



# Óbudai Egyetem

## Neumann János Informatikai Kar

### Beágyazott és érzékelőalapú rendszerek

Lovas István  
[lovash.istvan@nik.uni-obuda.hu](mailto:lovash.istvan@nik.uni-obuda.hu)





# Követelmény

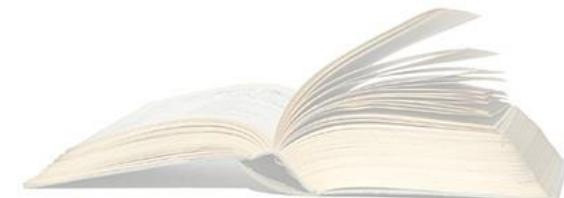
- Labor ZH 50%< = aláírás
  - ZH 13. hét
  - Javítás/pótlás: 14. hét
  - Alpót: 15. hét (Vizsgaidőszak első hete, teljes félév anyagából)
- Vizsga csak az előadás anyagából! Írásbeli
- Kötelező jelenlét ! (Késni max 5. perc TVSZ szerinti 30% hiányzás)





# Mivel foglakozik a tárgy?

- GPIO
- ADC
- UART
- I2C
- Jelgenerálás (PWM)
- Megszakítások
- Timer/Watchdog stb.





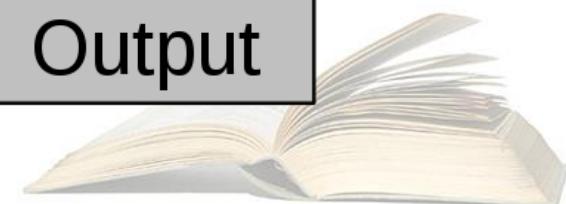
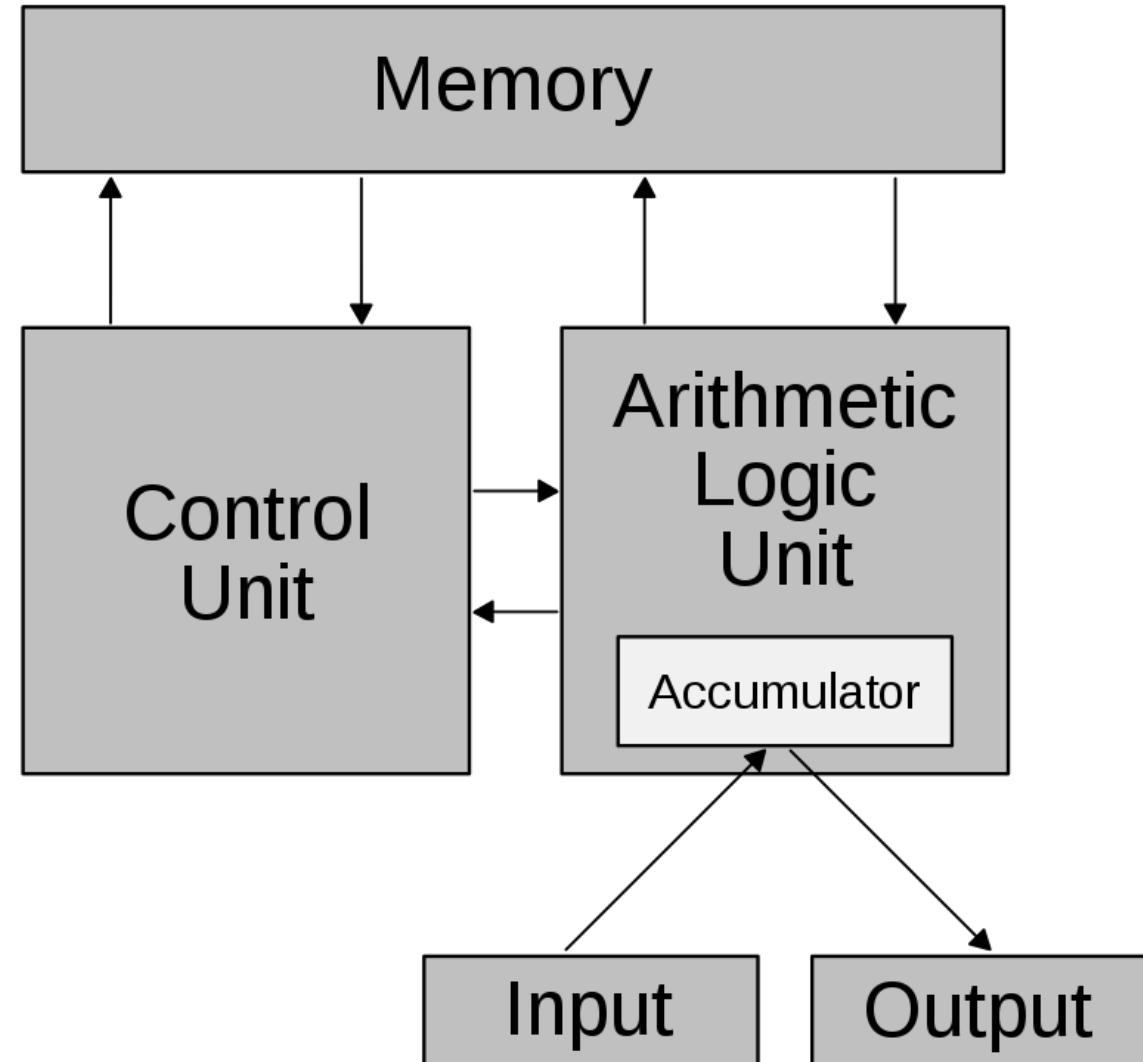
# Neumann-architektúra

- A programkód és az adat ugyan abban a memóriablokkban van eltárolva.
- 5 alapvető funkcionális egységből áll:
  - Bemeneti egység,
  - Memória,
  - Aritmetikai egység,
  - Vezérlőegység,
  - Kimeneti egység.
- A „gép” működését egy a memóriában tárolt programkód végzi.
- Előnye, hogy a programkód és az adat valamelyen külső tárolón vannak eltárolva (pl. HDD), és innen kerül betöltésre a központi memóriába futtatáskor. Ennek köszönhetően utólag könnyű módosítani.





# Neumann-architektúra





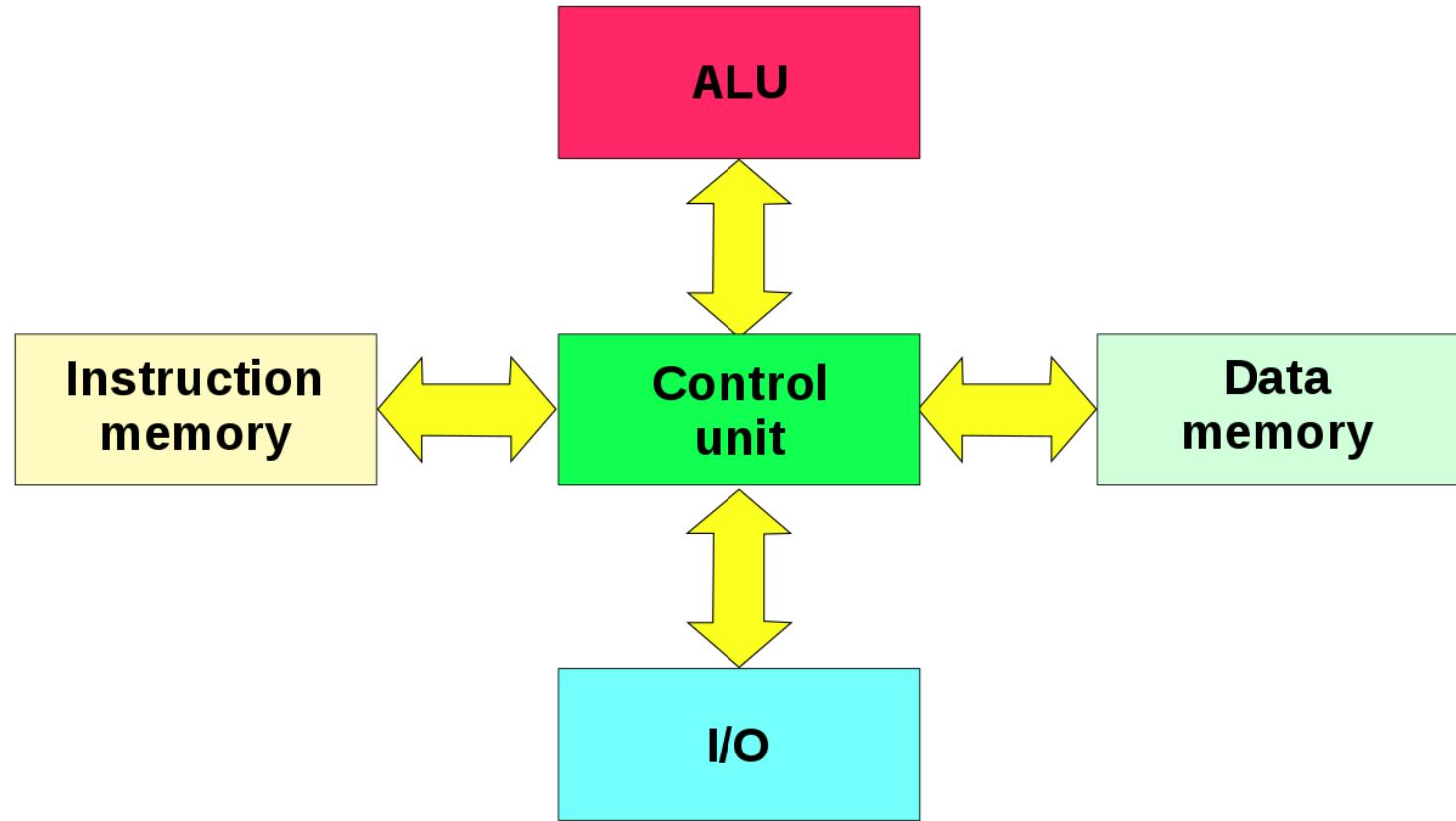
# Harvard-architektúra

- A programkód (sok esetben csak olvasható memóriában) és az adat (írható olvasható) külön van eltárolva
- Ez lehetővé teszi, hogy különböző paraméterekkel rendelkező memóriákat alkalmazzanak (pl. programmemória esetén nagyobb adatcím mint az adat esetén)
- Neumann esetén egyszerre nem lehet utasítást és adatot is beolvasni/írni, mivel ugyan azt az adatsínt használja, viszont Harvard esetén ez egyszerre lehetséges a kialakításnak köszönhetően.



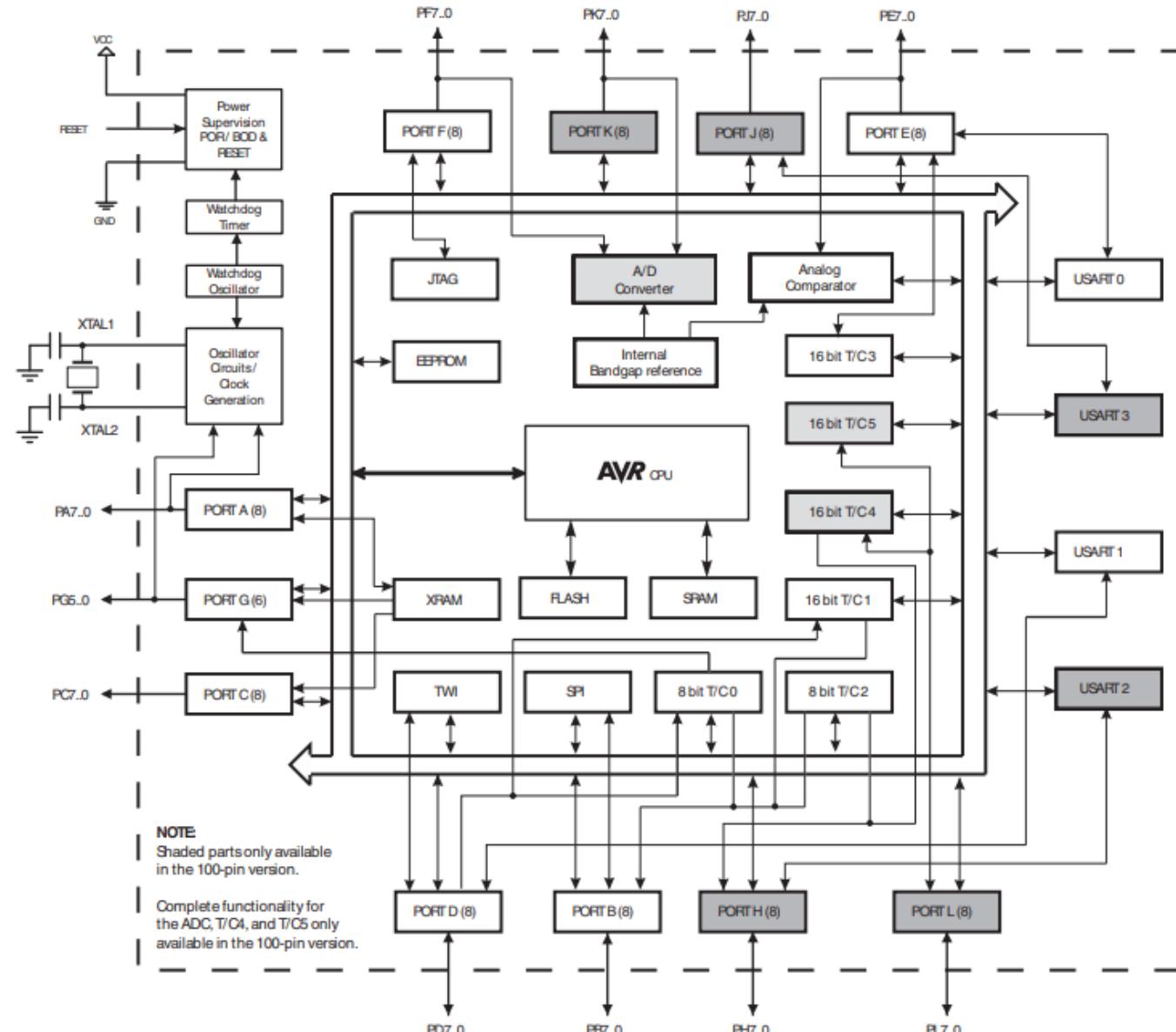


# Harvard-architektúra



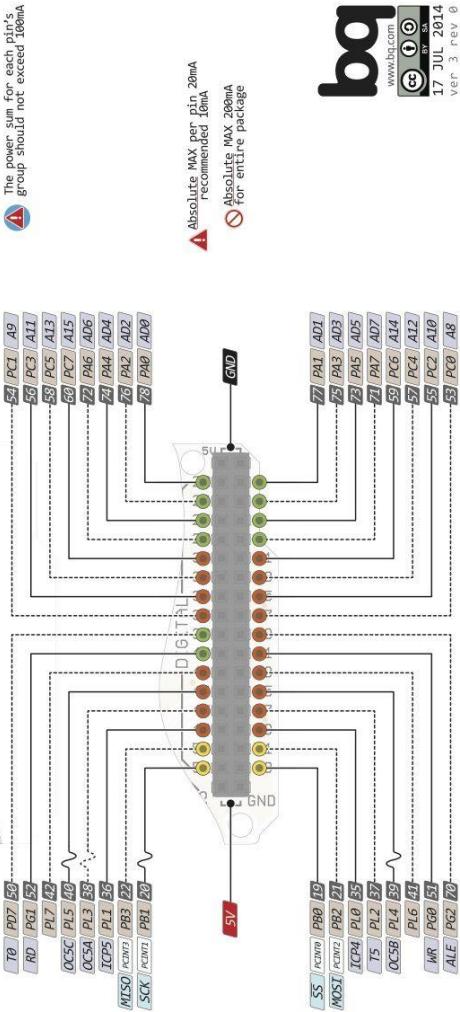
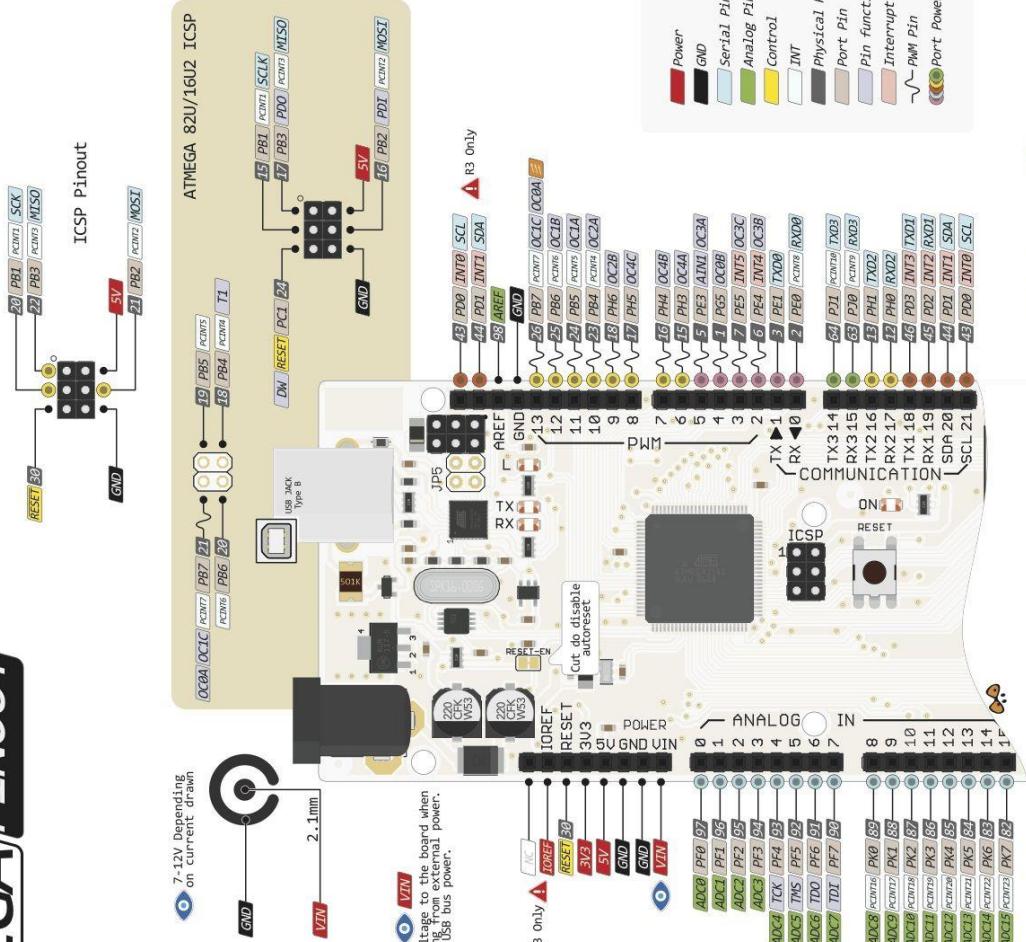


# Mikrokontroller ATmega2560 blokkvázlata





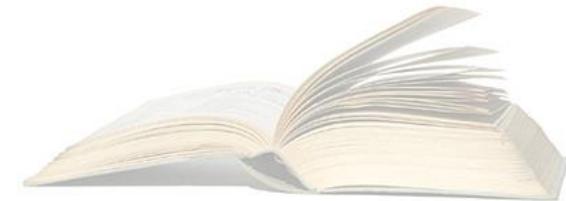
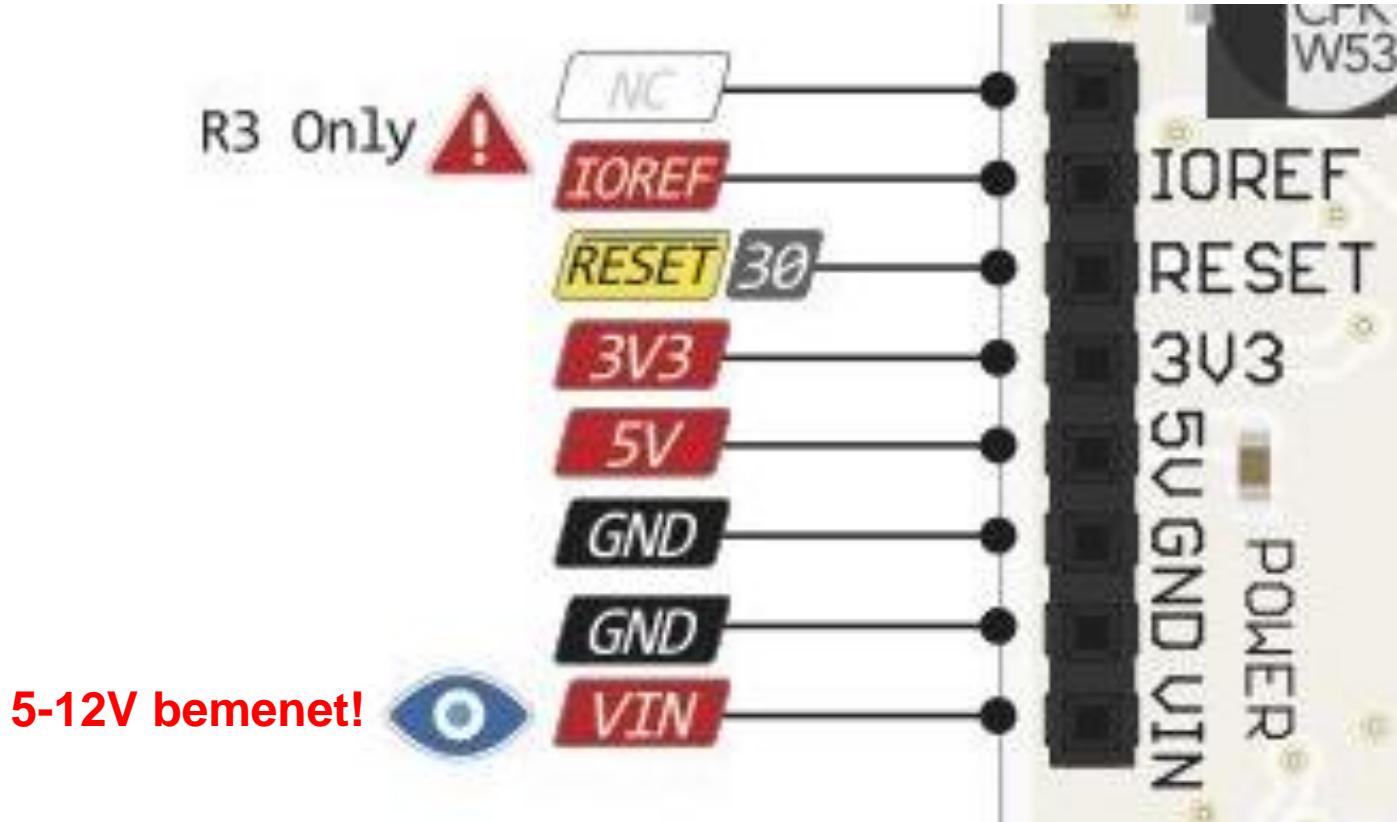
MEGA PINOUT



[www.uni-obuda.hu](http://www.uni-obuda.hu)

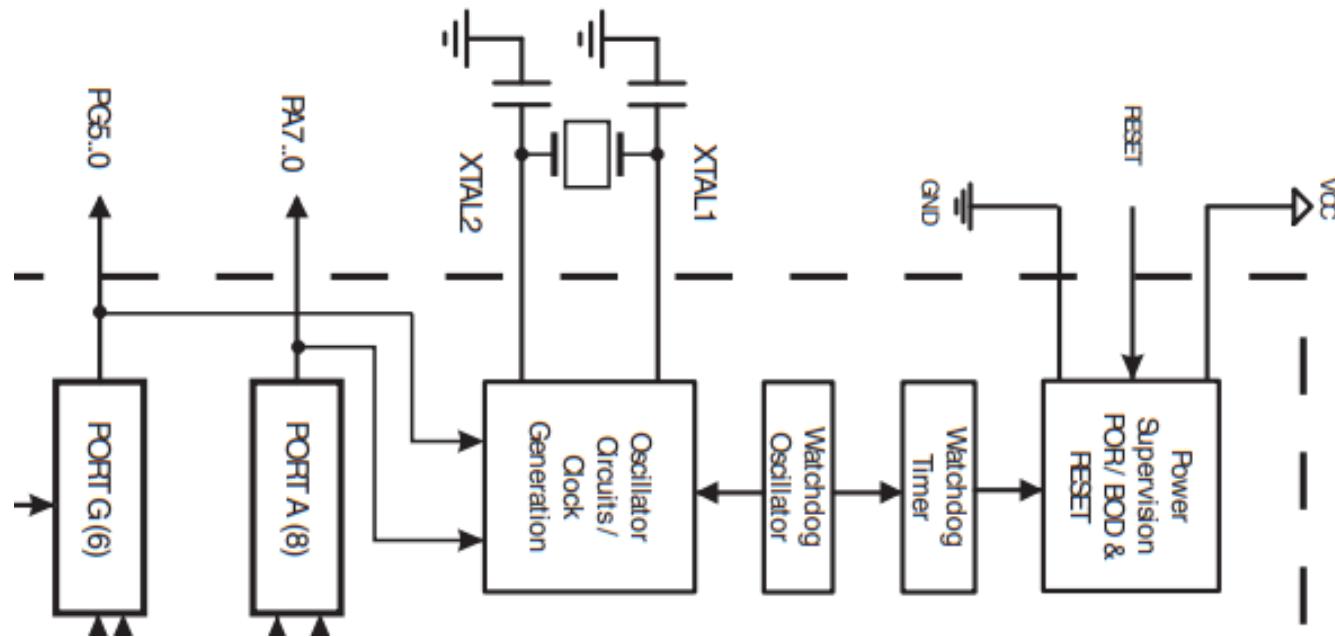


# Arduino ATmega2560 tápcsatlakozás





# Arduino ATmega2560 Tápcsatlakozás / órajel





# Arduino Atmega2560 Tápcsatlakozás / Órajel

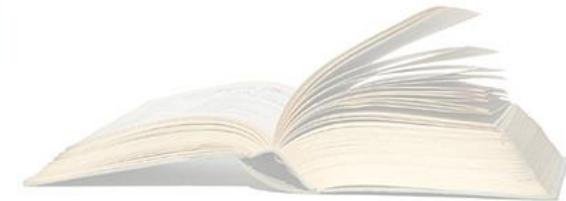
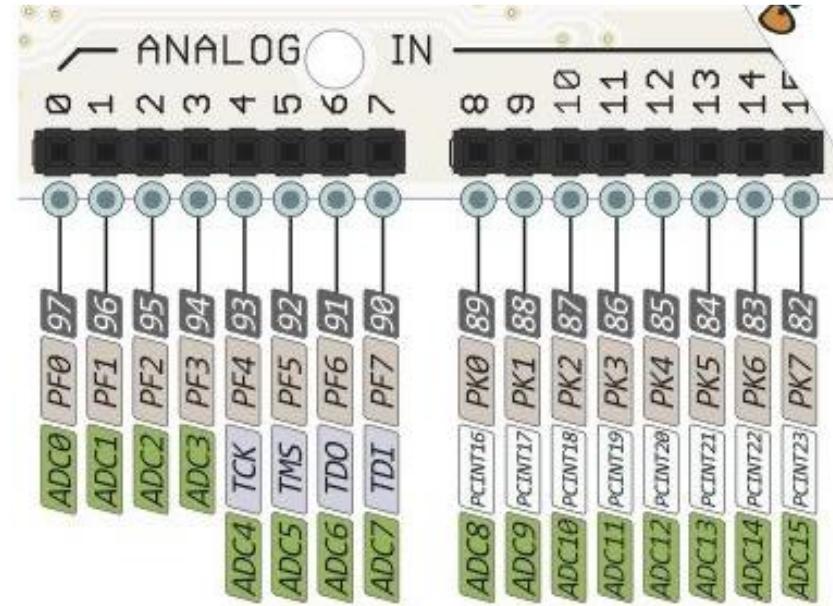
- Tápcsatlakozó: 5-12V
- 5 és 3.3V-os feszültségszabályzó
- 5V jelszint
- Külső 8-16 MHz kristály vagy a belső 8 MHz használható





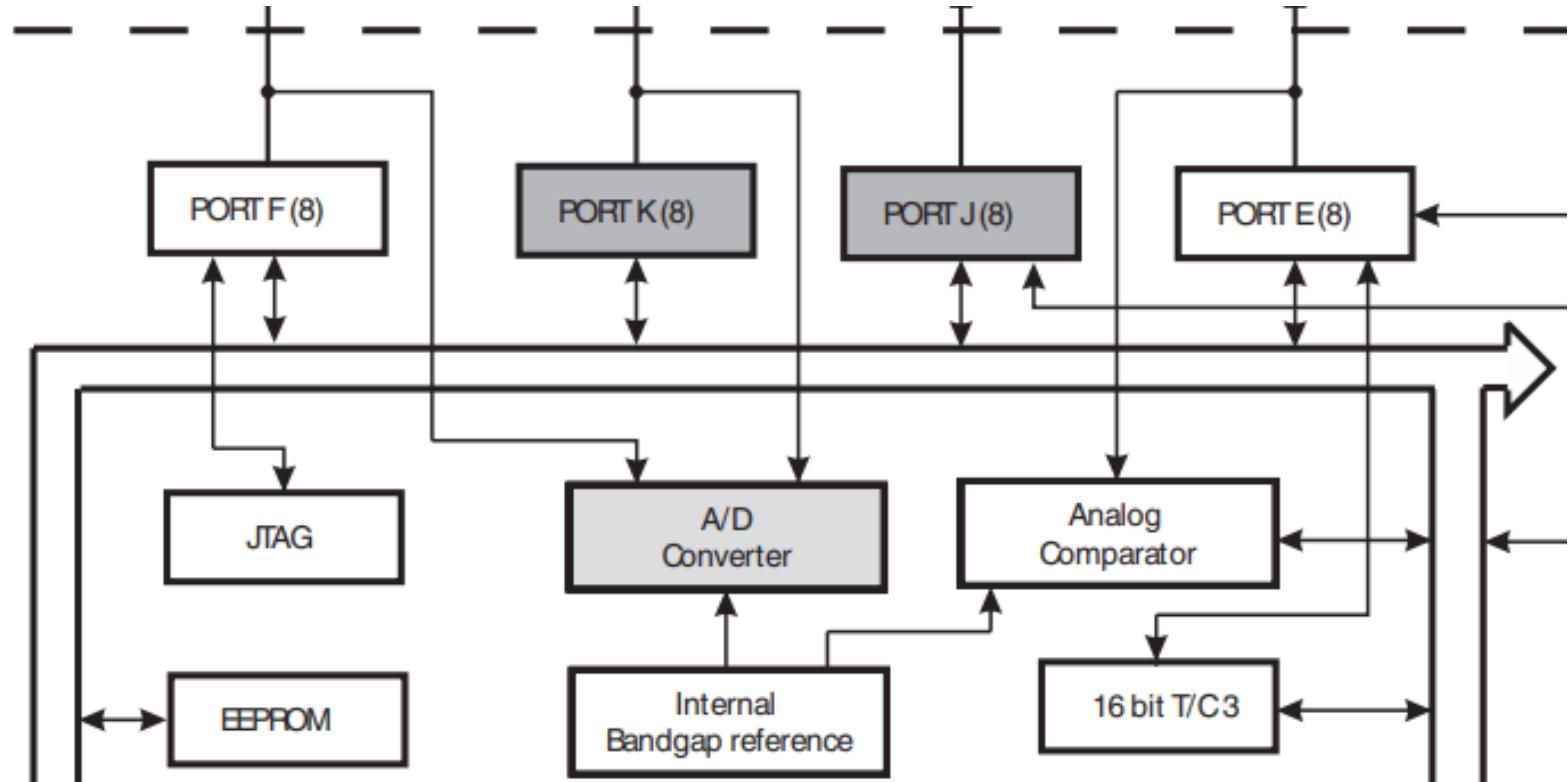
# Arduino Atmega2560 Analóg bemenetek

- 10 bit-es ADC
- 0-5 V között
- Digitális IO-ként is használható





# Arduino Atmega2560 analóg bemenetek





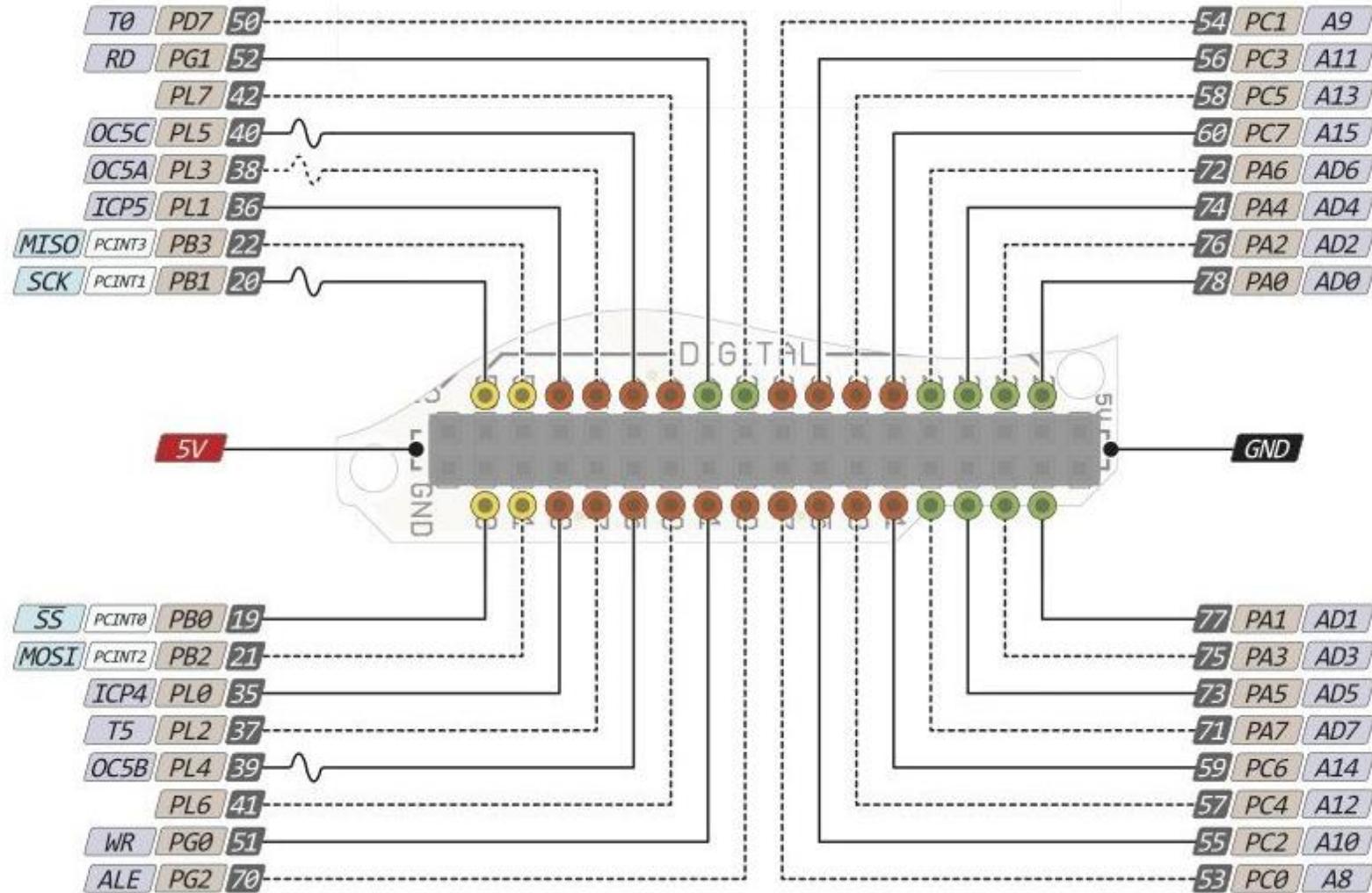
# Arduino Atmega2560 analóg bemenetek

- Felbontása: 10 bit
- $13\mu\text{s} - 260\mu\text{s}$  Conversion Time
- Up to 76.9kSPS (Up to 15kSPS at Maximum Resolution) (kSPS - Kilo Samples Per Second)
- Összesen 16 csatorna (ebből 4 csatorna esetén lehetőség van 10x és 200x erősítésre is)





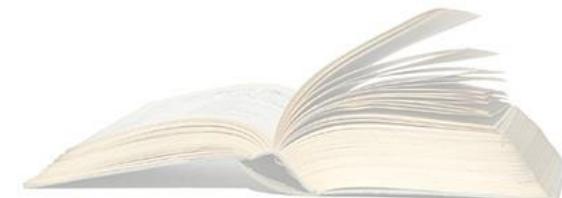
# Arduino ATmega2560 Digitális I/O-k





# Arduino ATmega2560 Digitális I/O-k

- Valójában az összes ( kivétel táp, órajel, referencia feszültség ) lába használható be vagy kimenetnek.
- Bemenet : 3V-tól logikai 1 szint (true, HIGH)
- Kimenet terhelhetősége:
  - kimenetenként max. 40mA
  - összesen maximálisan 200mA !!!





# Arduino ATmega2560 digitális I/O-k

- Mindegyik I/O csoporthozhoz tartozik regiszterek ( n => A-F)
  - PORTn (Pin Output Register) – kimenet esetén az állapotot leíró regiszter
  - PINn (Pin Input Register) – bemenet esetén az állapotot leíró regiszter
  - DDRn (Data Direction Register) – adott láb ki vagy bemenet beállítására szolgáló regiszter
- Az I/O-k 8-as csoportokba vannak rendezve ezért a regisztereik 8 bitesek és értelemben szerűen egy I/O egy bitnek felel meg.





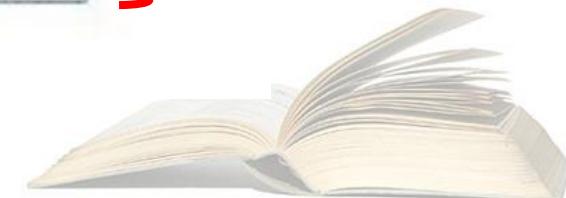
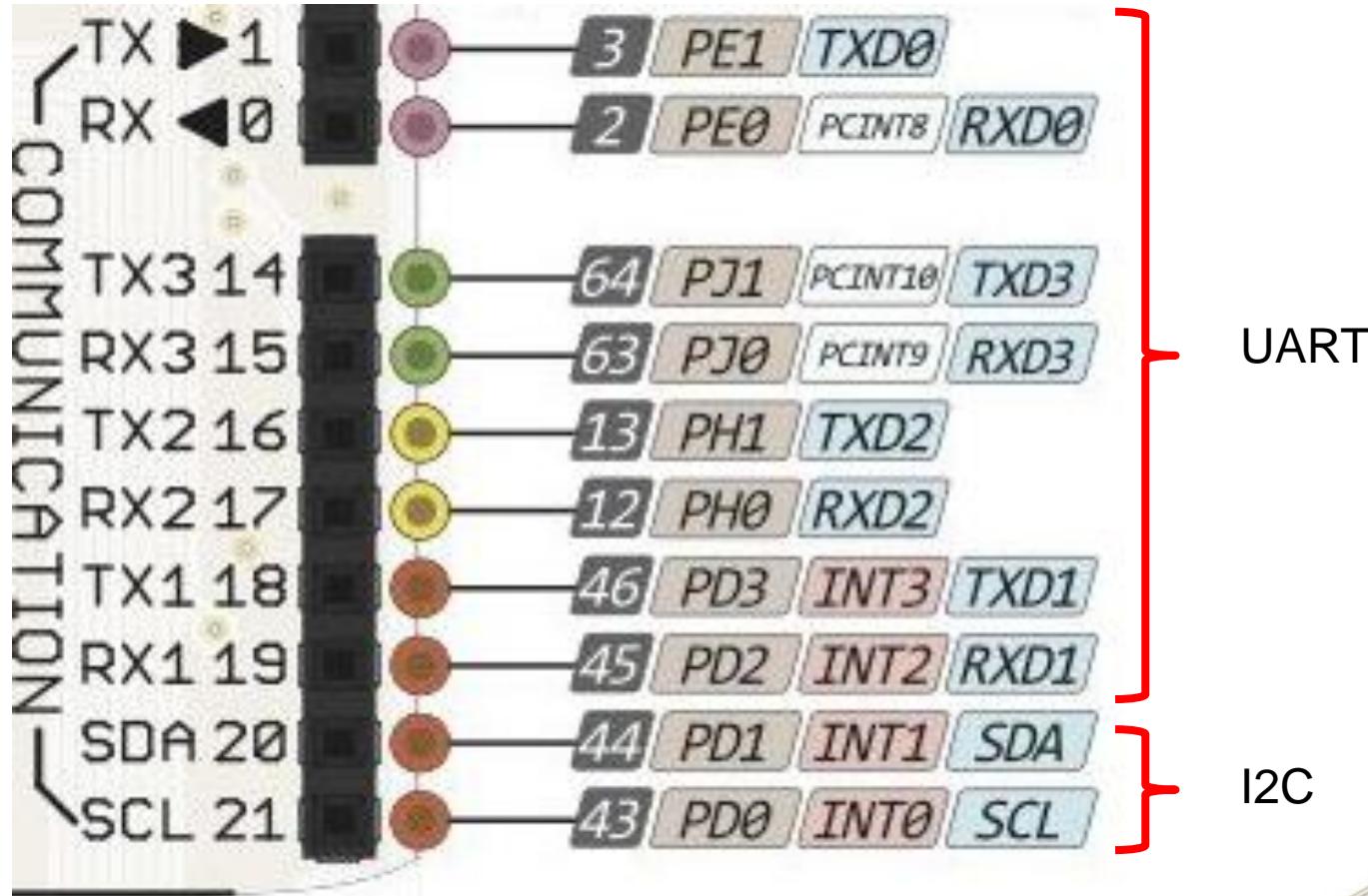
# Arduino ATmega2560 digitális I/O-k regiszterállapotok

DDRA0	PINA0	PORTA0	Állapot
0	0	0	Bemenet – LOW
0	0	1	Bemenet – HIGH
0	1	0	Bemenet – LOW - PULL_UP
0	1	1	Bemenet – HIGH - PULL_UP
1	0	x	Kimenet – LOW
1	1	x	Kimenet – HIGH





# Arduino ATmega2560 dedikált kommunikációs portok





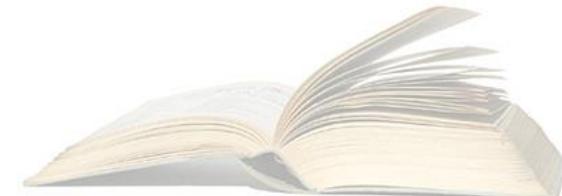
# Arduino ATmega2560 dedikált kommunikációs portok

**UART** (Universal asynchronous receiver-transmitter):

- Aszinkron kommunikáció
- Két eszköz esetén ugyanakkora Baudrate beállítása kötelező (bps)
- Általános baudrate értékek: 4800, 9600, 57600, 115200bps stb...
- Összesen 2+1 vezeték használata kell:
  - Tx - transmit
  - Rx - receive
  - Gnd - közös
- Előnye:
  - Egyszerű, otthoni/labor körülmények között is működik.
- Hátrány
  - Zajvédelem szempontjából nem a legjobb

**USART** (Universal synchronous and asynchronous receiver-transmitter)

- Teljesen ugyan az mint az UART, annyi különbséggel, hogy a kommunikáló eszközök között van egy +vezeték, amin keresztül a szinkronizálás egy órajel végzi.





# Arduino mega 2560 dedikált kommunikációs portok

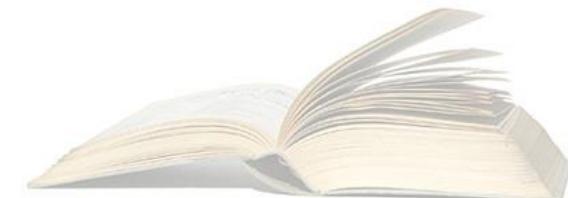
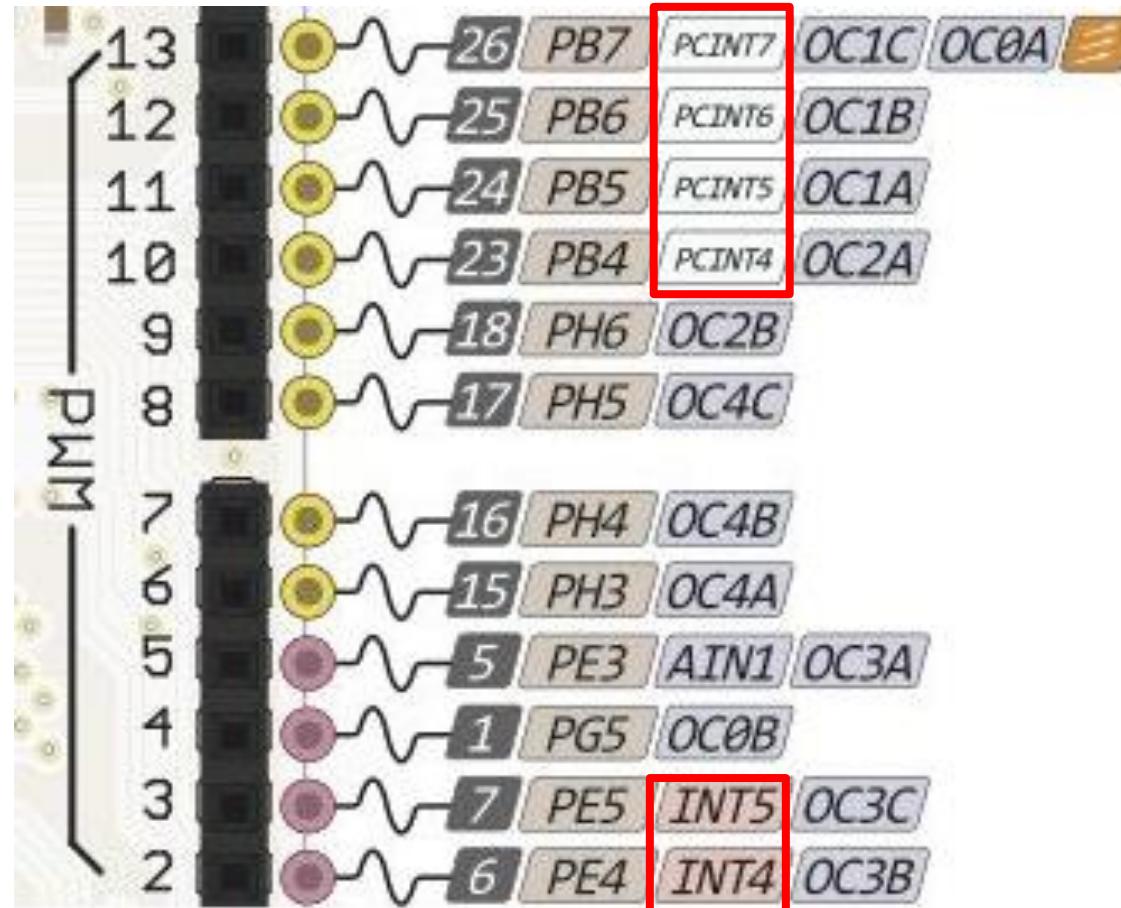
## I<sup>2</sup>C (Inter-Integrated Circuit):

- Széles körben használt kommunikációs módszer, mivel több eszközt lehet összekötni busz topológián keresztül.
- minden kommunikációban részt vevő eszköz saját címmel rendelkezik. (7 és 10 bites címek)
- Hardver szinten benne van a legtöbb mikrovezérlőben, szenzorban
- Master-Slave alapú, azaz minden hálózatban van egy mester eszköz aki megszólítja a többieket.
- NINCS csoportos címzési lehetőség!





# Arduino ATmega2560 PWM, Interrupt, Timer





# Arduino ATmega2560 PWM, Interrupt, Timer

Timer (számláló, időzítő):

- Egyik legfontosabb eleme a mikrovezérlőnek.
- Főbb felhasználása/feladata:
  - PWM jel előállítás,
  - Program ágak időzítése „párhuzamosítás”,
  - Számlálás,
  - Megszakítások generálása.
- 8 és 16 bites számlálóregisztereket használhatnak
  - 8 bit: Timer 0, Timer 2
  - 16 bit: Timer 1, 3, 4, 5
- Speciális Timer -> Watchdog (lsd. Később)





# Arduino ATmega2560 pwm, interrupt, timer I/O-k

## Interrupt (megszakítás)

- Dediált lábak esetén, előre meghatározott állapotváltozások lekezeléséhez lehet megszakításokat hozzárendelni (INT0, INT1, INT2, INT3 portok)
- Megszakítás esetén a hozzá rendelt lekezelő rutin fut le, a futó program megszakítása után.
- Vigyázat! Csak akkora program részt szabad megszakításban végrehajtani, ami biztosan lefut addig, amíg egy következő megszakítás hívás meg nem történik.
- Állapot megváltozások (INTn esetén):
  - LOW, CHANGE, RISING, FALLING, HIGH
- PCINTn (Pin Change Interrupt):
  - PCINT esetén csak a változás tényét detektálja a mikroprocesszor. A tényleges változást szoftveresen kell megvizsgálni (pl. felfutóél esetén, megszakítás előtt a port értéke 0, megszakításon belül pedig már az újra kiolvasott érték 1)





# Köszönöm a figyelmet

