Tartalomjegyzék

PICkit 2 programozó és hibakereső működésének ismertetése

Jelen dokumentumot készítette: **Varga László**

Lektorálta: **Juhász Róbert**

1 A lektor megjegyzései

Az alkotás öröme mindig is megfogott. Ettől mindig fiatalnak érzi az ember magát, hiába múlnak el felette az évek. Különösen nagy örömöt jelent számomra, hogy a szerző egyik régi kedves tanítványom.

Amikor még belekezdtünk ebbe a projektbe, akkor még fogalmam sem volt róla, milyen nagy fába vágtuk a fejszénket (köszönhetően a Microchip mérnökeinek). Azonban mindezen nehézségeket legyűrve elkészült a mű. Azt kell mondjam, hogy nagyon jól sikerült, minden elismerésem Varga Lászlónak (Boci). Igazi precíz mérnöki munka volt ez a javából, nem túlzok talán, ha azt mondom, hogy sokkal jobb lett, mint az eredeti!

Remélem még sok ilyen remekmű fogja elhagyni kezedet, amihez ezúton is sok sikert kívánok:

Juhász Róbert

2 A szerző megjegyzései

Még 2009-ben döntöttünk úgy, hogy tervezünk egy a PICkit 2 feladatait ellátni képes áramkört THT[1] technológiával. A gyári kapcsoláson kisebb módosításokat eszközöltünk, ennek következtében aktualizálni kellett a Microchip Firmware programját is. A megtervezett áramkör utánépítésének megkönnyítése céljából szükségessé vált egy leírás elkészítése.

Mikor a dokumentációt elkezdtem írni úgy gondoltam, hogy röviden bemutatom az elkészült kapcsolást, néhány szót ejtve a módosításokról, a működésről. Az áramkör használatához azonban elengedhetetlen a programozó szoftver és a gyári PICkit 2 működésének ismerete. A dokumentáció többszöri átolvasás és tisztázás után kerül nyilvánosságra, azonban sajnos így sem redukálható nullára a szemantikai és stilisztikai hibák száma. Az esetleges problémákat a http://plc.mechatronika.hu weblapon üzemelő fórumon, a PICkit 2 topicban lehet jelezni.

A leírás elkészítése során elsősorban az érthetőségre és a szövegkohézióra törekedtem. Néhány kevésbé fontos részletre terjedelmi okok miatt nem tudtam kitérni. Feltételeztem, hogy az olvasó tisztában van az elektronika alapjaival. Véleményem szerint a következő ismeretanyag megszerzése elengedhetetlen a mikrovezérlőkkel való megismerkedés előtt:

- Analóg áramköri ismeretek
  - Áram, feszültség, teljesítmény, ellenállás, kapacitás, induktivitás fogalma, számítások
  - Karakterisztika fogalma, értelmezése
  - Ohm törvény alkalmazása a gyakorlatban
  - Kirchhoff törvények, feszültség- és áramosztás
  - Félvezetők (dióda, bipoláris tranzisztor, FET) működése, munkapont beállítás, alkalmazási példák erősítő és kapcsolóüzemben
  - BODE diagramok
  - Integrált áramkörök tulajdonságai, határértékek és paraméterek értelmezése
  - Műveleti erősítő és alapáramkörei
- Digitális technika alapjai
  - Digitális és diszkrét jel értelmezése
  - Számrendszerek (decimális, bináris, hexadecimális)
  - Logikai függvények
  - Kombinációs és sorrendi hálózatok vizsgálata, tulajdonságai
  - TTL, CMOS áramkörök
  - A legfontosabb digitális technikai elemek
    - ➔ Kapuk
    - ➔ Kódoló, Dekódoló, MUX, DEMUX, Aritmetikai áramkörök
    - ➔ Tárolók
    - ➔ Számlálók
    - ➔ Regiszterek
  - Átjárás az analóg és a digitális világ között (AD, DA átalakítás)

1 Through Hole Technology – furatszerelt technológia.

A szerző megjegyzései

- Számítástechnikai alapismeretek
  - A számítógép neumanni felépítése, alapelvek
  - Egycímes architektúra
  - Mikroprocesszor belső felépítése
  - A mikrovezérlő és a mikroprocesszor közötti különbség
  - Perifériák illesztése a mikrovezérlőhöz
  - Programtervezés, szoftverfejlesztés lépései
  - Programozás gépi kódban, az assembly nyelv megjelenése
  - Magas szintű programnyelvek szükségessége és korlátaik

A dokumentációban a kapcsolási rajzokon a magyar szabványos rajzjeleket használtam, a szakkifejezések ismertetésekor törekedtem az angol/magyar megfelelőjük szerepeltetésére is – minimum lábjegyzet szintjén. A mértékegységekre vonatkozó magyar helyesírási szabályt – miszerint a számérték és a mértékegység között szóköz használata kötelező – tudatosan szegtem meg. Úgy gondolom, hogy nem csak nem esztétikus, de egyenesen félrevezető ez a fajta írásmód, ezért az angol szabály szerint a mértékegységek szóköz nélkül követik számértékeiket.

A leírás – Abonyi Zsoltnak (okl. villamosmérnök) köszönhetően – javításra került. Itt szeretnénk köszönetet mondani neki, hogy a hibák feltárásával segítette munkánkat. Többek között módosítottuk a *Target csatlakozófelület* menüpontot, valamint a klónáramkör kapcsolási rajzát és nyáktervét.

3 Bevezetés

A PICkit 2 a Microchip cég olcsó és megbízható programozásra és hibakeresésre is alkalmas – a PC oldal felé USB interfésszel ellátott – eszköze. Ismeri a legtöbb microchip flash mikrovezérlőt, valamint soros EEPROM memóriát (a részletes listát megtaláljuk a PICkit 2 v2.xx[2] programozó szoftver *Help* menüjének *ReadMe* pontjában – vagy MPLAB IDE fejlesztőkörnyezet esetén a *C:\Program Files\Microchip\MPLAB IDE\Readmes* alapértelmezett útvonalon[3]).

Az eredeti PICkit 2 firmware-e és kapcsolási rajza nyilvános, ezért sokan utánépítették az áramkört. Kis kutatómunka után az interneten számos verziót találunk – amelyek funkciójukban és az ebből adódó bonyolultságukban lényeges eltéréseket mutathatnak. Amennyiben úgy döntünk hogy egy ilyen klónverziót építünk meg az eredeti termék megvásárlása helyett mindig figyelmesen olvassuk el a hozzá tartozó dokumentációt. Lehetséges változtatások:

- Az eszköz nem működik együtt 3,3 voltos rendszerekkel[4].
- A Programmer – To – Go funkció nem használható[5].
- PGD, PGC lábak leválasztása programozóként történő használat során[6].
- Egyes alkatrészek más típusokkal történő helyettesítése[7].

A klónáramkör kiválasztása során ügyeljünk arra, hogy működő és nekünk megfelelő verziót építsünk meg. Nézzünk utána az adott honlapnak, a készítőnek különböző információs csatornákon (pl. ismerősök véleménye, hiteles fórumok). A megépítéshez ha lehet olyan áramkört válasszunk amihez részletes leírás, esetleg bemérési, élesztési útmutató tartozik.

A mellékletben egy a Juhász Róbert honlapján megtalálható PICkit 2 áramkört ismertetünk.

2 A program legfrissebb verziója ingyenesen letölthető a Microchip honlapjáról.
3 A PICkit 2 szoftvere, és az MPLAB beépített pluginja nem feltétlenül ugyanazokat a mikrovezérlőket ismeri.
4 Kihagyható a műveleti erősítővel felépített 5V to 3V3 áteresztő tranzisztoros feszültségszabályozó áramkör.
5 A Programmer – To – Go funkció (6.3.3.1. fejezet) elhagyása egyszerűsíti az áramkört, és csökkenti a költségeket (nincs szükség a két 24LC512 soros EEPROM memóriára).
6 Háromállapotú bufferek, vagy analóg kapcsolók használatával elérhetjük ezt a funkciót, de természetesen ez többletalkatrészt jelent.
7 Az egyes klónok tervezői gyakran módosítják az eredeti alkatrészlistát amihez általában célszerűségi okok vezetnek (pl. bipoláris tranzisztor helyett FET használata így csökkentve az ellenállások számát, valamint a fogyasztást, vagy egyes az angolszászok körében közkedvelt áramköri elemek helyett hazánkban is kapható azokkal egyenértékű alkatrész használata).

4 A jegyzetben használt fogalmak tisztázása

A dokumentációban használt kifejezések megértéséhez a legtöbb esetben elegendő az elektronikával hobbi szinten foglalkozók általános tudása. Ebben a fejezetben az esetleges félreértések elkerülése végett az általam használt fontosabb szakkifejezések[8] hátterét kívánom megvilágítani.

Hardver: A számítógép fizikai része – az áramköri egységek, vezetékek összességét tekintjük hardvernek. A PICkit 2 esetében a hardvert a szerelt (beforrasztott) NYÁK a szükséges csatlakozóval, és kábellel ellátva jelenti.

Szoftver: „A számítógép azon része, amelyért a hardvert rugdossuk.” Komolyra fordítva: szoftvernek tekintjük a számítógép memóriájában tárolt programot – és a hozzá tartozó adatot –, amely segítségével a hardvert különböző feladatok ellátására tudjuk felhasználni. A PICkit 2 esetén szoftver alatt a PC-re megírt PICkit 2 v2.xx, vagy MPLAB IDE programot értjük.

Firmware[9]: Egy speciális szoftvertípus, amely nélkül a hardveregység az alapvető feladatait se tudja ellátni. Ilyen a PC BIOS-a, vagy a DVD írókban lévő – EEPROM-ban tárolt – program. PICkit 2 esetén a PIC18F2550 típusú mikrovezérlőben tárolt programot tekintjük firmware-nek.

Klónáramkör: Egy már meglévő gyári alkalmazás funkcióinak megvalósítása. Egy klón megtervezésének általában anyagi okai vannak. A klón áramkörnek nem kell feltétlenül ugyanolyan felépítéssel rendelkeznie, mint az eredeti gyári alkalmazás, valamint egyes alfunkciókban is eltérhet – azonban az alapvető működésben meg kell hogy egyezzen vele.

Host, Target: Emuláció, hibakeresés és programozás során az emulátort/debuggert/programozót HOST, míg a céláramkört TARGET néven illetjük.

8 A jegyzetben a legtöbb helyen megpróbálom közérthető ún. konyhanyelven (is) megfogalmazni a tartalmat. Itt olyan definíciókat találhat a kedves olvasó, amelyek a leírás felépítésébe nem illeszkedtek.

9 Az adott szó még nem honosodott meg a hazai szakmai életben, ezért a jegyzetben az angolos írásmódot választjuk. A magyar helyesírás szabályai szerint alkalmazható még a kiejtéshelyes förmver, ill. létezik egy öszvér megoldás is: firmver.

A jegyzetben használt fogalmak tisztázása

V_{DD}, V_{SS}: Integrált áramkörök esetén a pozitív tápfeszültségre V_{DD}, vagy V_{CC}, míg a negatív tápfeszültségre[10] V_{SS}, vagy V_{EE} szimbólumokkal hivatkozunk. A D a drain, a C a kollektor, az S a source, míg az E az emitter elektródára utal. Általánosan igaz, hogy TTL és ECL[11] áramkörök esetén V_{CC}, V_{EE}, míg CMOS áramköröknél V_{DD} és V_{SS} jelölésekkel találkozunk, azonban néhány dokumentáció eltérhet ettől a szabálytól.

Programmer/Debugger: A PICkit 2 képes egy programozó és egy debugger funkcióját is ellátni. Programozás során a céláramkörbe töltjük, onnan kiolvassuk, és/vagy összehasonlítjuk a kívánt programot. Hibakeresés[12] során egy speciális ún. debug programot égetünk a mikrovezérlőbe, ez lehetővé teszi, hogy a program futását kedvünk szerint manipulálhassuk (futtatás, megállítás, léptetés, programmemória, és regiszterek kiolvasása). A PICkit 2 debugger funkciója alacsonyabb rendű, mint egy emulátor. A hibakereső funkcióhoz szükségünk van a céláramkör erőforrásaira (RB_6, RB_7, $\overline{MCLR}$, oszcillátor, stb.), valamint néhány megkötéssel is meg kell barátkoznunk: pl. korlátozott számú, és komplexitású töréspontok, WDT nem használható, stb[13].

Égető: Az első programozott áramkörökben (PROM[14]) még szó szerint égetéssel tudtuk kialakítani a nekünk szükséges bitmintát. Diódákat – később tranzisztorok átmeneteit – égette el az üzemi áramnál jóval nagyobb értékű árammal a programozó az egyik logikai állapothoz – ezzel szakadást létrehozva az adott helyen[15]. Az első programozókat így égetőknek nevezték el, és ez a kifejezés – bár a FLASH technológia már más elven alapul – a mai napig megmaradt a hőskor iránti tiszteletből.

MCU: Microcontroller (mikrokontroller), mikrovezérlő, µC. Egytokos mikroszámítógép, amely egy chipben egyesíti a számítógép részeit (CPU, memória, I/O egység).

10 PIC mikrovezérlő esetén a negatív tápfeszültség a GND, azonban léteznek olyan integrált áramkörök, amelyek kettős tápfeszültségről üzemelnek, és ezeknek 3 tápbemenetük lehet (V_{DD}, GND, V_{SS}).

11 Emittercsatolt logika (Emitter Coupling Logic) a mai leggyorsabb logikai áramkörcsalád – bipoláris tranzisztorokból épül fel.

12 A hibakeresést debugolásnak nevezzük. Az angol bug (bogár) kifejezést használjuk minden rejtélyes hibaforrásra az informatikában.. 1947-ben az egyik hőskori számítógép a MARK II meghibásodott, és a hiba forrása egy molylepke volt. A mérnökök belső köreiben már ezen esemény előtt is bugnak hívták a hibákat, azonban a történet után a kifejezés széles körben népszerűvé vált. Ma már kevesen vannak akik ne ismernék a bug-hiba, debug-hibakeresés összefüggést. A történet egyébként nem csak egy szép mese, szegény molylepke be is került a mérési jegyzőkönybe.

13 Részletesebben l. 7.1.2.1. fejezetben.

14 Programmabble Read Only Memory – a felhasználó által egyszer felprogramozható csak olvasható memória.

15 A diódák eredetileg rövidzárat képviseltek a mátrix szerkezetben.

5 PICkit 2 hardver bemutatása

5.1 Az áramkör használata

5.1.1. ábra: PICkit 2 programozó és hibakereső

Az eszközön (5.1.1. ábra) található 3 állapotjelző LED, egy nyomógomb, egy ICSP csatlakozófelület és egy USB csatlakozó[16].

A LED-ek funkciója:

Power: A zöld színű LED világít, ha az áramkör kap tápellátást az USB porton keresztül[17].

Target: A sárga színű LED jelzi, hogy a céláramkör számára a tápellátást a PICkit 2 biztosítja.

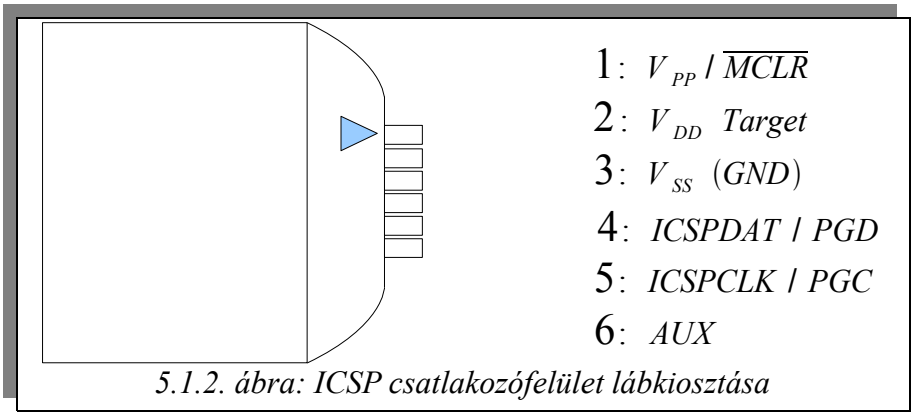
Busy: A piros színű LED programozás során jelzi a foglalt állapotot.

16 A PICkit2 fel van szerelve egy műanyag füllel, amellyel a mechanikai rögzítését végezhetjük el (pl. felfűzhetjük egy kulcstartóra).

17 A LED csupán a feszültség meglétét (>2V) és helyes polaritását jelzi, annak pontos értékét csak mérés útján tudnánk meghatározni.

A nyomógomb funkciója: Amennyiben engedélyezve van a *Programmer → Write on PICkit button* funkció a gomb megnyomásának hatására a megnyitott *.hex fájl beégetésre kerül[18]. A nyomógombnak a firmware-frissítés (6.14. fejezet), és a Programmer-To-Go funkció (6.3.3.1. fejezet) során is fontos szerepe van.

ICSP csatlakozófelület: A Microchip egy 6 pólusú hüvelysort[19] használ az áramkörben történő soros égetés biztosításához. A 5.1.1. és a 5.1.2. ábrán lévő háromszög jelzi az 1-es lábat. Az ICSP-ről az 7.2. fejezetben találunk részletesebb információt.



5.1.2. ábra: ICSP csatlakozófelület lábkiosztása

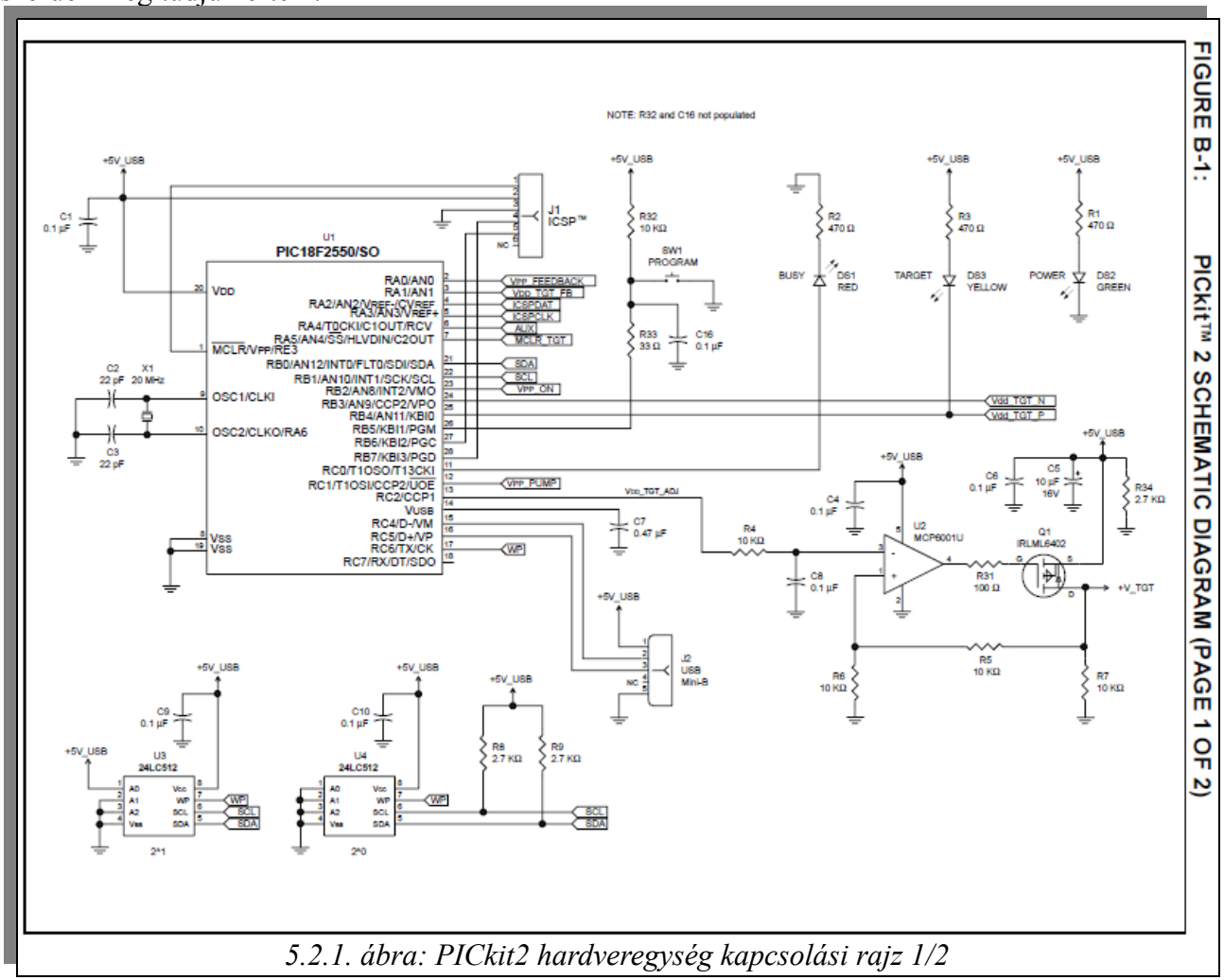
Az egyes céláramkörök esetén mindig fordítsunk kiemelt figyelmet a csatlakozó bekötésére, mert az eltérő lábkiosztás mind a programozó, mind a céláramkör tönkremeneteléhez vezethet.

18 Ez a lehetőség igen kényelmes lehet ugyanazon hex fájl több PIC-be történő beégetése esetén – hiszen nem kell a PC-n lévő szoftverrel foglalkoznunk, csupán a nyomógombot „kezelnünk”.

19 Lábtávolság: 2,54mm. Csatlakoztatható: 0,6 mm négyzetes keresztmetszetű tüske.

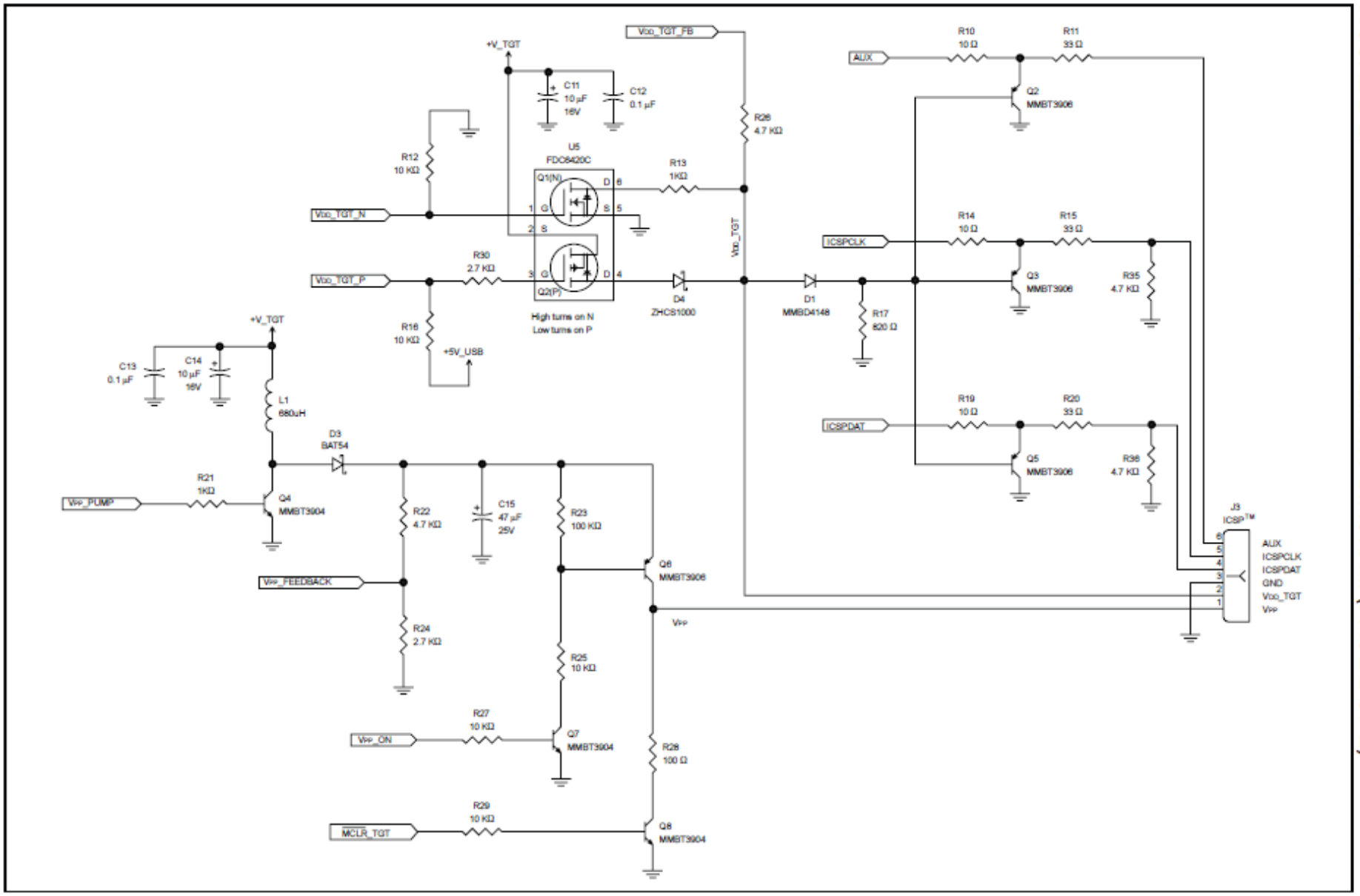
5.2 Az áramkör működése

A Microchip cég az áramkör kapcsolási rajzát és a működtető program gépi kódú, valamint C nyelvű forrását nyilvánosságra hozta[20], így a működést viszonylag egyszerűen meg tudjuk érteni.



5.2.1. ábra: PICkit2 hardveregység kapcsolási rajz 1/2

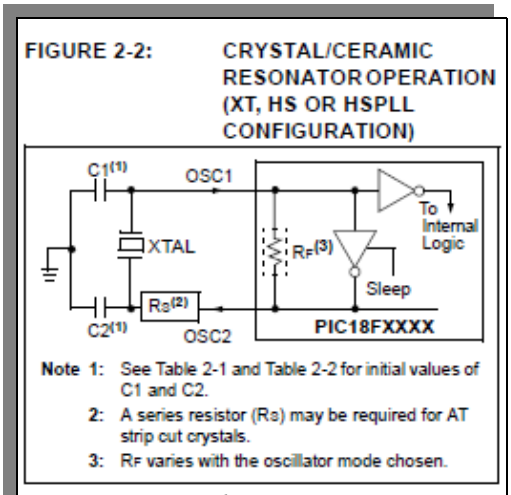
20 PICkit2 User Guide



5.2.2. ábra: PICkit2 kapcsolási rajz 2/2

5.2.1 Az áramkör felbontása tipikus részekre

5.2.1.1 Oszcillátor konfiguráció



5.2.1.1.1. ábra: PIC18F2550 mikrovezérlő kristály és kerámia oszcillátor konfigurációja

A PIC mikrovezérlők tartalmaznak egy belső oszcillátor áramkört (5.2.1.1.1. ábra), amelyhez kívülről elég ha a kvarckristályt és a hozzá tartozó hidegítőkondenzátorokat csatlakoztatjuk. Az R_S soros ellenállás opcionális – az AT strip cut[21] kiképzésű oszcillátor esetén javasolt. Részletes információért keresd fel az alábbi weboldalakat: http://www.icmfg.com/glossary.html

http://www.icmfg.com/crystalfaqs.html#q6

http://www.foxonline.com/techdata.htm

Bár a PIC18F2550 típusú mikrovezérlő tartalmaz belső RC oszcillátort is, a fejlesztők úgy döntöttek[22], hogy az órajelet egy külső kvarckristállyal biztosítják.

21 A strip cut és a cut verzió méretben, formában, és számos elektronikai paraméterben eltérést mutat. A strip cut – általában, azonos frekvencián – nagyobb soros ellenállással, szélesebb frekvenciaválasztékkal rendelkezik, szélesebb hőmérséklet tartományban működik, azonban kisebb a kristály hangolhatósága, nagyobb a mérete (tokozása).

22 Többek között az USB kommunikáció és a pontos órajel szükségessége miatt.

TABLE 2-2: CAPACITOR SELECTION FOR CRYSTAL OSCILLATOR

Osc Type	Crystal Freq	Typical Capacitor Values Tested:	
		C1	C2
XT	4 MHz	27 pF	27 pF
HS	4 MHz	27 pF	27 pF
	8 MHz	22 pF	22 pF
	20 MHz	15 pF	15 pF

Capacitor values are for design guidance only.

These capacitors were tested with the crystals listed below for basic start-up and operation. **These values are not optimized.**

Different capacitor values may be required to produce acceptable oscillator operation. The user should test the performance of the oscillator over the expected VDD and temperature range for the application.

See the notes following this table for additional information.

Crystals Used:
4 MHz
8 MHz
20 MHz

5.2.1.1.2. ábra: Kondenzátorértékek az alkalmazható oszcillátorfrekvenciák függvényében

A PIC mikrovezérlők adatlapja egy táblázatban (5.2.1.1.2. ábra) megadja, hogy a támogatott frekvenciákhoz milyen értékű hidegítőkondenzátor alkalmazása célszerű. A Microchip a PICkit2 áramkörénél a megadott 15pF helyett 22pF-ot választott – a kapacitás növelésével nő az eszköz stabilitása, azonban ezzel együtt a feléledési idő is. Az oszcillátort hidegítő kondenzátor kiválasztásánál nem csak az eszköz adatlapját kell figyelembe vennünk. Általánosan elfogadható, hogy ha a feléledési idő nem kritikus, akkor a stabilitás érdekében nagyobb értékű kapacitást használunk – azonban a konkrét értéket mindig a tesztek során tapasztalt működés alapján határozzuk meg.

5.2.1.2 Host[23] ICSP csatlakozófelület

A J_1-el jelölt ICSP csatlakozó a felhasználó számára nem hozzáférhető[24]. Az eszköz első felprogramozását[25] teszi lehetővé, amelyet a Microchip gyárilag elvégez.

5.2.1.3 Host USB csatlakozófelület

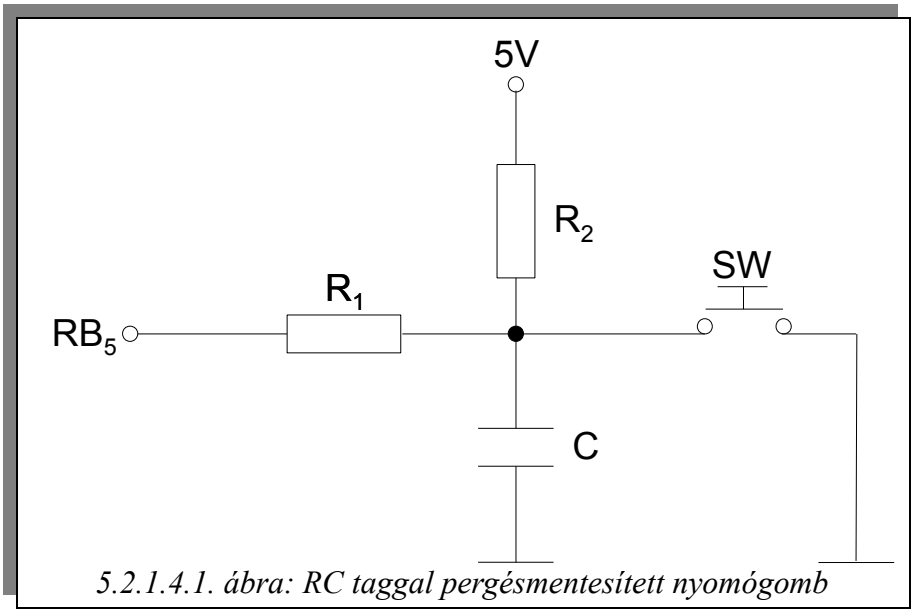
A PC-vel történő kommunikáció szabványos USB porton keresztül került biztosításra. A PICkit2 támogatja az USB 2.0 szabványban rögzített full-speed adatátvitelt. Az USB csatlakozó – mechanikai kialakítása miatt – rosszul nem köthető be, ezért védelem nem került kiépítésre.

23 A PICkit 2-vel történő munkánk során alapvetően két mikrovezérlő együttműködéséről beszélhetünk. A HOST jelzésű PIC18F2550 típusú mikrovezérlő foglal helyet a PICkit 2 -ben, míg a TARGET jelzésű mikrovezérlő a céláramkörben.

24 Ezért a csatlakozón védelmek nincsenek kiépítve.

25 A későbbi firmware frissítés (6.14. fejezet) már az USB porton keresztül, a PC oldali szoftver segítségével történik.

5.2.1.4 Nyomógomb illesztése a mikrovezérlőhöz



5.2.1.4.1. ábra: RC taggal pergésmentesített nyomógomb

A nyomógomb egy pergésmentesítő áramkörön keresztül került kiépítésre (5.2.1.5.1. ábra). Az RB_5 lábra a T időnél gyorsabban érkező impulzusok nem jutnak el. T-t feszültségosztás alapján tudjuk számolni: SW aktiválásánál RB_5 lábon kisebbnek kell lennie a potenciálnak 0,8V-nál[26]:

$$U_L = U_T \cdot \frac{X_C}{X_C + R_2} = U_T \cdot \frac{\frac{1}{2\Pi f C}}{\frac{1}{2\Pi f C} + R_2} = \left(\frac{2\Pi f C}{2\Pi f C}\right)\ U_T \cdot \frac{1}{1 + 2\Pi f RC} \quad [27]$$

$$\frac{U_L}{U_T} = \frac{1}{1 + 2\Pi f RC}$$

$$\frac{U_T}{U_L} = 1 + 2\Pi f RC$$

$$f = \frac{\frac{U_T}{U_L} - 1}{2\Pi RC} = \frac{\frac{5\text{V}}{0{,}8\text{ V}} - 1}{6{,}28 \cdot 10\text{k}\Omega \cdot 10\,\mu\text{F}} = \frac{5250}{628} = 8{,}36\text{ Hz} \rightarrow \boldsymbol{T = 120\text{ ms}} \quad [28]$$

Mindemellett a nyomógombot egy szoftveres prellmentesítő rutin[29] segítségével kérdezi le a PICkit2.

---

26 Katalógus adat. A nyomógomb elengedésekor U_H magas szintet kellene megvizsgálnunk (2V) ez azonban nagyobb frekvenciát, így kisebb periódusidőt eredményezne.

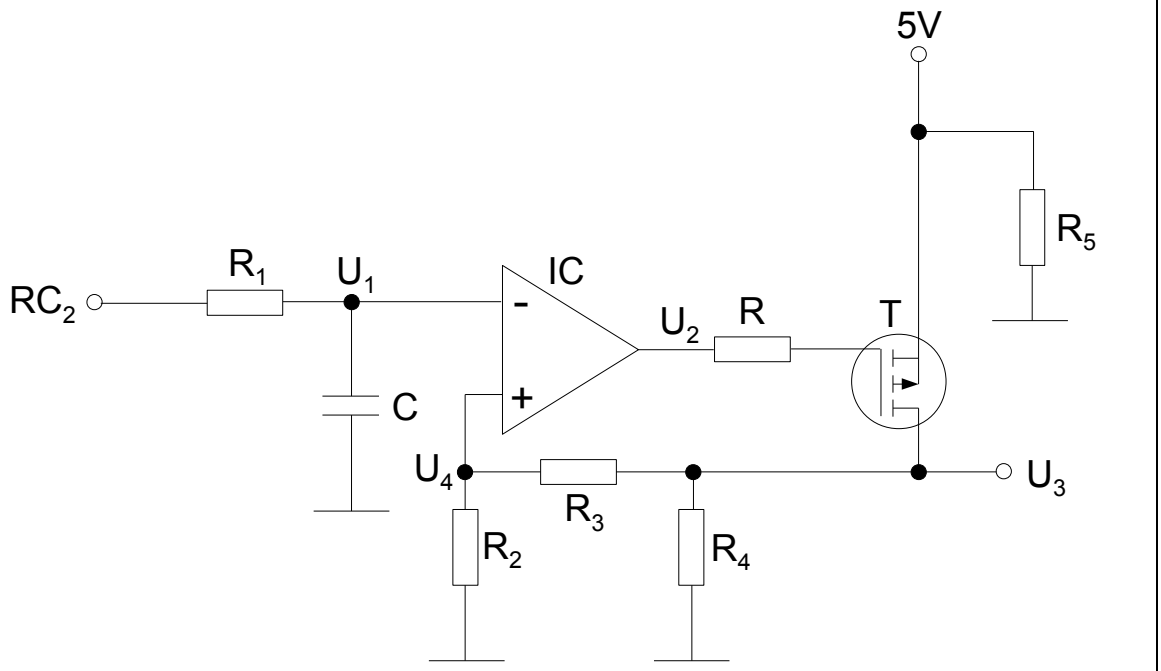
27 U_T-vel a tápfeszültséget (5V) U_L-el a TTL alacsony szint maximális értékét (0,8V) jelöltük.

28 A gyakorlatban, méretezési eljárásnál adott a frekvencia, amely feletti jeleket már nem kívánunk figyelembe venni. Ebben az esetben olyan ellenállás-kondenzátor párt választunk, amelyek esetében igaz, hogy a kondenzátor reaktanciája az adott frekvencián már jóval (1/10) kisebb, mint az ellenállás értéke. Az R_2 ellenállás felhúzó szerepet is betölt, ezért értéke nem tetszőleges (1kΩ-100kΩ).

29 200 ms.

5.2.1.5 Változtatható target VDD feszültség előállítása

A PICkit 2 képes együttműködni a hagyományos 5V-os, és az újabb 3V3-os rendszerekkel is. A megfelelő működés biztosítására az eszközt kiegészítették egy szabályozható tápfeszültség-előállító részáramkörrel.



5.2.1.5.1. ábra: Szabályozható VDD tápfeszültség előállító áramkör

Az RC_2 lábon egy belső periféria[30] segítségével PWM jelet állít elő a mikrovezérlő, amelyből egy elsőfokú aluláteresztő szűrő[31] segítségével létrejön az U_1 ponton jelzett – a kitöltési tényezővel arányos amplitúdójú – DC feszültség. Az IC-vel jelölt műveleti erősítő a kimenetét képes a tápfeszültség szintjéig (5V) emelni[32]. Alaphelyzetben az U_1 pontot kötnénk a műveleti erősítő neminvertáló bemenetére, míg a visszacsatolást az invertáló bemenetre vezetnénk, a negatív visszacsatolás. biztosítására. A kimenet pedig közvetlenül felhasználható szabályozott feszültségként. Ez a kimenet azonban erősen korlátozott áram tekintetében. A T jelzésű booster tranzisztor invertáló működése[33] miatt a visszacsatoló ág a neminvertálóra, a bemenet pedig az invertáló ágba kerül (5.2.1.6.1. ábra). A műveleti erősítővel soros feszültség visszacsatolást alkalmaztunk, a tranzisztor áteresztőként[34] erősítő üzemben (3,3V) és túlvezérelt (telített) állapotban (5V) működik. R ellenállás nagyfrekvenciás üzemben korlátozná[35] a C_{GS} kapacitás okozta nagy I_G-t.

30 Capture-Compare-PWM modul

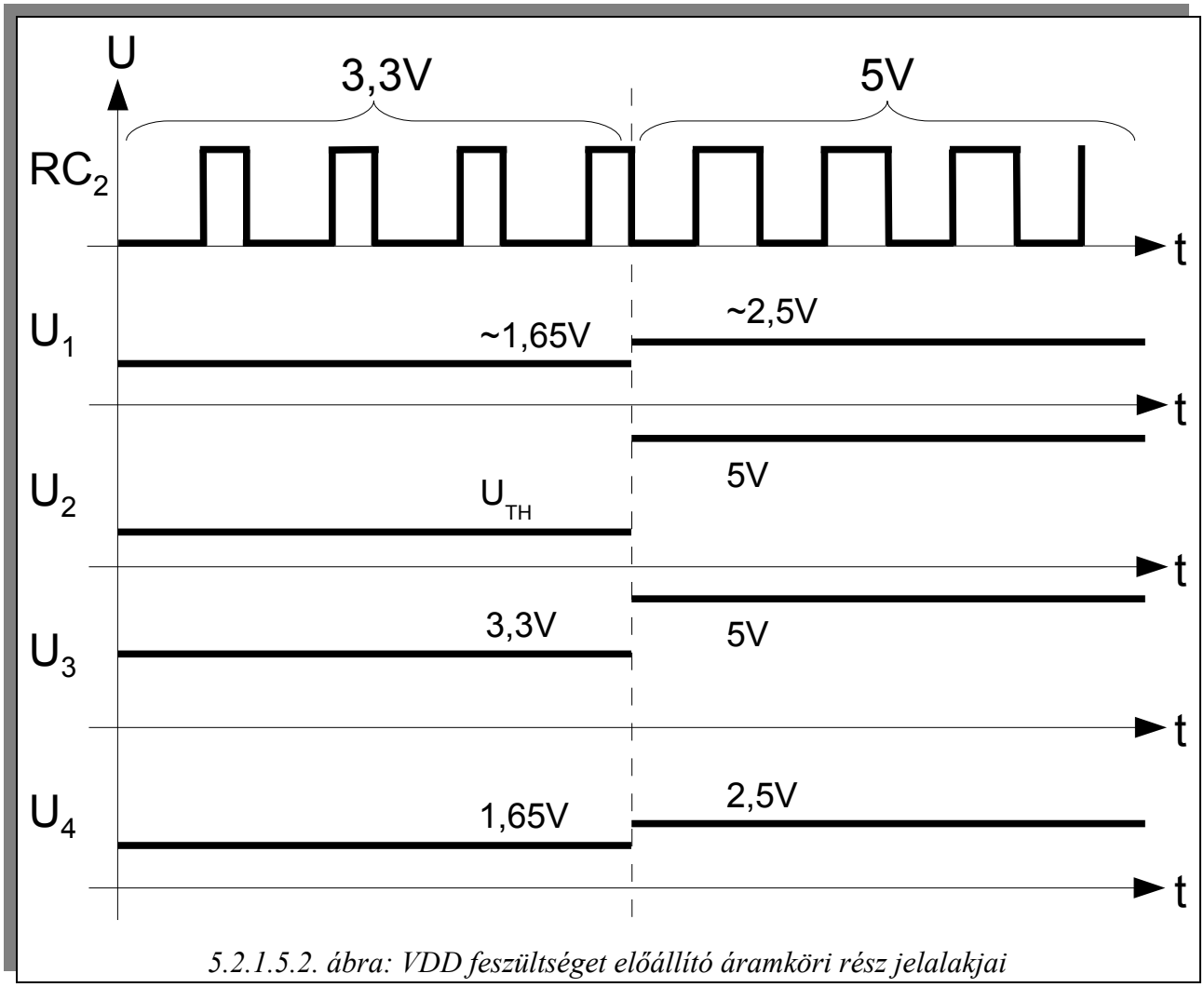
31 A szűrő itt az átlagoló szerepét tölti be.

32 Csak CMOS technológiával készült ún. Rail-to-Rail erősítő használható

33 P típusú FET-et kell alkalmaznunk, mert csak a tápfeszültséghez képest tudunk vezérelni. A kimeneten *nagyobb feszültség* előállításához a FET-nek kisebb ellenállást kell képviselni, vagyis jobban kell nyitnia – ehhez a műveleti erősítő kimenetén *kisebb potenciál* előállítása szükséges, és ebből a folyamatból adódik az invertálás.

34 Felfogható egy változtatható ellenállásként, ami pl. 3V3 táp mellett olyan értékű ellenállást képez, ami olyan feszültségosztót alkot a céláramkör okozta terheléssel, hogy utóbbin 3,3V essen függetlenül annak értékétől.

35 A T jelzésű FET a PWM jel kitöltési tényezőjével arányos DC feszültséget kap a vezérlő (Gate) lábán, ezért az R ellenállásnak csupán elvi jelentősége van.



5.2.1.5.2. ábra: VDD feszültséget előállító áramköri rész jelalakjai

Az áramkör működését érdemes az U_3 ponttól visszafelé követni. Erre a pontra kell biztosítani egyik esetben 3,3V, másik esetben 5V feszültséget. Az U_4 ponton lévő feszültség (1,65V; 2,5V) az U_3 pontról kerül visszacsatolásra az R_3, R_2 által alkotott 1:1 feszültségosztón keresztül.

A műveleti erősítő – működéséből adódóan – az U_1 ponton is ugyanekkora feszültségre törekszik. Az RC_2 lábon előállított PWM jelet egy aluláteresztő RC[36] szűri meg – így lesz az U_1 ponton a kitöltési tényezővel arányos feszültség (33% → 1,65V; 50% → 2,5V).

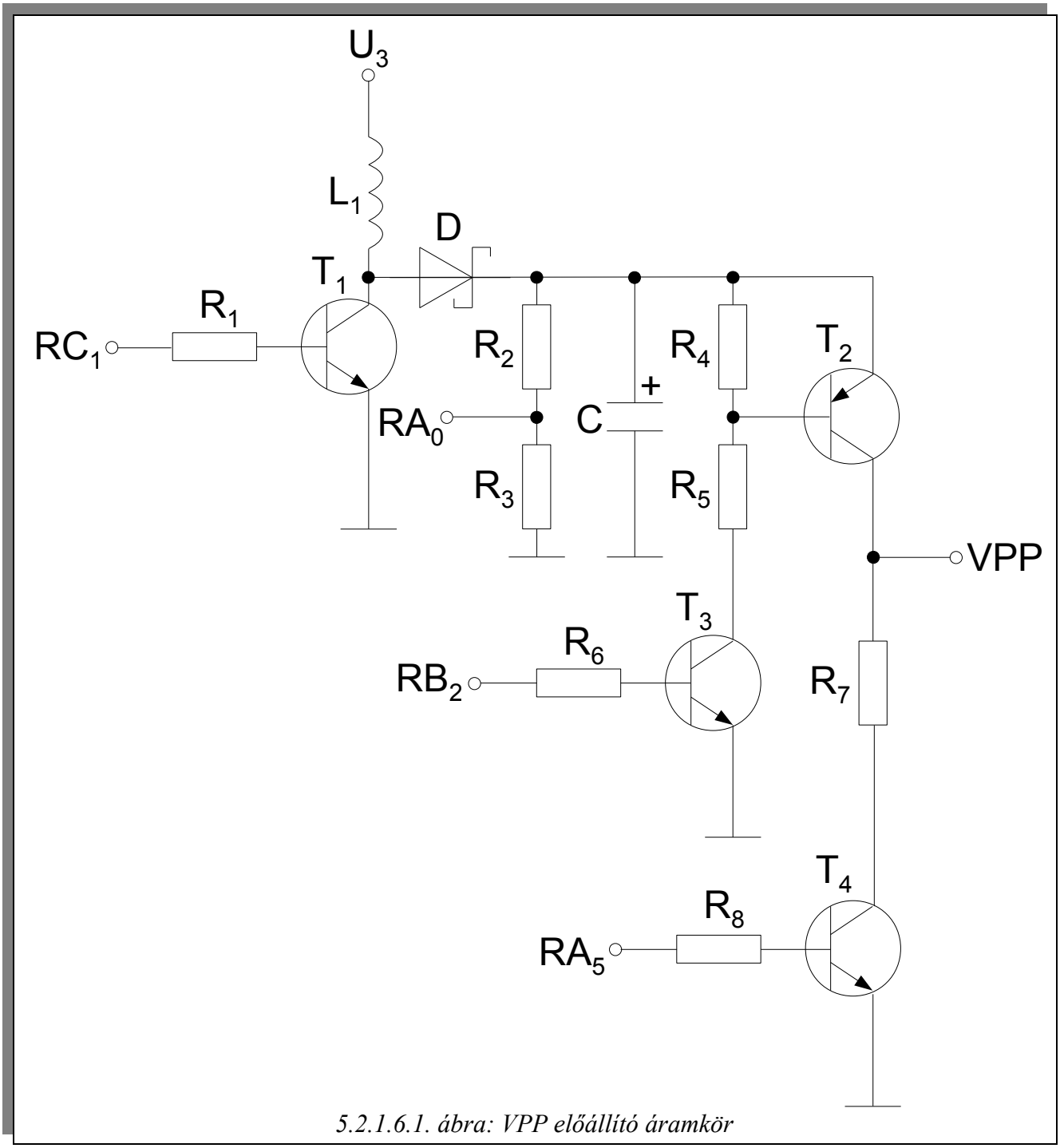
A műveleti erősítő kimenetének feszültsége a T jelzésű tranzisztor Threshold potenciáljának és az U_1 pont feszültségének függvénye (1,65V → U_2~U_{TH} ; 2,5V → U_2~5V).

Itt jegyeznénk meg, hogy mivel a FET a visszacsatolókörben van, ezért káros tulajdonságai – zaj, a Threshold feszültség szórása, hőmérsékletfüggés, stb. – nem befolyásolják a kimeneti feszültséget.

36 Itt átlagolóként működik.

5.2.1.6 VPP programozófeszültség előállítása

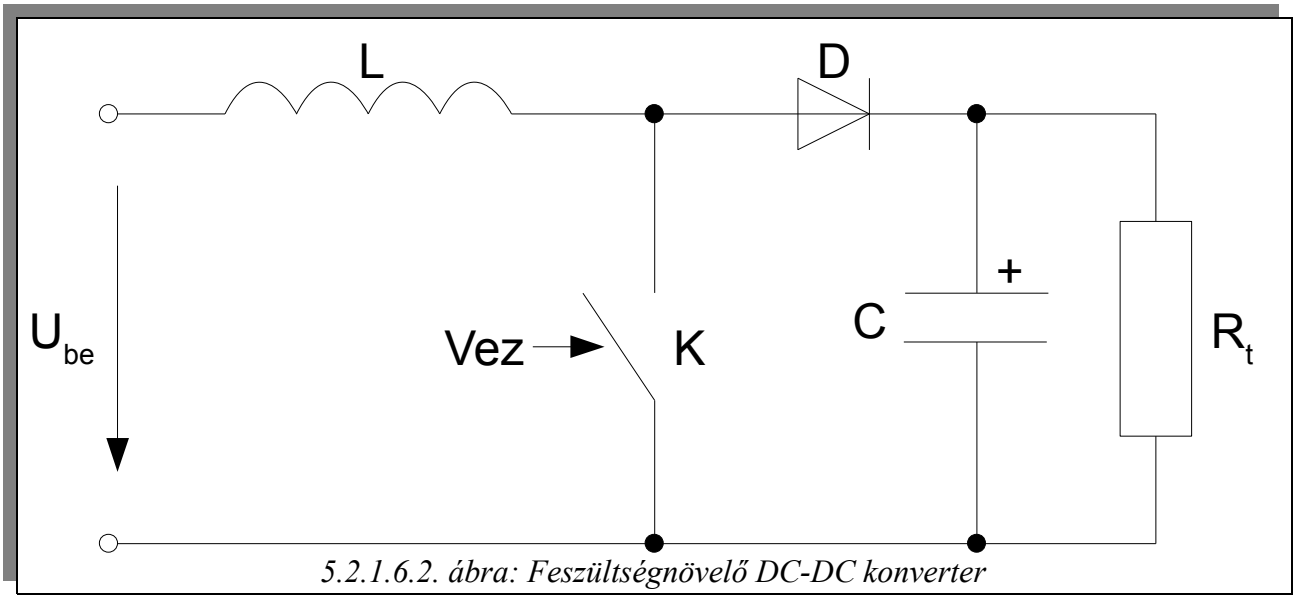
A programozófeszültség 5V-os rendszerek esetén 13V, 5V míg 3,3V-os rendszer esetén 3,3V. A PICkit 2 tartalmaz egy diszkrét elemekből felépített PWM vezérlésű Step Up[37] típusú DC-DC feszültségkonvertert.



5.2.1.6.1. ábra: VPP előállító áramkör

37 Feszültségnövelő.

Az áramkör működésének megértéséhez először vizsgáljuk meg a 5.2.1.6.2. ábrán látható elvi elrendezést.



5.2.1.6.2. ábra: Feszültségnövelő DC-DC konverter

Mind az U_{be}, mind az R_t-n eső U_{ki} feszültség DC. A K kapcsolót vezérlő feszültség kitöltési tényezőjével tudjuk beállítani a kimeneti feszültséget. A K kapcsoló nyitott állapotában a bemeneti feszültség – leszámítva a diódán eső részét – a terhelésre jut[38] és közben tölti a pufferkondenzátort. A K kapcsoló zárt állása esetén a bemeneti feszültség az L tekercset tölti. A D dióda meggátolja, hogy a kimeneti feszültséget földeljük, ebben az állapotban a C jelzésű kondenzátor látja el energiával a terhelést. A K kapcsoló nyitásakor az L tekercsben tárolt energia a bemeneti feszültséget erősítve jelenik meg a kimeneten. Mivel a tekercsben indukált feszültség és a bemeneti feszültség összeadódik[39], ezért a kimeneten a bemeneti feszültségnél nagyobb értékű feszültség keletkezik. Minél tovább töltjük a tekercset, ill. minél nagyobb az induktivitás értéke, a bemenethez hozzáadott indukált feszültség annál nagyobb lesz[40]. Az áramkörrel célszerűen szabályozott feszültséget szoktunk előállítani, vagyis a kimeneti feszültséget, vagy annak arányos részét visszacsatoljuk, és ennek megfelelően állítjuk be a kitöltési tényezőt. A PICkit2 áramkör ezt a visszacsatolást szoftveres úton valósítja meg, egy feszültségosztó és a belső A/D átalakító segítségével méri az előállított feszültséget (RA_0 láb), és ennek függvényében változtatja a CCP[41] modul kitöltési tényezőjét állító regiszter értékét.

Visszatérve a 5.2.1.6.1. ábrához belátható, hogy a K kapcsolóként ebben az elrendezésben a T_1 tranzisztor üzemel, L → L_1, D → D, C → C jelzéssel megtalálható. Az RB_2-es lábat a programozó üzemelteti, amennyiben ezen a lábon +5V van, a T_3 jelzésű tranzisztor a kollektor és emitter pontját rövidre zárja, földre kötve R_5 ellenállást. Ebben az esetben a T_2 tranzisztor az R_4 ellenállással párhuzamosan kapcsolt R_{BE} ellenállását úgy módosítja, hogy ki tudjon nyitni, vagyis rajtuk 0,6V essen (R_4=100kΩ, R_5=10kΩ).

38 A tekercs DC feszültségre rövidzárként viselkedik, a D dióda nyitóirányban van előfeszítve.

39 Lenz törvénye szerint a tekercsben indukált feszültség az őt keltő hatással (töltőáram) ellentétes lesz.

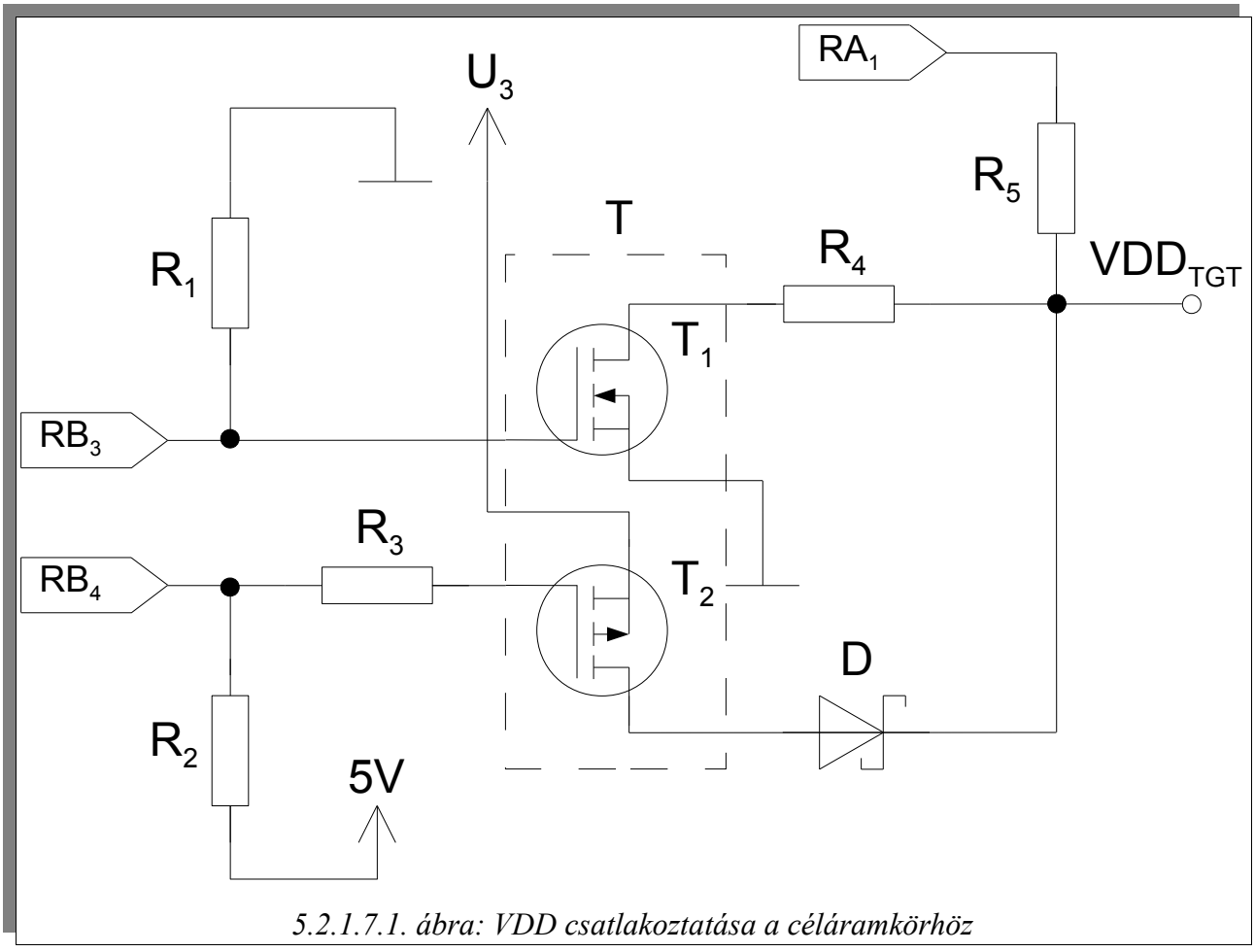
40 Természetesen a tekercs töltésének idejére a kondenzátornak kell árammal ellátnia az áramkört, ezért ezt az időszakot nem tudjuk minden határon túl növelni.

41 Capture/Compare/PWM – kiolvasás/összehasonlítás/impulzus szélesség moduláció. Mi a modult PWM jel előállítására alkalmazzuk.

T_2 tranzisztor kinyitásával (kollektor-emitter rövidzár) az előállított feszültség a V_{PP} pontra kerül. Ha a programozó az RB_5 lábon 0 potenciált hoz létre, akkor a T_3 tranzisztor kollektora és emittere között nagy ellenállást (szakadást) hoz létre, ezáltal meggátolja T_2 tranzisztor kinyitását, hiszen a bázisa az R_4 ellenálláson keresztül ugyanakkora potenciálra van felhúzva, mint az emittere. Az RA_5 lábat a felhasználó üzemelteti. Amennyiben 5V-ot adunk a T_4 tranzisztor bázisára az R_8 ellenálláson keresztül, akkor az kinyit, és a V_{PP} pontot R_7 ellenálláson (100Ω) keresztül a földre húzza ($\overline{MCLR}$ On[42] állapot). Ellenkező esetben – RA_5 lábon +5V – a T_4 tranzisztor szakadásként viselkedik, és elengedi a V_{PP} pontot ($\overline{MCLR}$ Off[43]).

5.2.1.7 VDD feszültség csatlakoztatása

A 5.2.1.5.1. ábrán már bemutattuk, hogy a PICkit2 hogyan állítja elő a Target számára a VDD feszültséget (U_3). Ezt a potenciált le kell tudnunk választani a céláramkörről, hiszen lehet, hogy saját táppal rendelkezik. Az ehhez szükséges áramköri részt a Microchip mérnökei a 5.2.1.7.1. ábra szerinti elrendezésben oldották meg.



5.2.1.7.1. ábra: VDD csatlakoztatása a céláramkörhöz

42 $\overline{MCLR}$ asserted
43 $\overline{MCLR}$ released

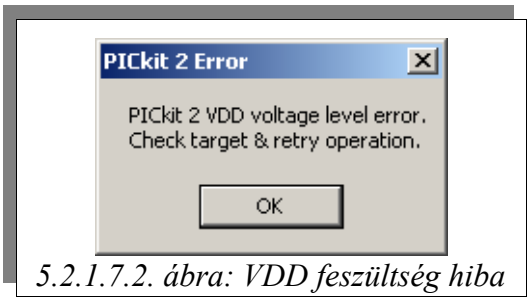
Az U_3 pontot a T_2 jelzésű P-FET kapcsolja a céláramkörre. A tranzisztor vezérlését az RB_4-es láb segítségével a mikrovezérlő végzi el a szoftveres beállítás figyelembevételével[44]. Az RA_1-es lábat analóg bemenetként konfigurálták a firmware-ben, ezért a belső A/D átalakító segítségével közvetlenül tudjuk mérni a VDD_{TGT} pont feszültségét[45].

Tegyük fel, hogy a céláramkör tápellátását a PICkit2 biztosítja. Amennyiben a mért feszültség kisebb, mint a beállított U_3-U_D-U_{max}[46], akkor a PICkit2 a VDD lábon rövidzárat jelez (5.2.1.7.2. ábra). A VDD_{TGT} pont közvetlenül csatlakozik a target-hez, ezért a D jelzésű Schottky[47] dióda gondoskodik a céláramkörtől esetlegesen érkező nagyfeszültség leválasztásáról.

Elképzelhető az az eset, hogy a beállítás szerint a PICkit 2-nek kellene ellátnia a céláramkört tápfeszültséggel, de az ennek ellenére rendelkezik saját energiaforrással. Ekkor a D jelzésű dióda megakadályozza a tápok összeakadását – az ilyenkor meginduló nagy értékű áramot a PICkit2 szoftvere érzékeli, és leállítja a tápellátást.

Abban az esetben, ha az eszköz rendelkezik saját tápforrással az RA_1 ponton tudjuk mérni ennek értékét. Ebben az üzemmódban a T_1 jelzésű FET az R_4 ellenálláson keresztül megadja a földpotenciált a VDD_{TGT} pontra. A GND-re azért van szükség, mert ha a céláramkör tápellátása mégse kielégítő – vagyis a VDD_{TGT} pontra nem kapcsol feszültséget – akkor az A/D átalakító segítségével 0V-ot tudunk mérni[48].

R_1, R_2 ellenállások a tranzisztorok – vezérlés nélküli – alaphelyzetbe (szakadás) állítására szolgálnak.

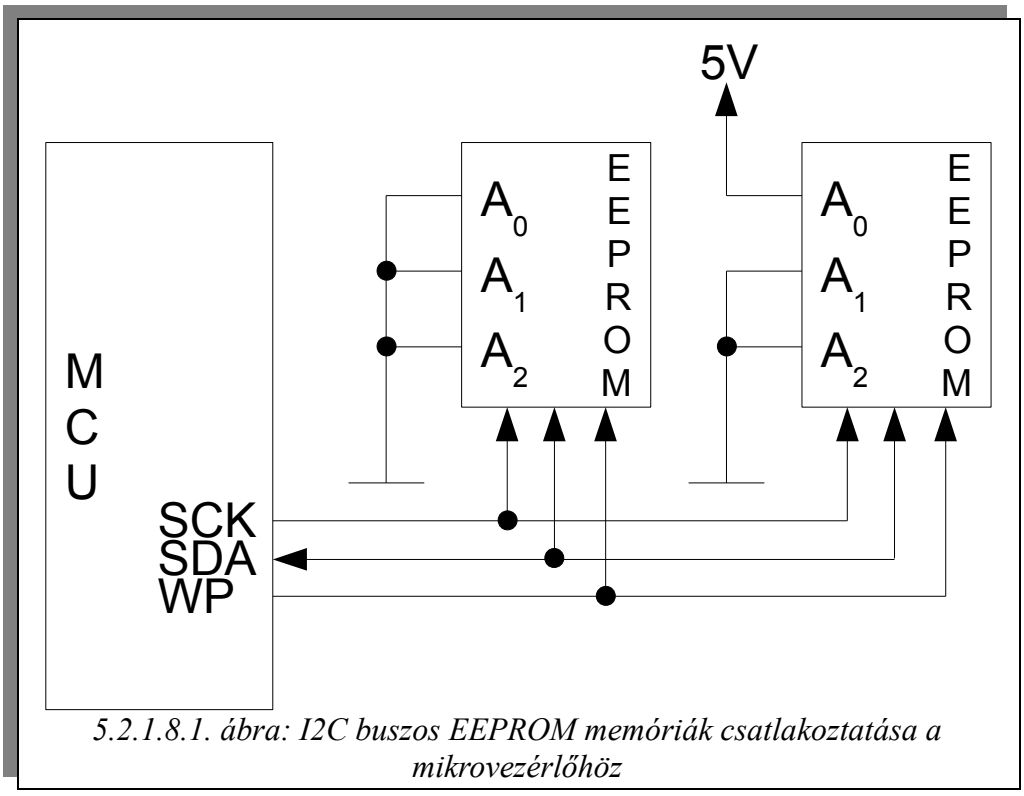


5.2.1.7.2. ábra: VDD feszültség hiba

44 A PICkit2 program *Tools → Target VDD Source* menüpontjában tudjuk megadni.
45 R5 ellenállás védelmi célokat szolgál – megakadályozza, hogy a target felől érkező nagyobb feszültségek tönkretegyék a PICkit2 programozó és hibakereső eszközt.
46 A VDD tápon megengedett maximális feszültségesés – alapértelmezetten: XV.
47 A helyes működés során bekövetkező kis feszültségesés okán választottak Schottkyt.
48 Egyébként a kapcsolásból, és nyáktervezésből adódó zavarfeszültséget érzékelnénk.

5.2.1.8 Soros EEPROM memóriák illesztése a mikrovezérlőhöz

A Programmer-To-Go funkció biztosításához szükség van a beégetni kívánt programot eltárolni képes memória áramkörre. A Microchip a saját fejlesztésű 24LC512 típusú alacsony fogyasztású 512 kiB-os I^2C[49] buszon kommunikáló EEPROM memóriája mellett tette le a voksot. A két memória IC címe nem változtatható – hardveresen került kialakításra (0, ill. 1). A 5.2.1.8.1. ábrán láthatjuk a memóriák mikrovezérlőhöz csatlakoztatását. Az SCK[50], és SDA[51] lábakat a Microchip ellátta a szabványos felhúzó ellenállásokkal. A PIC18F2550 típusú mikrovezérlőben az I^2C kommunikációs port hardveresen ki van építve, ezért az SCK, SDA lábak az RB_1, RB_0 lábakra kerültek. Ezen felül a Microchip alkalmazza a hardveres írásvédelmet is (WP[52]), ez az RC_6 lábra csatlakozik.

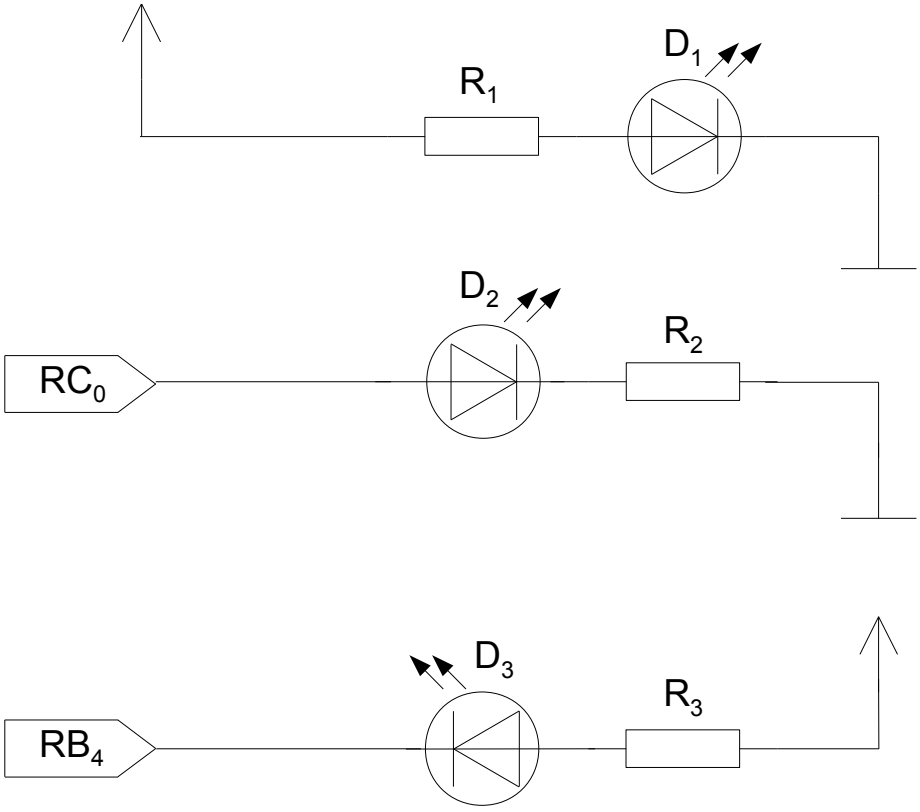


5.2.1.8.1. ábra: I2C buszos EEPROM memóriák csatlakoztatása a mikrovezérlőhöz

49 A Philips cég által kifejlesztett Inter IC kommunikációs busz segítségével két vezetéken történő kommunikációt valósíthatunk meg. Részletesebb információért l. www.muszeroldal.hu/measurenotes/i2c_hu.pdf.
50 Serial ClocK
51 Serial DatA
52 Write Protect

5.2.1.9 Állapotjelző LED-ek

A LED-ek soros ellenálláson keresztül csatlakoznak a mikrovezérlő lábaihoz, másik pontjuk 5V-ra, ill. földre kötve – funkcióiknak megfelelően (5.1. fejezet).



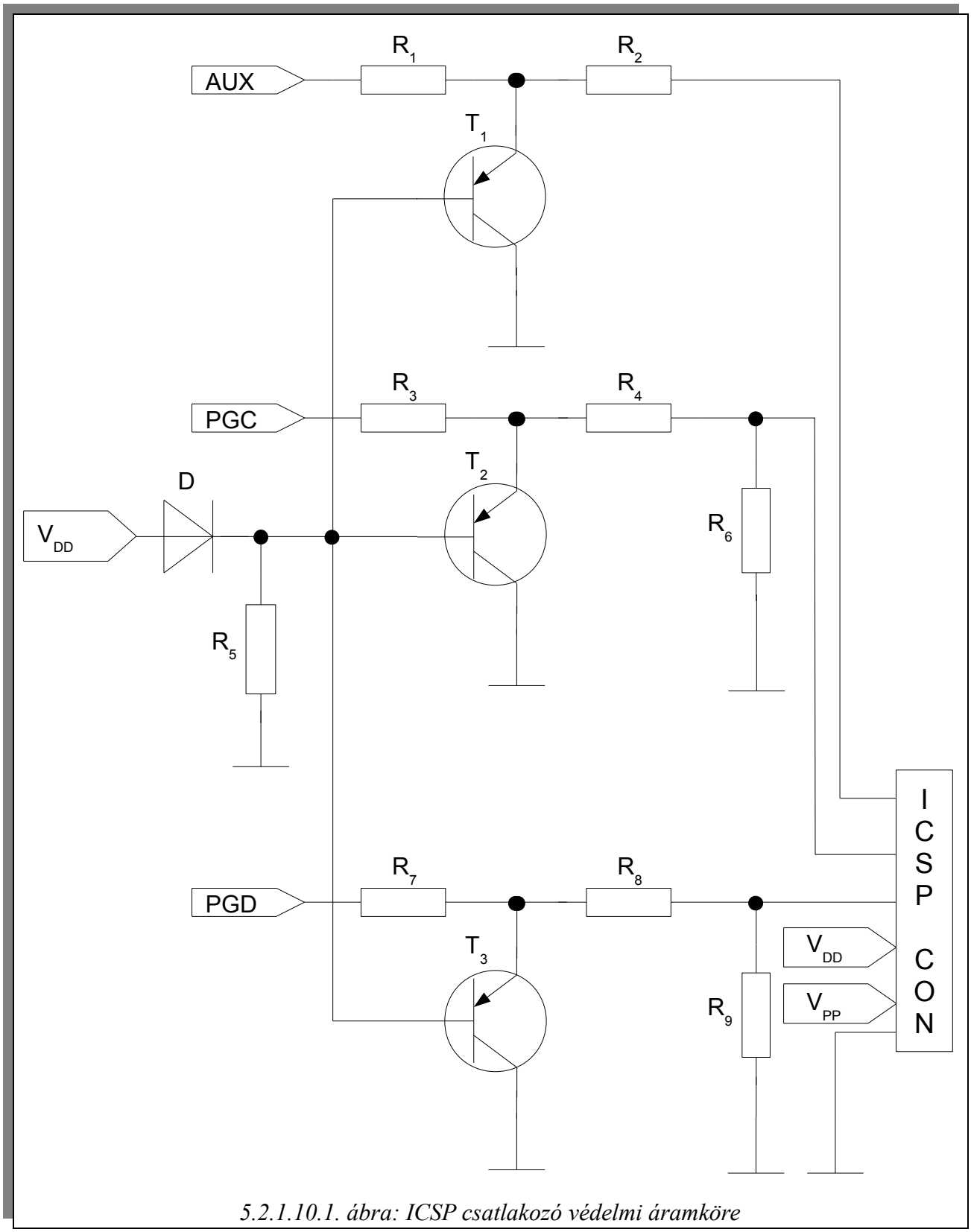
5.2.1.9.1. ábra: Állapotjelző LED-ek csatlakoztatása a mikrovezérlőhöz

Az ellenállások értéke 470Ω, így aktív állapotukban az áramigényük:

$$I_D = \frac{U_T - U_D}{R} = \frac{5\text{V} - 2\text{V}}{470\Omega} = 6{,}38\,mA$$

- D_1 zöld színű LED közvetlenül az USB tápra van kötve, így jelezve annak meglétét.
- D_2 piros színű LED-et az RC_0 lábon kiadott magas szint aktiválja. A HOST foglaltságát jelzi, ill. hibaüzenetek közlésére alkalmas.
- D_3 sárga jelzésű LED-et az RB_4 lábon kiadott alacsony szint aktiválja. A PICkit2 által a TARGET számára szolgáltatott tápfeszültséget kapcsoló FET állapotáról nyújt információt (ha világít, akkor a PICkit2 a TARGET számára biztosította a tápfeszültséget).

5.2.1.10 Target ICSP csatlakozófelület



5.2.1.10.1. ábra: ICSP csatlakozó védelmi áramköre

A céláramköri csatlakozó kialakításánál figyelemmel kell lennünk a target felől érkező helytelen jelekre (rossz csatlakozókiosztás a céláramkörön, hibás kialakítás, forrasztás, beültetés). A PICkit2 áramkör szoftveres áramkorláttal[53] van ellátva.

A 5.2.1.10.1. ábrán lévő tranzisztorok gondoskodnak az ICSP vonalak V_{DD} szinten[54] történő megfogásáról, ha van tápfeszültség[55] – ennek hiányában a vonalat maximum 0,6V-ig engedi emelkedni[56]. A PGC, PGD és AUX lábakon olyan céláramkörrel is kommunikálhatunk, amely nem tolerálja a TTL jelszintet, ezért szükséges, hogy a feszültségszintet a céláramkör tápfeszültségénél maximalizáljuk.

Mind a tápfeszültséget (a céláramkörét), mind a programozófeszültséget a PICkit 2 a belső A/D átalakító segítségével méri, és a gyárilag megadott hiszterézisnél nagyobb eltérés esetén a tápfeszültséget lekapcsolja a céláramkörről, és a piros Busy LED gyors villogásával jelzi a hibát.

53 A PICkit 2 100mA áramigényt jelez a PC felé, ezen áramérték átlépését az alaplapi vezérlőáramkörnek kell meggátolnia az operációs rendszer utasítása alapján.

54 Ez a céláramkör tápfeszültsége.

55 A D jelzésű diódán eső nyitóirányú feszültség közel megegyezik a tranzisztorok BE diódáján eső maximális nyitóirányú feszültséggel, ezért a tranzisztorok maximális U_{E0} feszültsége közel megegyezik a VDD pont feszültségével.

56 Az R_5 ellenálláson keresztül a tranzisztorok bázisukra GND-t kapnak, ezért az emitterük maximum ~0,6V-ra emelkedhet.

6 PICkit 2 V 2.61 programozószoftver bemutatása

6.1 Telepítési útmutató

Töltsük le a www.microchip.com/PICkit 2 weboldalról a legújabb szoftververziót Windows[57] operációs rendszer alá. A honlap jobb felső sarkában láthatjuk a szoftverújdonságokat (6.1.1. ábra).

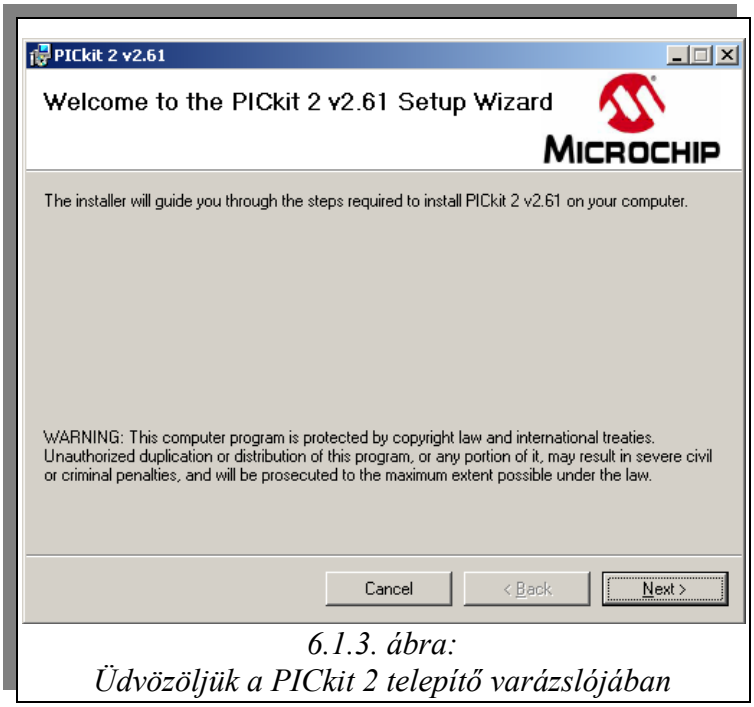
6.1.1. ábra: Microchip PICkit 2 programozó és hibakereső szoftver

6.1.2. ábra: Microchip PICkit 2 letöltés szekció

57 A Microchip biztosít egy parancssoros verziót is a PICkit 2 működtetésére – Linux operációs rendszer alatt ez használható.

A telepítőcsomag igényli a *.NET keretrendszer*t – amennyiben nem rendelkezünk vele, akkor a honlapon lefelé görgetve keressük meg a *Download* szekciót (6.1.2. ábra) és válasszuk a keretrendszert tartalmazó telepítőcsomagot. Itt megtalálhatjuk a PICkit 2 hivatalos – angol nyelvű – felhasználói útmutatóját, valamint több dokumentációt és szoftververziót is.

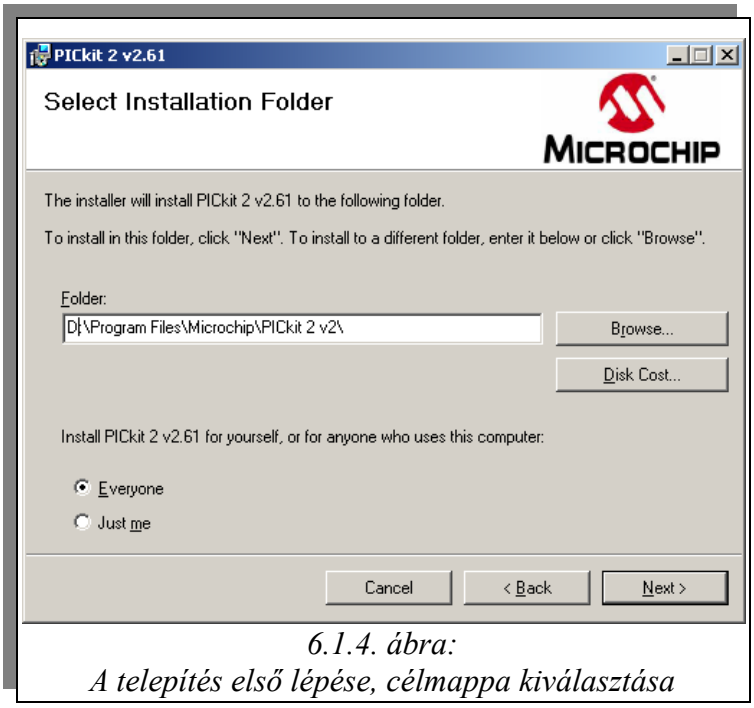
A telepítőcsomag zip tömörített formában áll rendelkezésünkre. Első lépésként csomagoljuk ki valamilyen tömörítőszoftverrel, vagy fájlkezelő programmal[58]. Az így kapott *Setup.exe* fájlt[59] indítsuk el és kövessük a telepítő program utasításait.



6.1.3. ábra:
Üdvözöljük a PICkit 2 telepítő varázslójában

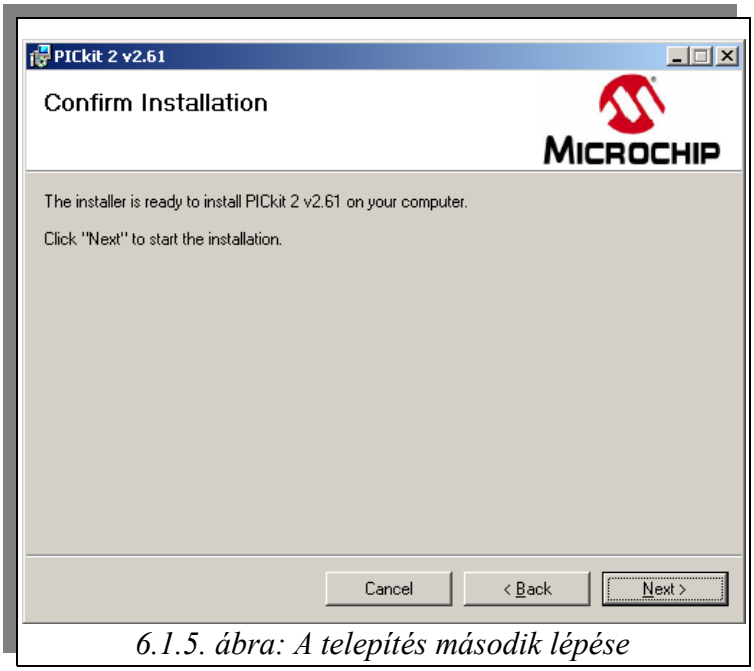
A telepítő végigvezet minket a PICkit 2 szoftver sikeres installálásán – valamint figyelmeztet a termék szerzői jogvédettségére (6.1.3. ábra). A *Next* gombra kattintva[60] válasszuk ki a telepítési útvonalat (6.1.4. ábra).

58 Használhatjuk a méltán népszerű Total commander-t vagy a Windows beépített tömörítőjét (Winzip) is.
59 A zip fájl tartalmaz még egy msi – microsoft telepítő információs – fájlt is
60 A telepítés során lehetőség van a billentyűzet alkalmazására is (TAB, Nyilak, Enter kombináció; vagy ALT + az opciófelirat aláhúzott karaktere; Kilépés: ESC).



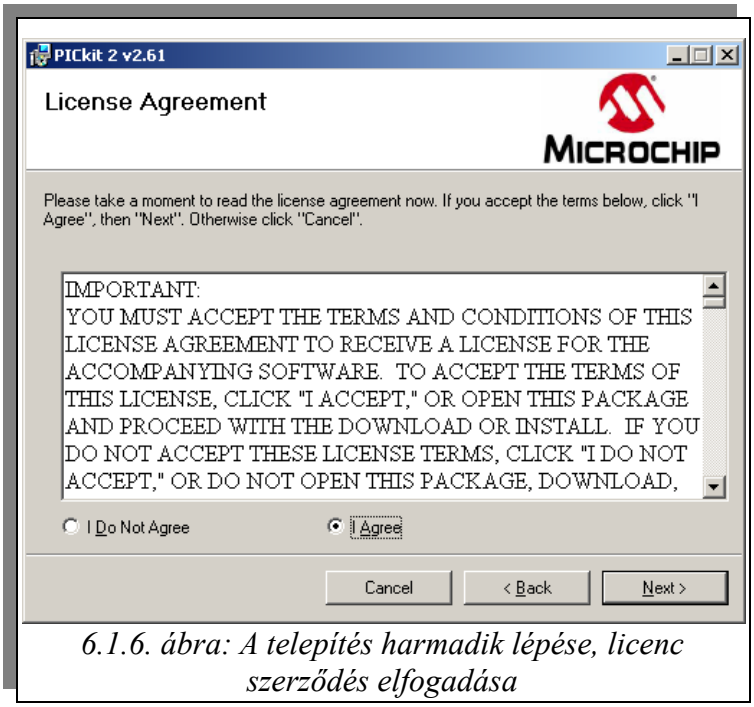
6.1.4. ábra:
A telepítés első lépése, célmappa kiválasztása

Az útvonal módosításához kattintsunk a *Browse* (Tallózás) gombra. A *Disk Cost...* *(*Lemezterület*)* gomb megnyomásával megvizsgálhatjuk a merevlemezünk foglaltságát. Végül válasszuk ki, hogy kinek engedélyezzük a program használatát – mindenkinek (*Everyone*), vagy csak saját magunknak (*Just me*). A megfelelő beállítások után kattintsunk a *Next* (Tovább) gombra!



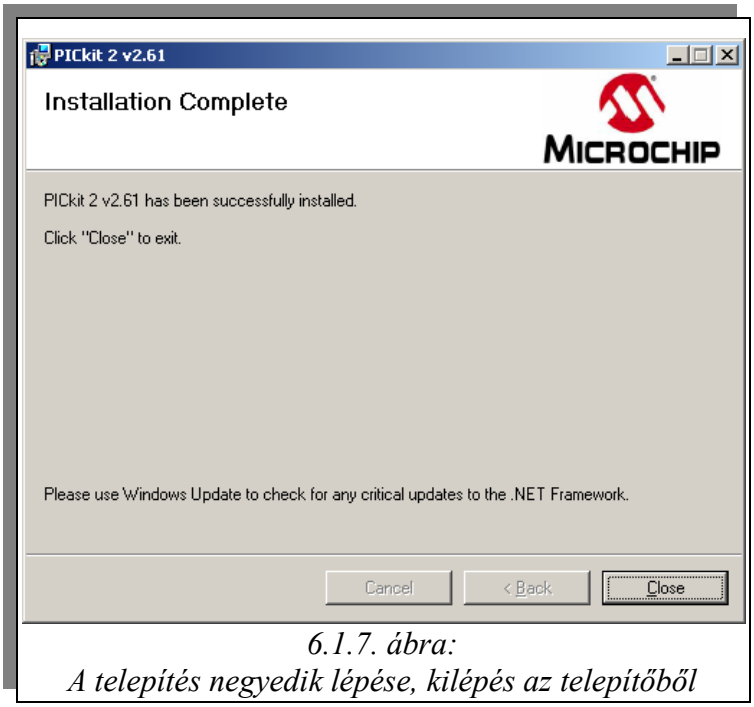
6.1.5. ábra: A telepítés második lépése

A következő lépésben (6.1.6. ábra) a telepítő közli velünk, hogy készen áll a program felinstallálására a számítógépünkre – ha változtatni szeretnénk a beállításokon kattintsunk a *< Back* gombra, a telepítés folytatásához válasszuk a *Next >* opciót, valamint lehetőségünk van a telepítés megszakítására is (*Cancel*).



6.1.6. ábra: A telepítés harmadik lépése, licenc szerződés elfogadása

Olvassuk el az angol nyelvű licenc szerződést (6.1.6. ábra) és ha egyetértünk vele[61] válasszuk az *I Agree* opciót, valamint kattintsunk a *Next* gombra.



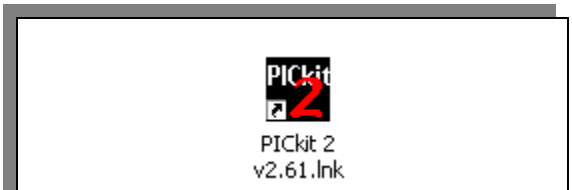
6.1.7. ábra:
A telepítés negyedik lépése, kilépés az telepítőből

A telepítő értesít minket (6.1.7. ábra), hogy az installáció sikeresen befejeződött, valamint figyelmeztet hogy mindig a legfrissebb Windows .NET keretrendszer összetevőt alkalmazzuk. A *Close* gombra kattintva lépjünk ki a telepítőből.

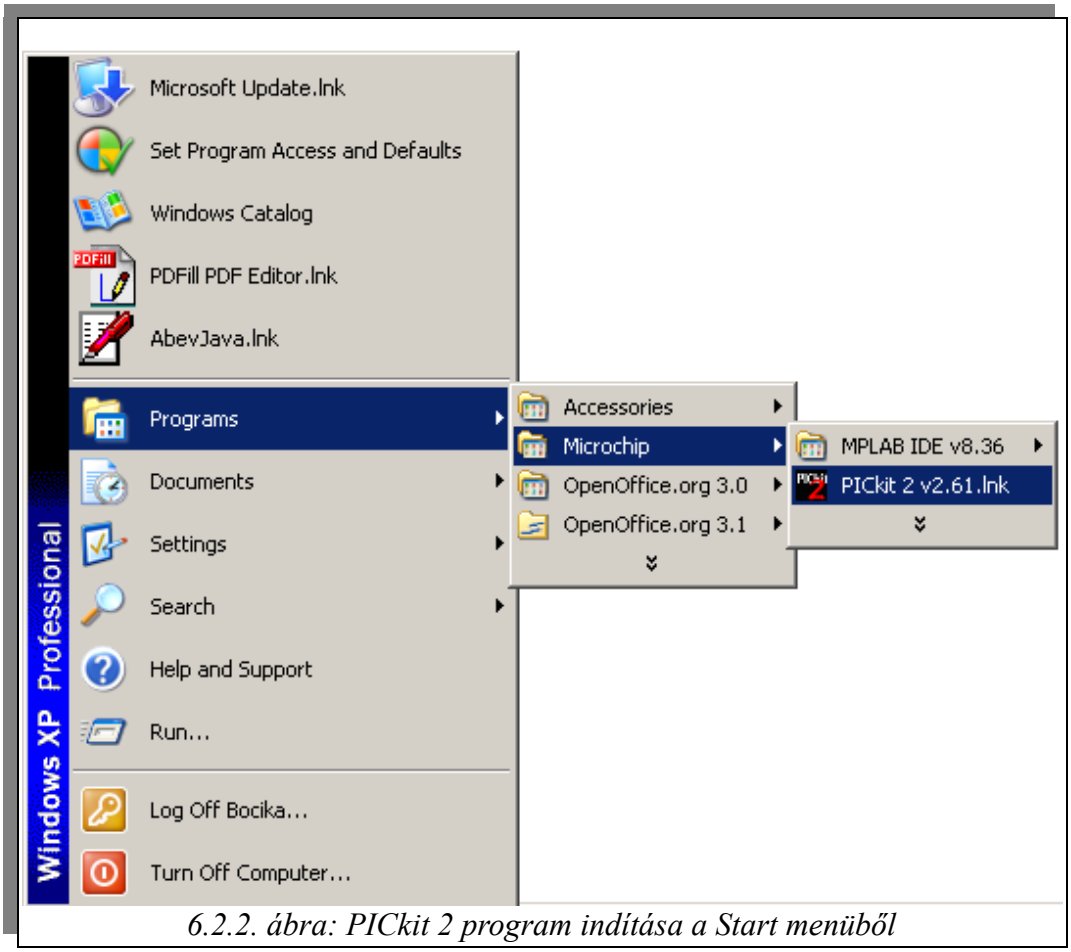
61 A szerződés elfogadása nélkül a szoftver nem telepíthető a számítógépre.

6.2 Az első indítás

Indítsuk el a PICkit 2 programot az asztalon található parancsikon (6.2.1. ábra) segítségével, vagy a Start menüben válasszuk ki a telepített programok közül (6.2.2. ábra).

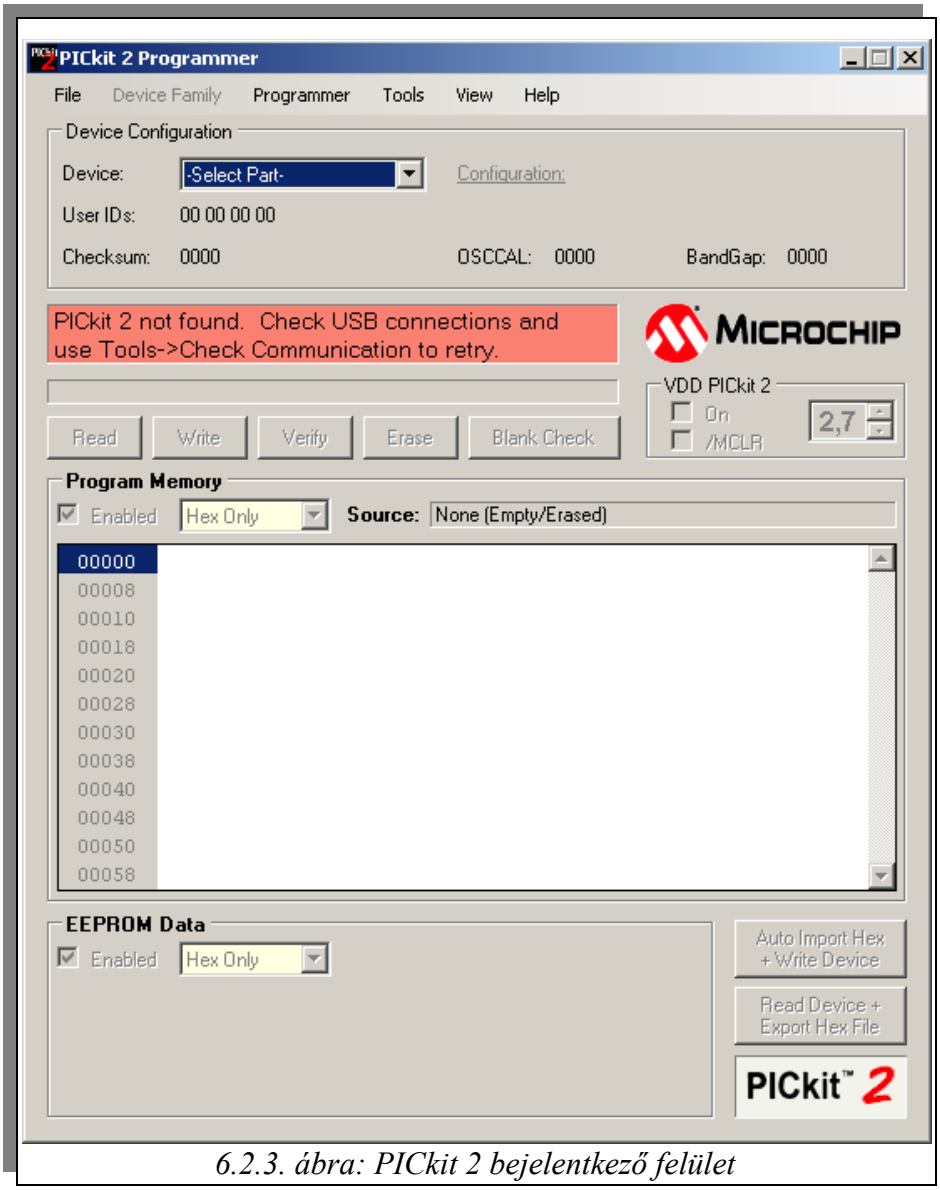


6.2.1. ábra: PICkit 2 parancsikon



6.2.2. ábra: PICkit 2 program indítása a Start menüből

PICkit 2 V 2.61 programozószoftver bemutatása

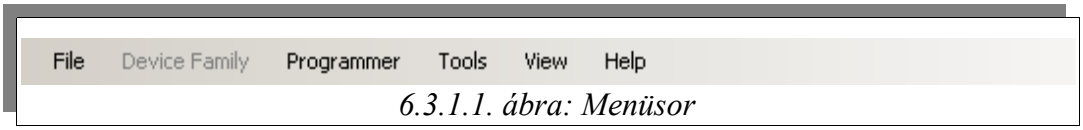


6.2.3. ábra: PICkit 2 bejelentkező felület

A program bejelentkező felületét több objektumra bonthatjuk.

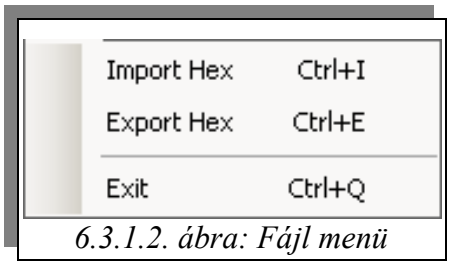
- Menubar (6.3.1.1. ábra)
- Device Configuration (6.5.1. ábra)
- PICkit 2 status (6.4.1. ábra)
- Programmer toolbar (6.7.1. ábra)
- VDD PICkit 2 (6.6.1. ábra)
- Program Memory Window (6.8.1. ábra)
- EEPROM Data Window (6.9.1. ábra)
- Special programming buttons (6.10.1. ábra)

6.3 Menürendszer ismertetése



6.3.1.1. ábra: Menüsor

6.3.1 File



6.3.1.2. ábra: Fájl menü

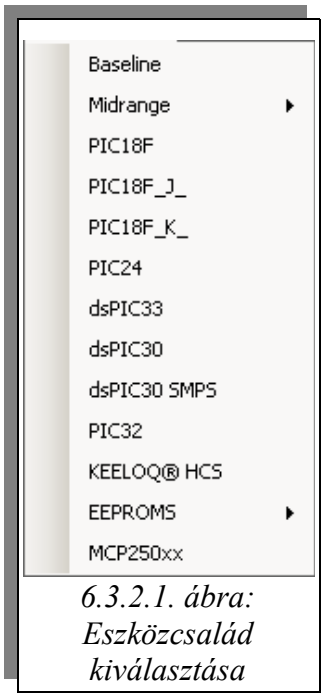
***Import Hex**:* Lehetőségünk van gépi kódú fájlt beimportálni. Az Intel Hexa 32 bites formátuma támogatott.

***Export Hex**:* A mikrovezérlőből kiolvasott kódot exportálhatjuk Intel Hexa 32 bites formátumú fájlba.

***Exit**:* Kilépés a programból.

A *Fájl* menü tartalmaz még egy *File History* menüpontot is[62], ahol az utolsó négy importált fájlt tudjuk megnyitni.

6.3.2 Device Family



6.3.2.1. ábra: Eszközcsalád kiválasztása

Baseline: Alapkonfiguráció beállítása (12 bites PIC mikrovezérlők).

Midrange: Középkategória (14 bites PIC mikrovezérlők). Kiválaszthatjuk az alap (*Standard*), és a már minimum 1,8V-os tápfeszültségről üzemelő mikrovezérlőket (*min 1,8V*) is.

PIC18F: 16 bites[63] PIC mikrovezérlők kiválasztása.

PIC18F_J_: 3,3V-os mikrovezérlők.

PIC18F_K_: Már 1,8V-ról is üzemelő mikrovezérlők.

PIC24: 16 bites[64] mikrovezérlő család.

dsPIC33: 3,3V-os mikrovezérlő család DSP[65] alkalmazásokhoz.

DsPIC30: 5V[66]-os mikrovezérlő család DSP alkalmazásokhoz.

dsPIC30 SMPS: Mikrovezérlőcsalád DSP alkalmazásokhoz, különösen kapcsolóüzemű tápokhoz.

62 Ameddig nem importáltunk be hex fájlt ez a menüpont nem látható.

63 Itt, és az előző két pontban is a bitszám a program memóriára vonatkozik, vagyis az utasítások hosszára. Mindhárom konfigurációban olyan mikrovezérlőket találunk, melyek 8 bites adatokkal dolgoznak. A programmemória szélességének változását a Harvard architektúra teszi lehetővé.

64 A bitszám itt és a következőkben az adatmemória szélességét jelöli.

65 Digital Signal Processing – digitális jelfeldolgozás.

66 A legtöbb mikrovezérlő már 2÷3V-os tápfeszültségről is üzemel, azonban mindig ellenőrizzük az adott típust! adatlapján, hogy alacsonyabb feszültségről milyen órajellel képes még működni.

PIC32: 32 bites mikrovezérlő család.

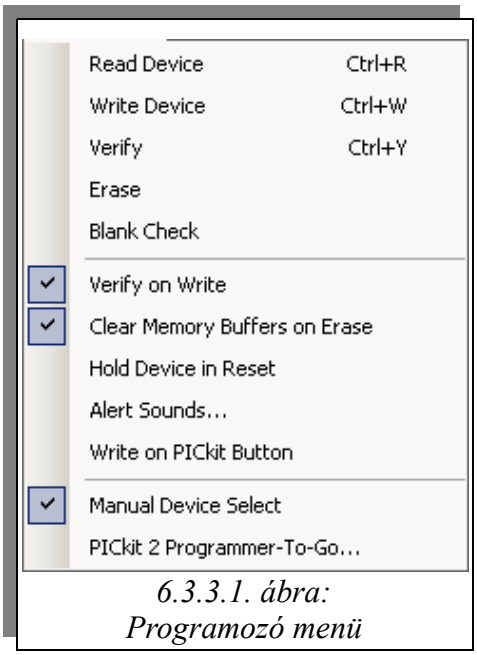
KEELOQ® HCS: KEELOQ ugrókódos dekóderek.

EEPROMS: Soros EEPROM-ok

MCP250xx: Beépített CAN-busszal ellátott mikrovezérlők.

A *Device Family* menüpontban választhatjuk ki, hogy a programozandó eszközünk mely családba tartozik. A későbbiek folyamán a *Device Configuration* ablakrészben csak ebből a családból választhatunk magunknak konkrét típust (6.5. fejezet).

6.3.3 Programmer



6.3.3.1. ábra: Programozó menü

Read Device: A kiválasztott eszköz kiolvasása (Programmemória, EEPROM adatmemória, azonosító, konfigurációs bitek).

Write Device: A kiválasztott eszköz felprogramozása (Programmemória, EEPROM adatmemória, azonosító, konfigurációs bitek).

Verify: A kiválasztott eszközből kiolvasott kódot (Programmemória, EEPROM adatmemória, azonosító, konfigurációs bitek) hasonlíthatjuk össze a programozó szoftverben tárolttal (pl. egy beimportált gépi kóddal).

Erase: A kiválasztott eszköz törlése[67].

Blank Check: Ellenőrzi, hogy a céláramkör üres-e.

Verify on Write: Ha be van jelölve, akkor a felprogramozás után automatikus ellenőrzés is történik.

Clear Memory Buffers on Erase: Ha be van jelölve az eszköz törlése során a szoftver memória pufferét is törli.

Hold Device in Reset: Ha be van jelölve, akkor a programozó az $\overline{MCLR}$ / V_{PP} lábat alacsony szinten tartja (folyamatos reszetben), amennyiben nem jelöljük be, akkor ezt a lábat elengedi (háromállapotú kimenet) és hagyja, hogy külső felhúzás hatására az eszköz elhagyja a reszet állapotot.

Alert Sounds...: A telepítés után a programozó szoftver nem használ hangjelzéseket, ezen azonban módosíthatunk és alapértelmezett – vagy bármilyen saját[68] – hangfájlt rendelhetünk a siker, veszély és hiba üzenetekhez (6.3.3.2. ábra).

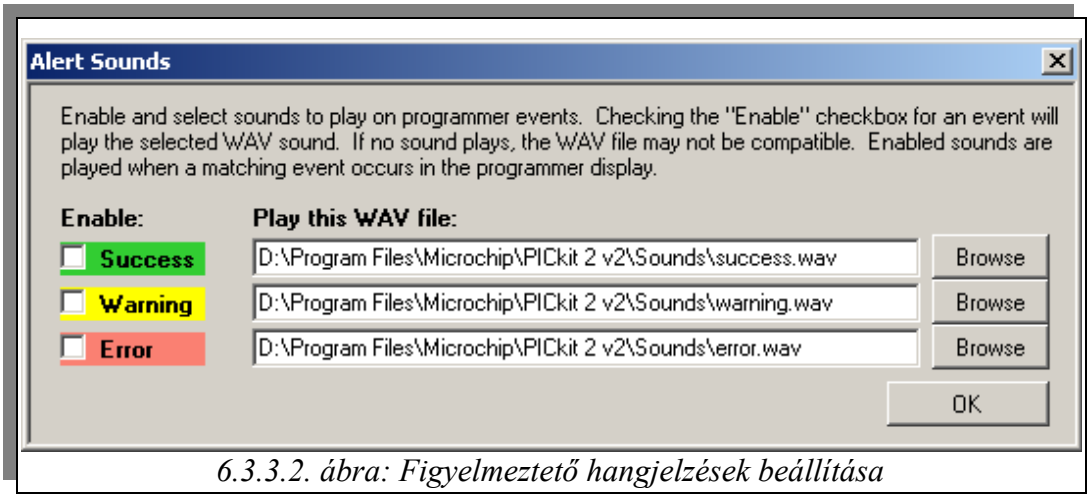
67 Ezen opció választásakor a programozó hardver bulk erase funkciót végez, melynek során a kódvédelemmel ellátott PIC mikrovezérlő is törlődik.

68 A hangnak wav formátumban kell lennie.

Write on PICkit Button: Egy forrás több eszközbe történő égetése során nem kell a PC oldali szoftverrel bajlódni, a beimportált fájlt a hardveren található nyomógombbal többször is beégethetjük ha ez az opció be van jelölve.

Manual Device Select: Kézi eszközkiválasztás, ha be van jelölve ez az opció, akkor nekünk van lehetőségünk kiválasztani az eszközt a *Device Configure* ablakrészben (6.5. fejezet).

PICkit 2 Programmer-To-Go...: A PICkit 2 programozó és hibakereső eszköz lehetőséget nyújt egy előre a hardverbe töltött program PC nélküli felprogramozására. A funkciót választva egy varázsló segítségével letölthetjük a PICkit 2 memóriájába a beégetni kívánt programot (6.3.3.1. fejezet).



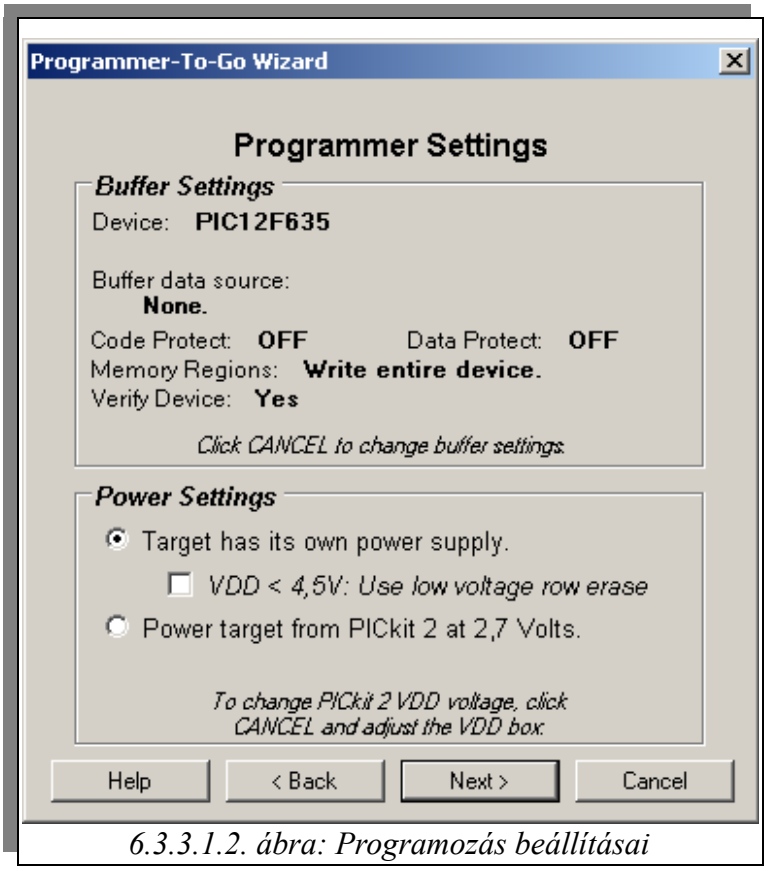
6.3.3.2. ábra: Figyelmeztető hangjelzések beállítása

6.3.3.1 Programmer – To – Go funkció használata

6.3.3.1.1. ábra: Üdvözöljük a PICkit 2 programozás és indulás funkció varázslójában

A PICkit 2 hardveregység tartalmaz két darab 24LC512 típusú soros EEPROM memóriát. Ezekbe letöltve a programot később PC és az azon futó szoftver nélkül is tudunk programozni. A PICkit 2-nek mindössze az 5V-os USB tápellátásra van szüksége.

A *Help* gombra kattintva további részleteket tudhatunk meg a Programmer-To-Go funkcióról – válasszuk a *Next* > opciót.



6.3.3.1.2. ábra: Programozás beállításai

A következő ablakban láthatjuk a puffer tartalmát – ezek a beállítások kerülnek majd az EEPROM memóriába – valamint a tápellátás beállításait.

Device: Az előzőleg beállított eszköz típusát mutatja.

Buffer data source: Itt láthatjuk a kiválasztott forrást (teljes elérési úttal).

Code Protect: Kódvédelem engedélyezett e, vagy sem[69].

Data Protect: Adatvédelem engedélyezett e, vagy sem.

Memory Regions: A programmemória és az EEPROM adatmemória fülnél lévő *Enabled* (engedélyezés) opciók függvényében három különböző beállítást olvashatunk.

Write entire device: A teljes eszköz írása (Programmemória és EEPROM adatmemória is)

Preserve EEPROM on write: Az eszköz írása az EEPROM adatmemória kihagyásával.

Write EEPROM data only: Csak az EEPROM adatmemória írása.

Verify Device: Programozás után az eszköz ellenőrzése.

Power Settings: Állítsuk be a programozás során alkalmazni kívánt tápfeszültségre vonatkozó beállításokat.

69 Alapértelmezetten a forrásfájl beállításai lesznek irányadóak, azonban – ezt felülírva – mind a kód-, mind az adatvédelmet külön beállíthatjuk a *Tools* menüben.

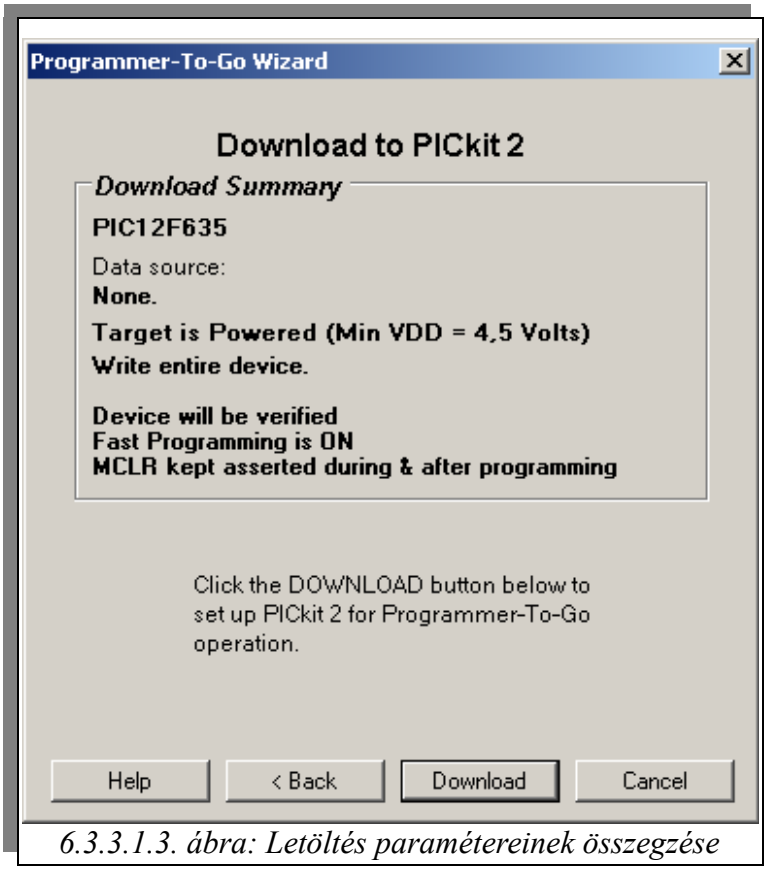
Target has its own power supply: Az égetéshez szükséges VDD feszültséget a céláramkör biztosítja.

V_{DD} < 4,5V use low voltage row erase:

Ha a tápfeszültség kisebb mint 4,5V row erase használata[70].

Power target from PICkit 2 at 2,7 Volts:

A programozás során a V_{DD} tápfeszültséget a PICkit 2 biztosítja[71].



6.3.3.1.3. ábra: Letöltés paramétereinek összegzése

A *Next* > gombra kattintva egy összegzést kapunk a beállításokról – amennyiben a tápfeszültség beállításával nem vagyunk megelégedve visszatérhetünk az előző ablakba a < *Back* gomb hatására (a *Cancel* opció választásával átállíthatjuk az egyéb paramétereket is). Megbizonyosodva a PICkit 2 hardver csatlakoztatásáról, kattintsunk a *Download* gombra.

Az összegzésben szereplő beállításokat a következő helyeken tudjuk módosítani:

70 Nem minden típusnál alkalmazható. Ha bejelöljük az opciót, akkor (V_{DD} < 4,5V esetén) ún. row erase funkciót alkalmaz a PICkit 2 az eszköz törlésénél programozás előtt. Ez a kódvédelemmel ellátott PIC mikrovezérlők esetén hatástalan, vagyis ezeket nem fogjuk tudni felülírni.

71 Értékét a V_{DD} ablakban (6.6. fejezet) módosíthatjuk.

Device: Válasszuk ki a *Device Family* menüpontban (6.3.2.1.) az alkalmazni kívánt eszköz családját, majd az eszköz kiválasztó ablakban (6.5.1. ábra) a konkrét típust.

Data Source: A *File → Import Hex* (6.3.1.2. ábra) menüponttal, vagy a *Ctrl+I* billentyűkombinációt alkalmazva nyissuk meg a beégetni kívánt hex fájlt.

Power Settings: A *Tools → PICkit 2 Programmer – To – Go* menüponttal megnyitott varázsló második lépésében (6.3.3.1.2. ábra) válasszuk ki a tápellátás beállításait.

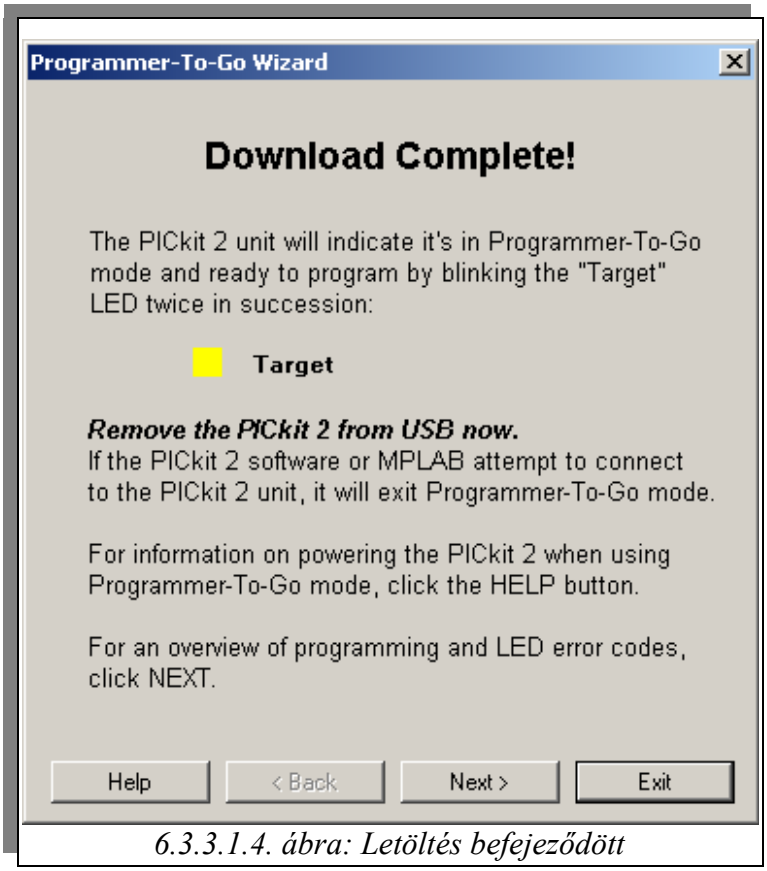
Memory Regions: A programmemóriához (6.8.1. ábra) és az EEPROM memóriához (6.9.1. ábra) tartozó ablakban használjuk az *Enable* (engedélyezés) opciót (36. oldal).

Device will be verified: A *Programmer → Verify on Write* opció bejelölésével programozás során az eszközbe töltött program – írás után – kiolvasásra és a beírandó programmal összehasonlításra (verify) kerül.

Fast Programming: Amennyiben a *Tools → Fast Programming* opció be van kapcsolva, akkor a PICkit2 a lehető legnagyobb sebességgel próbálja meg beégetni az áramkört. Ha nem jelöljük be, akkor a programozó csökkenti a kommunikáció sebességét – ez hasznos lehet, ha a céláramkör beterheli az ICSP vonalakat.

MCLR kept asserted during & after programming:

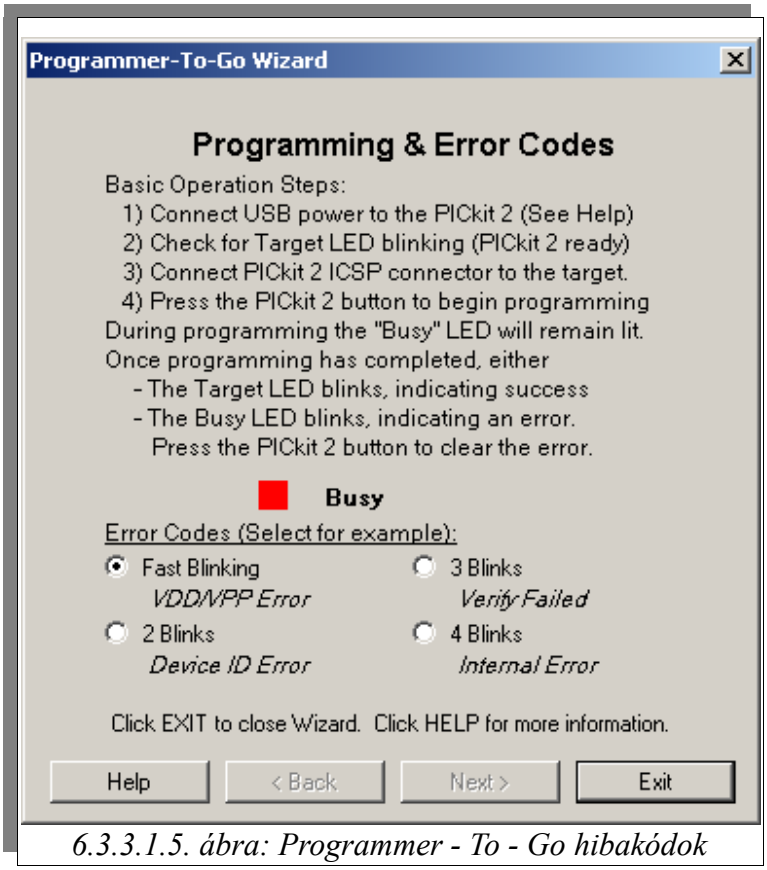
MCLR láb magas szinten tartása programozás után (*Programming → Hold Device in Reset*).



6.3.3.1.4. ábra: Letöltés befejeződött

A 6.3.3.1.4. ábrán látható ablakban a varázsló közli velünk, hogy a letöltés befejezéséről a hardveregységen található sárga – Target jelzésű – LED villogása fog minket értesíteni.

A piros – Busy jelzésű – LED villogása informál minket az esetleges hibajelenségekről. A *Next* > gombra kattintva láthatjuk az egyes hibákhoz tartozó kódokat. A hibákat a nyomógomb működtetésével tudjuk törölni – egyszeri fellépésükkor a nyomógomb többszöri megnyomásával próbáljuk kiküszöbölni a hibát.



6.3.3.1.5. ábra: Programmer - To - Go hibakódok

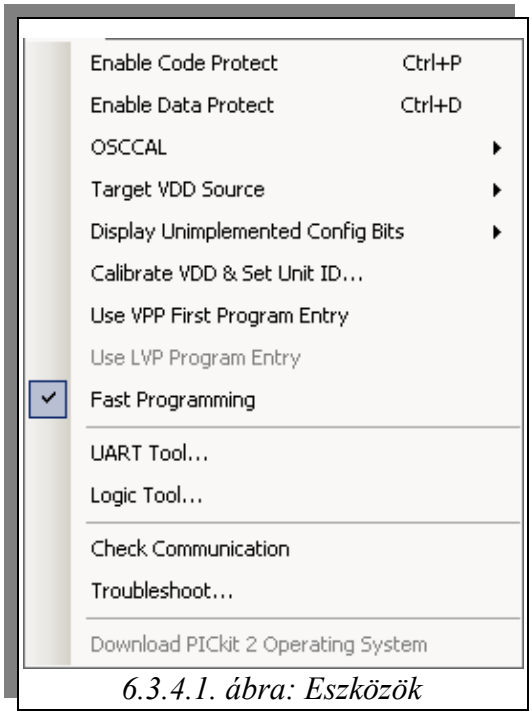
Gyors villogás: tápfeszültség, vagy programozófeszültség hiba. A gyári PICkit 2 sajnos csak nagyon korlátozott mértékben – 60mA – képes ellátni árammal a céláramkört. Egy átlagos pufferelés (470÷1000µF) is túlterhelheti a PICkit2-őt. A tápfeszültséget mindig a céláramkör csatlakoztatása nélkül ellenőrizzük.

Két villogás: eszközazonosító hiba. Ellenőrizzük a PIC típusát, egyezik e az előre megadottal. Vizsgáljuk meg a csatlakozást.

Három villogás: ellenőrzés nem sikerült. Amennyiben a kódvédelem be van kapcsolva a hibaüzenet normális (ebben az esetben nem érdemes ellenőrzést kérni, mert mindig nullákat olvas ki a mikrovezérlőből a programozó). Ha a kódvédelem nem lett aktiválva, akkor próbálkozzunk ismételt égetéssel, hátha valamilyen sztochasztikus zavar jelentkezett a programozás során.

Négy villogás: belső hiba történt. Próbálkozzunk a PICkit 2 reszetelésével. Ha nem oldja meg a problémát, akkor alkalmazzuk újra a Programmer-To-Go funkciót.

6.3.4 Tools



6.3.4.1. ábra: Eszközök

Enable Code Protect: Kódvédelem engedélyezése (a forráskódban lévő konfigurációs bit felülbírálása).

Enable Data Protect: Adatvédelem engedélyezése (a forráskódban lévő konfigurációs bit felülbírálása).

OSCCAL: Oszcillátor kalibrálása.

Target VDD Source: A céleszköz tápellátásának kiválasztása[72].

Display Unimplemented Config Bits: Az eszközben nem megvalósított konfigurációs biteket is mutassa a szoftver.

Calibrate VDD & Set Unit ID...: A PICkit 2 által mért tápfeszültség kalibrálása, és az eszköz azonosítójának beállítása (6.11. fejezet).

Use VPP First Program Entry: Amennyiben be van jelölve, akkor a PICkit 2 számára engedélyezett a csatlakozás, és felprogramozása akkor is, ha a konfiguráció és a kód olyan hatással van a PGD és PGC lábakra, amelyek megakadályoznák a mikrovezérlőt a programozómódba való belépésben. Használata megköveteli, hogy a PICkit 2 adja a tápfeszültséget.

Use LVP Program Entry: Alacsony feszültségű programozási mód engedélyezése. Lehetővé teszi az LVP programozást, ebben az esetben a mikrovezérlő PGM lábát az AUX vonalra kell csatlakoztatnunk.

Fast Programming: Ha be van jelölve, akkor engedélyezve van a gyors programozás[73].

UART Tool...: A PICkit 2 használható UART eszközként (6.12. fejezet).

Logic Tool...: A PICkit 2 képes egy elég kezdetleges 3 csatornás logikai analizátorként is működni (6.13. fejezet).

Check Communication: Kommunikáció ellenőrzése (USB, és ICSP). Az eredményt a PICkit 2 státuszablakban ellenőrizhetjük

72 Auto-Detect: a PICkit 2 hardveregység automatikusan felismeri, hogy a céleszköznek van e saját tápellátása, ha nincs, akkor biztosítja azt.
Force PICkit 2: A tápellátást a PICkit2 biztosítja.
Force Target: A tápellátást a céláramkör biztosítja.

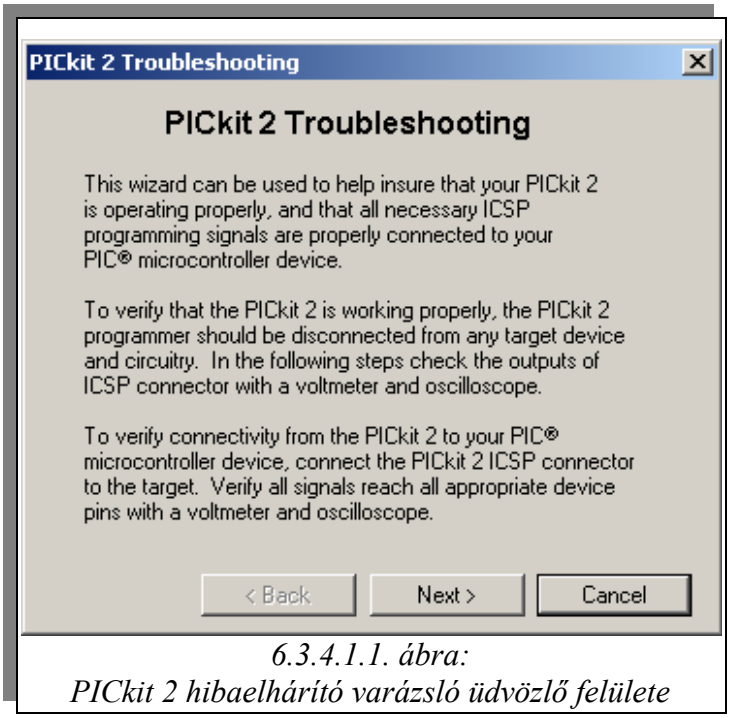
73 A PICkit 2 a lehető legnagyobb sebességgel (~1Mhz) használja az ICSP vonalat. Ha az opciót nem jelöljük be, akkor a sebességet a PICkit 2 lecsökkenti (~250kHz), így növelve az eszköz stabilitását.

Troubleshoot...: Hibaelhárítás. Egy varázsló (6.3.4.1. fejezet) segítségével ellenőrizhetjük a hardveregység legfontosabb paramétereit.

Download PICkit 2 Operating System: A PICkit 2 operációs rendszerének letöltése.

6.3.4.1 Hibaelhárítás

A PICkit 2 szoftver a *Tools* → *Troubleshoot...* menüpontban lehetőséget biztosít számunkra, hogy hibakeresést, és -elhárítást végezzünk. A hibakereső üzemmódot a sárga LED folyamatos világítása jelzi.



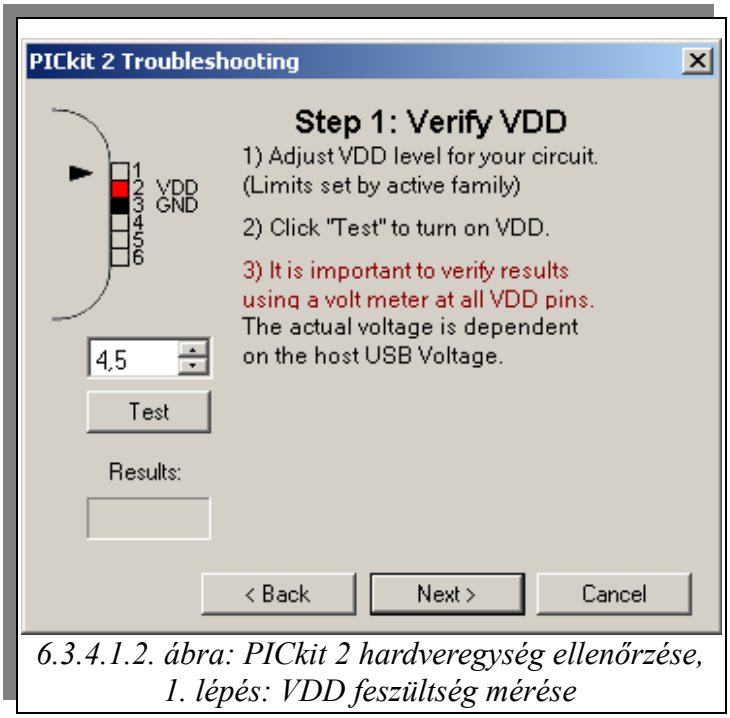
6.3.4.1.1. ábra:
PICkit 2 hibaelhárító varázsló üdvözlő felülete

Kezdetnek olvassuk el a varázsló kezdő instrukcióit, majd a *Next* > gombra kattintva kezdjünk hozzá a hibakereséshez, a *Cancel* gomb segítségével kiléphetünk a funkcióból.

A varázsló segítségével ellenőrizhetjük, hogy a PICkit 2 hardveregység helyesen működik, és helyesen csatlakozik az ICSP vonalakon keresztül a céláramkörhöz.

Válasszuk le a céláramkört a PICkit 2-ről[74], az ICSP vonalakra csatlakoztassunk voltmérőt (DMM), oszcilloszkópot, és a következő lépésben mérjük az adott pontok potenciálját[75].

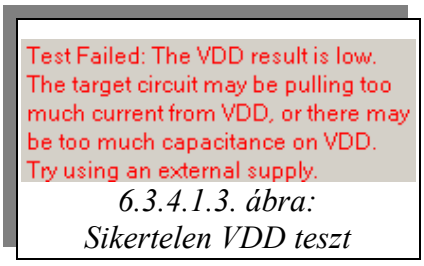
74 Ez a lépés nagyon fontos! A hibakeresést először a terheletlen PICkit 2 áramkörön kell elvégeznünk.
75 A varázsló az adott lépésben grafikusan jelzi számunkra, hogy mely lábat vizsgáljuk éppen.



6.3.4.1.2. ábra: PICkit 2 hardveregység ellenőrzése, 1. lépés: VDD feszültség mérése

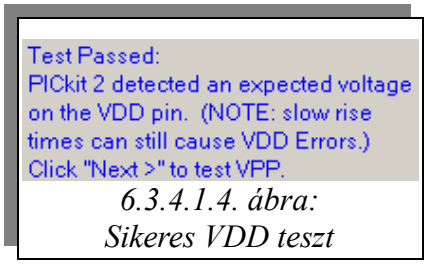
Az első lépésben a V_{DD} feszültséget fogjuk ellenőrizni. Állítsuk be a minimálisan szükséges feszültségszintet (ezt a programozni kívánt céláramkör határozza meg).

Kattintsunk a *Test* gombra, és közben mérjünk rá a vonalra multiméter, és oszcilloszkóp[76] segítségével.



6.3.4.1.3. ábra: Sikertelen VDD teszt

Amennyiben a 6.3.4.1.3. ábra szerinti sikertelen teszteredményt kapjuk[77], vizsgáljuk meg a V_{DD} vonal és a GND közötti impedanciát (ezt a mérést passzív panel esetén végezzük), valamint a csatlakozó épségét. A 6.3.3.1.4. ábrán látható üzenet esetén oszcilloszkóp segítségével ellenőrizzük a V_{DD} vonal statikus voltát.

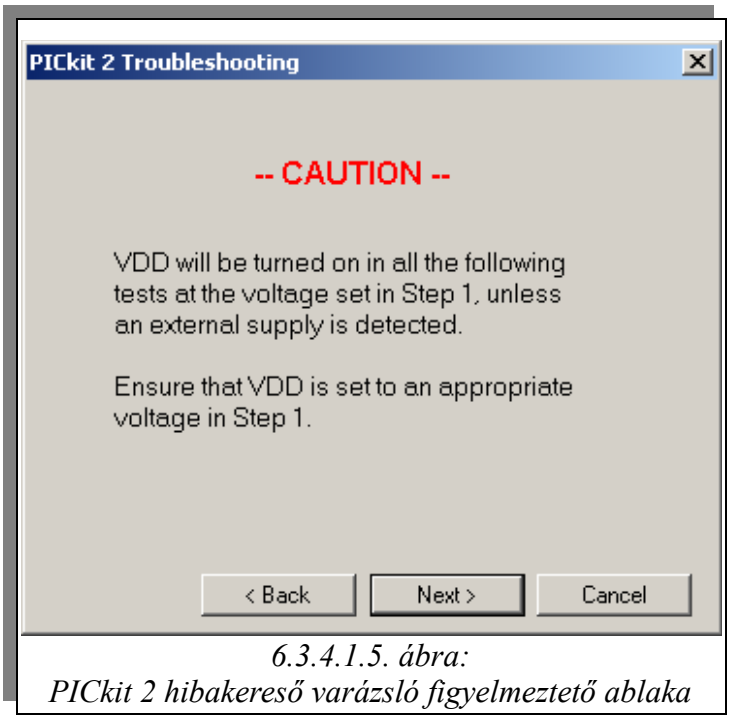


6.3.4.1.4. ábra: Sikeres VDD teszt

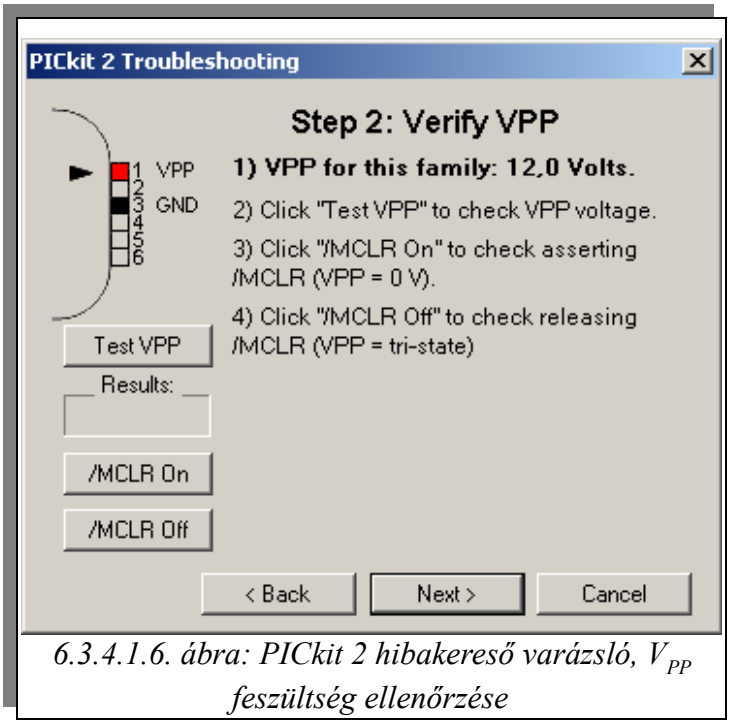
76 Az oszcilloszkópon láthatjuk az esetlegesen a tápvonalra szuperponálódott zavarjelet, valamint a tápvonal felfutási idejére is tudunk következtetni.

77 Ezt mindenképpen hasonlítsuk össze a valódi műszeres méréssel.

A *Next >* gombra kattintva lépjünk tovább a VPP feszültség vizsgálatára (6.3.4.1.6. ábra). A varázsló egy figyelmeztető ablakban (6.3.4.1.5. ábra) tájékoztat minket arról, hogy a következő tesztekhez szükség lesz a VDD feszültségre, amit az előző lépésben beállítottunk – kivéve ha a céláramkör biztosítja a tápfeszültséget.

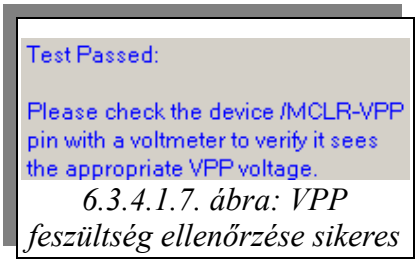


6.3.4.1.5. ábra:
PICkit 2 hibakereső varázsló figyelmeztető ablaka



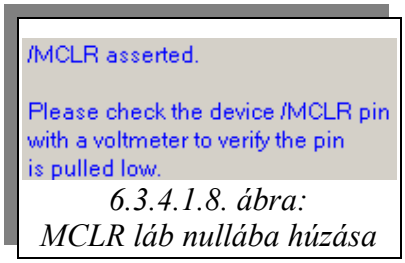
6.3.4.1.6. ábra: PICkit 2 hibakereső varázsló, V_{PP} feszültség ellenőrzése

A V_{PP} feszültség ellenőrzése során a varázsló tájékoztat minket a beállított PIC mikrovezérlő családhoz tartozó programozó feszültség értékéről, majd ellát minket a szükséges instrukciókkal. A *Test VPP* gombra kattintva a *Results* ablakban látjuk a PICkit 2 által kiadott, mért, és szabályozott V_{PP} programozó feszültséget – ezt ellenőrizzük multiméter, és oszcilloszkóp segítségével is.



6.3.4.1.7. ábra: VPP feszültség ellenőrzése sikeres

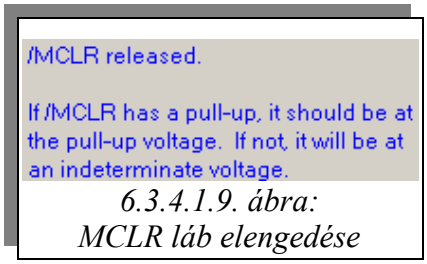
Amennyiben *Test Passed* üzenetet üzenetet kapunk (6.3.4.1.7. ábra), ellenőrizzük a programozófeszültséget multiméterrel is.



6.3.4.1.8. ábra: MCLR láb nullába húzása

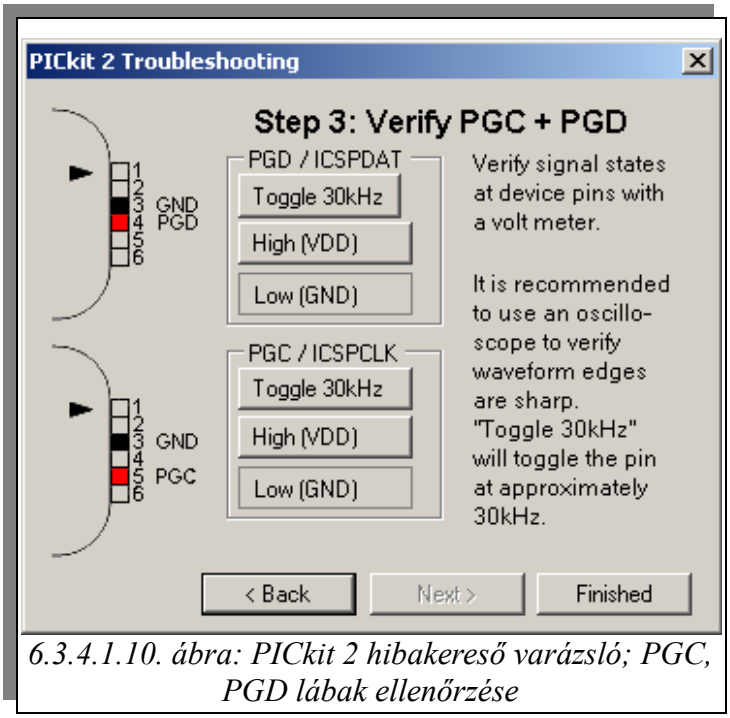
A */MCLR On* gombra kattintva a programozó az $\overline{MCLR}$ lábat alacsony szintre húzza. Mérjük az $\overline{MCLR}$ láb potenciálját – 0V-t kell jeleznie a műszernek.

Az */MCLR Off* gomb hatására a PICkit2 elengedi a V_{PP} lábat, próbáljuk felhúzni egy ellenálláson keresztül 5V-ra, és ellenőrizzük a multiméterrel, hogy ez sikerült-e.



6.3.4.1.9. ábra: MCLR láb elengedése

A 6.3.4.1.10. ábrán láthatjuk a hibakeresés 3. lépését, a PGC és PGD ellenőrzését. Lehetőségünk van magas (*High (VDD)*), ill alacsony szintbe (*Low (GND)*) billenteni az adott lábat, valamint beállíthatunk egy 30kHz-es 50%-os kitöltési tényezőjű TTL négyszögjelet (*Toggle 30kHz*).



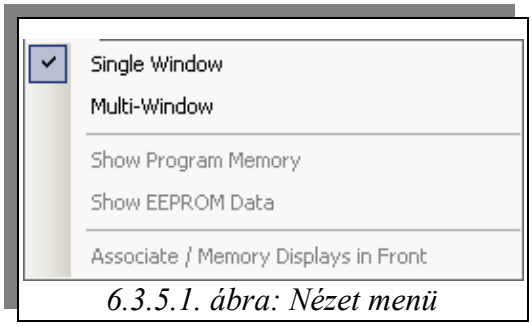
6.3.4.1.10. ábra: PICkit 2 hibakereső varázsló; PGC, PGD lábak ellenőrzése

A beállított értékeket ellenőrizzük multiméter, és oszcilloszkóp segítségével[78].

Amennyiben mindent rendben találtunk a *Finished* gomb megnyomásával befejezhetjük a hibakeresést.

78 Érdemes a két lábat egyszerre is mérni különböző beállítások mellet, így fel tudjuk deríteni a „lábak összezáródásából” adódó hibát is.

6.3.5 View



6.3.5.1. ábra: Nézet menü

Single Window: A teljes kezelőfelület egy ablakban történő elhelyezése és egy egységként kezelése.

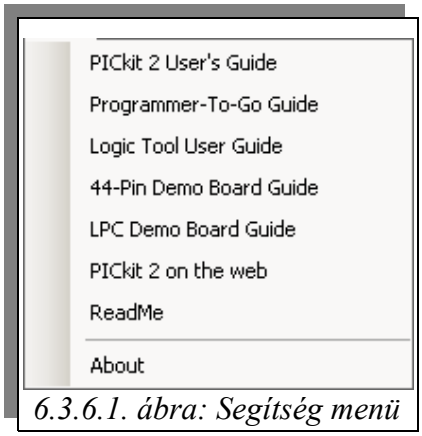
Multi Window: A memóriatérkép külön ablakba történő leválasztása.

Show Program Memory: Program memória mutatása (csak multi window beállításnál).

Show EEPROM Data: EEPROM adatmemória mutatása (csak multi window beállításnál).

Associate/Memory Displays in Front: Ha az opció nincs bejelölve, akkor a memória és a fő ablak teljesen elkülönül, amennyiben bejelöltük, akkor a főablakban történő módosítás magával vonzza a memóriaablakot is (pl. a főablak lecsukása a tálcára a memóriaablakot is lecsukja.). Az opció használatára csak multi window beállításnál van lehetőségünk.

6.3.6 Help



6.3.6.1. ábra: Segítség menü

PICkit 2 User's Guide: A hivatalos angol nyelvű PICkit 2 felhasználói kézikönyv megnyitása.

Programmer – To – Go Guide: A Programmer – To – Go funkcióhoz tartozó kézikönyv megnyitása.

Logic Tool User Guide: Leírás a PICkit 2 logikai analizátorként való használatáról.

44-Pin Demo Board Guide: A 44 lábú bemutatópanel leírása.

LPC Demo Board Guide: A kis lábszámú (Low Pin Count) bemutatópanelhez tartozó kézikönyv megnyitása. (8, 14, 20 lábú eszközök támogatottak, részletes listához l. a kézikönyv.)

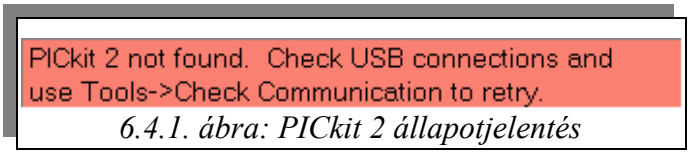
PICkit2 on the web: PICkit 2 a weben. A menüpont hatására az alapértelmezett böngészőben megjelenik a Microchip weboldalának PICkit 2 részlege (www.microchip.com/pickit2).

ReadMe: Olvass el fájl. Tartalmazza a támogatott eszközök listáját, és a használatukhoz szükséges bekötést.

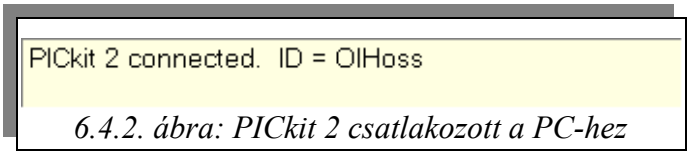
About: A PICkit 2 névjegye.

6.4 PICkit 2 csatlakoztatása

A hardveregység nem igényel különleges meghajtóprogramot (drivert). A windows operációs rendszer automatikusan felismeri USB eszközként. Csatlakoztassuk az áramkört a PICkit 2 programozó szoftver segítségével (a kommunikáció automatikusan megindul, és a státuszablakban látnunk kell ennek eredményét (6.4.1. ábra, 6.4.2. ábra).



6.4.1. ábra: PICkit 2 állapotjelentés



6.4.2. ábra: PICkit 2 csatlakozott a PC-hez

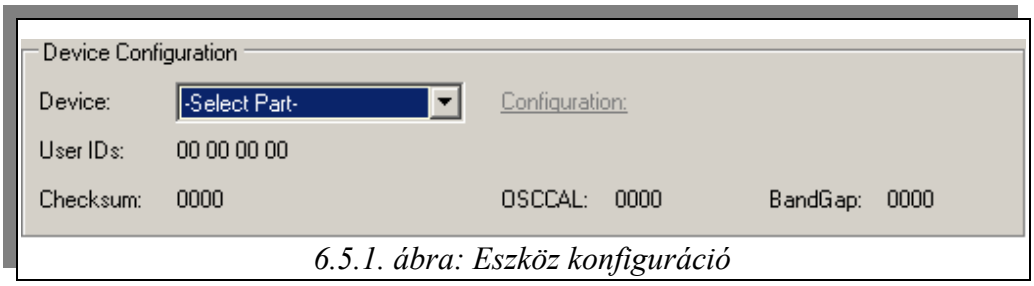
Amennyiben a 6.4.1. ábra szerinti üzenetet kapjuk, akkor valami probléma áll fenn a kommunikációban. Vizsgáljuk meg a PICkit 2 USB csatlakozójának épségét, valamint a kábelt. Ellenőrizzük, hogy a PICkit 2-őt az operációs rendszer felismerte egyáltalán mint USB-s eszközt. Sajnos a Windows operációs rendszert használók esetében előfordulhat, hogy egy korábban indított és nem megfelelően befejezett programfolyamat (pl. MPLAB) lefoglalta az USB portot, és csak az operációs rendszer újraindítása oldja meg a problémát.

A *Tools → Check Communication* menüpont segítségével tudjuk a csatlakozási kísérletet megismételni.

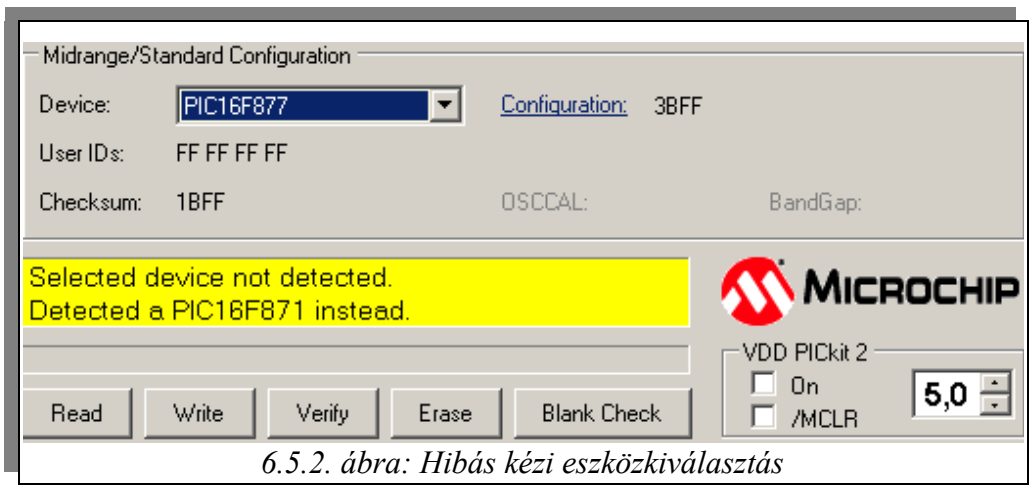
A 6.4.2. ábra szerinti üzenet esetén a PICkit 2-őt felismerte a szoftver, rögtön ki is olvassa az azonosítóját. Erre azért van szükség, mert egyszerre több PICkit 2-őt is használhatunk fejlesztésre, ilyenkor a fejlesztőrendszer mindig megkérdezi, hogy mely azonosítójú PICkit 2-vel szeretnénk végrehajtani az adott feladatot (6.15. fejezet).

6.5 Eszköz kiválasztása és beállítása

A PICkit 2 szoftvere lehetőséget nyújt kézi és automatikus eszközkiválasztásra. Amennyiben automatikus eszközfelismerést állítottunk be az eszközkonfigurációs panel read only[79]. Kézi eszközkiválasztáshoz jelöljük be a *Programmer → Manual Device Select* opciót! Válasszuk ki a *Device Family* menüben az alkalmazott eszköz családját.



6.5.1. ábra: Eszköz konfiguráció



6.5.2. ábra: Hibás kézi eszközkiválasztás

Az eszközkonfigurációs ablakban kézi eszközkiválasztás esetén megadhatjuk a céláramkör pontos típusát. Amennyiben a kiválasztott és az érzékelt eszköz típusa nem egyezik meg, a PICkit 2 erről a státuszablakban tájékoztat minket[80] (6.5.2. ábra). A többi opció:

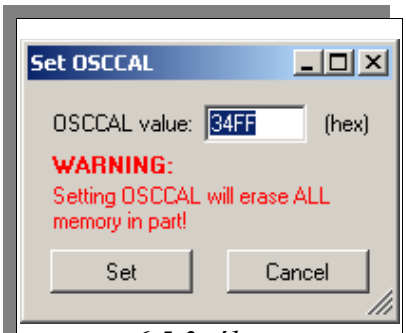
79 Csak olvasható

80 Ha a kiválasztott és az érzékelt eszköz nem ugyanazon család tagja, vagy nincs érzékelt eszköz, akkor a „No device detected.” üzenetet olvashatjuk.

User IDs: Felhasználói azonosító.

Checksum: Ellenőrző összeg.

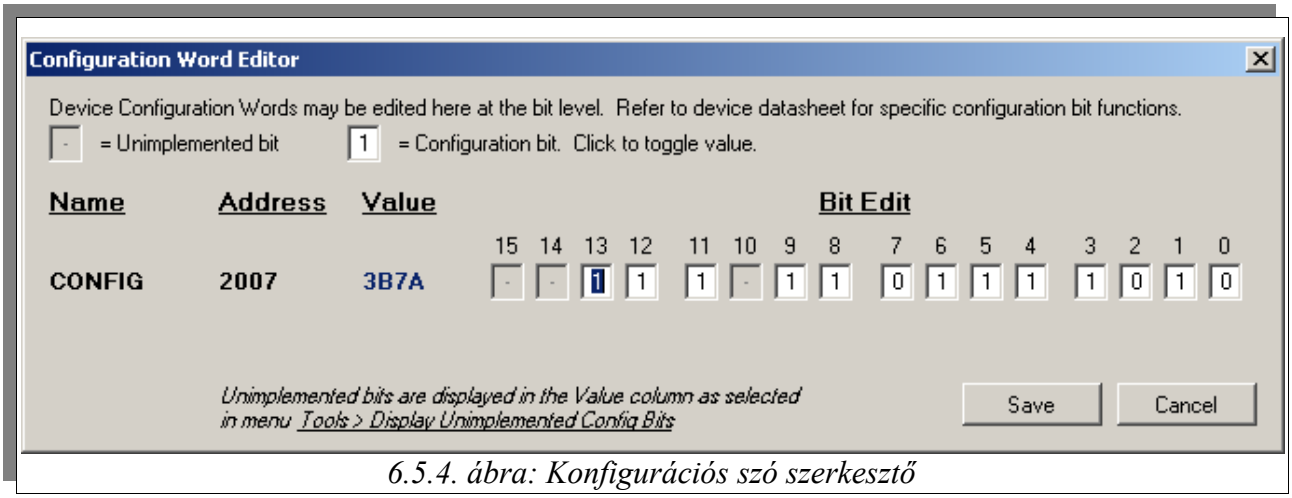
OSCCAL: Oszcillátor kalibráció. A *Tools* → *OSCCAL* menüpont segítségével a 6.5.3. ábrán látható ablakban tudjuk beállítani[81] az OSCCAL regiszter értékét azoknál a PIC mikorvezérlőknél, ahol erre a gyártó lehetőséget ad. A szoftver figyelmeztet minket, hogy ez a beállítás minden adatot töröl a mikrovezérlőről.



6.5.3. ábra: OSCCAL regiszter beállítása

BandGap: Tiltott sáv. A POR[82] és a BOD[83] feszültségszintek láthatóak itt.

Configuration: A kékkel kiemelt *Configuration* szövegre kattintva előtűnik a konfigurációs szó szerkesztő ablaka (6.5.4. ábra). Itt be tudjuk állítani a beégetni kívánt programtól függetlenül is a konfigurációt (az adott bitekre kattintva logikai értékük vált). A *Tools* → *Display Unimplemented Config Bits* menüpontban beállíthatjuk, hogy az adott típusban nem létező biteket miként vegyük figyelembe (logikai 0, logikai 1, vagy a kódból kiolvasott).



6.5.4. ábra: Konfigurációs szó szerkesztő

A *Configuration* szöveg alatt a konfigurációs szótól függően megjelenhet a *Code Protect* (kódvédelem engedélyezve), *Data Protect* (adatvédelem engedélyezve), vagy az *All Protect* (kód és adatvédelem is engedélyezve) piros felirat. Ezen beállításokat a konfigurációs szó módosítása nélkül a menüből is eltudjuk érni (*Tools*).

81 A szoftver lehetőséget nyújt autókalibrációra is, ekkor a gyári beállítás kerül alkalmazásra (a programmemória utolsó rekeszében található érték kerül az OSCCAL regiszterbe).
82 Power On Reset: Bekapcsolási reszet.
83 Brown Out Detect: Alacsony feszültség detektálása.

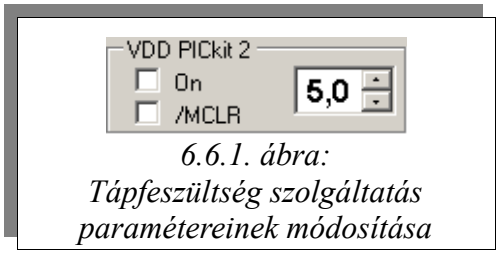
6.6 Tápfeszültség és reset biztosítása a céláramkör számára

A *Tools → Target VDD Source* ablakban kiválaszthatjuk a céláramkör tápfeszültségének forrását:

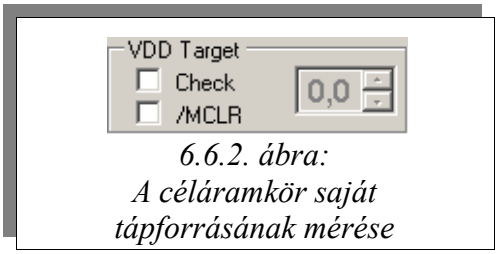
Auto-Detect: A PICkit 2 méri a Target V_{DD} vonalát, ha a céláramkörnek nincs saját tápforrása, akkor a PICkit 2 biztosítja a szükséges teljesítményt a működéshez (max. ~ 80mA).

Force PICkit 2: A tápfeszültséget a PICkit 2 biztosítja. A szolgáltatott feszültség értékét, be/kikapcsolását, valamint a reszetet a VDD ablakban (6.6.1. ábra) tudjuk beállítani.

Force Target: A céláramkör rendelkezik saját tápforrással. A tápforrás értékét a *Check* opció bepipálásával mérhetjük, a */MCLR* opcióval a hardveres reszetfeltételt kapcsolhatjuk be/ki (6.6.2. ábra).



6.6.1. ábra: Tápfeszültség szolgáltatás paramétereinek módosítása



6.6.2. ábra: A céláramkör saját tápforrásának mérése

6.7 Programozás

A céláramkörrel történő műveletvégzésre a 6.7.1. ábrán látható funkciógombok szolgálnak.

6.7.1. ábra: Programozó eszköztár

Read: Céláramkör kiolvasása.

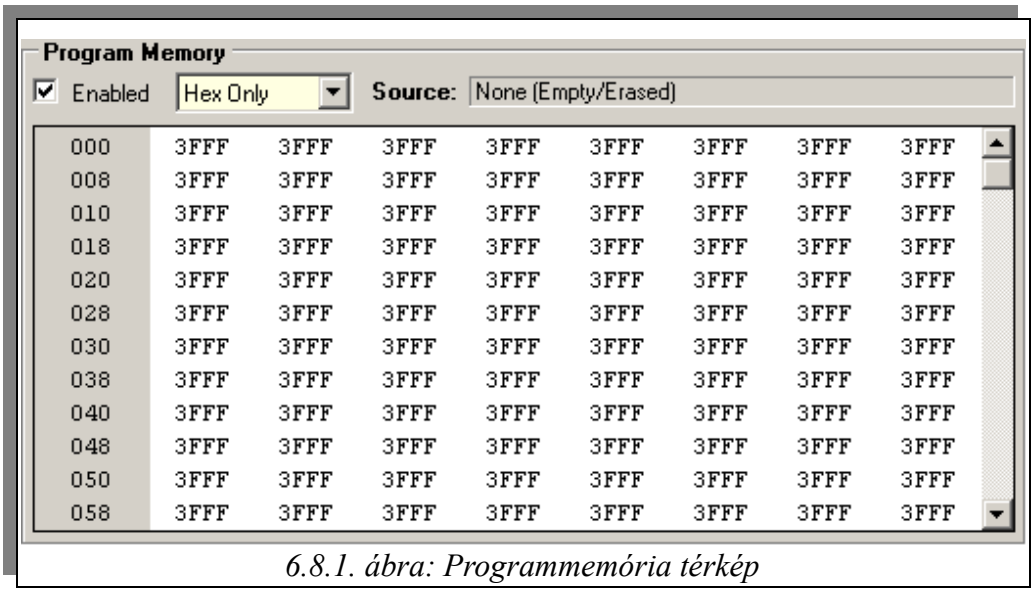
Write: Céláramkör írása.

Verify: Céláramkörben lévő információ összehasonlítása a memóriával.

Erase: Céláramkör törlése.

Blank Check: Céláramkör ellenőrzése, hogy üres-e.

6.8 Program memória



6.8.1. ábra: Programmemória térkép

A *Program Memory* ablakban láthatjuk a beimportált, vagy beolvasott programmemória tartalmát. Szürke háttérrel a kezdőcímeket, míg fehér háttérrel a tartalmat jelöli a szoftver.

Az *Enabled* opció bejelölésével a *Write* parancsra a program memória tartalma (is) letöltésre/kiolvasásra kerül a céláramkörbe/ből.

A legördülő menüből ki tudjuk választani, hogy milyen megjelenítési formát szeretnénk:

- ***Hex Only:*** a memóriatérkép csak a hexadecimális kódokat tartalmazza.
- ***Word ASCII[84]:*** a memóriatérkép a hexadecimális kód mellett megjeleníti a rekeszek tartalmát ASCII formátumban is – a szeparálás szavanként történik.
- ***Byte ASCII:*** a memóriatérkép a hexadecimális kód mellett megjeleníti a rekeszek tartalmát ASCII formátumban is – a szeparálás bájtonként történik.

A Source mezőben láthatjuk a programmemóriában (és az EEPROM adatmemóriában) megjelenített forrás típusát:

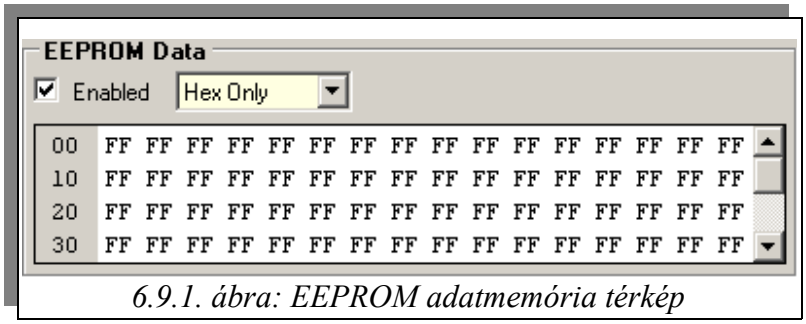
- *None (Empty\Erased):* Nincs, üres, vagy törölt. A programmemória minden rekesze FF értéket tartalmaz, a *Write* parancsnak ugyanaz lesz a hatása, mintha az *Erase* paranccsal törölnénk a céleszközt.
- ***Edited:*** Szerkesztett. A memóriarekeszek tartalmát kézi úton is állíthatjuk. Kettőt kattintva az adott rekeszen, annak tartalma módosítható.
- ***Read from …:*** A megnevezett eszközből kiolvasott kódot tartalmazza a programmemória.
- ***Fájlnév elérési úttal:*** A megadott fájl importálásának eredménye van a

84 American Standard Code for Information Interchange: Amerikai szabvány az információ kódolására (részletekért l. http://hu.wikipedia.org/wiki/ASCII)

programmemóriában.

6.9 EEPROM adatmemória

A 6.9.1. ábrán látható EEPROM adatmemória tartalmát a programmemóriához hasonlóan tudjuk kezelni.



6.9.1. ábra: EEPROM adatmemória térkép

A *EEPROM Data* ablakban láthatjuk a beimportált, vagy beolvasott EEPROM adatmemória tartalmát. Szürke háttérrel a kezdőcímeket, míg fehér háttérrel a tartalmat jelöli a szoftver.

Az *Enabled* opció bejelölésével a *Write* parancsra az EEPROM adatmemória tartalma (is) letöltésre/kiolvasásra kerül a céláramkörbe/ből.

A legördülő menüből ki tudjuk választani, hogy milyen megjelenítési formát szeretnénk:

- ***Hex Only:*** a memóriatérkép csak a hexadecimális kódokat tartalmazza.
- ***Word ASCII[85]:*** a memóriatérkép a hexadecimális kód mellett megjeleníti a rekeszek tartalmát ASCII formátumban is – a szeparálás szavanként történik.
- ***Byte ASCII:*** a memóriatérkép a hexadecimális kód mellett megjeleníti a rekeszek tartalmát ASCII formátumban is – a szeparálás bájtonként történik.

A programmemóriánál bemutatott *Source* mező az EEPROM adatmemória függvénye is.

6.10 Speciális programozás

6.10.1. ábra: Speciális programozás

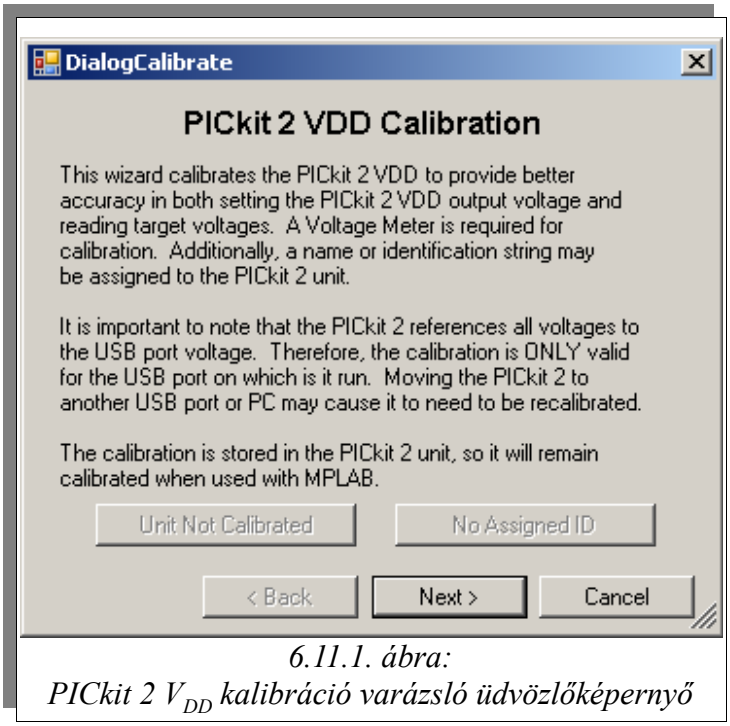
Amennyiben a beimportált fájlt további megerősítések nélkül le szeretnénk tölteni a céláramkörbe válasszuk az *Auto Import Hex + Write Device* opciót. A fájl megnyitása után a programletöltés automatikus. A normál módba való visszatéréshez klikkeljünk ismételten az említett opciógombra. A folyamat visszafelé is lejátszható a *Read Device + Export Hex File* gomb segítségével. Az opció kiválasztásakor az olvasás azonnal megindul, majd megnyílik egy ablak, ahol megadhatjuk a kiexportált fájl nevét és helyét.

85 American Standard Code for Information Interchange: amerikai szabvány kód az információ cseréhez.

6.11 A PICkit 2 által mért V_{DD} feszültség kalibrálása[86]

A PICkit 2 tervezésénél precíziós alkatrészek használata helyett szoftveres kalibrációt alkalmaztak. A V_{DD} és a V_{PP} feszültségeket a belső A/D átalakító méri egy védőellenálláson keresztül.

A *Tools → Calibrate VDD & Set Unit ID...* menüpont segítségével kalibráljuk be a PICkit 2 által szolgáltatott VDD feszültséget!



6.11.1. ábra:
PICkit 2 V_{DD} kalibráció varázsló üdvözlőképernyő

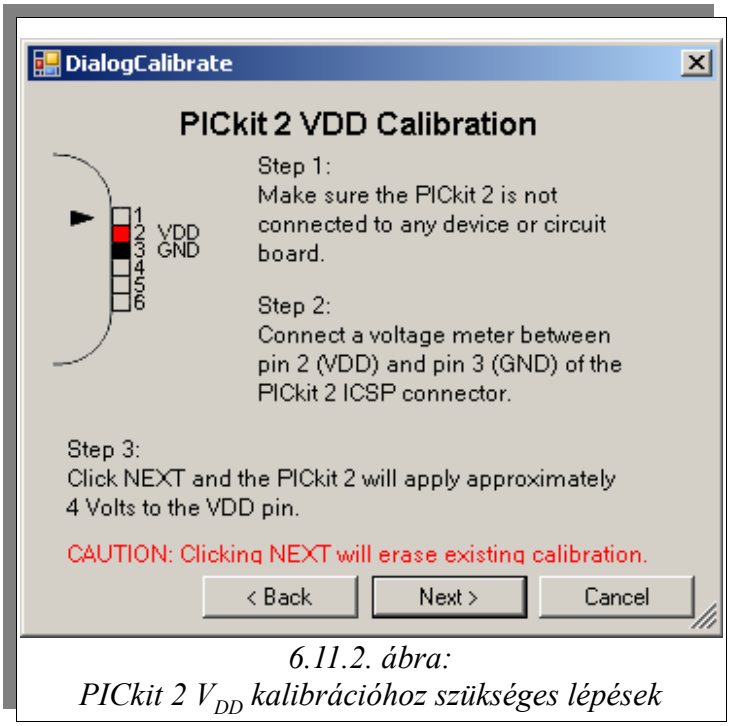
A következő lépésekben egy voltmérő segítségével bekalibrálhatjuk a PICkit 2 által szolgáltatott V_{DD} feszültség értékét, valamint új felhasználói azonosítót adhatunk a PICkit2 programozó és hibakereső eszközünknek[87]. Elképzelhető, hogy a kalibráció során nem tudjuk megfelelően beállítani a V_{DD} feszültséget, mert az általunk használt USB port nem képes biztosítani a szabványban előírt potenciált. Ebben az esetben próbálkozzunk másik porttal. A kalibrációs értéket a PICkit 2 elmenti[88], vagyis újbóli használatkor (pl. MPLAB) is kalibrált állapotban lesz.

86 Az itt ismertetett folyamat valójában felhasználói beszabályozásnak minősül méréstechnikailag. A kalibráció metrológiai definíciója ettől jelentősen eltér. Úgy döntöttem azonban, hogy mivel egyrészt az angol dokumentáció konzekvensen kalibrációnak nevezi a folyamatot, valamint a kalibráció a köznyelvben is ilyen – vagy ehhez nagyon hasonló – folyamatot takar, az elnevezésen jelen dokumentációban nem változtatok.
Kalibráció: Állapotfelvétel a mérőeszköz mérési képességéről. Nem hatósági tevékenység. Az elfogadása bizalmon alapul. A hitelesítéssel szemben lehet részleges jellegű is. Elvégezheti maga a gyártó is, vagy 3. fél ún. kalibráló laboratórium – mely lehet akkreditált és nem akkreditált is (Nemzeti Akkreditáló Testület). A kalibrálás passzív tevékenység – azoknak a műveleteknek az összessége, amelyekkel (meghatározott feltételek mellett) megállapítható az összefüggés a mérőeszköz vagy a mérőrendszer értékmutatása, illetve a mérendő mennyiségnek mértékkel vagy anyagminta által megtestesített, vagy használati etalonnal megvalósított (helyes) értéke között.
Beszabályozás: a mérőeszköz megváltoztatása a metrológiai tulajdonságok javítása érdekében, olyan művelet, amellyel a mérőeszköz használatra kész működési állapotba hozható. A beszabályozás lehet automatikus, fél-automatikus és kézi. A folyamat során az eszköz megváltozik. A felhasználói beszabályozás olyan beszabályozás, ami a felhasználó rendelkezésére álló eszközökkel elvégezhető.

87 Ez különösen akkor érdekes, ha több PICkit 2-őt szeretnénk egy operációs rendszer alatt működtetni (l. 6.15.fejezet).

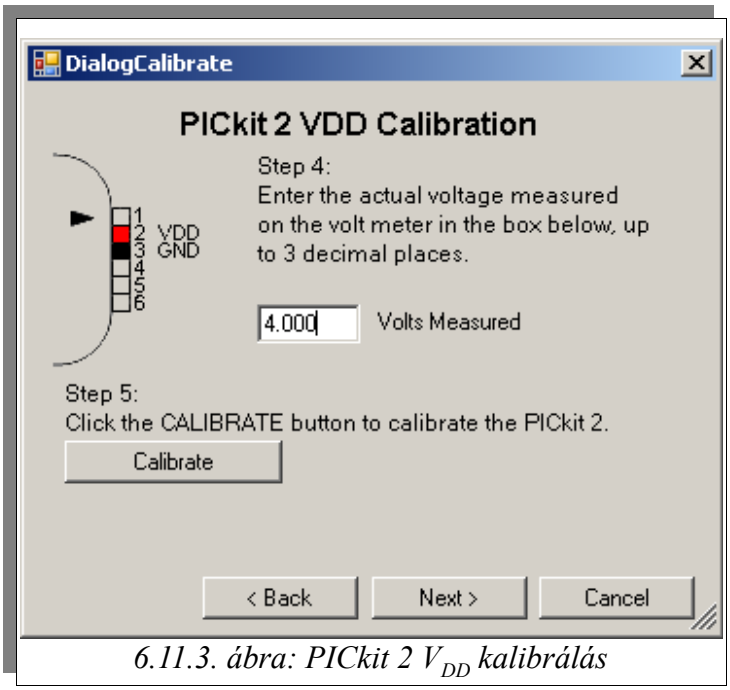
88 A felhasználói azonosítóval egyetemben a belső EEPROM adatmemóriában tárolja.

A kalibrációs folyamat megkezdéséhez válasszuk a *Next>* opciót, a varázslóból való kilépéshez a *Cancel* gombot.



6.11.2. ábra:
PICkit 2 V_{DD} kalibrációhoz szükséges lépések

A kalibrálás első lépéseként a PICkit 2 programozó és hibakereső eszközről távolítsuk el az esetlegesen csatlakoztatott céláramkört. Egy voltmérőt csatlakoztassunk a 6.11.2. ábrán jelzett V_{DD} és GND lábakra. A *Next>* gombra kattintva a PICkit2 4V-os DC feszültségértéket állít elő a V_{DD} lábra a GND-t használva referenciapontként. Vigyázat, a *Next>* gomb megnyomásával az előző kalibrációs eredmény törlésre kerül.

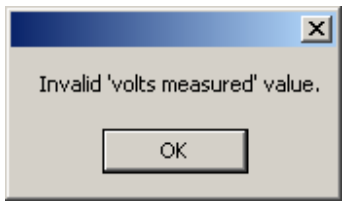


6.11.3. ábra: PICkit 2 V_{DD} kalibrálás

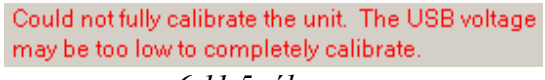
A negyedik lépésben az általunk voltmérővel mért feszültségértéket adjuk meg a *Volts Measured* mezőben. A szoftverben itt sajnos egy hiba van az alapértelmezetten megadott 4.000 értékre a 6.11.3. ábrán látható hibaüzenetet kapjuk. Bár valóban 3 tizedesjegy pontosan adhatjuk meg (nem kötelező, csupán maximum ennyi lehet), a **tizedesjegyeket az egyesektől** nem ponttal, hanem **vesszővel kell kötelezően elválasztani.** A mért érték beírása után kattintsunk a *Calibrate* gombra.

Amennyiben a kalibrálás eredményeként a 6.11.5. ábrán látható üzeneteket kaphjuk, az USB port nem képes megfelelő szintű feszültséget szolgáltatni, próbálkozzunk másik porttal, HUB-bal. A PICkit 2 az USB port által szolgáltatott feszültséget egy védelmi feladatokat ellátó schottky diódán keresztül csatlakoztatja a céláramkörre. Különösen LAPTOP-ok esetén szokott előfordulni, hogy az USB port feszültség sajnos túlságosan is alacsony a megfelelő kalibráció elvégzéséhez. Ha a *CALIBRATION SUCCESFUL!* üzenetet kaptuk a kalibráció sikerült.

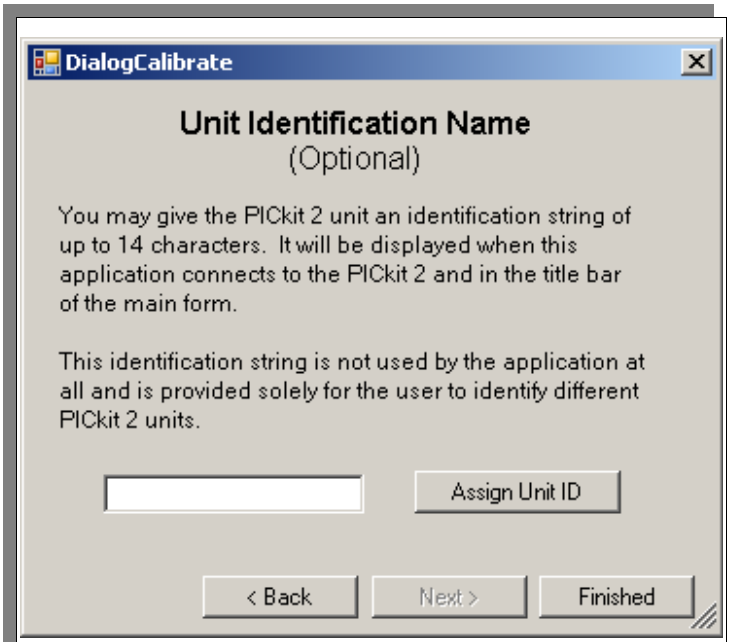
Természetesen a kalibrációs folyamat eredménye csak az adott USB portra vonatkozik, hiszen a PICkit 2 mindig ebből eredezteti a VDD target feszültséget.



6.11.4. ábra:
Érvénytelen formátumú mért érték megadása

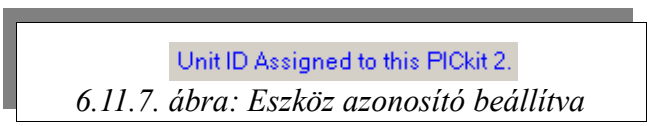


6.11.5. ábra:
Az eszköz nem kalibrálható, mert túl alacsony az USB port által szolgáltatott feszültség



6.11.6. ábra:
PICkit 2 felhasználói azonosító megadása

Az utolsó lépésben fakultatív módon megadhatjuk a PICkit2 felhasználói azonosítóját. A későbbiekben több PICkit2 üzemeltetésekor ezzel hivatkozhatunk rá. Maximum 14 karakter hosszú sztring lehet az azonosító (bár tartalmazhat különleges szimbólumokat – pl. ékezetes karakter – ezek helyén a ? karakter fog megjelenni). Amennyiben változtatni szeretnénk az eredetin a név beírása után kattintsunk az *Assign Unit ID* gombra. A 6.11.7. ábrán látható feliratnak kell megjelennie az ablak alsó részében. A kalibráció befejezéséhez kattintsunk a *Finished* gombra.



6.11.7. ábra: Eszköz azonosító beállítva

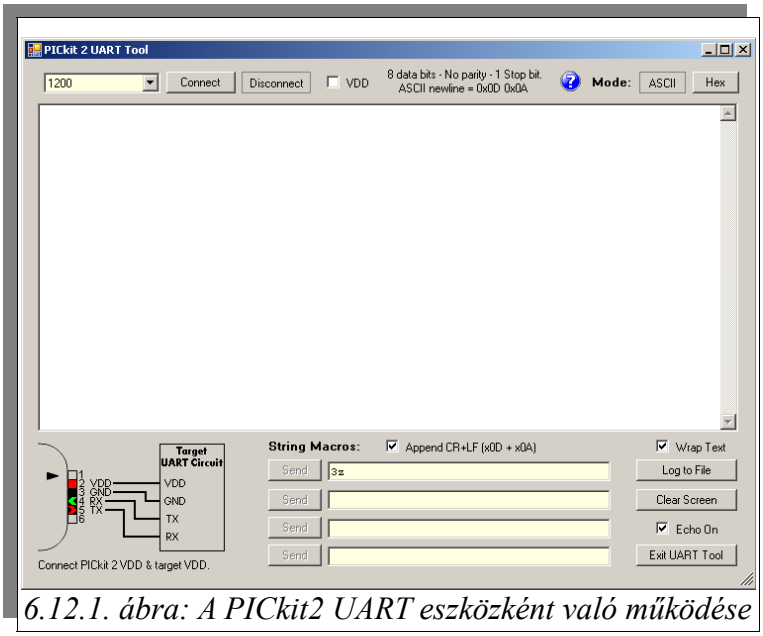
A folyamat befejezése után a PICkit2 operációs rendszere újraindul.

6.12 UART eszközként történő használat

A PICkit2 képes a PGC és PGD lábakat felhasználva egy virtuális UART eszközként viselkedni. Nullmodemes (csak az RX, TX, GND lábakat használjuk, nincs hardveres címzés) kapcsolatot tudunk vele megvalósítani. Ez a funkció hasznos lehet, ha a céláramkörre írt soros porti kommunikációt szeretnénk letesztelni.

Nagyon fontos tisztázni, hogy a PICkit 2 áramkört közvetlenül[89] NEM SZABAD csatlakoztatni egy szabványos RS-232C szerint kommunikáló eszközhöz. Anélkül hogy részletesebben belemennénk a szabvány ismertetésébe az RS-232C kimondja, hogy a vonalon a -3V-nál kisebb értékű feszültségszint MARK (logikai 1), míg a +3V-nál nagyobb értékű feszültségszint SPACE (logikai 0) értéknek felel meg. A PIC mikrovezérlő TTL jelszinttel dolgozik, 5V → logikai 1, 0V → logikai 0.

Válasszuk a *Tools → UART Tool...* menüpontot!

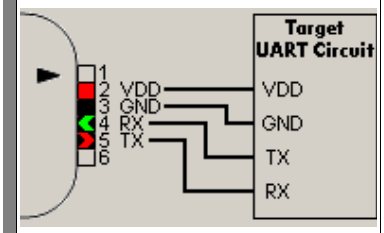


6.12.1. ábra: A PICkit2 UART eszközként való működése

A 6.12.1. ábrát a fogadott és küldött információt megjelenítő fehér háttérrel ellátott ablakon kívül a következő részfelületekre tudjuk bontani:

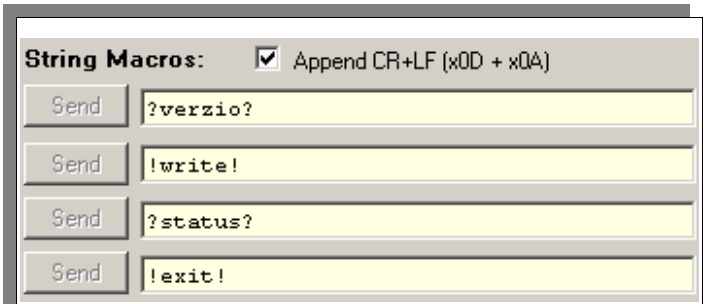
- Hardveregység csatlakoztatása (6.12.2. ábra)
- Üzenetküldés (6.12.3. ábra)
- Funkciógombok (6.12.4. ábra)
- Baud Rate kiválasztása (6.12.5. ábra)
- Csatlakoztatás/leválasztás (6.12.6. ábra)
- Tápfeszültség (6.12.7. ábra)
- Rövid ismertető és súgó lehetőség (6.12.8. ábra)
- Üzemmód (6.12.9. ábra)

89 Létezik diszkrét elemekből, ill. integrált áramkörből (MAX232) felépített konverter.



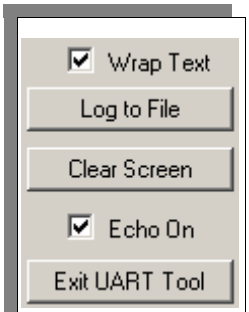
6.12.2. ábra: PICkit 2 csatlakoztatása

A 6.12.2. ábrán láthatjuk, hogy a PICkit 2 áramkört milyen módon tudjuk összekötni a tesztelni kívánt UART áramkörrel. A normál üzemmódban PGD funkciót ellátó RB7-es láb a PICkit2 szempontjából nézve az RX (vétel), míg a PGC funkciót ellátó RB6-os láb TX (adás) láb lesz. A kommunikáció irányát az ábrán látható piros és zöld nyilak is jelzik. A tápfeszültségek összekötése nem feltétel, az RX, TX lábak mellett a GND-k csatlakoztatása a minimális követelmény a kommunikáció megindításához.



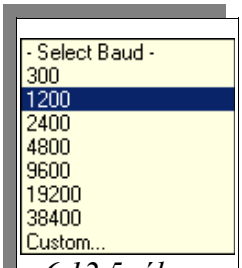
6.12.3. ábra: Elküldhető szövegmezők

A 6.12.3. ábrán látható szövegmezőkbe előre rögzített üzeneteket írhatunk, amiket a Send gomb megnyomásával tudunk elküldeni a céláramkör számára. Az *Append CR+LF (x0D + x0A)* opció bejelölésével a szöveghez a program hozzáfűzi a kocsi vissza (Carriage Return) és a soremelés (Line Feed) karaktereket.



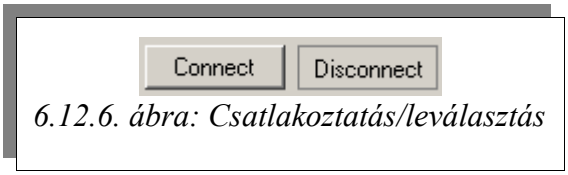
6.12.4. ábra: Funkciógombok

A *Wrap Text* opció a szöveg tördelését teszi lehetővé, amennyiben bejelöljük, akkor a szövegablak oldalirányban nem görgethető az ablak szélét elérve a szoftver sortörést alkalmaz. Az *Echo On* bepipálásának hatására nem csak a vett, hanem a küldött szöveg is megjelenik a szövegablakban. A *Log To File* gomb megnyomásával szövegfájlba tudjuk menteni a kommunikációt. A program zöld háttérrel *Logging Data* jelzésre változtatja a nyomógombot – így jelezve, hogy a kommunikációról jegyzőkönyv készül. Amennyiben le szeretnénk törölni a szövegablakot, nyomjuk meg a *Clear Screen* gombot. Az *Exit UART Tool*-ra kattintva kiléphetünk az UART módból.



6.12.5. ábra: Baud-Rate kiválasztása

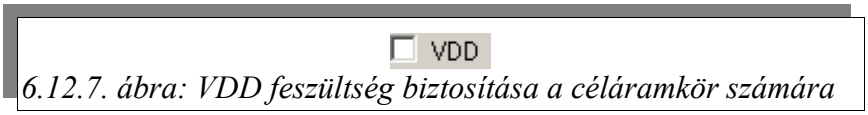
A 6.12.5. ábrán látható legördülő menüből ki tudjuk választani a kommunikáció sebességét (Baud-Rate: az 1s alatt átvitt bitek száma). Amennyiben nem szabványos sebességet szeretnénk beállítani válasszuk a *Custom...* lehetőséget, és adjuk meg a speciális Baud-Rate-et. Amennyiben a tesztelni kívánt eszköz több sebességet is támogat, akkor a kiválasztásnál vegyük figyelembe a külső zavaró tényezőket, és a PICkit 2 erős zavarérzékenységét is.



6.12.6. ábra: Csatlakoztatás/leválasztás

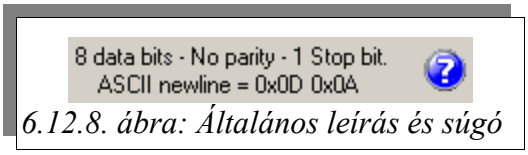
A kommunikációs folyamat megkezdéséhez válasszuk a 6.12.6. ábrán lévő *Connect*, befejezéséhez a *Disconnect* gombot.

Amennyiben a *Tools → Target VDD Source* menüpontban nem a céláramkört választottuk tápforrásként, lehetőségünk nyílik a PICkit2 felől megtáplálni a tesztelni kívánt UART áramkört. Jelöljük be a 6.12.7. ábrán látható opciót a VDD feszültség[90] előállításához.



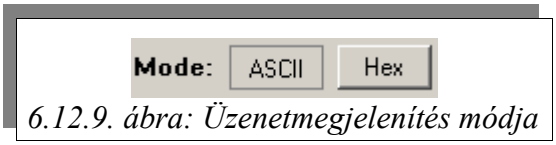
6.12.7. ábra: VDD feszültség biztosítása a céláramkör számára

A 6.12.8. ábrán láthatjuk, hogy paritás nélküli 1 Stop bittel ellátott 8 bites adatcsomagokat tudunk küldeni. Ezen nincs lehetőségünk módosítani. Az újsor (kocsi vissza, soremelés) karakter az ASCII-ból ismert: 0D0A. A kék hátérrel ellátott kérdőjelre kattintva a



6.12.8. ábra: Általános leírás és súgó

A 6.12.9. ábrán láthatjuk az üzenetek megjelenítésének módját, ami lehet ASCII, vagy hexadecimális formátum.

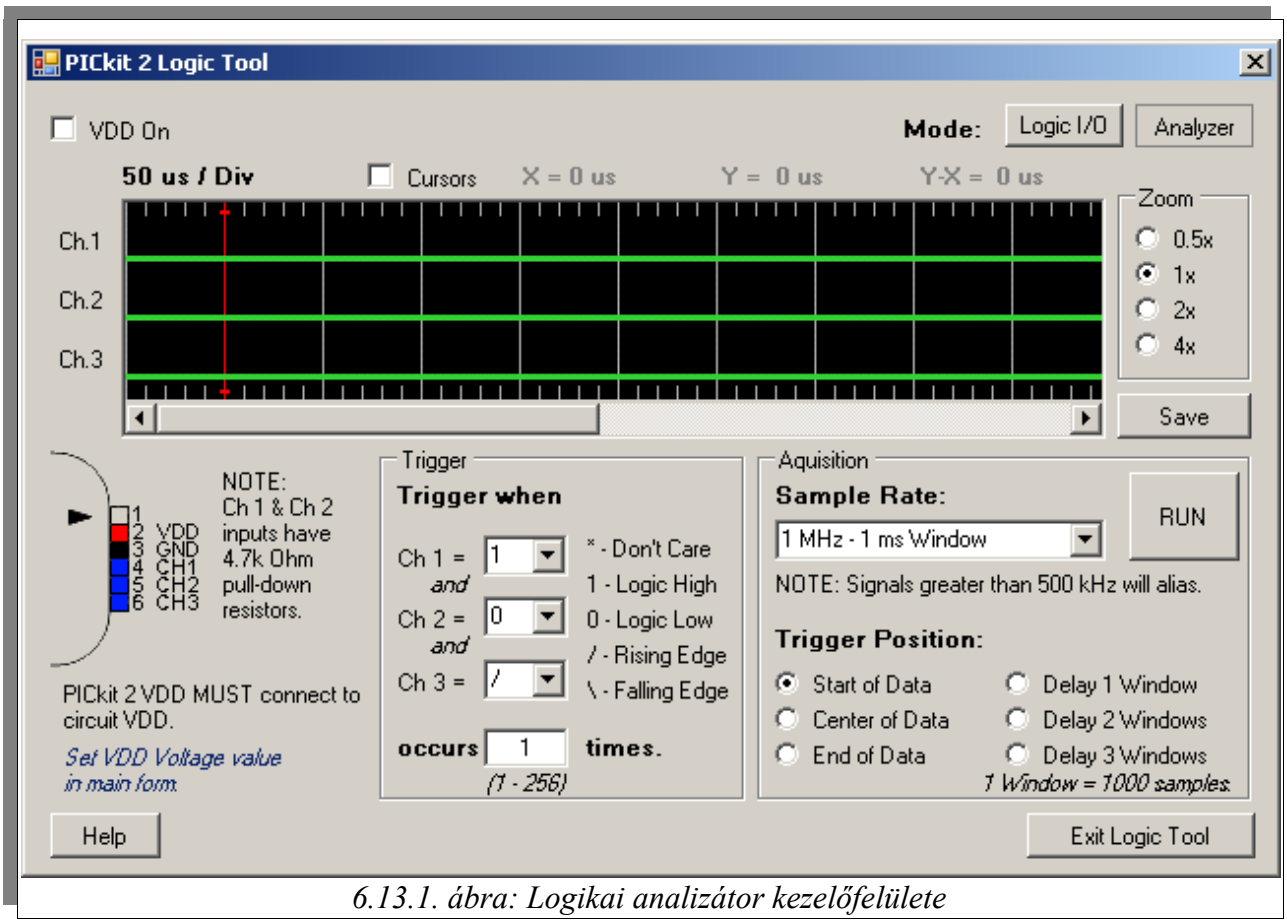


6.12.9. ábra: Üzenetmegjelenítés módja

90 Értékét a PICkit 2 főablakában a 6.6. fejezetben leírtak szerint tudjuk beállítani.

6.13 Logikai analizátorként történő használat

A *Tools → Logic Tool...* menüpont segítségével léphetünk be a PICkit 2 logikai analizátor módjába.

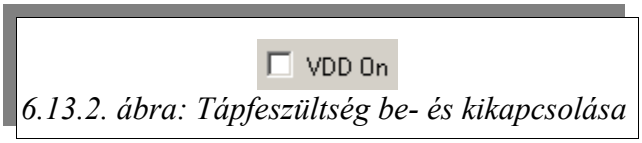


6.13.1. ábra: Logikai analizátor kezelőfelülete

A logikai analizátor felületét az alábbi részekre bonthatjuk:

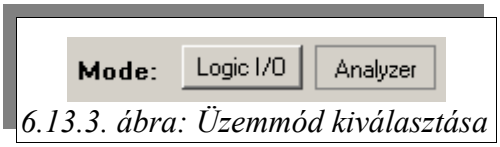
- Tápfeszültség (6.13.2. ábra)
- Üzemmód (6.13.3. ábra)
- Megjelenítés (6.13.4. ábra)
- Közelítés (6.13.5. ábra)
- Funkciógombok (6.13.6. ábra)
- Hardvereszköz csatlakoztatása (6.13.7. ábra)
- Indítási feltételek (6.13.8. ábra)
- Adatgyűjtés (6.13.9. ábra)

A PICkit 2 logikai analizátor üzemmódban is képes tápfeszültséggel[91] ellátni a céláramkört. A bekapcsoláshoz jelöljük be a 6.13.2. ábrán látható opciót. Amennyiben a *Tools → Target VDD Source* menüpontban a céláramkört jelöltük meg tápforrásként, akkor ez az opció nem választható.



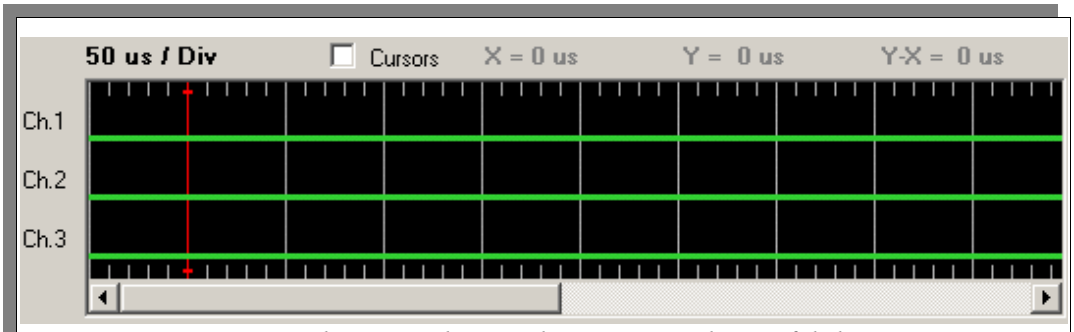
6.13.2. ábra: Tápfeszültség be- és kikapcsolása

A 6.13.3. ábrán látható gombok segítségével tudunk váltani a Logikai I/O egység és az analizátor üzemmód között.

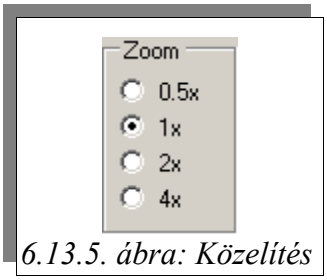


6.13.3. ábra: Üzemmód kiválasztása

A 6.13.4. ábrán láthatjuk a megjelenítő felületet, amelynek segítségével a mérést ki tudjuk értékelni. A csatornákat az angol channel rövidítésből *Ch.1*, *Ch.2*, *Ch.3* névvel jelzi a szoftver. A bal felső sarokban láthatjuk az aktuális időalapot time/DIV (idő/osztás) kijelzéssel. Lehetőségünk van a képernyőn két elhelyezésére (*Cursors* opció bejelölése). Ezeket a program kék (X) ill. rózsaszín (Y) függőleges vonallal jelzi. A mozgatásukhoz használjuk a bal (X) ill. a jobb (Y) egérgombokat. A Cursors felirat mellett láthatjuk az aktuális pozíciójukat. Az abszolút távolságot mindig a piros színű függőleges vonallal jelölt indítási feltételhez képest mérjük. Mindemellett láthatjuk a köztük lévő relatív (Y-X) távolságot is.



6.13.4. ábra: Logikai analizátor megjelenítő felülete

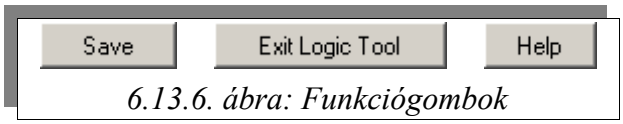


6.13.5. ábra: Közelítés

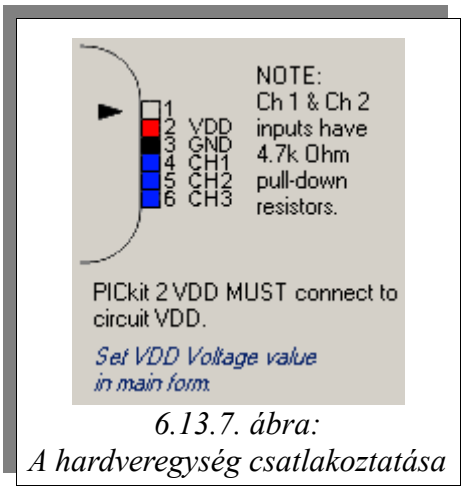
A 6.13.5. ábrán látható *Zoom* ablakban tudunk a megjelenítő képernyőn látható jelalakoktól távolítani (0,5x), vagy rájuk közelíteni (1x; 2x; 4x).

91 Értékét a PICkit 2 főablakában a 6.6. fejezetben leírtak szerint tudjuk beállítani.

A 6.13.6. ábrán látható funkciógombok segítségével előhívhatjuk a súgót (*Help*), elmenthetjük bitmap formátumban az elkészített felvételt (*Save*), vagy visszatérhetünk a PICkit 2 normál üzemmódjába (*Exit Logic Tool*).

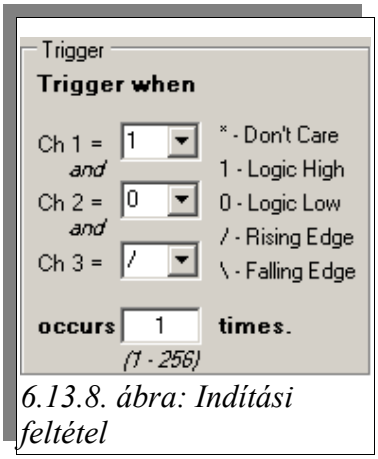


6.13.6. ábra: Funkciógombok



6.13.7. ábra:
A hardveregység csatlakoztatása

A 6.13.7. ábrán láthatjuk, hogy a PICkit 2 mely lábait, milyen csatornaszámoknak feleltethetjük meg. Vegyük figyelembe, hogy az 1-es és a 2-es csatorna (eredetileg PGC, PGD) a PICkit 2-ben található 4,7kΩ-os ellenállásokon keresztül földre van kötve. A GND és VDD lábakat[92] kötelezően csatlakoztatnunk kell az áramkörhöz. A tápfeszültség értékét, és azt, hogy melyik áramköri egység biztosítsa azt a PICkit 2 főablakában tudjuk beállítani (6.6. fejezet).



6.13.8. ábra: Indítási feltétel

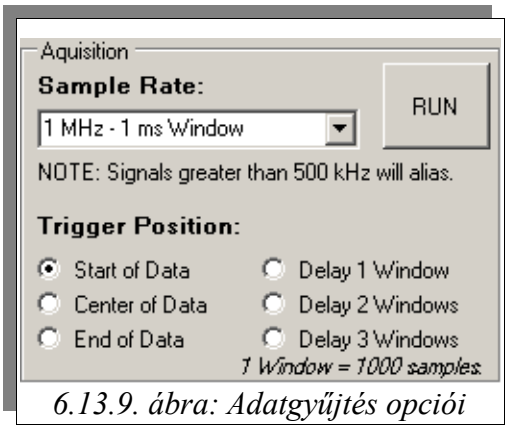
A *Trigger* ablakban be tudjuk állítani, hogy a PICkit 2 milyen feltétel hatására kezdje meg az adatgyűjtést.

Feltételek:

- *: Don't Care – nem meghatározott (bármi)
- 1: Logic High – magas szint
- 0: Logic Low – alacsony szint
- /: Rising Edge – felfutó él
- \: Falling Edge – lefutó él

Az egyes csatornáknál beállított feltételek logikai ÉS kapcsolatban állnak egymással. Vagyis a 6.13.8. ábrán látható beállítás azt jelenti, hogy az egyes csatorna logikai 1, a kettes csatorna logikai nulla állása mellett akkor kezdődik meg az adatgyűjtés, ha a hármas csatornán felfutó él keletkezik. Amennyiben valamelyik csatornát ki szeretnénk szedni a feltétellistából, akkor esetében állítsuk be a * - Don't Care lehetőséget!

92 Ennél a folyamatnál különösen ügyeljünk arra, hogy a PICkit 2, vagy a céláramkör biztosítja a tápfeszültséget, ill. hogy annak mekkora értéknek kell lennie.

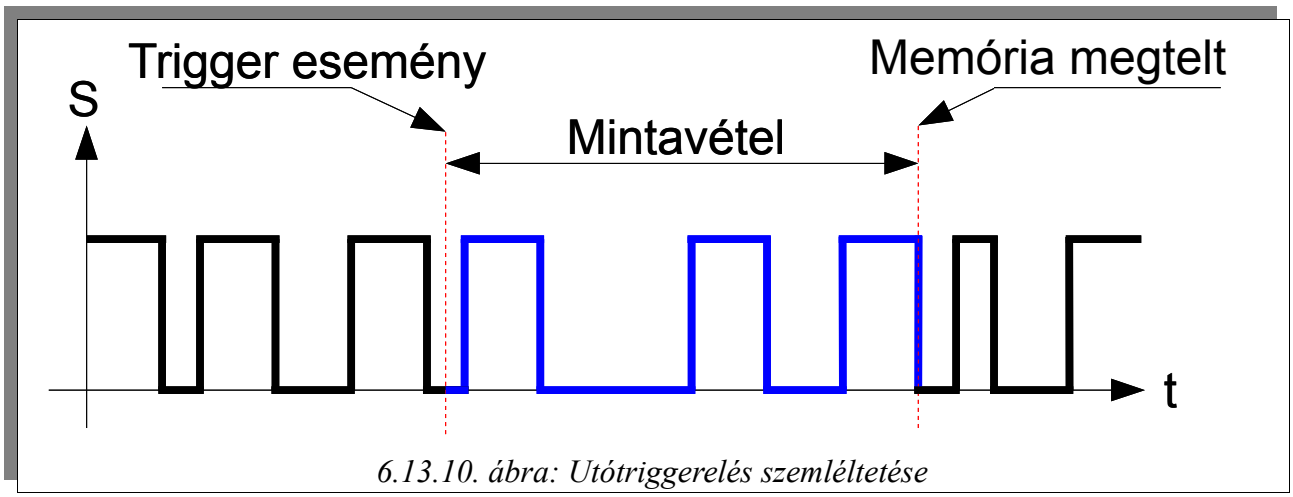


6.13.9. ábra: Adatgyűjtés opciói

Az *Acquisition* ablakban tudjuk beállítani a mintavételezési frekvenciát, valamint a triggerelés módját. A *RUN* gombbal ebből az ablakból indítható az adatgyűjtés folyamata. A *Sample Rate* mezőben a legördülő menüből válasszuk ki a mintavételezési frekvenciát! A szoftver megjegyzi, hogy a mintavételezési frekvencia fele feletti jeleknél kialakulhat az aliasing jelenség[93].

A triggerelési módok egy logikai analizátornál a következők lehetnek:

Utótriggerelés Az adatgyűjtés az indítási feltétel megtörténtekor kezdődik. A leggyakrabban használt indítási forma, segítségével megtudhatjuk, hogy egy előre meghatározott esemény milyen más eseményeket von maga után (6.13.10. ábra).

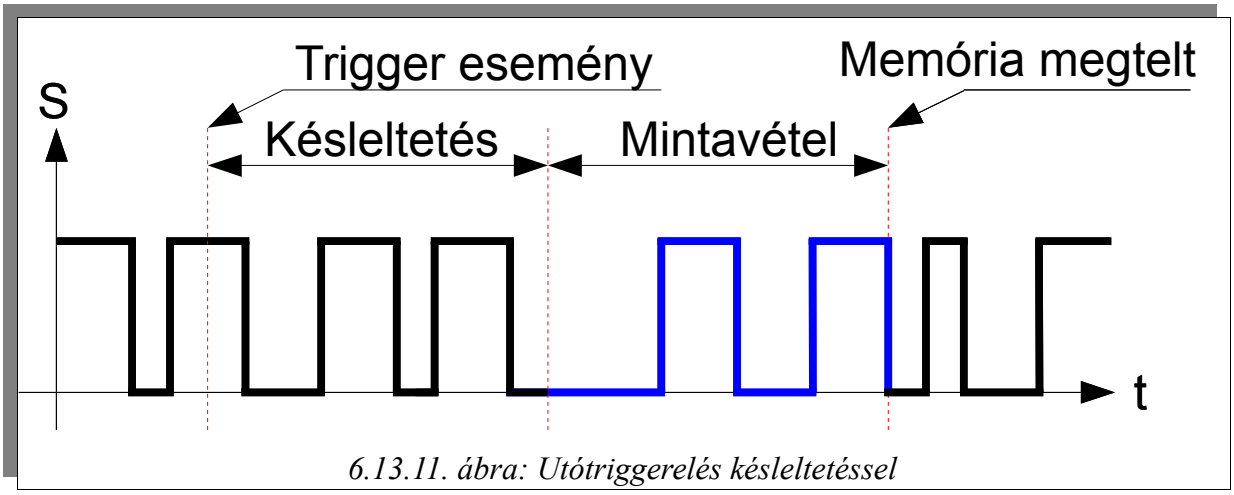


6.13.10. ábra: Utótriggerelés szemléltetése

Utótriggerelés késleltetéssel: Az adatgyűjtés az indítási feltétel megtörténte után egy előre beállított

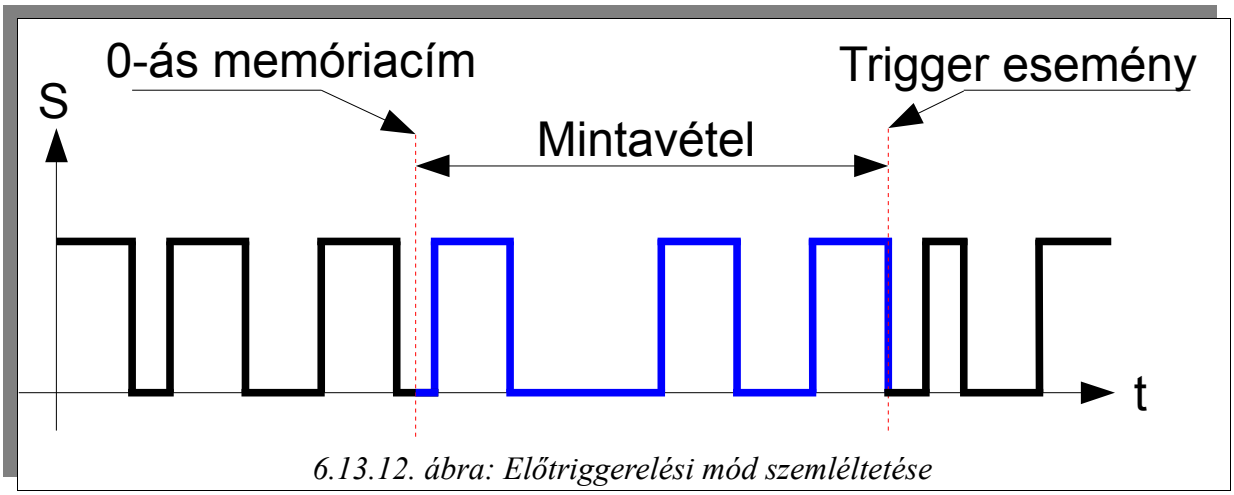
93 Shannon törvénye szerint periodikus jelek esetén minimum kétszeresnek kell lennie (Nyquist-küszöb) a mintavételezési frekvenciának a mintavételezett jelhez viszonyítva. Amennyiben ez a feltétel nem teljesül kialakul az ún. aliasing jelenség (átlapolódás), vagyis olyan jelek is megjelennek a mintavételezési folyamat során, amelyek az eredeti spektrumban nem szerepeltek.

késleltetéssel valósul meg (6.13.11. ábra).



6.13.11. ábra: Utótriggerelés késleltetéssel

Előtriggerelés: Az adatgyűjtés a feltétel előtt folyamatosan zajlik, és a feltétel teljesülésekor áll le. Nagyon hasznos beállítási lehetőség. Amennyiben egy adott hiba bekövetkezett, megtudhatjuk, hogy milyen események vezettek el ideáig (6.13.12. ábra).

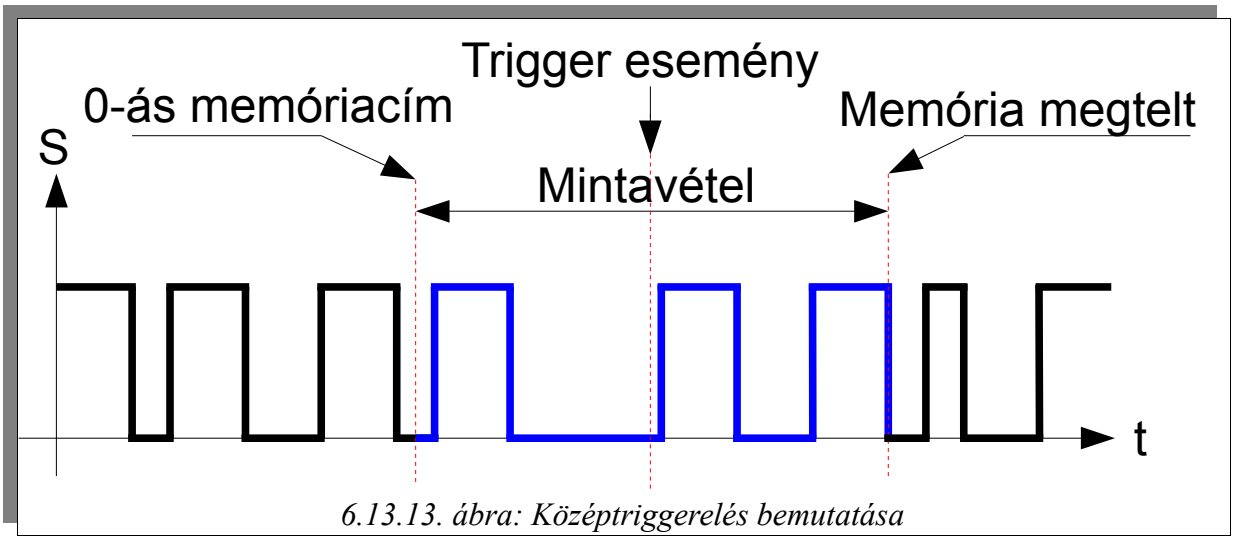


6.13.12. ábra: Előtriggerelési mód szemléltetése

Előtriggerelés késleltetéssel: Az adatgyűjtés a feltétel előtt folyamatosan zajlik, és a feltétel teljesülése után egy előre beállított késleltetéssel később áll le.

Középtriggerelés: Az adatgyűjtés a feltétel előtt és után is zajlik, úgy, hogy a triggeresemény álljon a tárolható események középpontjában.

Alapvetően az elő- és utótriggerelés jó tulajdonságait ötvözi. Lehetőségünk van egy esemény előzményeit, és következményeit is egy felvételen megtekinteni (6.13.13. ábra).



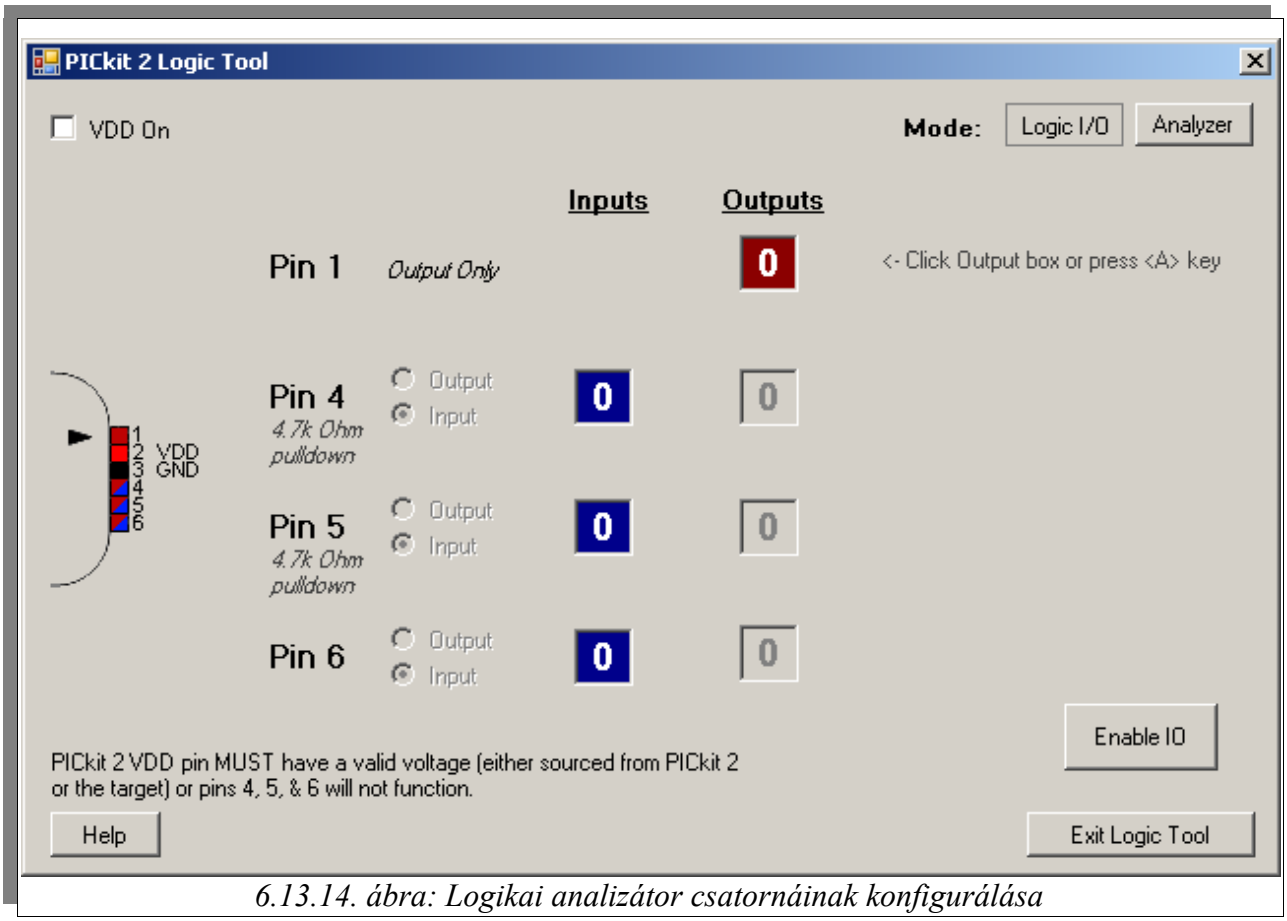
6.13.13. ábra: Középtriggerelés bemutatása

Középtriggerelés késleltetéssel: A középtriggerelésnél leírtak szerint zajlik a folyamat, de beállíthatunk egy késleltetést ezzel eltolva a középpontot az esemény halmaz időbeli vége felé.

A triggerelési módok közül a PICkit 2 az alábbi lehetőségeket támogatja:

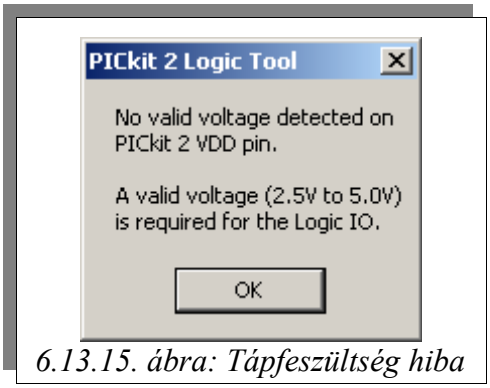
- Utótriggerelés – *Start of Data*
- Utótriggerelés késleltetéssel – *Delay Windows* (1 ablak 1000 mintányi késleltetést jelent)
- Előtriggerelés – *End of Data*
- Középtriggerelés – *Center of Data*

A logikai analizátor üzemeltetése mellett lehetőségünk van egy egyszerű I/O egység működtetésére. Az átváltást a különböző üzemmódok között a 6.13.3. ábrán látható gombok segítségével tudjuk megvalósítani.



6.13.14. ábra: Logikai analizátor csatornáinak konfigurálása

Kattintsunk az *Enable IO* gombra a logikai egység üzemeltetésének megkezdéséhez. A működéshez szükségünk van tápfeszültségre, ennek hiányában a 6.13.15. ábrán látható hibaüzenetet kapjuk.



6.13.15. ábra: Tápfeszültség hiba

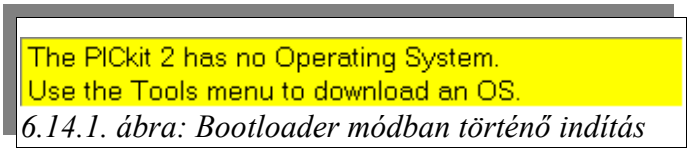
Amennyiben a céláramkör nem rendelkezik saját tápforrással jelöljük be a V_{DD} *On* opciót, így a PICkit 2 látja el tápfeszültséggel az áramkört. Figyelem, az áramkörnek megfelelő

feszültségértéket a PICkit 2 főablakában tudjuk beállítani (6.6. fejezet).

A *Help* gombra kattintva megnyílik a Logic Tool User Guide – ami segítséget nyújt a logikai analizátor és az I/O egység használatához.

6.14 Firmware update

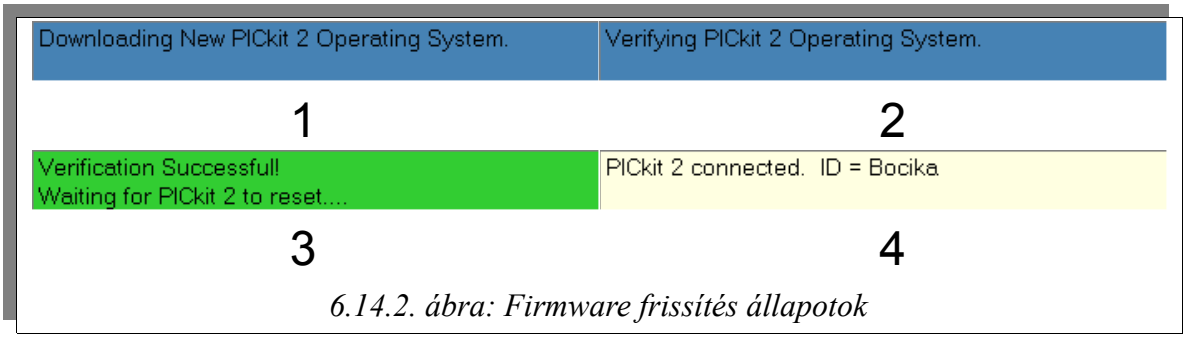
Tartsuk lenyomva a PICkit 2 hardveregységen lévő nyomógombot, és így csatlakoztassuk a számítógéphez. A piros Busy jelzésű LED-nek villognia kell, ezzel jelezve hogy a PICkit 2-őt bootloader módban indítottuk el. A PICkit 2 szoftverének indításakor a 6.14.1. ábrán üzenetet kell kapnunk[94].



6.14.1. ábra: Bootloader módban történő indítás

A *Tools → Download Pickit 2 Operating System* menüpontot választva adjuk meg a letölteni kívánt firmware helyét. Amennyiben a fájl neve nem egyezik meg a szabványos Pickit2 OS formátummal válasszuk a keresés során a minden fájltípus mutatását (All Files) .

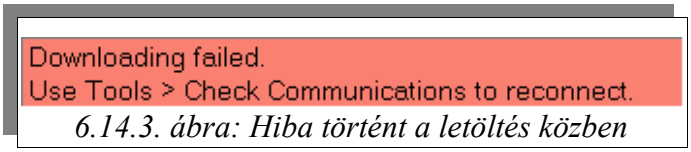
A kiválasztás után a 6.14.2. ábrán látható jelentéseknek kell megjelenniük a státuszablakban.



6.14.2. ábra: Firmware frissítés állapotok

Lehetőségünk van normál módban is firmware-t frissíteni a *Tools → Download Pickit 2 Operating System* menüpont segítségével[95], ill. automatikusan frissíthetünk az MPLAB IDE rendszerrel is.

A 6.14.3. ábrán látható hibaüzenetet kapva a *Tools → Check Communication* menüpont segítségével próbáljuk újra csatlakoztatni a PICkit 2 áramkört és ismételjük meg a folyamatot.

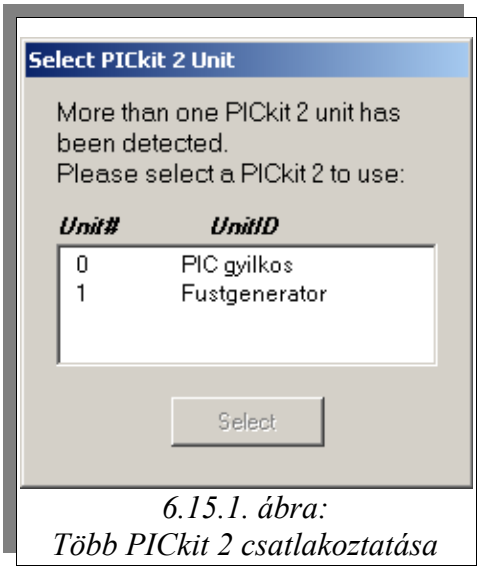


6.14.3. ábra: Hiba történt a letöltés közben

94 Amennyiben véletlenül indítottuk bootloader módban a PICkit 2-őt válasszuk le és csatlakoztassuk újra a nyomógomb nyomva tartása nélkül.

95 Ilyenkor a PICkit 2 0-át ír be a 0x7FFE címre, majd reszetel. A firmware induláskor ellenőrzi, hogy a hardveren található gomb lenyomva volt-e, vagy a 7FFE címen 0 van e. Bármely feltétel igaz volta magával vonzza a bootloader módba történő belépést.

6.15 Több PICkit2 használata egy platform alatt



6.15.1. ábra:
Több PICkit 2 csatlakoztatása

Amennyiben több PICkit 2 áramkört szeretnénk használni egy operációs rendszer alatt, akkor első lépésként a 6.11. fejezetben látható módon lássuk el őket különböző azonosítókkal. Csatlakoztatva két, vagy több eszközt a programozószoftver indításakor a 6.15.1. ábrán látható választás előtt állunk. Válasszuk ki a használni kívánt eszközt, majd kattintsunk a *Select* gombra.

Ha a programozási folyamat során bármikor váltani szeretnénk az eszközök között, válasszuk a *Tools → Check Communication* menüpontot, és a 6.15.1. ábrán felbukkanó ablakban adjuk meg az aktuális PICkit 2 felhasználói azonosítóját.

Az MPLAB IDE rendszerben is használhatunk több PICkit 2 egységet, azonban ott az átváltások során mindig programozóeszközt kell váltanunk (*Programmer → Select Programmer* menüpontban[96])

96 Az átváltáshoz szükséges ablak előhívásához válasszuk a None, majd a PICkit2 opciókat.

7 Melléklet

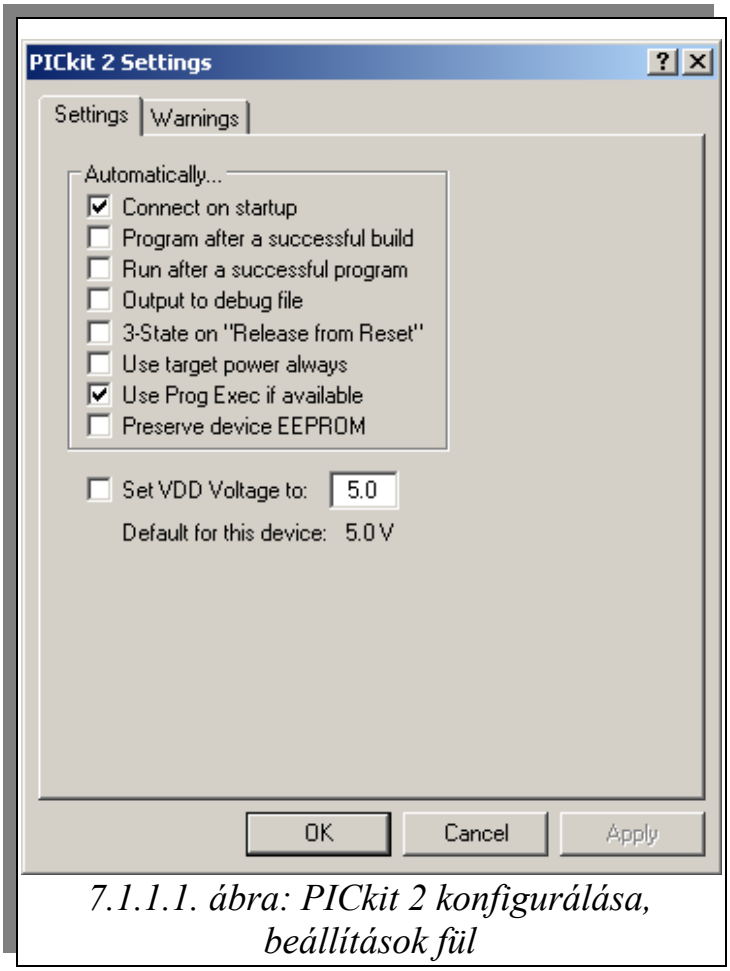
7.1 PICkit2 debugger és az MPLAB IDE fejlesztőkörnyezet együttműködése

Az MPLAB IDE programról – az interneten történő rövid keresés után – részletes angol, és magyar leírást találunk, ezért itt csak a legfontosabb funkciókra térünk ki a PICkit 2 kapcsán.

7.1.1 Programozóként történő használat

Gyakran előfordul, hogy csupán a program beégetésére van szükségünk, és a PICkit 2 nyomkövető funkcióját nem szeretnénk kihasználni – ebben az esetben használhatjuk a hozzá tartozó szoftvert, amit már a korábbiakban bemutattunk, vagy az MPLAB IDE fejlesztőkörnyezetet.

Válasszuk ki programozóként[97] a PICkit 2 hardveregységet – majd a *Programmer → Settings*[98] menüpont segítségével állítsuk be a PICkit 2 tulajdonságait.



7.1.1.1. ábra: PICkit 2 konfigurálása, beállítások fül

97 *Programmer → Select Programmer → PICkit 2*
98 *Programozó → Beállítások*

Automatically...: A bejelölt funkciók automatikus végrehajtása.

Connect on startup: Csatlakoztatás bekapcsoláskor.

Program after a successful build: Sikeres fordítás esetén programozás (a céleszközbe történő égetés).

Run after a successful program: Sikeres programozás után futtatás.

Output to debug file: Kimeneti debug fájl generálása a program futtatásáról.

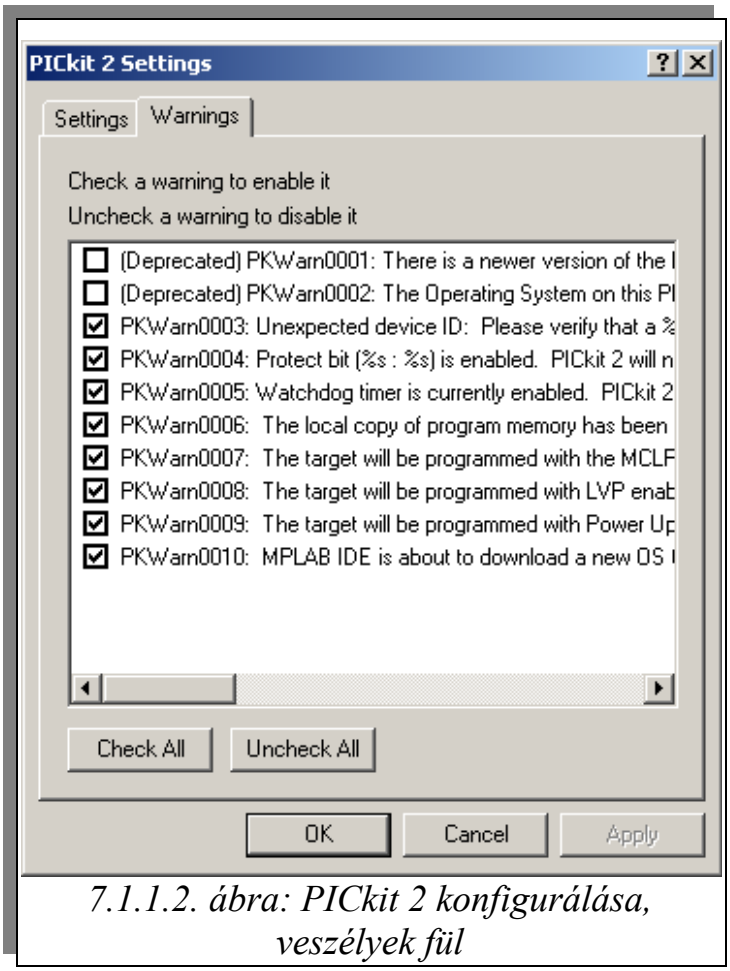
3-State on „Release from Reset”: A *Release from Reset* opció hatására az $\overline{MCLR}$ lábat tri-state állapotba húzza.

Use target power always: Mindig a céláramkör adja a tápfeszültséget.

Use Prog Exec if available: 16 bites vezérlők (24-es, ill. dsPIC-ek) esetén alkalmazható beállítás, a Programming Executive (PE) mód lerövidíti a programozási időt.

Preserve device EEPROM: Az eszköz EEPROM memóriatartalmát megőrzi.

Set VDD Voltage to: V_{DD} tápfeszültség beállítása (alapértelmezetten a *Configure* → *Settings*-ben megadott típus határozza meg).



7.1.1.2. ábra: PICkit 2 konfigurálása, veszélyek fül

Melléklet

A 7.1.1.2. ábrán látható ablakban jelöljük be azokat a veszélyeket, amelyekről értesítést kívánunk kapni, ill. a feleslegeseket pipáljuk ki. Mindemellett lehetőségünk van az összes veszély együttes bejelölésére (*Check All*), valamint elhagyására (*Uncheck All*).

(***Deprecated***[99]***) PKWarn0001:*** There is a newer version of the PICkit 2 Operating System.

(Deprecated) PKWarn0002: The Operating System on this PICkit 2 is newer than the one intended for use with the current version of MPLAB IDE.

PKWarn0003: Unexpected device ID: Please verify that a %s is correctly installed in the application. (Expexted ID= 0x%X, ID Read = 0x%X).

PKWarn0004: Protect bit (%s: %s) is enabled. PICkit 2 will not enter debug mode in this configuration. Do you wish to disable this settings?

PKWarn0005: Watchdog timer is currently enabled. PICkit 2 cannot be used while the watchdog timer is active. Disable watchdog timer?

PKWarn0006: The local copy of program memory has been changed since the last program operation. Should PICkit 2 program the target (fix) before proceeding?

PKWarn0007: The target will be programmed with the MCLR pin disabled. PICkit 2 cannot debug the target in the state. Would you like to enable the MCLR pin?

PKWarn0008: The target will be programmed with LVP enabled. PICkit 2 cannot debug the target in the state. Would you like to disable LVP?

PKWarn0009: The target will be programmed with Power Up Timer enabled. PICkit 2 cannot debug the target in the state. Would you like to disable the PUT?

PKWarn0010: MPLAB IDE is about to download a new OS to the PICkit2. Please disconnect PICkit 2 from any device or target board before continuing. If more than 1 PICkit 2 unit is connected to the PC select „Cancel”, remove all units except the o:

PKWarn0001: A szoftver a jelenleginél újabb verziójú firmware-t tartalmaz. A frissítés indokolt.

99 Azon veszélyforrások mellett, melyeket nem jelöltük meg megtaláljuk a deprecated-érvénytelenített jelzőt.

PKWarn0002: A PICkit2 firmware-e újabb, mint a jelenleg használatos MPLAB IDE rendszer.

PKWarn0003: Váratlan eszközazonosító: kérlek ellenőrizd, hogy a %s helyes-e az adott alkalmazásban! (Várt azonosító: 0x%X, Olvasott azonosító: 0x %X)

PKWarn0004: Védelmi (kód, adat, stb.) bit engedélyezve. A PICkit2 nem tud belépni a hibakereső módba az adott konfigurációban. Kikapcsolod a védelmet?

PKWarn0005: A Watch Dog Timer jelenleg engedélyezett. A PICkit2 nem alkalmazható ha a WDT aktív! Kikapcsolod?

PKWarn0006: A programmemóriában változás történt a legutóbbi művelet végrehajtása óta. Beégessük az új információt a céláramkörbe, mielőtt az aktuális műveletet végrehajtjuk?

PKWarn0007: A céláramkörben az $\overline{MCLR}$ láb ki van kapcsolva (bemenetként konfigurálva). Ilyen beállítás mellett a PICkit2 nem tud belépni a hibakereső módba! Engedélyezi az $\overline{MCLR}$ funkciót?

PKWarn0008: Alacsony feszültségű programozási mód engedélyezve. A PICkit2 így nem használható hibakeresőként. Kikapcsolod az LVP-t?

PKWarn0009: A céláramkörben a PWRT engedélyezve van. Ilyen beállítás mellett a PICkit2 nem tud belépni a hibakereső módba. Kikapcsolod a bekapcsolási reszet időzítőjét?

PKWarn0010: Az MPLAB IDE új operációs rendszert tölt le a PICkit2-be. Kérlek, minden eszközt, céláramkört válassz le a PICkit2-ről. Amennyiben egyszerre több PICkit2-őt használsz, válaszd a *Mégse* opciót, és válaszd le az összes többi egységet a PC-ről.

Tegyük aktívvá a beégetni kívánt forrást[100], majd válasszuk ki a *Configure → Select Device...*[101] menüpont segítségével a mikrovezérlőnket – amennyiben szükséges állítsuk be a *Configure → Configuration Bits...* menüpontban a konfigurációs biteket. A programozást a *Programmer*[102] (7.1.1.1. fejezet) menüpont, ill. a *Programmer* eszköztár (7.1.1.2. fejezet)

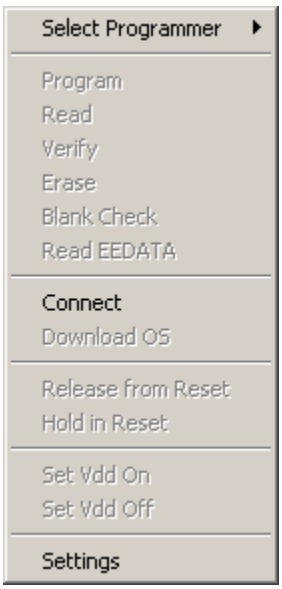
100 Ha csak a hex fájl áll rendelkezésünkre, akkor válasszuk a *Project → Set Active Project* menüpontnál a *None (Quickbuild)* opciót és importáljuk be a gépi kódú fájlt (*File → Import...*). Ha projekt rendszerben dolgozunk a *Project → Set Active Project* menüpontnál válasszuk ki a projektünket (szükség esetén nyissuk meg) majd fordítsuk le a forrásfájlt (*Project → Build*).

101 *Konfiguráció → Eszköz* kiválasztás

102 Programozó

segítségével tudjuk elvégezni.

7.1.1.1 Programozó menü



7.1.1.1.1. ábra: Programozó menü

Select Programmer: Programozó hardveregység kiválasztása.

Program: Az aktuális forrás céleszközbe történő beégetése. (Program memória, EEPROM adatmemória, konfigurációs bitek, azonosító.)

Read: Targetből való kiolvasás. (Program memória, EEPROM adatmemória, konfigurációs bitek, azonosító.)

Verify: Ellenőrzés – az aktív forrás és a céleszközben történő program összehasonlítása. (Program memória, EEPROM adatmemória, konfigurációs bitek, azonosító.)

Erase: Céleszköz törlése. (Program memória, EEPROM adatmemória, konfigurációs bitek, azonosító.)

Blank Check: Ellenőrzi, hogy a céleszköz üres-e.

Read EEDATA: EEPROM adatmemória kiolvasása a mikrovezérlőből.

Connect: Csatlakozás a PICkit 2 hardveregységhez.

Download OS: Az operációs rendszer letöltése.

Release from Reset: A reszetfeltétel megszüntetése[103].

Hold in Reset: Reszetben tartás[104].

Set Vdd On: Target_V_{DD} bekapcsolása (csak akkor aktív, ha a PICkit2 adja a tápfeszültséget).

Set Vdd Off: Target_V_{DD} kikapcsolása (csak akkor aktív, ha a PICkit2 adja a tápfeszültséget).

Settings: Beállítások (7.1.1. fejezet)

7.1.1.2 Programozó eszköztár

7.1.1.2.1. ábra: Programozó eszköztár

103 $\overline{MCLR}$ láb nagyimpedanciás (Hi-Z) állapotba vitele.

104 $\overline{MCLR}$ láb nullában tartása.

Az eszköztár[105] funkciói balról jobbra[106]

Program the target device: Az aktuális forrás beégetése a céleszközbe.

Read target device memories: A mikrovezérlő memóriájának kiolvasása.

Read the target EEDATA memory: EEPROM adatmemória kiolvasása a mikrovezérlőből.

Verify the contents of the target device: Az eszköz tartalma és az aktuális forrás összehasonlítása.

Erase the target device memories: A céleszköz memóriájának törlése.

Verify that target memories are erased: A céleszköz törölt voltának ellenőrzése.

7.1.2 Debuggerként történő használat

7.1.2.1 Korlátozások a hibakeresés során

A PICkit2 egy debugger és nem emulátor, így – az anyagi előnyök mellett – meg kell barátkoznunk néhány megkötéssel a használata során.

Az $\overline{MCLR}$, RB_6, RB_7 lábak nem használhatóak általános I/O-ként, vagy a rájuk multiplexált perifériaként, mert fent vannak tartva a hibakeresés során történő kommunikációs eljárás számára.

A töréspontok komplexitása és száma korlátozott (az adott típushoz tartozó értéket a *Debugger → Breakpoints...* menüpont alatt az *Active Breakpoint Limit* mezőben találjuk).

A hibakereső üzemmódban történő használathoz tanulmányozzuk a http://plc.mechatronika.hu weboldalon található MPLAB leírást!

105 A *View → Toolbars → PICkit 2 toolbar* opció segítségével tudjuk be- és kikapcsolni az eszköztárat.

106 Az MPLAB IDE fejlesztőkörnyezet alkalmazza az ún. mouseover lehetőséget, vagyis ha az egeret az ikon fölé visszük, látszódik a funkciója.

7.2 ICSP

7.2.1 Az ICSP fogalmának tisztázása

Az ICSP mozaikszó jelentése: In Circuit Serial Program – áramkörben történő soros programozás. A helyben történő programozás (In-System Programming – ISP) nem újdonság a programozható alkatrészgyártók körében. Legismertebb képviselője a Joint Test Action Group (JTAG[107]) protokoll, melyet az elektronikus eszközök peremfigyelésére fejlesztettek ki. Az áramkörben történő programozás előnyei:

- Csökkenő költségek (nem kell külön foglalat a programozóban, ill. az áramkörben)
- Csökkenő programozási idő (nem kell az integrált áramkört kiszedegetni)
- Kényelmesebb fejlesztés (a különböző frissítésekhez nem kell az áramkört megbontani)
- Csökkenő mechanikai igénybevétel
- Peremfigyelés (A Boundary Scan technológia segítségével 4, esetleg 5 vezetéken keresztül tudjuk tesztelni az elektronikus eszközöket, ill. felprogramozni az erre alkalmas elemeket.)
- Nincs más lehetőség[108]

A Microchip cég a mikrovezérlőihez kifejlesztett ISP technikát ICSP-nek nevezi.

107 Később az IEEE 1149.1 nemzetközi szabványként fogadták el Standard Test Access Port and Boundary-Scan Architecture néven

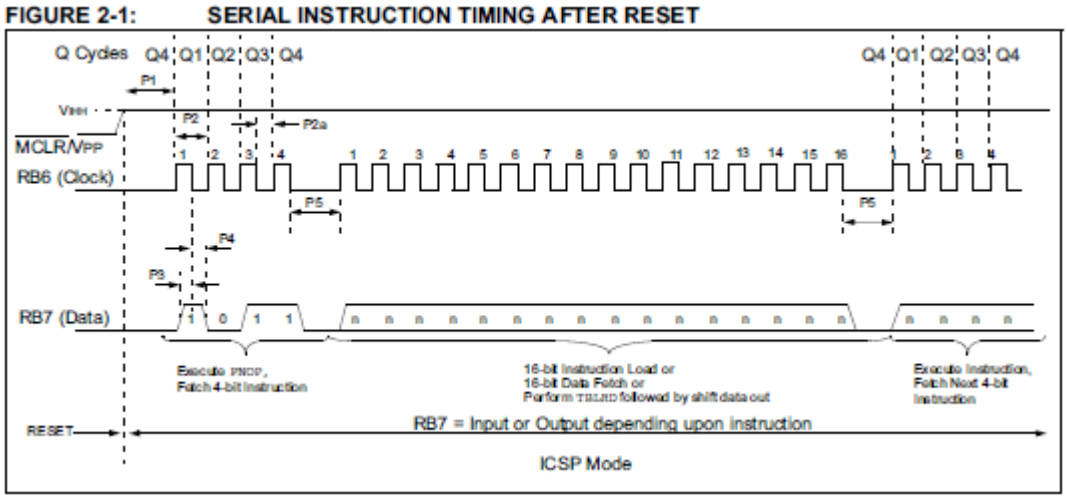
108 Képzeljünk el egy 256 lábú BGA tokozású integrált áramkört amelyet többször is fel kell programoznunk ill. tesztelnünk a fejlesztés során.

7.2.2 Az ICSP működése

Az ICSP technika során 5 kivezetést használunk a mikrovezérlő felprogramozására:

- **V_{DD}:** A mikrovezérlő pozitív tápfeszültség kivezetése.
- **V_{SS}:** A mikrovezérlő földpontja (GND).
- **$\overline{\text{MCLR}}$:** Master CleaR láb, amellyel a hardveres reszet mellett speciális programozómódba tudjuk juttatni a mikrovezérlőt (V_{PP} láb).
- **PGC:** ProGramming Clock láb (RB_6), órajelvonal (ICSPCLK).
- **PGD:** ProGramming Data (RB_7), adatvonal (ICSPDAT).

Az ICSP során először ki kell választanunk, hogy az égető áramkör (HOST), vagy pedig a céláramkör (TARGET) biztosítsa e a tápellátást. Amennyiben a TARGET adja a V_{DD} feszültséget, akkor aktív égető[109] esetén nincs szükség a V_{DD} vonalak összekötésére. Az $\overline{\text{MCLR}}$ láb magas szintbe billentése után kezdődik meg a kommunikáció a kétirányú PGD lábon, míg a PGC lábon a HOST biztosítja a kommunikációhoz szükséges órajelet (7.2.2.1. ábra). Az égetőáramkörtől függően a PGD és PGC lábakon történő kommunikáció sebessége állítható (pl. PICkit 2), ill. ezen lábak tartalmaznak puffert (pl. PICkit 3) a nagyobb távolságra történő kommunikáció miatt.

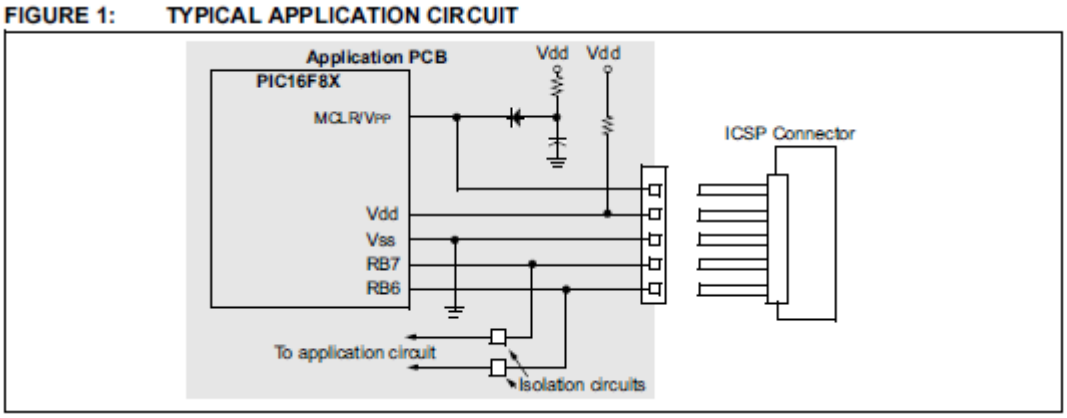


7.2.2.1. ábra: Az áramkörben történő programozás során használt lábak idődiagramja

109 Az égető áramkörök általában rendelkeznek saját tápfeszültséggel, bár elképzelhető olyan kivitel is, amely a tápforrás szempontjából a céláramkörre van utalva.

Az idődiagram alapján végig tudjuk követni a kommunikációs eljárást. Az $\overline{MCLR}$ láb magas szintbe billenése RESET-et okoz, és a mikrovezérlő ICSP módba kerül. Az RB_6 lábon kapott órajel alapján[110] értelmezi az RB_7 lábon kapott utasítást. A vizsgált diagram a 18-as családra vonatkozik, ebben az esetben a programozás során a TBLRD (memória olvasás) és a TBLWT (memória írás) utasításokat használjuk. Az egyes családok, és speciális típusokra vonatkozó specifikáció jelentősen eltérhet[111], itt csupán az elvet kívántuk megvilágítani. Részletesebb információért l. ICSP User Guide.

Az ICSP-re előre fel kell készítenünk áramkörünket. A Microchip a 7.2.2.2. ábrán látható elrendezést ajánlja. Ennek egy kicsit módosított verziója látható a 7.2.2.3. ábrán



7.2.2.2. ábra: A gyártó által ajánlott elrendezés

Vizsgáljuk meg először a gyártó által ajánlott kapcsolást! Az $\overline{MCLR}$ lábat egy RC tag húzza fel tápra a stabil működés biztosítása érdekében[112] A 7.2.2.2. ábrán látható dióda[113] az ICSP csatlakozó felől érkező magasabb feszültség[114] leválasztására szolgál. A V_{DD} tápfeszültségre felhúzó ellenállást rövidzárral[115] helyettesítsük! A GND-ket összekötjük. Az RB_6 és RB_7 lábakon elhelyezett leválasztó áramkör a perifériától függően kerül kiválasztásra. Ezeken a lábakon a kommunikáció nagy sebességű, ezért az egyes égetőáramkörök általában nem tolerálják a nagy értékű kapacitásokat. Amennyiben nincs a lábakon nagykapacitású eszköz, és nem okoz problémát ha kívülről táplálást kap az adott periféria, akkor az izolációs áramköri rész elhagyható.

110 Minden lefutó élnél vesz mintát, elvileg ez a bitközép.

111 Pl. régebbi mikrovezérlők (PIC16F84) még 13V-ot igényeltek az $\overline{MCLR}$ lábon programozói módba történő lépéskor..

112 A használt PIC típusának megfelelő adatlapon a RESET szekcióban láthatunk külső áramkört a POR (Power On Reset – bekapcsolási reszet) biztosítására. Ezt minimálisan egy RC taggal érhetjük el, a katalógusajánlás tartalmaz még egy soros védőellenállást (1kΩ), valamint egy a kondenzátor kisütését segítő diódát. Az RC tag elemeinek értékét katalógus alapján határozhatjuk meg R általában kisebb mint 40kΩ (tipikus értéke 10kΩ), míg a kondenzátor értékét a V_{DD} minimum felfutási ideje alapján számolhatjuk.

113 A nagyobb zajtartalék érdekében alkalmazható Schottky dióda is.

114 A PIC típusától függően az ICSP módba történő átkapcsolás az MCLR lábon 3,3V-ot, 5V-ot, vagy 13V-ot igényel.

115 Gondoljunk bele, hogy pl. 100mA befolyó áram esetén egy 10Ω-os ellenálláson 1V feszültségesés keletkezne.

Melléklet

Néhány példa az RB_6, RB_7 lábakon lévő eszközre:

LED, LED kijelző, LCD kijelző: amennyiben nem zavar minket, hogy a programozás során a LED, vagy LED kijelző villog, nem kell leválasztanunk az ICSP csatlakozóról.

Nyomógomb, kapcsoló: ha nincs a csak védelmi célokat ellátó lehúzó ellenállás, akkor – szerencsétlen kapcsolóállásban – az adatvonalon jövő jelfolyamot egy az egyben rövidre zárhatjuk, ezért mindenképpen szükséges a leválasztás.

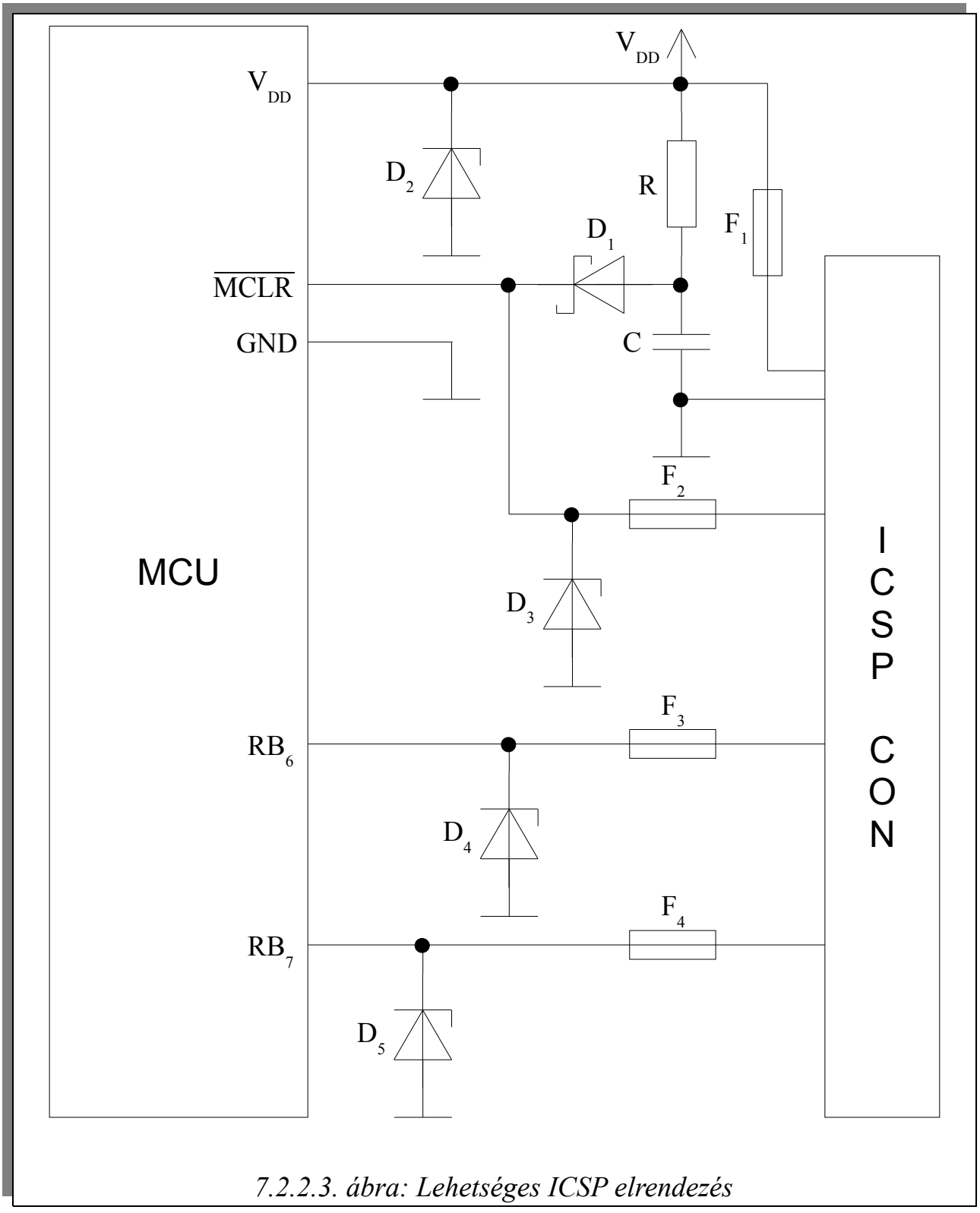
Tranzisztoros meghajtó: egyes tranzisztorok – különösen igaz ez a FET-ekre[116] – nagy bemeneti kapacitással rendelkeznek. A kapcsolás tervezésénél figyelemmel kell lennünk arra, hogy az RB_6, RB_7 lábakon normál működés esetén fellépő jeleknél programozás során nagyságrendekkel nagyobb frekvenciájú jeleknek is érvényesülniük kell.

Statikus vonal: a lassan változó vonalakat gyakran védjük szűrőkondenzátorokkal a nagyfrekvenciás zajok ellen, ill. az energiakiesés (áramlökés) kiküszöbölésének érdekében pufferkondenzátorokkal tartjuk a megfelelő feszültségértéket. Ezek a kondenzátorok programozás során károsak, ezért az ilyen vonalakat is le kell választanunk az égetőről.

Itt jegyeznénk meg, hogy nagyfrekvencián az ezekre a lábakra kötött alkatrészeket különös gonddal kell megválasztanunk. Nem használhatunk pl. szupresszor diódát nagy kapacitása miatt.

A leválasztás történhet jumper, kapcsoló, elektronikus kapcsoló, vagy optocsatoló segítségével.

116 Jellemző példa: IRF530 típusú FET-nél a C_{GS} = 640pF.



7.2.2.3. ábra: Lehetséges ICSP elrendezés

Feltételezzük, hogy az égetőn nincsen kialakítva védelem! Ebben az esetben a csatlakozó fordítva való ráhelyezése, vagy rossz lábkiosztása alkalmazása tönkreteheti nemcsak a céláramkört, de a programozót is. A 7.2.2.3. ábra az esetleges meghibásodásból, rossz alkalmazásból eredő hibák minimalizálását teszi lehetővé. Az R, C, D_1 elemek a gyári elrendezésből megismert szerepeket látják el. Az F_1, F_2, F_3, F_4[117] elemekkel áramkorlátozást, míg a D_2, D_3, D_4, D_5[118] diódákkal túlfeszültségvédelmet valósítunk meg. A biztosítékokkal sorba kötve használhatunk védőellenállást is – ez alól kivételt képez a tápfeszültség vonala, hiszen itt az ellenálláson ha a programozó táplálja az áramkört túl nagy feszültség esne. Ezek tipikus értéke: 270Ω.

A különböző égetőáramkörök ICSP lábkiosztása nem feltétlenül ugyanaz. Használat előtt mindig ellenőrizzük, hogy a céláramkörön kivezetett ICSP csatlakozó és az általunk használt programozó lábkiosztása megegyezik-e – kell-e esetleg valamilyen átalakítót alkalmaznunk. A lábak felcserélése a céláramkör, ill. adott esetben az égető tönkremeneteléhez vezethet.

117 Ajánlott FF jelzésű ún. félvezetővédő biztosíték alkalmazása.

118 A Zener diódák feszültségértéke – a PIC típusától függően – 3V3, 5V, vagy az $\overline{\text{MCLR}}$ lábon 15V.

7.3 PICkit 2 klón építése

Az áramkör megépítését csupán olyan embereknek tudom ajánlani, akiknek örömet okoz az építés, és a tudat, hogy egy általunk tervezett/készített áramköri egységet használunk fel munkánk, vagy hobbink során. Anyagilag nem éri meg ezt az áramkört elkészíteni[119] abban az esetben, ha az eredeti PICkit 2 funkcióira van szükségünk. Mindemellett a klónáramkör a PGD, PGC lábak leválasztása, a nagyobb áramigény kielégítése és a kialakított hardveres védelem miatt nagyobb tudású áramkörnek számít az eredeti konfigurációnél.

7.3.1 Hardver felépítés

A Microchip PICkit 2® áramkörhöz képest több változtatás is történt, amelyeket az alábbi fejezetekben ismertetünk.

7.3.1.1 Alkatrészek kiválasztása

Az alkatrészek kiválasztása során alkalmazott fő szempontok:

- Kizárólag[120] THT[121] alkatrészek alkalmazása
- Könnyen beszerezhető, és helyettesíthető típusok
- Ismert és gyakran használt tokozás

A fentieket figyelembe véve a bipoláris tranzisztorokat BC337, és BC327 típusokra cseréltük[122]. A booster FET-et a népszerű IRF5305 típussal[123], míg a kapcsoló FET-et az SI4435[124] típussal váltottuk fel. A mikrovezérlőt, a memória IC-ket, a műveleti erősítőt, a diódákat, a kondenzátorokat és az ellenállásokat THT tokozású párjaikkal helyettesítettük.

7.3.1.2 Oszcillátorkonfiguráció

A kvarckristályt hidegítő kondenzátor értéke 22pF-re módosult, az áramkör stabilitása a tesztek során így is megfelelőnek bizonyult és ezzel csökkentettük a feléledési időt.

119 Az eredeti PICkit 2 megvásárolható a Microchip cég kizárólagos hazai disztribútoránál a ChipCad kft-nél 8000 Ft. + ÁFA áron, míg ezen áramköri egység megépítésének alkatrészköltsége hozzávetőlegesen 4000 Ft.

120 Az egyik FET-et csak nagyon nagy költségráfordítással lehetett volna THT kiszerelésben alkalmazni, ezért egy helyen kénytelenek voltunk SO-8-as tokozást használni.

121 Through Hole Technology – furatszerelt technológia

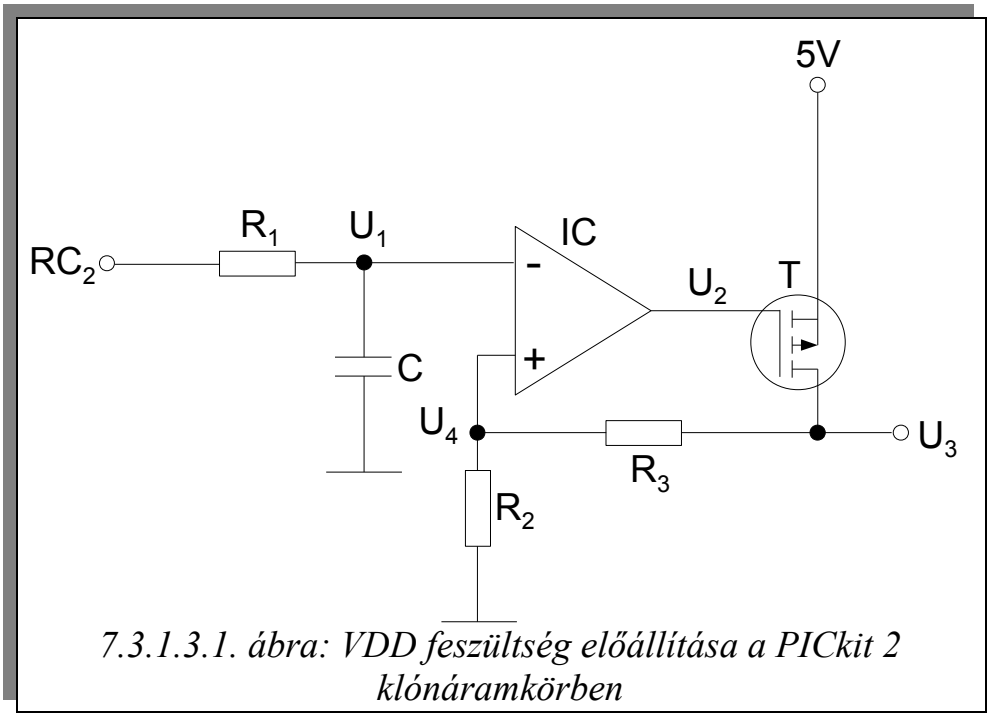
122 Bármely lábkompatibilis univerzális tranzisztor használható helyettük. Csupán arra figyeljünk, hogy a BC337 NPN, a BC327 PNP típusú legyen.

123 Itt is alkalmazható helyettesítés, de az R_{DSON} értékére különösen ügyeljünk!

124 Itt is alkalmazható helyettesítés, de az R_{DSON} érték mellett az alacsony U_{GS} értékre különösen ügyeljünk!

7.3.1.3 VDD előállító áramkör

Az eredeti kapcsolási rajzban (5.2.1. ábra) szereplő R_{34} jelzésű ellenállás szerepe, hogy tápellátás nélkül a FET source pontját stabil GND-re húzza. Ez a tranzisztor tápfeszültség nélkül vezérlést se kaphat, ezért ezt az ellenállást feleslegesnek ítéltem. Az R_7 jelzésű ellenállás gondoskodik a +V_TGT[125] pont földre húzásáról, ha a FET r_{DS} ellenállása szakadást képvisel. A GND-t azonban megkapja ez a pont az R_5, R_6 soros tagon keresztül is[126], tehát az R_7 ellenállás elhagyható. Az R_{31} jelzésű ellenállás szerepe számomra nem világos. A Q_1 jelzésű FET gate-jén nem folyik számottevő áram, mert áteresztő tranzisztorként működik. A gate-en csupán nagyfrekvenciás működtetéskor indulhat meg jelentős áram a C_{GS} kapacitás miatt[127], ezért ezt az ellenállást nem építettük be. Az így módosított áramköri részlet a 7.3.2.1.4. ábrán látható.

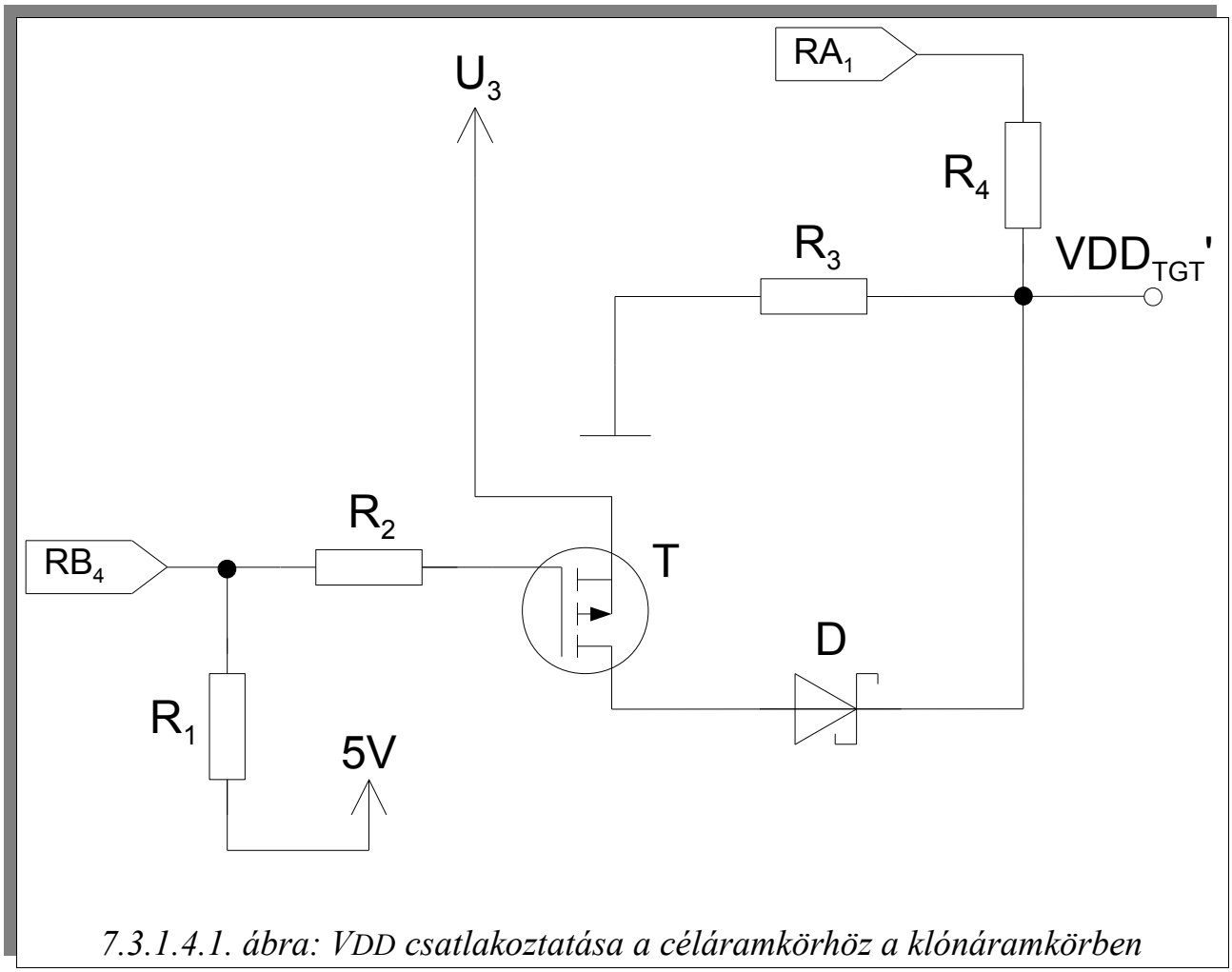


7.3.1.3.1. ábra: VDD feszültség előállítása a PICkit 2 klónáramkörben

125 U3 jelzésű feszültség
126 10k helyett valójában 20k-val van biztosítva a GND.
127 Az átlagoló RC tag miatt még hibás program esetén se kerülhet erre a pontra nagyfrekvenciás vezérlőjel.

7.3.1.4 VDD csatlakoztatása a céleszközhöz

Az eredeti PICkit 2 áramkörben lévő elgondolást (5.2.1.7.1. ábra) egy kissé erőltetettnek éreztem, ezért a 7.3.1.4.1. ábrán látható változtatásokat eszközöltem.

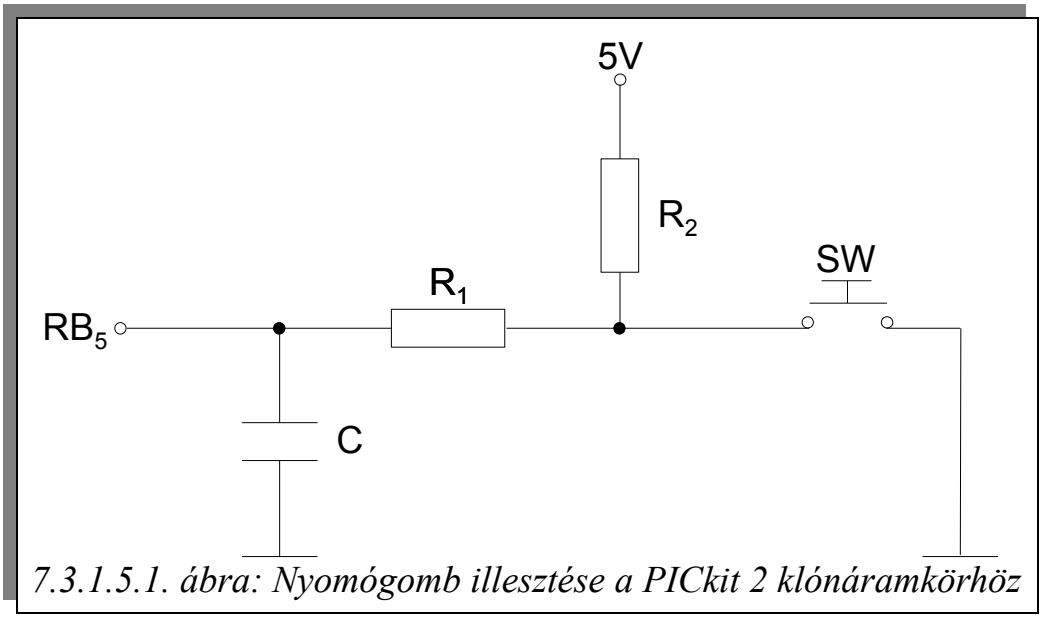


7.3.1.4.1. ábra: VDD csatlakoztatása a céláramkörhöz a klónáramkörben

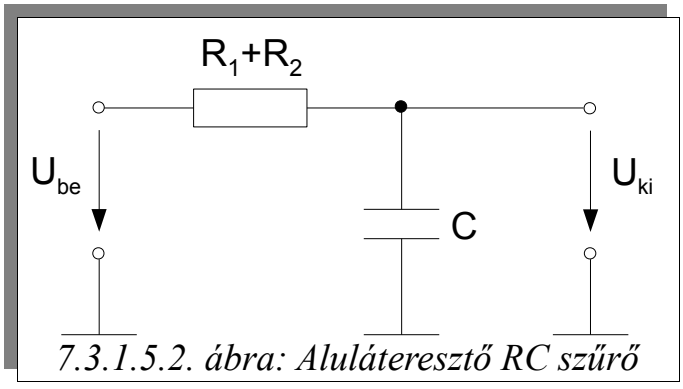
A két FET egy tokban megoldást a Microchip ki szerette volna használni, ezért az N csatornás FET-tel húzták le az VDD_{TGT}' pontot földre. Ezzel a módszerrel valamivel csökkenthető a fogyasztás. A PICkit 2 klónáramkörben kialakított megoldás szerint az VDD_{TGT}' pontot stabilan egy ellenálláson (4,7kΩ) keresztül csatlakoztatjuk a GND ponthoz. Az ellenálláson folyamatosan átfolyó veszteségi áram hozzávetőlegesen 1mA, amit elfogadhatónak ítéltem a megspórolt költségek figyelembevételével.

7.3.1.5 Nyomógomb illesztése

A nyomógombot a 7.3.1.5.1. ábrán látható módon csatlakoztattuk a mikrovezérlőhöz. A szűrőáramkörhöz tartozó R_1 ellenállás védelmi célokat[128] is ellát.



7.3.1.5.1. ábra: Nyomógomb illesztése a PICkit 2 klónáramkörhöz



7.3.1.5.2. ábra: Aluláteresztő RC szűrő

Az áramköri részlet felfogható egy egyszerű szűrőráramkörnek, amelyet 5V-os AC generátorral hajtunk meg. A frekvenciát a nyomógomb prellideje határozza meg. R_1 és R_2 ellenállás nem növelhető a végtelenségig, hiszen a mikrovezérlő a nagy impedanciákat nem tolerálja a bemeneten. Mindemellett a nyomógomb megfelelő üzemeltetéséhez is szükséges egy minimális (öntisztítási) áram. R_1 ellenállás minimális értéke 220Ω[129]

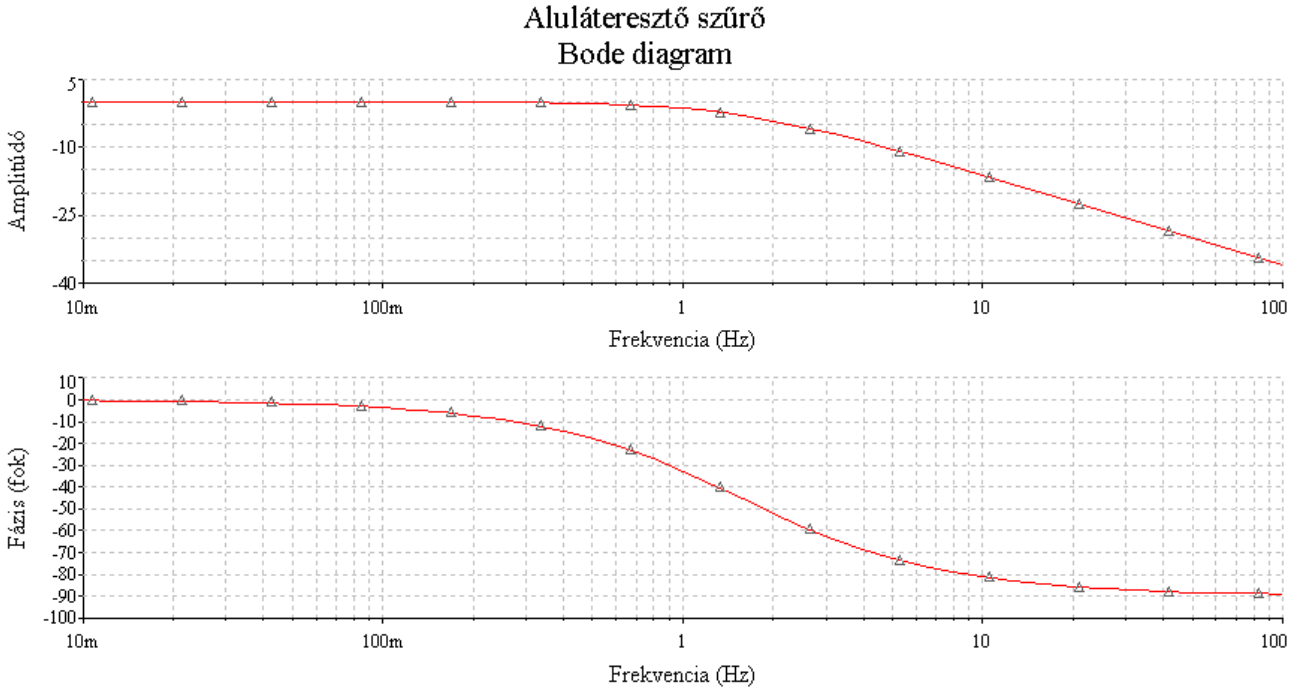
Az R_1, R_2 és C elemek által alkotott aluláteresztő szűrő méretezése Hendrik Wade Bode hálózati analízis módszere szerint történik. A . ábrán látható áramkör átviteli függvénye a következő:

$$A_u = \frac{U_{ki}}{U_{be}} = \frac{\frac{1}{j\omega C}}{R + \frac{1}{j\omega C}} = \frac{\frac{1}{SC}}{R + \frac{1}{SC}} = \frac{\frac{1}{SC}}{R + \frac{1}{SC}} \cdot \frac{SC}{SC} = \frac{1}{1 + SRC}$$

128 330Ω-os soros védő ellenállásként működik.

129 Rossz iránybeállítás esetén: $R_{min} = \frac{Umax}{I_{max}} = \frac{5\text{V}}{25\text{mA}} = 200\Omega$

Az $\frac{1}{1+ST}$ szabványos Bode-alakban szereplő időállandót az RC elemekből kapjuk: $T = RC = 10{,}33\,k\Omega \cdot 10\mu F = 103{,}3\,ms$. Az ebből adódó frekvencián ($\omega = \frac{1}{T} \rightarrow f = \frac{1}{2\Pi \cdot 103{,}3\,ms} \approx 1{,}5\,Hz$) az áramkör már 3dB-es csillapítással rendelkezik.

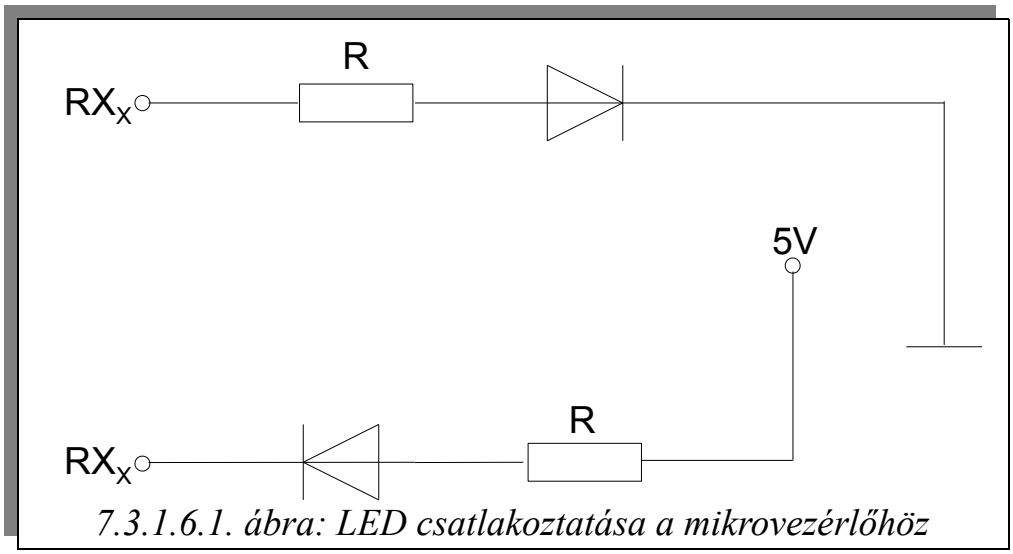


7.3.1.5.3. ábra:
Nyomógomb pergésmentesítő áramkörének amplitúdó- és fázismenete

Az amplitúdó-frekvencia karakterisztikából jól látható, hogy 10Hz környékén már 20dB-es (10-szeres) csillapítást érünk el, így az 5V-os pergő jelből már csak 0,5V jut a mikrovezérlő bemenetére, amit már nullának érzékel.

7.3.1.6 Állapotjelző LED-ek

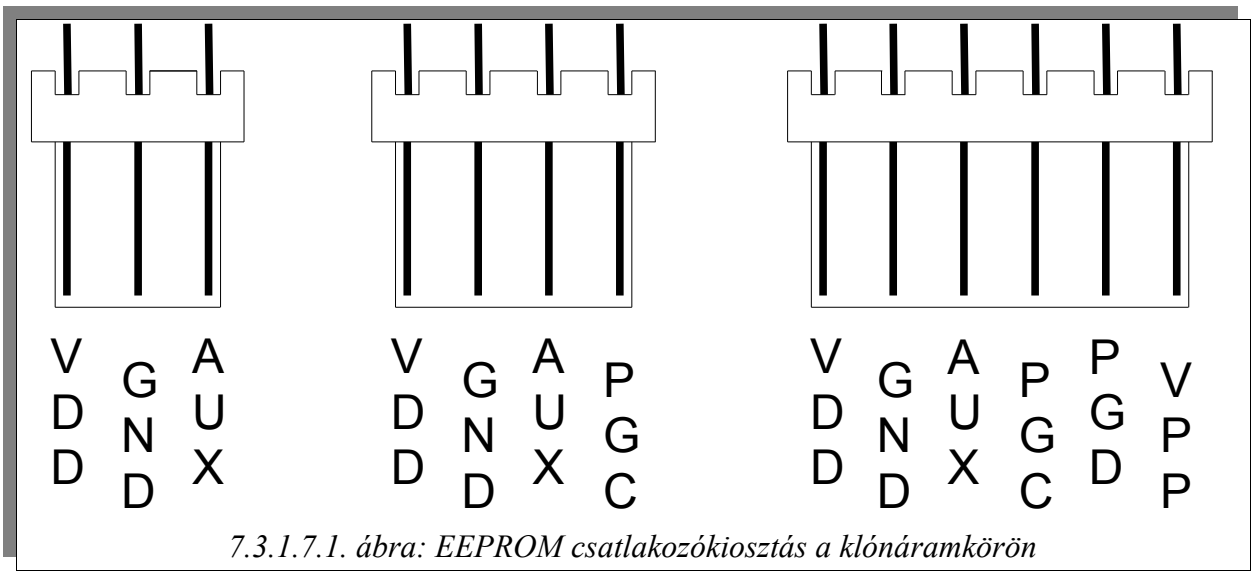
A 7.3.1.6.1. ábrán látható kapcsolási rajzon DS_1 jelzésű Busy LED áramkorlátozó ellenállása a negatív szálba került – a műszaki gyakorlatban mindig a pozitív ágba helyezzük el a védőellenállásokat (7.3.1.6.1. ábra).



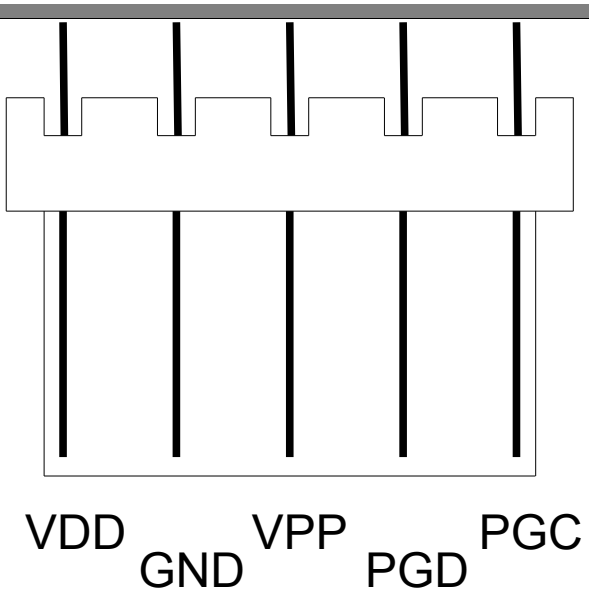
7.3.1.6.1. ábra: LED csatlakoztatása a mikrovezérlőhöz

7.3.1.7 Target csatlakozófelület

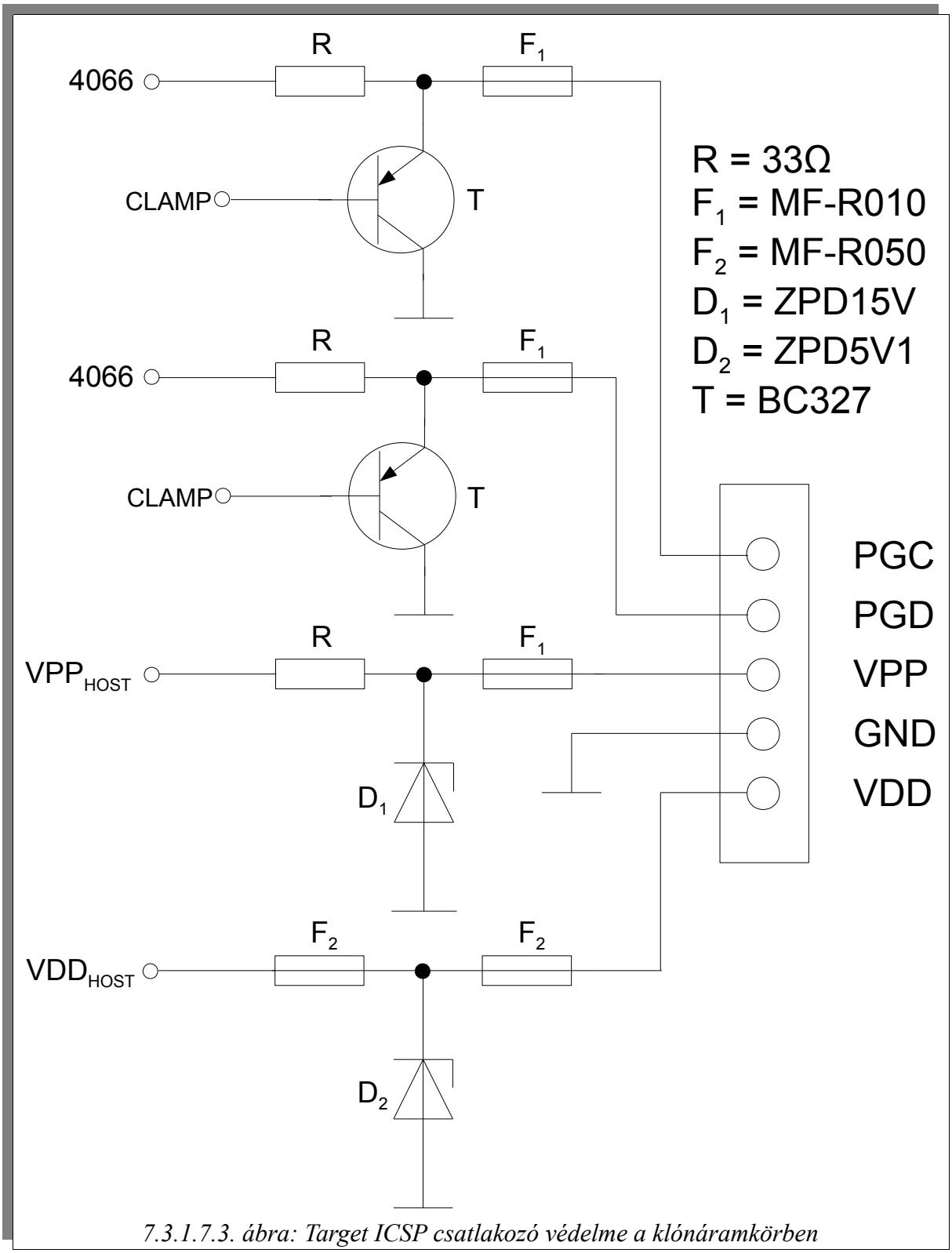
A klónáramkör megtervezésekor a céláramkörrel való kapcsolatot szabványos tápcsatlakozókkal biztosítottuk. A memória IC-k külön csatlakozókat kaptak (7.3.1.7.1. ábra). A mikrovezérlőhöz tartozó csatlakozó kiosztása (7.3.1.7.2. ábra) kompatibilis a http://plc.mechatronika.hu weboldalon megtalálható ICD2-vel. A HOST mikrovezérlőhöz kívülről az eredeti PICkit 2-ben is megtalálható tápfeszültség megfogó és egyben túlfeszültségvédő tranzisztoros áramköri részleten, valamint multifuse regenerálódó biztosítékkal megvalósított túláram-védőn keresztül kapcsolódhatunk (7.3.1.7.3. ábra). A PGC és PGD lábak ezen felül egy 4066 analóg kapcsolóval szoftveres úton leválaszthatók.



7.3.1.7.1. ábra: EEPROM csatlakozókiosztás a klónáramkörön



7.3.1.7.2. ábra:Target mikrovezérlő lábkiosztás a klónáramkörön



7.3.1.7.3. ábra: Target ICSP csatlakozó védelme a klónáramkörben

7.3.1.8 USB csatlakozó felület

A PC-vel történő kommunikációt biztosító USB csatlakozót elláttuk túlfeszültség és túláram-védelemmel. A kapcsolási rajzon látható módon a Zener-dióda 5,1V-on megfogja a vonalat, a multifuse típusú regenerálódó biztosíték pedig nem engedi az adott érték (50mA, és 500mA) fölé menni az áramot.

7.3.1.9 Tápfeszültség kiválasztása

A Programmer – To – Go funkció lényege, hogy PC nélkül is tudjunk programozni, felhasználva a PICkit 2 áramkörben lévő EEPROM memóriákat. A funkció komoly hibája, hogy USB-s csatlakozón igényli az 5V-os tápfeszültséget a működéshez. A klónáramkörön egy jumper segítségével ki tudjuk választani, hogy az USB csatlakozóról, vagy a Target áramkörről szeretnénk meghajtani a Host mikrovezérlőt.

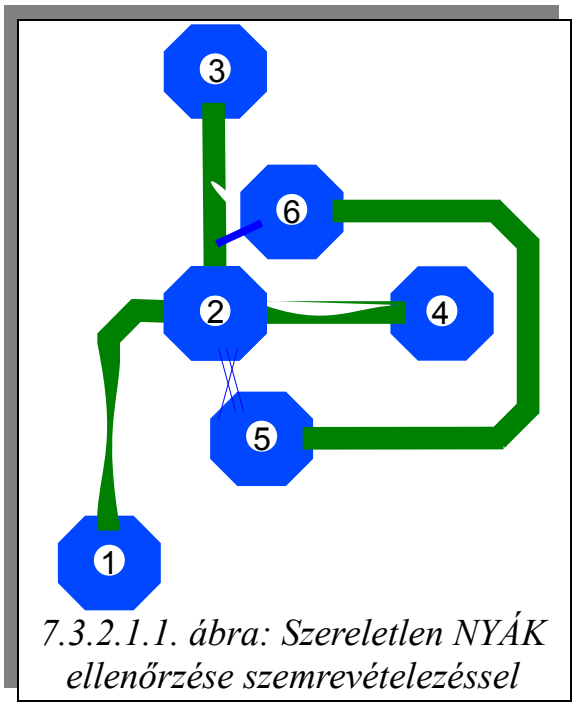
7.3.2 PICkit 2 klónáramkör elkészítése[130]

Csak a 7.3.1. fejezet elolvasása után kezdjünk hozzá az áramkör megépítéséhez. A http://plc.mechatronika.hu weboldalról töltsük le a szükséges dokumentációt (leírás, kapcsolási rajz, nyákterv).

7.3.2.1 Szereletlen NYÁK[131] ellenőrzése

A NYÁK kimaratása után ellenőrizzük a lemezt szemrevételezéssel, és végezzünk műszeres mérést(!) is. A vezetősávokkal összekapcsolni kívánt pontok között DMM[132] segítségével mérjünk ellenállást[133]! Ezek után a vezetősávokkal összekötött szigeteket egy pontként kezelve ellenőrizzük a szakadást közöttük! Egy NYÁK-on maximum 3 hibát javítsunk, ha ennél többel találkozunk, készítsünk új áramköri lapot!

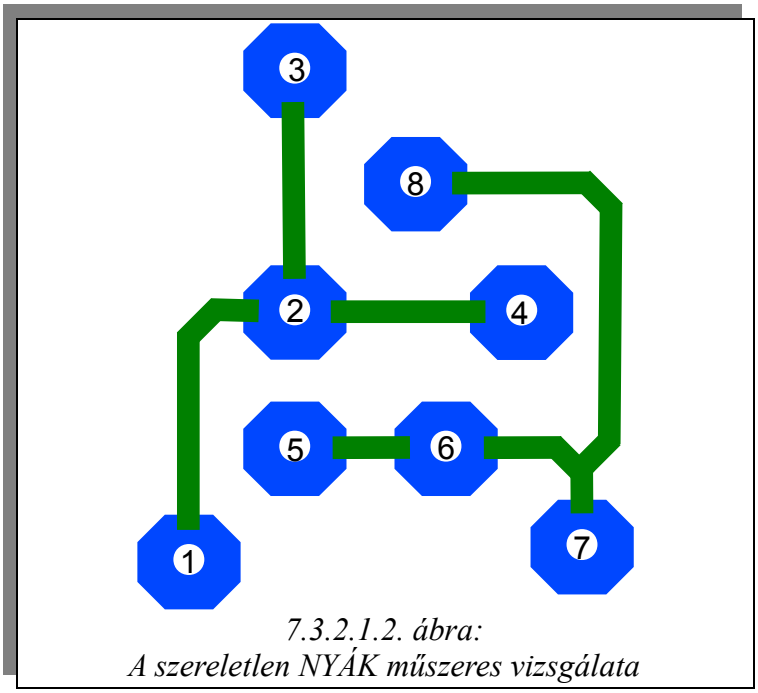
Szemrevételezéssel a 7.3.2.1.1. ábrán látható vezetősáv-elvékonyodás (túlmaratás, alámarás) és vezetősáv-zárlat hibákat tudjuk felfedezni.



7.3.2.1.1. ábra: Szereletlen NYÁK ellenőrzése szemrevételezéssel

Az 1-2 pontok közötti vezetősávon a jellemzően előforduló alámarást, míg a 2-3, és 2-4 pontok között a nem megfelelően előkészített NYÁK[134] miatti hibát láthatjuk. A 6, és az 5 pontok nem különülnek el az 1-2-3-4 pontok által alkotott szigettől – ez lehet valamilyen szennyeződés miatt, ill. a túl rövid maratástól.

130 Az itt olvasható leírás célja, hogy elkerüljük az „összeraktam, de nem működik, mi lehet a baj?!?” típusú hibajelentéseket, kérdéseket.
131 Nyomtatott Áramköri Kártya, vagy NYHL – Nyomtatott Huzalozású Lemez
132 Digitális Multiméter
133 A legtöbb multiméter diódamérés állásban csipogással jelzi a rövidzárt (ez típusonként változhat, pl. <10Ω; <90Ω)
134 Ha a lemezt nem tisztítjuk meg kellőképpen, a szennyeződések komoly hibákat, vezetősáv kihagyásokat, ill. zárlatokat okozhatnak.



7.3.2.1.2. ábra:
A szereletlen NYÁK műszeres vizsgálata

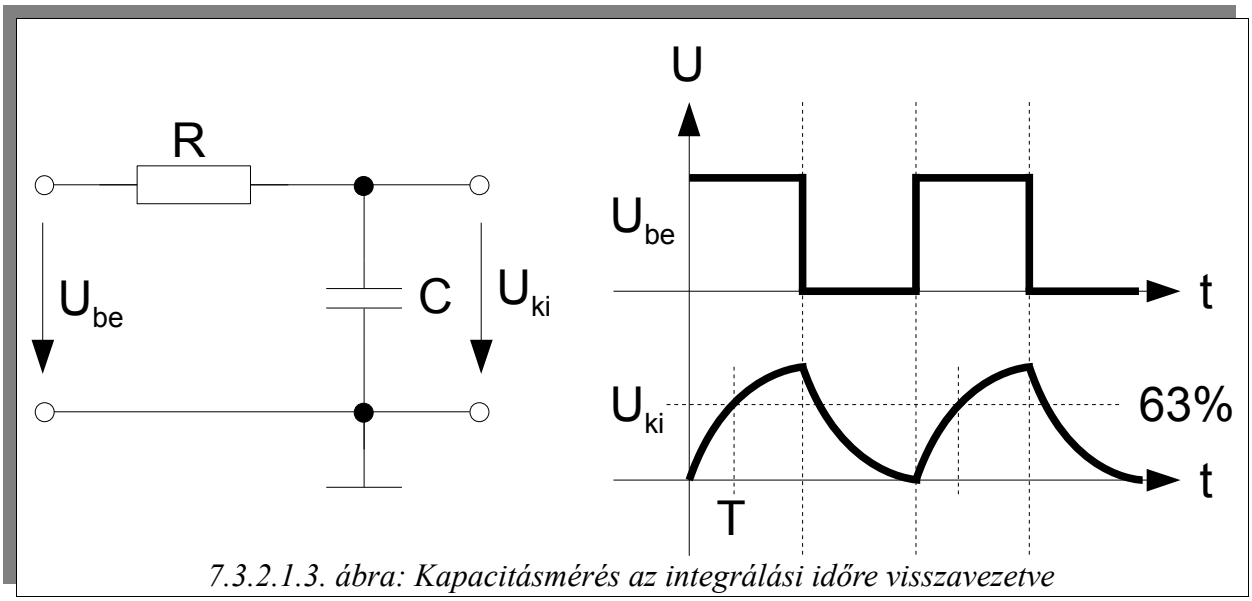
Műszeres vizsgálat: Állítsuk a multimétert rövidzár méréshatárba (folyamatosságvizsgálat, szakadás mérés). A 7.3.2.1.2. ábrán látható példa esetében a műszer egyik mérővezetékét csatlakoztassuk az 1. pontra. A másik mérővezetéket érintsük hozzá sorrendben a 2, 3, 4 pontokhoz – mindegyik esetben rövidzárt kell mérnünk. Tegyük át az eddig nem mozgatott mérővezetéket az 5. pontra, majd a másikat sorban a 6-os a 7-es és a 8-as pontokra – mindhárom esetben rövidzárt kell mérnünk. A két sziget tetszőleges pontját (pl. 1, 5) kiválasztva közöttük szakadást kell mérnünk.

Beültetendő alkatrészek vizsgálata

Ellenállások: Az ellenállásokat színkódjuk, vagy feliratozásuk alapján azonosítsuk be, majd ellenőrizzük értékeiket. Az emberi ellenállás – függően a személytől, helytől, lelkiállapottól, stb. – n*100kΩ[135] nagyságrendbe esik, ezért 10kΩ-os ellenállásig nem követünk el komoly hibát, ha belemérjük a testünket is.

Kapacitások: Amennyiben olyan multiméterrel rendelkezünk, amelynek van kapacitásmérő méréshatára, használjuk ezt az opciót. A 7.3.2.1.3. ábra egy alternatív mérési elrendezést mutat, melyben az integrálási idő alapján számolhatunk. (Elektrolit kondenzátoroknál – a névleges feszültség mellett – ügyelnünk kell a helyes polaritásra)

135 Figyelem: érintésvédelem esetén egységesen 1kΩ-al számolunk a legrosszabb esetre (worst-case).



7.3.2.1.3. ábra: Kapacitásmérés az integrálási időre visszavezetve

Induktivitás: Induktivitást csupán a drágább multiméterek képesek korlátozott mértékben mérni. A kapacitásmérésnél bemutatott elrendezéshez hasonlóan azonban az induktivitás értéke is meghatározható (csupán az alkatrészek pozícióit kell felcserélni).

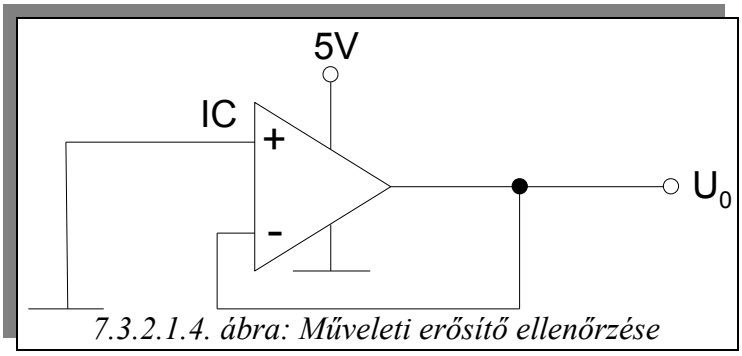
Diódák: A DMM mérőműszert állítsuk dióda méréshatárba. A pozitív mérővezetéket az anódra, a negatívat a katódra téve mérjük a dióda nyitófeszültségét (egyenirányító: 0,5V÷0,75V; schottky: 0,25V÷0,4V). A mérőzsinórokat megcserélve szakadást kell mérnünk[136].

Tranzisztorok: A DMM mérőműszert állítsuk dióda méréshatárba. NPN tranzisztor esetén a pozitív mérővezetéket helyezzük a bázisra, a negatívat pedig az emitterre és mérjük a tranzisztor BE nyitófeszültségét (~0,6V). Helyezzük át a negatív mérőzsinórt a kollektor pontra, és mérjük a BC nyitófeszültséget (~0,6V). A két mérővezetéket a kollektor és az emitter közé helyezve szakadást kell mérnünk. PNP tranzisztor esetén hasonlóan járunk el, azonban a mérővezetékeket megcseréljük[137].

136 Típustól függően a mérőműszer vagy a maximális értéket jelzi (pl. 1500), vagy méréshatár túllépést jelez (pl. villog az érték).

137 Mivel a DMM lebegő bemenetű műszer ezért nem követünk el hibát, ha felcseréljük a vezetékeket, csak ekkor negatív értéket kapunk. Ebből következően egyszerűen el tudjuk dönteni, egy tranzisztorról, hogy NPN, vagy PNP típusú-e: a pozitív mérővezetéket a bázisra helyezve, a negatívot az emitterre, vagy a kollektorra csatlakoztatva – pozitív érték esetén NPN, negatív érték esetén PNP tranzisztorral van dolgunk.

Integrált áramkörök: A PICkit2-ben lévő integrált áramkörök közül a műveleti erősítőt tudjuk egyszerűen tesztelni – a 7.3.2.1.4. ábra szerinti elrendezésben mérjük a műveleti erősítő ofszet feszültségét (katalógus adat [mV nagyságrend]).



7.3.2.1.4. ábra: Műveleti erősítő ellenőrzése

7.3.2.2 Alkatrészek beültetése

Az alkatrészek beültetésénél ügyeljünk a gondos és precíz munkára. Csak hőfokszabályozóval ellátott Weller pákával kezdjünk neki a munkának – a pillanatpákákat hanyagoljuk. Vegyük figyelembe, hogy az egyes alkatrészek eltérő mértékben tolerálják a nagy hőmérsékletet.

Fokozott figyelemmel helyezzük el a polaritással rendelkező alkatrészeket, mert ezek helytelen pozícióba történő ültetése nem csak hibás működéshez, de az eszköz tönkremeneteléhez is vezethet.

Ha rossz helyre, pozícióba ültettünk be alkatrészt a kiszedésénél mindig használjunk önszippantót, és/vagy önszívó sodratot.

A PICkit2 klónáramkör alkatrészeinek beültetési sorrendje a következő:

- átkötések
- ellenállások
- diódák
- kvarckristály
- induktivitás
- nyomógombok
- dip foglalatok
- száraz kondenzátorok
- LED-ek
- Target ICSP csatlakozók
- 50mA multifuse biztosítékok
- bipoláris tranzisztorok
- Host ICSP csatlakozó
- elektrolit kondenzátorok (kivétel 1000µF)
- 500mA multifuse biztosítékok
- USB csatlakozó
- 1000µF
- FET-ek

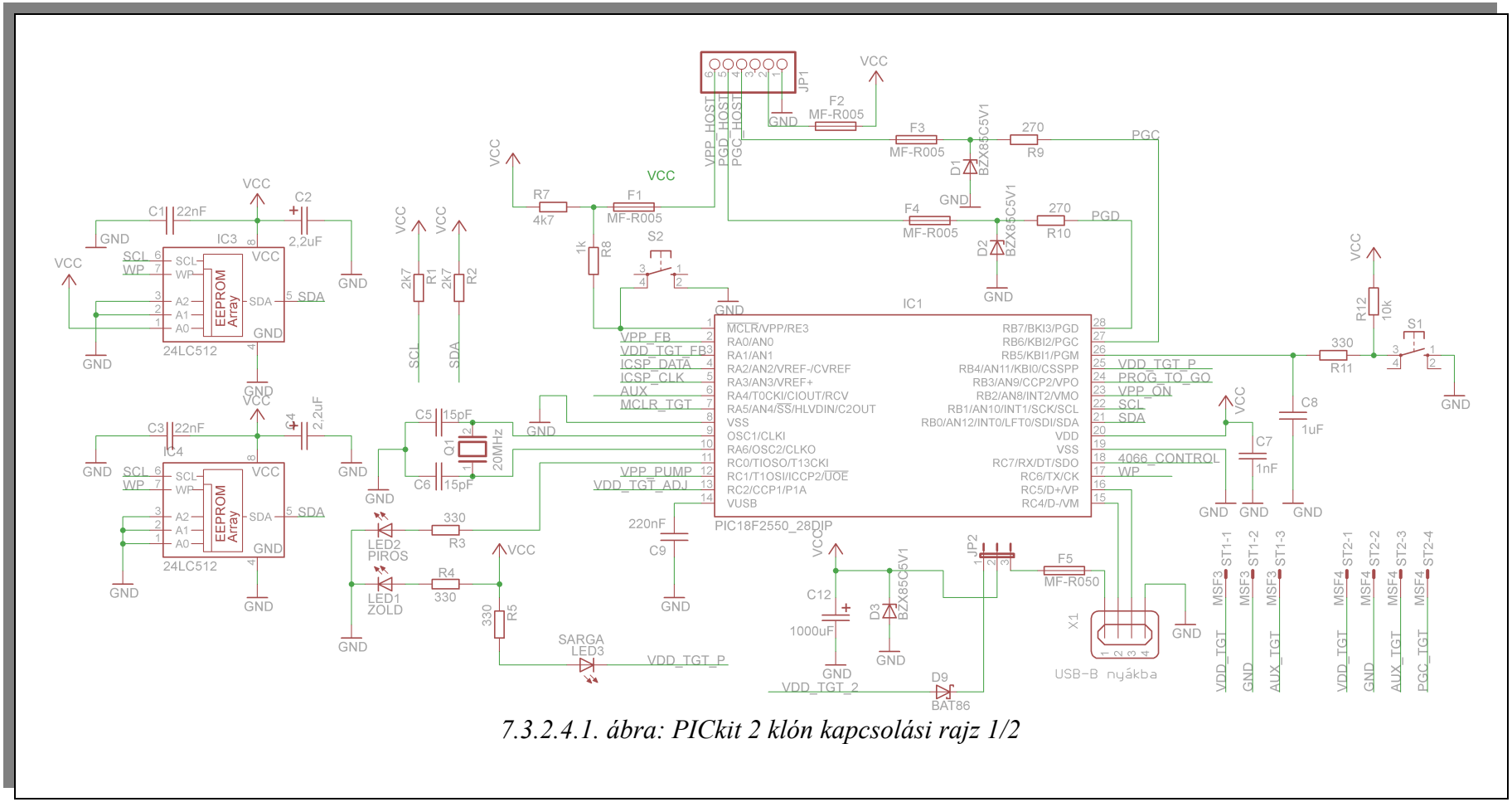
7.3.2.3 Bemérés

A kész PICkit 2 klónáramkört a 6.4. fejezetben leírtak szerint csatlakoztassuk a PC-hez és ellenőrizzük a működőképességét. Helyes működés esetén használjuk egészséggel a szerkezetet. Amennyiben hibát tapasztalunk a következők szerint járjunk el[138]:

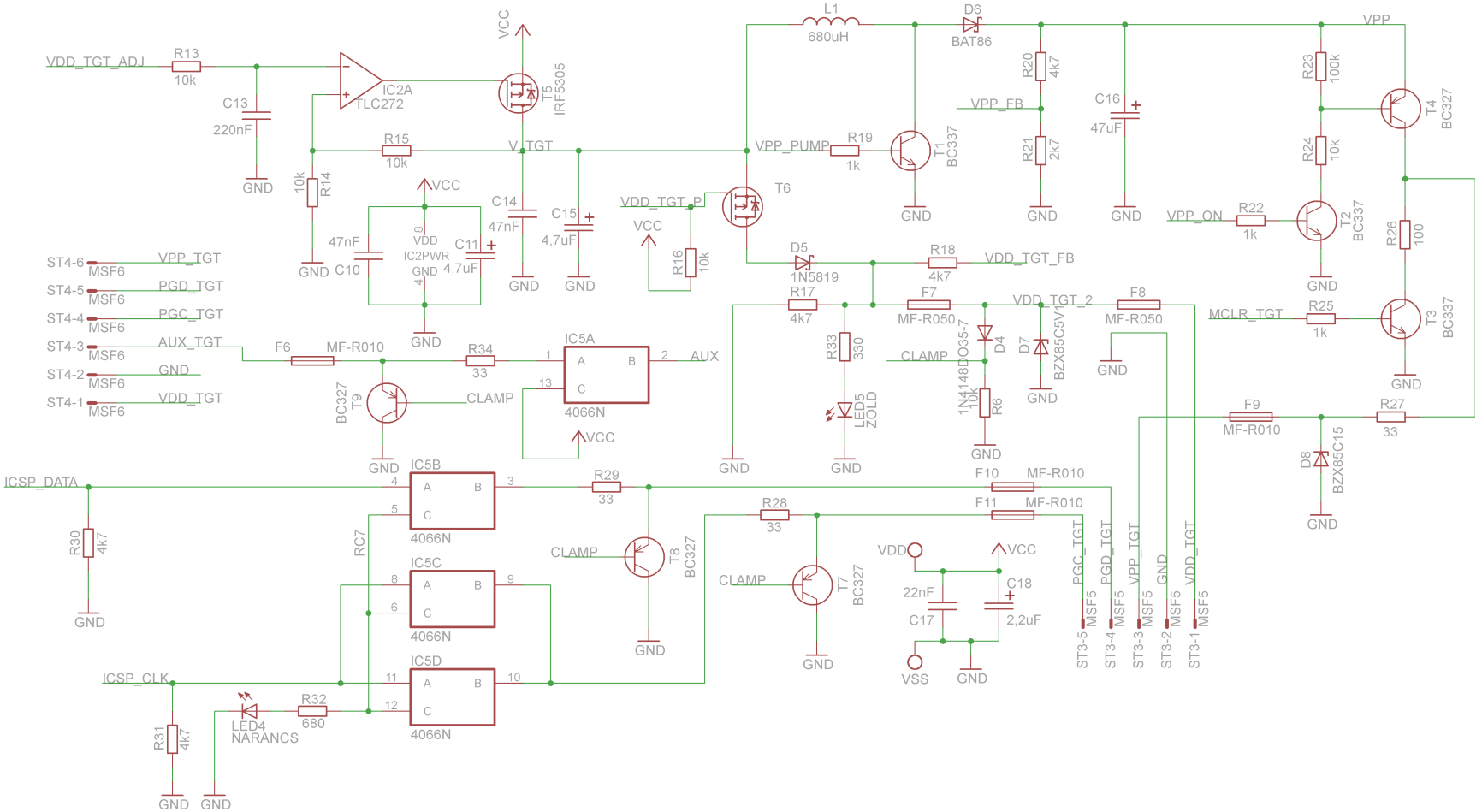
- Passzív panelen szemrevételezéssel ellenőrizzük a beforrasztott nyákot – nem egyértelmű helyzetben alkalmazzunk multimétert a folyamatosságvizsgálathoz.
- Passzív panelen ellenőrizzük a PC-hez történő csatlakozás épségét multiméter segítségével.
- Aktív panelen ellenőrizzük a szükséges tápfeszültségeket (PIC mikrovezérlő, műveleti erősítő).
- Oszcilloszkóp segítségével ellenőrizzük, hogy a PIC mikrovezérlő oszcillátora rezeg-e.
- Aktív panel esetén a biztosítékokon helyes konstrukcióban csupán néhány mV feszültség eshet – amennyiben ennél többet mérünk valahol az áramkörben rövidzárlat történt.
- Lépjünk be a PICkit 2 szoftver hibakereső üzemmódjába (6.3.4.1. fejezet).
- Mérjük végig az egyes részáramköröket, hogy a leírásban bemutatottaknak megfelelően működnek-e.

138 Mielőtt a hibakeresést megkezdjük – amennyiben eddig nem tettük – mindenképpen vizsgáljuk meg, hogy milyen eltérések mutatkoznak a klónáramkör és az eredeti PICkit 2 között – s ennek függvényében végezzük a hibaelhárítást.

7.3.2.4 Kapcsolási rajz



7.3.2.4.1. ábra: PICkit 2 klón kapcsolási rajz 1/2



7.3.2.4.2. ábra: PICkit 2 klón kapcsolási rajz 2/2

7.3.2.5 Alkatrészlista

Megnevezés	Érték	Tokozás	Megjegyzés/ darabszám	Cikkszám (Lomex)
Ellenállások (Fémréteg, vagy szénréteg 2 raszteres ellenállások, E24)				
R_{27}, R_{28}, R_{29}, R_{34}	33Ω	MF0204	0,4W / 1db	02-07-64
R_{26}	100Ω	MF0204	0,4W / 1db	02-06-77
R_9, R_{10}	270Ω	MF0204	0,4W / 6db	02-07-88
R_3, R_4, R_5, R_{11}, R_{33}	330Ω	MF0204	0,4W / 4db	02-06-79
R_{32}	680Ω	MF0204	0,4W / 1db	02-07-97
R_8, R_{19}, R_{22}, R_{25}	1kΩ	MF0204	0,4W / 4db	02-06-80
R_1, R_2, R_{21}	2,7kΩ	MF0204	0,4W / 3db	02-08-09
R_7, R_{17}, R_{18}, R_{20}, R_{30}, R_{31}	4,7kΩ	MF0204	0,4W / 6db	02-08-14
R_6, R_{12}, R_{13}, R_{14}, R_{15}, R_{16}, R_{24}	10kΩ	MF0204	0,4W / 7db	02-06-83
R_{23}	100kΩ	MF0204	0,4W / 1db	02-06-85
Kondenzátor				
C_5, C_6	15pF	SSM 15	NPO / 2db	07-09-64
C_7	1nF	SSM 15	NPO / 1db	07-10-09
C_1, C_3, C_{17}	22nF	SSM 15	X7R / 3db	07-01-30
C_{10}, C_{14}	47nF	SSM 16	X7R / 2db	07-03-08
C_9	220nF	SSM 16	Y5V / 1db	07-09-03
C_{13}	220nF	WIM	Fólia / 1db	07-05-30
C_8	1µF	SSM 16	Y5V / 1db	07-03-27
C_2, C_4, C_{18}	2,2µF	4X5 (YAG)	50V / 3db	12-04-28
C_{11}, C_{15}	4,7µF	4X7(YAG)	50V / 2db	12-01-36
C_{16}	47µF	6X11 (YAG)	50V / 1db	12-06-54
C_{12}	1000µF	10X19 (YAG)	25V / 1db	12-04-70
Induktivitás				
L_1	680µH	Axiális 4X10,5 (YAG)	0,113A / 1db	55-00-93

Melléklet

Dióda				
D_1, D_2, D_3, D_7	BZX85C5V1	DO-41	5,1V 1,3W Zener / 7db	18-00-75
D_8	BZX85C15	DO-41	15V 1,3W Zener / 1db	18-00-77
D_4	1N4148	DO-35	Kapcsoló (0,15A) / 1db	16-02-45
D_6, D_9	BAT86	DO-34	Schottky (0,2A) / 1db	16-03-49
D_5	1N5819	DO-41	Schottky (1A) / 1db	16-02-71
LED_1, LED_5	L-934PGT	Ø3mm	Zöld / 1db	34-01-22
LED_2	L-934IT	Ø3mm	Piros / 1db	34-01-51
LED_3	L-934YDLK	Ø3mm	Sárga / 1db	34-05-20
LED_4	L-934NC	Ø3mm	Narancs / 1db	34-01-41
Tranzisztor				
T_1, T_2, T_3	BC337	TO-92	NPN (0,8A) / 3db	21-07-97
T_4, T_7, T_8, T_9	BC327	TO-92	PNP (0,8A) / 1db	21-07-19
T_5	IRF5305	TO-220	P-FET $R_{DS_{on}}$<0,06Ω / 1db	21-08-08
T_6	SI4435BDY-E3	SO-8	P-FET $R_{DS_{on}}$<0,015Ω / 1db	86-01-63
Kvarckristály				
Q_1	HC49/S	Low profil	20Mhz / 1db	40-01-21
Integrált áramkör (IC)				
IC_1	PIC18F2550	DIP-28	1db	ChipCad
IC_3, IC_4	24LC512	DIP-8	2 db	ChipCad
IC_2	MCP6002[139]	Dip-8	1 db	ChipCad
IC_5	CD4066BE	DIP-14	1 db	31-02-38
Biztosíték				
F_1, F_2, F_3, F_4	MF-R005	D=7,4	Multifuse / 8 db	44-01-72
F_6, F_9, F_{10}, F_{11}	MF-R010	D=7,4	Multifuse / 8 db	44-00-66
F_5, F_7, F_8	MF-R050	D=7,9	Multifuse / 3 db	44-00-71

139 Helyettesíthető: TLC-272CP (Cikkszám: 32-04-13)

Melléklet

Nyomógomb				
S_1, S_2	FSM6JH	TACT 6X6	Nyák nyomógomb / 2 db	45-05-38
Foglalat				
	PDIP	DIP-28	1db	41-00-47
	PDIP	DIP-14	1 db	41-00-51
	PDIP	DIP-8	2 db	41-00-58
Csatlakozó				
ST_1	NCW254-03R	3P 90°	Tápcsatlakozó apa / 1 db	43-09-95
ST_2	NCW254-04R	4P 90°	Tápcsatlakozó apa / 1db	43-09-96
ST_3	NCW254-05R	5P 90°	Tápcsatlakozó apa / 1db	43-09-97
ST_4	NCW254-06R	6P 90°	Tápcsatlakozó apa / 1db	43-09-98
	NCH254-03	3P	Tápcsatlakozó anya ház / 1db	43-09-07
	NCH254-04	4P	Tápcsatlakozó anya ház / 1db	43-09-08
	NCH254-05	5P	Tápcsatlakozó anya ház / 1db	43-09-09
	NCH254-06	6P	Tápcsatlakozó anya ház / 1db	43-09-10
	NCT254-T	Krimpelhető	Tápcsatlakozó anya / 20 db	43-09-17
JP_1		6P	2,54 / 1db	43-07-21
JP_2		3P	2,54 / 1db	43-07-21
JP_3		2P	2,54 / 1db	43-07-21
	Hüvelysor	20P[140]	2,54 / 1db	43-00-08
X_1	USB-B	USB-B04FH-S	90° anya / 1db	43-14-75
Vezeték				
	USB A-B		1,8m / 1db	54-01-42
	ICSP vezeték	6X0,22	2m	54-00-47
	Zsugorcső	Ø1mm	1m	53-00-21
	Zsugorcső	Ø4mm	1m	53-00-19

140 Függ attól, hogy a Target felé milyen csatlakozókat szeretnénk kialakítani.

A PIC mikrovezérlőket, az MCP6002 típusú műveleti erősítőt és a soros EEPROM memóriákat a CHIPCAD kft-től lehet beszerezni. A többi alkatrész kapható a legtöbb kiskereskedelmi egységben pl. Lomex.

7.3.3 Firmware ismertetése

A PICkit 2 áramkört működtető firmware forrása nyilvános. A klón működéséhez az aktuális firmware-t módosítani kell. A plc.mechatronika.hu weboldalon megtalálható a hex fájl, melyben a 4066 analóg kapcsoló működtetése már meg van oldva. A firmware leíró részében módosítottuk az igényelt áramot is, így a szoftveres áramkorlát már 480mA lett. Mindemellett a Programmer - To Go funkciónál, ill. a V_{DD} bekapcsolásánál is apróbb módosításokat eszközöltünk. A következő fejezet részletesen ismerteti a változtatásokat és azok okait.

7.3.3.1 Változtatások a gyári firmware-ben

A nyilvánosságra hozott módosított firmware-ben a változtatásokat megjegyzésben elláttuk a „Javítás” karaktersorozattal.

7.3.3.1.1 PICkit2_FWv2 projekt módosítása

Módosított fájlok:

- IO_CFG.H lábkonfigurációs headerfájl
- USBDSC.C leíró fájl
- PICKIT.C főprogram
- PICKIT2.ASM assembly primitív
- PK_PROG2GO.C Programmer – To – Go funkcióhoz készített függvények

Az IO_CFG.H fájlban definiáltuk a 4066 típusú analóg kapcsolót vezérlő RC7 lábat (7.3.3.1.1.1. ábra).

```
124
125     #define tris_WP             TRISCbits.TRISC6    // RC6 Output
126     #define WP                  LATCbits.LATC6
127
128     #define tris_ICSP_HiZ       TRISCbits.TRISC7    // Javítás; RC7: 40
129     #define ICSP_HiZ            LATCbits.LATC7      //Javítás
130
131     #endif //IO_CFG_H
```

7.3.3.1.1.1. ábra: Analóg kapcsoló vezérlőlábának definiálása

Az USBDSC.C leíró fájlban módosítottuk a PICkit2 által küldött áramigényt az eredeti 100mA-ről 480mA-re[141] (7.3.3.1.1.2. és 7.3.3.1.1.3. ábra).

```
70  CFG01={
71      /* Configuration Descriptor */
72      sizeof(USB_CFG_DSC),    // Size of this descriptor in bytes
73      DSC_CFG,                // CONFIGURATION descriptor type
74      sizeof(cfg01),          // Total length of data for this cfg
75      1,                      // Number of interfaces in this cfg
76      1,                      // Index value of this configuration
77      2,                      // Configuration string index
78      _DEFAULT,               // Attributes, see usbdefs_std_dsc.h
79      240,                     // Max power consumption (2X mA)
80
```

7.3.3.1.1.2. ábra: CFG01 módosítása

```
119 CFG02={
120     /* Configuration Descriptor */
121     sizeof(USB_CFG_DSC),    // Size of this descriptor in bytes
122     DSC_CFG,                // CONFIGURATION descriptor type
123     sizeof(cfg02),          // Total length of data for this cfg
124     1,                      // Number of interfaces in this cfg
125     2,                      // Index value of this configuration
126     4,                      // Configuration string index
127     _DEFAULT,               // Attributes, see usbdefs_std_dsc.h
128     240,                     // Max power consumption (2X mA)
129
```

7.3.3.1.1.3. ábra: CFG02 módosítása

141 Ezt az értéket a CFG01 és CFG02 konfigurációs leíróban találjuk. Az eredetileg megadott 50-es számértéket 240-re módosítottuk.

A PICKIT.C fájlban több módosítást is el kellett végezni.

```
355         // ************************************************************
356         // * WARNING * WARNING * WARNING * WARNING * WARNING * WARNING *
357         // * Do not turn on both half-bridge gates at the same time.   *
358         // * It will cause a direct short of +V_TGT to GROUND.         *
359         // ************************************************************
360
361         Vdd_TGT_N = 0;                  // initialize half-bridge N-gate off
362         tris_Vdd_TGT_N = 1;             // Output -> INPUT Javítás
363
364         Vdd_TGT_P = 1;                  // initialize half-bridge P-gate off
365         tris_Vdd_TGT_P = 0;             // Output
366
367         tris_PROG_SWITCH = 1;           // Input
368     //    PROG_SWITCH_pin = 1;            // initialize port to 1   //Javítás: bemenetre n
369         INTCON2bits.NOT_RBPU = 1;       // initialize PORTB pull-ups off    //Javítás:0->1
```

7.3.3.1.1.4. ábra: PORTB felhúzó ellenállásainak kikapcsolása

A 7.3.3.1.1.4. ábrán láthatjuk, hogy a VDD_TGT_N láb (RB_3) irányát bemenetre módosítottuk, mivel a klónáramkörben ez a funkció nincs megvalósítva. A PORTB-n lévő felhúzó ellenállásokat kikapcsoltuk, mert az RB_5-re kötött nyomógomb rendelkezik saját felhúzó ellenállással. Mindemellett a PROG_SWITCH_pin = 1 sort kikommenteztük, hiszen a bemenetre nem érdemes ráírni.

```
374         // ------
375         // Port C
376         // ------
377
378         BUSY_LED = 0;                   // initialize Busy LED to off
379         tris_BUSY_LED = 0;              // Output
380
381         Vpp_PUMP = 0;                   // initialize latch to 0 (low)
382         tris_Vpp_PUMP = 0;              // RC1 Output (CCP2)
383
384         Vdd_TGT_ADJ = 0;                // initialize latch to 0 (low)
385         tris_Vdd_TGT_ADJ = 0;           // RC2 Output (CCP1)
386
387         tris_WP = 0;                    // Output
388         WP = 1;                         // initialize WP to protect (1)
389
390         ICSP_HiZ = 0;                   // Javítás: analóg kapcsoló ki
391         tris_ICSP_HiZ = 0;              // Javítás
```

7.3.3.1.1.5. ábra: A 4066 típusú analóg kapcsoló kikapcsolása, irányának megadása

A 7.3.3.1.1.5. ábrán látható utasítások az PICkitInit() inicializáló függvényben helyezkednek el. Bekapcsoláskor, vagy reszet esetén biztosítanunk kell, hogy a 4066 analóg kapcsoló a PGC, PGD lábakat leválassza a targetről (vagyis kimenetként kell konfigurálnunk a vezérlőlábát és 0-s kezdőértéket kell kapnia).

```
487
488         // Check for PK2GO mode
489         if ((EE_ReadByte(PK2GO_KEY1) == 0x50) && (EE_ReadByte(PK2GO_KEY2) == 0x4B) && (EE_Re
490             {
491                 // get EEPROM size
492                 pk2go_memsize = EE_ReadByte(PK2GO_MEM);
493                 Pk2Go_SET_VDD();
494                 PK2Go_Mode = PK2GO_MODE_GO;
495                 asm_templ = 0;          // for Power LED blink
496                                         // response:    -
497     //            CalAndSetCCP1(0x11, 0x00); // about 1.8V
498     //            VddVppLevels.VddThreshold = CalThresholdByte(64);          // Set error th
499                 inbuffer[0] = ~GETVERSION;
500             }
501
```

7.3.3.1.1.6. ábra: Programmer – To – Go funkció esetén V_{DD} alapállapot biztosítása

A 7.3.3.1.1.6. ábrán láthatjuk, hogy egy új – az eredeti firmware-ben nem szereplő – függvényhívás történik. A PK2Go_SET_VDD() függvény a belső EEPROM-ból kiolvassa az előre megadott VDD feszültségértéket és ezt állítja be. Erre azért volt szükség, mert a funkcióba való belépést a PICkit2 hardveregység a V_{DD} be- és kikapcsolgatásával jelzi. Azért, hogy a 3V3 rendszerű PIC mikrovezérlőket ne tegye tönkre a firmware a V_{DD} feszültségértékét a target LED villogtatásához visszaveszi bekapcsoláskor 3,3V, majd a későbbi működés során 1,8V értékűre[142].

A probléma ott adódik, hogy ha nem csak a PIC mikrovezérlőt, hanem egy azt tartalmazó panelt szeretnénk céláramkörnek választani. Amennyiben a panelen minimális pufferelem (pl. 100µF) ki van alakítva a 3,3V, ill. az 1,8V már kevés egy 5V-ra tervezett áramkör tápellátására. A módosítás következtében a PICkit2 a Programmer – To – Go funkcióba történő belépéskor az előre megadott értékű V_{DD} feszültséggel fogja villogtatni a target LED-et. Ezzel a változtatással előáll az a veszély, hogy ha rossz eszközt és feszültségértéket választottunk ki a beállítások során, akkor nagyobb feszültséget adunk a panelre mint amit elviselne. Mivel ennek esélye korrekt használat esetén elhanyagolható, ezért úgy ítéltük meg, hogy a módosítás elfogadható kockázattal jár[143].

Mindemellett a 7.3.4. fejezetben láthatunk egy alternatívát erre a problémára.

142 Ennek elkerülése végett a CallAndSetCCP1() függvény, ill. az utána lévő utasítás kikommentezésre került.

143 A PICkit2 firmware-je 125µs-onként ellenőrzi a V_{DD} feszültség értékét, és meghatározott eltérés esetén lekapcsolja azt.

```
847
848                 caseENTER_UART_MODE:
849                     ICSP_HiZ = 1;       // Javítás: analóg kapcsoló be
850                     // format:      0xB3
851                     // response:    -
852                     usb_idx++;
853                     EnterUARTMode(&usb_idx);
854                     _asm
855                         bra     JumpTableEnd
856                     _endasm
857
858                 caseEXIT_UART_MODE:         // Exits the firmware from UART Mode
859                     // format:      0xB4
860                     // response:    -
861                     usb_idx++;
862                     ExitUARTMode();
863                     ICSP_HiZ = 0;       // Javítás: analóg kapcsoló ki
864                     _asm
865                         bra     JumpTableEnd
866                     _endasm
867
```

7.3.3.1.1.7. ábra: UART módban történő működés esetén a 4066 kapcsolása

A 7.3.3.1.1.7. ábrán láthatjuk, hogy az UART módba történő belépéskor a 4066 analóg kapcsolót be, kilépéskor pedig ki kell kapcsolnunk.

```
894
895                 caseLOGIC_ANALYZER_GO:
896                     // format:      0xB8<EdgeRising><TrigMask><TrigStates><EdgeMask>
897                     //                  <TrigCount><PostTrigCountL><PostTrigCountH><Sam
898                     // response:    <TrigLocL><TrigLocH>
899                     usb_idx++;
900                     asm_temp12 = inbuffer[usb_idx++];
901                     asm_temp1 = inbuffer[usb_idx++];
902                     asm_temp3 = inbuffer[usb_idx++];
903                     asm_temp5 = inbuffer[usb_idx++];
904                     asm_temp6 = inbuffer[usb_idx++];
905                     asm_temp7 = inbuffer[usb_idx++];
906                     asm_temp8 = inbuffer[usb_idx++] + 1;
907                     asm_temp11 = inbuffer[usb_idx++];
908                     ICSP_HiZ = 1;       // Javítás: analóg kapcsoló be
909                     LogicAnalyzer();
910                     ICSP_HiZ = 0;       // Javítás: analóg kapcsoló ki
911                     _asm
912                         bra     JumpTableEnd
913                     _endasm
914
```

7.3.3.1.1.8. ábra: A 4066 vezérlése logikai analizátor üzemmódban

A 7.3.3.1.1.8. ábrán láthatjuk, hogy a logikai analizátor mód is magával vonzza a 4066 analóg kapcsoló működtetését.

```
978           PK2GoExecute();
979           Vdd_TGT_P = 1; // VDD off
980     //      Vdd_TGT_N = 1; // VDD GND on      Javítás
981           while(!PROG_SWITCH_pin){}; // wait for button to be released.
982           asm_templ = 0;          // for Power LED blink
983     //      CalAndSetCCP1(0x11, 0x00); // about 1.8V
984     //      VddVppLevels.VddThreshold = CalThresholdByte(64);          // Set error thre
985           // look for VDD/VPP error(s)
986           if (Pk2Status.StatusLow & 0x30)
```

7.3.3.1.1.9. ábra: VDD 1,8V-ra történő csökkentésének elhagyása

A 7.3.3.1.1.9. és a 7.3.3.1.1.10. ábrán látható módosítások okairól már beszéltünk. A képeken a Programmer – To – Go funkció esetén 1,8V-ra visszavett V_{DD} feszültség kikommentezését láthatjuk. A 7.3.3.1.1.9. ábrán mindemellett kiszedtük a VDD_TGT_N láb kapcsolását is.

```
1105      if (CheckKeySequence(usbindex))
1106      {
1107          PK2Go_Mode = PK2GO_MODE_GO;
1108          asm_templ = 0;          // for Power LED blink
1109                                  // response:     -
1110    //      CalAndSetCCP1(0x11, 0x00); // about 1.8V
1111    //      VddVppLevels.VddThreshold = CalThresholdByte(64);          // Set error thresh
1112
1113          // save key to EEPROM so unit powers up in PK2GO
1114          EE_WriteByte(PK2GO_KEY1, 0x50);
1115          EE_WriteByte(PK2GO_KEY2, 0x4B);
1116          EE_WriteByte(PK2GO_KEY3, 0x32);
1117          // save pk2go_memsize
1118          pk2go_memsize = inbuffer[usbindex + 3];
1119          EE_WriteByte(PK2GO_MEM, pk2go_memsize);
```

7.3.3.1.1.10. ábra: VDD 1,8V-ra történő csökkentésének elhagyása

```
E:\Doksi\PICkit2_Firmware_V10.0\FirmwareV2\user\pickit.c
2185  caseVDD_ON:             //Javítás, hogy a panelen lévő kondit fel tudjuk tölteni
2186      VppVddADCTMR1_Stop();Vdd_TGT_P = 0;Delay10KTCYx(1000);VppVddADCTmr1_Start(); //Javítás eredetile
2187      //scriptindex++;
2188      _asm
2189          bra ScriptIdxIncJumpEnd
2190      _endasm
2191
```

7.3.3.1.1.11. ábra: V_{DD} figyelés kikapcsolása a bekapcsolási jelenségek lefutásáig

A 7.3.3.1.1.12. ábrán láthatjuk, hogy a V_{DD} FET-el történő földre kapcsolását kikommenteztük, hiszen ez a pont egy ellenálláson keresztül már stabil földre van kötve.

```
2198
2199            caseVDD_GND_ON:
2200        //      Vdd_TGT_N = 1;          // Javítás, ez a klónáramkörben nem szerepel
2201                //scriptindex++;
2202                _asm
2203                    bra ScriptIdxIncJumpEnd
2204                _endasm
2205
2206            caseVDD_GND_OFF:
2207        //      Vdd_TGT_N = 0;          // Javítás, ez a klónáramkörben nem szerepel
2208                //scriptindex++;
2209                _asm
2210                    bra ScriptIdxIncJumpEnd
2211                _endasm
2212
```

7.3.3.1.1.12. ábra: V_{DD} láb FET-el való földre húzásának kikommentezése

A 7.3.3.1.1.11. ábrán látható, hogy a V_{DD} bekapcsolása előtt kikapcsoljuk a V_{DD} figyelést, majd egy várakozási rutin után ismét bekapcsoljuk. Erre azért van szükség, hogy egy panelen lévő pufferkondenzátort is fel tudjunk tölteni. A kondenzátor a feszültség rákapcsolásának pillanatában rövidzárként viselkedik, és a PICkit2 V_{DD} error hibaüzenettel lekapcsolja a tápfeszültséget. Ennek megakadályozására bekapcsoláskor egy kis időt adunk az áramkörnek, majd folytatjuk a V_{DD} monitorozását.

```
2213            caseVPP_ON:
2214                ICSP_HiZ = 1;       // Javítás: analóg kapcsoló be
2215                Vpp_ON = 1;
2216                //scriptindex++;
2217                _asm
2218                    bra ScriptIdxIncJumpEnd
2219                _endasm
2220
2221            caseVPP_OFF:
2222                ICSP_HiZ = 0;       // Javítás: analóg kapcsoló ki
2223                Vpp_ON = 0;
2224                //scriptindex++;
2225                _asm
2226                    bra ScriptIdxIncJumpEnd
2227                _endasm
```

7.3.3.1.1.13. ábra: 4066 vezérlése a V_{PP} függvényében

A PGC és PGD lábakat a programozás/hibakeresés idejére csatlakoztatnunk kell a céláramkörhöz a 4066 analóg kapcsoló segítségével. A programozási, vagy hibakeresési módba való belépésről a V_{PP} láb üzemeltetése nyújt számunkra információt. A 7.3.3.1.1.13. ábrán látható ennek fényében a 4066 vezérlése.

```
3577    ********************************************************************************/
3578    int MeasurePulse(void)
3579    {
3580        BOOL interrupts_on = 0;
3581        int retvalue;
3582        ICSP_HiZ = 1;       // Javítás: analóg kapcsoló be
3583        INTCONbits.TOIF = 0; //clear flag
3584        TMR0H = 0x80;   // roll over in 699 ms
3585
3586        TMR0L = 0;
3587        T0CONbits.TMR0ON = 1;
3588
3589        if (INTCONbits.GIE == 1)
3590            interrupts_on = 1;
3591        INTCONbits.GIE = 0;         // uninterruptable routine
```

7.3.3.1.1.14. ábra: Impulzusmérő függvény kezdete

A 7.3.3.1.1.14. és a 7.3.3.1.1.15. ábrán látható függvény egy maximum 700ms ideig tartó impulzus mérését végzi a PGD lábon – ahhoz hogy ezt végre tudja hajtani a 4066 kapcsolót a függvény elején be, a végén pedig ki kell kapcsolnunk.

```
3629        T0CONbits.TMR0ON = 0;   // shut off timer
3630
3631        if (interrupts_on == 1)     // turn interrupts back on if enabled.
3632            INTCONbits.GIE = 1;
3633
3634        retvalue = TMR0L;
3635        retvalue += ((TMR0H & 0x7F) * 0x100); // mask off set MSb
3636        ICSP_HiZ = 0;       // Javítás: analóg kapcsoló ki
3637        return retvalue;
```

7.3.3.1.1.15. ábra: Impulzusmérő függvény vége

A 7.3.3.1.1.16. és a 7.3.3.1.1.17. ábrán látható függvények az I^2C-n kommunikáló EEPROM-ok felprogramozásához szükséges kommunikációt indítják, ill. állítják le – a javítás itt is a 4066 vezérlését szolgálja.

```
4260    ********************************************************************************/
4261    void I2C_Start(void)
4262    {   ICSP_HiZ = 1;       // Javítás: analóg kapcsoló be
4263        ICSPCLK_out = 1;            // SCL high
4264        Delay10TCYx(6);             // delay 5us
4265        AUX = 0;                    // ensure LAT bit is zero
4266        tris_AUX = 0;               // SDA low
4267        Delay10TCYx(6);             // delay 5us
4268        ICSPCLK_out = 0;            // SCL Low
4269        Delay10TCYx(6);             // delay 5us
4270        tris_AUX = 1;               // SDA released
4271    }
4272
4273    /*******************************************************************************
```

7.3.3.1.1.16. ábra: 4066 bekapcsolása az I2C START bit generáláskor

```
4287    ********************************************************************************/
4288    void I2C_Stop(void)
4289    {   ICSPCLK_out = 0;            // SCL low
4290        Delay10TCYx(6);             // delay 5us
4291        AUX = 0;                    // ensure LAT bit is zero
4292        tris_AUX = 0;               // SDA low
4293        Delay10TCYx(6);             // delay 5us
4294        ICSPCLK_out = 1;            // SCL high
4295        Delay10TCYx(6);             // delay 5us
4296        tris_AUX = 1;               // SDA released
4297        ICSP_HiZ = 0;       // Javítás: analóg kapcsoló ki
4298    }
4299
4300    /*******************************************************************************
```

7.3.3.1.1.17. ábra: 4066 bekapcsolása az I2C STOP bit generáláskor

A 7.3.3.1.1.18. és a 7.3.3.1.1.19. ábrán látható függvények az SPI kommunikációt biztosítják az AUX és PGD lábakon. Itt is a 4066 működtetésére szolgáló utasításokat szúrtuk be.

```
4752    unsigned char SPI_ReadWrite(unsigned char outputbyte)
4753    {
4754        //BOOL interrupts_on = 0;
4755        char i;
4756        ICSP_HiZ = 1;       // Javítás: analóg kapcsoló be
4757        //if (INTCONbits.GIE == 1)
4758        //  interrupts_on = 1;
4759        //INTCONbits.GIE = 0;           // uninterruptable routine
4760
4761        asm_temp1 = outputbyte;         // read byte is shifted in here as well
4762        asm_temp2 = 8;
```

7.3.3.1.1.18. ábra: 4066 bekapcsolása az SPI kommunikáció indításakor

```
4815
4816        //if (interrupts_on == 1)       // turn interrupts back on if enabled.
4817        //  INTCONbits.GIE = 1;
4818
4819        AUX = 0;        // leave low
4820        ICSP_HiZ = 0;       // Javítás: analóg kapcsoló ki
4821        return asm_temp1;
4822
4823    }
4824
4825    /*******************************************************************************
```

7.3.3.1.1.19. ábra: 4066 kikapcsolása az SPI kommunikáció befejezésekor

A PICKIT2.ASM fájlban assembly rutinokat találunk, amelyeket általában a sebességoptimalizálás, vagy az egyszerűbb értelmezés végett alkalmaztak.

```
25      JTAG2W4PH:
26          bsf     LATC, 7, 0      ;// Javítás: 4066 analóg kapcsoló be
27          bsf     LATA, 3, 0      ;// CLK high phase 1
28          bcf     LATA, 2, 0      ;// TDI = 0?
29          btfsc   ASM_TEMP1_RAM, 0, 0 ;// Test for TDI value
30          bsf     LATA, 2, 0      ;// TDI = 1
31          bcf     TRISA, 2, 0     ;// Output TDI (PGD = output)
32          nop
33          bcf     LATA, 3, 0      ;// CLK low Phase 1
34          btfss   ASM_TEMP2_RAM, 0, 0 ;// Test for TMS value low
35          bcf     LATA, 2, 0      ;// TMS = 0
36          bsf     LATA, 3, 0      ;// CLK high phase 2
37          btfsc   ASM_TEMP2_RAM, 0, 0 ;// Test for TMS value high
38          bsf     LATA, 2, 0      ;// TMS = 1
39          bcf     ASM_TEMP3_RAM, 7, 0 ;// TDO = 0?
40          bcf     LATA, 3, 0      ;// CLK low Phase 2
41          bsf     LATA, 3, 0      ;// CLK high phase 3
42          bsf     TRISA, 2, 0     ;// Input TDO (PGD = input)
43          bcf     LATA, 3, 0      ;// CLK low Phase 3
44          bsf     LATA, 3, 0      ;// CLK high phase 4
45          btfsc   PORTA, 2, 0     ;// Test for TDO value
46          bsf     ASM_TEMP3_RAM, 7, 0 ;// TDO = 1
47          bcf     LATA, 3, 0      ;// CLK low Phase 4
48          bcf     LATC, 7, 0      ;// Javítás: 4066 analóg kapcsoló ki
49          return
```

7.3.3.1.1.20. ábra:
4066 analóg kapcsoló működtetése PIC32 JTAG primitív esetén

A 7.3.3.1.1.20. ábrán láthatjuk, hogy a 32-es sorozatú PIC mikrovezérlők programozásához egy JTAG[144] assembly rutint alkalmaztak. A helyes működés megköveteli, hogy itt is csatlakoztassuk, majd leválasszuk a PGC, PGD lábakat.

144 A JTAG a boundary scan egyik lehetséges megvalósítási protokollja – jóval fejlettebb tesztelési és programozási eljárás, mint az ICSP.

A PK_PROG2GO.C fájl tartalmazza a Programmer – To – Go funkció működtetéséhez szükséges függvényeket.

```
76      void ReadOSCCAL(unsigned char addr_l, unsigned char addr_h);
77      void WriteOSCCAL(unsigned char addr_l, unsigned char addr_h);
78      void CalcUploadChecksum(void);
79      char CheckDeviceID(void);
80      void ReadBandGap(void);
81      void WriteCfgBandGap(void);
82      void Pk2Go_SET_VDD(void);
83      /** D E C L A R A T I O N S ************************************
```

7.3.3.1.1.21. ábra: PkGo_SET_VDD() függvény deklarálása

A 7.3.3.1.1.21. ábrán láthatjuk a Programmer – To – Go funkció esetén meghívott VDD beállító függvény (l. 104. o.) deklarációját, a 7.3.3.1.1.22. ábrán pedig a függvény definíciót.

```
1164    void Pk2Go_SET_VDD(void)
1165    {
1166        unsigned char temp;
1167        unsigned char Next_cmd;
1168        PK2GoInit();
1169        eebuffer_idx = 64; // force a read on first access
1170        while(1)
1171        {if((Next_cmd=RdByteExtEE())==SETVDD)
1172        {
1173                    // format:      0xA0 <CCPL><CCPH><VDDLim>
1174                    //      CCPH:CCPL = ((Vdd * 32) + 10.5) << 6     where Vdd is de
1175                    //      VDDLim = (Vfault / 5) * 255              where Vdd < VFa
1176                    // response: -
1177                    temp = RdByteExtEE();
1178                    CalAndSetCCPl(RdByteExtEE(), temp);
1179                    VddVppLevels.VddThreshold = CalThresholdByte(RdByteExtEE());
1180                    break;
1181
1182        }
1183        if(Next_cmd== END_OF_BUFFER) break;
1184        }
1185    }
```

7.3.3.1.1.22. ábra: PkGo_SET_VDD() függvény definíciója

7.3.3.1.2 PICkit2Bootloader projekt módosítása

A bootloaderben a BOOT_MAIN.C fájlt kellett megváltoztatnunk.

```
38
39      /** C O N F I G U R A T I O N   B I T S ******************************************/
40
41      #pragma config PLLDIV = 5, CPUDIV = OSC1_PLL2, USBDIV = 2                  // CONFIG1L
42      #pragma config FOSC = HSPLL_HS, FCMEN = OFF, IESO = OFF                    // CONFIG1H     //Javítás:
43      #pragma config PWRT = ON, BOR = ON, BORV = 2, VREGEN = ON                  // CONFIG2L     //Javítás:
44      #pragma config WDT = OFF, WDTPS = 32768                                    // CONFIG2H
45      #pragma config MCLRE = ON, LPT1OSC = OFF, PBADEN = OFF, CCP2MX = ON       // CONFIG3H     //Javítás:
46      #pragma config STVREN = ON, LVP = OFF, XINST = OFF, DEBUG = OFF            // CONFIG4L     //Javítás:
47      #pragma config CP0 = OFF, CP1 = OFF, CP2 = OFF, CP3 = OFF                  // CONFIG5L
48      #pragma config CPB = OFF, CPD = OFF                                        // CONFIG5H
49      #pragma config WRT0 = ON, WRT1 = OFF, WRT2 = OFF, WRT3 = OFF               // CONFIG6L
50      #pragma config WRTB = ON, WRTC = OFF, WRTD = OFF                           // CONFIG6H
51      #pragma config EBTR0 = OFF, EBTR1 = OFF, EBTR2 = OFF, EBTR3 = OFF          // CONFIG7L
52      #pragma config EBTRB = OFF                                                 // CONFIG7H
53
```

7.3.3.1.2.1. ábra: Konfigurációs bitek módosítása

Első lépésként a 7.3.3.1.2.1. ábrán látható konfigurációs biteket javítottuk. A szükségesnek ítélt módosítások:

- Az eredeti firmware-ben FCMEM szerepelt, ezt az alkalmazott fordító miatt FCMEN-re kellett változtatnunk.
- A Brown Out Reset-et a stabilabb működés érdekében engedélyeztük.
- A Master Clear Reset-et engedélyeztük, mert a klónáramkör kiegészült egy nyomógombbal, amivel reszetelni lehet az eszközt.
- Az ICPRT bitbeállítást kiszedtük, erre szintén a fordító miatt volt szükség.

```
118     // Check Bootload Mode Entry Condition
119     //----------------------------------------
120
121     tris_PROG_SWITCH = 1;           // RB4 Input
122 //    PROG_SWITCH = 1;                // initialize port to 1   //Javítás: bemenetre nem írunk rá
123     INTCON2bits.NOT_RBPU = 1;       // initialize PORTB pull-ups on     //Javítás:0->1 az áramkör külso felh
124
125     Delay10KTCYx(600);              // delay ~500 ms
126
127     if (PROG_SWITCH_pin) {         // is the push button pressed?   //Javítás: eredetiben: PROG_SWITCH (igaz
128
129         // if no, does program memory location 0x7FFE contain 0x55?
130
131         Temp = * ((rom far char *) 0x7FFE);
132
133        if(Temp != 0x00) {                      //Javítás: itt eredetileg ==0x55 volt, de az hibás, hiszen a
134                                                //EnterBootloader() fgv. a 0x7FFE memóriarekszt törli, egyébl
135             _asm goto RM_RESET_VECTOR _endasm  //és ide akkor kell ugrani, ha nem volt gomb, és nem volt En
136                                                //az if(Temp) rövidebb írásmódot is lehetne alkalmazni
137         }
138
139     } // end if (PROG_SWITCH)
140
```

7.3.3.1.2.2. ábra: Bootloader módba történő belépés feltételvizsgálatának módosítása

A 7.3.3.1.2.2. ábrán láthatjuk, hogy a bootloader módba való belépésen is változtatást eszközöltünk. A 122. sorban kikommenteztük a bemenetre történő ráírást. A 127. sorban szintaktikai hiba volt, az eredeti verzióban PROG_SWITCH szerepelt, de az IO_CFG.H fájlban a nyomógombot PROG_SWITCH_PIN-ként adtuk meg, ezért erre módosítottunk itt is.

A 133. sorban egy feltételvizsgálatot láthatunk. A letöltőprogramot kétféleképpen tudjuk elindítani. Egyik lehetőség, hogy a nyomógombot lenyomva tartva csatlakoztatjuk a PICkit2-őt a PC-hez, ekkor a 127. sorban lévő feltétel nem teljesül[145] és tovább megyünk a bootloader programban – ellenkező esetben megvizsgáljuk a másik lehetőséget. A PICkit2 szoftveréből egy parancs segítségével is a bootloader módba léphetünk (*Tools → Download PICkit 2 Operating System*). A parancs hatására a PICkit 2 firmware-e kitörli az utolsó memóriarekesz tartalmát (0x7FFE), majd reszeteli az áramkört. Amikor tehát elindul a program meg kell néznünk, hogy ebben a memóriarekeszben nulla van-e[146], amennyiben igen akkor bootloader módba kell lépnünk, ha nem akkor a normál működés esete áll fenn (GOTO RM_RESET_VECTOR). Az eredeti verzióban 0x55-öt vizsgált a program, ami helytelen, ezt módosítottuk 0x00-ra.

145 A nyomógomb nullába húz (aktív nullás eszköz).

146 Normál működés esetén ezt a rekeszt nem használjuk, mivel a programmemóriában eddig nem foglalunk helyet felprogramozásnál 0xFF értékű marad.

```
140
141         //---------------------------------------
142         // Bootloader Mode
143         //---------------------------------------
144
145         InitializeSystem();
146         USBCheckBusStatus();            //Javítás ez nem volt benne
147         while(1) {
148
149           //  USBTasks();                 // USB Tasks Javítás, ez benne volt
150             USBDriverService();         //Javítás ez nem volt benne
151             BootService();              // See boot.c
152
153             if (Counter == 0)           // blink Busy LED
154                 BUSY_LED ^= 1;
155
156             Counter--;
157
158         }
```

7.3.3.1.2.3. ábra: USB kommunikációs eljárás módosítása

A 7.3.3.1.2.3. ábrán látható módosításokat az USB kommunikáció helyes működése végett eszközöltük.

7.3.3.2 A firmware-n történő módosítások elvégzése MPLAB környezet alatt

A gyári firmware-t két külön projektből állították össze az Microchip mérnökei. Az egyik projekt tartalmazza a pickit 2 funkcióit ellátni képes forrásokat, míg a másik projektben a bootloader[147] kapott helyet.

Az MPLAB *Configure → Settings* menüpontjának *Projects* fülénél vegyük ki a pipát a *Use one-to-one project-workspace model* opcióból! Így már tudunk egyszerre több projektet is kezelni a munkaasztalunkon. A Program Loading fülnél tudjuk beállítani, hogy a projektünk fordítása előtt (*Clear memory before building a project*), vagy sikeres fordítása után (*Clear memory after succesfully building a project*) melyik memóriaterületet törölje a fejlesztőrendszer. Ügyeljünk rá, hogy egyik lehetőség se legyen bepipálva, hiszen pl. amennyiben a programmemóriát töröljük a projekt fordítása előtt, akkor mindig csak az egyik projekt tartalma jelenik meg a memóriában amit majd letöltünk a mikrovezérlőbe[148].

A *Configure → Select Device...* menüpontban válasszuk ki a PIC18F2550 típusú mikrovezérlőt!

Nyissuk meg a Microchip honlapjáról letöltött PICkit2_FWv2.mcp és a PICkit2Bootloader.mcp nevű projekteket! A fordításhoz használjuk a Microchip C18 fordítóját[149] (*Project → Select Language Toolsuite...*). Amennyiben szükséges adjuk meg a fordítóeszköz komponenseinek elérési útját a *Project → Set Language Tool Locations...* menüpont segítségével. A *Project → Build Options...* menüpont *Project* fülében állítsuk be a könyvtárakat! A libary és a linker elérési útjaként adjuk meg az XXX\MCC18\LIB és az XXX\MCC18\LKR könyvtárakat. Az include fájlokhoz rendeljük hozzá az XXX\MCC18\H és a firmware fő könyvtárának (XXX\FirmwareV2) elérési útját. A *Project → Build Configuration* menüpontban válasszuk a *Release* opciót! A projektet mentsük el, majd a *Project Set Active* projekt fülnél válasszuk ki a másik projektünket és ismételjük meg a folyamatot.

Tegyük aktívvá a PICkit2_FWv2.mcp nevű projektet, végezzük el a szükséges módosításokat, majd a Ctrl+F10 billentyűkombinációval fordítsuk le. A PICkit2Bootloader.mcp nevű projekt aktívvá tétele és a változtatások elvégzése után ismét alkalmazzuk a Ctrl+F10 billentyűkombinációt a fordításhoz. **A sorrend fontos**, mivel két main() függvény szerepel a fordító mindkettőt a nullás címre kívánja fordítani, ott pedig a bootloader helye van – vagyis midig a letöltőprogram projektjét kell utoljára fordítanunk! A programmemóriában szerepelnie kell mindkét projekt eredményének. A *File → Export...*[150] menüpont segítségével a programmemóriából egy hex fájlba tölthetjük a programunkat, amit már bármilyen égetőprogram tud kezelni.

147 Bootloader: letöltő program. Lehetséges a mikrovezérlőbe olyan programrészletet írni, ami – felhasználói beavatkozásra – valamilyen kommunikációs protokollon (pl. USB) keresztül érkező információval tölti fel a programmemóriát (kivéve a letöltőprogram által elfoglalt memóriaterületet). Egy ilyen letöltőprogram nagyban megkönnyíti a fejlesztést, ill. a későbbi update-eket, hiszen nem kell majd egy külön programozó áramkör a firmware frissítéséhez.

148 Igazából számunkra itt most csak a *Clear program memory upon loading a program* lehetőség fontos hogy ne legyen bejelölve. Azonban célszerű, ha a projektek fordításánál se a konfigurációt, se a felhasználói azonosítót, se az EEPROM adatmemóriát nem töröljük.

149 Szintén a Microchip honlapjáról tudjuk letölteni a fordításhoz szükséges MCC18 nevű komponenst.

150 A teljes programmemóriát, a konfigurációs bitekkel, és felhasználói azonosítóval exportáljuk INHX32 fájlformátumba.

7.3.4 Változtatás a klónáramkörön

A 104. oldalon már ismertetett problémát egyszerűen megoldhatjuk, ha egy külön LED-et csatlakoztatunk a mikrovezérlőhöz, és ennek segítségével jelezzük a Programmer – To – Go funkciót. Hozzá kell tennünk, hogy ez így kulturáltabb is mint az eredeti verzió, hiszen státuszjelentésre nem túl korrekt a tápfeszültséget felhasználni. A módosítások elvégzéséhez meg kell változtatni a nyáktervet, és a firmware-t is. Ezeket a módosításokat a dokumentáció elkészítésekor részben már elvégeztük, mind a módosított nyáktervet, mind az új firmware-t a http://plc.mechatronika.hu, vagy a www.picnick.hu weboldalról le lehet majd tölteni. A módosításokról egy külön rövid leírást is készítünk.

8 Irodalomjegyzék

- http://plc.mechatronika.hu
- www.alldatasheet.com
- www.microchip.com

- Pickit2 User Guide
- ICSP User Guide
- PIC18F2550 adatlap