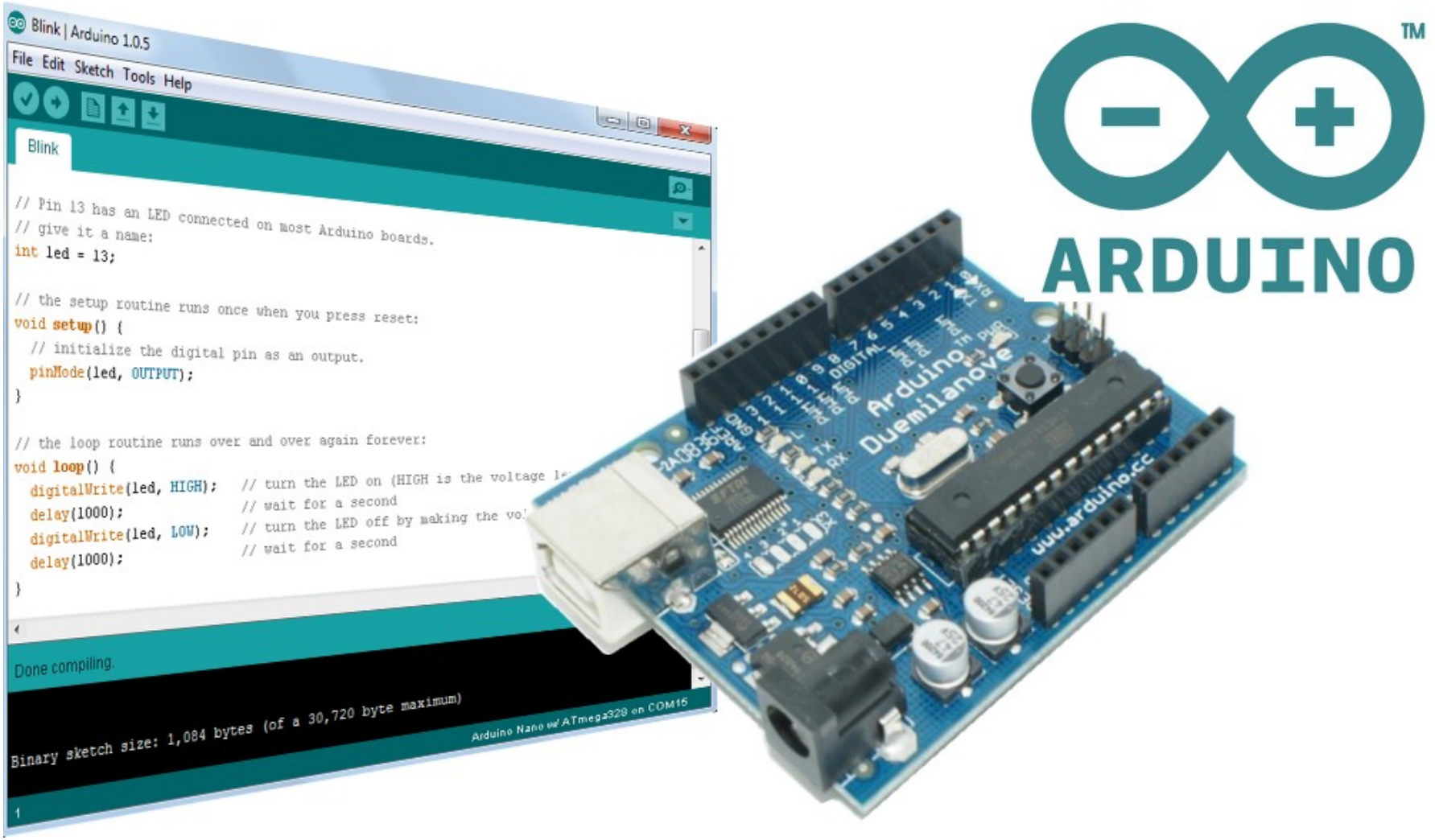
Bevezetés az elektronikába



12. Arduino programozás – analóg bemenetek

Vezérlési szerkezetek

A műveletek végrehajtási sorrendjét határozzák meg.

- Utasítások és blokkok
- Feltételes elágazás
  - **if – else** utasítás
  - **else – if** utasítás
  - **switch** utasítás
- Ciklusszervezés
  - **while** utasítás
  - **for** utasítás
  - **do – while** utasítás
- A **break** és **continue** utasítások
- A **goto** utasítás és **a címkék**

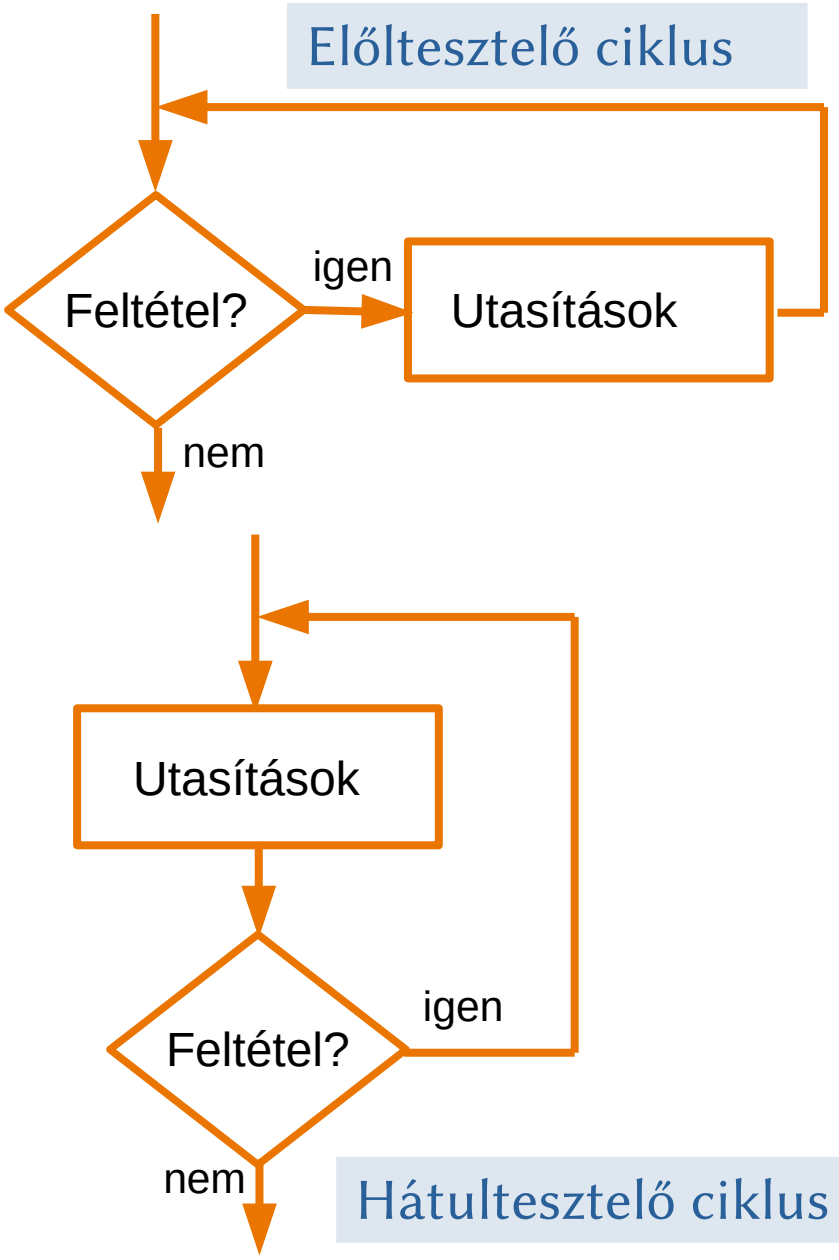
Az előző előadásokban már tárgyalt vezérlési szerkezetek

Ma ezekről lesz szó

Forrás: BRIAN W. KERNIGHAN – DENNIS M. RITCHIE: A C programozási nyelv, 3. fejezet

Ciklusok szervezése

- Ismétlődő feladatok esetén az utasításainkat „újrahasznosíthatjuk” **ciklusok** szervezésével, amikor egy utasításblokkot többször lefuttatunk
- Nem (csak) végtelen ciklust akarunk, ezért kell bennmaradási (vagy kilépési) **feltétel**
- A feltételt az utasításblokk végrehajtása előtt vagy után is vizsgálhatjuk:
  - Előltesztelő ciklus: **while, for**
  - Hátultesztelő ciklus: **do ... while**



Ciklusszervezés `for` utasítással

`for` `(int i=0; i<5; i++)` **`utasítás`** (vagy **`blokk`**)

A **`for`** utasítás is elöltesztelő ciklust szervez

1) Az első kifejezésben szereplő változó felveszi a kezdőértéket
2) Kiértékelésre kerül a második kifejezés, s teljesülés (nem nulla érték) esetén végrehajtásra kerül a ciklus törzse
3) A ciklustörzs lefutása után végrehajtásra kerül a harmadik kifejezésben szereplő művelet, majd visszatérünk a 2. ponthoz

Kezdőérték feltétel ciklusváltozó léptetés

```
for(i = 0; i < 5; i++) {
  digitalWrite(LED,HIGH);
  delay(50);
  digitalWrite(LED,LOW);
  delay(500);
}
```

FOR ciklus

Kezdőérték
Feltétel

```
int i = 0;
while (i < 5) {
  digitalWrite(LED,HIGH);
  delay(50);
  digitalWrite(LED,LOW);
  delay(500);
  i++;
}
```

Ciklusváltozó léptetés

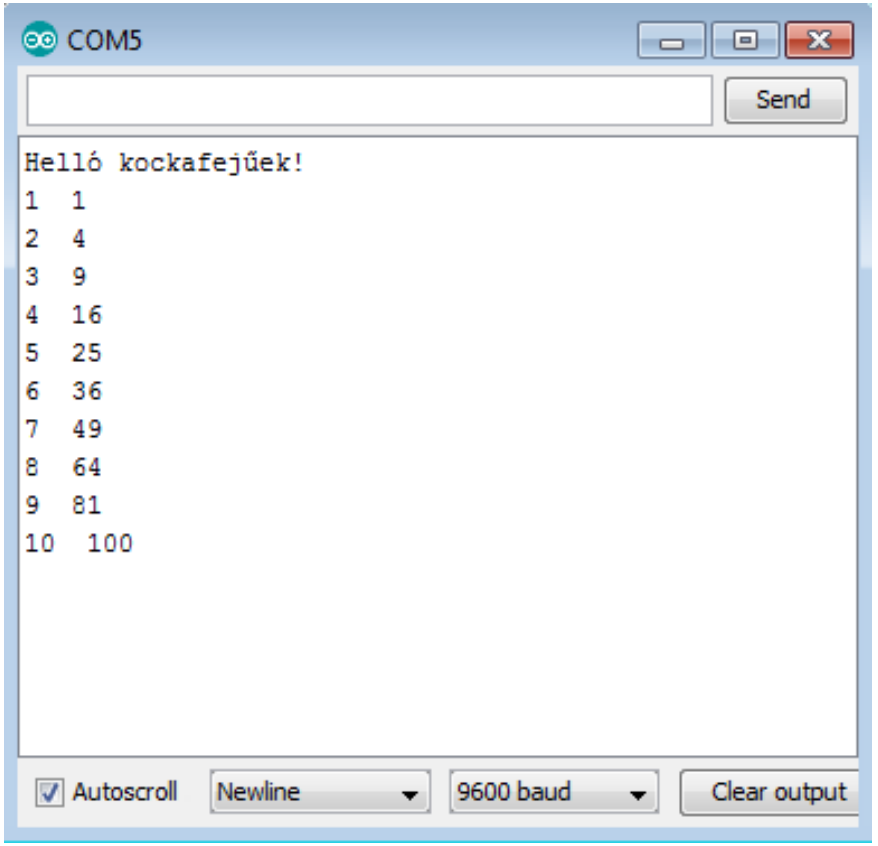
WHILE ciklus

Kiírás a terminál ablakba

- Az **USB – soros átalakítón** keresztül kapcsolatban vagyunk a számítógéppel (ezen keresztül történik a programletöltés), használjuk ki ezt a lehetőséget eredmények kiíratására!

Adatsebesség

```
void setup() {
  Serial.begin(9600);
  Serial.println("Helló kockafejűek!");
  for (int i = 1; i <= 10; i++) {
    Serial.print(i);
    Serial.print("  ");
    Serial.println(i * i);
  }
}

void loop() {
  // Nincs tennivaló!
}
```



if utasítások egymásba ágyazása

- Összetett feltételek esetén az **if** utasításokat egymásba ágyazhatjuk
- Ügyeljünk a zárójelezésre az **else** ágak megfelelő társításához!

Egymásba ágyazott **if-else** szerkezeteknél, hiányzó **else** ágak esetén nem világos, hogy a meglévő **else** ág melyik **if** utasításhoz tartozik. Például az

```
if (n > 0)
    if (a > b) z = a;
    else z = b;
```

programrészletben az **else** a belső **if** utasításhoz tartozik.

Általános szabály: az **else** mindig a hozzá legközelebb eső, **else** ág nélküli **if** utasításhoz tartozik. Ha nem így szeretnénk, akkor használjunk kapcsos zárójeleket! Például:

```
if (n > 0) {
    if (a > b) z = a;
}
else z = b;
```

A nem egyértelmű helyzet különösen zavaró az olyan szerkezetekben, mint az alábbi:

```
if (n >= 0)
    for (i = 0; i < n; i++)
        if (s[i] > 0) {
            printf("...");
            return i;
        }
else      /* ez így hibás */
    printf("n értéke negatív") ;
```

A tagolás ellenére a fordító itt az **else** ágat a belső **if** utasításhoz kapcsolja.

Az ilyen, nehezen felderíthető hibák megelőzésére használjunk kapcsos zárójeleket a második **if** utasítás körül!

Az `else if` utasítás

Az **else if** szerkezet adja a többszörös elágazások programozásának egyik legáltalánosabb lehetőségét.

A szerkezet úgy működik, hogy a program sorra kiértékeli a *kifejezéseket* és ha bármelyik ezek közül igaz, akkor végrehajtja a megfelelő *utasítást*, majd befejezi az egész vizsgáló láncot.

Itt is, mint bárhol hasonló esetben, az utasítás helyén kapcsos zárójelek között elhelyezett blokk is állhat.

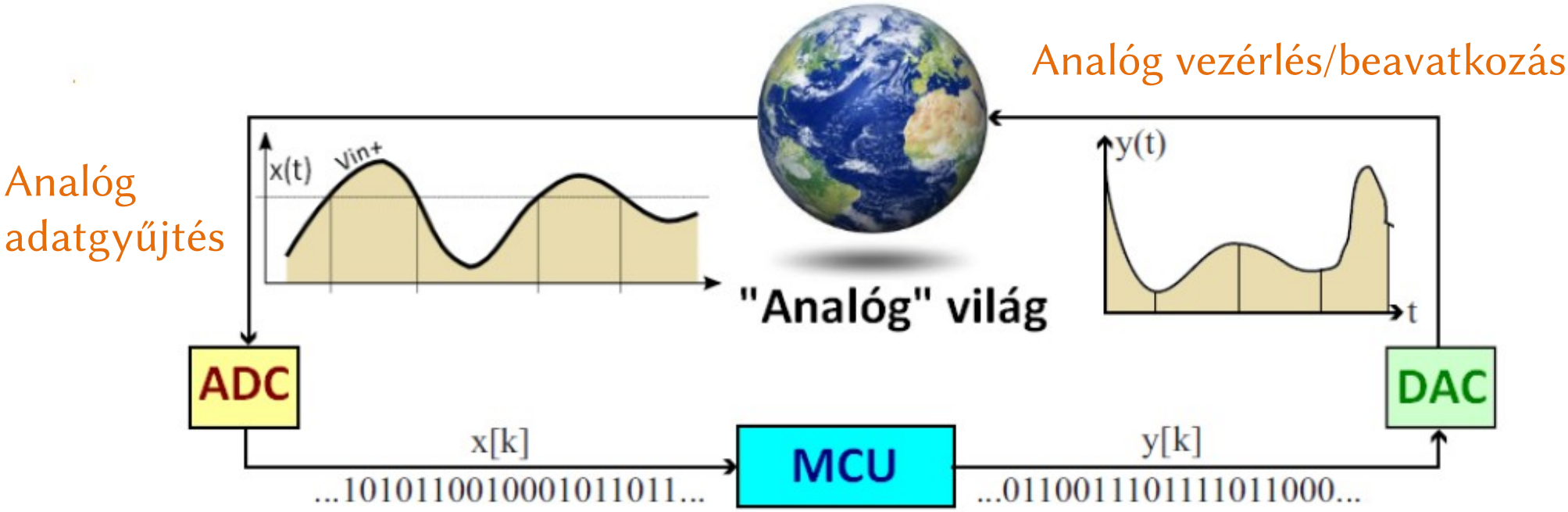
Az utolsó **else** ág alapértelmezés szerint a *„fentiek közül egyik sem”* esetet kezeli. Ha ilyenkor semmit sem kell csinálni, ez az ág elhagyható.

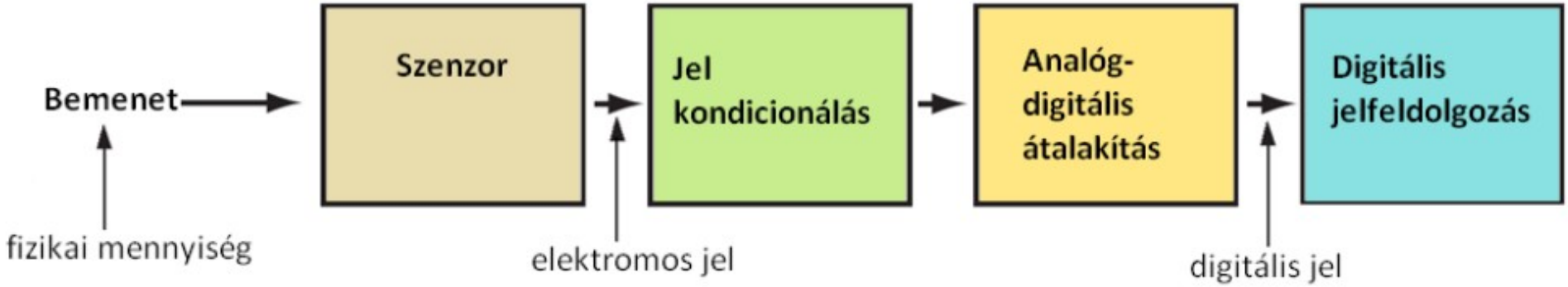
Szintaxis:

```
if (kifejezés)
    utasítás
else if (kifejezés)
    utasítás
else if (kifejezés)
    utasítás
else if (kifejezés)
    utasítás
.
.
.
else
    utasítás
```

Analóg jelfeldolgozás

- Analóg világban élünk, de digitális mikrovezérlővel dolgozunk…





Az analóg adatgyűjtő ág elemei

- **Szenzor:**
  - A folytonos fizikai mennyiségeket (pl. hőmérséklet, nyomás, páratartalom, sebesség, áramlási sebesség, elmozdulás, gyorsulás, szöggyorsulás) elektromos jellé alakítja (feszültséggé vagy árammá).
- **Jel kondicionálás (szűrés, erősítés, stb.):**
  - A mérendő mennyiség elektromos jellé alakítása után még szűrésre, jel erősítésre, impedancia illesztésre is szükség lehet, hogy az analóg-digitális átalakító (ADC) bemeneti tartományába transzformáljuk az átalakítandó jelet.
- **Analóg-Digitális Átalakító (ADC):**
  - *Bemenet:* a mérendő jel
  - *Kimenet:* a mérendő jellel arányos számot reprezentáló digitális kód

Analóg–digitális átalakító (ADC)

- Az ADC feladata az, hogy diszkrét kódokká alakítsa a bejövő jelet
- A konverzió digitális értéke (N_{ADC}):
  - Végkitérés: N_{ADC}= 1023, ha a bemenő jel ≥ V_{R+} –1.5*LSB
  - Nulla: N_{ADC} = 0, ha a bemenő jel ≤ V_{R-} + 0.5 LSB
  - Közbeeső értékekre:

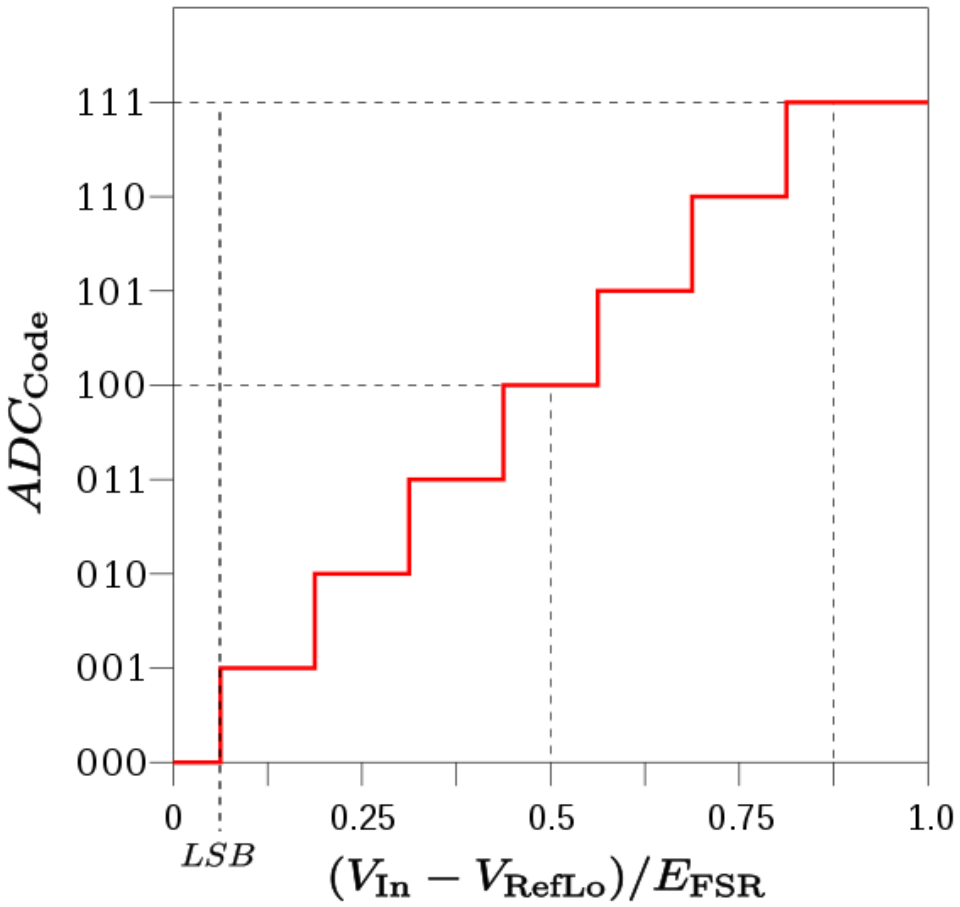
$$N_{ADC} = 1024 * (V_{IN} - V_{R-})/(V_{R+} - V_{R-})$$

- A fenti képletből V_{IN}-t kifejezve, ezt kapjuk:

$$V_{IN} = (V_{R+} - V_{R-}) * N_{ADC}/1024 + V_{R-}$$

- V_{R-} általában = 0

Egy 3-bites átalakító ideális átviteli függvénye



adc_potmeter.ino

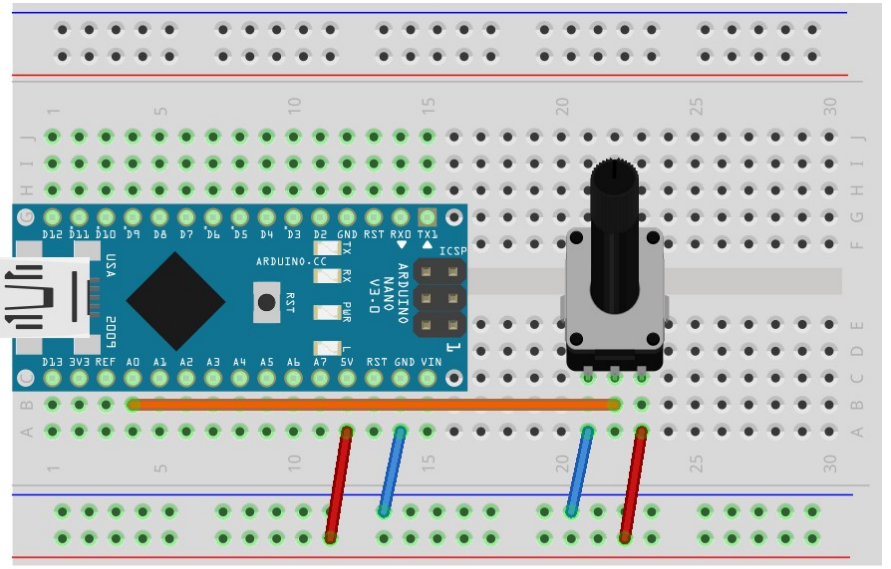
- Mérjük meg a potméterrel leosztott feszültséget és írassuk ki!

```
void setup() {
  Serial.begin(9600);
  analogReference(DEFAULT);
  Serial.println("ADC potmeter reading");
}

void loop() {
  int reading = analogRead (A0);
  float voltage=(reading*4.71)/1024.0;
  Serial.print("ADC: ");
  Serial.print(reading);
  Serial.print("  Voltage: ");
  Serial.print(voltage, 2);
  Serial.println(" V");
  delay(2000);
}
```

VCC a referencia

A0 a bemenet

VCC tényleges értéke



fritzing

```
COM5

ADC potmeter reading
ADC: 551  Voltage: 2.53 V
ADC: 550  Voltage: 2.53 V
ADC: 550  Voltage: 2.53 V
ADC: 550  Voltage: 2.53 V
ADC: 550  Voltage: 2.53 V
ADC: 550  Voltage: 2.53 V
ADC: 550  Voltage: 2.53 V
ADC: 550  Voltage: 2.53 V
ADC: 550  Voltage: 2.53 V
ADC: 550  Voltage: 2.53 V
```

A beépített hőmérő használata

- **ATmega328p** beépített hőmérő a 0b1000 ADC csatornán érhető el

```
long readTemp() {
  long result;
  // Hőmérés , 1V1 belső referenciával
  ADMUX = _BV(REFS1) | _BV(REFS0) | _BV(MUX3);
  delay(20);                            // Vref beállási idő
  ADCSRA |= _BV(ADSC);                  // Konverzió indítása
  while (bit_is_set(ADCSRA,ADSC));      // Konverzió végére vár
  result = ADCL;                        // Előbb ezt kell kiolvasni
  result |= ADCH<<8;                    // Magasabb helyiértékű bitek
  result = (result - 125) * 1075;       // Hőmérséklet * 10 000 Celsius fokban
  return result;
}

void setup() {
  Serial.begin(9600);
  Serial.println("Belső hőmérő");
}

void loop() {
  long temp = readTemp();
  Serial.print("Temperature = ");
  Serial.print(temp / 10000.0f,1);
  Serial.println(" °C");
  delay(5000);
}
```

Mérés belső referenciával, a 0b1000 csatornában

beepitett_homero.ino

Eredmények
(egy kis melegítéssel)

Belső hőmérő
Temperature = 25.6 °C
Temperature = 25.6 °C
Temperature = 25.8 °C
Temperature = 25.8 °C
Temperature = 25.9 °C
Temperature = 26.1 °C
Temperature = 25.9 °C
Temperature = 25.7 °C
Temperature = 25.8 °C
Temperature = 26.8 °C
Temperature = 26.9 °C
Temperature = 27.5 °C

A tápfeszültség meghatározása

- **ATmega328p** belső referenciája a 0b1110 ADC csatornán érhető el

tapfeszultseg_meghatarozasa.ino

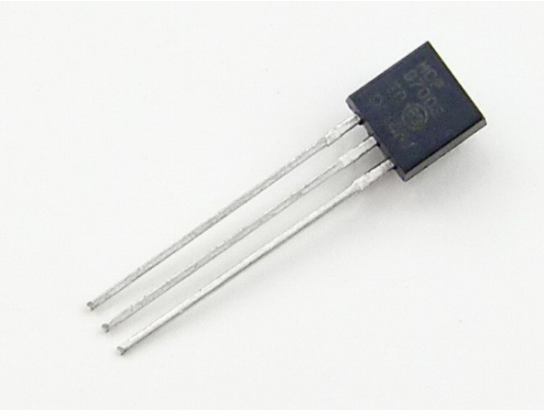
```
long readVcc() {
  long result;
  // 1.1V referenciafeszültség mérése, AVcc a referencia
  ADMUX = _BV(REFS0) | _BV(MUX3) | _BV(MUX2) | _BV(MUX1);
  delay(20);                            // Vref beállási idő
  ADCSRA |= _BV(ADSC);                  // Konverzió indítása
  while (bit_is_set(ADCSRA,ADSC));      // Konverzió végére vár
  result = ADCL;                        // Előbb ezt kell kiolvasni
  result |= ADCH<<8;                    // Magasabb helyiértékű bitek
  result = 1100L * 1024L / result;      // AVcc kiszámítása [mV]
  return result;                        // 1100 mV Vref * 1024 ADC felbontás
}
void setup() {
  Serial.begin(9600);
  Serial.println("Tápfeszültség meghatározása belső referenciával");
}

void loop() {
  long vcc = readVcc();
  Serial.print("Vcc = ");
  Serial.print(vcc);
  Serial.println(" mV");
  delay(5000);
}
```

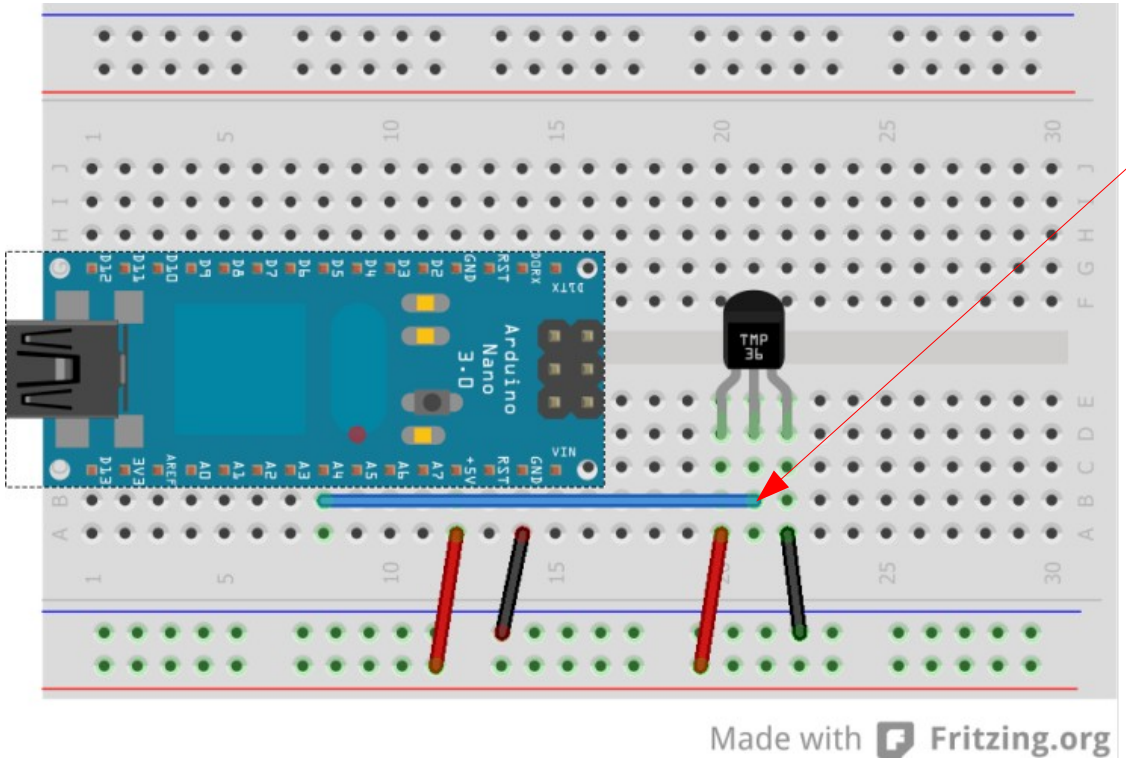
Tápfeszültség meghatározása belső referenciával
Vcc = 4752 mV
Vcc = 4752 mV
Vcc = 4752 mV
Vcc = 4752 mV
Vcc = 4752 mV
Vcc = 4752 mV

Futási eredmény

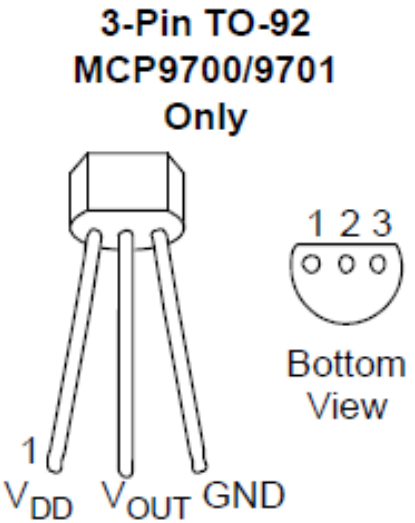
Hőmérséklet mérése analóg hőmérővel

- Microchip **MCP9700**
  - VDD = 2,5 – 5,5 V
  - Mérési tart.: –40 – 150 °C
  - Érzékenység: 10 mV / °C
  - Nullapont: 500 mV @ 0 °C





Most az **A4** bemenetre kötöttük a hőmérőt, de más bemenetet is használhatunk!



AnalogThermometer.ino

```
void setup () {
   Serial.begin(9600);
   analogReference(INTERNAL);
   Serial.println("Analóg hőmérő");
}

void loop () {
   long reading = analogRead (A4);
//--- Átszámítjuk mV-ba
   long voltage = reading*1100/1024;
   Serial.print (voltage);
   Serial.print (" mV, ");
   float tempC = (voltage - 500)/10.0;
//--- Átszámítjuk Celsius fokokra
   Serial.print (tempC,1);
   Serial.print (" °C, ");
//--- Celsiusból Fahrenheit fokokba
   float tempF = (tempC * 9/5) + 32;
   Serial.print (tempF,1);
   Serial.println (" °F");
   delay (5000);
}
```

Hőmérés **MCP9700A** hőmérővel

Az előző oldali példában az **A4** bemenetre kötöttük a hőmérőt, de ez egyáltalán nem kötelező!

Analóg hőmérő
763 mV, 26.3 °C, 79.3 °F
771 mV, 27.1 °C, 80.8 °F
770 mV, 27.0 °C, 80.6 °F
770 mV, 27.0 °C, 80.6 °F
771 mV, 27.1 °C, 80.8 °F
770 mV, 27.0 °C, 80.6 °F
770 mV, 27.0 °C, 80.6 °F
771 mV, 27.1 °C, 80.8 °F
770 mV, 27.0 °C, 80.6 °F
768 mV, 26.8 °C, 80.2 °F
779 mV, 27.9 °C, 82.2 °F

AnalogThermometer2.ino

```
void setup () {
  Serial.begin(9600);
  analogReference(INTERNAL);
  Serial.println("Analóg hőmérő átlagolással");
}

void loop () {
  long reading = 0;
  for (int i = 0; i < 1100; i++) {
    reading += analogRead(A4);
  }
  long voltage = reading/1024;
  Serial.print (voltage);
  Serial.print (" mV, ");
  float tempC = (voltage-500)/10.0;
  Serial.print (tempC, 1);
  Serial.print (" °C, ");
  float tempF = (tempC*9/5)+32;
  Serial.print (tempF, 1);
  Serial.println (" °F");
  delay (5000);
}
```

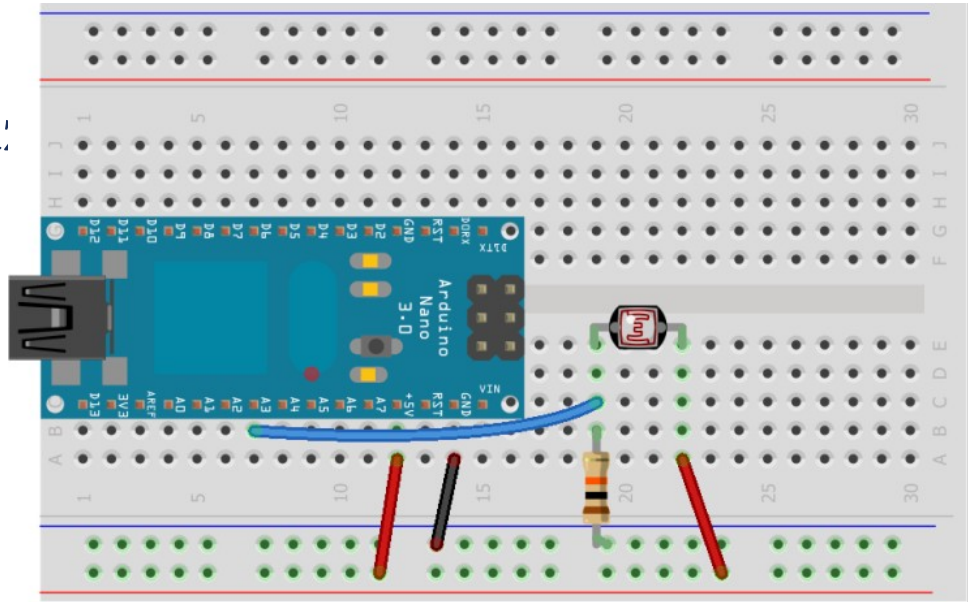
Hőmérés **MCP9700A** hőmérővel, átlagolással

1100 db mérés eredményét összeadjuk és nem szorzunk 1100-za!

Analóg hőmérő átlagolással
772 mV, 27.2 °C, 81.0 °F
771 mV, 27.1 °C, 80.8 °F
772 mV, 27.2 °C, 81.0 °F
772 mV, 27.2 °C, 81.0 °F
771 mV, 27.1 °C, 80.8 °F
771 mV, 27.1 °C, 80.8 °F
771 mV, 27.1 °C, 80.8 °F
771 mV, 27.1 °C, 80.8 °F
771 mV, 27.1 °C, 80.8 °F
771 mV, 27.1 °C, 80.8 °F

Fénymérés fényérzékeny ellenállással

- A kapcsolás feszültségosztóként működik, amelyikben a felső tag egy CdS fényérzékeny ellenállás, melynek ellenállása a megvilágítástól függően széles határok között változik. A megvilágítás hatására az ellenállása csökken…
- Az ellenállásosztó közös pontját a **A3** analóg bemenetre kötöttük
- Az eredményt soros porton kiküldjük a számítógépre, s a terminálablakban jelenik meg.



Photoresistor.ino

```
void setup () {
   Serial.begin(9600);
   analogReference(DEFAULT);   //VCC a referencia
   Serial.print("Photoresistor");
}

void loop () {
   long reading = 0;
   for(int i=0; i<4750; i++) {
      reading += analogRead(A3);
   }
   // Trükkös osztás 1024-gyel
   float voltage = reading>>10;
   Serial.print(voltage,0);
   Serial.print(" mV, ");
   // Átszámítás kOhm-ra
   // Rx = VCC*10k/voltage - 10k
   float rx = 47500/voltage - 10;
   Serial.print(rx,3);
   Serial.println(" kOhm");
   delay (5000);
}
```

4750 db mérés eredményét összeadjuk és nem szorzunk Vref-fel!

Photoresistor
3145 mV, 5.103 kOhm
3149 mV, 5.084 kOhm
3036 mV, 5.646 kOhm
1393 mV, 24.099 kOhm
1322 mV, 25.930 kOhm
4406 mV, 0.781 kOhm
4449 mV, 0.677 kOhm
4134 mV, 1.490 kOhm
2856 mV, 6.632 kOhm
2836 mV, 6.749 kOhm

Az Arduino nano kártya kivezetései