Bevezető a mikrokontrollerek, az IoT és az Arduino platform világába

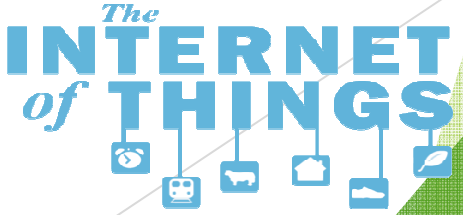
Előadó:

Ruzsinszki Gábor
https://webmaster442.hu


Webmaster442.hu


@webmaster442

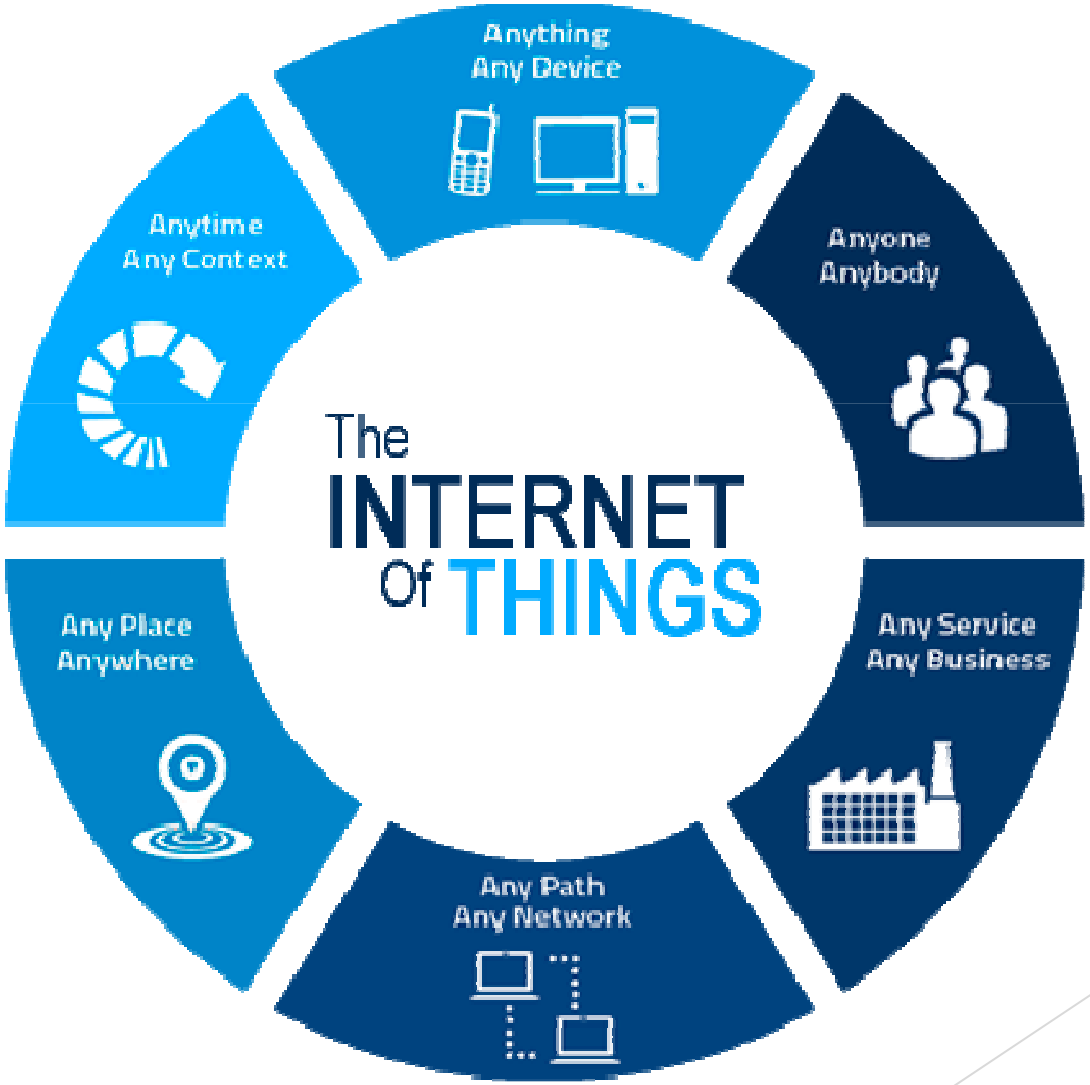

webmaster442



Történeti áttekintés

- 1958 – Texas Instruments, Jack Kilby – Első integrált áramkör
- 1971 – Intel 4004 – Első Mikroprocesszor
- 1977 – Intel 8048 – Első Mikrovezérlő (IBM AT Keyboard)
- 1984 – Atmel megalapítása
- 1985 – ARM Architektúra megjelenése
- 1989 – Microchip Technology megalapítása
- 2005 - Arduino létrejötte/megszületése
- 2007 – Apple iPhone
- 2012 – RaspberryPI v.1
- Jelenleg: IoT/4. ipari forradalom

Az IoT és a 4. ipari forradalom



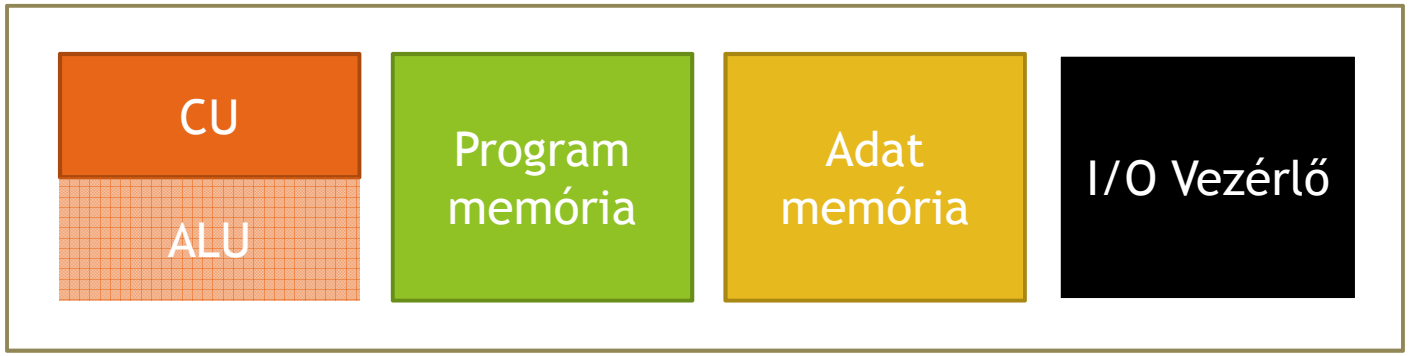
Mikrovezérlő vagy SOC?

- Az SOC lényegében…
  - Egy magasan integrált számítógép
  - Legfőbb felhasználási területek: Telefonok, Laptopok, Játék konzolok
  - Nincs integrált memória -> külső DRAM
  - Nincs integrált tárhely -> külső NAND Flash
  - Neumann Architektúra
  - Operációs rendszer kell a működtetéséhez
  - Komplex feladat megoldáshoz fejlesztve

Mikrovezérlő

- Célfeladat ellátására tervezett integrált áramkör
  - Harvard architektúrával rendelkezik, de létezik Neumann architektúrás kivitel is
  - Van belső memória -> SRAM
  - Van belső tárhely -> NOR Flash
  - Nem kell hozzá operációs rendszer
  - Vezérlési célok megvalósítására fejlesztve

Egy mikrovezérlő általános felépítése



Központi egység

- Jellemző a lebegőpontos egység hiánya
- Domináns utasításkészlet: RISC alapú -> fix utasításhossz, könnyű huzalozott logikából implementálni
- Jellemzően 8 vagy 32 bites (ARM) szóhossz
- Széles órajel tartomány (30 kHz -> pár száz MHz)
- Fő tervezési szempont általában: energiatakarékosság

Adat és kód memória

- Kódmemória:
  - Program kódot tárol
  - Régebben EPROM/EEPROM, manapság NOR FLASH
  - Egyszer programozható modellek manapság is léteznek
  - Tipikusan kiB méretekről beszélünk
- Adatmemória:
  - Változókat, adatokat tárol, amivel dolgozik épp a processzor.
  - SRAM
  - Tipikusan byte a mérőszám*

I/O vezérlő

- Legfontosabb eleme
- Minden esetben a felhasználási igény határozza meg
- Számos periféria
  - GPIO
  - I^2C
  - SPI
  - Soros port
  - USB*

I/O vezérlő

- Általában a chip összes funkciójának külön kihasználásához jóval több fizikai kivezetés kellene, mint amennyi adott.
- Ebből adódóan a kivezetések működése regiszterek segítségével szoftveresen konfigurálható.
- Megszakítások kezelése
- Integrált ADC 10-12 bit felbontással
  - Külön feszültségforrásról hajtott
  - Ref. feszültség bemenet
  - Általában zajos bemenet, az IO miatt

Mikor kell mikrovezérlő és mikor SOC ?

- Mikrovezérlőt akkor érdemes használni, ha:
  - Idő kritikus a feladat (ns és ms időzítés kell)
  - Szabályzási kör megvalósítása a cél
  - Operációs rendszer nélkül is megoldható
  - Nem kell bonyolult GUI a feladathoz
- SOC rendszert akkor érdemes használni, ha:
  - A feladat nagy számítási igénnyel jár
  - Komplex és több protokollt / technológiát ölel fel a projekt
  - Innovatív, modern GUI kell

Mikrovezérlő és SOC

- Nem érdemes ellenségeknek tekinteni őket ☺

Mikrovezérlős fejlesztéshez kell:

- Egy mikrovezérlő
- Programozó eszköz, feltöltő
- Fordító / fejlesztő program
- Elektronikai alapismeretek a nyomtatott áramkör megtervezéséhez és legyártásához.

A fejlesztés klasszikus folyamata

- Kiválasztott mikrovezérlő adatlapjának elolvasása (50-1200 oldal)
- Fejlesztőeszköz megismerése, dokumentációjának elolvasása (50-1000 oldal)
- Kapcsolás megtervezése, összeállítása
- Szoftver megírása

A klasszikus folyamat problémái

- Minden mikrovezérlő típus külön belső felépítéssel rendelkezik, így a konfigurációs regiszterek működésének elsajátítása hosszadalmas és frusztráló.
- Ezt minden egyes típus esetén el kell sajátítani, ami nem túl kellemes.
- A legtöbb programozó szoftver csak Windows platformra érhető el (Microchip főleg).

A klasszikus folyamat problémája

A kiút

- A problémát többen felismerték, de valahogy elbeszéltek egymás mellett ☺
- 2001: Ben Fry & Casey Reas (MIT)
  - Processing: Java alapú programozást tanító környezet
- 2003: Hernando Barragán, Brett Hagman and Alexander Brevig
  - Wiring: Mikrovezérlő fejlesztő környezet, Atmel alapokon, Processing IDE segítségével
- 2005: Massimo Banzi
  - Arduino projekt

Az Arduino projekt

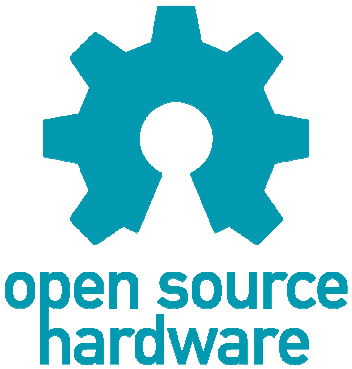
- A tervező mikrovezérlős rendszerfejlesztést oktatott az Iverai egyetemen és felismerte, hogy a piacon nem igen van olyan termék, amit egy diák is megengedhet magának
- A Projekt a Wiring platformra épül, de jelentősen továbbfejlődött azóta

Mit jelent az, hogy nyílt forráskódú?

- A nyílt forráskód azt jelenti, hogy bárki megismerheti, megnézheti, módosíthatja a program forráskódját.
- Azonban a szerző semmilyen felelősséget nem vállal a szoftverre.
- Számos jogi megállapodás (licenc) létezik, ami ezt biztosítja.
  - Népszerűek: GPL, BSD, MIT
- A nyílt forrás nem azonos az ingyenességgel.

A nyílt forráskód jelentősége hardver tervezés esetén

- A hardvertervezés nem egyszerű feladat, mivel a való világ nem olyan megbocsájtó, mint egy szoftveres környezet.
- Ezért jó dolog az, hogy meg lehet nézni mások terveit egy adott probléma megoldására.
- Egy-egy nyílt forráskódú hardverből olyan dolgok alkothatóak, amire a készítők nem is gondoltak.



Mikor nyílt ténylegesen a hardver?

- Ha a működéséhez szükséges szoftver és hardver forrása ténylegesen megismerhető (valamint reprodukálható)
  - Ilyen szempontból a RaspberryPi nem nyílt hardver
    - A rendszer alapját adó Broadcom SOC nem vásárolható meg bárki által
    - Az SOC betöltő programja nem nyílt forráskódú

Fejlesztés Arduino platformok esetén

- Fejlesztőeszköz megismerése, dokumentációjának elolvasása (50-120 oldal)
- A megszerzett tudás bármelyik Arduino modell esetén alkalmazható, mivel a fejlesztőkörnyezet könyvtárai elfedik a hardver egyediségét.
- Így a kód nagyon minimális módosítással hordozható a típusok között.

Fejlesztés Arduino platformok esetén

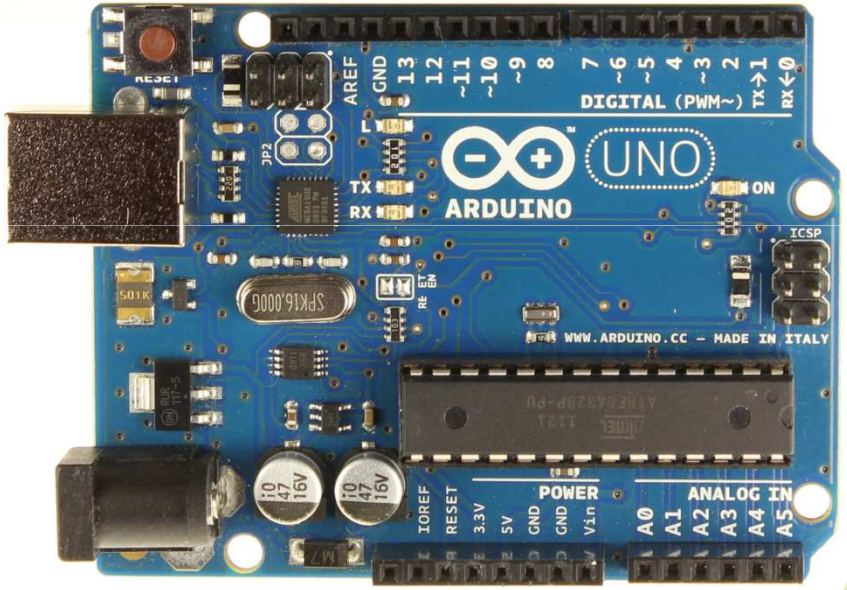
- Az igazi forradalmiság ebben van, mert:
  - Nem kell foglalkozni a hardver belső lelki világával
  - A kód ugyanúgy fog működni mindegyik mikrovezérlő esetén
  - Elég egy sémát megtanulni, nem feltétlen kell többet.
  - Rövid idő alatt is lehet látványos dolgokat alkotni, amely oktatás szempontjából kifejezetten fontos.

Az Arduino hardvere

- Jelenleg számos modell érhető el, mindegyikről nem lesz szó, csak a népszerűbbek a teljesség igénye nélkül:
  - Uno
  - Mega
  - Leonardo
  - Due
  - Yún

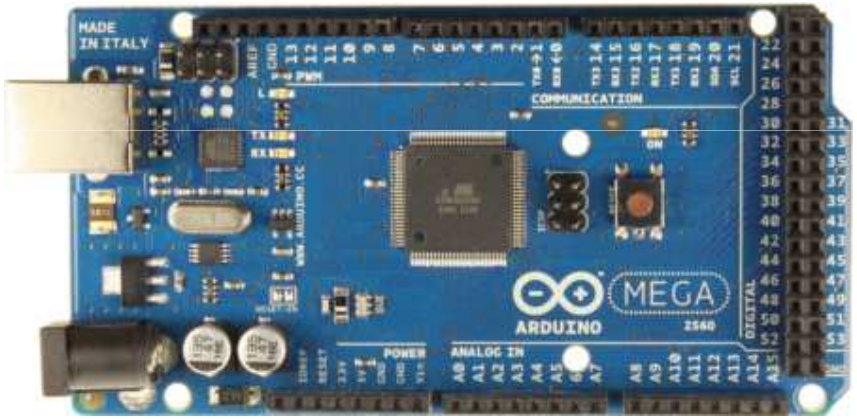
Arduino Uno

- ATmega 328 mikrovezérlő
- 16MHz órajel
- 13 digitális I/O
- 6db 10 bites ADC
- 32Kb kódmemória
- 2Kb adatmemória



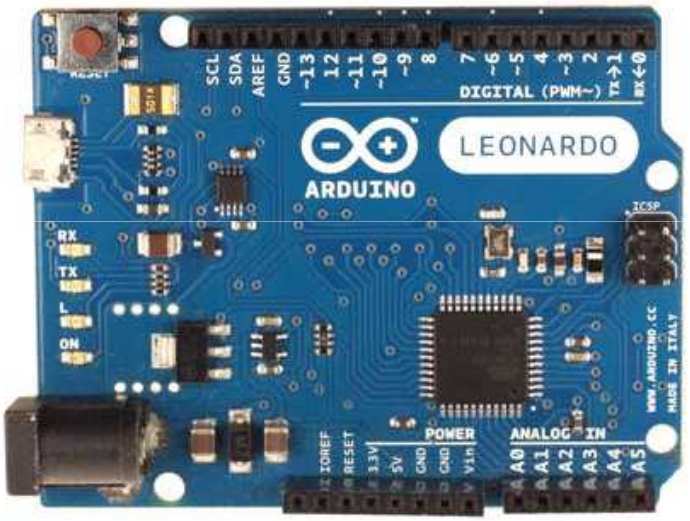
Arduino Mega

- ATmega 2560 mikrovezérlő
- 16MHz órajel
- 54 digitális I/O
- 16db 10 bites ADC
- 256Kb kódmemória
- 8Kb adatmemória



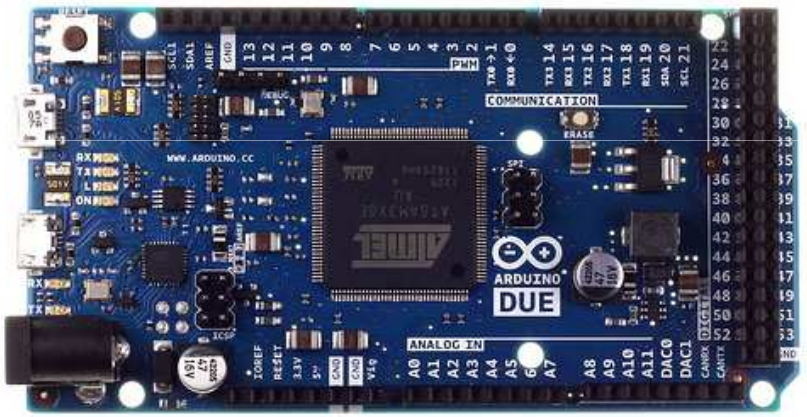
Arduino Leonardo

- ATmega32u4 mikrovezérlő
- 16MHz órajel
- 13 digitális I/O
- 6db 10 bites ADC
- 32Kb kódmemória
- 2Kb adatmemória
- Valódi USB támogatás*



Arduino Due

- AT91SAM3X8E mikrovezérlő
- 32 bites ARM
- 84MHz órajel
- 54 digitális I/O
- 12db 12 bites ADC
- 2db 12 bites DAC
- 512Kb kódmemória
- 96KB adatmemória

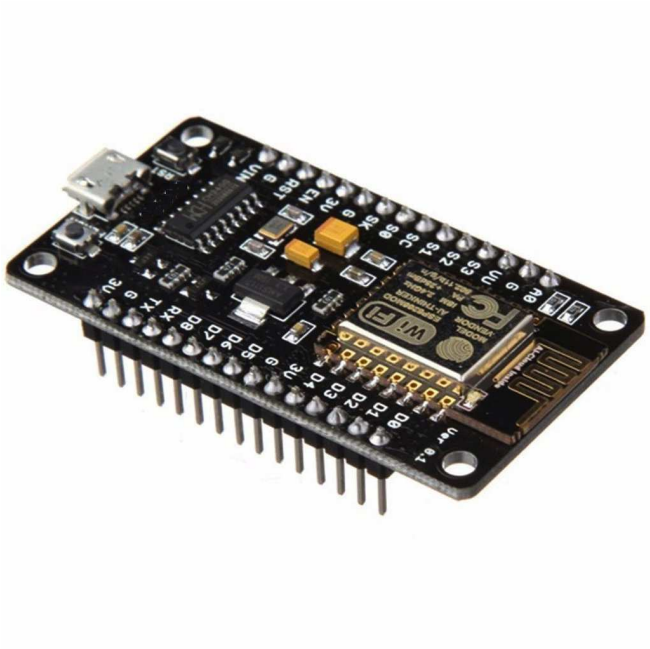


Arduino Yún

- Arduino Leonardo hardver kiegészítve egy Atheros AR9331 processzorral
- WLAN képességek
- Linux támogatás

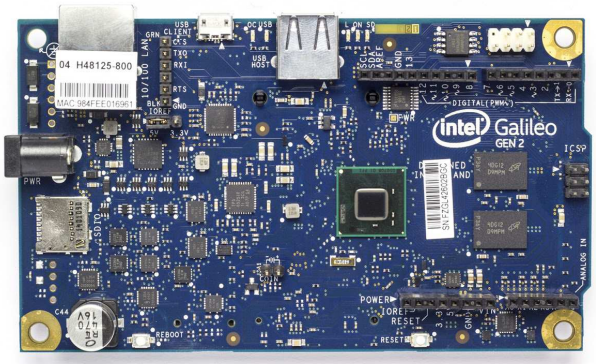
NodeMCU

- ESP8266 SOC
- WLAN
- 128kiB RAM
- 4MiB külső flash kód és adat tároláshoz



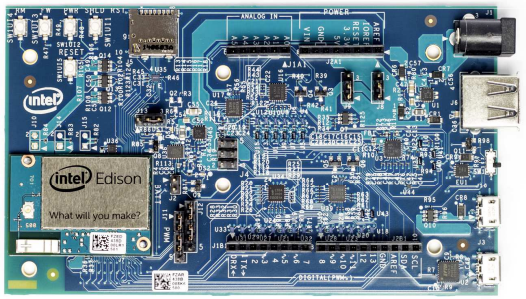
Intel Galieo

- Quark SoC X1000
- Intel Pentium kategóriás CPU 400 MHz
- 512KiB Flash sketch tárolásra 512KiB RAM-al
- 256 MiB DDR II az operációs rendszer számára



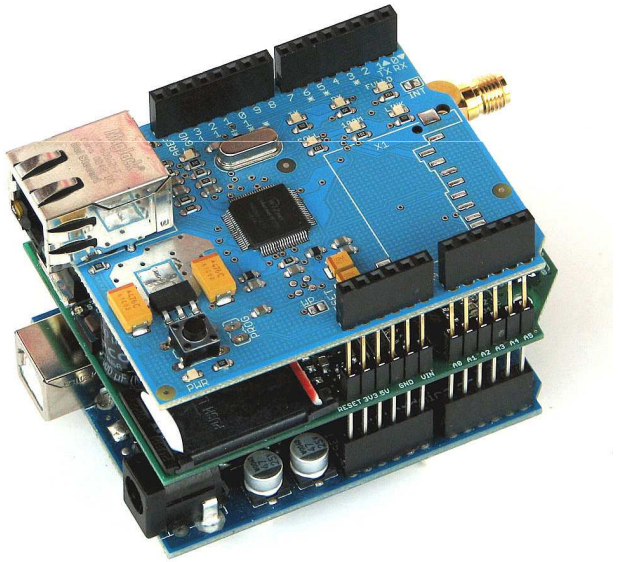
Intel Edison

- WiFi
- BTLE 4.0
- 4 GiB Flash
- 1 GiB DDR3 RAM
- 500 MHz Intel Atom CPU
- 100 MHz Intel Quark mikrovezérlő



Shieldek

- Minden Arduino lap fizikai elrendezésben és lábkiosztásban hasonló, kompatibilis egymással
- Ez könnyen lehetővé teszi további kiegészítő panelek, shield-ek használatát
- Számos hivatalos és nem hivatalos shield érhető el
- Határ: Csillagos ég ☺



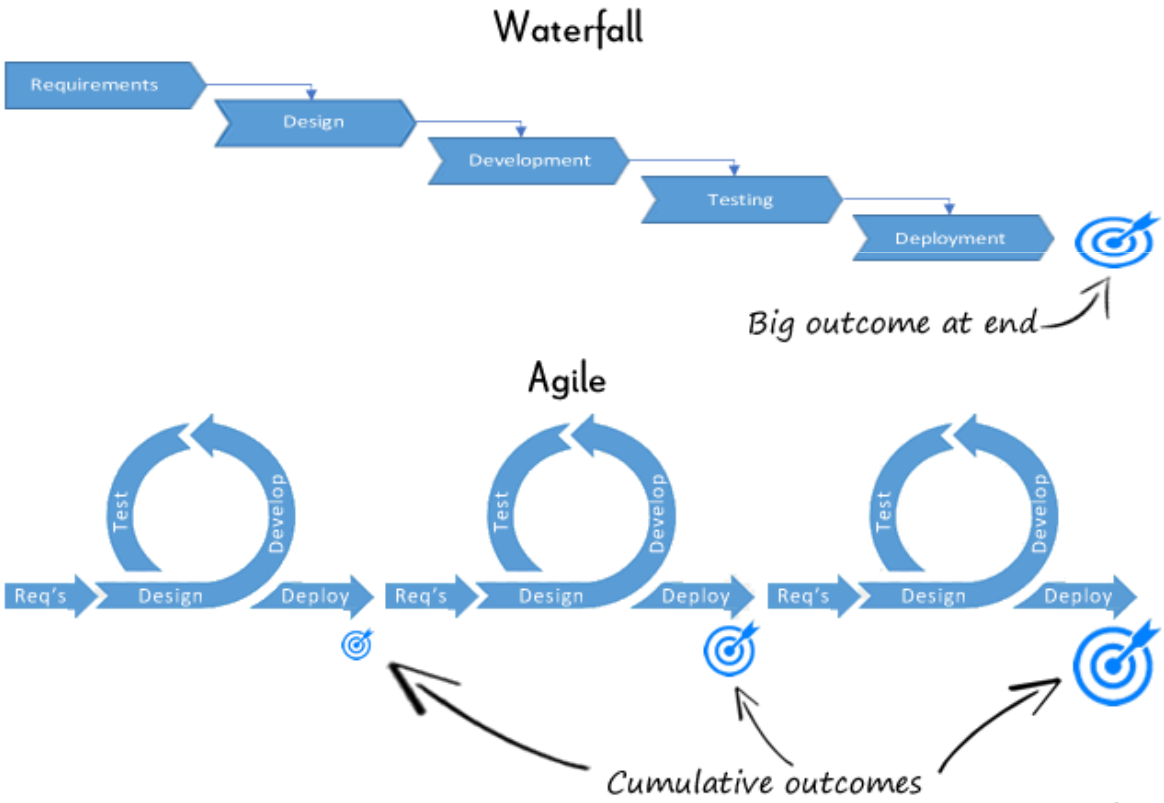
A programozás folyamata

- Nem a nyelvet kell megtanulni, hanem a gondolkodásmódot, szemléletet.
- Tapasztalat:
  - Probléma egészét egyben próbáljuk kezelni ahelyett, hogy részfeladatokra, kisebb egységekre bontanánk azt.
- Elsőre senki sem fog jó kódot írni, de gyakorlással menni fog -> Gyakorolni kell.
- Sokat segít más kódjának a megnézése -> Előbb utóbb rájövünk, mi jó és mi nem.

A programozás folyamata

- Segít, ha megkérünk mást, hogy nézze meg a kódunkat*
- Előbbi nem azt jelenti, hogy mással írassuk meg a programot
  - Nem tanulunk belőle
  - Más viszont igen, és még esetlegesen pénzt is keres
- Nem feltétlen a C/C++ a legjobb kezdő programozási nyelv, ha még nem tudunk programozni
  - MIT Scratch

Fejlesztés folyamata



A szoftver

- Java-ban íródott, Multiplatform
  - Windows/Linux/OS-X
- Windows esetén különösebb telepítést nem igényel
- A kommunikációs driver telepítése minden esetben kell, de Windows Update-ben is benne van*
- A programozási nyelv C++ alapú
  - Nincs teljes stdlib a szűkös hely miatt

Arduino IDE


sketch_apr20a | Arduino 1.0.5-r2
File Szerkesztés Sketch Eszközök Súgó
sketch_apr20a §
void setup()
{
pinMode(13, OUTPUT);
}
void loop()
{
digitalWrite(13, HIGH);
delay(500);
digitalWrite(13, LOW);
delay(500);
}
5
Arduino Leonardo on COM9

Programírás

1. Egyszerű LED villogtatás

```
#define LED1 2 //LED1 - D2 láb

void setup() {
     pinMode(LED1, OUTPUT);
}

void loop() {
     digitalWrite(LED1, HIGH);
     delay(100);
     digitalWrite(LED1, LOW);
     delay(100);
}
```

2. Fejlettebb LED villogtatás

```
#define LED1 2
#define LED2 3
#define LED3 4
#define LED4 5
#define LED5 6
#define LED6 7
#define LED7 8
#define LED8 9
#define LEDEK_SZAMA 8

int LED_TOMB[LEDEK_SZAMA] = { LED1, LED2, LED3, LED4, LED5,
LED6, LED7, LED8 };

void setup() {
     for (int i=0; i<LEDEK_SZAMA; i++) {
          pinMode(LED_TOMB[i], OUTPUT);
     }
}
```

2. Fejlettebb LED villogtatás

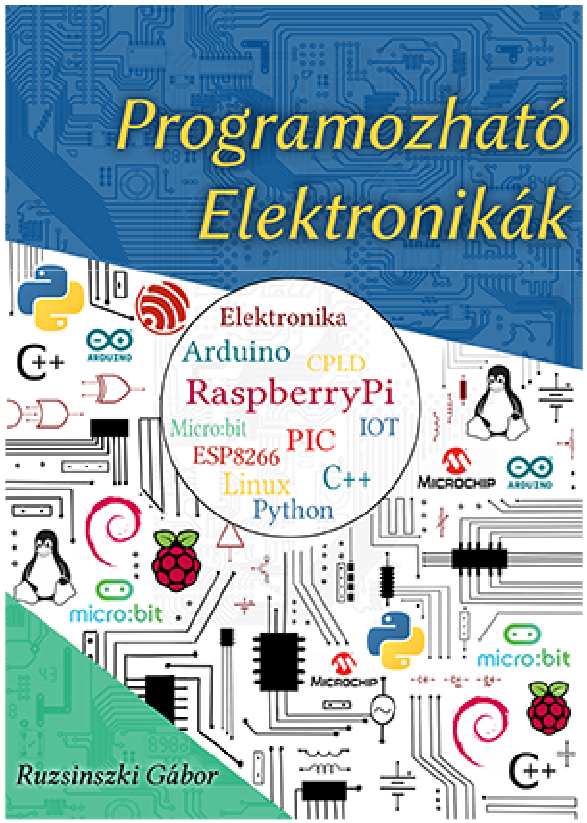
```
void bekapcsol(int melyiket) {
     for (int i=0; i<LEDEK_SZAMA; i++) {
          if (i == melyiket) {
               digitalWrite(LED_TOMB[i], LOW);
          }
          else {
               digitalWrite(LED_TOMB[i], HIGH);
          }
     }
}

void loop() {
     for (int i=0; i<LEDEK_SZAMA; i++) {
          bekapcsol(i);
          delay(200);
     }
}
```

Túl az IDE-n

- Léteznek más eszközök is, amelyekkel komoly munkára fogható a platform ☺
  - Visual Studio*
  - PuTTY/KiTTY
  - Autodesk Eagle
  - EasyEDA

Szakirodalom, érdekes olvasnivalók



- http://bit.do/progelektronikak
- mee-szeged kuponnal 50% kedvezmény
- arduino.com.cc
- Facebook-on:
  Magyar Arduino csoport

Kérdések