Újrakonfigurálhatóság szenzorhálózatokban

BSc önálló laboratórium, 2014
Door Attila, U9ONC9

BME-Méréstechnika és Információs Rendszerek Tanszék
Konzulens: Orosz György

Bevezetés

- Mérésekhez szenzorokat szoktunk használni
- A szenzort a méréshez beépíthetjük
- Később felmerülhet a programjának a módosítása
- Nem tudjuk rákötni a PC-re
- Távolról kell újra konfigurálnunk

Amin mindezt megvalósítjuk:

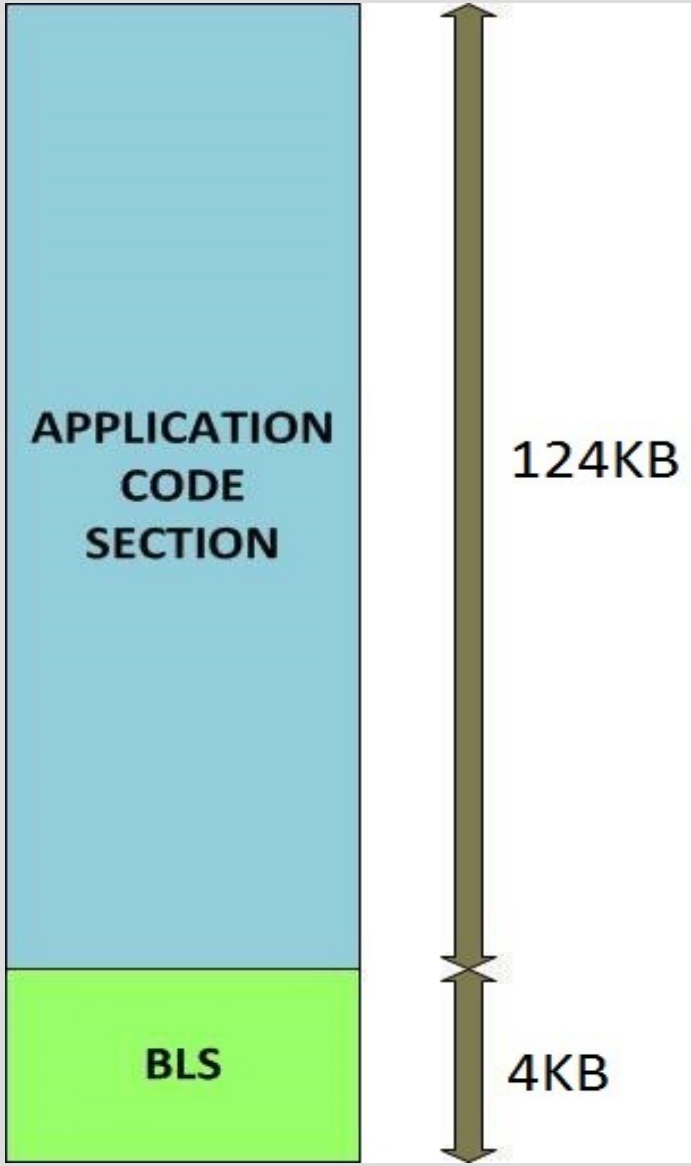
- MitMót: BME MIT saját fejlesztésű platformja
- ATmega128:
  - 128kbyte programmemória
  - Bemenetek:gombok, kapcsolók
  - Kimenetek:LEDek, 7 szegmensű kijelző
- Fejlesztői környezet: AVR studio 4
- Program feltöltése: Happy JTEG-el
- Kommunikáció a PC-vel: UART-on keresztül a HTERM-el

Atmega memória tipusai

- 4kbyte SRAM
  - Kikapcsolás után törlődik
  - Itt tárolja a kontroller a változókat
- 128kbyte flash
  - Itt tárolja a kontroller a programmot
  - Kikapcsolás után nem törlődik
- 4kbyte EEPROM
  - Ezt ebben a feladatban nem fogjuk használni

A memória felépítése

- A flash memória két részből áll
  - RWW: cím tartomány elején
  - NRWW: cím tartomány végén
- NRWW-ban található függvények futási időben módosíthatják a RWW-t
- NRWW csak halt állapotban módosítható
- A BLS section a NRWW-ben található
- Mérete a fuse bitektől függ: a mi esetünkben a legnagyobb (4kbyte)



A program fordítása

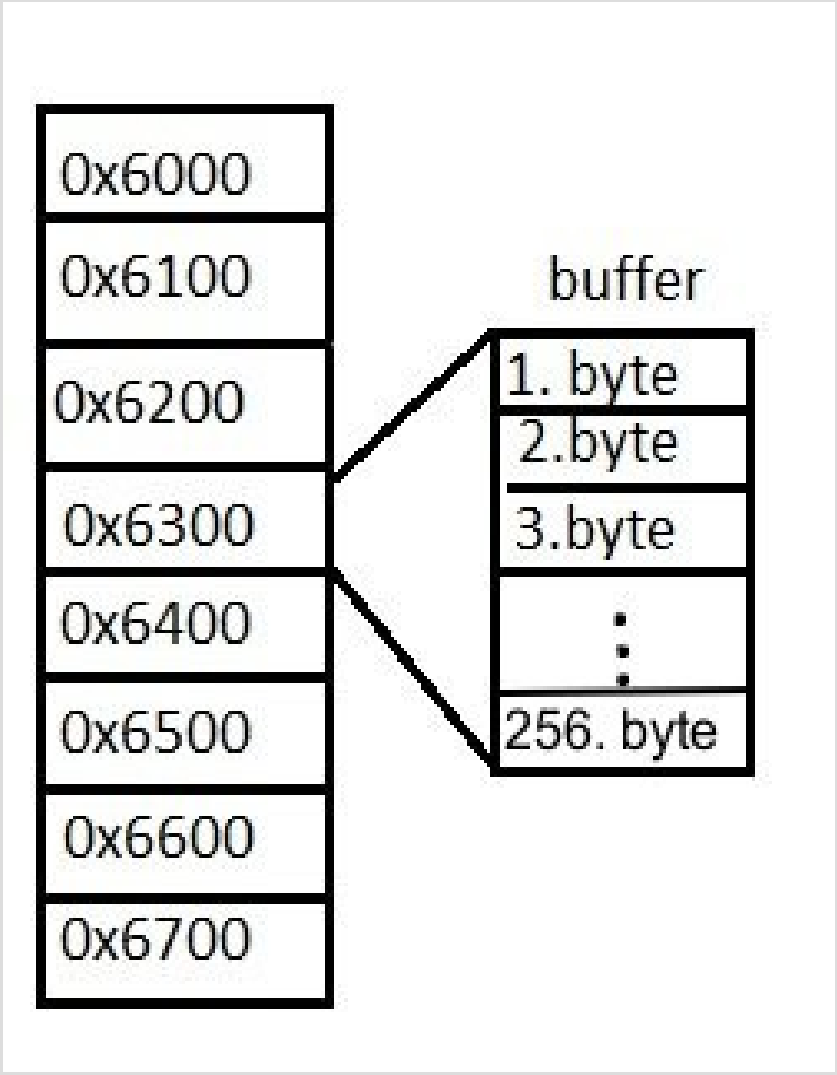
- Bootloadernek a fordításkor a BLS-ben kell lennie
- Makefile-ban tujduk megtenni
  `LDFLAGS+=-Wl,-section-start=.bootloader=0x1F000`
- Függvények helyének megadása C-ben:
  ```
  void bootfv(void)
  __attribute__((section(".bootloader")));
  ```
  - A boot függvény a 0x1F00 címtől kezdődik
  - Kimeneti hexa fájlban látszik is:
    ```
    :10F000000F931F93CF93DF93CDB7DEB7CA50D9408C
    :10F010000FB6F894DEBF0FBECDBF0E94B4F940E03A
    ```
  - A lock bitekkel engedélyezni kell az SPM folyamatokat (Self programming)

Kihívások

- Több alkalmazás létezik, de rosszul dokumentáltak
- Nehezen visszafejthető
- Sok fajta kontroller (családon belül is eltérések)
- Vagyis kell egy saját, feladatspecifikus megvalósítás
- Kontroller alapos ismerete szükséges
- Sok függhet a fejlesztő környezettől
- Az újabb fejlesztő környezetek már sok mindent biztosítanak a felhasználónak, ami nagyban meg tudja könnyíteni a munkánkat

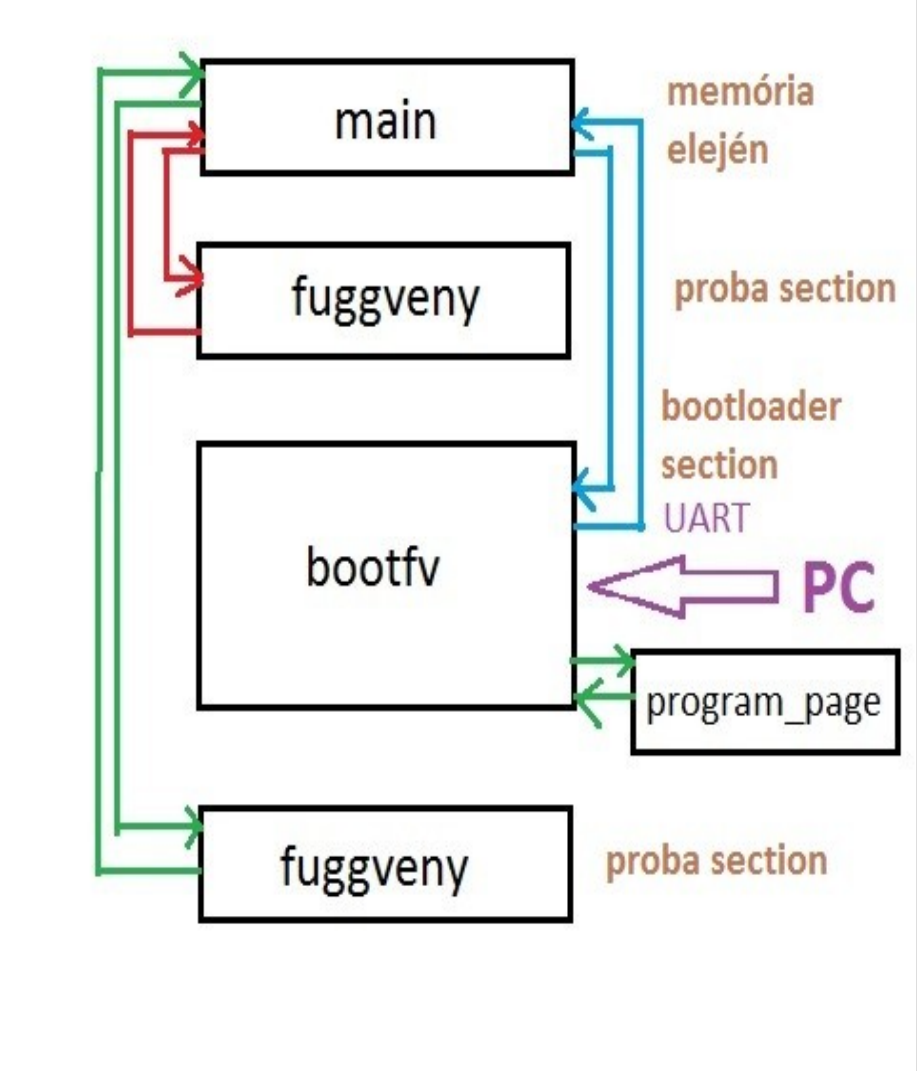
Flash kezelése

- Flash memória page-ekbe rendezve
- Egy page mérete 256 byte
- Flash írása page-enként
- Flash kezelése könyvtári függvényekkel: boot.h



Program futása, függvényhívások

- `main` fv. a memória elején
- Proba “`fuggveny`”-ünk a “`proba section`”-be
- Meghívjuk az eredetit
- `bootfv` hívása, gombnyomás után várja az adatot az UART-on keresztül
- `program_page` beírja a buffer tartalmát a program memóriába
- Módosított “`fuggveny`” hívása



Program futása, függvényhívások

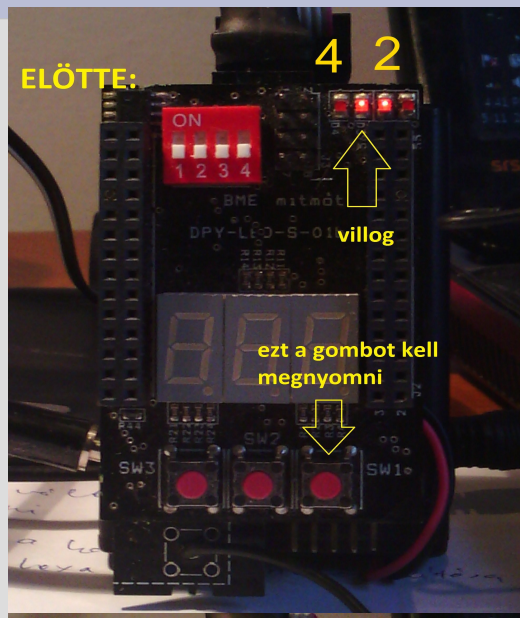
A kimenti hexa kód:

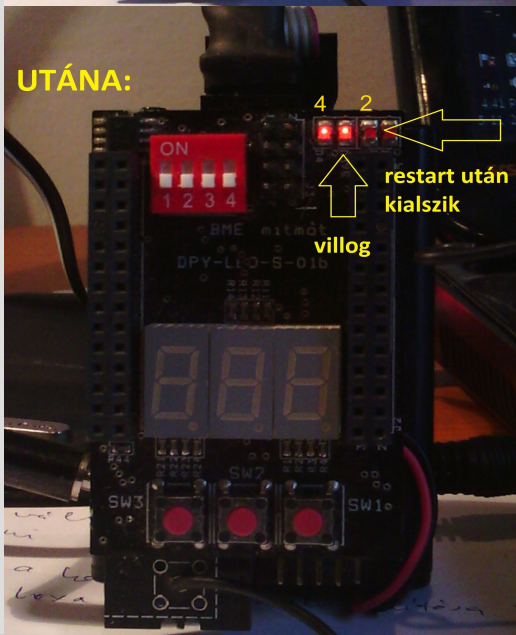
:10600000CF93DF93CDB7DEB7AAE3B0E0....
:1060100080818062**2**8C93ABE3B0E0EBE3....
:0A6020008062**2**8C93DF91CF91089508

Kiolvasás után a hexa kód:

:10600000CF93DF93CDB7DEB7AAE3B0E0....
:1060100080818068C93ABE3B0E0EBE3....
:106020008068C93DF91CF910895ACAC...
:10603000ACACACACACACACACACACACA....

Rátöltöttünk a 0x6000 címre egy függvényt ami kivillantja a 2-es ledet, utána ezt felülírtük egy függvénnyel ami kivillantja a 4-es ledet. A hexa kódból jól látszik ,hogy hanyas számú led változott. A 2-es led csak egy restart után fog kialudni. A page fentmaradó helyét 0xAC-val töltöttük fel.





Távlati célok, kitekintés

- Jelenleg csak UART-on keresztül tudunk adatot küldeni, ami maximum 2kbyte hosszú lehet
- Ezt tovább lehet bővíteni egy küldő oldali kliens programmal, és így tetszőleges mennyiségű adatot tudunk küldeni a kontrollernek
- A mostani vezetékes kommunikációval szemben megvalósítható a vezeték nélküli komunikáció a master-slave egység között, a mitmót rendelkezik is erre alkalmas rádiós kártyával
- Különböző hálózati struktúrák megvalósítása, ehhez több slave egységre van szükség