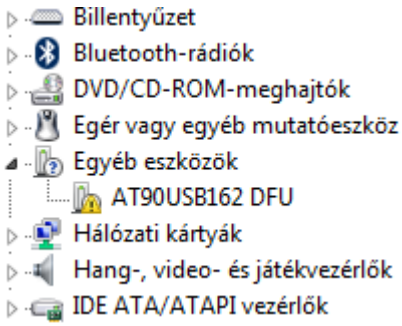
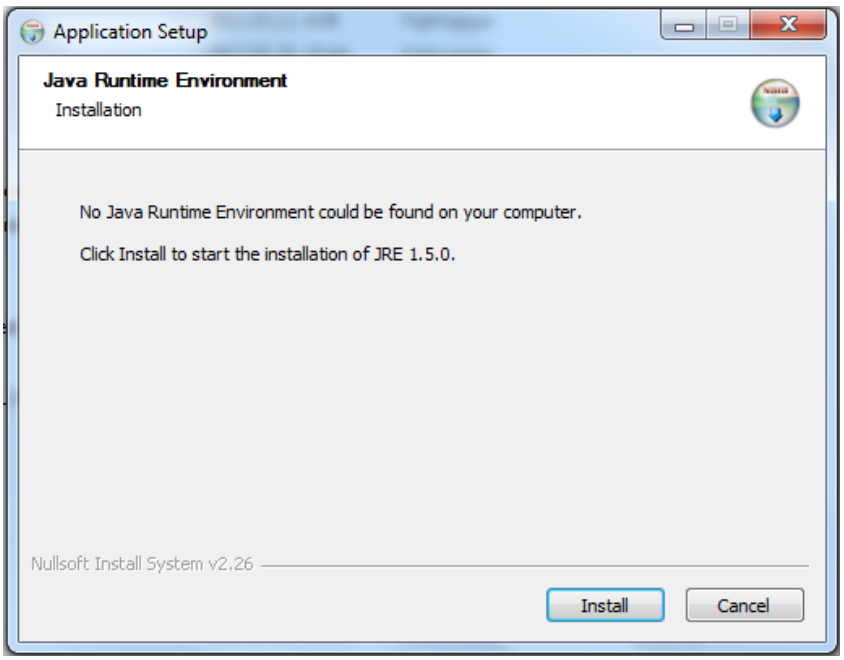
USBTiny-MKII programozó

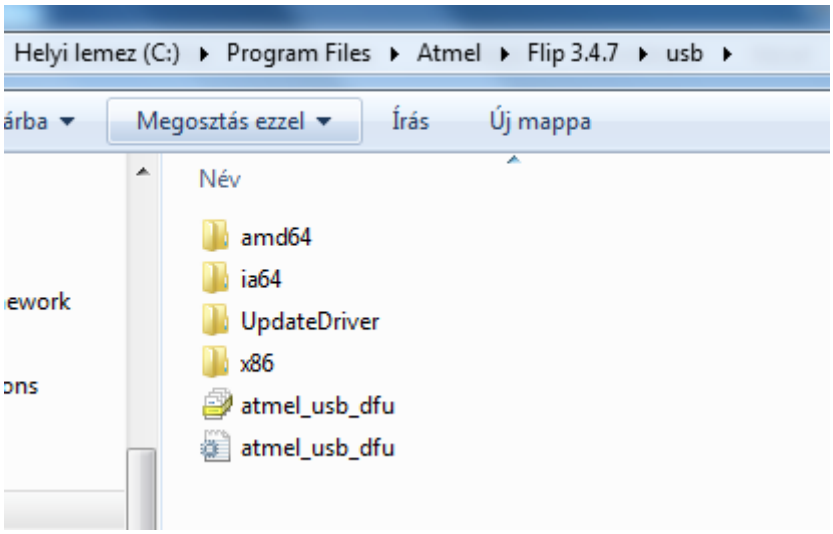
Software telepítés

Miután elkészítettük a programozónkat és az hibátlanra sikerült (forrasztások, összes átkötés – via – megléte, szemrevételezés legalább 12x-s nagyítóval, különös tekeintettel a GTL szintillesztő lábai közötti terület ellenőrzése, stb …) elméletileg azonnal működőképes állapotban kell lennie. Ezért egy USB kábel segítségével azonnal csatlakoztatjuk a számítógéphez. Mivel még egy szűz processzorról van szó, ezért maga az AT90USBxxx jelentkezik. Mivel a gépnek fogalma sincs, hogy mivel van dolga, ezért ezt látjuk az eszközkezelőben: ismeretlen eszköz. Viszont ez egy nagyon jó és megnyugtató hír. Mivel így már biztosak lehetünk benne, hogy a processzor működik és az USB rész szintén, mivel létre jött a kapcsolat a programozó és a PC között. Ez már fél siker, a fele már működik. Ilyenkor az egyik LED-nek sem kell világítania. Ha netán a Win szeretné a meghajtó software-t telepíteni, ne hagyjuk neki, mert teljesen felesleges. Feltéve, ha ez az első telepítésünk, mert különben úgyis felismerte volna. A következő lépés, hogy telepítjük a FLIP-t, amivel életet lehelünk a programozónkba. Ez egy JAVA alapú programozó software, amivel az összes nativ USB-s Atmel processzort tudjuk programozni, ugyanis ezekbe már gyárilag található egy bootloader. Itt már egyébként láthatjuk a bootloader-t működés közben, mivel már felépített egy kommunikációt a PC-vel. Ezért tudja a processzor típusát. Most letöltjük a programot innen: http://www.atmel.com/tools/FLIP.aspx. Kiválasztjuk a nekünk megfelelőt. Ha nem vagyunk biztosak benne, hogy a JRE már telepítve van a gépünkön, akkor ezt „**FLIP 3.4.7 for Windows (Java Runtime Environement included)**” töltjük le. Kezdjük el a telepítést. Ha a JRE nincs telepítve, ezt látjuk:

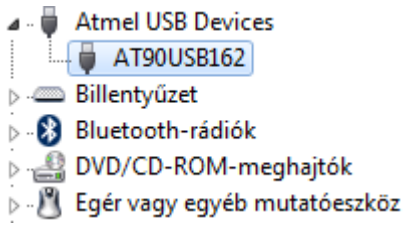




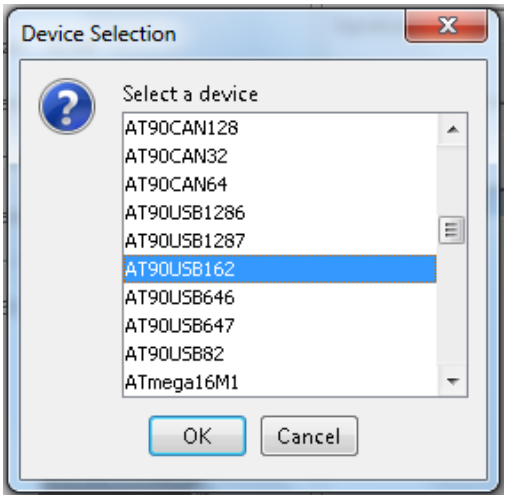
Végig megyünk a telepítésen. Viszont, ha a teljes csomagot telepítjük, vagyis JRE-vel együtt, akkor mindenképpen frissítsük a JRE-t, mert ez már nem a legfrissebb. És érdekes módon, ha a JAVA a háttérben frissít, a Flip hajlamos meglepetésszerűen kilépni. Frissítés után stabil. A telepítés után elballagunk ide:



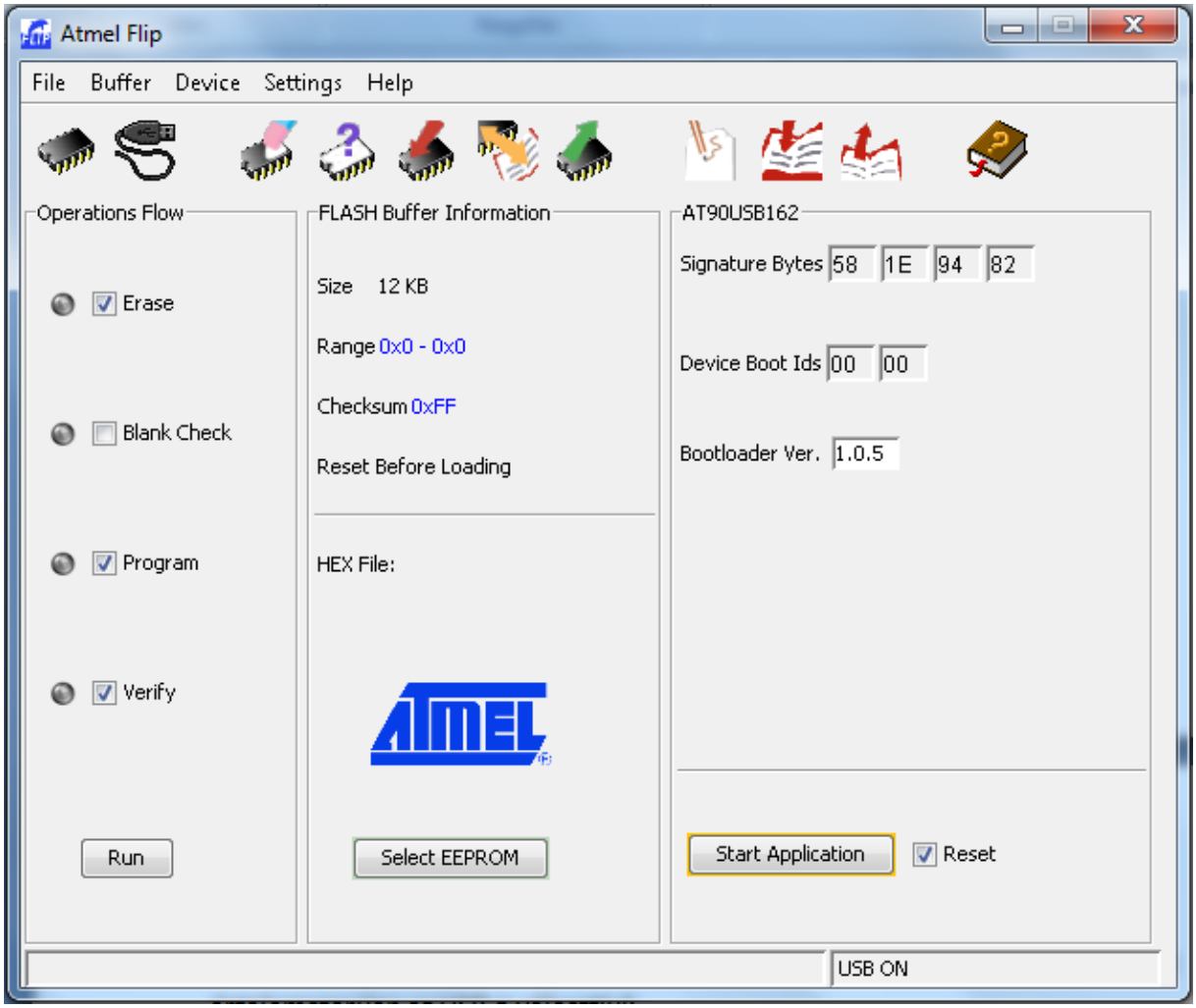
Itt találjuk ugyanis az USB meghajtót, hogy végre működjön az ügy. Ide csak azért jöttünk, hogy lássuk, honnan kell telepíteni a meghajtót. Az eszközkezelőben jobb klikk az ismeretlen eszközre AT90USBxxx, majd illesztő program frissítése. Itt kiválasztjuk, hogy mi keressük meg az illesztő programot. Tehát kijelöljük a C:\Program Files\Atmel\Flip 3.4.7\usb könyvtárat és hagyjuk, hogy telepítse az illesztőt. Ha ez megtörtént, ezt kell látnunk az eszközkezelőben.



Innentől kezdve, már elérhető lesz a Flip számára is. Most már fel tudjuk tölteni az általunk kiválasztott firmware-t a programozónkra. Itt már látható, hogy talán egy kicsit bonyolultnak tűnő előkészítés után már nagyon egyszerű dolgunk lesz. És valóban a firmware csere nagyon gyors. Nem beszélve arról, hogy már látható, hogy ha valaki egy ilyen nativ USB processzorral szerelt áramkört tervez, egyáltalán nincs szükség programozóra. De most hívjuk fel a Flip-t. Persze az áramkörünk továbbra is ott figyel az USB porton. Az első dolgunk, hogy rákattintunk a Chip ikonra, a bal felső sarokban „Select Target Device” és a megjelenő ablakban kiválasztjuk a programozandó chip típusát. Hogy miért nem ismeri fel magától, az számomra egy rejtély. Nem hiszem, hogy rettenetesen nagy problémát jelentett volna, de hát a slamposságnak nincs határa.

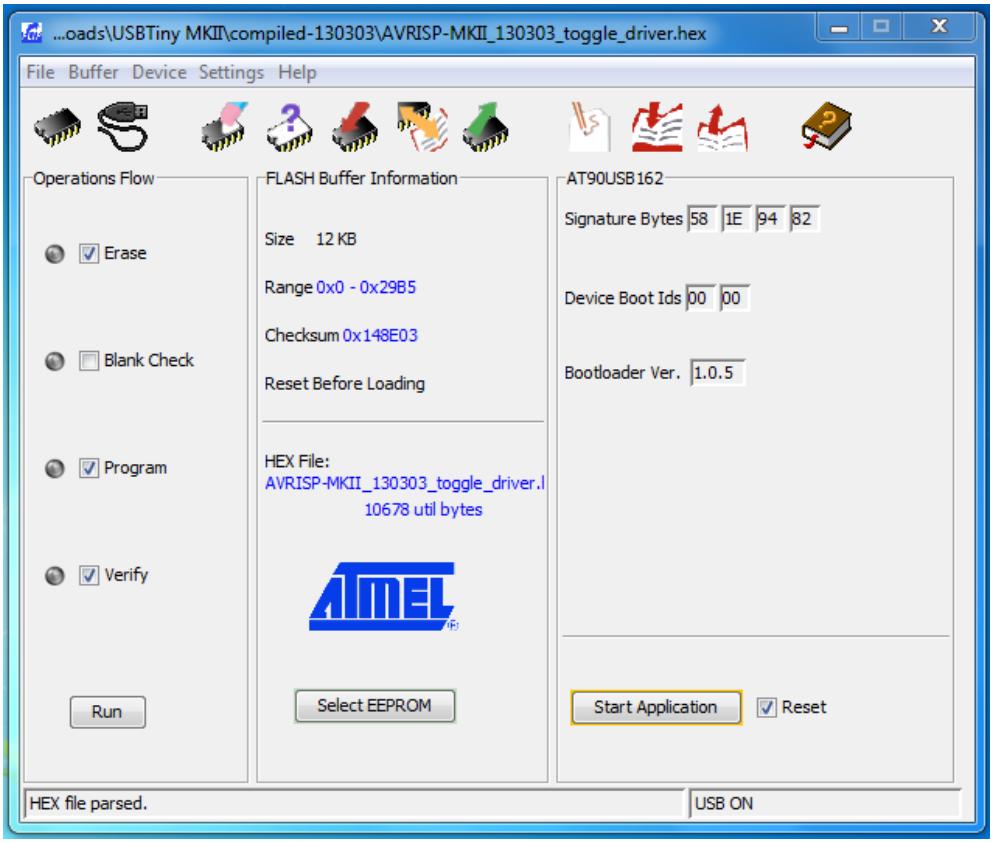


Ezután a kommunikációs vonalat kell kiválasztanunk. Ez az USB kábeles ikon. Itt értelemszerűen az USB-t választjuk. Ezek után megnyitjuk a csatornát és ezt látjuk:

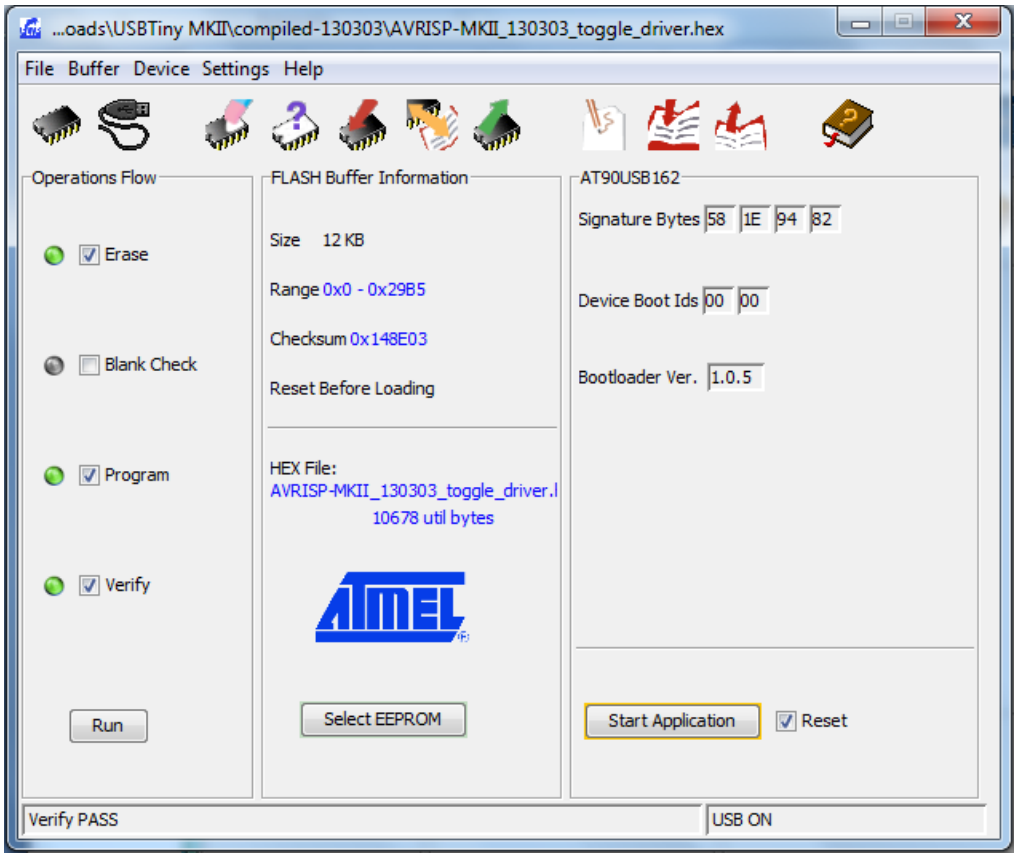


Láthatólag mindent felismert, amit csak kell. A következő lépés az általunk választott hex file, vagyis a firmware betöltése. File -> Load HEX file … vagy a megfelelő ikonra klikkelve betöltjük a HEX-t.

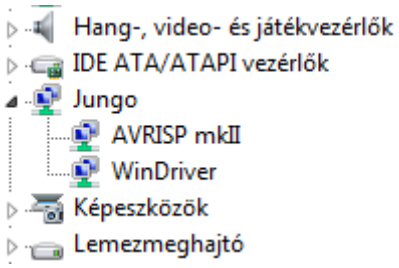
Ezt kell látnunk.



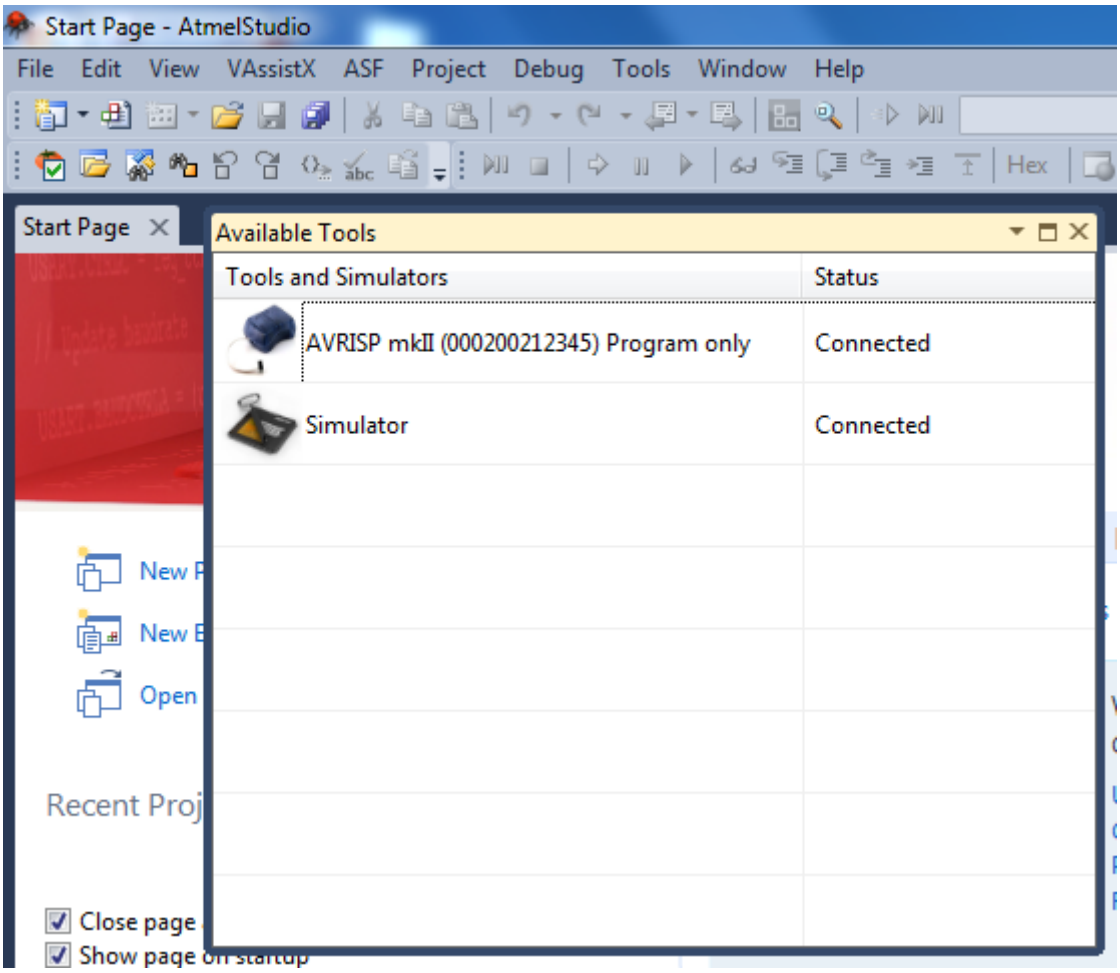
Aztán klikk a Run-ra. A programozás után minden zöld.



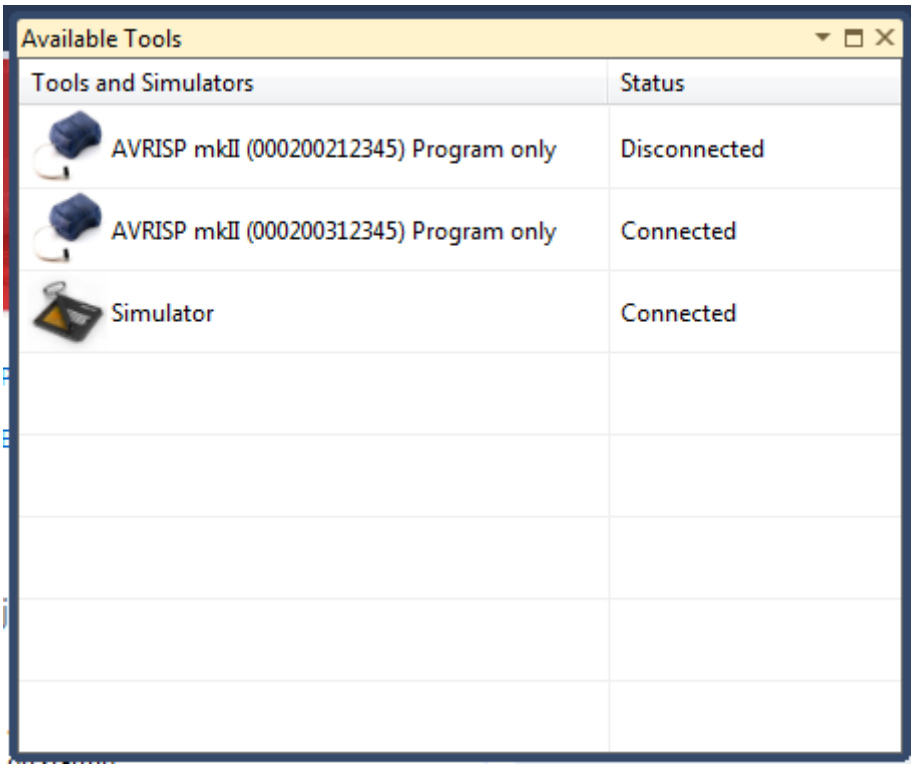
Ezután klikk Start Application ikon. Ezzel indítja a programot a processzorban. Ha az AVR Studio még nem lett telepítve, azt tegyük most meg. A programozót a telepítés alatt válasszuk le a PC-ről. A telepítés után csatlakoztassuk a programozót. A Reset gombját ne nyomkodjuk. Később bővebben, hogy miért. Azonnal felismeri, hogy új hardware és elkezdi telepíteni az illesztő programot. Ha kész van, így néz ki az eszközkezelő:



Már fel is ismerte. Ez már igazi sikerélmény. És már látjuk, hogy ég az egyik LED. Ez a LED mindig ég, ha fut a programozó software. Indítsuk el az Atmel Studiot. A menüpont View -> Available Atmel Tools alatt láthatjuk fáradozásaink eredményét.



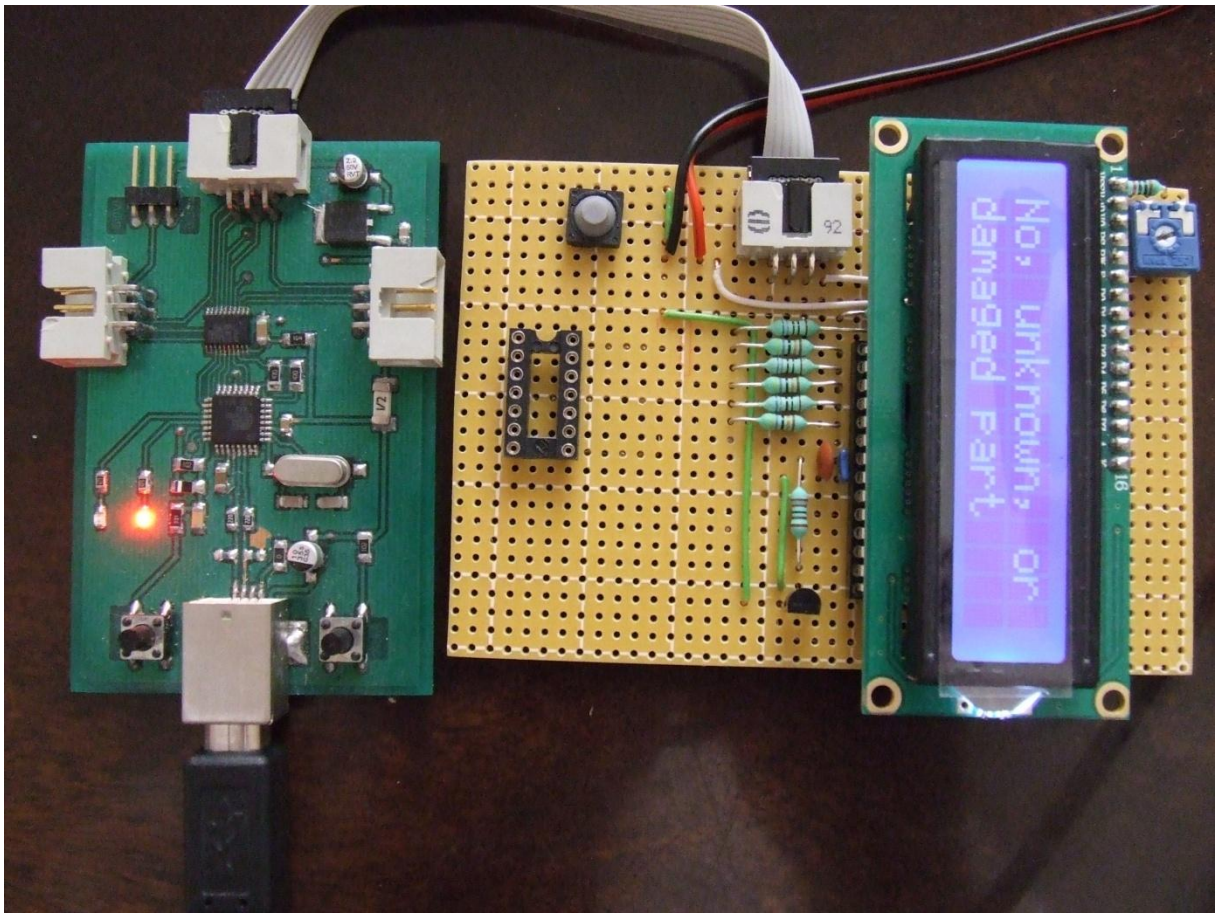
Ez azért már valami. És most egy gyors teszt. Viszont még játszunk egy kicsit, hogy megértsük mit is csinálunk és mink van. Nyomjuk meg a programozón a RESET gombot és nézzük mereven a képernyőt.



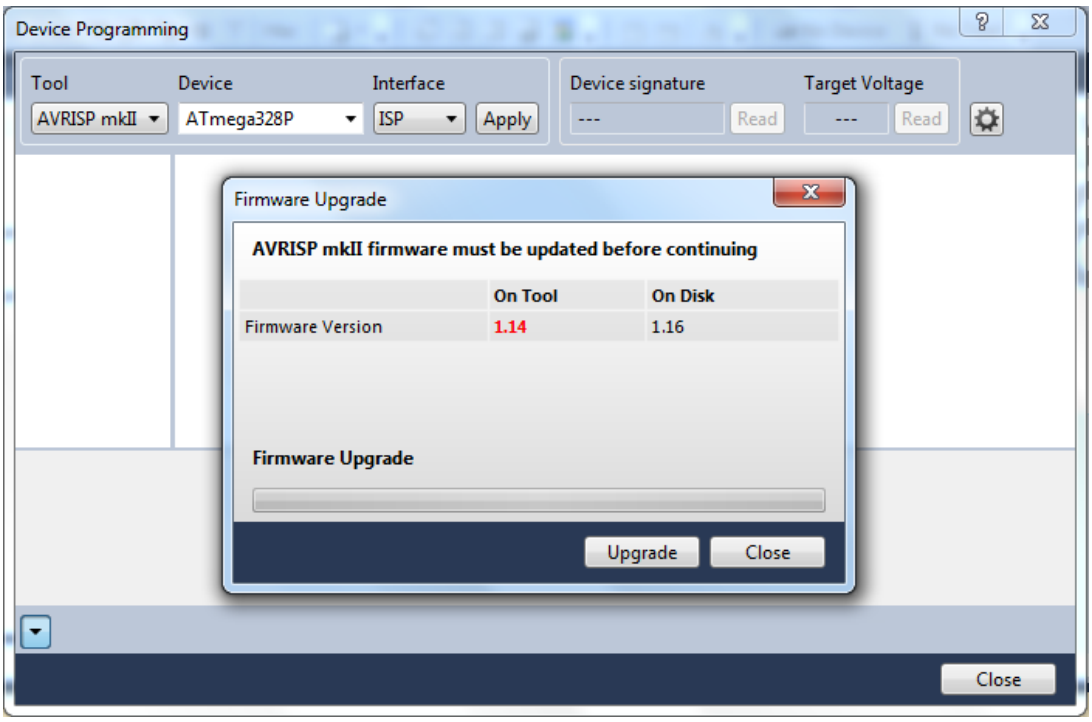
A 212345-s eszközt leválasztotta és csatlakoztatta a 312345-s eszközt. Vagyis mostantól a másik firmware él. Mégpedig a LibUSB-s firmware. Tehát összefoglalva:

AVRStudio Jungo Driver	-> 212345	startnál 2x villog
AVRDude, BASCOM, LINUX LibUSB Driver	-> 312345	startnál 4x villog

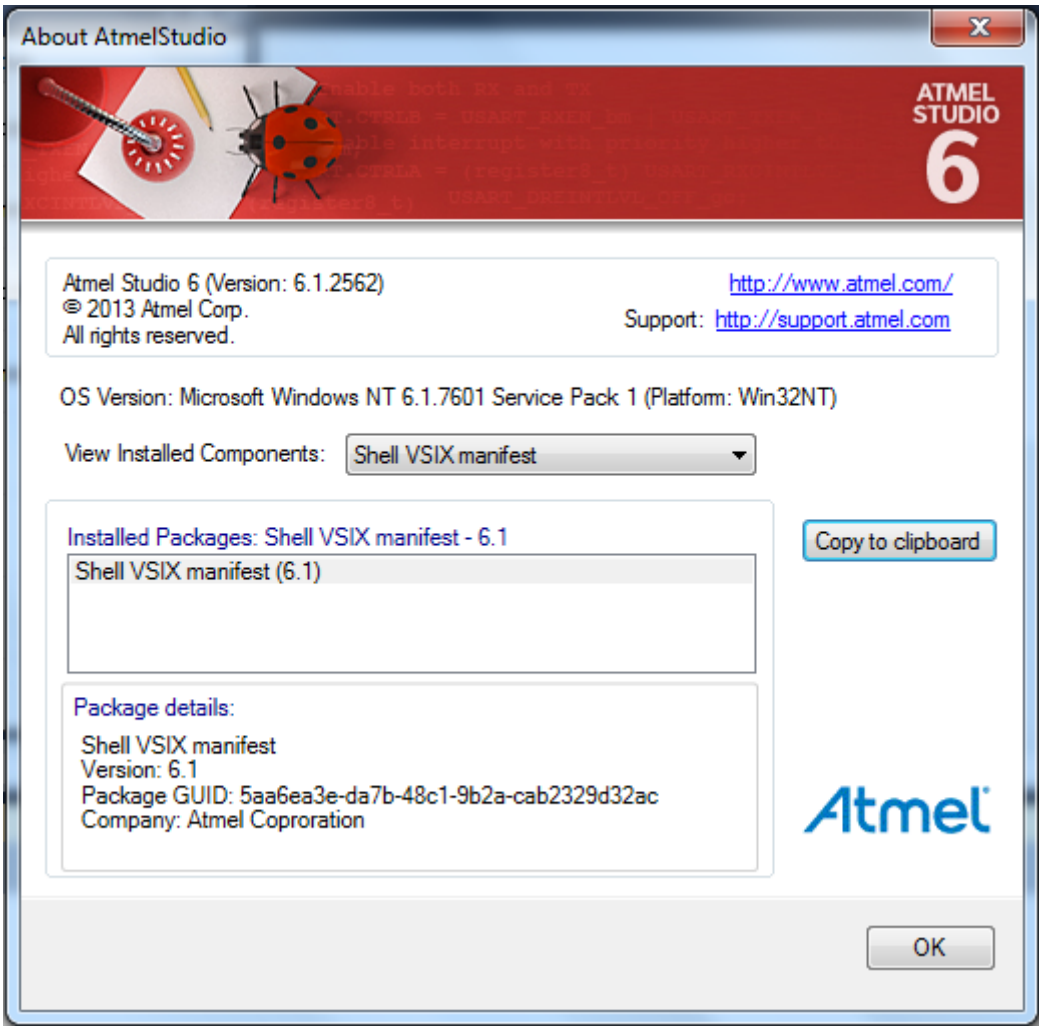
Nyomjunk még egyszer egy RESTET, hogy vissza térjünk a Jungo Driver-hez. Ha netán a Studio alatt véletlenül a LibUSB-s firmware-el szeretnénk emberkedni, nem történik semmi, csak hibajelentést kapunk. Tehát ne a hardware-ünkben keressük a hibát. Most pedig vegyünk elő valamilyen áramkört, amin tesztelhetjük az áramkörünk másik oldalát. Az eddigi visszajelzések szerint itt szokott problémás lenni az ügy. Az egyik Forum társunk majd egy hétig kereste a hibát, ami végül is egy hidegforrasztás volt. Persze, hogy a GTL-nél. Ezzel csak arra szeretném felhívni mindenki figyelmét, ha netán nem működne, hideg fejjel és módszeresen zárjunk ki mindent ami jó és ami marad, ott van a kutya elásva. Én egy próbapanelen felépített tranzisztor tesztert kötöttem rá, ami egyébként is a kísérletező kedvemnek esett áldozatul.



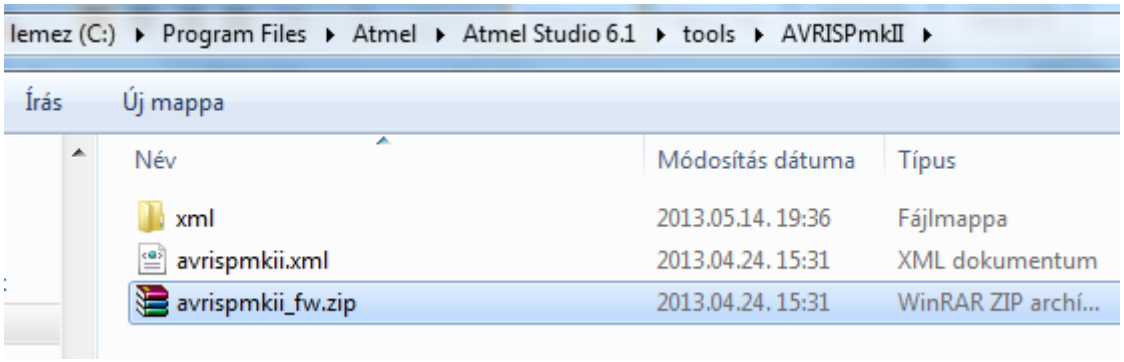
Aztán elballagunk az AVRStudioba, és ami most jön, az verziófüggő. Menü pont Tools -> Device programming. A felugró ablakban beállítjuk a programozónk típusát, a céláramkörön található AVR chip típusát, majd Apply.



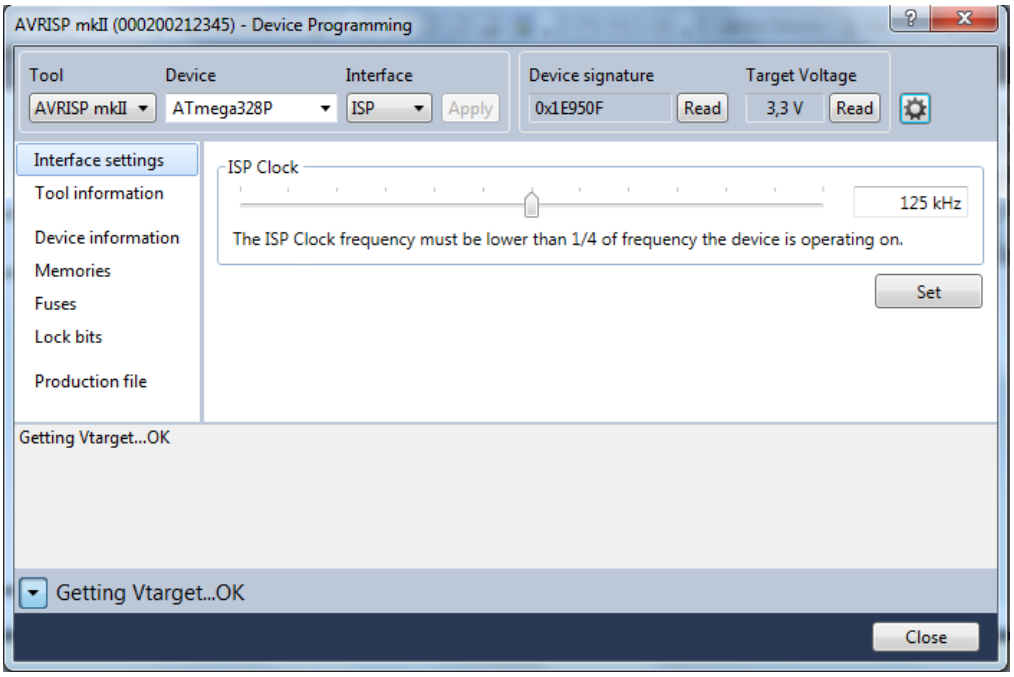
Na, ez konkrétan úgy hiányzott, mint hátunkra a púp. Frissíteni ugyanis innen nem tudunk. Ez az eredeti Atmel programozónak szól, ami teljesen más processzort használ. Persze ha vicces kedvünkben vagyunk, nyomhatunk egy Upgrade-t, amire elkezd méltatlankodni, amúgy nem történik semmi, de tovább sem enged. Ezt ezzel a verzióval csinálja:



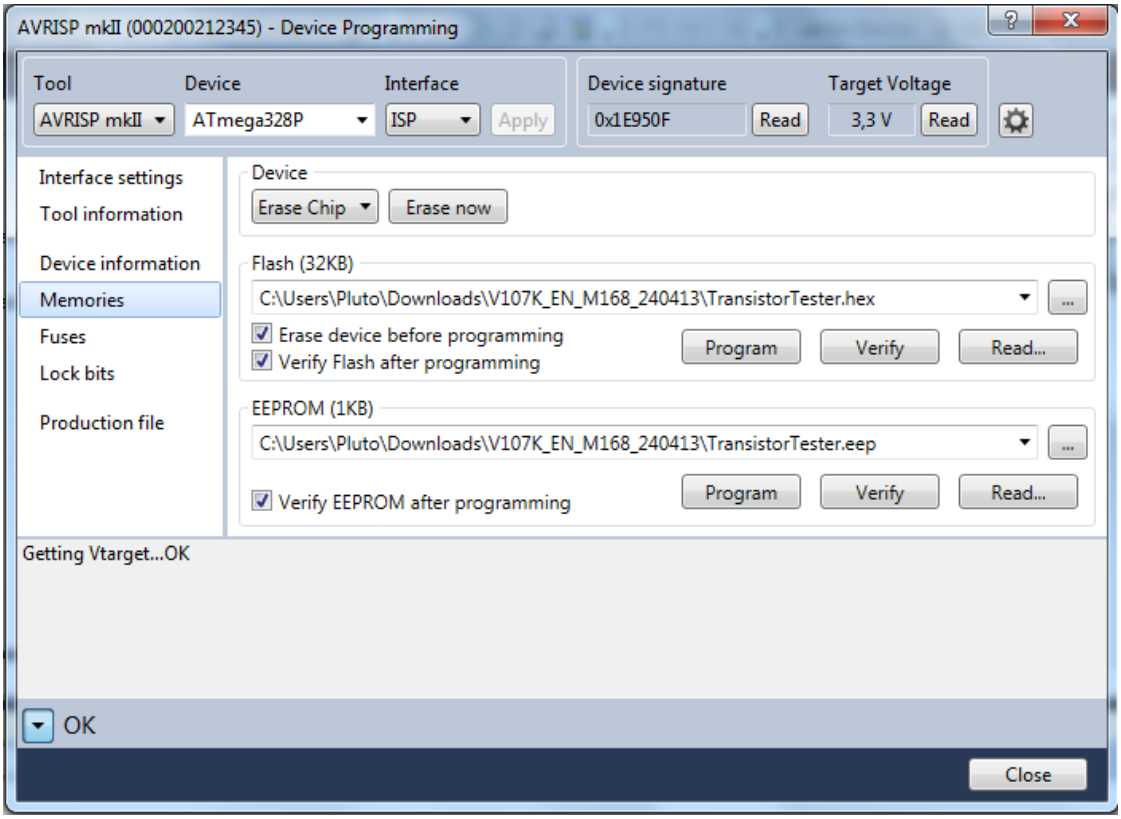
Ha régebbi verziótok van, azzal feltehetőleg megy. Ez valószínűleg azzal lehet összefüggésben, hogy a firmware 2013 márciusi verzió, az Atmel meg utána hozta ki ezt a software csomagot. Tehát le kell erről az upgrade mániáról szoktatnunk. Ezért először elballagunk Vezérlőpult -> Mappa beállításai, majd Nézet fül és kivesszük a pipát az „Ismert fájl típusok kiterjesztésének elrejtése” elöli négyzetből. Ezután ide megyünk:



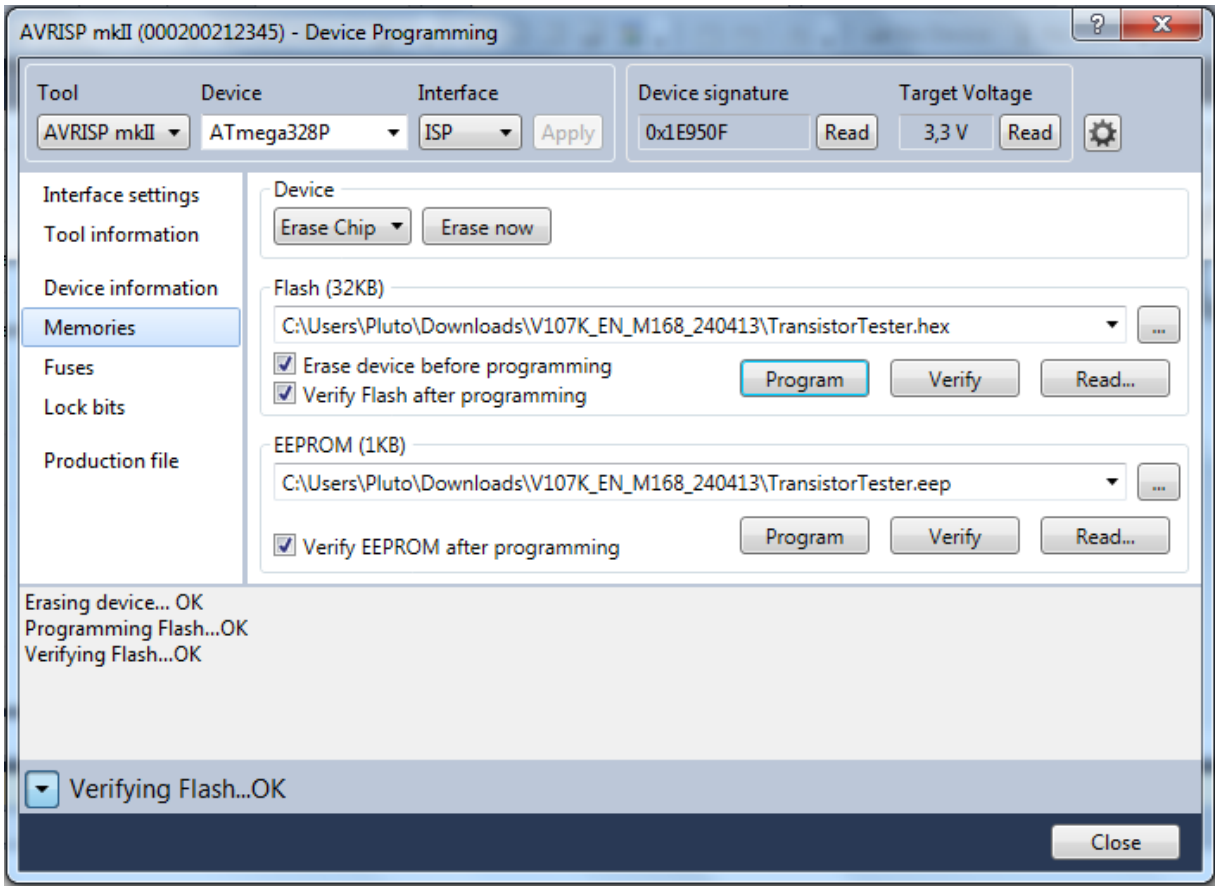
Majd átnevezzük pl.: avrispmkii.zip_ , hogy még véletlenül se találja meg. Ez ugyanis a gyári programozóba való firmware. Most vissza a Studio-ba és újbóli próbálkozás.



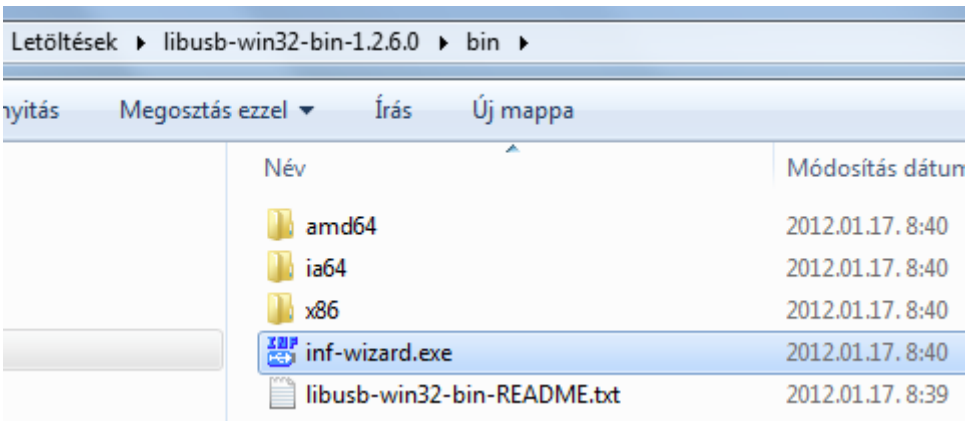
Ha ezt látjuk, jó a programozónk. A Device signature -> Read és Target Voltage -> Read gombot azért nyomjuk meg, hogy lássunk is valamit. A gyári programozó méri a céláramkör feszültségét. A miénk nem. A 3,3V csak azért van hogy a Studio ne izguljon. Mindig ennyi, legyen az a valóságban bármi is. Most viszont csináljunk is valamit.



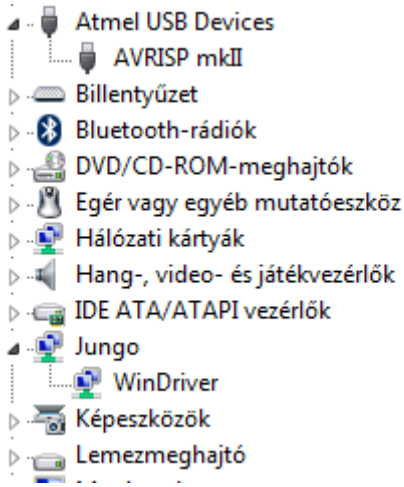
Kiválasztjuk a HEX-t amit be szeretnénk égetni a céláramkörbe, ha van EEPROM fájlunk akkor azt is és kattintunk a Program gombra.



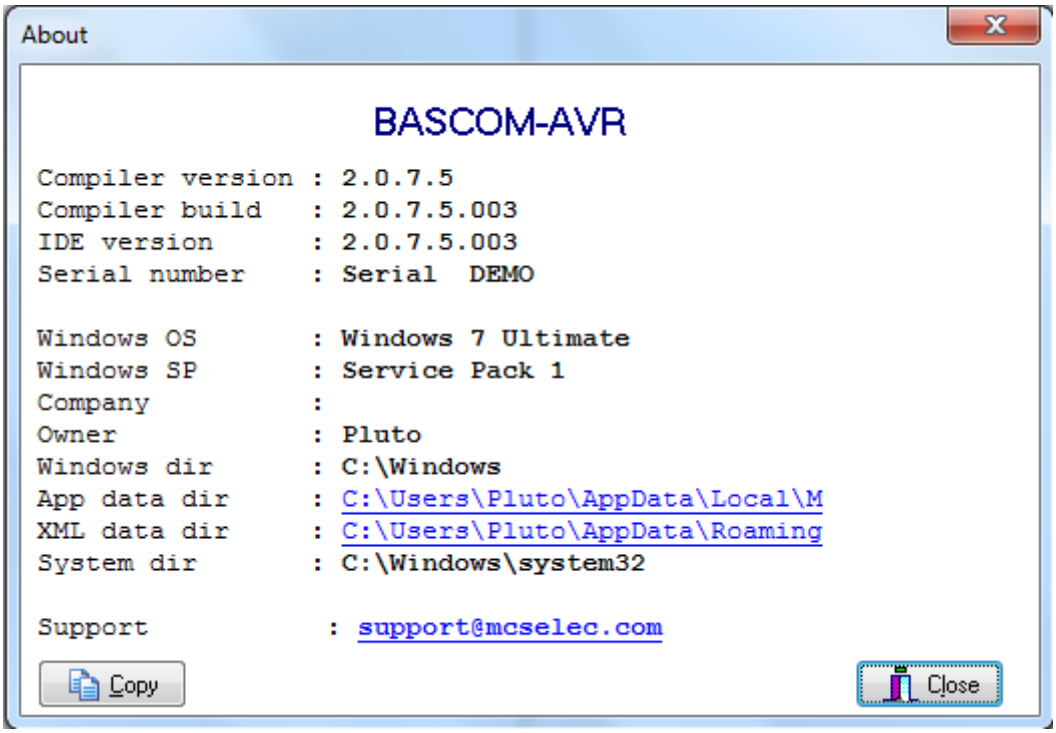
Ha ezt látjuk, hátradőlünk és megiszunk egy sört! Ugyan ez az EEPROM fájlal is, ha van. És ezzel készen is vagyunk a Jungo-Driver résszel. Itt végül is akár le is állhatnánk, mivel bármivel is készítünk HEX-t, ha a Studio nyitva van, egyszerűen be tudjuk égetni a segítségével. Mi viszont tovább megyünk. Mindenek előtt kell egy LibUSB driver. Innen tudjuk letölteni: http://sourceforge.net/projects/libusb-win32/?source=directory Ha megvan, kicsomagoljuk. A programozó mindenképpen legyen csatlakoztatva a PC-re és a RESET gomb megnyomásával váltsunk át a LibUSB-s firmware-re. Ellenőrizzük, hogy az átkapcsolás után valóban 4x villognak a LED-k. Ha ez megvan, felhívjuk ai inf-wizard.exe-t, és végigmegyünk a telepítésen.



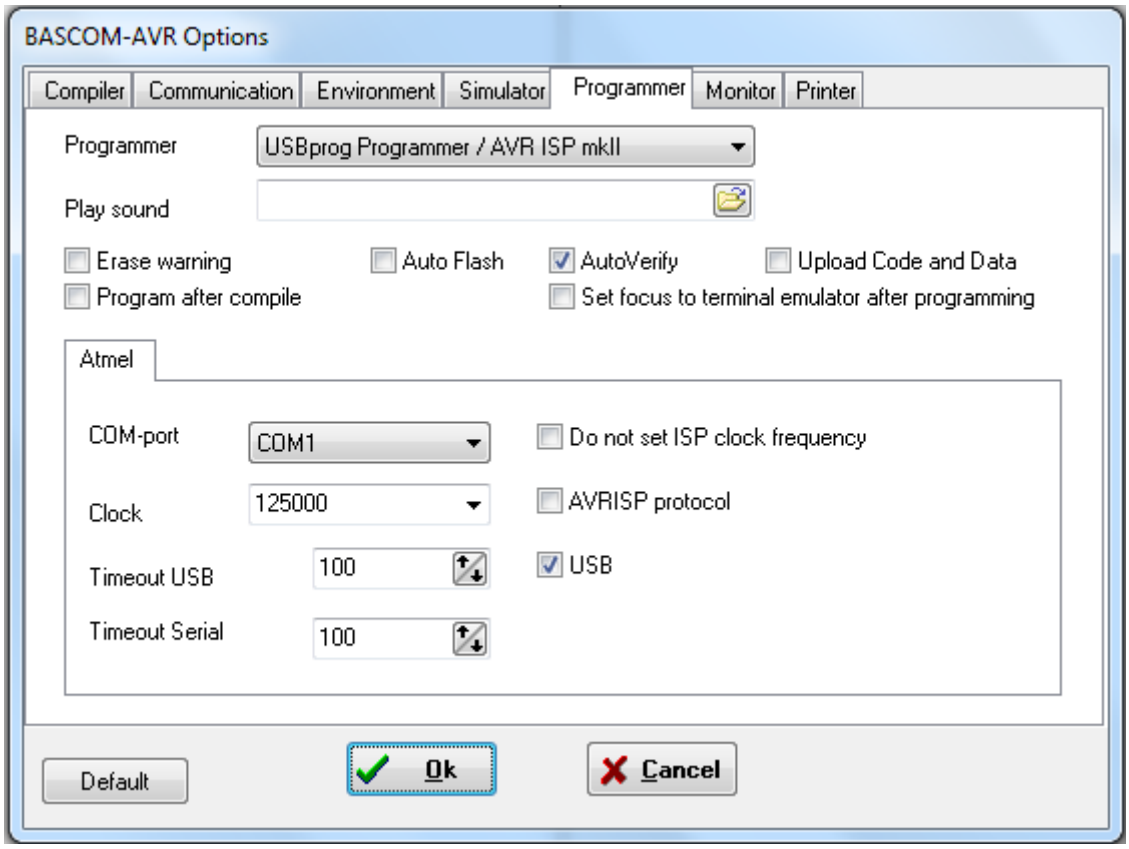
Ha megvagyunk, az eszközkezelőben ezt látjuk:



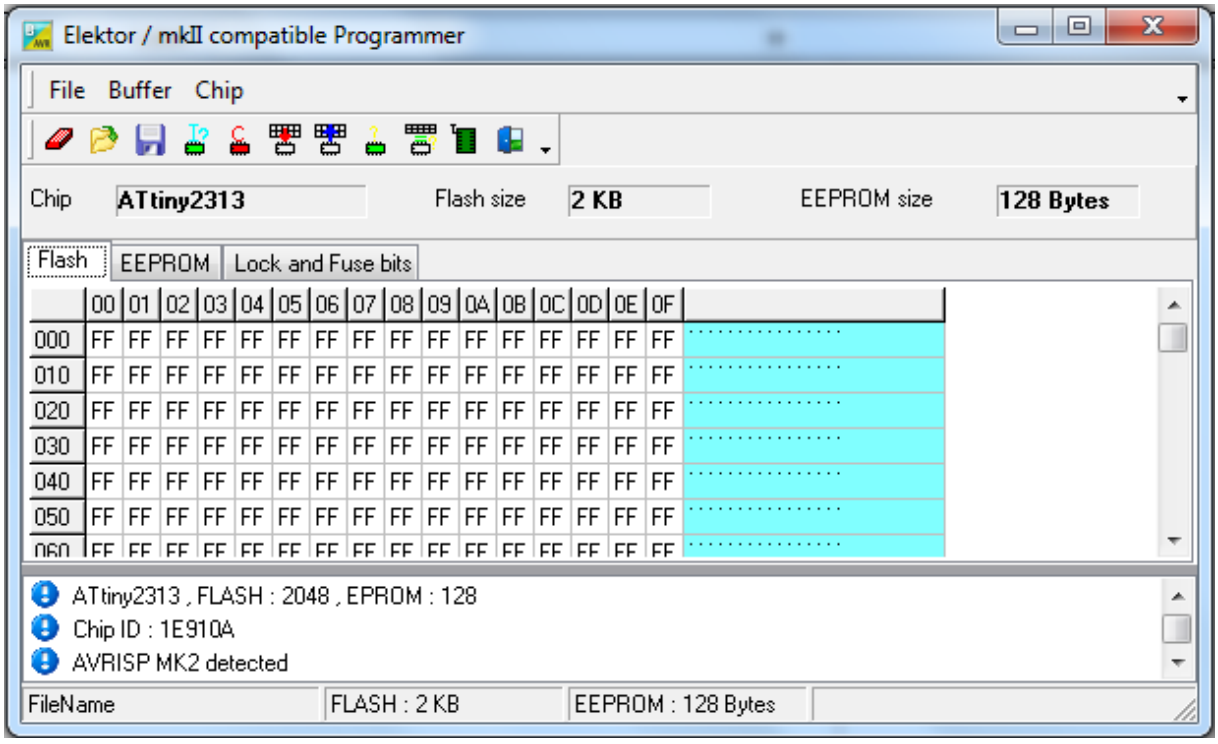
A programozónk „kijött” a Jungó Driver alól és megjelent az „Atmel USB Devices” alatt. Így már tudjuk használni LINUX és AVR Dude alatt is. És most ugrik a majom a vízbe, mert idáig a BASCOM nem mindegyik firmware verzióval volt hajlandó együttműködni. Most ez egy demo verzió:



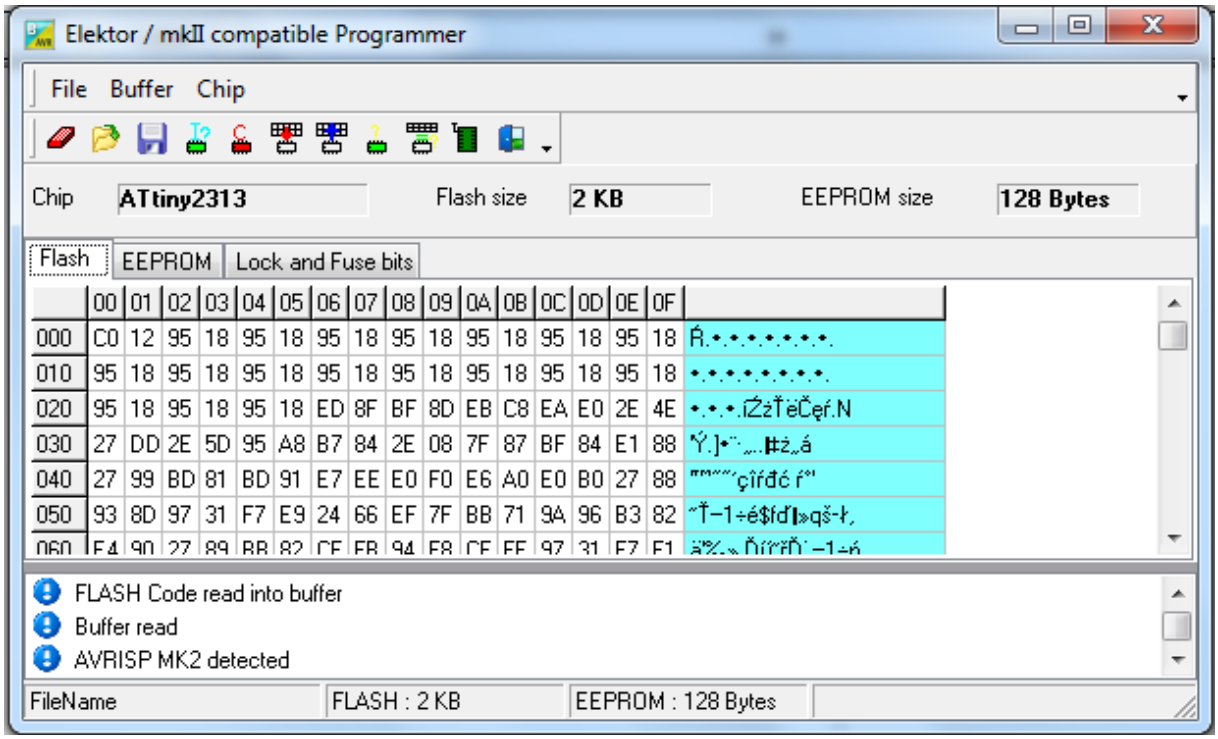
Tehát minden erre vonatkozik. Régebbi verziókkal nem teszteltem. Legelőször megnézzük a programozónkat az Options -> Programmer menüpont alatt.



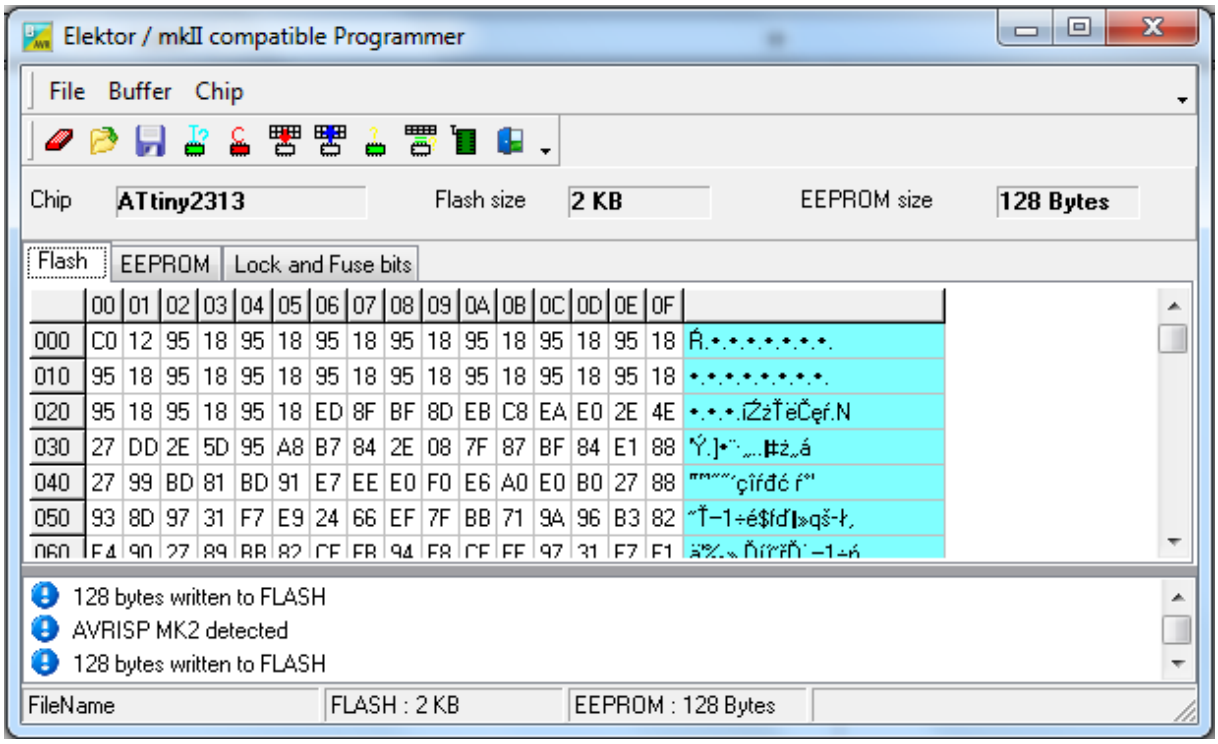
Ő az. Aztán elővettem egy dugaszpaneles 2313-t. Mégpedig azért, mert a demó verzióban, ha jól emlékszem 4kByte a határ. Nehogy az okozzon a végén problémát. Menüpont Program -> Send to Chip -> Manual és ismertessük fel vele a Chipet.



Na, ez teljes siker. Most kiolvassuk a Chip tartalmát, mert az ördög nem alszik.



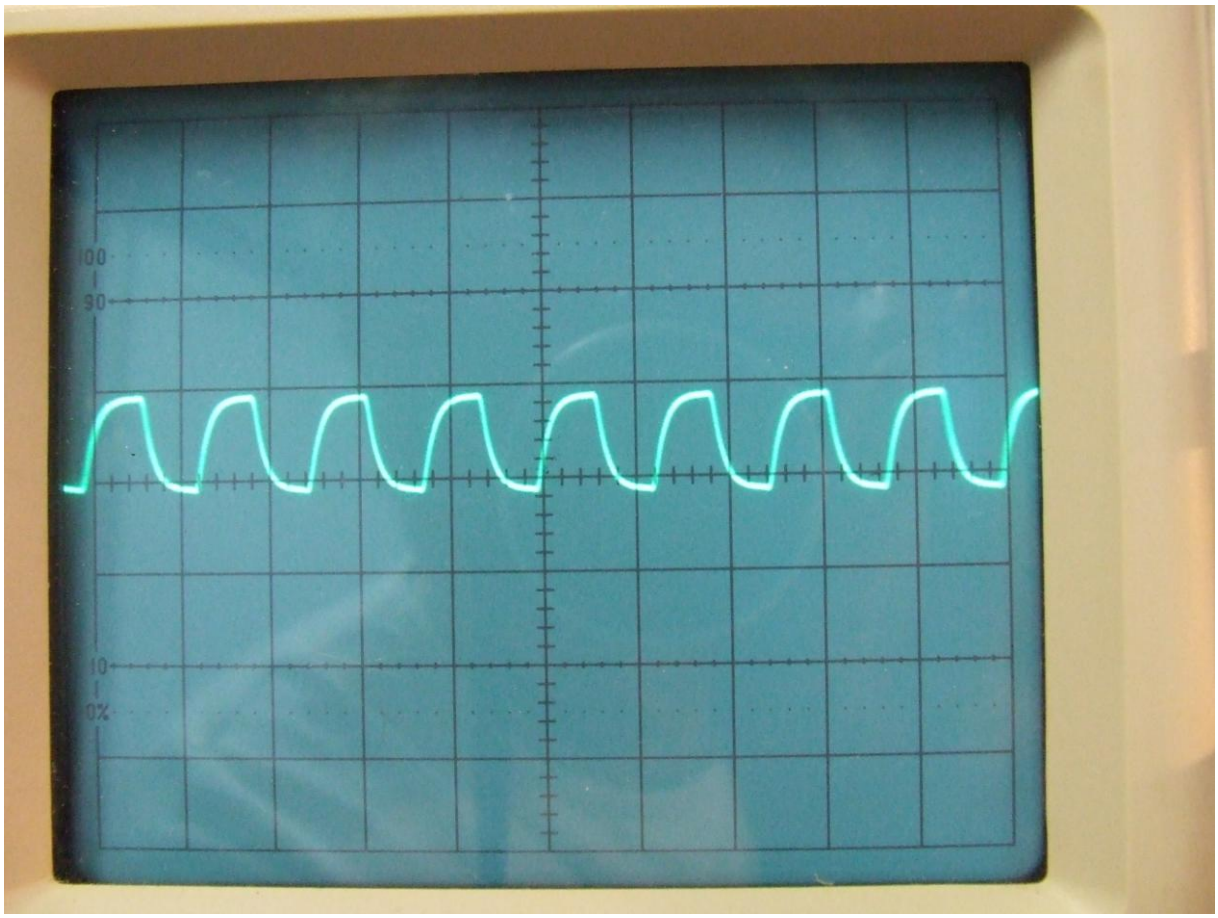
Ez is működik. Majd teljes törlés és újra írás.



És így kijelenthetjük, hogy működik BASCOM alatt is.

A végére hagytam a programozónak egy olyan tulajdonságát, ami már tényleg csak hab a tortán. Ez pedig a „RESCUE_CLOCK”. Ami annyit jelent, hogy a PC6(OC.1A/PCINT8) lábon állandó 4MHz található, arra az esetre, ha a céláramkörünkben tanyázó AVR biztosíték bitjeit sikerült úgy beállítani, hogy csak külső órajelre indul. Ilyenkor csatlakozunk az XTAL1 lábra és

ott adjuk neki a jelet. Viszont ügyeljünk arra, hogy ilyen esetben a programozó – vagyis SPI – órajel csak 125kHz legyen, mert csak így biztosítható a stabil működés. Az általam megépített programozón a mérőm 3,97MHz órajelet mutat. Így néz ki a szkópon:



A beállítás: 5V/div és 0.2µs/div. Azért arra vigyázzunk, hogy itt 5V játszik. Egy céláramkörben, ahol esetleg csak 3,3V vagy akár 2,7V található, ott esetleg zokon veszi az AVR. Ha saját magunk fordítjuk a firmware-t, a config.h-ban van egy lehetőség a 4MHz-s jelet átirányítani a PD5(XCK/PCINT12)-s lábra (#define XCK_RESCUE_CLOCK_ENABLE), ami annyit tesz, hogy az órajelet a szintillesztőn keresztül zavarjuk. Akkor viszont megjelenik a PDI és TPI csati lábain. Ezt mélyebben nem tanulmányoztam. Akinek van kedve, az kivesézheti.

Pluto, 2013-06-17