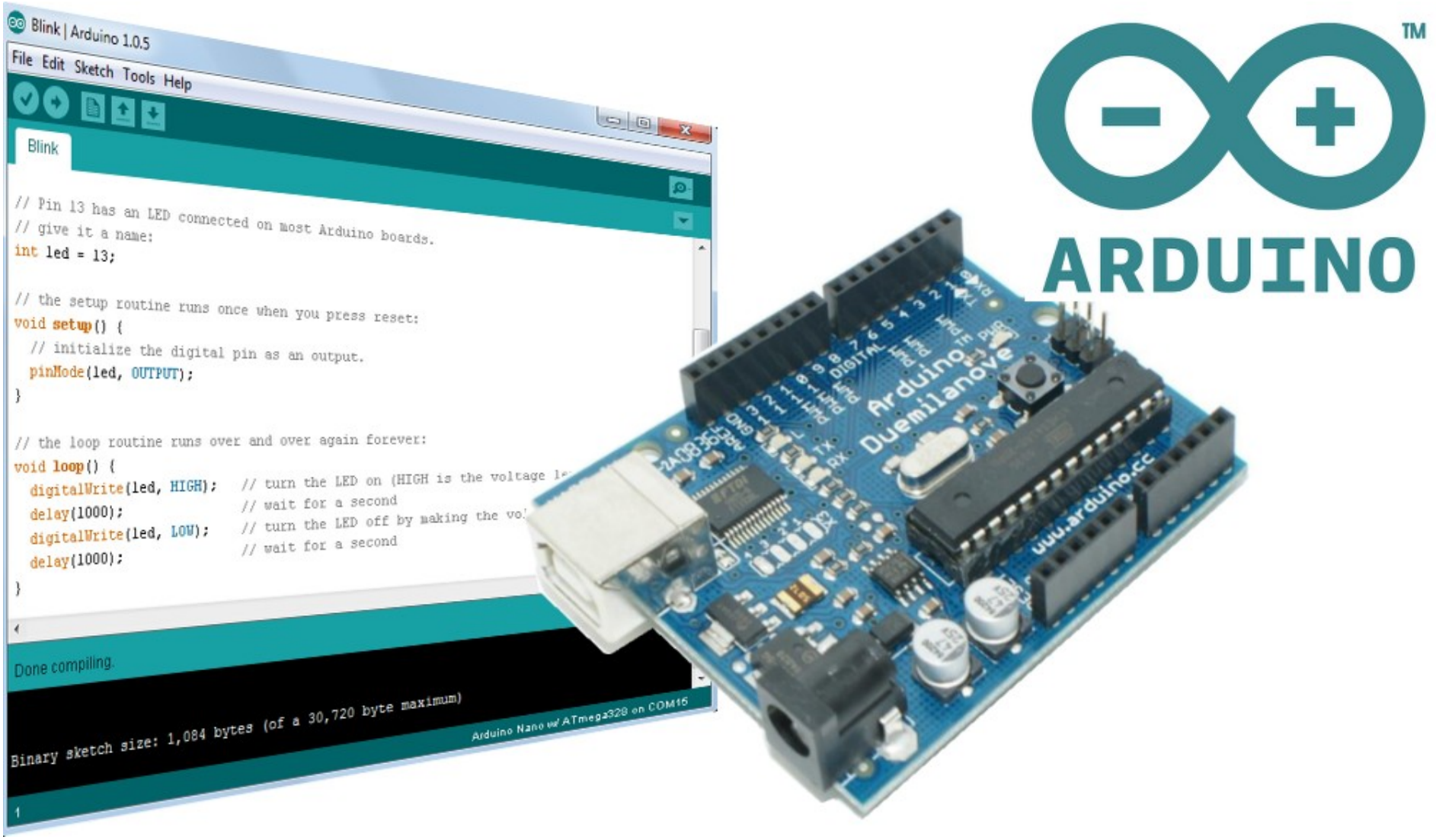
Bevezetés az elektronikába

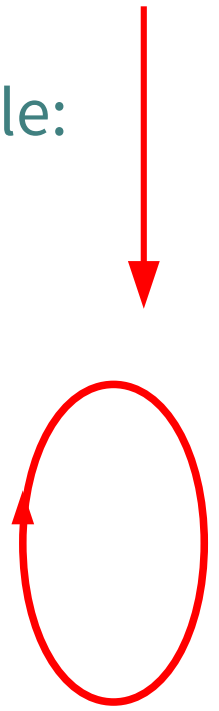


9. Arduino programozás – digitális kimenetek vezérlése

Az Arduino programok szerkezete

- Minden **Arduino** program kötelező elemei:
  - **setup()** függvény (előkészítés)
    Csak egyszer fut le bekapcsoláskor, vagy újraindításkor
  - **loop()** függvény (programhurok)
    Vég nélkül ismétlődik
- A { } kapcsos zárójelpár a **C** programnyelvhez hasonlóan blokkba foglalja az utasításokat
- Függvénynév előtt a **void** szó azt jelzi, hogy a függvény nem ad vissza eredményt (eljárás)
- A **setup** és **loop** függvényeknek nincs bemenő paramétere, ezért a ( ) zárójelek közé nem írunk semmit

```
void setup() {
  // csak egyszer futnak le:
  utasítások;
}

void loop() {
  // ismétlődő rész:
  utasítások;
}
```



Digitális kimenetek vezérlése

- A **D1 – D13** digitális kivezetésekre sorszámukkal hivatkozhatunk (a ***D*** betű elhagyásával)
- Az **A0 – A5** kivezetések választásunk szerint analóg bemenetként vagy digitális ki/bemenetként is használhatók. Digitális kivezetésként **D14 – D19** a nevük, de ez a kártyán nincs feltüntetve
- A digitális kivezetések üzemmódja lehet kimenet (`OUTPUT`) vagy bemenet (`INPUT`). Használatuk előtt be kell állítani a kívánt üzemmódot a `pinMode(pin,mode)` paranccsal. Például:

```
pinMode(13,OUTPUT);   // Kimenetre állítja a D13 kivezetést
```

- A `digitalWrite(pin,state)` paranccsal állíthatjuk a digitális kimenetek állapotát magas, illetve alacsony szintre. Például:

```
digitalWrite(13,HIGH); // Magasra szintre állítja D13-at
digitalWrite(13,LOW);  // Alacsony szintre állítja D13-at
```

LED villogtatása: Blink.ino

- Az alábbi programmal a kártyára épített felhasználói LED-et (L) villogtatjuk a `LED_BUILTIN` (D13) kimenet le/fel kapcsolgatásával
- A villogtatás ütemét programozott késleltetésekkel szabályozzuk

```
// A setup függvény csak egyszer fut le, induláskor
void setup() {
  pinMode(LED_BUILTIN, OUTPUT);      // D13 legyen kimenet!
}

// A loop függvény vég nélkül, újra és újra lefut
void loop() {
  digitalWrite(LED_BUILTIN, HIGH);   // bekapcsoljuk a LED-et (magas szint)
  delay(1000);                       // 1000 ms (1 s) várakozás
  digitalWrite(LED_BUILTIN, LOW);    // kikapcsoljuk a LED-et (alacsony szint)
  delay(1000);                       // 1000 ms (1 s) várakozás
}
```

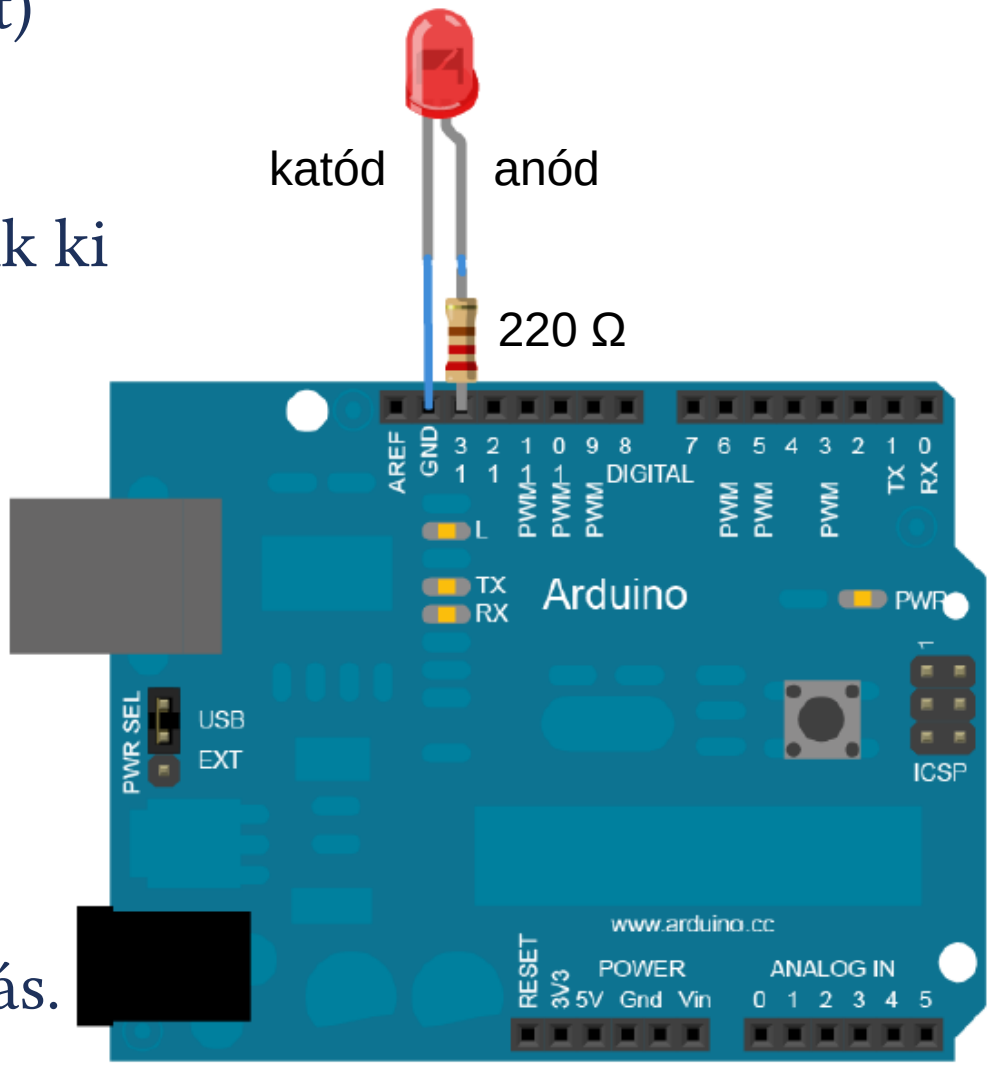
- A `delay()` függvény paramétere a várakozási idő, amit milliszekundumokban (ezredmásodpercekben) kell megadni

LED villogtatása: Blink.ino

- Az Arduino kártyához külső LED-et is kapcsolhatunk (ne feledkezzünk meg az áramkorlátozó ellenállásról, 220 Ω – 1 kΩ közötti érték javasolt)

Kísérletezzünk!

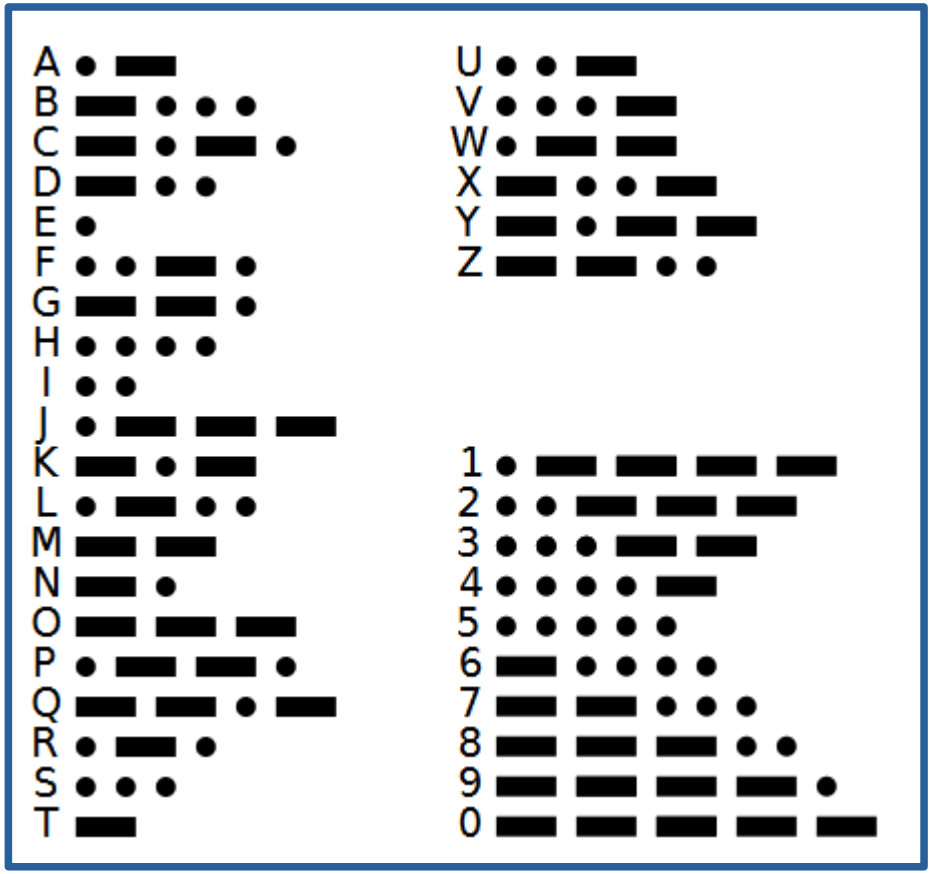
- A **D13** kivezetés helyett próbáljunk ki más kivezetést is! **D0** és **D1** a soros kommunikáció számára fentartott kivezetések (ezeken történik a programletöltés is), ezeket ne használjuk!
- Változtassunk a késleltetések időtartamán!
  Például: bekapcsolás után 100 ms, kikapcsolás után 2900 ms várakozás.



SOS Morze jeladó

- Az időzítések variálásával Morze jeladót is építhetünk (továbbra is a **D13** kimenetet kapcsolgatjuk)
- A Morze kódokban (Samuel Morse) minden karaktert rövid és hosszú jelek (pont és vonás) valamilyen kombinációja képviseli
- SOS = ... - - - ...
- Időzítések:
  - A pont időtartama 1 egység
  - A vonás időtartama 3 egység
  - Jelek közötti szünet 1 egység
  - Betűk közötti szünet 3 egység
  - Szavak kötötti szünet 7 egység
  - 10 WPM (szó / perc) ütem esetén 1 egység = 120 ms

Morze kódtábla



Morse_SOS.ino

- Ha nem akarunk sokat körmölni, meg kell tanulnunk függvényt definiálni!
- A pontot és a vonást előállító `ti()` és `ta()` függvényekkel gyerekjáték a program összeállítása

```
#define t_pont    120              // Egy pont ideje (120 ms)
#define t_vonas   3*t_pont         // Egy vonás időtartama = 3 pont
#define t_betukoz 2*t_pont         // Valójában 3*t_pont - t_pont
#define t_szokoz  6*t_pont         // Valójában 7*t_pont - t_pont

void setup() {
  pinMode(LED_BUILTIN, OUTPUT);    // A LED_BUILTIN láb legyen kimenet
}

void loop() {
  ti(); ti(); ti(); delay(t_betukoz);
  ta(); ta(); ta(); delay(t_betukoz);
  ti(); ti(); ti(); delay(t_szokoz);
}
```

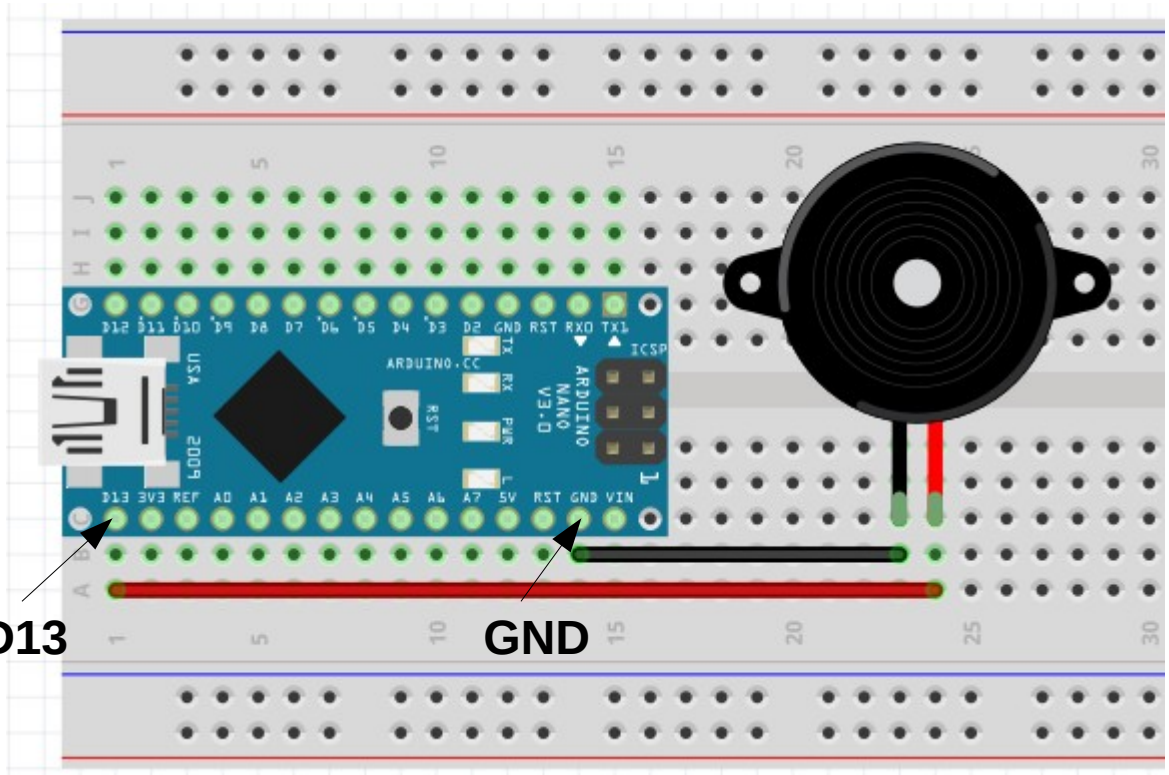
Morse_SOS.ino

- Függvényeinknek nincs sem bemenő paramétere, sem visszatérési értéke, ezt jelzik a „void” kulcsszavak
- A függvény neve után { } jelek között adjuk meg a végrehajtani kívánt utasításokat (a leírás sorrendjében lesznek végrehajtva)

```
// Egy pont generálása
void ti(void) {
  digitalWrite(LED_BUILTIN, HIGH);   // bekapcsoljuk a LED-et
  delay(t_pont);                     // 120 ms (1 pont) várakozás
  digitalWrite(LED_BUILTIN, LOW);    // kikapcsoljuk a LED-et
  delay(t_pont);                     // 120 ms (1 pont) várakozás
}


// Egy vonás generálása
void ta(void) {
  digitalWrite(LED_BUILTIN, HIGH);   // bekapcsoljuk a LED-et
  delay(t_vonas);                    // 360 ms (3 pont) várakozás
  digitalWrite(LED_BUILTIN, LOW);    // kikapcsoljuk a LED-et
  delay(t_pont);                     // 120 ms (1 pont) várakozás
}
```

SOS Morze jeladó - hanggal

- Kössünk egy „folytonos” piezo csipogót a **D13** kimenet és a **GND** kivezetés közé! Ügyeljünk a polaritásra: a csipogó + jelzésű lábát kössük a **D13** kimenetre! Ekkor a Morze adást hallani is fogjuk
- **Ne tegyük vele hosszú ideig próbára családtagjaink türelmét!**

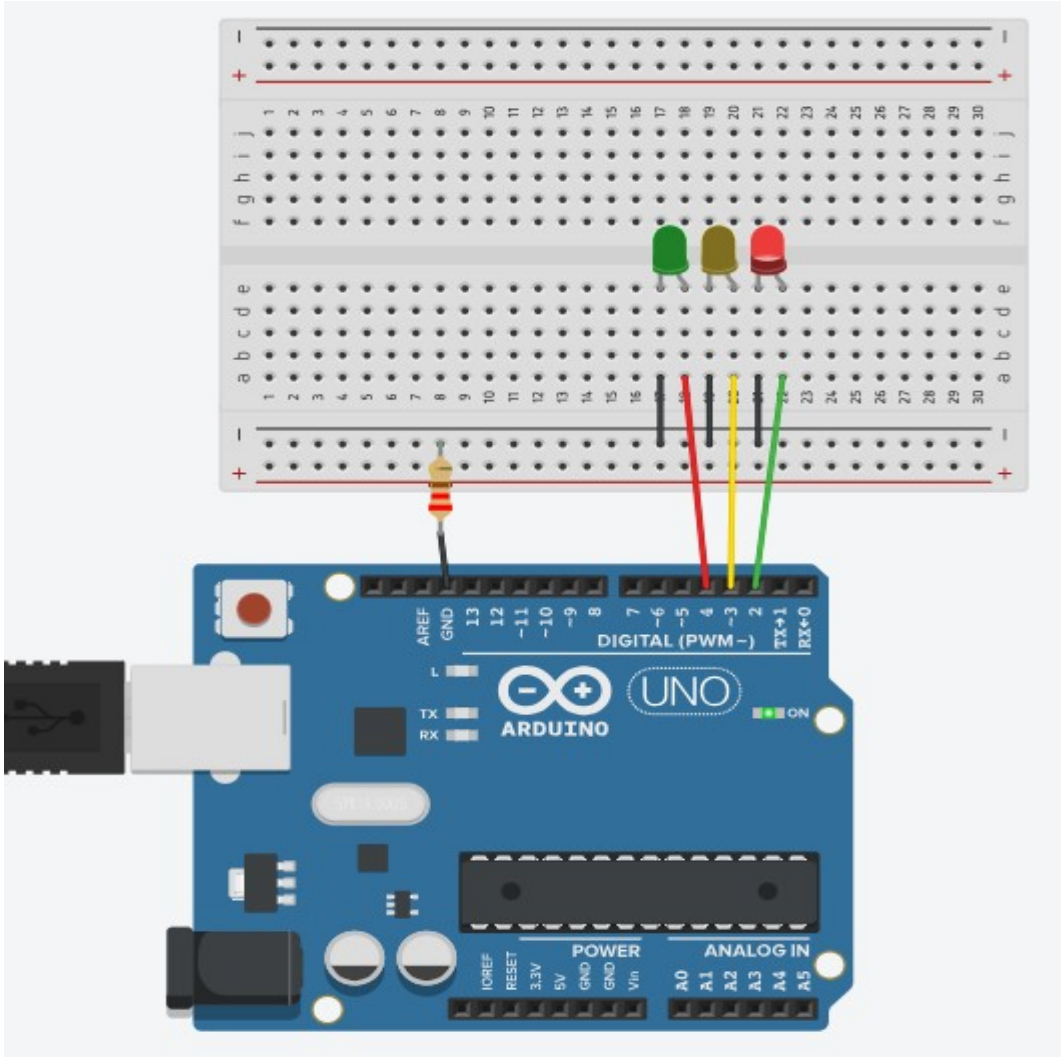


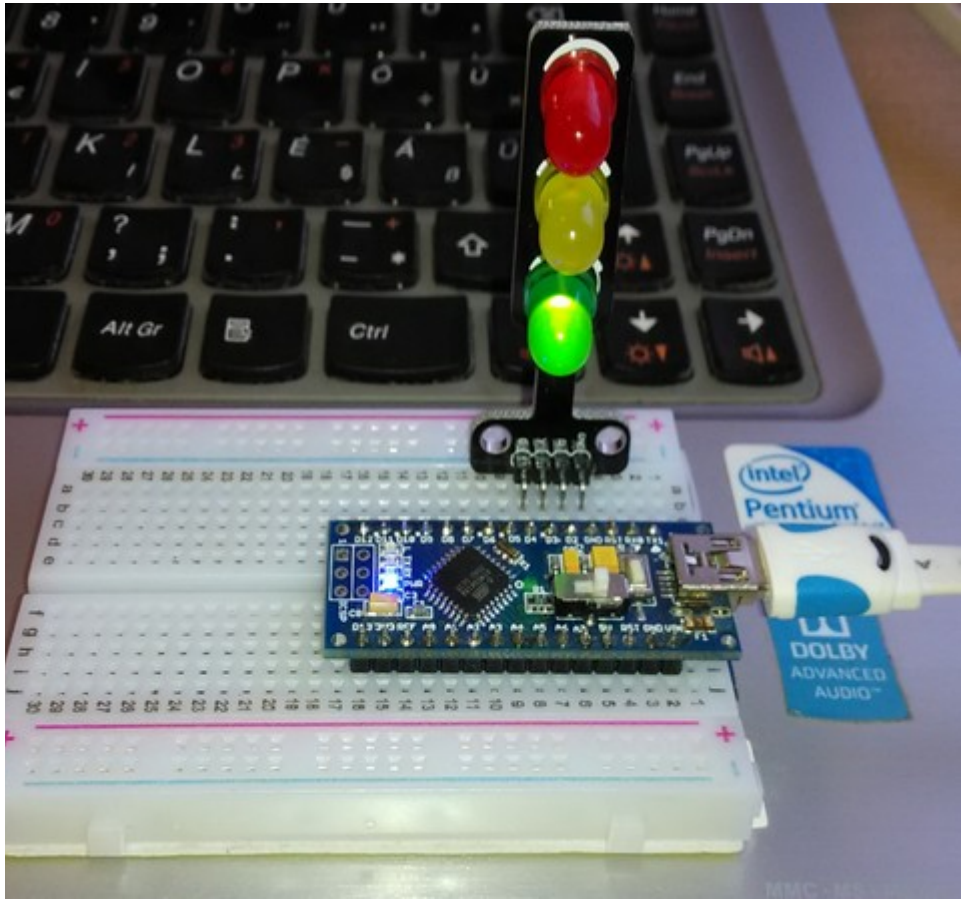
A „folytonos” csipogó saját oszcillátorral rendelkezik.

Ha tápfeszültséget kap, folyamatosan sípol

Egyszerű közlekedési lámpa

- Kössük három LED anódját a **D2, D3, D4** kimenetekre, összekötött katódjaikat pedig egy áramkorlátozó ellenálláson keresztül a **GND** kivezetésre!





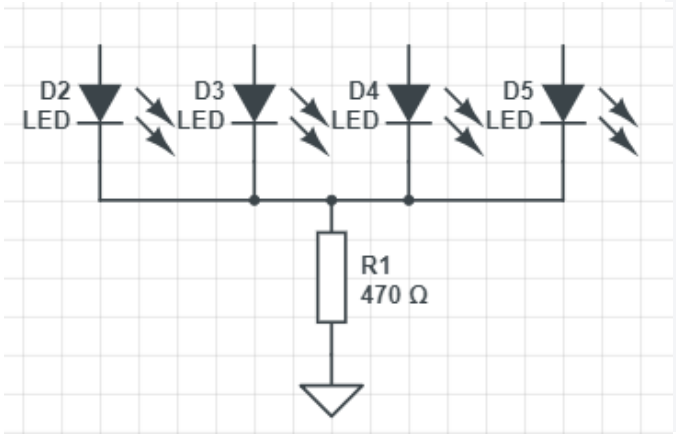
traffic_simple.ino

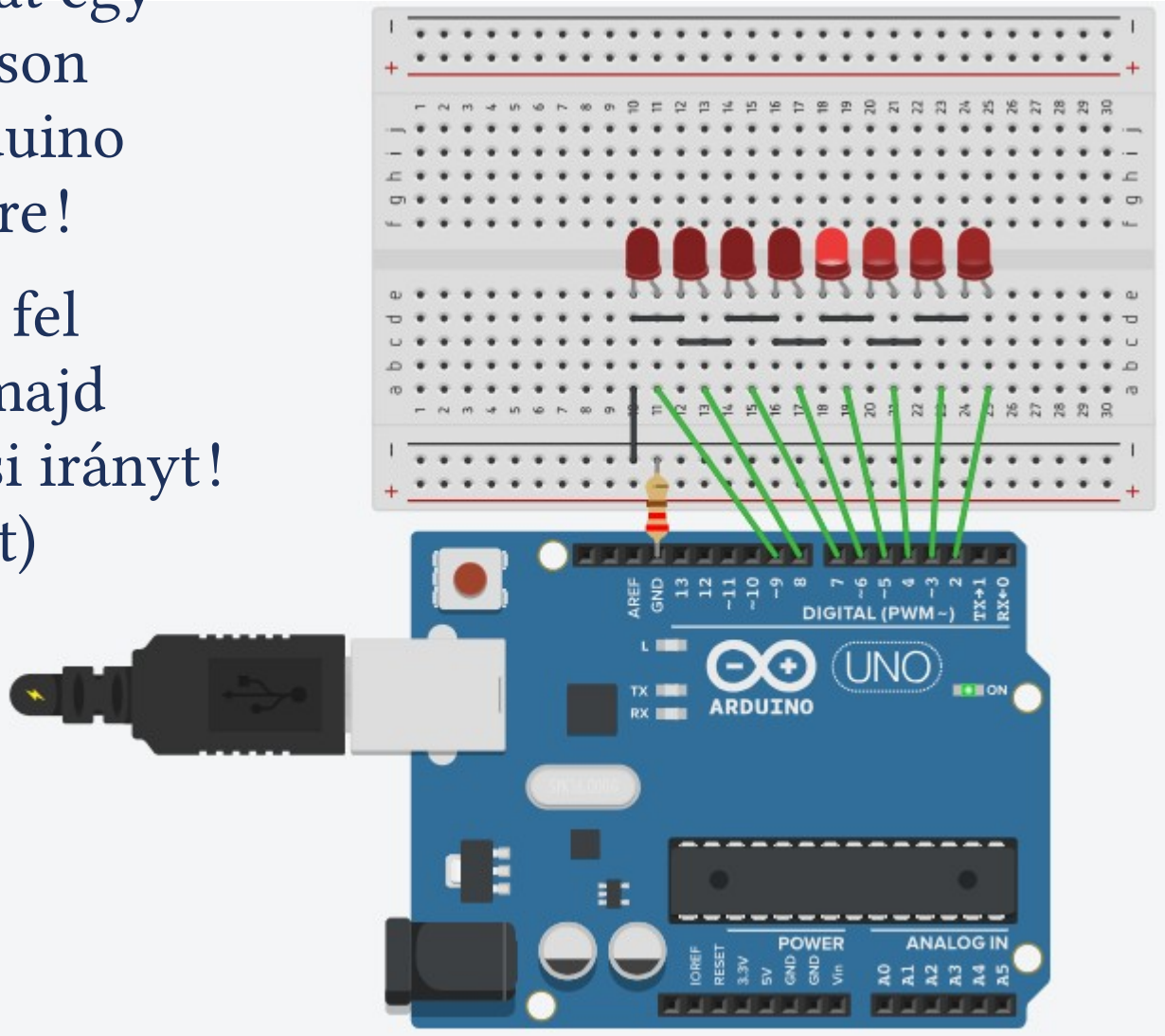
```
#define redled    2                 // Piros led D2-re kötve
#define yellowled 3                 // Sárga led D3-re kötve
#define greenled  4                 // Zöld led D4-re kötve
void setup() {
  pinMode(redled, OUTPUT);          // A redled kivezetés legyen kimenet!
  pinMode(yellowled, OUTPUT);       // A yellowled kivezetés legyen kimenet!
  pinMode(greenled, OUTPUT);        // A greenled kivezetés legyen kimenet!
}
void loop() {
  digitalWrite(redled,HIGH);        // Tilosra vált a jelző (piros fény)
  delay(9000);
  digitalWrite(redled,LOW);         // 9 s várakoz után lekapcsoljuk
  digitalWrite (yellowled,HIGH);    // Sárga jelzés
  delay (3000);                     // 3 másodpercig
  digitalWrite(yellowled,LOW);
  digitalWrite(greenled,HIGH);      // Folyamatos zöld jelzés 9 másodpercig
  delay (9000);
  digitalWrite(greenled,LOW);     delay (500);    // Zöld jelzés villogni kezd
  digitalWrite(greenled,HIGH);    delay (500);
  digitalWrite(greenled,LOW);     delay (500);
  digitalWrite(greenled,HIGH);    delay (500);
  digitalWrite(greenled,LOW);     delay (500);
  digitalWrite(greenled,HIGH);    delay (500);
  digitalWrite(greenled,LOW);     delay (500);
  digitalWrite(greenled,HIGH);    delay (500);
  digitalWrite(greenled,LOW);     // Zöld jelzés vége
}
```

A Blink.ino-nál tanultak alapján a közlekedési lámpát működtető programot sem nehéz megírni. Az ismétlődő részek ciklusba szervezésére tanulunk majd jobb módszert!

LED futófény

- Kössünk 8 db LED anódot a **D2 – D9** kimenetekre!
- Az összekötött katódokat egy áramkorlátozó ellenálláson keresztül kössük az Arduino kártya **GND** kivezetésére!
- Egymás után villantsuk fel 100 ms-ra a LED-eket, majd fordítsuk meg a haladási irányt! (Knight Rider fényeffekt)



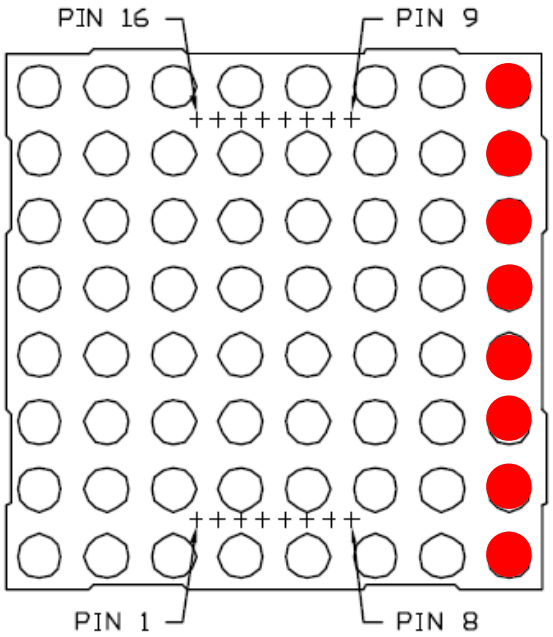


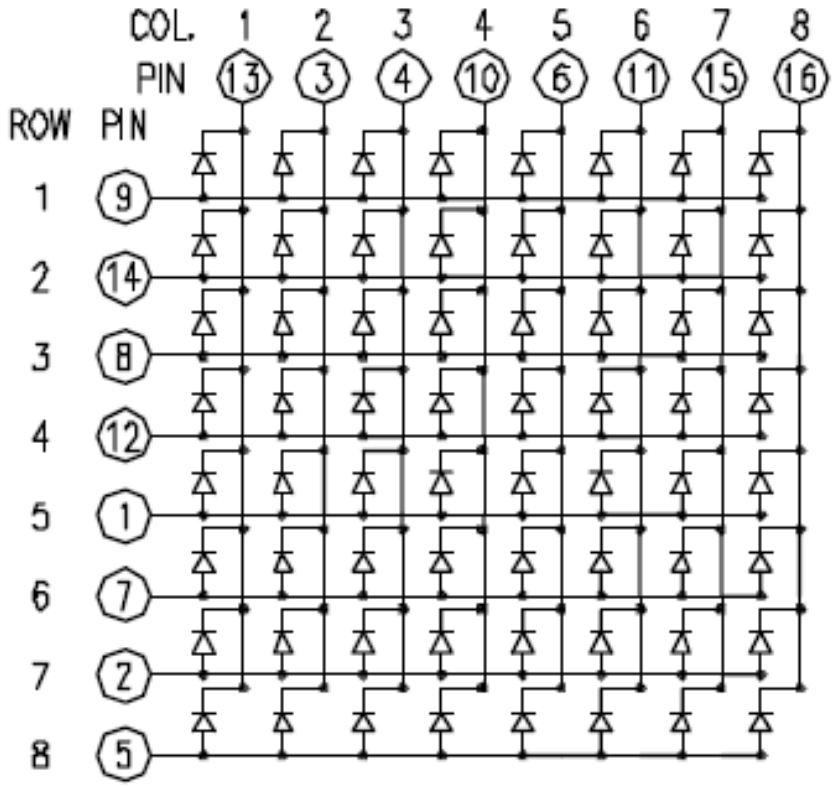
LED_futofeny.ino

```
void setup() {
  pinMode(2, OUTPUT);
  pinMode(3, OUTPUT);
  pinMode(4, OUTPUT);
  pinMode(5, OUTPUT);
  pinMode(6, OUTPUT);
  pinMode(7, OUTPUT);
  pinMode(8, OUTPUT);
  pinMode(9, OUTPUT);
}
void loop() {
  // pásztázás előre
  digitalWrite(2, HIGH); delay(100);
  digitalWrite(2, LOW);  delay(50);

  digitalWrite(3, HIGH); delay(100);
  digitalWrite(3, LOW);  delay(50);

  digitalWrite(4, HIGH); delay(100);
  digitalWrite(4, LOW);  delay(50);

  digitalWrite(5, HIGH); delay(100);
  digitalWrite(5, LOW);  delay(50);

  digitalWrite(6, HIGH); delay(100);
  digitalWrite(6, LOW);  delay(50);

  digitalWrite(7, HIGH); delay(100);
  digitalWrite(7, LOW);  delay(50);

  digitalWrite(8, HIGH); delay(100);
  digitalWrite(8, LOW);  delay(50);

  digitalWrite(9, HIGH); delay(100);
  digitalWrite(9, LOW);  delay(50);

  digitalWrite(8, HIGH); delay(100);
  digitalWrite(8, LOW);  delay(50);

  digitalWrite(7, HIGH); delay(100);
  digitalWrite(7, LOW);  delay(50);

  digitalWrite(6, HIGH); delay(100);
  digitalWrite(6, LOW);  delay(50);

  digitalWrite(5, HIGH); delay(100);
  digitalWrite(5, LOW);  delay(50);

  digitalWrite(4, HIGH); delay(100);
  digitalWrite(4, LOW);  delay(50);

  digitalWrite(3, HIGH); delay(100);
  digitalWrite(3, LOW);  delay(50); }
```

Futófény 8x8-as LED mátrix-szal

- **Különálló LED-ek helyett 8x8-as mátrixot is használhatunk: 1088BS** LED mátrix esetén csak a jobb szélső oszlopot hajtjuk meg
- Ezeknek a LED-eknek a 16. láb a közös katódja, ezt egy áramkorlátozó ellenálláson keresztül kötjük a GND kivezetésre
- A LED anódokat (5,2,7,1,18,8,14,9 lábak) sorban a D2 - D9 kimenetekre kötjük

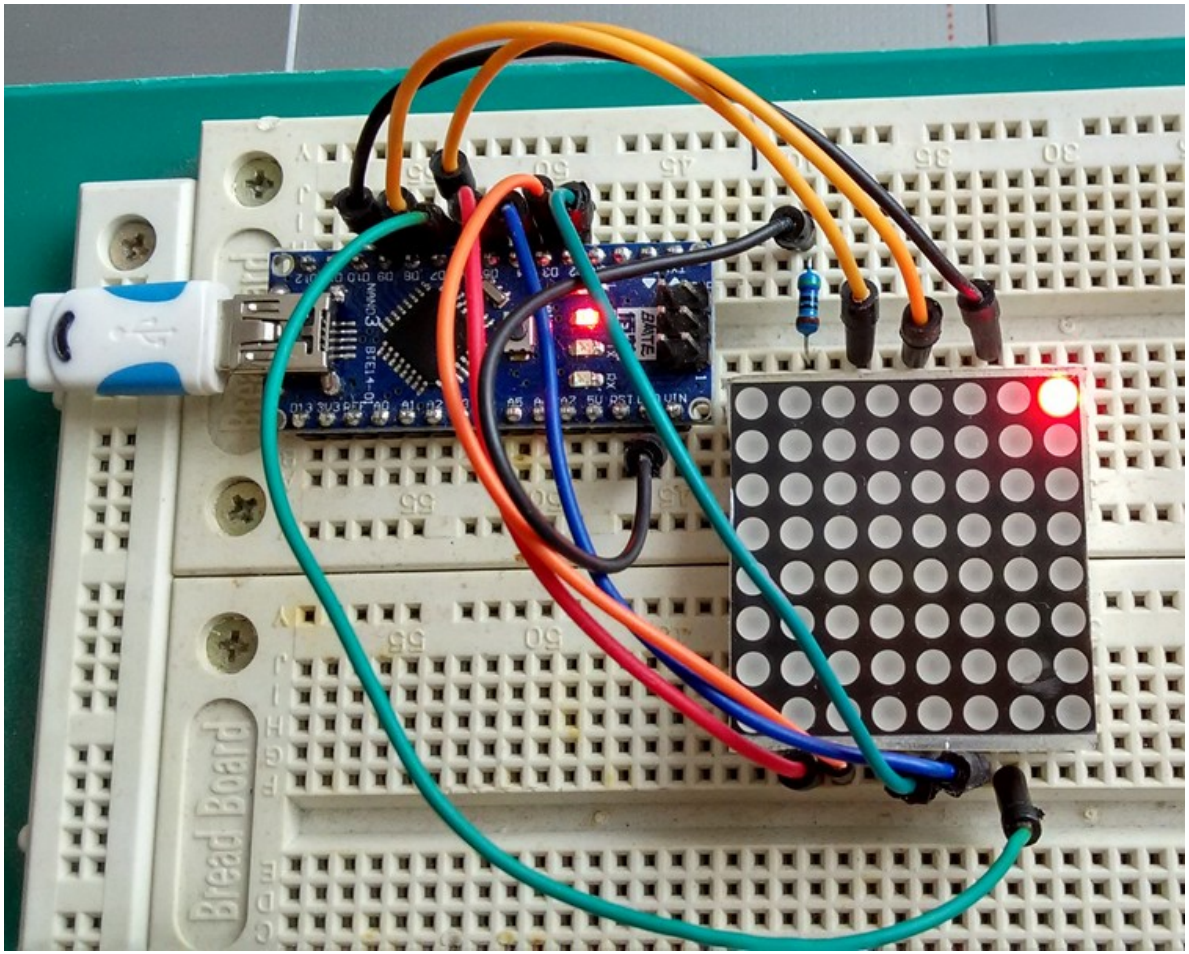


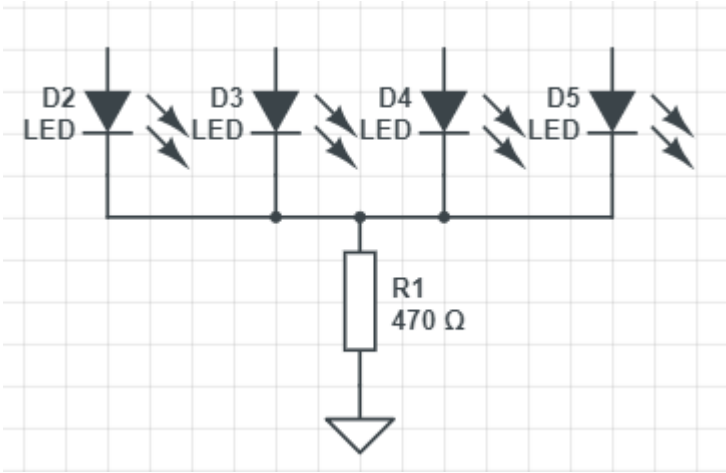


16	GND
5	D2
2	D3
7	D4
1	D5
12	D6
8	D7
14	D8
9	D9

Futófény 8x8-as LED mátrix-szal

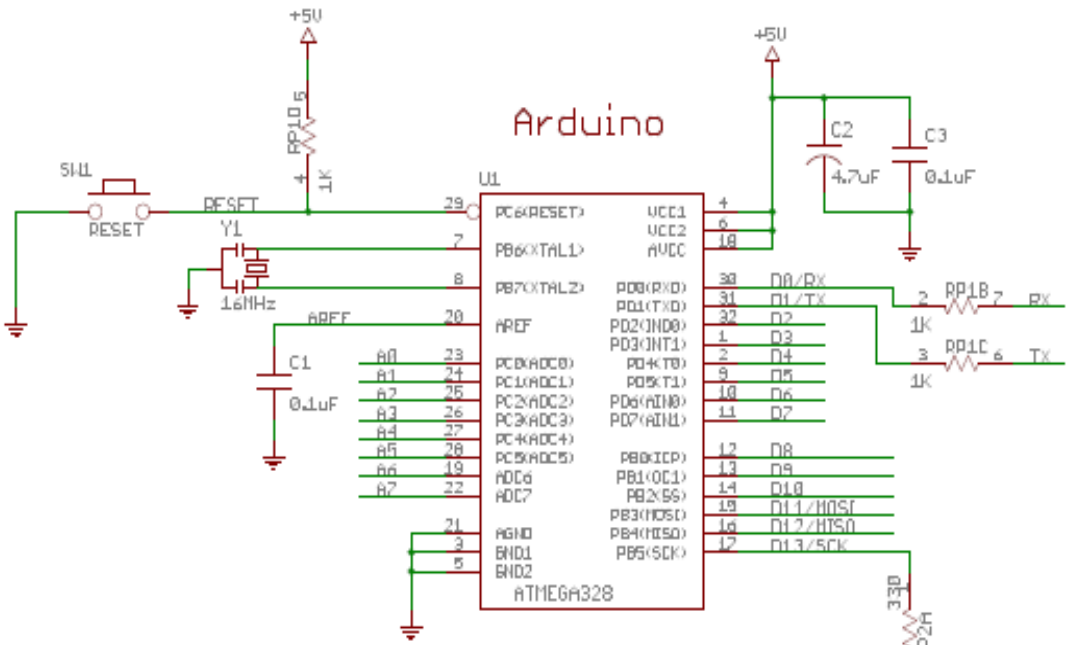
- Az egyszerűség kedvéért most is csak egyetlen áramkorlátozó ellenállást építettünk be (a közös katódok és a GND közé), kihasználva, hogy egyszerre mindig csak egy LED aktív.

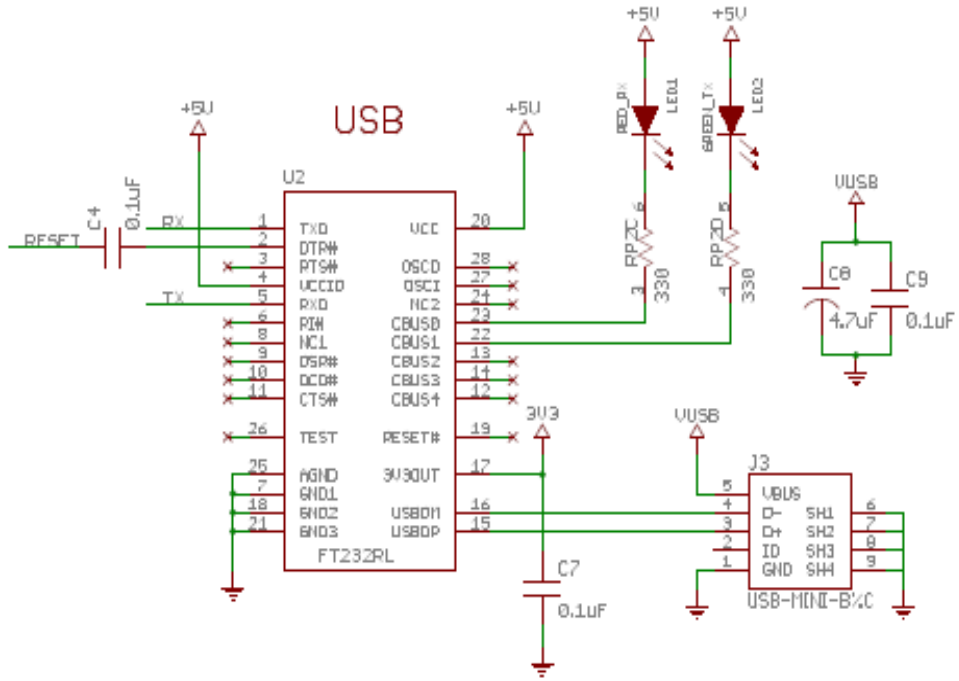


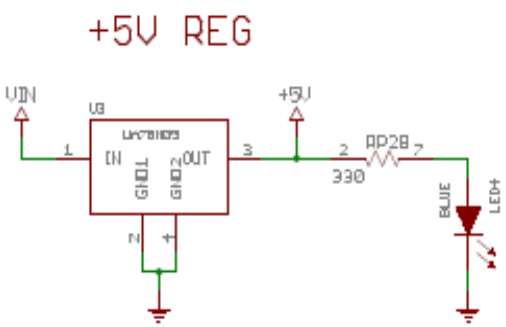


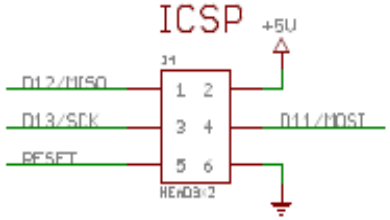
Az ábrán csak négy diódát tüntettünk fel, de a valóságban 8 db közös katódú LED van egy oszlopban

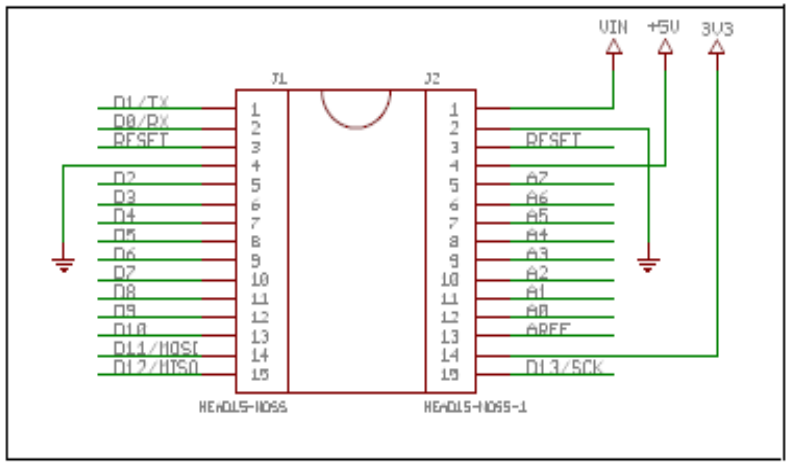
Arduino nano v3.0











Az Arduino nano kártya kivezetései