

Debreceni Egyetem  
Műszaki Kar  
Villamosmérnöki és Mechatronikai Tanszék



## Hexapod robot tervezése és építése mikrokontrolleres vezérléssel

Készítette: **Szilágyi Péter** mechatronikai mérnöki hallgató

Belső konzulens: Vitéz Attila

Tanszékvezető: Husi Géza

Debrecen  
2013. október

# Tartalomjegyzék

|        |                                    |    |
|--------|------------------------------------|----|
| 1.     | Bevezetés és célkitűzések .....    | 3  |
| 2.     | A hexapodokról általában [5] ..... | 4  |
| 2.1.   | Konstrukció .....                  | 5  |
| 2.2.   | Mozgás típusok .....               | 6  |
| 2.3.   | Biológiai inspiráció .....         | 6  |
| 3.     | Tervezési folyamat.....            | 8  |
| 3.1.   | A mechanizmus .....                | 8  |
| 3.2.   | A vezérlő elektronika .....        | 14 |
| 3.2.1. | Szervomotorok vezérlése [7] .....  | 14 |
| 3.2.2. | A BTM-222 bluetooth modul [8]..... | 16 |
| 3.2.3. | A PIC16F877A mikrokontroller ..... | 17 |
| 3.2.4. | Kapcsolási rajzok.....             | 19 |
| 3.2.5. | Az elkészített áramkörök .....     | 21 |
| 3.3.   | A szoftver .....                   | 22 |
| 3.3.1. | A program forráskódja.....         | 24 |
| 3.3.2. | Felhasznált szoftverek.....        | 27 |
| 4.     | Eredmények .....                   | 28 |
| 5.     | Összefoglalás .....                | 30 |
| 6.     | Irodalomjegyzék .....              | 31 |
| 7.     | Ábrajegyzék .....                  | 32 |

# 1. Bevezetés és célkitűzések

Böngészve az internetet, manapság nagyon sok olyan mobil robottal találkozhatunk, melyek mozgási formájuk, de esetlegesen még a külalakjuk is a természetből inspirálódtak. Ezek a legkülönbözőbbek lehetnek, kezdve a hullóktól egészen a négylábú emlősökig. Egy nagyon gyakori robot-modell a természetből a rovarok csoportja (ezen belül is főleg a sáska és a csótány). Ez nem véletlen, hiszen a rovarok idegrendszere jóval egyszerűbb a többi állathoz képest, a viszonylag komplex viselkedésüket mindössze néhány idegsejt határozza meg, valamint az érzékszervek és a végtagok közti csatorna nagyon rövid, így az általuk végzett mozgásformák sokkal könnyebben rekonstruálhatóak mechatronikai eszközök segítségével.

A választásom azért esett erre a témára, mert egy ilyen felépítésű robot esetében nagyon összetett vezérlést lehet megvalósítani, viszonylag kis teljesítményű mikrokontrollerrel is, továbbá a robot megtervezése és megépítése után egy olyan robotplatform születne, amely az érzékelőket, beavatkozókat és nem utolsósorban a vezérlést végző programot tekintve a végtelenségig fejleszthető. Egy másik szempont volt a választásom során, hogy egy lábakon járó robot a felépítéséből adódóan olyan környezetben is képes mozogni, ahol kerekes, illetve lánc talpas testvérei csődöt mondanak, mindemellett nagyon látványos járási algoritmusok demonstrálására is alkalmas.

Célkitűzéseim között egy olyan hatlábú robot megtervezése és megépítése szerepelt, amely bluetooth segítségével kommunikál számítógéppel, így a processzor igényesebb feladatokat a PC végezhetné el, míg a mikrokontrollerre mindössze a lábak mozgatási algoritmus, a szenzorok jeleinek feldolgozása és a PC-vel való kommunikáció lekezelése hárulna. Ezen kívül további előnye a PC-s vezérlésnek, hogy az interfész, sokkal inkább emberközelí, naplózhatók az események, valamint Joystick-al és egyéb eszközökkel is vezérelhető a robot.

Külön köszönetet szeretnék mondani konzulensemnek, Vitéz Attilának, aki nélkül ez a dolgozat nem jöhetett volna létre, valamint Szabó Gábor hallgatótársamnak a mechanikai munkálatokban való közreműködéséért.

## 2. A hexapodokról általában [5]

A hexapod egy hat lábon járó jármű (1. ábra). Mivel egy robot három vagy több lábon állva már statikailag határozott, így egy hexapod robot nagyon rugalmas járási módokra képes. Ha egy láb mozgásképtelenné is válik, a robot továbbra is képes járni, mivel nincs szüksége az összes lábra ahhoz, hogy stabil maradjon; a többi láb szabadon képes helyet változtatni vagy tömegeket elmozdítani.

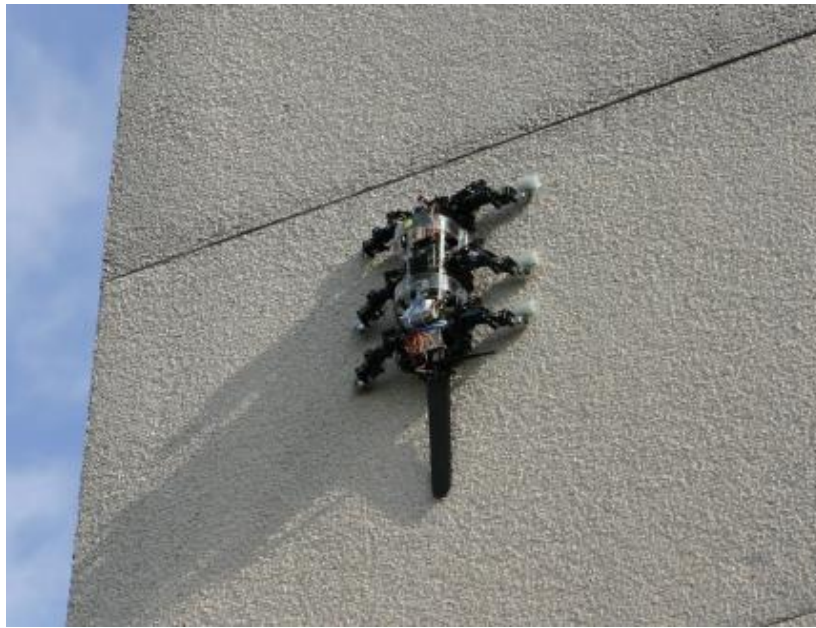
A hexapod típusú szerkezetek kiválóan alkalmasak biológiai elméletek tanulmányozására a rovarok mozgásairól, motorikus idegekről és a neurobiológiáról.



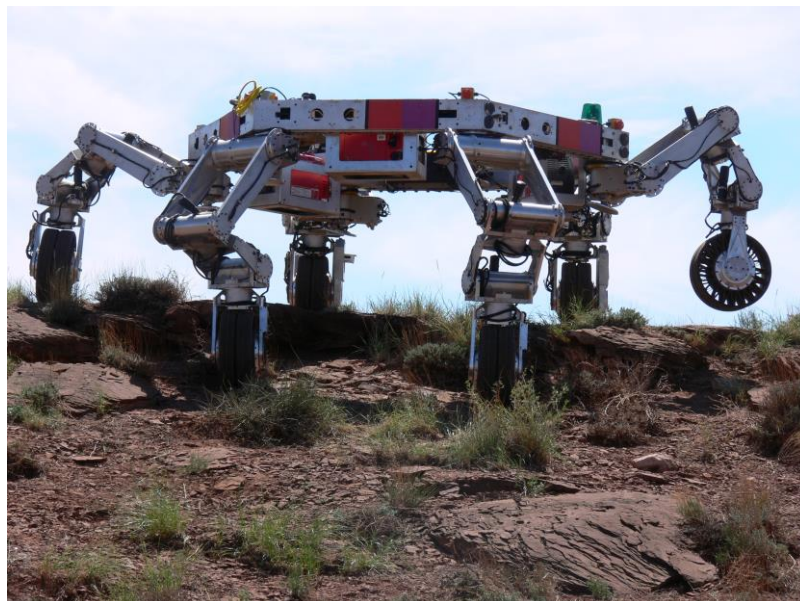
**1. ábra - Hexapod robot a valóságban**

## 2.1. Konstrukció

A hexapodok szerkezete általában a lábak elhelyezésében tér el. A rovarokról mintázott robotok általában tengelyesen szimmetrikusak a középvonalukra, mint például a Carnegie Mellon egyetem RiSE nevű robotja (2. ábra). Radiálisan szimmetrikus hexapod pedig a JPL (Jet Propulsion Laboratory) által kifejlesztett ATHLETE (All-Terrain Hex-Legged Extra-Terrestrial Explorer) (3. ábra).



**2. ábra - RiSE robot falmászás közben**



**3. ábra - ATHLETE robot hegymászás közben**

A lábak szabadságfoka általában kettőtől hatig terjed. A hexapodok lábai általában hegyesek, néha be vannak vonva valamilyen ragadós anyaggal, amely a falmászásban segíti a robotot, azonban a hexapodok lapos talajon tudnak a leggyorsabban mozogni.

## 2.2. Mozgás típusok

A hexapod robotok leggyakrabban betanítás alapján mozognak, ami lehetővé teszi, hogy a robot menjen előre, forduljon és oldalra lépkedjen. A leggyakrabban használt mozgás típusok a következők:

- Alternáló tripod: 3 láb van a földön azonos időben
- Quadruped (négylábás mód)
- Mászás: egyszerre csak egy láb mozog

A hexapodok mozgási módjai a legtöbb esetben stabilak még kissé sziklás, egyenlőtlen talajon is.

A mozgás nem feltétlenül kell betanított legyen, bizonyos esetekben a lábak mozgási sorrendje nincs előre leszögezve, hanem egy számítógép választja ki az érzékelt környezet alapján. Ez a tulajdonság főleg a nagyon sziklás talajon hasznos, bár a létező technikák a mozgás automatikus tervezésére nagyon erőforrás-igényesek.

## 2.3. Biológiai inspiráció

A biológiai modell-választás azért esett a rovarok csoportjára, mert az idegrendszerük sokkal egyszerűbb mint a többi állaté, ennek ellenére nagyon komplex viselkedést képesek mindössze néhány neuronnal produkálni. Az érzékszervek és lábak közti csatorna viszonylag rövid, így mozgási szokásaik nagyon jól felhasználhatóak a robotok fejlesztésében.

A biológiailag inspirált hexapodok tulajdonságai nagyban függnek a felhasznált rovar-modelltől. A csótány és a sáska a két leggyakrabban használt modell, mivel mindkét rovar neuropszichológiai szempontból alaposan ismert. Mivel azonban jelenleg még a teljes idegrendszerük nem ismert, a robotok tervezésénél több rovar tulajdonságát ötvözve használják fel.

A hexapodok esetében két féle mozgási mód terjedt el: centralizált és decentralizált vezérlő architektúra. A centralizált vezérlő közvetlenül vezérli az összes

láb mozgását, míg a decentralizált architektúra esetében a hat láb párhuzamos rendszerben működik; a lábak mozgatása a szomszédos lábak helyzetétől függ.

## 3. Tervezési folyamat

### 3.1. A mechanizmus

A mechanizmus megtervezésének legfontosabb szempontjai:

- legyen minél egyszerűbb
- legyen minél olcsóbb
- a funkcióját lássa el

Az internetes webshopokat böngészve rengeteg előre legyártott hexapod vázat találhatunk, azonban ezek vagy nem elégítik ki teljes mértékben az elvárásokat, vagy túl drágák, ezért végül a saját váz tervezését és legyártását választottam, mivel ez bizonyult a legmegfelelőbbnek mindenféle szempontból.

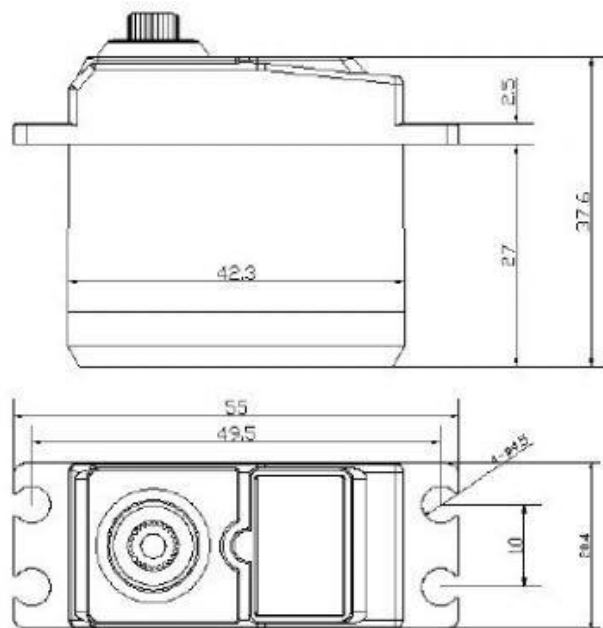
A váz megtervezésének első lépése a motorok típusának kiválasztása volt. Az analóg DC szervomotor bizonyult a legmegfelelőbbnek (4. ábra), mivel a  $180^\circ$ -os elfordulás bőven elengedő a lábak mozgatásához, valamint ennél a típusnál az áttét és a motorvezérlő elektronika is integrálva van a motor tokozásába, ami nagy számú motor esetében nagy előnyt jelent, mivel nem kell az összes motorhoz külön vezérlőt tervezni és készíteni, az áttét megtervezéséről nem is beszélve. További nagy előny, hogy szervomotor használata esetén nincs szükség szöghelyzet jeladóra, sem pedig végállás kapcsolókra, mivel ezen motorok vezérlőjének pontosan megadható, hogy milyen szögben kívánjuk elfordítani, így a motor tengelyének helyzete mindig ismert, kivéve azt az esetet, amikor a motor tengelye (a hexapod lába) valamilyen akadályba ütközik és megszorul.



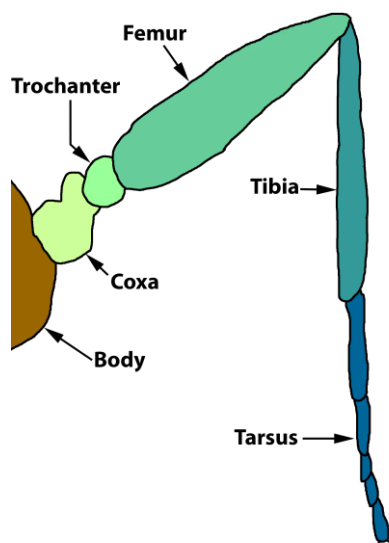
4. ábra - szervomotor belső felépítése

A különböző szervomotorok között kutatva találtam rá a TowerPro cég MG995-ös típusú szervomotorjára, amelynek nagy előnye, hogy az árához képest igen nagy nyomatékkal és sebességgel rendelkezik (6 voltos tápfeszültség mellett 13 kg/cm és 0,13 sec/60°), valamint fém fogaskerekes kialakításának, és duplacsapágyazásának köszönhetően igen időtálló is.

A következő lépés a lábak megtervezése volt, amely a TowerPro MG995-ös szervomotor méreteinek (5. ábra) valamint a rovarok lábainak (6. ábra) figyelembevételével történt az AutoCad nevű program segítségével.



5. ábra - szabványos szervómotor méretei

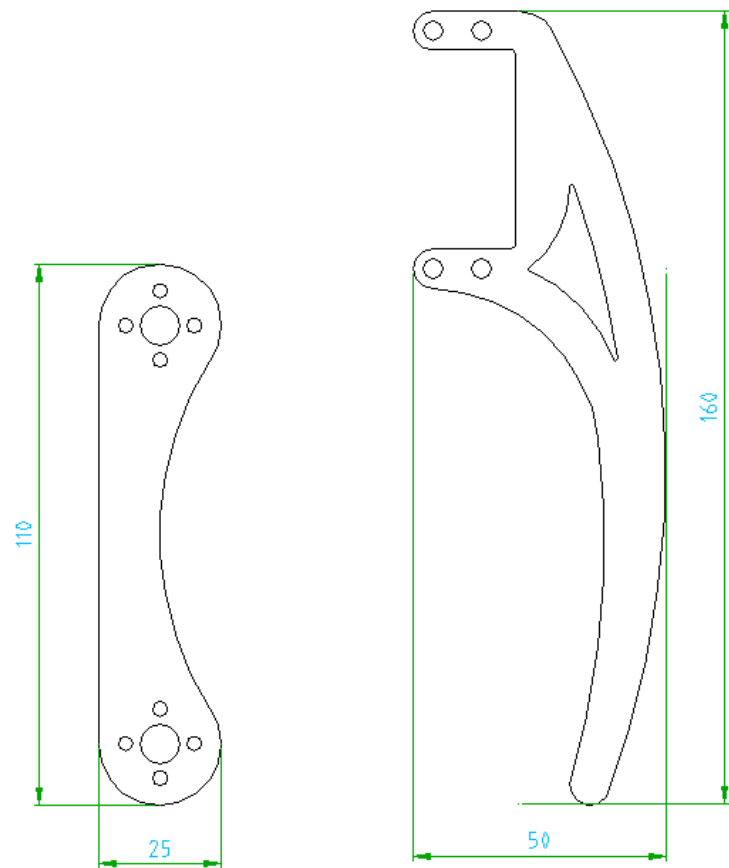


6. ábra - Lepidoptera (lepke) lábának felépítése

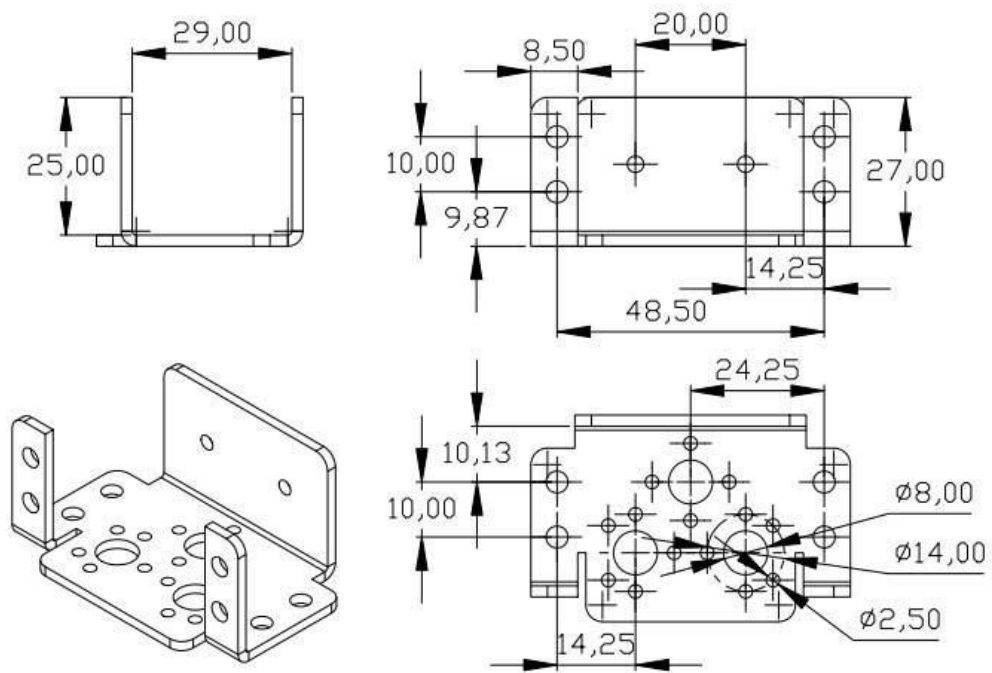
A legtöbb rovar lábához hasonlóan a hexapod lábát három szabadságfokúra terveztem, ami azt jelenti, hogy lábanként három ízület szükséges. A váz egyszerűségének kedvéért az ízületek maguk a szervomotorok, mivel a nagy nyomaték lehetővé teszi ezt. Összesen tehát 18 darab szervomotorra van szükség. A TowerPro MG995-ös típusú szervomotor súlya 55 g, ami azt jelenti, hogy a motorok összsúlya 990 g ( $\approx 1$  kg). Ehhez még hozzáadódik a későbbiekben betervezendő akkumulátor, ami 295 g ( $\approx 0,3$  kg), valamint az elektronika súlya, de ez többnyire elhanyagolható. A robot gyors mozgásához elengedhetetlenül szükséges, hogy összsúlya minél kisebb legyen. Az előbb említett alkatrészek tovább már nem könnyíthetők, így az egyetlen megoldás, hogy a vázszerkezetet a lehető legkönnyebbre tervezzük. Ennek érdekében arra a döntésre jutottam, hogy a váz vékony (1,5 mm-es) alumínium lemezből készüljön, mivel a váz bizonyos alkatrészeit hajtogatni kell, valamint lemezes alkatrészek esetén a váz legyártása történhet lézeres kivágással, amihez pedig mindössze az AutoCad-es rajz megszerkesztése szükséges. További előny, hogy így nem csak látványos, rádiuszos alkatrészek készíthetők, de a könnyítő sávok elhelyezése is sokkal egyszerűbb.

A láb köztes része (Femur) és hosszanti része (Tibia + Tarsus) a könnyítő sávokkal és befoglaló méreteikkel együtt a 7. ábrán láthatóak.

A törzs és a láb köztes része közti ízület (Coxa + Trochanter) már némileg bonyolultabb, mivel itt két forgáspont is található, az egyik a láb vízszintes tengelyen való elfordulásáért felel, míg a másik a láb emeléséért. Más hexapodok vázát tanulmányozva találtam rá egy olyan megoldásra, amely egy úgynevezett szervomotor konzolt (servo bracket) használ. Ez egy univerzálisan felhasználható keret (8. ábra) szabványos szervomotorokhoz, mely különböző szögekben kapcsolható a többi ugyanilyen konzolhoz, így szinte bármilyen mechanizmus létrehozható a segítségével. Főleg szervomotorokat használó hobbirobotok esetében elterjedt (robot kar, humanoid robot, hexapod, stb.). A konzol segítségével könnyen megoldható a szervomotorok egymáshoz képesti  $90^\circ$ -os helyzete. Természetesen kereskedelmi forgalomban ez is beszerezhető, azonban viszonylag drága, a hexapodhoz pedig 12 darabra lenne szükség, ami igen csak megnövelné a robot összköltségét. Az interneten azonban megtalálhatóak a konzol méretei (8. ábra), így némi tervezgetéssel és hajtogatással egyedileg legyártható ugyancsak alumínium lemezből, lézeres kivágással.

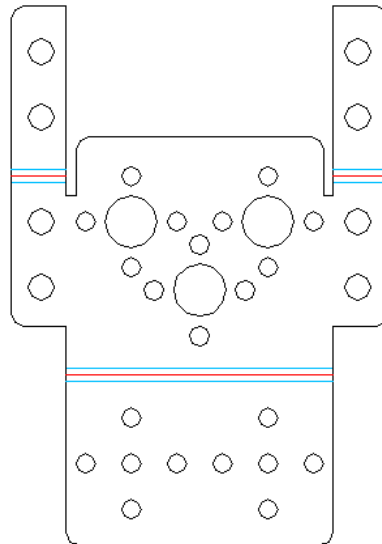


7. ábra - a láb köztes és hosszanti részének befoglaló méretei



8. ábra - univerzális szervo konzol méretei és 3D-s látképe

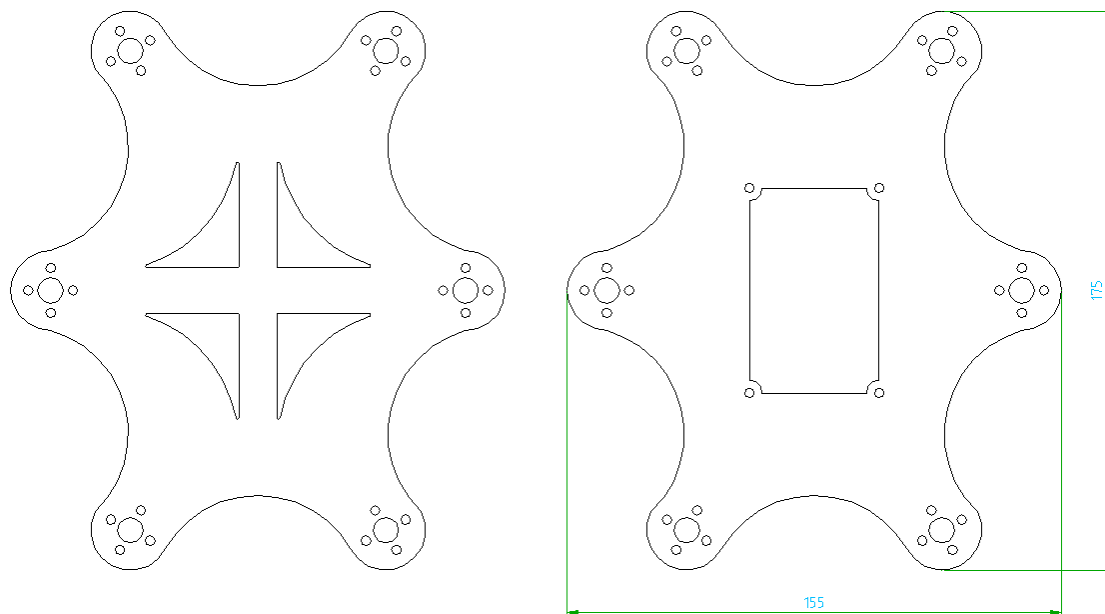
A konzol általam tervezett, kiterített AutoCad-es változata a 9. ábrán látható. A piros vonalak a hajlítás helyét, míg a kék vonalak a hajlítás utáni méretét jelzik az egyes részeknek, figyelembe véve az anyagvastagságot.



**9. ábra - szervo konzol kiterített látképe**

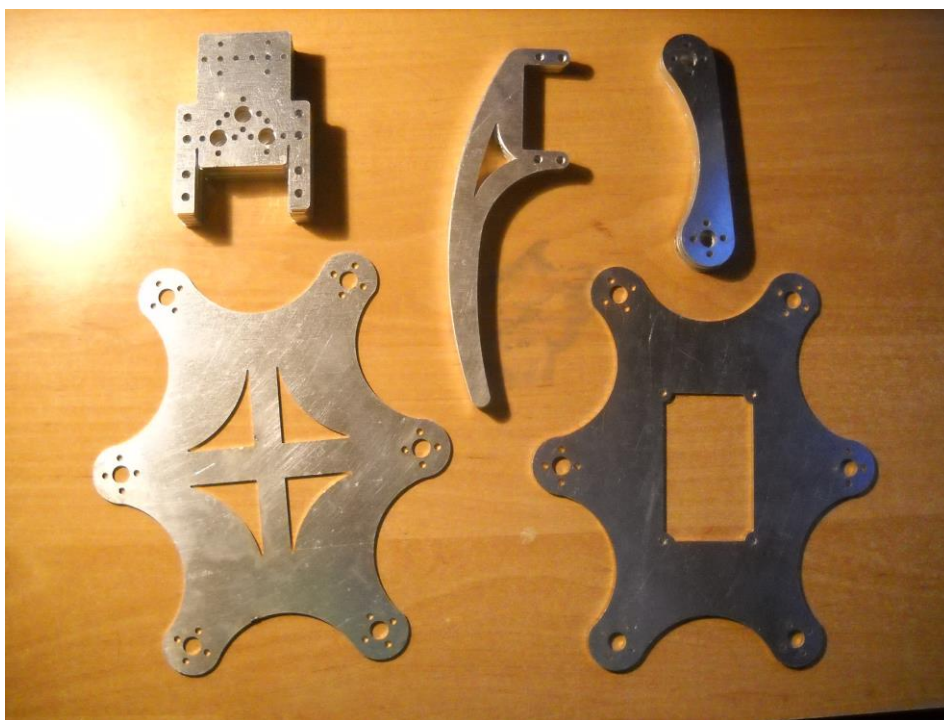
Ez után már csak a robot törzsének megtervezése maradt hátra. Ennek méreteit úgy alakítottam, hogy körül-belül arányos legyen a lábak méretével, valamint úgy, hogy a lábak mozgástere egyenként biztosított legyen. További szempont volt a tervezésnél, hogy a robot felépítése szimmetrikus legyen két tengely körül is. A törzset két lemezből formáltam meg, így biztosítva helyet mind az akkumulátornak, mind pedig a vezérlő elektronika részére. Az akkumulátor a két lemez között kapott helyet, így védve az esetleges mechanikai behatásoktól. A törzs súlycsökkentése érdekében terveztem könnyítő sávokat a lemezekbe, melyek közül az alsó lemez sávjaira egyszerűen felrögzíthető az akkumulátor, míg a felső könnyítő sávon átvezethetők a szervomotorok vezetékei. A rádiuszos kialakítás itt is nagy szerepet kapott, mivel segítségével a robot külleme látványosabb, rovarhoz jobban hasonlító kinézet nyerhető vele. A felső lemezen helyet kapott még négy rögzítő furat, mely a vezérlő elektronika lefogatásáért felel, négy darab M3-as fém távtartó és csavar segítségével, így az egyes szervomotorok vezetékei könnyedén kivezethetők a robot törzséből a vezérlő nyák tetejére. A felső és az alsó lemez szintén négy darab távtartó segítségével csatlakozik egymáshoz, közvetlenül a vezérlő elektronika nyomtatott áramköri lapjának rögzítő furatain keresztül, így biztosítva a robot törzsének merevségét, valamint a lemezek közti távolságot a belső szervomotorok számára. A törzs lemezei kellően nagyok ahhoz, hogy

a későbbi fejlesztések során felkerülő szenzorok és beavatkozók (pl. megfogó) felfogatásához szükséges további furatok felkerülhessenek. A robot törzsének alsó és felső lemezei, azok befoglaló méreteivel együtt a 10. ábrán láthatóak:



**10. ábra - az törzs alsó és felső lemeze befoglaló méreteivel**

A robot legyártott alkatrészei a 11. ábrán láthatóak:



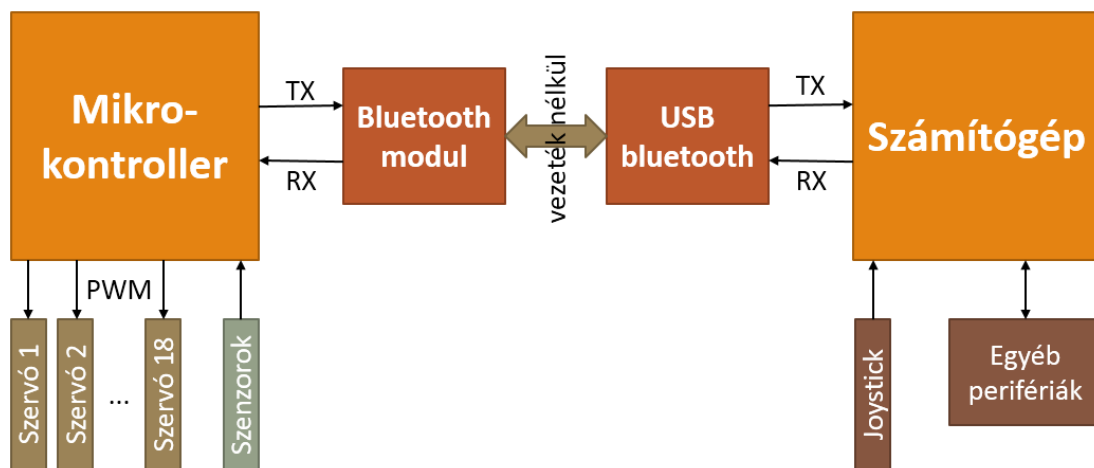
**11. ábra - a kész alkatrészek**

## 3.2. A vezérlő elektronika

A hexapod robot vezérlőjének a következő feladatokat kell ellátnia:

- vezérelnie kell párhuzamosan a 18 darab szervomotort
- kezelnie kell a bluetooth-os kommunikáción érkező parancsokat
- fogadnia kell az esetleges szenzorok jeleit, majd továbbítani bluetooth-on a PC felé
- vezérelnie kell az esetleges megfogókat

A vezérlés sematikus rajza a 12. ábrán látható:



12. ábra: A vezérlés sematikus ábrája

### 3.2.1. Szervomotorok vezérlése [7]

A szervomotoroknak három kivezetésük van. Ebből kettő a tápfeszültség kivezetése, ami 4,8 – 7,2 V között változhat. A nagyobb tápfeszültség nagyobb nyomatékot és nagyobb sebességet eredményez, azonban csökkenti a szervomotor élettartamát. A harmadik vezeték a vezérlésért felel (erre 5 V-os impulzusok kerülnek).

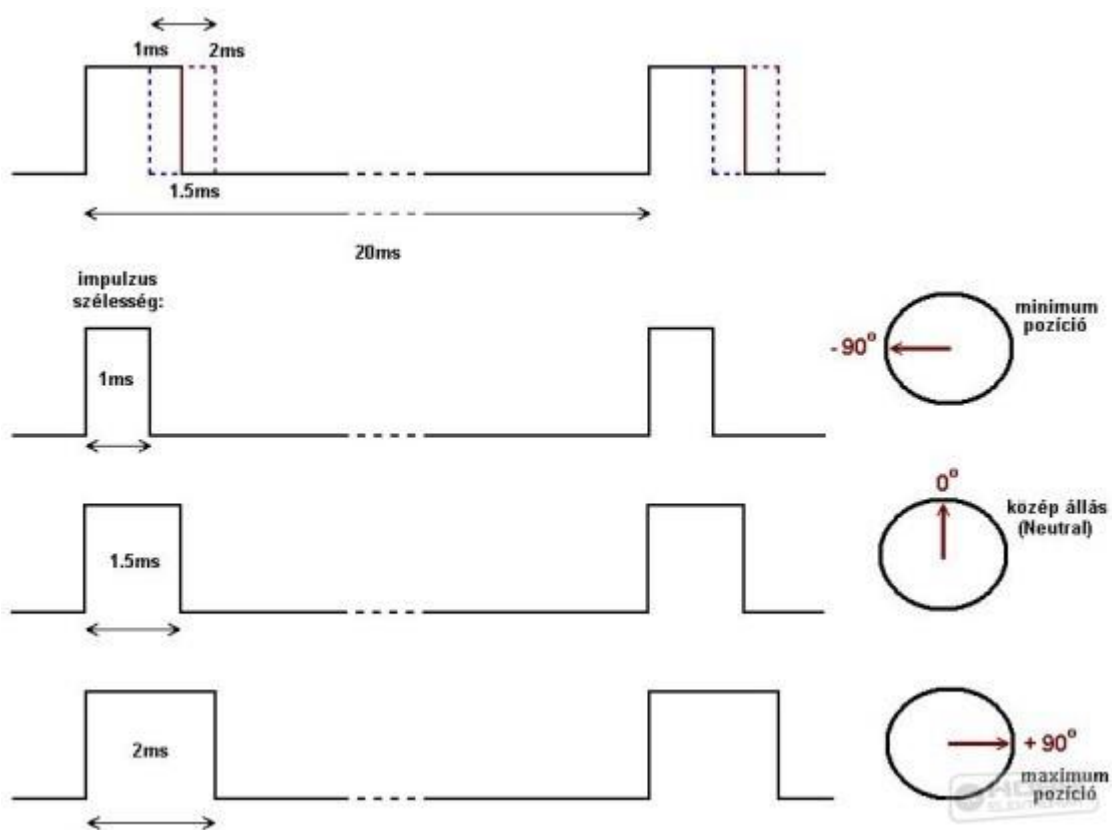
A szervomotorok vezérlése impulzus szélesség modulációval (PWM) történik, ami tulajdonképpen egy 50 Hz-es négyszögjel impulzus sorozat. Ez azt jelenti, hogy másodpercenként kb. 50-szer kell elküldenünk a jelet (tehát 20 ms-os időközönként), különben a szervomotor terhelés alatt nem tartja meg az adott pozíciót. Az impulzusok hossza határozza meg azt, hogy a szervomotor tengelye milyen szögbe áll be:

- 1 ms-os impulzus hatására a baloldali véghelyzetét veszi fel ( $-90^\circ$ )
- 1,5 ms-os impulzus hatására középpállásba áll ( $0^\circ$ )

- 2 ms-os impulzus hatására a jobboldali véghelyzetét veszi fel ( $+90^\circ$ )

Az említett jelalakokat a 13. ábra szemlélteti. Ezek az értékek szervónként változhatnak, az viszont minden szervomotorban közös, hogy 1,5 ms-os impulzus hatására középpállásba állítják a tengelyüket.

A szervomotorok felépítésüket tekintve tartalmaznak egy DC motort, egy ezt kivezélő H hidat, egy impulzusszélesség  $\rightarrow$  feszültség konvertert, amely egy referencia feszültséget állít elő, egy potenciómétert a tengelyre rögzítve, mely egy második referencia feszültség előállításáért felel, valamint egy komparátort, amely az előbb említett két referenciafeszültséget összehasonlítva eldönti, hogy a motort melyik irányba kell fordítani. Ennek köszönhetően nincs szükség szögadóra, mivel a motor mindig a kívánt szögbe áll be, ha csak valamilyen külső fizikai tényező nem akadályozza meg ebben.



13. ábra - szervomotor vezérlő impulzusai

Beszélhetünk továbbá analóg és digitális szervomotorokról. Az analóg szervomotoroknál fontos, hogy a kívánt szögnek megfelelő jel mindaddig jelen legyen a szervomotor vezérlőbemenetén, amíg a tengely eléri a kívánt pozíciót, digitális

szervomotorok esetében ez nem szükséges, mivel a digitális vezérlő mindaddig megtartja a jelet, míg a tengely el nem éri a kívánt szöget.

### 3.2.2. A BTM-222 bluetooth modul [8]

A robot vezetékek nélküli irányításához feltétlenül szükséges valamilyen nagyfrekvenciás interfész. A számítógéppel (vagy telefonnal) való kommunikáció miatt a választásom a Bluetooth protokollra esett, mivel ilyen jellegű modul már nagyon olcsón kapható, és a kezelése sem túl bonyolult (ellentétben a Wifi-s modulokkal).

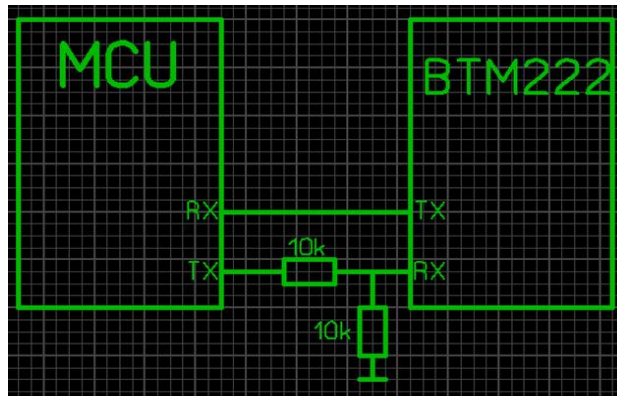
Az általam választott BTM-222 modul (13-as ábra) jellege szerint Class 1-es, ami azt jelenti, hogy a hatótávolsága körül-belül 100 m, természetesen ez nagyban függ a környezeti viszonyoktól.

A modul 3,3 V-os tápfeszültséget igényel, így a tápegységbe be kellett építeni még egy 3,3 V-os feszültség szabályzó IC-t a már meglévő 5 V-os mellé. Egy másik probléma, hogy a modul kommunikációs (RX) bemeneti magas szintje szintén 3,3 V, ellentétben a mikrokontroller 5 V-os (TTL szintű) kimenetével, így építeni kellett a két eszköz közé egy szintillesztő áramkört (esetünkben egy feszültségosztót) a 14. ábra szerint. A tranzisztoros szintillesztés elegánsabb megoldás lenne, azonban ez még jobban megbonyolítaná az áramkörünket, miközben a feszültségosztós szintillesztő is megbízhatóan működik esetünkben.

A modul az UART-on kívül alkalmas még USB és SPP kommunikációra is.



**13. ábra: BTM-222 Bluetooth modul**



14. ábra: az MCU és a BTM222 közti szintillesztés

Mivel a modult csak SMD kivitelben gyártják, így mindenképp folírozott lemezt kell hozzá készíteni, hogy könnyebben illeszthető legyen a moduláris vezérléshez.

A modul nem rendelkezik antennával, így erről is nekünk kell gondoskodni. Kereskedelmi forgalomban olcsón kaphatók Wifi antennák, amelyek szintén 2,4 GHz-en üzemelnek. Így már csak egy RPSMA csatlakozót kell a modult tartalmazó áramköri lapra forrasztanunk.

A modul rendelkezik még 3 állapotjelző led kivezetéssel, melyek közül egy a tápfeszültség jelző, egy a bluetooth kapcsolat jelző (ha villog: nincs párosítva eszköz, ha folyamatosan világít: kapcsolatban van) és egy adatforgalom jelző led.

Használat előtt a modult konfigurálnunk kell soros porton keresztül, valamilyen soros terminál program segítségével. A konfigurációhoz szükséges egy USB -> RS232 konverter, valamint egy szintillesztő IC (pl. MAX3232). A modult úgynevezett AT parancsok segítségével állíthatjuk be, ezek megtalálhatóak a modul adatlapjában.

### 3.2.3. A PIC16F877A mikrokontroller

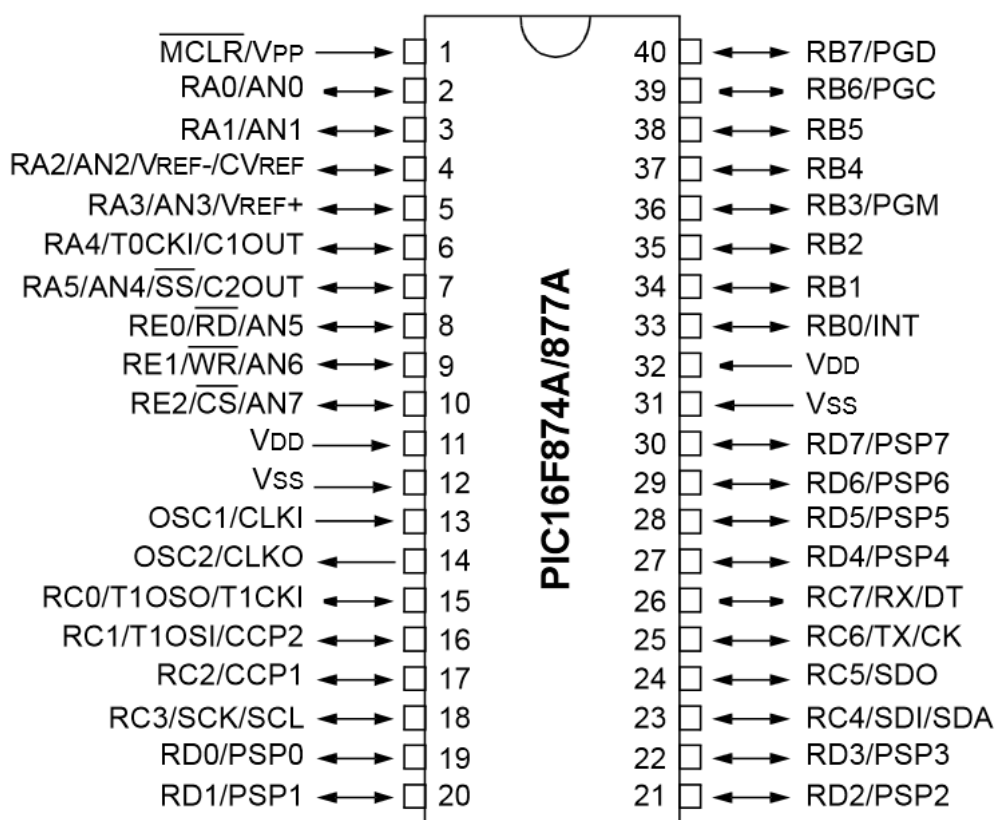
Azért választottam ezt a típusú mikrokontrollert a feladat megoldásához, mivel nagy számú I/O-al (33 db.) rendelkezik, valamint tartalmaz perifériás USART-ot, amivel a Bluetooth modullal való kommunikáció további szoftveres terhelés nélkül megvalósítható, nagy program memóriával (14,3 Kbyte) és RAM memóriával (368 Byte) rendelkezik. A mikrokontrollerbe ezen kívül számos hasznos perifériát építettek, ami megkönnyíti a későbbi fejlesztés során a szenzorok és egyéb kiegészítők illesztését a vezérlőhöz. Ezek a következők:

- 8 csatornás 10 bit-es ADC
- 2 analóg komparátor programozható referencia feszültséggel
- 3 darab timer modul (2 darab 8 bites és egy 16 bites)

- SPI és I<sup>2</sup>C kommunikáció
- EEPROM (256 Byte)
- 2 darab CAPTURE / komparátor / PWM modul (10 bit)

Mivel a mikrokontroller csak 2 PWM modult tartalmaz (míg a szervomotorok miatt nekünk 18-ra lenne szükségünk), kihasználhatjuk a nagy számú I/O portot, és a PWM modulokat helyettesíthetjük szoftveresen előállított PWM-el. Ennek hátránya, hogy bizonyos mértékben csökkenti a főprogram kihasználhatóságát, azonban megfelelő hangolással ez nem okoz különösebb problémát. Egy alternatív megoldás lenne a PWM modulok szétosztása külső multiplexer IC-k segítségével, azonban ez jóval bonyolultabb vezérlő áramkört és átláthatatlanabb programot eredményezne.

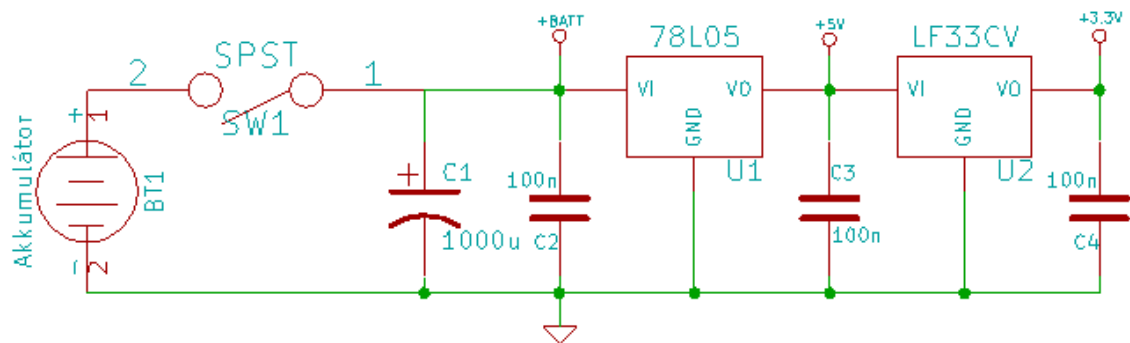
A PIC16F877A mikrokontroller lábkiosztása a 15. ábrán látható:



15. ábra - a mikrokontroller lábkiosztása

### 3.2.4. Kapcsolási rajzok

A kapcsolási rajzokat a KiCad nevű ingyenes szoftver segítségével készítettem. A tápegység kapcsolási rajza a 16. ábrán látható:

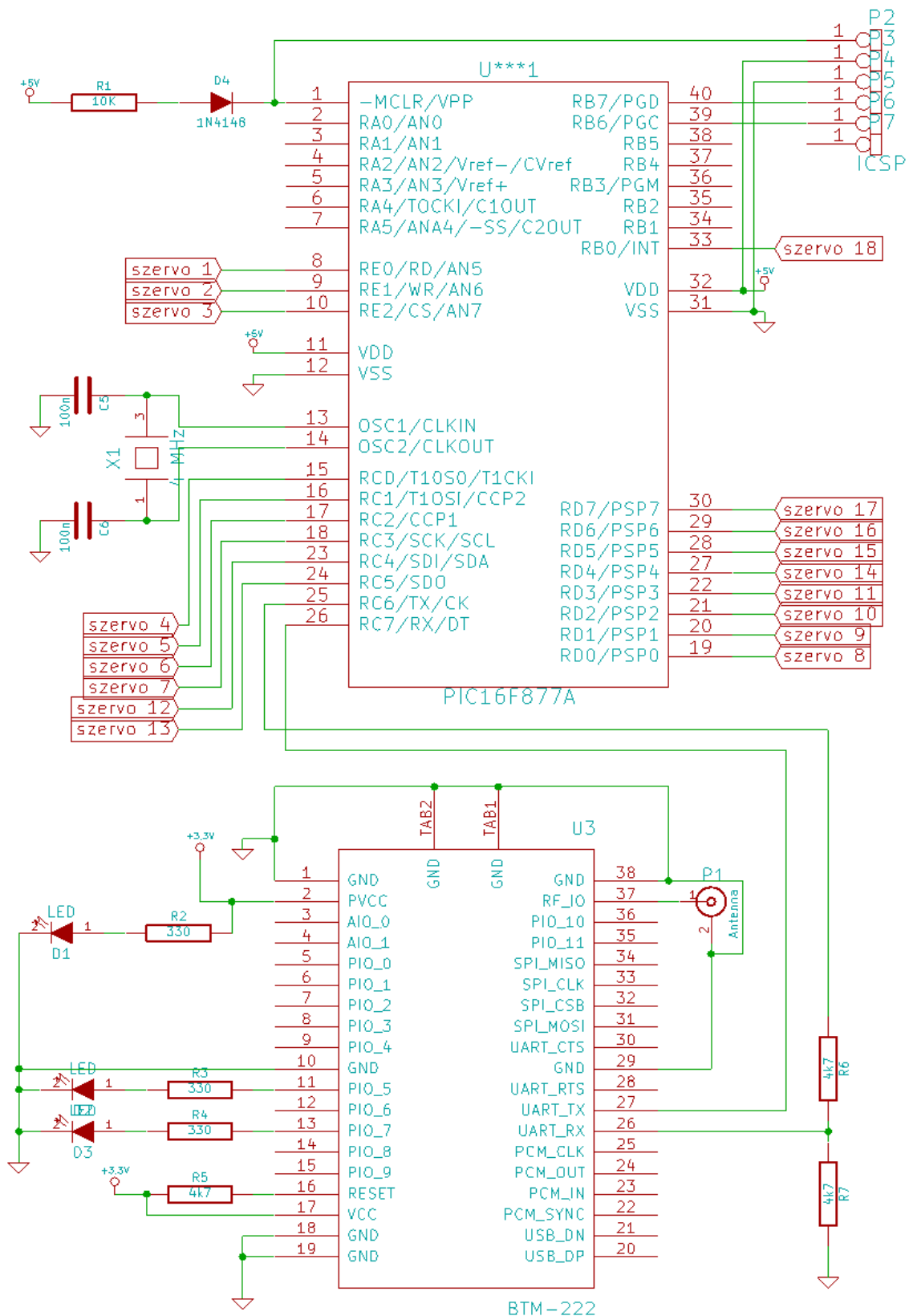


16. ábra - a tápegység kapcsolási rajza

A BT1 jelű alkatrész egy Winmax E73 típusú 7.4 voltos (2 cellás), 5200 mAh-s, 80C-s RC modellekhez tervezett lítium polimer akkumulátor. Ennek a típusú akkumulátornak az egyik nagy előnye, hogy az igen nagy teljesítményhez képest kis tömeggel (295 g) és mérettel (154 x 45 x 17 mm) rendelkezik, ami mobil robotok esetében nagy előnyt jelent. Hátránya azonban, hogy érzékeny a mechanikai behatásokra (ütés, hajlítás, hőmérséklet), emiatt nagyon körültekintően kell bánni vele. Töltés közben a túltöltés veszélye is fennáll, ezért ajánlott úgynevezett intelligens töltőt használni.

Az SW1 jelű kapcsoló a robot teljes áramtalanítására szolgál. A C1 elektrolit kondenzátor a tápegység puffer szerepét tölti be, erre főleg a motorok által létrehozott tranziensek miatt van szükség, míg a C2, C3 és C4 unipoláris kondenzátorok a magas frekvenciás zajok kiszűrésére szolgálnak. Az U1 és U2 egy-egy feszültség stabilizátor IC, a LM78L05 a mikrokontroller számára állít elő 5 V-os feszültséget, míg az LF33CV a Bluetooth modul számára állít elő 3,3 V-os feszültséget. A szervomotorokat feszültségstabilizálás nélkül táplálom, ugyanis az LM78L05 IC maximális terhelhetősége 1 A, ami a 18 darab szervomotorok számára nem lenne elegendő, még a stabilizátorra szerelt hűtőborda alkalmazása esetén sem.

A vezérlő kapcsolási rajza a 17. ábrán látható:

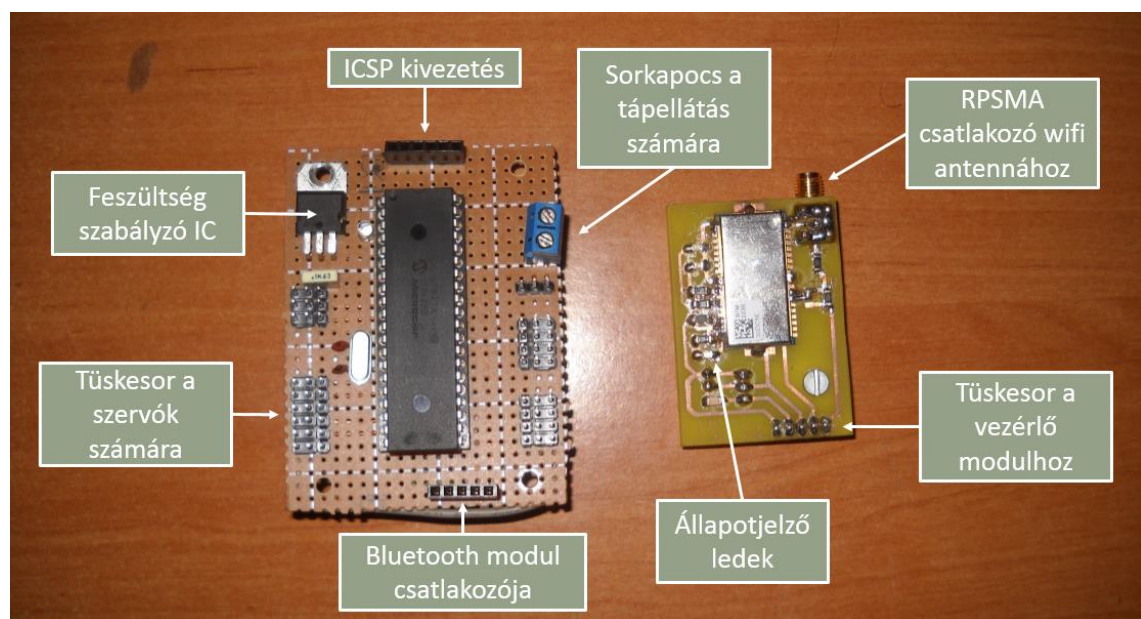


17. ábra - a vezérlő kapcsolási rajza

Az R1 ellenállás a mikrokontroller MCRL lábát húzza magas szintre a D4 diódán keresztül, melynek szerepe, hogy programozás esetén megvédje a programozó

áramkört. Az X1 kristály a C5 és C6 kondenzátorokkal együtt a mikrokontroller órajelét biztosítja. Az ICSP csatlakozó szerepe, hogy a későbbiekben a mikrokontroller áramkörből való kivétele nélkül is programozható, ill. debuggolható legyen. Az R2, R3 és R5 ellenállások szerepe a D1, D2 és D3 SMD ledek áramának korlátozása, mind a ledek élettartamának, mind pedig a modul portjainak védelme érdekében. A D1 jelű alkatrész a bekapcsolást jelző led (zöld), a D2 a bluetooth kapcsolat jelző led (kék), a D3 pedig az adatforgalom jelző led (piros). Az R5 ellenállás a BTM-222 modul RESET lábát húzza magas szintre, mivel ez egy negált bemenet, így alacsony jelszint esetén resetelődik a modul, amire nekünk jelen esetben nincs szükségünk. A P1 jelű alkatrész egy RPSMA csatlakozó wifi antennához. Az R6 és R7 ellenállások egy feszültségosztót képeznek, amely a mikrokontroller TX portjáról érkező jel feszültséget osztja le a felére (kb 2.5 V), ugyanis a BTM-222-es modul az 5 V-os jelszint hatására nagy valószínűséggel tönkremenne.

### 3.2.5. Az elkészített áramkörök



18. ábra - a vezérlő- és a bluetooth áramkör képei

Az általam készített vezérlő és bluetooth áramköröket a 18. ábrán láthatjuk. Az ábrán feltüntettem a fontosabb egységeket. A könnyű szerelhetőség érdekében a szervomotorok kivezetéseit tűskesorral oldottam meg, így az összerakás, illetve szétszedés nem igényel forrasztást. A a bluetooth modul áramköre külön nyomtatott áramkört kapott helyet, így a modul könnyebben konfigurálható, valamint felhasználható más robotok kommunikációjának kialakítására is. Ez szintén tűskesor

segítségével csatlakoztatható a főáramkörhöz. Az akkumulátor egy nyomtatott áramkörbe ültethető sorkapocs segítségével kapcsolódik a főáramkörhöz. Az ICSP kivezetésre a robot programozása miatt van szükség, ennek segítségével nem szükséges a mikrokontroller átültetése programozó áramkörbe, mindössze egy programozó kábelt kell a robothoz csatlakoztatni és már programozható, illetve debuggolható is. A bluetooth áramkörön helyet kaptak még az előző fejezetben említett állapotjelző led, valamint egy RPSMA csatlakozó Wifi antenna csatlakoztatásához, ez azonban kis távolságok esetén elhanyagolható.

### 3.3. A szoftver

A mikrokontroller szoftvere C nyelven íródott a Microchip cég által forgalmazott Mplab környezetbe integrálható Hitech C fordító segítségével. Ennek előnye a hagyományos Assembly fordítóval szemben, hogy magasabb szintű programozási nyelven írhatjuk a mikrokontrollerre szánt programunkat, ami lényegesen egyszerűbb és átláthatóbb, hátránya azonban, hogy az igen „kényelmes” C nyelv függvényeit használva sokkal kevésbé optimalizált kódot kapunk, ami sajnos a programmemória felesleges kihasználtságához vezet, különösképpen ha a Hitech C fordító ingyenes változatát használjuk, ami a gyártó szerint 40%-al optimalizálatlanabb kódot eredményez fizetős változatánál. Szerencsére középkategóriás mikrokontrollerek esetében, különösképp a Microchip PIC16F877A processzora esetén ez egyáltalán nem okoz problémát, mivel még az optimalizálatlan kód is a programmemória mindössze 16%-át foglalja el, a Flash memóriának pedig 9%-át, így a mikrokontroller az alapfeladatok ellátása mellett még alkalmas egyéb feladatok ellátására is (mozgási algoritmusok futtatása, szenzorok jeleinek feldolgozása, stb..).

A program úgy lett megírva, hogy a PWM jelek előállítása egy interrupt (megszakítás) rutin segítségével történik, amelyet egy a jel periódus idejére beállított timer (időzítő) hív meg folyamatosan, függetlenül a főprogramban éppen futó eseményektől. Ennek előnye, hogy a servók számára előállított magas jelszint mindig azonos időben kerül kiküldésre az összes szervomotornak, így látszólag azok párhuzamosan működhetnek és a főprogramban történő egyéb feladatokat sem kell 2 ms-nél hosszabb időtartamra megszakítani. Az előállítandó PWM jel frekvenciája 50 Hz, ami azt jelenti, hogy 20 ms-os periódus időre van szükségünk. A PIC mikrokontrollereknek architektúrájukból adódóan egy utasítás végrehajtásához 4 órajel

szükséges, ami azt jelenti, hogy 4 MHz-es oszcillátor használatával egy utasítást a processzor 1 us alatt hajt végre (tehát egy másodperc alatt egy millió utasítás hajtódik végre), kivéve azokat az utasításokat, amelyekhez két vagy még több instrukciós órajel szükséges. Alapesetben az időzítőnk minden egyes instrukciós órajel hatására növelné az értékét egyel, azonban a mikrokontroller tartalmaz egy úgynevezett előosztót (prescaler), amelyet többek között az időzítőhöz is be lehet állítani. Az előosztó értékét mi választhatjuk ki, ezek a kettő hatványai 1-től egészen 256-ig. A megszakítás rutin meghívódik az időzítő minden egyes túlsordulásánál (amikor 255-ből 0-ba vált). 64-es előosztót választva a megszakítás rutinunk pontosan 16 384 us-ként fog meghívódni ( $64 \times 256 \times 1 \text{ us}$ ), ami ugyan nem 20 ms, de szerencsére a TowerPro MG995 típusú szervomotor vezérlőjének még beleesik a toleranciahatárába.

A megszakítás rutin viszonylag kevés kódot tartalmaz, feladata annyi, hogy az összes szervomotorok fenntartott digitális kimenetet felkapcsolja, majd indít egy „for” ciklust, melyen belül az egyes szervomotorok számára fenntartott változók értékeit összehasonlítja a ciklus futóindexével, és amennyiben ezek megegyeznek, az illető szervomotor kimenetét lekapcsolja, így szabályozva a PWM jelben lévő magas szint idejének hosszát. Itt tulajdonképpen késleltetés már nem is szükséges, az „if” elágazáshoz ugyanis 4 instrukciós órajel szükséges (tehát 4 us), ami a 18 szervomotor esetén már 72 us. Ez azt jelenti, hogy a „for” ciklus futóindexének minden egyes értékére 72 us-os késleltetés esik. 27-ig való számlálás esetén ez már  $27 \times 72 = 1\,944 \text{ us}$   $\approx 2 \text{ ms}$ , tehát késleltetés nélkül is előáll a szervomotor véghelyzetbe való forgatásához szükséges időintervallum.

### 3.3.1. A program forráskódja

```
#include <xc.h>
#include <stdint.h>
#include "uart.h"
```

Az xc.h header fájl a mikrokontroller regisztereit és makróit tartalmazza. Az stdint.h header fájl az „uint8\_t” adattípus definícióját tartalmazza, mely egy előjelnélküli 8 bites egész típus, azonban a fordító erre nem tartalmaz beépített adattípust és mivel a programban használt változók nem haladják meg a 255-ös értéket ( $2^8$ ), így a memória szempontjából nagyon veszteséges lenne integerként definiálni őket. Az uart.h felhasználó által definiált header fájl az UART kommunikációhoz szükséges függvényeket tartalmazza.

```
#define _XTAL_FREQ 4000000
```

Ezzel a sorral tudatjuk a fordítóval, hogy az általunk használt kvarc kristály 4 MHz-es. Ez a késleltető (delay) függvények miatt szükséges.

```
#define MIN 6+2
#define MID 16
#define MAX 26-2
```

Előre definiáljuk a szervomotorok véghelyzeteit, így a program további részeiben már a MIN, MID és MAX makrókkal tudunk rájuk hivatkozni, valamint a szervomotorok szinkronizálása esetén elég csak a program elején megadnunk ezeket az értékeket.

```
__CONFIG(DEBUGDIS & XT & WDTDIS & PWRTEN & BORDIS & LVPDIS
& WRTEEN & DUNPROT & UNPROTECT);
```

Ezzel a makróval tudatjuk a fordítóval a mikrokontrollerünk alapbeállításait: áramkörön belüli debug-olás kikapcsolva, alacsony sebességű kristály (XT) oszcillátort használunk, Watch Dog Timer kikapcsolva, Power-up időzítő bekapcsolva, Brown-Out Reset kikapcsolva, áramkörön belüli kifeszültségű soros programozás kikapcsolva, Flash program memória írás bekapcsolva, DUN védelem bekapcsolva, kódvédelem kikapcsolva.

```
uint8_t i,s[18];  
char buff;
```

Definiáljuk a változókat: „i” a ciklusváltozónk, az „s” egy 18 elemű tömb, amelyben a szervók beállítani kívánt értékeit tároljuk, ezek 8 bites előjelnélküli egész típusok. A „buff” változó egy 8 bites karakter típus, amely ideiglenesen tárolja az UART-on érkező karaktert a kommunikációhoz.

```
void main()  
{  
    TRISB = 0;  
    TRISC = 0;  
    TRISD = 0;  
    TRISE = 0;
```

Az szervomotorok számára fenntartott I/O-kat beállítjuk kimenetként.

```
UART_Init(9600);
```

Ezzel a függvénnyel inicializáljuk a mikrokontroller UART perifériáját 9600 kbit/s-os átviteli sebességgel.

```
OPTION_REGbits.T0CS = 0;  
OPTION_REGbits.PSA = 0;  
OPTION_REGbits.PS = 0b101;
```

Engedélyezzük az időzítő üzemmódot, az előosztót is ehhez állítjuk be, majd az előosztást 64-re állítjuk.

```
INTCONbits.T0IE = 1;  
ei();
```

Engedélyezzük, hogy a megszakítás rutin az időzítő hatására hívódjon meg, ezután engedélyezzük a globális megszakításokat.

```
for(i=0; i<18; i++) s[i] = MID;
```

Az összes szervomotort középhelyzetbe állítjuk.

```
UART_Write_Text("A hexapod keszen all!");
```

Üzenetet küldünk a PC számára, hogy a robot üzemképes.

```

for(;;)
{
    buff = UART_Read();

    switch(buff)
    {
        case 'a': s[0] = UART_Read(); break;
        case 'b': s[1] = UART_Read(); break;
        case 'c': s[2] = UART_Read(); break;

        ...

        case 'r': s[17] = UART_Read(); break;
        default : break;
    }
}

```

Végtelen ciklusban UART-ról beolvasunk egy karaktert, majd ez alapján eldöntjük, hogy a következő karakter melyik szervomotor változójába kerüljön.

```

void interrupt isr(void)
{
    PORTB = 255;
    PORTD = 255;
    PORTC = 255;
    PORTE = 7;

    for(i=0; i<27; i++)
    {
        if(s[0]==i) PORTEbits.RE0 = 0;
        if(s[1]==i) PORTEbits.RE1 = 0;

        ...

        if(s[17]==i) PORTBbits.RB0 = 0;
    }

    INTCONbits.T0IF = 0; // clear interrupt flag
}

```

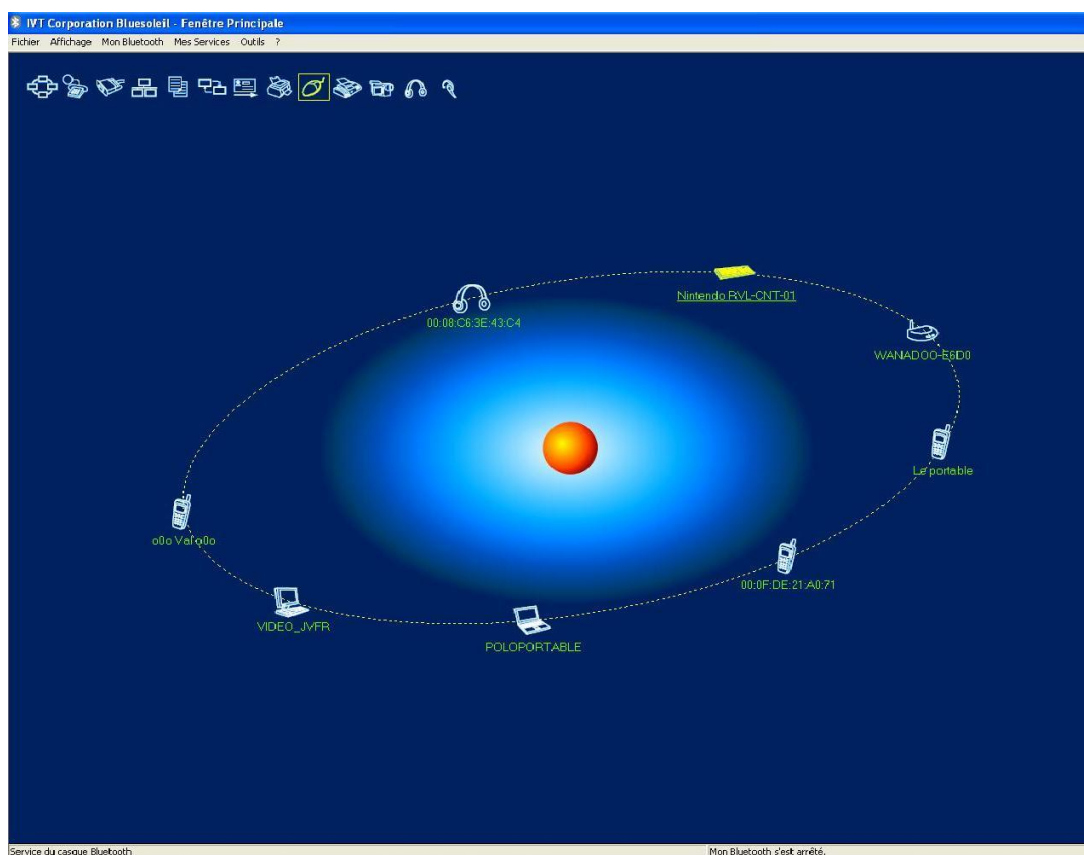
A megszakítás rutinban először az összes szervomotor kimenetet magas szintre állítjuk, majd a „for” cikluson belül ha az illető szervomotor változójában található érték megegyezik a ciklusváltozóval, akkor kikapcsoljuk. Végül az interrupt flag (T0IF) törlésével jelezzük a mikrokontrollernek, hogy lekezeltük a megszakítást.

### 3.3.2. Felhasznált szoftverek

Az USB bluetooth eszköz csatlakoztatása után egy Blue Soleil nevű programot (19. ábra) kell futtatnunk, mely segítségével Bluetooth modulunkat mint soros port láthatjuk a gépen a megfelelő beállítások elvégzésével. Ez után már bármilyen soros terminál programból ellenőrizhetjük a kommunikációt a robottal.

A robot szervomotor pozícióinak változtatásához a terminálban egy betűt és egy számot kell elküldeni a robot számára, ahol a betű jelöli a szervomotort „a”-tól „r”-ig, ahol az „a” a legelső szervomotor az „r” pedig a legutolsó. Az ezután következő szám jelöli a szervomotor tengelyének pozícióját, ez 6 és 26 között változhat.

Pl: „c16” -> a 3. szervomotor tengelyét középhelyzetbe állítjuk.



19. ábra - Blue Soleil program

A kód ismeretével már könnyedén írhatunk bármilyen programot soros portra, amivel a robotot vezérelhetjük, így már az algoritmusaink erőforrásigényeinek csak a számítógépünk szabhat határt, nem pedig az erősen korlátozott erőforrásokkal rendelkező mikrokontroller.

## 4. Eredmények

A tényleges munka előtti irodalomkutatásnak köszönhetően kellő információt sikerült gyűjteni a már meglévő technológiákról, hexapodok mechanizmusának konstrukciós megoldásairól, vezérlő áramkörökről, szervomotorokról, és a vezérléshez szükséges szoftverekről.

Ez után sikerült megtervezni egy saját kialakítású hexapod robot vázat, majd ezt dokumentálni AutoCad segítségével. Később ezt sikerült fizikailag is megvalósítani lézeres kivágással.

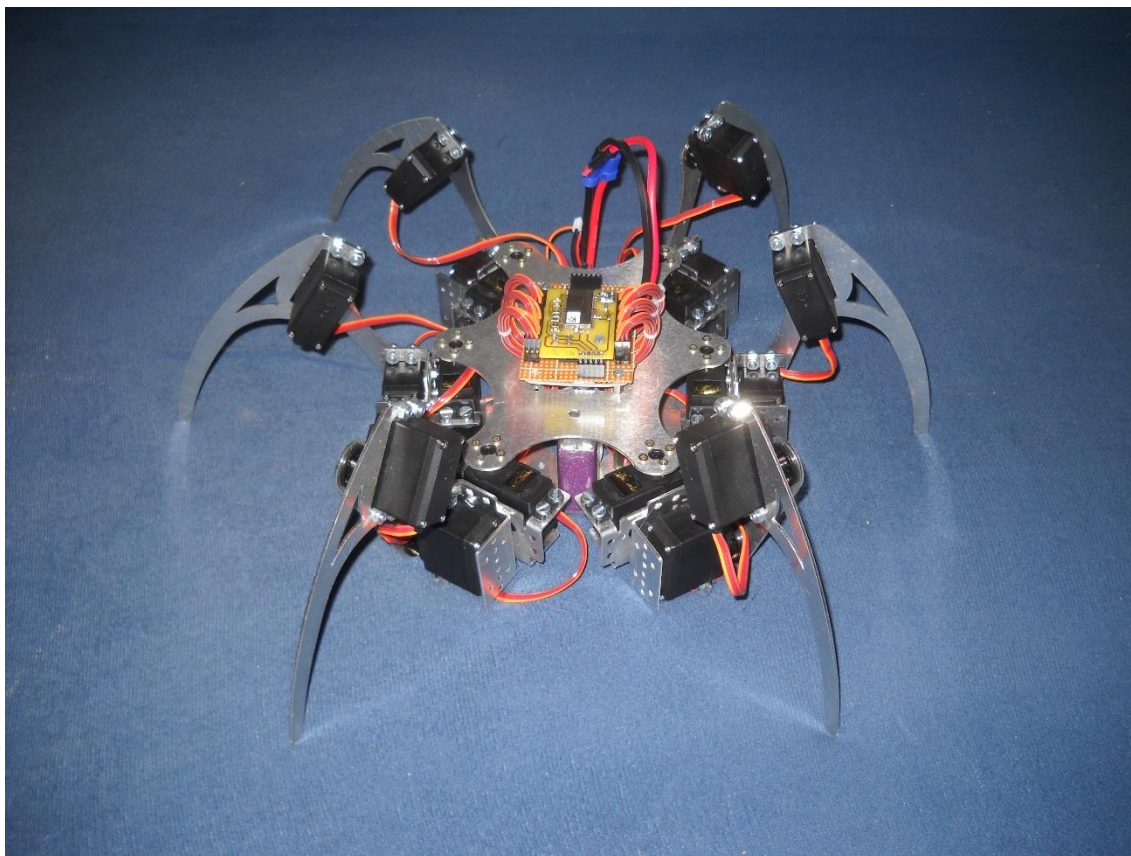
A következő lépés a vezérlő áramkör megtervezése volt, ami a megfelelő mikrokontroller és Bluetooth modul kiválasztásának köszönhetően nem jelentett különösebb problémát.

Az tervezés utolsó fázisa, a mikrokontroller szoftverének megírása viszonylag bonyolult feladatnak bizonyult. 18 szervomotor számára PWM jelet egy időben, párhuzamosan előállítani szoftveresen úgy, hogy közben a mikrokontroller más feladatok ellátására is alkalmas legyen, embert próbáló feladat, de szerencsére, ha nem is elsőre, ezt a problémát is sikerült megoldani.

A robot felélesztése után már csak a tesztelés maradt hátra. Az egyénileg kifejlesztett PWM-es szervomotor vezérlő a vártnál sokkal jobban teljesített, a megszakítás rutinba helyezett PWM jel generálás a megfelelő hangolásnak és a ciklusidő jó közelítésének köszönhetően szinte észrevehetetlen a robot mozgásában, a program további részében pedig rengeteg idő (18 ms) marad a további feladatok végrehajtására (kb. 18 000 utasítást hajt végre a mikrokontroller ennyi idő alatt). A mechanizmus (és ezáltal az egész robot) kellően könnyű lett ahhoz, hogy a robot gyors mozgásra is képes legyen, valamint igen látványos „fémes” rovar kinézetet kölcsönöz a robot számára. A vezérlő áramkör és a Bluetooth-os kommunikáció stabilan működik, a robot jól vezérelhető nagyobb távolságok esetén is.

A továbbiakban elkezdtem a különböző mozgási és járási algoritmusokat, valamint viselkedési formákat leprogramozni, melyek fejlesztése és tesztelése a mai napig is tart.

Az elkészült robot a 20. ábrán látható.



**20. ábra - az elkészült robot**

Az elkészült robot első mozdulatai megtekinthetők a következő videón:

[https://www.youtube.com/watch?v=Mw2woZGBb\\_Q](https://www.youtube.com/watch?v=Mw2woZGBb_Q)

Külön köszönetet szeretnék mondani konzulensemnek, Vitéz Attilának, aki nélkül ez a dolgozat nem jöhetett volna létre, valamint Szabó Gábor hallgatótársamnak a mechanikai munkálatokban való közreműködéséért.

## 5. Összefoglalás

A dolgozat lényege egy olyan hat lábon járó irányítható robot (platform) tervezési fázisainak bemutatása, amely szenzorok és beavatkozók segítségével egyénileg testre szabható, így a legkülönbébb feladatok ellátására lehet alkalmas, mint:

- anyagmozgatás egyenlőtlen, sziklás talajon
- biológiai elméletek tanulmányozása a rovarok mozgásairól, motorikus idegekről és a neurobiológiáról
- különböző járási algoritmusok prezentálására
- természetből inspirálódott robotok mechanizmusainak fejlesztésére
- robotok számítógéppel való kommunikációjának fejlesztésére

A számítógépes vezérlésnek köszönhetően olyan komplex algoritmusok is fejleszthetők a robotra, amelyeket egy mikrokontroller erőforráshiánya vagy egyéb okok miatt már nem lenne képes futtatni, mint például:

- inverz kinematikai algoritmus a lábak nyílt láncú mozgásához
- valós idejű 3D-s szimuláció a robot helyzetéről, mozgásáról
- ultrahangos szenzor segítségével információ szerzés a térbeli elemekről
- kép-, illetve hangfelismerés és ezáltal mesterséges intelligencia kialakítása
- naplózható eseménysorozat

## 6. Irodalomjegyzék

1. Dr. Husi Géza: A mechatronika alapjai, Debrecen, ISBN: 978-615-5088-09-4
2. Ulrich Tzietze, Cristoph Schenk: Analóg és digitális áramkörök, Budapest, 1990, ISBN: 9631082091, Fordította: [Dr. Huszár Imre](#)
3. Kónya László-Kopják József: PIC Mikrovezérlők alkalmazástechnikája, PIC programozás C nyelven (harmadik, bővített kiadás)
4. [Benkő László](#), [Benkő Tiborné](#), [Tóth Bertalan](#): Programozzunk C nyelven!, 2010, Kiadó: Computerbooks, ISBN: 9789636183660
5. Wikipédia: Hexapods - [http://en.wikipedia.org/wiki/Hexapod\\_%28robotics%29](http://en.wikipedia.org/wiki/Hexapod_%28robotics%29) (2013.11.30)
6. Majláth Zoltán: AVR vezérlésű Hexapod robot építése - <http://hobbyrobot.hu/content/avr-vezerlesu-hexapod-robot-epitese> (2013.11.30)
7. [Hogyan működnek a szervók?](#) - [http://kecskemetircmodell.blog.hu/2011/03/22/hogyan\\_mukodnek\\_a\\_szervok](http://kecskemetircmodell.blog.hu/2011/03/22/hogyan_mukodnek_a_szervok) (2013.11.30)
8. BTM-222 Bluetooth modul - <http://hobbyrobot.hu/content/btm-222-bluetooth-modul> (2013.11.30)
9. Ligo George: Using UART of PIC Microcontroller – Hi Tech C - <http://www.electrosome.com/uart-pic-microcontroller-hi-tech-c/> (2013.11.30)

## 7. Ábrajegyzék

|                                                                                                                                                                                                 |    |
|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----|
| 1. ábra - Hexapod robot a valóságban.....                                                                                                                                                       | 4  |
| <a href="http://en.wikipedia.org/wiki/Hexapod_%28robotics%29">http://en.wikipedia.org/wiki/Hexapod_%28robotics%29</a>                                                                           |    |
| 2. ábra - RiSE robot falmászás közben .....                                                                                                                                                     | 5  |
| <a href="http://kodlab.seas.upenn.edu/RiSE/Home">http://kodlab.seas.upenn.edu/RiSE/Home</a>                                                                                                     |    |
| 3. ábra - ATHLETE robot hegymászás közben .....                                                                                                                                                 | 5  |
| <a href="http://myprojects.hu/pic/servo-motorok-vezерlese.html">http://myprojects.hu/pic/servo-motorok-vezерlese.html</a>                                                                       |    |
| 4. ábra - szervomotor belső felépítése .....                                                                                                                                                    | 8  |
| <a href="http://myprojects.hu/pic/servo-motorok-vezерlese.html">http://myprojects.hu/pic/servo-motorok-vezерlese.html</a>                                                                       |    |
| 5. ábra - szabványos szervomotor méretei .....                                                                                                                                                  | 9  |
| <a href="http://www.myrcmart.com/rcx-mg5509-standard-metal-gear-servo-55g-95kg-023sec-p-3854.html">http://www.myrcmart.com/rcx-mg5509-standard-metal-gear-servo-55g-95kg-023sec-p-3854.html</a> |    |
| 6. ábra - Lepidoptera (lepke) lábának felépítése .....                                                                                                                                          | 9  |
| <a href="http://en.wikipedia.org/wiki/Arthropod_leg">http://en.wikipedia.org/wiki/Arthropod_leg</a>                                                                                             |    |
| 7. ábra - a láb köztes és hosszanti részének befoglaló méretei .....                                                                                                                            | 11 |
| 8. ábra - univerzális servo konzol méretei és 3D-s látképe.....                                                                                                                                 | 11 |
| <a href="http://imall.iteadstudio.com/im120815010.html">http://imall.iteadstudio.com/im120815010.html</a>                                                                                       |    |
| 9. ábra - servo konzol kiterített látképe .....                                                                                                                                                 | 12 |
| 10. ábra - az törzs alsó és felső lemeze befoglaló méretekkel .....                                                                                                                             | 13 |
| 11. ábra - a kész alkatrészek .....                                                                                                                                                             | 13 |
| 12. ábra: A vezérlés sematikus ábrája.....                                                                                                                                                      | 14 |
| 13. ábra: BTM-222 Bluetooth modul .....                                                                                                                                                         | 16 |
| <a href="http://www.tme.eu/hu/katalog/bluetooth-modulok_113196/">http://www.tme.eu/hu/katalog/bluetooth-modulok_113196/</a>                                                                     |    |
| 14. ábra: az MCU és a BTM222 közti szintillesztés.....                                                                                                                                          | 17 |
| <a href="http://hobbyrobot.hu/content/btm-222-bluetooth-modul">http://hobbyrobot.hu/content/btm-222-bluetooth-modul</a>                                                                         |    |
| 15. ábra - a mikrokontroller lábkiosztása.....                                                                                                                                                  | 18 |
| <a href="http://ww1.microchip.com/downloads/jp/DeviceDoc/39582b.pdf">http://ww1.microchip.com/downloads/jp/DeviceDoc/39582b.pdf</a>                                                             |    |
| 16. ábra - a tápegység kapcsolási rajza .....                                                                                                                                                   | 19 |
| 17. ábra - a vezérlő kapcsolási rajza .....                                                                                                                                                     | 20 |
| 18. ábra - a vezérlő- és a bluetooth áramkör képei .....                                                                                                                                        | 21 |
| 19. ábra - Blue Soleil program.....                                                                                                                                                             | 27 |
| <a href="http://www.harisprasetyo.web.id/2012/05/data-transfer-fungsi-lewat-bluetooth.html">http://www.harisprasetyo.web.id/2012/05/data-transfer-fungsi-lewat-bluetooth.html</a>               |    |
| 20. ábra - az elkészült robot .....                                                                                                                                                             | 29 |