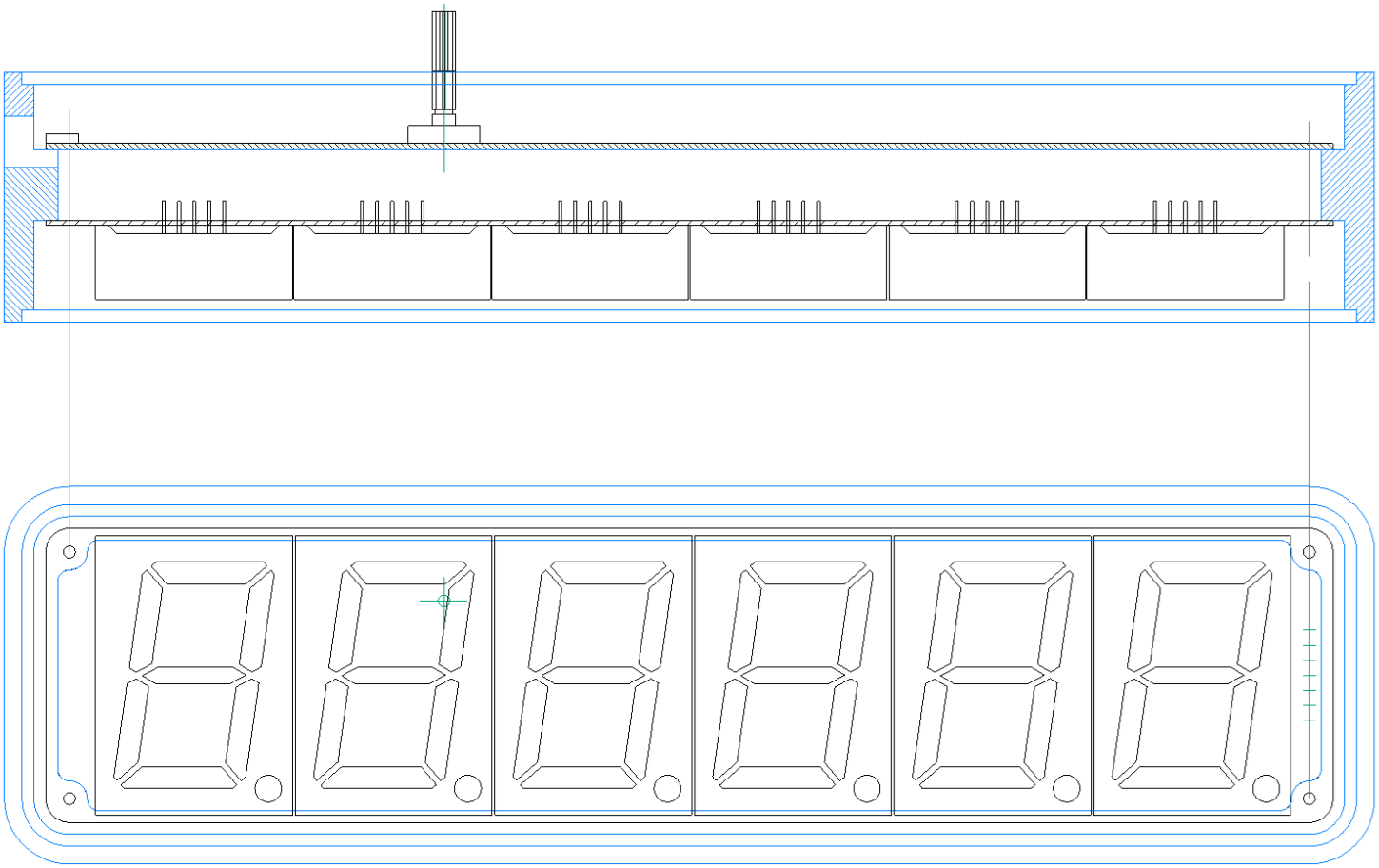
Név: Szabó Benjámin **Neptun kód: RIK88G**

Házi feladat címe: IoT okos ébresztőóra

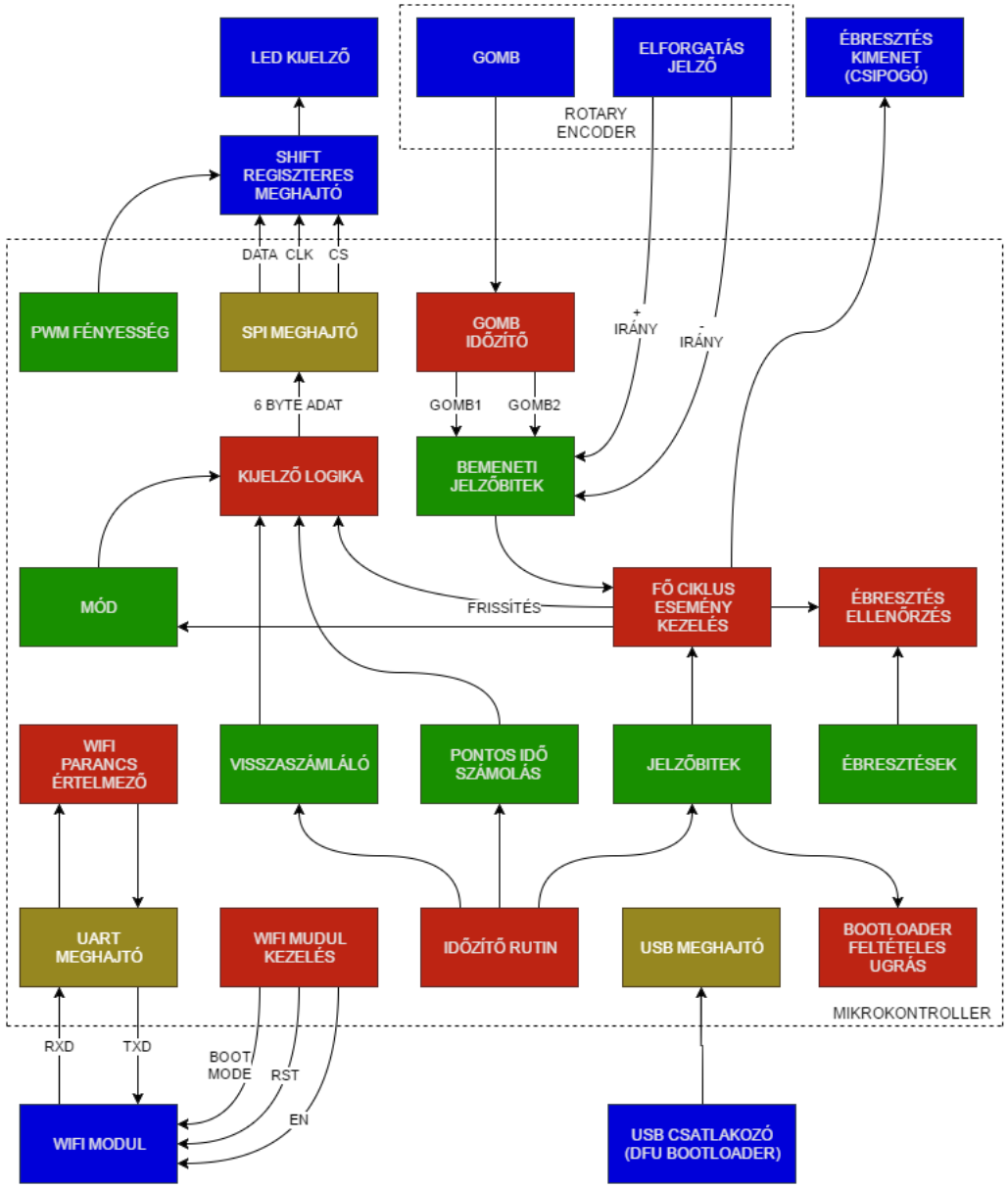
Mikrokontrollerek alkalmazástechnikája
Dokumentáció



Előszó:

A tervezett ébresztőórának rendelkeznie kell egy kijelzővel, személy szerint hat darab hétszegmenses LED-es kiejlzőt választottam. Továbbá másik kucsfontosságú tényező, hogy igény esetén az ébresztés funkció fejleszthető legyen (hangosabb), vagyis teteszőleges eszközt képes legyen kapcsolni, akár relével is. Ezen felül szoftverét lehessen frissíteni, lehetőleg egyszerűen, USB-n keresztül. Ha igény van rá, tetszőleges funkciókat lehssen USB-n keresztül ellátni, például soros port vagy számítógépes beviteli eszköz. És hogy tényleg „okos" legyen, WiFi-n keresztül tudjon csatlakozni egy hálózatra, hogy ott később más okos eszközöket tudjon vezérelni, például okos kapcsolót (tervben van a megépítése), amivel reggel fokozatosan lehetne felkapcsolni a villanyt (vagy akár közvetlenül is), sötétítőt kihúzni vagy egyéb, a készülék funkcióját érintő folyamatot vezérelni.
A megvalósított kapcsolást nem mellékelem, ugyanis több konstrukciós hiba is felmerült, amiket természetesen a kapcsolásban javítottam, de NYÁK rajzolatban nem, ugyanis túl sok munka lett volna, tekintve, hogy már működött a kapcsolás.
Ilyen volt például a elforgatás mérő gomb, amit szoftveresen nem lehetett pergésmentesíteni (bontott darab) vagy az esp modul boot mode választó bemenetei, amik egyébként általános IO-ként is szolgálnak.

Blokkvázlat:



Magyarázat:

A jelenlegi blokkvázlat értelem szerűen a pillanatnyilag megvalósított hardware/software együttest ábrázolja.
KÉK – harwdvare funkciók
BARNA – belső perifériák
ZÖLD – tárolók, regiszterek
PIROS – szoftveres függvények
Sok apró változó nincs feltütntetve, ami az adott funkció megvalósítását teszi lehetővé, például számlálók a pergésmentesítéshez és második gombfunkcióhoz.

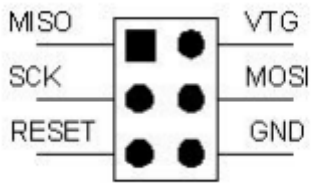
Bootloader:
Az eszközbe olyan bootloader van írva, ami USB-n keresztül, DFU protokollal tudja programozni az eszközt, ezzel fölöslegessé téve külső programozót. Ide belépni egy szoftveres ugrással lehet, amit egy kódzárral láttam el, hogy illetéktelen kezekben ne írják felül a programot, mivel elég egyszerű lenne belépni DFU módba. Ennek ellenére ISP-n keresztül továbbra is tetszőlegesen programozható, nincs lezérva. Arra azonban ügyelni kell, hogy új szoftver esetén is legyen lehetőség a bootloader-be lépni. Ha mégsem, vagy elfelejtettük a kódot, akkor csak ISP-n keresztül tudjuk visszaállítani.

WiFi parancsértelemző:
A kommunikáció úgy történik, hogy a kiválasztott konfigurációk közül tudunk választani a modul indításakor. Két konfiguráció van, AP vagyis elérési pont vagy állomás, ami egy beállított SSID és jelszó párossal fellép a megadott hálózatra, ahonnan később el tudjuk érni (jelenleg a 47-es TCP porton). Az utasítások egy parancskódból és egy paraméterből állnak, így tudunk beállítani bizonyos paramétereket, amiket másképp nem lehetséges jelenleg (kijelző fényerő).
Sajnos az eszköz ismert „hibája”, hogy a felhasznált modul 2 perc után lecsatlakozik, hogy energiatakarékosabb legyen, így csak extra munkával lehetne tényleg jól használni állomás módban. Ezt szerencsére javítható, de a feladat komlexitásán jóval túlmutatna.

Alkatrészlista:
-rotary encoder:ALPS EC11E15244G1
-mikrokontroller: ATMEL AT90USB162
-hétszegmenses: KINGBRIGHT SA15-11GWA
-MOSFET: AP2602GY-HF-3
-kijelző meghajtó: TI TPIC6C595
-wifi modul: ESP8266-03
-feszültségstabilizátor: LD1117V33
-usb csatlakozó: GCT USB3095
-kvarc: 16MHz (USB miatt kell ez)
+ellenállások, kondenzátorok és szükséges átkötések

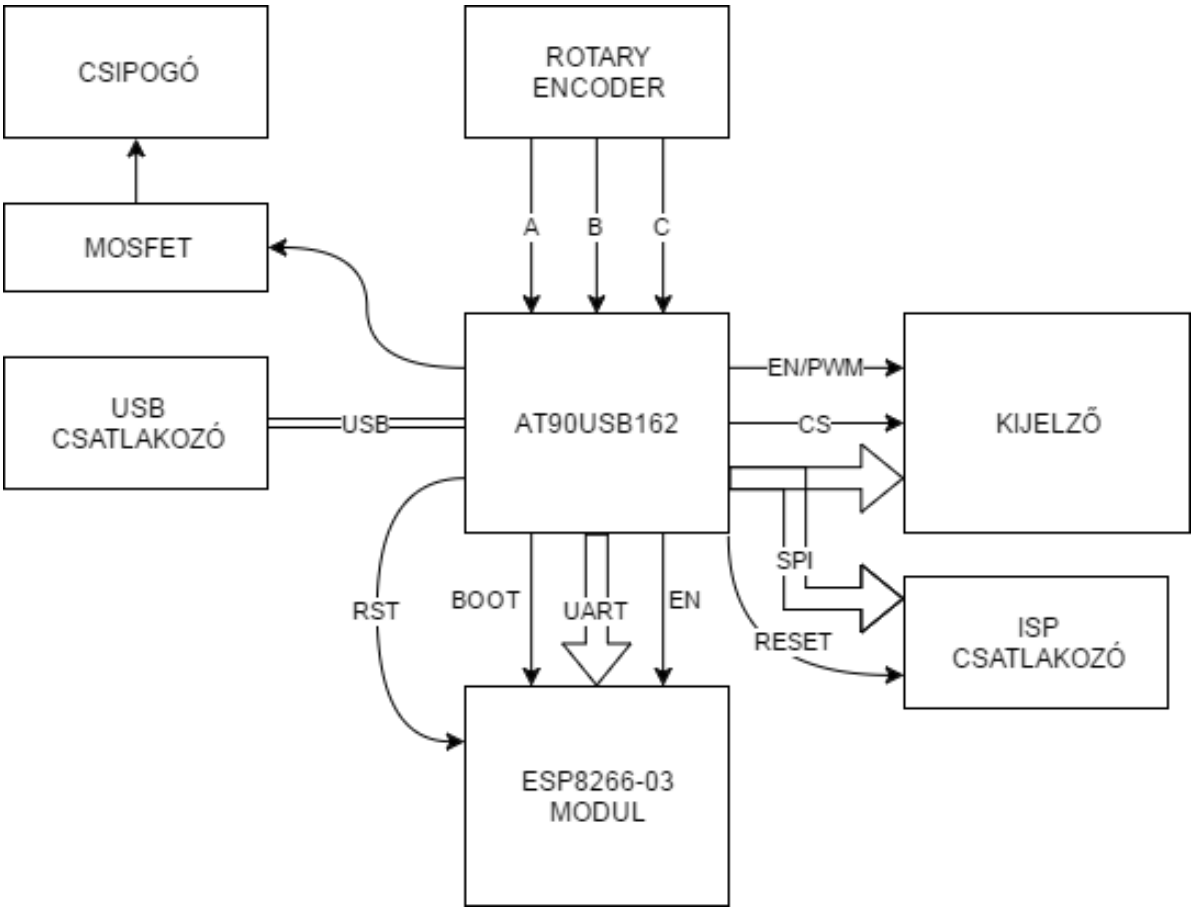
Ébresztés kimenet:
A MOSFET el van látva védódiódával, így akár tekercseket, induktív terheléseket is kapcsolhatunk vele. Ez nyilván kell egy relé vagy rezgőmotor kapcsolásához.
Maximum 6.3 A terhelhetőségű, de vegyük figyelembe, hogy nem erre lett kitatlálva, hűtése nincs erre méretezve.

ISP/debugWire csatlakozó:
Soros programfeltöltéshez a lapra van szerelve egy hat tűs standard kiosztású ISP csatlakozó (AVR Dragon kompatibilis), amivel az eszközt programozni lehet, fuse biteket állítani esetleg hibakereséshez használni. Az első láb pad-je nyégzetes, innen ismerhető meg (saját NYÁK, nincs védőmaszk, se szitanyomás).



ISP (debugWire) lábkiosztás
forrás:
AVR Dragon dokumentáció:
(10., 18., 22. oldal)

Hardver rendszerterv:



A végleges kapcsolás a mellékletben található. Az aktuális hardvert ábrázolja, néhány ellenállás és kondenzátort leszámítva. Sosem árt ha az opcionális alkatrészek is benne vannak a tervben, legfeljebb nem lesznek beforrasztva. Amit lehetett természetesen szoftverből valósítottam meg.

Szoftver:

A jelenlegi program a dokumentációhoz csatolva.
A kódban levő kommentek nagy része angol (lehet maradt magyar is benne).

A fejlesztés Atmel Studio 7-ben készült, WinAVR programkönyvtárral.

Forrásfájlok:

-*sysio.h*

Alacsonyszintű illesztőprogramok, funkciókat látnak el. Céljuk, hogy a hadveres változókat és algoritmusokat elrejtsék, ezzel megkönnyítve a magasabb szintű programozást.

-*sysio.c*
Az illesztő függvények implementációja.

-*sys.h*
Alapvető, a rendszerrel kapcsolatos definíciók, konstansok és függvények gyűjteménye. Viszonylag magas szintű függvények, itt már szinte nem látjuk a hardware-t. Itt van, ami nem nagyon speciálisan csak az adott szoftverhez kell, hanem valósíznű kisebb változtatásokkal később is kell majd.

-*sys.c*
A rendszerfüggvények implementációja.

-*cmd.h*
WiFi parancskódok definíciója.

-*main.c*
A főprogramunk itt lép be. Bootloader feltétel ellenőrzés, szükség esetén ugrás a bootlader címre. A szükséges perifériák és változók inicializációja, az állapotgép futtatása.

A firmware frissítése:

A gyári DFU bootloader van feltöltve, de természetesen tetszőleges bootloader-t is lehetne használni. A hex fájl feltöltéséhez A FLIP programot használom, gyárilag ezt adja az ATMEL (Microchip) hozzá.

Bootloader módba lépéhez a rotary encoder gombját kell nyomva tartani, mikozben a készüléket tápfeszültségre kapcsoljuk. Ekkor hat darab 0 jelenik meg.

A következő helyiértékre lépéshez a gombot az ellenkező irányba forgassuk egyet. További forgatva növelhetjük az értéket. A pont az aktuális helyiértéknél villog.

A kód: **271828**

Ha a beírt kód készen van, nyomjuk meg a gombot. Ekkor a készülék ellenőrzi a bírt kóódot és ha helyes akkor ezt egy (boot) üzenettel jelzi rövid ideig, majd a képernyő elsötétül. Ha helytelen a kód, akkor normál üzemmódban indul az eszköz.

Üzemmódok:

-óra
-visszaszámláló
-ébresztés 1
-ébresztés 2
-ébresztés 3
-ébresztés 4
-wifi konfiguráció

A módok között a gomb rövid megnyomásával lehet körbe menni. Ha valamit szeretnénk beállítani, akkor azt a gomb hosszú megnyomásával lehet megtenni (>0.7sec).

Ekkor a pont az aktuális értéknél villog, máshol sehol nem világít. A gomb forgatásával lehet beállítani a kívánt értéket. Általában az óramutató járásával megegyező elforgatás (tehát negatív elforgatás) a negatív, az óramutató járásával ellentétes (tehát pozitív) elforgatás a pozitív gombot helyettesíti. Vannak persze, amikor ki-be kapcsolásnál az elforgatás iránya lényegtelen.

Egy mód beállítása tipikusan 3 almódra osztható. Ha már beléptünk a beállításba, akkor egy rövid gombnyomással léphetünk a következő érték beállítására. Háromszori megnyomás után visszatérünk a főmódba. Innen negyedszeri megnyomásra a következő főmódba lépünk. Ezen felül hossz nyomásra mindig visszatér a főmódba. (ez nincs az ábrán).