē, bet tikai nosaka pirmo paaudzi, kura tālāk "evolucionē" bez viņa līdzdalības.

Lai arī spēles noteikumi ir ļoti vienkārši, tajā ir milzīga formu daudzveidība un tās izpēte joprojām turpinās. Tā var kalpot kā evolūcijas modelis un teorētiski ar šīs spēles palīdzību varētu modelēt skaitļošanas sistēmas. Attēlā 9 parādītas dažas vienkāršas figūras:

A - figūra, kas pilnīgi mirst piektajā paaudzē

B - figūra, kas trešajā paaudzē kļūst par stabilu figūru

C - vienkāršākā periodiskā figūra

D - vienkāršākā kustīgā periodiskā figūra

Listing 5: GameLife.ino

```
1 #include <LedControl.h>
2
3 const int pinCLK = 12;
4 const int pinCS  = 11;
5 const int pinDIN = 10;
6
7 LedControl lc=LedControl(pinDIN, pinCLK, pinCS, 1);
```

```
byte s2[8][8], s1[8][8] = {
  {0, 0, 0, 0, 0, 0, 0, 0},
  {0, 0, 1, 0, 0, 0, 0, 0},
  {0, 0, 0, 1, 0, 0, 0, 0},
  {0, 1, 1, 1, 0, 0, 0, 0},
  {0, 0, 0, 0, 0, 0, 0, 0},
  {0, 0, 0, 0, 0, 0, 0, 0},
  {0, 0, 0, 0, 0, 0, 0, 0},
  {0, 0, 0, 0, 0, 0, 0, 0},
};

void setup()
{
  Serial.begin( 9600 );
  lc.shutdown(0, false);
  lc.setIntensity(0, 8);
  lc.clearDisplay( 0 );

  for(int col=0; col<8; col++) {
    for(int row=0; row<8; row++) {
      lc.setLed(0, col, row, s1[col][row]);
      s2[col][row] = s1[col][row];
    }
  }
}

void loop( void )
{
int row, col, sum, i, j;

  for(col=0; col<8; col++) {
    for(row=0; row<8; row++) {
      for(sum=0, i=col-1; i<=col+1; i++)
        for(j=row-1; j<=row+1; j++) {
          if((i==col) && (j == row)) continue;
          sum += s1[i&0x07][j&0x07];
        }

      if( ((sum==2)&&(s1[col][row]==1)) || (sum==3) )
        s2[col][row] = 1;
      else
        s2[col][row] = 0;
    }
  }

  for(row=0; row<8; row++) {
    for(col=0; col<8; col++) {
      //if( s1[col][row] != s2[col][row] )
        lc.setLed(0, col, row, s1[col][row]);
      s1[col][row] = s2[col][row];
    }
  }

  delay( 1000 );
}
```

Literatūras saraksts

[1] Arduino Reference. analogRead(). https://www.arduino.cc/reference/en/language/

functions/analog-io/analogread/. [Online; accessed 2022.09.25].

[2] Serially Interfaced, 8-Digit LED Display Drivers MAX7219/MAX7221. https://datasheets.maximintegrated.com/en/ds/MAX7219-MAX7221.pdf. [Online; accessed 2022.09.25].

[3] Arduino Reference. LedControl library. https://www.arduino.cc/reference/en/libraries/ledcontrol/. [Online; accessed 2022.09.25].

[4] Conway's Game of Life. https://en.wikipedia.org/wiki/Conway%27s_Game_of_Life. [Online; accessed 2017.02.10].