

Programfejlesztés PIC mikrovezérlőkre I.

Írta: Molnár Zsolt

Tartalomjegyzék

1. Az MPLAB integrált fejlesztői környezet általános ismertetése.....	3
2. Mintafeladatok megoldásának ismertetése.....	15
2.1. példa	15
2.2. példa	22
3. Mérési feladatok.....	28
4. Házi feladat	29

1. Az MPLAB integrált fejlesztői környezet általános ismertetése

Az **MPLAB IDE** a **Microchip** cég **PIC mikrovezérlői**hez készült ingyenes **integrált fejlesztői környezet** (Integrated Design Environment).

A rendszer tartalmaz egy **kezelőfelületet**, **szövegszerkesztőt**, képes **kezelni** a Microchip cég saját **fordítóit** (MPASM, Microchip C), valamint sok más cég által gyártott fordítót. Ezen felül tartalmaz egy viszonylag fejlett **szimulátort**, képes lekezelni többféle **programozó** és **hibakereső** eszközt (emulátor, debugger). Telepítéstől függően tartalmaz még néhány (főként a témával ismerkedők számára hasznos) beépített **segédprogramot** (pl. grafikus programváz generátor).

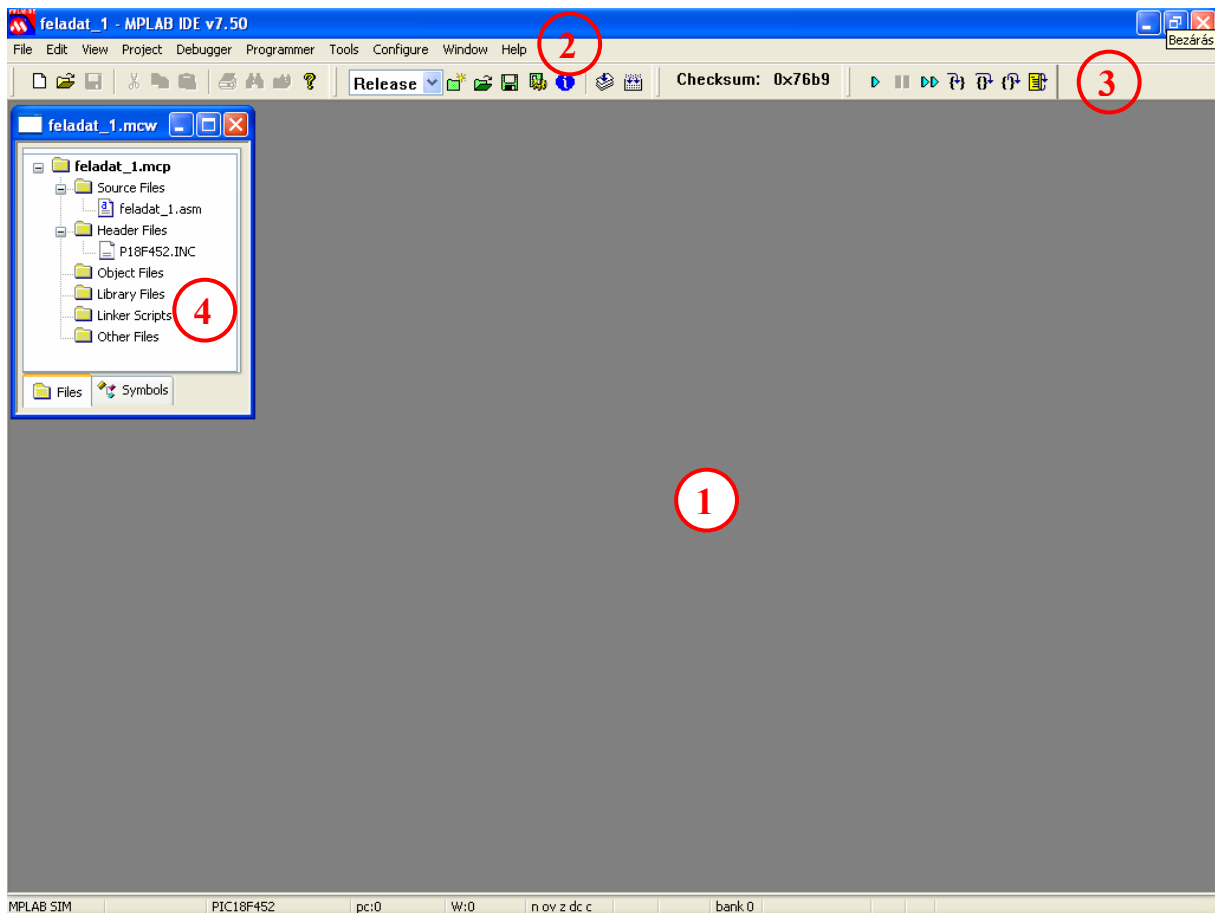
A segédlet megírásakor az MPLAB legfrissebb elérhető verziója a 7.50-es. (A fejlesztőrendszer állandó bővítés és fejlesztés alatt van.)

A mérési segédlet **bevezetést** kíván nyújtani – a PIC mikrovezérlők programozásában korábban megszerzett ismeretekre alapozva – az **MPLAB használatába**, főként a szimulációs és hibakeresési lehetőségekbe. A mérés sikeres elvégzését nagymértékben segíti a mérést előkészítő előadások anyagának elsajátítása.

Az MPLAB IDE **elindítása** az asztalon lévő ikon segítségével, vagy a *Start* menüből a Microchip programcsoportból lehetséges. Ügyeljen rá, hogy ne az esetlegesen telepített más verziót indítsa, hanem a 7.50-est!

A mérés során a 18F452 típussal dolgozunk, mivel a későbbiekben ezt a típust tartalmazó demonstrációs panelt fogjuk használni.

Az MPLAB IDE program **kezelőfelülete** az alábbi ábrán látható:



A felület **4 alapvető területre** bontható, amelyeket számokkal jelöltünk:

1. Munkafelület
2. Menüsor
3. Kiemelt parancsikonok (eszköztárak)
4. Projekt navigációs ablak






A **munkafelületen** nyithatjuk meg az ablakokat, amelyekben például a kódot szerkeszthetjük, a regiszterek tartalmát jeleníthetjük, stb.

A **menüsorból** minden lehetséges parancs a szokásos módon kategorizálva elérhető, de a leggyakrabban használt parancsok a **kiemelt parancsikonok** között, az eszköztárakban is megtalálhatóak.







A **projekt navigációs ablak** kiemelt szerepű a többi ablak mellett, ebben láthatjuk a projektünk felépítését, innen nyithatjuk meg a projekt fájljait, illetve egyéb információt ad.

A következőkben röviden a **menüket** ismertetjük.

File (fájl) menü:

- *New* (*Ctrl+N*, ): új fájl létrehozása és megnyitása egy ablakban.
- *Add New File to Project...*: létező fájl hozzáadása a megnyitott projekthez.
- *Open...* (*Ctrl+O*, ): fájl megnyitása. A felugró ablakban szűrőként megadható a megnyitni kívánt fájl típusa.
- *Close*: a kiválasztott (aktív) fájl bezárása.
- *Save* (*Ctrl+S*, ): a kiválasztott (aktív) fájl mentése.
- *Save As...*: a kiválasztott (aktív) fájl mentése más néven.
- *Save All*: minden megnyitott fájl mentése.
- *Open Workspace...*: létező munkafelület megnyitása.
- *Save Workspace* (): az aktív munkafelület mentése.
- *Save Workspace As...*: az aktív munkafelület mentése más néven.
- *Close Workspace*: aktív munkafelület bezárása.
- *Import*: memóriatartalom (.hex) vagy hibakereséssel kapcsolatos információkat tartalmazó fájl (.cod, .cof, .elf) betöltése.
- *Export*: memóriatartalom mentése (.hex). A felugró ablakban többek között beállítható, hogy melyik típusú memóriát (pl. program, EEPROM, konfigurációs) szeretnénk menteni.
- *Print* (*Ctrl+P*, ): nyomtatás, és nyomtatási beállítások.
- *Recent Files*: a legutóbb megnyitott fájlok listája, amelyből egy kattintással megnyithatjuk azokat.
- *Recent Workspaces*: a legutóbb megnyitott munkafelületek listája, amelyből egy kattintással megnyithatjuk azokat.
- *Exit*: kilépés a programból.



Edit (szerkesztés) menü:




- *Undo* (*Ctrl+Z*): a legutóbbi művelet visszavonása.
- *Redo* (*Ctrl+Y*): a legutóbb visszavont művelet visszaállítása.
- *Cut* (*Ctrl+X*, ): a kijelölt szöveg kivágása.
- *Copy* (*Ctrl+C*, ): a kijelölt szöveg vágólapra másolása.
- *Paste* (*Ctrl+V*, ): a vágólap tartalmának beillesztése.
- *Delete* (*Del*): a kijelölt szöveg törlése.
- *Select All* (*Ctrl+A*): az aktív fájl teljes tartalmának kijelölése.
- *Find...* (*Ctrl+F*, ): szöveg keresése az aktív fájlban.
- *Find Next* (*F3*): a *Find* parancsnál megadott szöveg ismételt keresése.
- *Find in Files...* (*Ctrl+Shift+F*, ): a projekt fájljainak mindegyikében keresi a megadott szöveget.
- *Replace...* (*Ctrl+H*): megadott szöveg helyettesítése más szöveggel.
- *Go To...* (*Ctrl+G*): ugrás adott számú sorhoz vagy címkéhez.
- *Advanced*: emelt szintű szerkesztési műveletek:
 - *Uppercase*: a kijelölt szöveg nagybetűsre változtatása.
 - *Lowercase*: a kijelölt szöveg kisbetűsre változtatása.
 - *Comment Block*: kijelölt szöveg megjegyzéssé alakítása (pontosvessző elhelyezése minden sor elején, így az nem kerül lefordításra).
 - *Uncomment Block*: a kijelölt szöveg sorai elejéről a megjegyzést jelző pontosvesszők eltávolítása.
 - *Indent Block*: a kijelölt szöveg minden sorának elejére elhelyez egy tabulátort.
 - *Outdent Block*: a kijelölt szöveg minden sorának elejéről kivesz egy tabulátort.
 - *Match* (*Ctrl+M*): megkeresi a sorban a kurzorpozíciónál lévő zárójel „párját”
- *Bookmarks*: könyvjelzők kezelése:
 - *Toggle Bookmark* (*Ctrl+K*): arra a sorra, amelyben a kurzor van, egy könyvjelző jelölést () tesz, vagy ha már volt rajta, leveszi.
 - *Next Bookmark* (*Ctrl+L*): a következő könyvjelzőhöz ugrik.
 - *Previous Bookmark* (*Ctrl+J*): az előző könyvjelzőhöz ugrik.
 - *Clear All Bookmarks*: az összes könyvjelzőt törli.
- *Properties...*: a szövegszerkesztő beállításai (pl. betűtípus, színek, tabulátor mérete...)

View (nézet) menü:

- *Project*: projektablak ki/bekapcsolása.
- *Output*: üzeneteket tartalmazó (output) ablak bezárása.
- *Toolbars*: eszköztárak megjelenítésének ki/bekapcsolása. (*Standard*: általános eszköztár, *Project Manager*: projekt eszköztár, *Checksum*: ellenőrző összeg, *Debug*: hibakeresési eszköztár.)
- *Call Stack*: dsPIC mikrovezérlőknél a Call Stack ablakot nyitja meg.
- *Disassembly Listing*: megnyitja a diszasszemblált lista fájlt, amelyben látható, hogy a fordító a szövegfájlból (forrás) milyen kódot generált.
- *EEPROM*: megnyitja az EEPROM tartalmat kijelző ablakot.
- *File Register*: megnyitja a regiszterfájl ablakát.
- *Hardware Stack*: megnyitja a hardveres verem tartalmát kijelző ablakot.
- *LCD Pixel*: beépített LCD meghajtóval rendelkező típusoknál megmutatja, hogy az LCD egyes képpontjai milyen állapotban vannak.
- *Local*: magas szintű nyelveken (pl. C) megírt kód esetén mutatja az automatikusan létrehozott helyi változókat, és azok értékét. Assemblyben való programozáskor nincs jelentősége.
- *Program Memory*: megmutatja a programmemória tartalmát.
- *Special Function Register*: megmutatja a speciális funkciójú regiszterek tartalmát.
- *Watch*: megnyit egy felügyelő ablakot, amelyben a kiválasztott regiszterek és változók tartalmát követhetjük nyomon.
- *Memory Usage Gauge*: a memória felhasználásáról ad információt.
- *Simulator Trace*: megnyitja a nyomkövetés ablakot.
- *Simulator Logic Analyzer*: megnyitja a logikai analizátor ablakot.

Project (projekt) menü:

- *Project Wizard...*: megnyitja a projekt létrehozását segítő varázslót.
- *New...* (): új projektet hoz létre.
- *Open...* (): megnyit egy létező projektet.
- *Close*: bezárja az aktív projektet.


- *Set Active Project*: több projektet tartalmazó munkafelület esetén válthatunk a megnyitott projektek között.
- *Quickbuild*: projekt létrehozása, és linker használata nélkül fordít le egy assembly fájlt.
- *Clean*: törli a legutóbbi fordítás közben létrehozott fájlokat (.hex, .cod, .cof...), amelyek a következő fordításkor automatikusan ismét létrejönnek.
- *Build All* (*Ctrl+F10*, ): lefordítja a teljes projektet.
- *Make* (*F10*, ): a projekt azon fájljait fordítja csak le, amelyek megváltoztak az előző fordítás óta.
- *Build Options* (): az aktív projekt általános, illetve a fordítóval kapcsolatos beállításait végezhetjük el.
- *Save Project*: elmenti az aktív projektet.
- *Save Project As...*: elmenti más néven az aktív projektet.
- *Add Files to Project...*: létező fájlokat adhatunk hozzá az aktív projekthez.
- *Add New File to Project...*: új fájlt adhatunk hozzá az aktív projekthez. Ha már létező fájlnevet adunk meg, azt felülírja az új fájlal.
- *Remove File From Project*: a projekt fájljai közül távolíthatjuk el a kiválasztottakat.
- *Select Language Toolsuite...*: beállíthatjuk a projekthez rendelt fordítót.
- *Set Language Tool Locations...*: beállíthatjuk a projekthez rendelt fordító elérési útját.
- *Version Control...*: az esetlegesen a gépre telepített verzió-követő program elérési útját adhatjuk meg.





Debugger (hibakereső) menü:

- *Select Tool*: kiválaszthatjuk a hibakeresési üzemmódot (az alábbi lista tartalma függ a telepítéstől):
 - *None*: nincs hibakereső eszköz, csak a szövegszerkesztési és fordítási lehetőségek érhetőek el.
 - *MPLAB ICD 2*: ICD2 hibakereső eszközként való kiválasztása.
 - *MPLAB SIM*: hibakeresési eszközként a szimulátort állítja be.
 - Egyéb telepített hibakereső eszközök...
- *Clear Memory*: az MPLAB által használt különféle típusú memóriákat törölhetünk:







- *All Memory*: minden, az MPLAB által használt memóriatípus törlése.
- *Program Memory*: a programmemória törlése.
- *GPRs*: általános felhasználású regiszterek törlése (adatmemória).
- *EEPROM*: EEPROM memória törlése
- *Configurations Bits*: konfigurációs bitek alaphelyzetbe állítása.

A menüben valamely hibakeresési üzemmód kiválasztása után megjelennek a hibakeresési üzemmódtól függő menüpontok. **MPLAB ICD2** kiválasztása esetén megjelenő menüpontok:


- *Run* (F9, 9/29

- *Program* (): a céleszköz beprogramozása a hibakereséshez szükséges tartalommal.
- *Read* (): a céleszköz tartalmának visszaolvasása.
- *Read EEPROM* (): a céleszköz EEPROM memóriájának visszaolvasása.
- *Abort Operation*: az aktuális művelet megszakítása.
- *Connect* (): alaphelyzetbe állítja az ICD2-t, és az MPLAB megteremti vele a kapcsoltot.
- *Download ICD2 Operating System*: az ICD2 operációs rendszerének frissítése.
- *Settings...*: az ICD2-vel kapcsolatos beállításokat végezhetjük el.

MPLAB SIM szimulátor kiválasztása esetén megjelenő menüpontok:

- *Run* (F9, ): teljes sebességgel futtatja a programot töréspontig, vagy a futtatás leállításáig. A futtatás a programszámláló aktuális értékétől történik. Futtatás közben számos funkció le van tiltva. A megjelenített ablakok tartalma csak a futtatás befejezésekor frissülnek.
- *Animate* (): animált programfuttatás. Beállítható időközönként lépésenként hajtja végre a programot, minden lépés után frissül a megjelenített ablakok tartalma.
- *Halt* (F5, ): a program teljes sebességű vagy animált futtatásának leállítása.
- *Step Into* (F7, ): egyszerű léptetés, a program minden utasítását egyenként végrehajtja (a szubrutinok utasításait is).
- *Step Over* (F8, ): a program lépésenkénti végrehajtása úgy, hogy a szubrutinokat egy utasításként kezeli, azaz a meghívásuktól a visszatérésig egyben, teljes sebességű futtatással végrehajtásra kerülnek.
- *Step Out* (): kilépés a szubrutinból, az aktuális utasítástól a szubrutinból való visszatérésig teljes sebességű programfuttatás történik.
- *Reset*: alaphelyzetbe állítás
 - *MCLR Reset*: alaphelyzetbe állítást szimulál, amelyet a processzor MCLR lábán megjelenő alacsony szint okozna.
 - *Watchdog Timer Reset*: alaphelyzetbe állítást szimulál, amely a watchdog időzítő túlsordulása miatt következne be.
 - *Brown Out Reset*: alaphelyzetbe állítást szimulál, amely a tápfeszültség-kimaradást










detektáló áramkör miatt következne be.

- *Processor Reset* (F6, 

Programmer (programozó) menü:




- *Select Programmer*: programozó eszköz kiválasztása (az alábbi lista tartalma függ a telepítéstől):
 - *None*: nincs kiválasztott programozó eszköz.
 - *MPLAB ICD 2*: ICD2, mint programozó kiválasztva
 - Egyéb programozó eszközök...


MPLAB ICD 2 kiválasztása esetén megjelenő menüpontok:

- *Mplab ICD 2 Setup Wizard*: az ICD2 beállítását segítő varázsló elindítása.
- *Program* (): a céleszköz beprogramozása.
- *Read* (): a céleszköz tartalmának visszaolvasása.
- *Verify* (): a céleszköz tartalmának összehasonlítása az MPLAB memóriájának tartalmával.
- *Erase Part* (): a céleszköz törlése.
- *Blank Check* (): a céleszköz ürességének (törölt állapotának) ellenőrzése.
- *Read EEPROM* (): a céleszköz EEPROM memóriájának visszaolvasása.
- *Release from Reset* (): a céleszköz MCLR lábának felengedése alacsony logikai szintről (reset megszüntetése).
- *Hold in Reset* (): a céleszköz MCLR lábának alacsony logikai szinten tartása (reset állapot).
- *Abort Operation*: az aktuális művelet megszakítása.
- *Connect* (): alaphelyzetbe állítja az ICD2-t, és az MPLAB megteremti vele a kapcsolatot.
- *Download ICD2 Operating System*: az ICD2 operációs rendszerének frissítése.
- *Settings...*: az ICD2-vel kapcsolatos beállításokat végezhetjük el.

Tools (eszközök) menü:

A rendelkezésre álló telepített eszközök (az alábbi lista tartalma függ a telepítéstől):

- *MPLAB Macros*: lehetőség van a Windows makró funkcióinak használatára. Megjelenik a Macros menü:
 - *Record Macro* (): makró felvétele.
 - *Stop Recerding* (): makró-felvétel leállítása.
 - *Play Macro* (): makró lejátszása.

- *Open Macro* (


Configure (konfigurálás) menü:

- *Select Device...*: fejlesztéshez használt eszköz kiválasztása.
- *Configuration Bits...*: a konfigurációs bitek beállítása. Számos bitnek a szimulációnál is van hatása, a többinek csak a programozáskor.
- *External Memory...*: esetlegesen használt külső memória beállításai.
- *ID Memory...*: ID (egyedi azonosító) memória tartalmának beállítása.
- *Settings...*: az MPLAB IDE beállításai.

Window (ablak) menü:

- *Close All*: az összes megnyitott ablak bezárása.
- *Cascade*: az ablakok elrendezése úgy, hogy mindegyiknek látsszon a fejléce.
- *Tile Horizontally*: az ablakok között a munkafelület felosztása úgy, hogy az ablakok a munkafelület teljes szélességét kitöltik.
- *Tile Vertically*: az ablakok között a munkafelület felosztása úgy, hogy az ablakok a munkafelület teljes magasságát kitöltik.
- *Arrange Icons*: ikon állapotban lévő ablakok elrendezése.
- *Window Sets*: ablak-elrendezések listája. (Az ablak-elrendezések alkalmazásával bevált, személyre szabott felületen dolgozhatunk.)
- *Create Window Set...*: ablak-elrendezés elmentése.
- *Destroy Window Set...*: ablak-elrendezés törlése.
- A menüben megjelennek a megnyitott ablakok.

Help (súgó) menü:

Különböző témákban megjelenő súgók, a témák listája a  ikonra kattintva is megjelenik:

- *System:* rendszerelemek
 - *MPLAB IDE:* tudnivalók az MPLAB kezelői felület.
 - *MPLAB Editor:* súgó az MPLAB szövegszerkesztőjéről.
- *Language Tools:* fordítók.
 - *MPASM Assembler:* információk az MPASM assemblerről.
 - *MPLINK Object Linker:* súgó az MPLINK linkerről.
 - *PIC18 Config Settings:* PIC18 család konfigurációs lehetőségeinek kódban elhelyezhető direktívákkal való beállítása.
 - *COFF File Format:* információk a COFF formátumú hibakeresési információkat tartalmazó fájlformátumról.
- *Debuggers:* hibakeresők.
 - *MPLAB ICD 2:* információk az ICD2-ről.
 - *MPLAB SIM:* információk a szimulátorról.

A lista bővíülhet a telepítéstől függő egyéb témákkal.

2. Mintafeladatok megoldásának ismertetése

A következőkben **egyszerű feladatok** megoldásán keresztül bemutatjuk az MPLAB IDE kezelését, és a rendelkezésre álló szimulációs lehetőségek egy részét. A szimulációs lehetőségekről továbbiakat a mérés második részében tanulhat.

A két mérési segédletben található példákon keresztül a programfejlesztésre és hibakeresésre rendelkezésre álló nagyszámú eszköz és lehetőség teljes körű bemutatására időbeli (és területi) korlátok miatt nem kerülhet sor. Ebben az útmutatóban **csak a legfontosabb, illetve leghasznosabb eszközöket és lehetőségeket mutatjuk be**. Az ismeretek önálló munkával, részben a laboratóriumi gyakorlatokon, részben azon kívüli tevékenységgel, a Microchip oldaláról és az Internet más részeiből letölthető mintapéldák és a sugó tanulmányozásával **bővíthetők**.

2.1. példa

Írjunk programot, amely az előjel nélküli számokat tartalmazó A_1 és A_2 regiszter értékét összeadja, majd az eredményt hozzáadja a 16 bites előjel nélküli számot tartalmazó B_H:B_L regiszter-párhoz! Ezek után a folyamat előlről kezdődik. A feladat megoldásához kövessük az **alábbi lépéseket!**

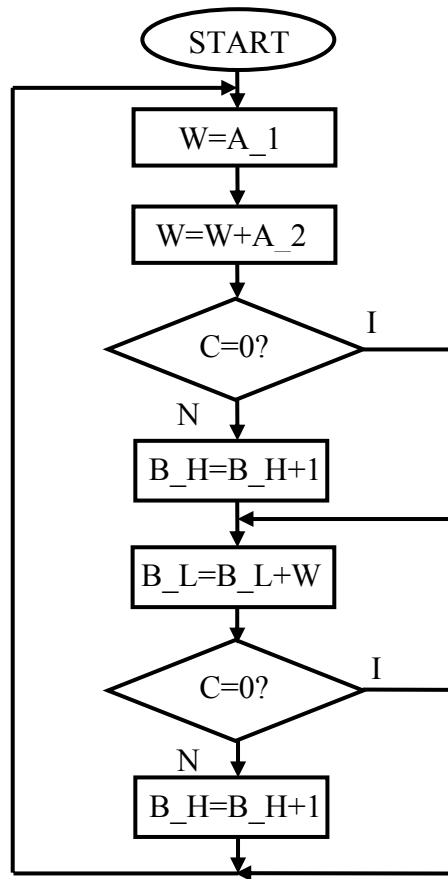
- a. Tervezzük meg a programot folyamatábra segítségével!
- b. Hozzunk létre projektet és a kódot, majd végezzük el a fordítást!
- c. Ellenőrizzük a program működőképességét szimulációval!

1.a. A program megtervezése

A program felépítésére **többféle lehetőség** van. Az alábbi példában törekedtünk a legrövidebb kód kialakítására és a processzor utasításkészlete lehetőségeinek kihasználására.

Első lépés az A_1 és A_2 regiszterek összeadása. Itt két eset lehetséges: az eredmény vagy elfér 8 biten, vagy nem. Amennyiben keletkezik átvitel, akkor azt a B_H regiszterhez hozzá kell adni. Következő lépés az összegnek a B_H:B_L regiszter-párhoz való hozzáadása. A

keletkezett összeget B_L-hez kell adni. Ha keletkezik átvitel, azt B_H-hoz kell adni. (A fenti algoritmus nem dolgozza fel a B_H:B_L regiszter-pár túlcsondulását!) Az előbbieket szerint kialakított program folyamatábráját az alábbi ábrán láthatjuk.

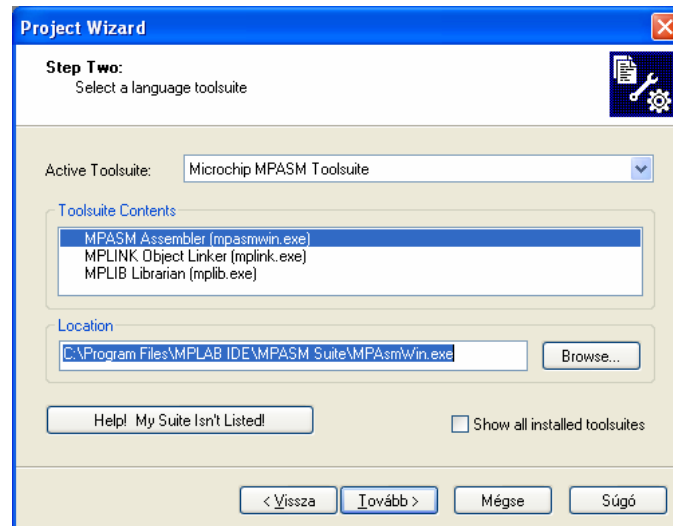


1.b. A projekt és a kód létrehozása, fordítás

A következőkben nézzük meg, hogyan lehet **projektet létrehozni és beállítani** az MPLAB-ban!

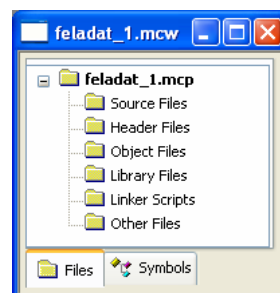
Indítsuk el az MPLAB-ot! Amennyiben egy előzőleg használt projektet betöltve indul el, zárjuk be azt a *Projekt→Close* paranccsal!

Indítsuk el a projekt varázslót (*Project→Project Wizard...*)! A felugró ablakban kattintsunk a *Tovább* gombra! A következő ablakban válasszuk ki a **processzor típusát** (18F452), majd kattintsunk a *Tovább* gombra! A megjelenő ablakban a **fordítót** választhatjuk ki, amely esetünkben a Microchip MPASM Toolsuite.



A *Toolsuite Contents* ablakban megtekinthetjük, hogy a fordító készlet milyen programokat tartalmaz. Jelen esetben az MPASM assembler, az MPLINK linker és az MPLIB könyvtárkezelő jelenik meg a listában. Amennyiben a fordító elérési útja még nincs beállítva (a *Location* mező üres), akkor a *Browse* paranccsal beállíthatjuk azt. (A laboratóriumban lévő gépeken ilyen probléma nem léphet fel.) Lépünk tovább!

Itt adjuk meg a projekt nevét, és azt a könyvtárat, ahová menteni szeretnénk! A *Tovább* gombra kattintva egy olyan ablakhoz jutunk, amelyben a projekthez adhatunk már létező fájlokat. Ezt a lépést a *Tovább* gomb újbóli megnyomásával ugorjuk át, mivel még nincs forrásunk, amit a projekthez adhatnánk. A *Befejezés* gombra kattintva **létrejön a projektünk**, amelynek felépítését az alábbi ábrán bemutatott projekt ablakban láthatjuk. (Amennyiben ez az ablak nem látszik, akkor a *View*→*Project* paranccsal láthatóvá tehetjük.)



A projekt **további beállításait** is végezzük el! A konfigurációs bitek közül azokat, amelyek hatással vannak a szimulációra, a *Configure*→*Configuration Bits* menüben állíthatjuk be. Mostani projektünknel csak a Watchdog Timer-t kell letiltani (*Disabled-Controlled by*

SWDTEN bit), hogy hosszú szimuláció esetén ne okozzon problémát. Az ablak bezárásával érvényesíthetjük a beállításokat.

A programfejlesztés során **hibakeresőként a szimulátort** szeretnénk használni, ezt a *Debugger→Select Tool→MPLAB SIM* parancs kiadásával tehetjük meg. A szimulátor beállításait ezek után a *Debugger→Settings...* ablakban végezhetjük el. A feladatokhoz fontos beállítások: *Osc/Trace* fülön: órajel (*Processor Frequency*): 4MHz, *Animation/Realtime Updates* fülön: animálási időköz (*Animate step time*): 100ms.

Hozzuk létre a kódot, vagy abban az esetben, ha elérhető a gépen, másoljuk be a projekt könyvtárába! Ha a kódot létre kell hozni, akkor nyissunk egy új ablakot (*File→New*). Ebbe az ablakba gépeljük be a kódot, majd mentjük el a projekt könyvtárába, .asm kiterjesztéssel! Ha a forrás elérhető a gépen, akkor fájlkezelővel másoljuk a projektkönyvtárba! Ezek után adjuk hozzá a projekthez a forrást! A projekt ablakban a forrásfájlok (*Source Files*) felett jobb gombbal kattintva előjön egy menü, amelyből válasszuk a fájl hozzáadása (*Add Files...*) parancsot, és adjuk meg a forrásfájl nevét és elérhetőségét! A *Megnyitás* gombra kattintva a forrásfájl hozzáadódik a projekthez. Közvetlenül nem használjuk, de a könnyebb tájékozódás kedvéért adjuk hozzá a „*P18F452.INC*” fájlt a projekt *Header Files* csoportjához (a fájl az MPASM fordító könyvtárában van). Az alábbiakban közöljük a forrás egy lehetséges változatát.

```

*****
;
; * feladat_1.asm
;
*****
; A program két regiszter (A_1 és A_2) értékét adja össze, majd az
; eredményt a B regiszterpár értékéhez (B_H és B_L) adja hozzá
;
*****

list p=18f452

include "p18f452.inc" ; A processzorfüggő deklarációkat tartalmazó include fájl

A_1      equ 0x60      ; Változók elhelyezése az általános felhasználású
A_2      equ 0x61      ; területre
B_H      equ 0x70
B_L      equ 0x71

Rst_vect      org 0x0000 ; Reset vektor, tápbekapcsoláskor, resetkor innen indul
; a program végrehajtása

goto Start

Start ; A program kezdete

```

```

org    0x0040

movf   A_1, 0      ; A_1 betöltése a munkaregiszterbe (W)
addwf  A_2, 0      ; A_2 W-hez adása, eredmény W-be
btfsc  STATUS, C   ; Carry flag (átvitel) vizsgálata,
incf   B_H, 1      ; ha volt Carry, B_H növelése
addwf  B_L, 1      ; Az eredményt B_L-hez adjuk
btfsc  STATUS, C   ; Carry flag (átvitel) vizsgálata,
incf   B_H, 1      ; ha volt Carry, B_H növelése
goto   Start       ; Újrakezdés

end

```

Néhány **szintaktikai** szempontból fontos **megjegyzés**:

- Megjegyzéseket pontosvessző után helyezhetünk el a forrásban.
- A direktívákat és az utasításokat tartalmazó sorokat egy tabulátorral beljebb kell kezdeni. Bár hibát nem okoz, de a fordító figyelmeztetést (Warning) ad.
- A mnemonikokat kis és nagybetűvel is írhatjuk.
- A szimbólumok (SFR-ek, bitazonosítók), változók és konstansok nevei érzékenyek a kis/nagybetűkre.

Egyéb megjegyzések:

- A projekthez beállított, és a programban definiált processzor típusnak egyeznie kell.
- Az „end” direktíva utáni sorokat a fordító nem dolgozza fel.

A projekt ablakban lévő fájlok kettős kattintással nyithatóak meg. Nyissuk meg a forrást, és a *P18F452.INC* fájlt, amely a processzorral kapcsolatos szimbólumok definícióit tartalmazza!

Vizsgálja meg, és értelmezze ez utóbbi fájl tartalmát!

A továbbiakban akkor szükséges e fájl megnyitása, ha a szimbólumok használata során valamilyen probléma fordul elő (pl. ismeretlen szimbólum, esetleges hibás definíció).

A forrás **fordítását** a *Make* vagy a *Build All* parancs kiadásával végezhetjük el. A fordítás után részben egy felugró ablakban, részben pedig az *Output* ablakban kaphatunk információt. Sikeres fordítás esetén a felugró ablakban zöld jelzést, az *Output* ablakban pedig a „*BUILD SUCCEEDED*” üzenetet kapjuk. Hiba esetén a felugró ablakban piros jelzést, az *Output* ablakban pedig a részletes hibajelentést, valamint a „*BUILD FAILED*” üzenetet kapjuk. A hiba okát jelző sorban szögletes zárójelben a hiba jellegének azonosítója, a hiba helye (fájl

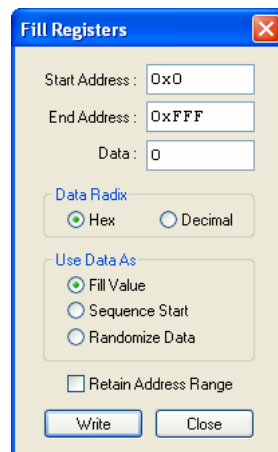
neve, elérési útja, és a hiba sora), a sor végén pedig a hiba szöveges leírása található. Érdeemes a hibák (*Error*) mellett a figyelmeztetések (*Warning*) üzeneteit is figyelmesen elolvasni!

1.c. Ellenőrizzük a program működőképességét!

A program **működőképességének ellenőrzésére** többféle megoldás lehetséges.

Egyik megoldásként szóba jöhet a bemeneti értékek kézi beállítása, ez gyors vizsgálathoz megfelelő. Ehhez nyissuk meg az általános felhasználású regiszterek ablakát (*View→File Registers*)! Itt látható az általános felhasználású regiszterek tartalma. Ha egy regiszter tartalma változik, kiemelten (alapértelmezésben piros színnel) jelenik meg. Ha egy regiszter tartalmára kettőt kattintunk, a tartalom átírható.

A regiszterablak felett jobb gombbal kattintva több lehetőség érhető még el a tartalmak megváltoztatására. Például a *Fill Registers...* parancs kiadása után a regiszterek megadható tartományát tudjuk feltölteni állandó, szekvenciális vagy véletlenszerű értékekkel. A *Fill Registers...* ablak az alábbi ábrán látható, az általános felhasználású regiszterek teljes tartományának csupa 0-val való feltöltéséhez előkészítve.



Változóink – ahogyan azt a forrásból láthatjuk – a 60h-61h (A_1, A_2) és a 70h-71h (B_H, B_L) címen vannak. **Vizsgáljuk meg** a program **működését** úgy, hogy mielőtt a tevékenységek elkezdődnének (pl. amikor a PC a Start címkére mutat, azaz a programlista melletti zöld nyíl ide mutat), töltsük fel a regisztereket a vizsgálandó értékekkel. Ügyeljünk, hogy a program minden lehetséges végrehajtási szálát teszteljük!

Egy lehetséges tesztadat-sorozat az alábbi táblázat mutat.

Tesztadat sorszáma	Bemenet		Kimenet előző értéke		Kimenet új értéke	
	A_1	A_2	B_H	B_L	B_H	B_L
1.	06h	10h	00h	00h	00h	16h
2.	06h	FEh	00h	16h	01h	1Ah
3.	00h	F0h	01h	1Ah	02h	0Ah

Az 1. sor szerint mindkét elágazás I ágát teszteljük, mivel az eredmény mindkét összeadásnál elfér 8 biten. A 2. sor tesztadatai az első elágazásnál a N, a második elágazásnál az I ágat teszteli. A 3. sor adataival az első elágazás I, a második elágazás N ágát vizsgálhatjuk. A táblázatban az eredményeket is közöljük.

A **másik lehetőség** a fájlból történő **automatizált bemeneti értékadás**. Hozzuk létre szövegszerkesztővel a gerjesztéshez szükséges fájlokat (a_1.txt és a_2.txt)! A fájlok tartalma az előző táblázat A_1 és A_2 oszlopa szerint lett kialakítva, tartalmukat a következőkben közöljük. Az egyes sorok hexadecimális formában a gerjesztést adják meg, minden sor vége egy-egy ENTER-rel van lezárva.

„a_1.txt” tartalma:

06

06

00

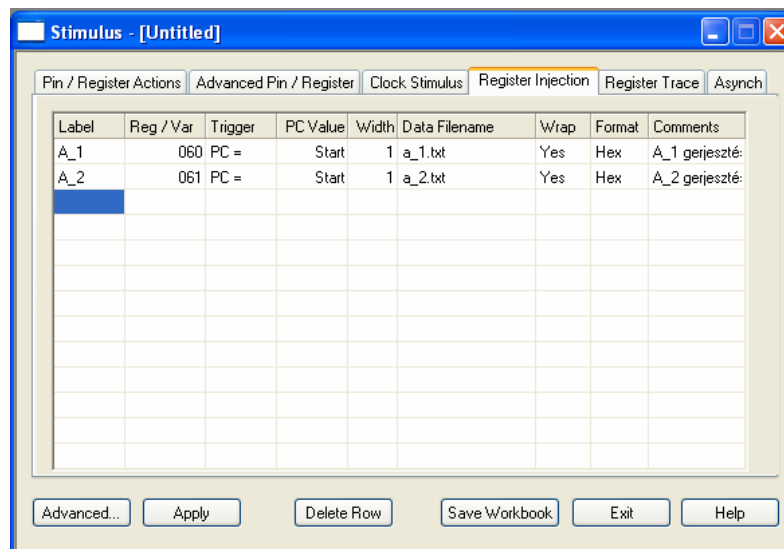
„a_2.txt” tartalma:

10

FE

F0

A létrehozott fájlok tartalmának betöltése a szimulátor **gerjesztés-generátorán** (*Debugger*→*Stimulus*→*New Workbook*) keresztül történhet. A regiszter töltés (*Register Injection*) fülre kattintva, töltsük ki az alábbiak szerint a mezőket, majd kattintsunk az *Apply* (alkalmaz) gombra!



A program **futtatása előtt biztosítsa a 70h-71h regiszterek 0 értékét** (lásd korábban)!

Helyezzünk el töréspontot az utolsó sorra (*goto Start*)! Ehhez a lista melletti szürke területen az adott sor mellett kettőt kell kattintani. A töréspont helyét egy **B** szimbólum jelzi. Ezek után minden futtatásnál a töréspontnál megáll a program, és a bemeneti értékek, valamint az eredmény az általános felhasználású regiszterek ablakában látható. A negyedik futtatástól kezdve a gerjesztések ciklikusan ismétlődnek, mivel a gerjesztés *Wrap* (újrakezdés) oszlopában *Yes* (igen) van beállítva.

2.2. példa

Írjunk programot, amely a PORTB-n a 0. bittől a 7. bitig, majd a 7. bittől a 0. bitig oda-vissza futtat egy 1-est, de csak akkor, ha a PORTA 0. bitje magas szintű. Ha a PORTA 0. bitje alacsony szintű, akkor a futtatás álljon meg. Ha újra magas szintű, a futtatás az előző pozíciótól induljon tovább! A feladat megoldásához kövessük az **alábbi lépéseket**!

- Tervezzük meg a programot folyamatábra segítségével!
- Hozzunk létre projektet és a kódot, majd végezzük el a fordítást!
- Ellenőrizzük a program működőképességét szimulációval!

2.a. Tervezze meg a programot folyamatábra, és a mikrovezérlő adatlapja segítségével!

A tervezésnek a portok beállítására vonatkozó részét a következőkben ismertetjük.

A PORTA és a PORTB használatához azokat **inicializálni kell**. **Le kell tiltani** a használt portlábak **esetleges multiplexelt funkcióit**, és **be kell állítani az irányukat**.

A **katalógus** szerint a PORTA 0. bitjének másodlagos funkciója az AN0 analóg bemenet. Ez az alapértelmezett funkció (lásd katalógus 9.1 pont, kiszürkített rész). A portlábnak digitális lábbá konfigurálásához a PCFG3:0 bitek segítségével (lásd katalógus 17.0 pont, ADCON1 regiszter). Látható, hogy a PORTA 0. bitje (RA0) egyetlen esetben van digitális lábbá konfigurálva, akkor, ha a PCFG3:0 bitek 011x értéket vesznek fel (az x azt jelenti, hogy PCFG0 bit értéke irreleváns).

A PORTB-vel könnyebb dolgunk van, mert a multiplexelt funkciók alapértelmezésben le vannak tiltva. A portlábak iránybeállításához tudni kell, hogy azok bekapcsoláskor és alaphelyzetbe állítás (reset) után bemenetek. Ha kimenetté szeretnénk egy portlábát konfigurálni, akkor a porthoz tartozó TRIS regiszter megfelelő bitjére 0-t kell írni. A kimeneti lábak inicializálásánál a hozzájuk tartozó LAT regiszter (vagy alternatív lehetőségként a PORT regiszter) megfelelő bitjén be kell állítani a kezdeti állapotot (kezdeti magas vagy alacsony kimenet).

Ezek szerint a következő beállításokat kell elvégezni:

PCFG3 = 0

PCFG2 = 1

PCFG1 = 1

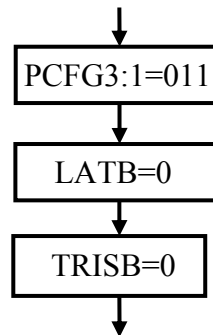
LATB = 00h (minden bit kimenetként 0 értékű lesz)

TRISB = 00h (minden bit kimenet)

PCFG0 értékét nem kell módosítani, nem befolyásolja RA0 digitális lábbá konfigurálását

TRISA értékét nem kell módosítani, alaphelyzetben minden láb bemenet (RA0 is).

A folyamatábrának a port-inicializálásra vonatkozó része a következők szerint alakul:



2.b. Hozza létre a projektet az 1. feladatban megismert módon! Írja meg a programot!

A program egy lehetséges változatát segítségképpen a következőkben közöljük.

```

*****
;
;* feladat_2.asm
;*****
; A program RA0 magas szintje esetén végigfuttat PORTB0-tól
; PORTB7-ig egy egyest, utána PORTB7-től PORTB0-ig, majd
; kezdődik előlről a folyamat. Amennyiben az újratekérésnél
; vagy az indulásnál RA0 alacsony szintű, a folyamat szünetel.
;* Knight Rider :)
;*****
;

    list p=18f452

    include "p18f452.inc"          ; A processzorfüggő deklarációkat tartalmazó include fájl

Rst_vect          ; Reset vektor, tápbekapcsoláskor, resetkor innen indul
org 0x0000        ; a program végrehajtása

    goto Start

Start              ; A program kezdete
org 0x0040

    bcf  ADCON1, PCFG3            ; RA0 digitális lábbá való konfigurálása
    bsf  ADCON1, PCFG2
    bsf  ADCON1, PCFG1
    clrf LATB                    ; LATB minden bitje indulás után 0 lesz
    movlw 0x00                   ; PORTB minden bitje kimenet
    movwf TRISB
    bsf  STATUS, C                ; Beállítjuk 1-be a C flaget, hogy az első léptetésnél
                                ; ez az egyes kerüljön PORTB0-ra

Check_RA0
    btfss PORTA, 0                ; RA0 vizsgálata
    goto Check_RA0               ; Ha 0, várunk
Rotate_L           ; Ha 1, indul a folyamat
    rlc  PORTB, 1                 ; Forgatás balra C-n keresztül
    btfss PORTB, 7                ; Addig, amíg a legfelső helyértékre nem érkezik az 1-es
    goto Rotate_L
Rotate_R
    rrc  PORTB, 1                 ; Forgatás jobbra C kihagyásával

```


btfss	PORTB, 0	; Addig, amíg a legalsó helyértékre nem érkeznek az 1-es
bra	Rotate_R	; Másik lehetőség (goto helyett) feltétel nélküli ugrásra
goto	Check_RA0	; A folyamat újraindul
end		

Fordítsa le a programot az előző feladatnál leírtak szerint, **javítsa az esetleges szintaktikai hibákat!**

2.c. Ellenőrizzük a program működőképességét szimulációval!

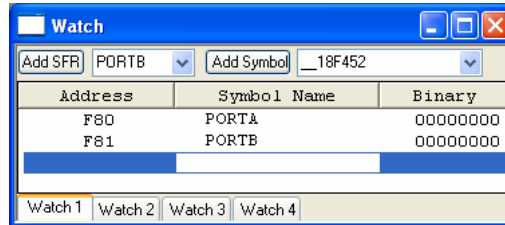
A szimuláció során gerjeszteni kell RA0-t, és vizsgálni kell a PORTB-n megjelenő értékek sorrendjét. A bemenetet **aszinkron gerjesztéssel** fogjuk működtetni, miközben a portok állapotát figyeljük.

Az aszinkron gerjesztés **beállításához** nyissuk meg a gerjesztés-generátort (*Debugger→Stimulus→New Workbook*)! Válasszuk ki az aszinkron (*Asynch*) fület! Az első sor *Pin/SFR* mezőjébe kattintva állítsuk be a gerjeszteni kívánt lábat (RA0)! Az *Action* mezőben beállíthatjuk a gerjesztés aktiválásakor elvégzendő beavatkozást, itt válasszuk a *Toggle* (átkapcsolás, „negálás”) lehetőséget, majd alkalmazzuk (*Apply*) a beállításokat! Ezek után a *Fire* oszlopban lévő „>” jelre kattintva adhatjuk rá a kívánt lábra vagy regiszterre a gerjesztést. A **gerjesztés hatása** a program következő sorának **végrehajtásakor válik láthatóvá**, mivel ekkor történik a regiszterek frissítése. A gerjesztés-generátor ablakát **ne zárjuk be!**

A két port állapotának láthatóvá tételéhez vagy nyissuk meg a speciális funkciójú regiszterek (SFR) ablakát (*View→Special Function Registers*), vagy pedig nyissunk egy megfigyelő (*Watch*) ablakot (*View→Watch*). Ez utóbbit választva a Watch ablakban kiemelhetjük a számunkra fontos regisztereket (jelen esetben PORTA-t és PORTB-t), és nem kell ezeket egy hosszú listában nyomon követni.

A két regiszternek a Watch ablakba való felvételéhez kattintsunk az *Add SFR* gomb melletti mezőre, válasszuk ki PORTA-t, majd adjuk ki az *Add SFR* (SFR hozzáadása) parancsot! Ekkor a PORTA megjelenik a lejjebb látható listában. Ugyanezt ismételjük el PORTB-vel is! A portok állapotát jelen esetben célszerűen **bináris megjelenítésben** kövessük nyomon (a

végigfutó „1” így látványos)! Ha a regiszterek értéke nem látszik bináris formában, akkor a lista feletti sávon kattintsunk jobb gombbal, majd kattintsunk a bináris (Binary) megjelenítésre. Az alábbi ábrán a fentiek szerint létrehozott Watch ablak látható.

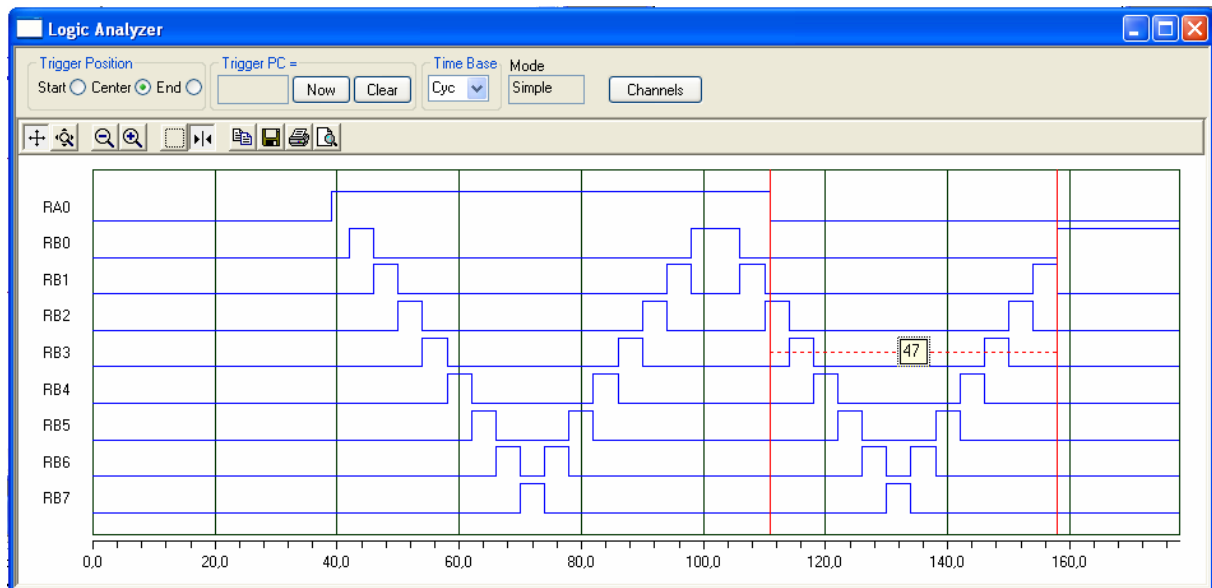



Futtassa a programot animált módban (*Debugger→Animate*)! **Adja rá** az RA0 lábra **a gerjesztést** (kattintás a definiált gerjesztés előtti „>” gombon)! **Vizsgálja meg**, hogy RA0 állapotot vált-e, illetve PORTB-n RA0 állapotának megfelelően történnek-e a változások!

A szimulátor rendelkezik **logikai analízátor funkcióval** is. A logikai analízátor a *View→Simulator Logic Analyzer* paranccsal nyitható meg.

Vizsgáljuk meg PORTB változását az idő függvényében a logikai analízátor segítségével! A *Channels* gombra kattintva válasszuk ki a listából RA0 és az RB0...RB7 biteket, majd kattintsunk az *Add* (hozzáadás) gombra! (Több jel kiválasztása a Shift és a Ctrl billentyűk használatával lehetséges, a Windowsban megszokott módon.) OK-val zárjuk be az ablakot!

Futtassuk újra a programot! Futtatás után helyes programműködés esetén az alábbihoz hasonló ábrát kapunk.



Az ábrában látható két függőleges piros vonal **kurzor**, amelyet a  gombbal lehet bekapcsolni. A két vonal közötti mezőben megjelenő szám a két kurzor távolságát mutatja ciklusszámban (utasításciklus). A grafikon felett található ikonokkal testre szabhatjuk az időfüggvény megjelenését.

Nyissa meg a szimulátor **nyomkövetőjét** (*View*→*Simulator Trace*)! Kattintson az ablak felső részében egy nem NOP-ot tartalmazó sorra! **Értelmezze** az ablak alsó és felső felében található információkat! Mi az oka a megjelenő NOP-oknak?

3. Mérési feladatok

1. Végezze el a házi feladat 2. pontja szerinti program szimulációját, és az esetleges hibakeresését és javítását!
2. Végezze el a házi feladat 3. pontja szerinti program szimulációját, és az esetleges hibakeresését és javítását!
3. Készítsen programot, amely a PORTC alsó 4 bitjének értékét fordított sorrendben átmásolja a felső 4 bitre! Szimulációval ellenőrizze a program működését, végezze el az esetleges hibakeresést és javítást!
4. Oldja meg a mérésvezető által kiadott feladatot!

4. Házi feladat

1. Korábbi tanulmányai, a mérést előkészítő előadásokon elhangzottak, és a katalógus alapján készüljön fel a következő témákból:
 - Általános aritmetikai és logikai műveletek
 - PIC18F452
 - blokkvázlata
 - Adatmemória felépítése
 - Speciális funkciójú regiszterek (különös tekintettel a processzormaghoz kötődő regiszterek, pl. STATUS, és a portokhoz kapcsolódó regiszterek)
 - Digitális portok felépítése, konfigurálása, kezelése
 - Utasításkészlet
 - MPLAB szimulátor
 - Általános kezelési ismeretek
 - Gerjesztések (aszinkron, regiszter töltő) és regiszter naplózás
 - Logikai analízátor, nyomkövető
2. Tervezzen és írjon programot PIC18F452-re, amely két regiszter (A_1 és A_2) 8 bites, előjel nélküli tartalmának relációját megállapítja: ha egyenlők, B-be 0-t tölt, ha A_1 nagyobb, akkor 1-et, ha A_2, akkor pedig 2-t! Tervezze meg a program működőképességének vizsgálatát!
3. Tervezzen és írjon programot, amely RB0 0→1 átmenetére inkrementálja a SZAMOL regiszter értékét! Tervezze meg a program működőképességének vizsgálatát!