



SZENT ISTVÁN EGYETEM
Gépészmérnői Kar
Gépipari Technológiai Intézet
Mechatronika Tanszék



TDK DOLGOZAT
HAGYOMÁNYOS SZERSZÁMGÉPEK UTÓLAGOS
AUTOMATIZÁLÁSA LÉPTETŐMOTOROKKAL

Készítette:

Erdélyi Viktor Ferenc
Mechatronikai Mérnői szak, IV. évfolyam

Témavezető: Dr. Jánosi László egyetemi docens, tanszékvezető

GÖDÖLLŐ
2012

1. Bevezetés	1
1.1. Témaválasztás	1
1.2. A téma indoklása	1
1.3. Célkitűzés	1
2. Automatikus gyártógépek főbb elemeinek áttekintése	2
2.2. Léptetőmotoros CNC gépek	3
2.3. A léptetőmotorok bemutatása	4
2.4. Mikrokontroller, mint a beágyazott rendszer alapja	12
4.2. Mikrokontrollerek alkalmazása	14
3. Szerszámgépek utólagos automatizálásának tervezése	15
3.1. Rendszerterv	15
3.2. Hardver	17
3.3. Felhasznált villamos elemek	24
3.4. A működtető szoftver tervezése	31
4. A kísérleti eszköz megépítése és vizsgálata	34
4.1. A kísérleti eszköz megépítése	34
4.2. A kísérleti eszköz vizsgálata	37
5. Gazdasági megfontolás	43
6. Összefoglalás	45
7. Irodalomjegyzék	47
8. Köszönetnyilvánítás	48
9. Mellékletek	49
1. Melléklet: Kapcsolási rajz	50
2. Melléklet: PIC16F913 blokkdiagramja	51
3. Melléklet: PIC16F917 blokkdiagramja	52
4. Melléklet: A motorvezérlő programja	53
5. Melléklet: Az LCD vezérlő programja	60
6. Melléklet: Az USB/RS232 illesztő modul programja	66
7. Melléklet: A kész kísérleti eszköz fényképe	70

1. Bevezetés

1.1.Témaválasztás

Mechatronikai mérnöki szakon, azon belül is gyártásmechatronika szakirányon folytatom tanulmányaimat, így kézenfekvőnek tűnt valamilyen gyártásmechatronikai problémát választani a TDK dolgozatom témájaként. A korábbi években C tárgyak keretein belül (Agromechatronika, Beágyazott rendszerek) megismerkedtem a mikrokontrollerek világával, azok alkalmazástechnikájával, a TDK dolgozatomban nagy szerepet kap a beágyazott számítástechnika. Ezen két témát kimeríti a CNC gépek vezérlése. Az Exasol Kft.-nél végzett nyári gyakorlat alkalmával megismerkedtem egyszerű lézergravírozógépekkel, így a téma tálcán kínálta magát. Mivel egy CNC gépet léptetőmotorok segítségével jóval egyszerűbb és költségkímélőbb megvalósítani, mint szervorendszerek alkalmazásával, így erre a megoldásra esett a választásom.

1.2.A téma indoklása

A világban az utóbbi évek során rohamosan elterjednek a számítógép vezérlésű szerszámgépek[1.][2.], mivel jó tulajdonságaik keresetté tették őket az ipar számára. A hagyományos szerszámgépekkel szemben a CNC gépek, -mivel programot hajtanak végre- a gyártás során azonos minőséget biztosítanak, Gyorsabb megmunkálást tesznek lehetővé, alacsonyabb képzettségű kezelőszemélyzetre van szükség, és a gyártott munkadarabok geometriája is bonyolultabb lehet. [3.] A hátránya a CNC gépeknek azonban, hogy a hagyományos manuális szerszámgépekhez képest jóval nagyobb a beszerzési költségük. Így a kis és középvállalatoknál még gyakran előfordulnak hagyományos megmunkáló gépek is. Ezek utólagos automatizálása költségkímélő alternatívát kínál a költséges CNC gépek beszerzésére.

1.3. Célkítűzés

Hagyományos szerszámgépek utólagos automatizálását lehetővé tévő, költséghatékony, moduláris felépítésű rendszer megtervezése.

2. Automatikus gyártógépek főbb elemeinek áttekintése

2.1. A CNC technológia fejlődése

„A különböző gépi megmunkálási technológiák szüntelen továbbfejlődésén belül a gépek számvezérlésének megjelenése nagyon jelentős lépés. A bonyolult NC gép beszerzési ára miatt a gép költségek általában magasak, azonban az előkészületi költség kicsi. A hagyományos, mechanikus vezérlésű automatáknál ez éppen fordítva van. Egy bizonyos darabszámnál kisebb gyártási volumen esetében a számvezérlésű gép lesz gazdaságosabb. A számvezérlésű gép felépítésénél fogva ugyanakkor még, például adaptív vezérléssel, jobban biztosítja a maximális, illetve optimális teljesítményt, amely gazdaságosságát növeli. Az NC - technika, amely a Numerical Control angol szavak kezdőbetűiből származik és a magyar fordítása "számvezérlés". Létrejöttét, fejlődését az alábbi évszámokhoz és kutatókhoz köthetjük:

- 1808. J. M. Jaoquard, francia selyemszövő és mechanikus, a róla elnevezett szövőszéken lyukkártyákat alkalmazott. Ez jelentette a cserélhető adathordozó feltalálását.
- 1938. C. E. Shannon, amerikai matematikus doktori disszertációjában arra az eredményre jutott, hogy gyors számítás, adatátvitel csak bináris formában, a Boole algebra felhasználásával történhet és a megvalósítás eszközei az elektronikus kapcsolók lehetnének. Shannon dolgozta ki a mai számítógépek, illetve számjegyvezérlések alapelveit.
- 1946. J. W. Mauchly és J. P. Eckert az USA-ban elkészítette az első digitális számítógépet. Ők teremtették meg az elektronikus adatfeldolgozás alapjait.
- 1949-52. J. Parsons és az M. I. T. (Massachusetts Technológiai Intézet) fejlesztett ki egy rendszert, amelyben a munkadarab megmunkálását közvetlenül egy számítógép vezérelte

.Az elképzelés négy lényeges tétele, a következő volt:

1. Egy pálya elérendő pontjainak tárolása lyukkártyákon.
2. A lyukkártyák adatainak automatikus beolvasása a gépbe.
3. A beolvasott helyzetek folyamatos kiadása és közbensőértékek számítása annak érdekében, hogy
4. a szervomotorok a tengelyek mozgását vezérelni tudják

.Az első működő magyar NC-gép az ERS 200 szakaszvezérlésű eszterga volt, amit Csepelen gyártottak és az 1960-as BNV-n mutattak be.” [4.]

2.2. Léptetőmotoros CNC gépek

Régebben a szerszámgépeknél pl CNC gépeknél léptetőmotorokat alkalmaztak, melyeknek nagy hátrányuk volt az álló helyzetben magas hődisszipáció, és a lassú mozgás(kb 500 l/min). Ezen kívül a vezérlésük is meglehetősen bonyolult volt, hiszen, ha sokpólusú léptető motorról volt szó, az irányváltás már komolyabb programozási feladat volt. Ellehetetlenítette még a léptetőmotorok használatát az a tény, hogy léptetőmotorok esetében a kisebb teljesítményű motorok voltak gazdaságosak, a nagyteljesítményű stepperek már igen drágák voltak, és igen nagyok. A léptetőmotorok alkalmazásának egy nagy előnye volt, mégpedig az, hogy nem kellett visszacsatolást helyezni a körbe, mivel a léptetőmotort elég vezérelni, és jellegéből adódóan a kívánt (a motor felbontásától függő) helyzetbe áll. Ebben rejlett a további hátránya is, pl ha a gép nem tudott a kívánt helyzetbe állni, arról nem volt semmiféle visszajelzés. Ilyen gépek voltak pl. az EMCO CNC-k.

Természetesen a mai napig alacsony teljesítményszükséglet esetén (pl. gravírozás lézer, plazmavágás) kedveltek ezen megoldások már csak a vezérlés olcsó mivolta miatt is.

2.3. A léptetőmotorok bemutatása

Fontos előre tisztázni, hogy a léptetőmotor nem minden területre jó, és nem tudja soha kiszorítani a hagyományos motorokat.

A léptetőmotor alapvetően különbözik az egyszerű egyenáramú motoroktól. Ez a különbség abban áll, hogy a léptetőmotort digitális jellel kell vezérelni. Ennek a motortípusnak hatalmas előnye, hogy a jel hatására adott, előre meghatározott pozícióba fordul. Elsősorban olyan helyeken alkalmazzák e motorokat, ahol amúgy is digitális jelekkel dolgoznak, illetve ahol lényeges a pontos pozícióba állás. Tehát megtalálhatók a számítógép perifériákban (nyomtató, floppy, winchester, scanner, CD-rom), az ipari termelés sok területén (pl. EMCO CNC, hobby célra épített kis szerszámgépek), valamint a modellezésben. Egyéb területen is találkozhatunk vele, hisz például egy mutatós kvarcóra is léptetőmotorral működik... Lényeges előnye a léptetőmotornak a többi hasonló megoldáshoz képest, hogy az egyes lépések hibái nem adódnak össze (pozícióba állás pontossága általában 1-5%), és ezt mindenféle bonyolult visszacsatolás nélkül valósítja meg, jóval olcsóbban, mint a többi (pl. szervo) megoldás. Ez nem teljesen igaz, ugyanis kb. 300 W teljesítményig valóban olcsóbb, felette azonban már egyre gazdaságtalanabb.

Érdekes lehet, hogy egyes számítógép hűtőventillátorok is léptetőmotorok, annyi különbséggel, hogy ezek magukban foglalják a vezérlést is. Gyakorlatilag Brushless DC motorok.

Példa:

Léptetőmotorok az EMCO CNC-nél: A SZIE tulajdonában lévő EMCO típusú CNC gépek léptetőmotor hajtásúak. A készülék az elmozduláshoz szükséges lépéseket úgy határozza meg, hogy ismeri az egy teljes körülforduláshoz szükséges lépések számát, és a golyósorsó egy fordulat alatti menetemelkedését. Ebből a két adatból könnyen ki lehet számolni a kívánt pozíció eléréséhez szükséges lépések számát.



1. ábra: EMCO CNC gép

Pro/kontra[5.]

-Előnyei:

1. Nincs benne szénkefe, így tartósabb (gyakorlatilag a csapágyminősége határozza meg az élettartamot)
2. A pontos pozícióba álláshoz nem szükséges bonyolult visszacsatolás (tehát megfelelő vezérlés, és határértéken belüli terhelés esetén biztos, hogy a megfelelő pozícióba fordul)
3. Majdnem minden környezetben használható

-Hátrányai:

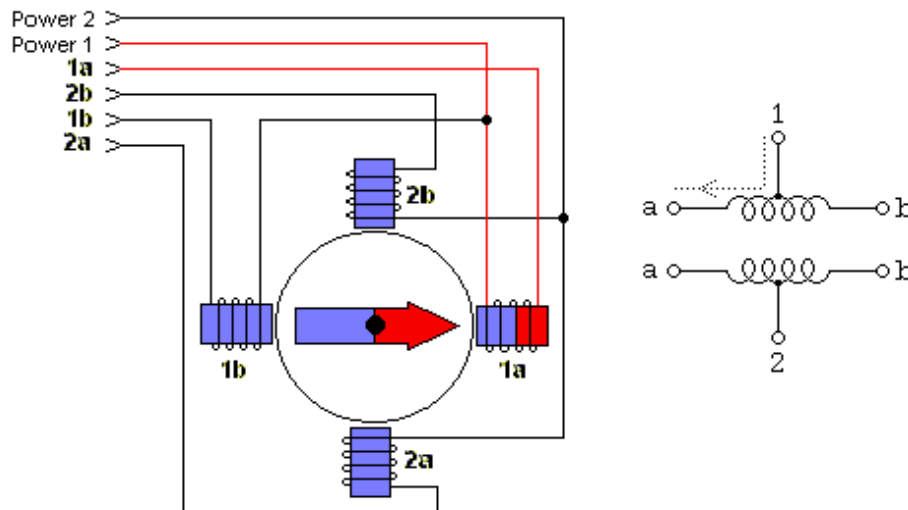
1. Alacsony fordulatszám (a maximális fordulatszám tipikusan 500-600 fordulat/perc)
2. Azonos teljesítmény mellett nagyobb méret és tömeg
3. Nagy pólusszám mellett bonyolultabb vezérlés.
4. Magas teljesítményű motorok drágábbak, mint az azonos teljesítményű szervomegoldások.
5. Vibráció és zaj
6. Visszacsatolás hiányában hajlamos az érzékelhetetlen pozícióvesztésre
7. Nehezen csillapítható pozíciólengés egy lépés végén

A forgórész kialakítása szerint a léptetőmotorok három típusát különböztetjük meg. [5.]

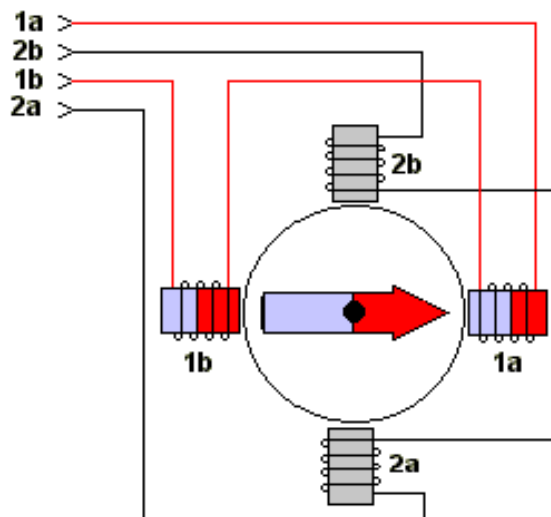
- Állandó mágnesű (PM) aktív forgórész
- Variable reluctance (VR – változó mágneses ellenállású) passzív forgórész
- Hibrid forgórész (a PM és a VR kombinációja)

Fajtái[6.]

A léptetőmotoroknak alapvetően két fajtája létezik, az unipoláris és a bipoláris. A bipoláris léptetőmotorok általában 4 kivezetése van, míg az unipolárisnak 5 vagy 6. (természetesen ettől eltérhetnek.)



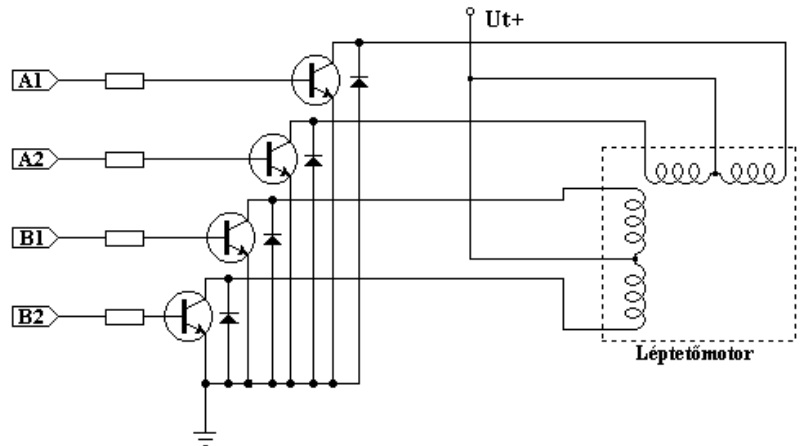
2. ábra: unipoláris léptetőmotor vázlat



3. ábra: Bipoláris léptetőmotor vázlat

Az Unipoláris motor:

Ez a motorfajta sokkal elterjedtebb, mint a bipoláris, mivel sokkal egyszerűbb a vezérlése. Az unipoláris motornak fázisonként egy, -középen is kivezetett- tekercse van. A tekercs darabjai külön vannak kapcsolva, annak megfelelően, hogy milyen mágneses mezőt kívánunk létrehozni, Ez jócskán megkönnyíti a dolgunkat, hisz a

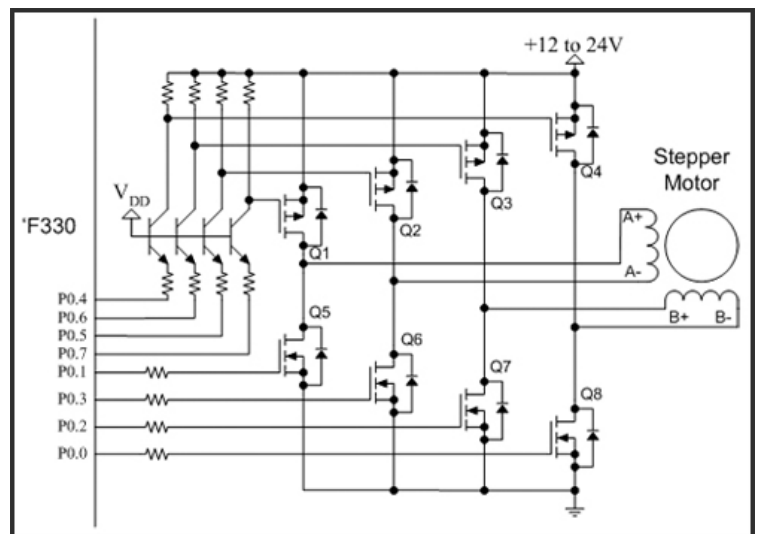


4. ábra: unipoláris motor vezérlése

mágneses pólusváltáshoz nem szükséges az áram irányának megfordítása.

A Bipoláris motor:

A bipoláris motorok egy tekercset tartalmaznak fázisonként. A mágneses pólusváltáshoz az áramirány megfordítása szükséges, ez megnöveli a vezérlés bonyolultságát, amely így sokszor H-híddal történik. (Manapság sok gyártó készít Bipoláris léptetőmotor meghajtó IC-t, így ez a probléma már egyre kisebb.) Fázisonként két



5. ábra: bipoláris motor vezérlése

kivezetés található, ezek közül egyik sem közös.

A bipoláris motor ugyanazon súly mellett nagyobb teljesítményre képes, mint egy unipoláris társa. Ennek oka abban keresendő, hogy míg egy Unipoláris motornál kétszer annyi tekercs van ugyanabban a tér-részben, de csak a felét használjuk. Ez Azt jelenti, hogy 50-és 70% teljesítménye egy bipolárishoz képest.

Egy léptetőmotor fontos paramétere[i6.]:

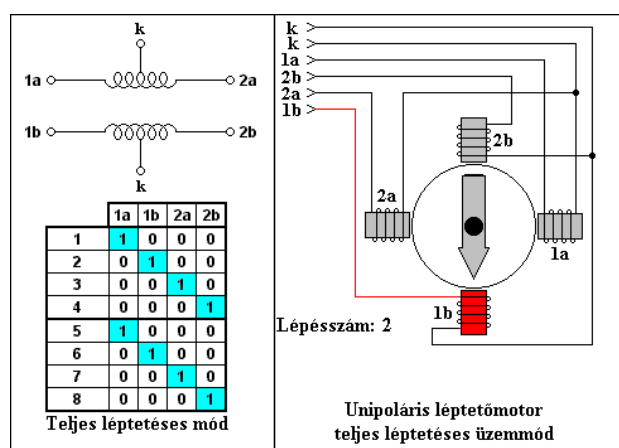
A léptetőmotorok lényeges tulajdonsága, hogy egy körülforduláshoz hány lépés szükséges. Ezt megadhatják fokban, ekkor az egy lépésre eső szögelfordulást adják meg. Illetve megadhatják az egy körülforduláshoz szükséges lépésszámot. Természetesen fontos paraméter a motor terhelhetősége, teljesítménye és nyomatéka is, ezen felül igen fontos tényező a tartónyomaték is, és az azt előállító áram nagysága is.. A kivezetések száma azért változhat, mert növelhetik a tekercsek számát.

Mivel a léptetőmotorok rövid szakaszos üzemre vannak méretezve. Amikor nem léptet, akkor egy alacsonyabb feszültséget kell rákapcsolni, ellenkező esetben akár le is éghet. (Az EMCO CNC-k motorja is jól fel tud forrósodni) Ez a tartófeszültség azért szükséges, hogy a motor ne tudjon kimozdulni az adott pozícióból. Egyébként a motort amikor magára hagyod (kikapcsolod a tápfeszültséget), akkor megpróbál a legközelebbi teljes lépés pozícióba fordulni. Tehát a féllépés pozíció nem stabil. Erre a program tervezésekor figyelni kell.

Vezérlési módok[i6.]

Egy kép többet mond ezer szónál mottóval mutatnám be a vezérlések módjait.

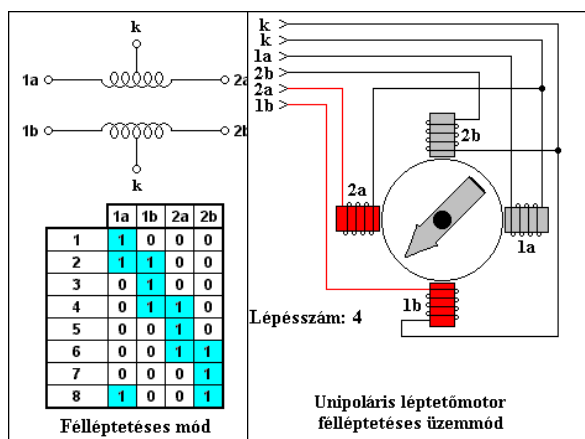
(a teljesség igény nélkül)



6. ábra: Unipoláris motor teljes léptetési üzemmódban

Teljes léptetési (egyfázisú) mód

A forgórész mindig egy tekercs felé mutat. Nincs átmenet. Természetesen a forgásirány megfordítható, ehhez csupán a vezérlés sorrendjén kell változtatni. (egyszerűen fordított sorrendben kapcsolgatjuk.)

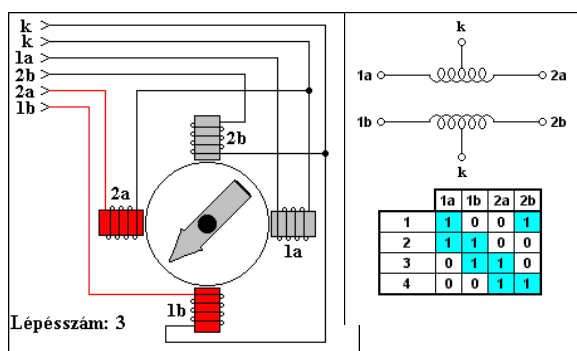


7. ábra: Unipoláris motor félléptetési üzemmódban

Félléptetési mód

Mint látható, az egyik tekercs kapcsolása után az adott tekercsre és a mellette lévőre egyszerre kapcsolunk áramot, majd csak a következőre. Ezzel a módszerrel az egy körülforduláshoz szükséges lépésszám megduplázódik. Így tehát pontosabb pozicionálásra képes ugyanaz a motor.

Teljes léptetése (kéttekercses) mód



Mint látható, mindig egyszerre két tekercsre adunk áramot. A motor majdnem kétszer olyan teljesítményű lesz, viszont sokkal jobban melegszik, így ez a kapcsolási mód csupán rövid szakaszos üzemre alkalmas. (8

8. ábra: Kéttekercs üzemmód unipoláris motornál (Antal, 2008)

Mikroléptetési mód

Ebben az esetben a motor névleges feszültségét szétsztyják a két szomszédos tekercs között. Minél több részre osztják, annál több lépésből áll két egész lépés közötti szögelfordulás.

Egy példa: Feltételezve, hogy a motor névleges feszültsége 5 Volt, a feszültséget 1 voltonként osztjuk szét. Tehát a példánknál maradva, a két tekercsre 1-4, majd 2-3, utána 3-2, végül 4-1 voltot adunk. Tehát ebben az esetben 4 lépésből teszi meg a két teljes lépés közötti távolságot. Valójában szinte tetszőleges mértékben feloszthatjuk így a két lépés közti elfordulást.

Áramerősség korlátozás

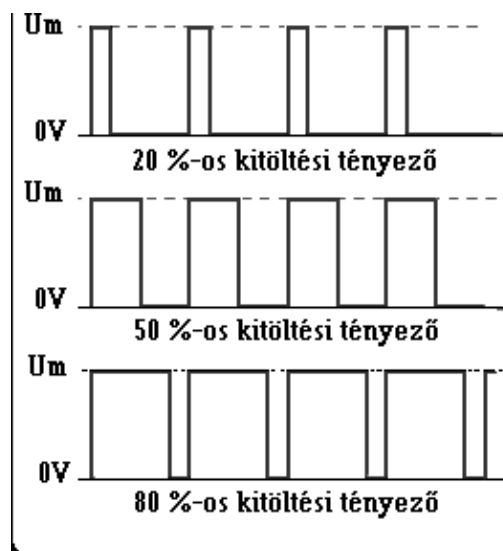
Mivel a léptetőmotorok tekercseinek Ohm-os ellenállása igen alacsony, ezért már alacsony feszültség rákapcsolásakor is igen nagy áram folyhat át a tekercseken. A tekercseken átfolyó áram hőt termel, ezért szükségszerűvé válik az átfolyó áramerősség korlátozása. A korlátozást megoldhatnánk egyetlen a tekercssel sorbakötött ellenállással, ez azonban nem egy ideális megoldás, hiszen ekkor a tekercseket csak kisebb árammal tudnánk gerjeszteni, melynek hatására az elérhető nyomaték nagymértékben lecsökkenne.

A legjobb módszer, ha a motor pozíció váltásakor az aktuális tekercset egy bizonyos ideig gerjesztjük, majd egy rövid idő eltelte után a gerjesztőáramot lekorlátozzuk, hogy a tekercs ne melegedjen túl. Ezt PWM, azaz impulzus-szélesség modulációs eljárással tudjuk megvalósítani legegyszerűbben és leghatékonyabban.

PWM

A PWM (Pulse Width Modulation) impulzus szélesség modulációs eljárás igen elterjedt a vezérléstechnikában.

Lényege, hogy mikrovezérlő vagy más elektronikai (jelen esetben 555-ös IC-vel megvalósított astabil multivibrátor) eszköz segítségével modulált jelet hozunk létre és a tekercset ezzel a jellel gerjesztjük, nem pedig egyenárammal. A PWM jellemzői az alapfrekvencia és a kitöltési tényező.



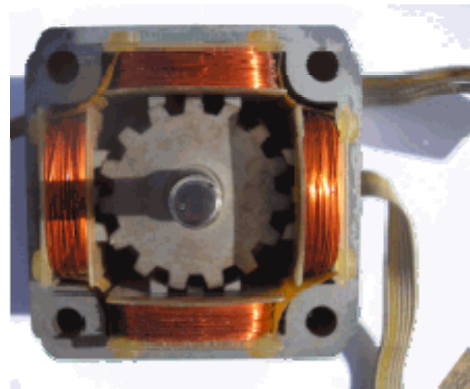
9. ábra: PWM kitöltési tényező alakulása

Az alapfrekvencia meghatározza, hogy a modulált jel hányszor változik másodpercenként. Bevolt szokás, hogy az alapfrekvenciát 20KHz felettire választják, mert ez a frekvencia az emberi fül számára már nem hallható.

A kitöltési tényező azt határozza meg, hogy egy perióduson belül a négyszögjel a teljes periódusidő hány százalékában vesz fel logikai 1-es értéket. Belátható, hogyha például az áramforrásom 12V-os, ezt a 12V-ot nagy frekvencián ugyanannyi ideig be-, majd ugyanannyi ideig kikapcsolva tartom, azaz 50%-os kitöltési tényezőt alkalmazok, a kimeneti négyszögjelet integrálva 6V-ot kapok. Az integrálást jelen esetben a motor maga végzi, hiszen a vezérlő frekvencia jóval nagyobb, mint a motor tehetetlenségéből adódó válaszidő. [7.]

A hibrid unipoláris motor Szerkezeti felépítése

A képen láthatjuk belülről a motort belülről. A forgó rész (rotor) inkább egy fogaskerékre hasonlít. Oldalt láthatók a tekercsek. Bár 4 tekercsnek tűnik, valójában ez 8 tekercs. Azért tesznek két tekercset egybe, hogy a tekercsek között elosszák a teljesítményt, (és ezzel a keletkező hőt), valamint a motor csapágait kímélik, hiszen nem egy oldalra próbálja meg elhúzni a forgó részt.



10. ábra: Léptetőmotor szerkezete

Amennyiben a tekercsek száma páros (legtöbbször az), akkor a forgó rész "fogai" páratlan számúak. És mindez fordítva is igaz. Egyébként a tekercsek száma (jelenleg 4) és a rotor "fogainak" számát összeszorozva (jelenleg 15) megkapjuk a motor egy körülfordulásához szükséges lépések számát (ez a motor tehát 60 lépésből tesz meg egy fordulatot).

[6.]

Amint azt a fentiekből láthattuk, és melyet a Magyar Elektronika c. folyóiratban megjelent Zsuffa Attila által írt cikk is alátámaszt, egyszerű vezérléstechnikai megoldásokra, ahol nem lényeges a nagy teljesítmény, és sebesség, de fontos az olcsó üzemeltethetőség, és magas élettartam, valamint a készülék ára, kiváló megoldást ajánl a léptetőmotoros hajtás. A különféle motorjellemzőket (5 Zsuffa, 2006.) egy táblázatban gyűjtötte össze, mely alább látható.

1. táblázat: különféle elektromos aktuátorok összehasonlítása

	AC indukciós motorok	DC- motorok (kefés – BDC)	DC- szervomotorok (kefe nélküli – BLDC)	AC- szervomotorok (kefe nélküli – BLAC)	Léptetőmotorok
Fordulatszám-tartomány	100 – 2500/min	0 – 5000/min	0 – 60 000/min	0 – 60 000/min	0 – 300/min
Teljesítmény-tartomány	100 W – 50 kW	0,1 W – 200 kW	0,1 W – 20 kW	0,1 W – 20 kW	0,1 W – 1 kW
Nyomaték-tartomány	0,02 – 300 Nm	0,01 – 1000 Nm	0,01 – 100 Nm	0,01 – 100 Nm	0,01 – 30 Nm
Dinamika	alacsony	közepes	kiváló	kiváló	közepes
Motor ár	alacsony	közepes	magas	magas	alacsony
A vezérlés bonyolultsága, ára	egyszerű, alacsony	közepesen bonyolult, közepes	közepesen bonyolult, közepes	bonyolult, magas	egyszerű, alacsony
Karbantartási igény	alacsony	magas	alacsony	alacsony	alacsony
Élettartam	magas	alacsony	magas	magas	magas

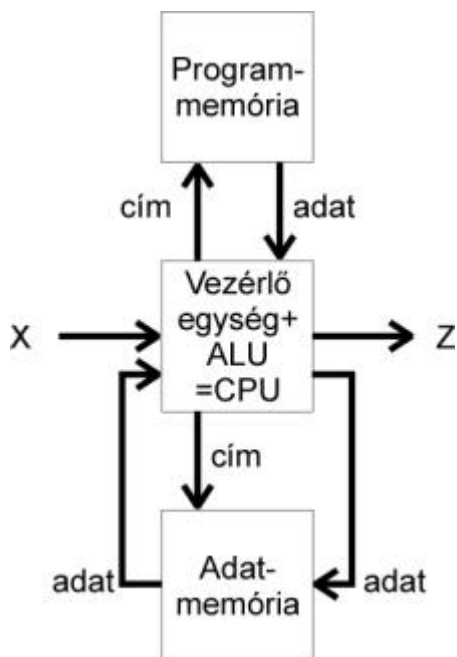
2.4. Mikrokontroller, mint a beágyazott rendszer alapja

„A mikrokontroller egyetlen lapkára integrált, általában vezérlési feladatokra optimalizált számítógép.” (10 Wikipedia) A szilícium lapkán megtalálható a mikrokontroller processzora, perifériák és a mikrovezérlő memóriája is. Régebbi típusoknál előfordul, hogy a memória nincs beleintegrálva a tokba, de a modern eszközökben ez szinte mindig megtalálható. A mikrokontrollerek beépített perifériái igen sokrétűek lehetnek, kezdve a hagyományos TTL vagy CMOS logikai szintű ki- és bemenetektől, az analóg-digitális, digitális-analóg konvertereken (ADC, DAC) és komparátorokon keresztül, egészen az analóg LCD kimenetek és kommunikációs portokig. (I2C, RS232, CAN, USB, Ethernet, stb.)

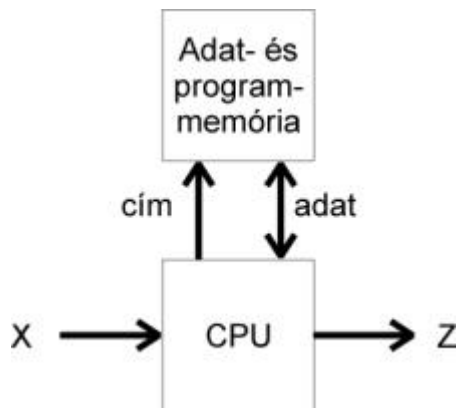
Általában a mikrokontrollerek eltérő perifériákkal vannak szerelve, így fontos a vásárlás előtt tájékozódni, a kiszemelt mikrokontroller perifériái után, (ezt leggyakrabban az adatlapok segítségével tehetjük meg) hogy megfelel-e az adott irányítási feladat megoldásához. Fontos tulajdonsága még a mikrokontrollereknek, hogy hány bites felbontással képes dolgozni, és hogy az ALU-ja képes-e kezelni a lebegőpontos számításokat, valamint, hogy mekkora utasításkészlet áll rendelkezésre az adott mikrovezérlőn. (8. Byte Craft Limited, 2002.)

Harvard-architektúra, a mikrokontrollerek szokásos architektúrája

A modern számítógépek, és mikrovezérlők architektúrája alapvetően különbözik egymástól, hisz, míg az asztali számítógépünk Neumann-architektúrát használ, addig a mikrokontrollerek a Harvard-architektúrán alapszanak. A két architektúra közti lényeges különbséget először is a memóriánál találjuk meg, hisz, míg a Neumann féle megoldásnál nincs külön adat és programmemória, addig ezt a Harvard-architektúrájú gépeknél külön találjuk. A másik feltűnő különbség, hogy míg a Neumann-architektúrájú szerkezetekben egyetlen busz található, melyet ütemezetten vagy adat transzferre vagy utasítás átvitelre használhatunk, addig a Harvard-architektúrájú készülékekben az utasítás és adatbuszok el vannak különítve, így egyidejűleg, azaz párhuzamosan folyhat a kommunikáció. Ez azt jelenti, hogy, bár bonyolultabb a mikrokontrollerek felépítése, mint egy általános célú processzoré, mégis gyorsabb műveletvégzést tesz lehetővé köszönhetően a párhuzamos adat, és utasításkommunikációnak. (8 Byte Craft Limited, 2002)



11. ábra: Harvard-architektúra



12. ábra: Neumann-architektúra

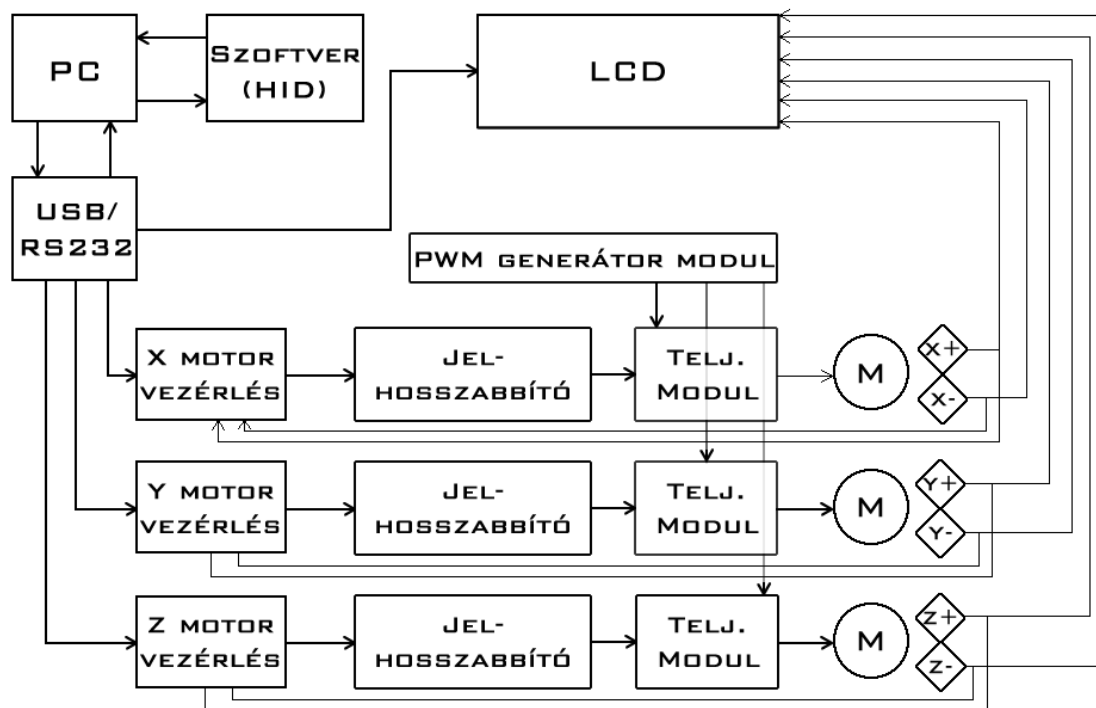
4.2. Mikrokontrollerek alkalmazása

Mivel a mikrokontrollerek viszonylag olcsók (ugyanazon vezérléstechnikai feladat megvalósítása analóg módon jóval drágább, vagy lehetetlen lenne) igen alacsony fogyasztásúak, és változatos felszereltséggel, változatos tokozásban kaphatók, így hamar elterjedt az alkalmazásuk. A legnagyobb gyártók a teljesség igénye nélkül Atmel, TexasInstruments, Microchip (PIC), Intel, Silicon Labs.

A mikrokontrollereket igen széles körben használja az ipar. Gyakorlatilag ma már egy kenyérpirítót sem lehet venni, hogy abban ne legyen egy mikrovezérlő. Minden gép, ami körülöttünk van, és elektronikus vezérlésű, biztos tartalmaz valamilyen mikrokontrollert. Egy modern autóban több száz ilyen kis számítógépet találhatunk.

3. Szerszámgépek utólagos automatizálásának tervezése

3.1. Rendszerterv



113. ábra: Rendszerdiagram

Látható, hogy a készüléket –terveim szerint- leginkább a modularitás jellemzi. A rendszerdiagramban szürkével jelölt modulokat finansziális okokból nem állt módomban elkészíteni, de a folyamatábrán jól látható módon ezek az első másolatai, így a tesztelést el tudtam végezni a rendszeren. Mivel az USB/RS232 modul egyszerre 4 motor vezérlését képes ellátni, az időzítéseket mikrokontroller szinten végezve, így belátható, hogy a vezérlés megfelelő számú motorvezérlő, jelhosszabbító és teljesítménymodulok felhasználásával akár 4 tengely egyidejű mozgását képes szavatolni.

Moduláris felépítés előnyei

A moduláris felépítésnek köszönhetően a készülék könnyen, egyszerűen testreszabható. Ha 1 Motoros rendszerre van szükség, elég egy motorvezérlő modult egy teljesítménymodullal összekötni, így is működőképes. Ha komplexebb feladatokra szánjuk a készüléket, mondjuk 3, esetleg 4 tengelyes CNC megvalósításához (pl., ha szükségünk van forgatóasztalra) akkor a megfelelő számú motorvezérlő, és teljesítménymodult építjük be. Amennyiben szükség van egyszerű megjelenítőre az áramkörök változtatása nélkül behelyezhetünk egy kijelző modult, továbbá ha a helyzet úgy kívánja meg, könnyűszerrel hozzáilleszthetünk további audiovizuális eszközt, pl. hibajelző szirénát, állapotjelző lámpákkal. Amennyiben nagy nyomatékot kell elérni, akkor ajánlatos a motor tartóáramát passzív szakaszban korlátozni, hogy az ne melegedjen túl. Erre használható együttesen a Jelhosszabbító és a PWM modul. A rendszert alapvetően LPT porton, vagy mikroszámítógép GPIO-in keresztül terveztem vezérelni, de felmerülhet az igény a manapság a számítógépeken egyre ritkábban fellelhető LPT port kiváltására RS232, ill. a minden számítógépen fellelhető USB port használatára. Így az alapvetően párhuzamos vezérlésre tervezett áramkör az USB/RS232 illesztő modul segítségével bármikor könnyűszerrel kompatibilissé tehető mondjuk egy notebook-hoz, ami nagyban megkönnyíti a felhasználást azon mindazok részére, akik nem mikroszámítógéppel, ill. kompatibilis LPT porttal rendelkező géppel szeretnék használni a készüléket.

Felhasznált modulok

- Motorvezérlő modul (X,Y,Z)
- Teljesítmény modul
- Kijelző modul (LCD)
- USB/RS232 illesztő modul
- Jelhosszabbító modul
- PWM modul

3.2. Hardver

A motorvezérlő modul megtervezése

Feladata:

Két diszkrét jel (STEP, DIR) hatására a megfelelő irányban léptetni a motort. Kezeli a véghelyzeteket (véghelyzetre való futáskor nem engedi abban az irányban tovább léptetni a motort), és a program indításakor manuális beavatkozás hatására végrehajtja a HOME (indításkor kötelező végrehajtani, a motor negatív véghelyzetre áll, majd visszább jön, így kb 4 lépéssel leáll a véghelyzetről. Innentől használható.) rutint, amely arra irányul, hogy a motor felvegye a tényleges nullpontot, és szinkronba álljon.

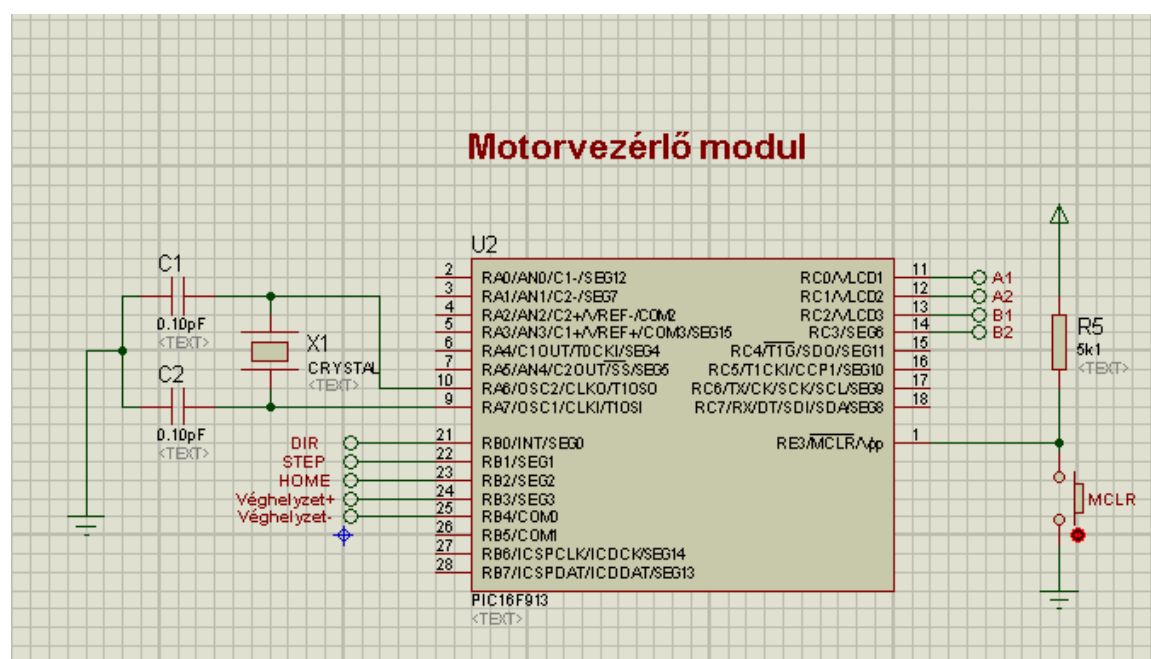
Főbb tulajdonságai:

Mikrokontroller: PIC16F913 PDIP28

Órajel: HS, 8MHz Kristály oszcillátor

Bemenetek: HOME, Dir, Step, +végállás, -végállás

Kimenetek: A1, A2, B1, B2



12. ábra: Motorvezérlő modul kapcsolása

A teljesítmény modul felépítése

Feladata:

A motorvezérlő modul által kiadott utasítások elvégzése, motoráram korlátozása a PWM és jelhosszabbító modul értékeitől függően.

Főbb tulajdonságai:

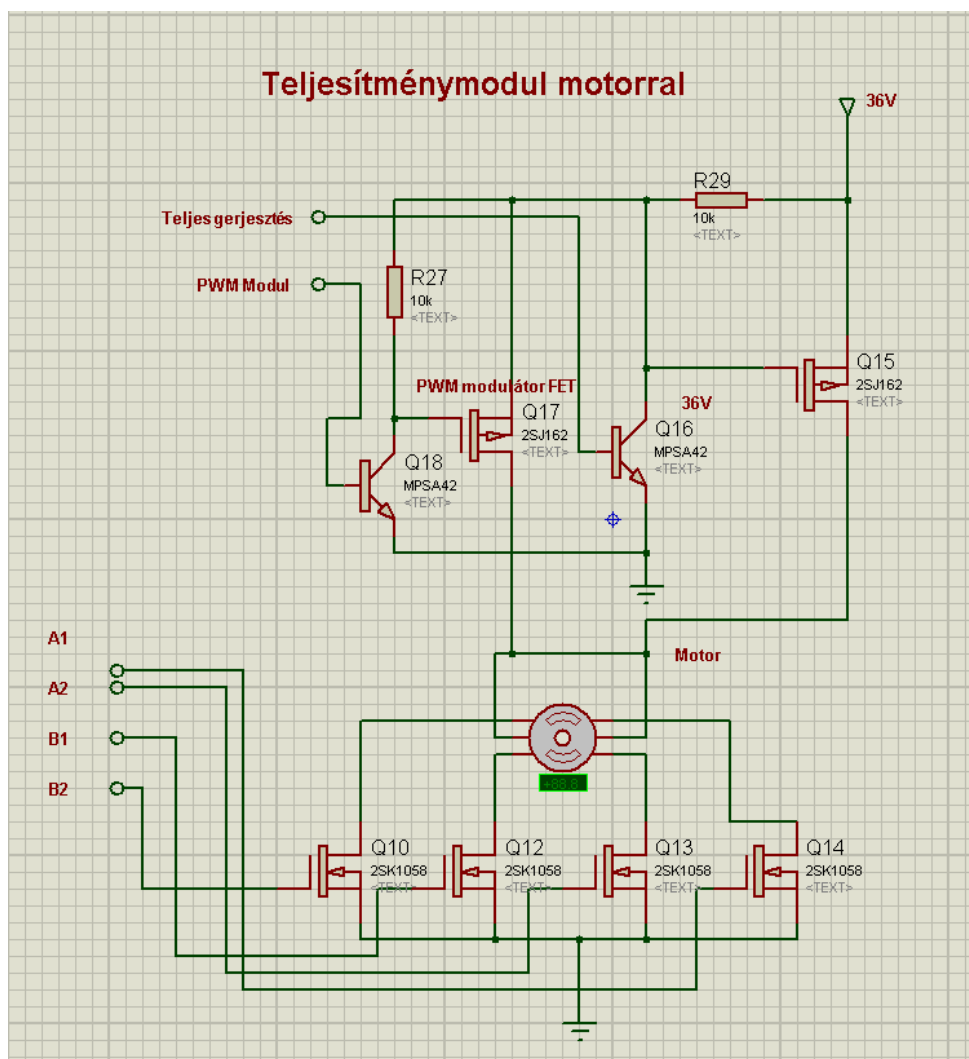
N-csatornás FET: IRF540N

P-Csatornás FET működtető tranzisztora: MPSA42

P-Csatornás FET:

Bemenetek: A1, A2, B1, B2, PWM, Teljesítmény

Kimenetek: A1, A2, B1, B2



13. ábra: Teljesítménymodul kapcsolása

A kijelző modul elkészítése

Opcionális kiegészítő.

Feladata:

Figyelmeztetések (véghelyzetre futás, HOME-olás, készenlétben várakozás), elért pozíció kiírása.

Főbb tulajdonságai:

Mikrokontroller: PIC16F917

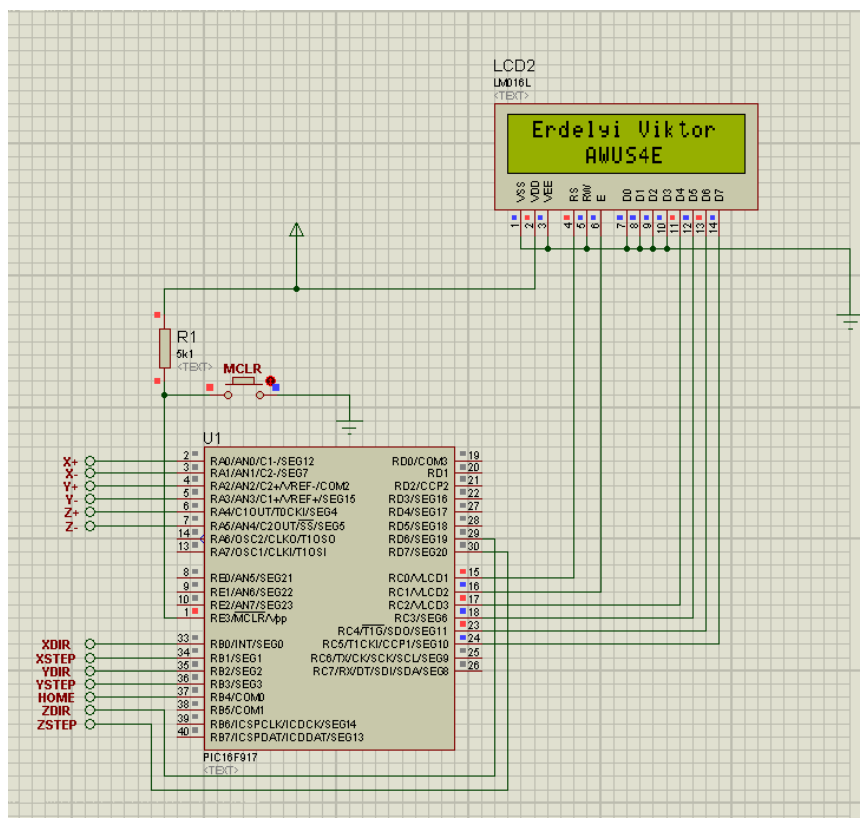
Órajel: HS, 8MHz Kristály oszcillátor

LCD modul: LM016L 2x16 karakteres háttérvilágítással ellátott Alfa numerikus LCD

Opc: LM044L 20x4 karakteres, háttérvilágítással ellátott Alfa numerikus LCD (2-nél több tengelyű rendszerekhez)

Bemenetek: Home, XDir, XStep, YDir, YStep, ZDir, ZStep, X+, X-, Y+, Y-, Z+, Z-

Kimenetek: LCD/RS, LCD/E, LCD/D4/, LCD/D5, LCD/D6, LCD/D7



16. ábra: kijelzőmodul kapcsolása

USB/RS232 illesztő modul tervezése

Opcionális kiegészítő.

Feladata:

RS232 (ill. megfelelő adapterrel USB) kommunikációra képessé tenni a motorvezérlést.

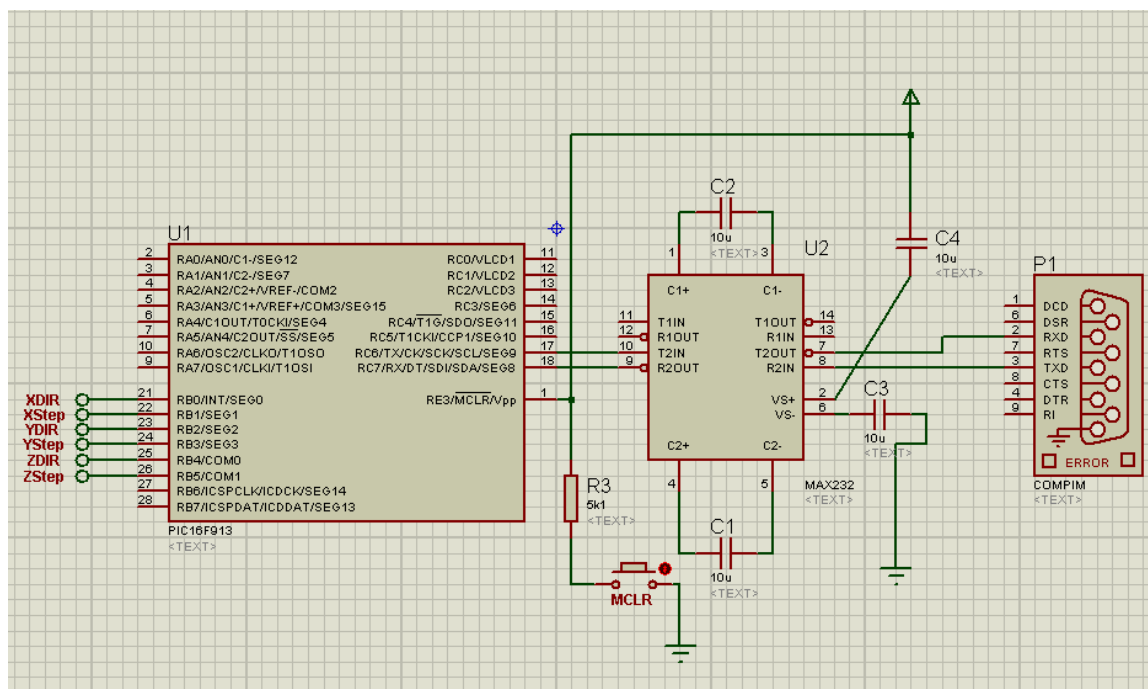
Főbb tulajdonságai:

Mikrokontroller: PIC16F913

Órajel: HS, 8MHz Kristály oszcillátor

Jelillesztő IC: MAX232 RS232 jelszintek(-15V - +15V) TTL szintűvé konvertálása(0-5V)

Csatlakozó: Szabványos soros aljzat



17. ábra: illesztőmodul kapcsolása

Az USB-RS232 átalakító egy LogiLink gyártmányú AU0002B kódjelű eszköz.

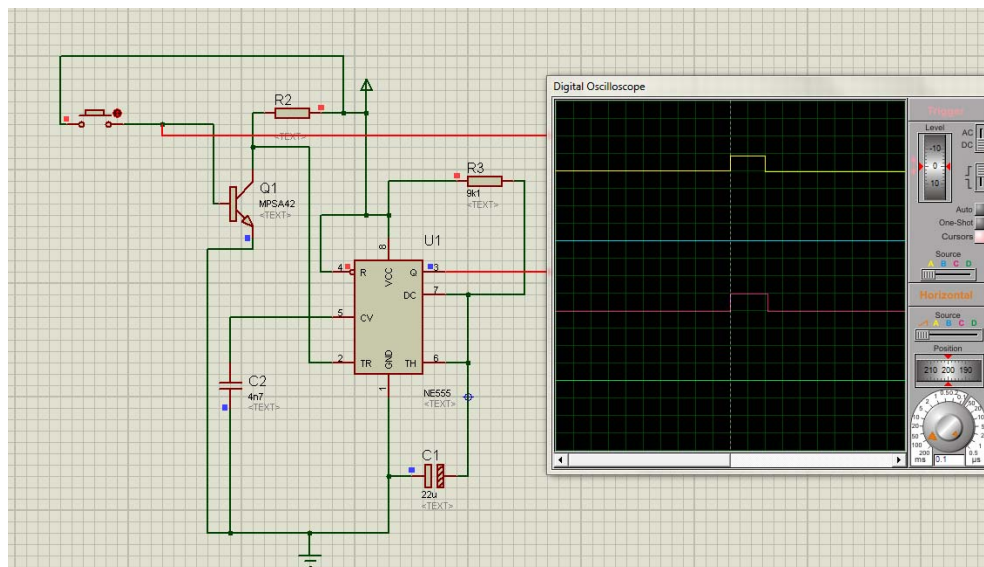
A jelhosszabbító modul működése

Feladata:

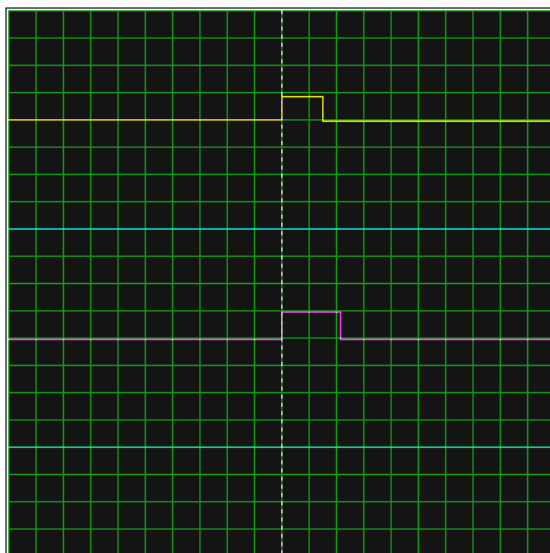
A vezérlőjel megkapása után, ha nem érkezik újabb vezérlőjel, 40ms után a motor teljes gerjesztését megszünteti, így PWM-el szabályozott tartóáram működteti a tekercset, ami ezáltal kevésbé melegszik.

Főbb tulajdonságai:

IC: LM555



18. ábra: Jelhosszabbítómodul kapcsolása



19. ábra: jelhosszabbítómodul működése

A jelhosszabbítás értéke a következő módon számítható ki:

$$\Delta t = 1,1 \cdot C1 \cdot R3 - t0 [\text{ms}]$$

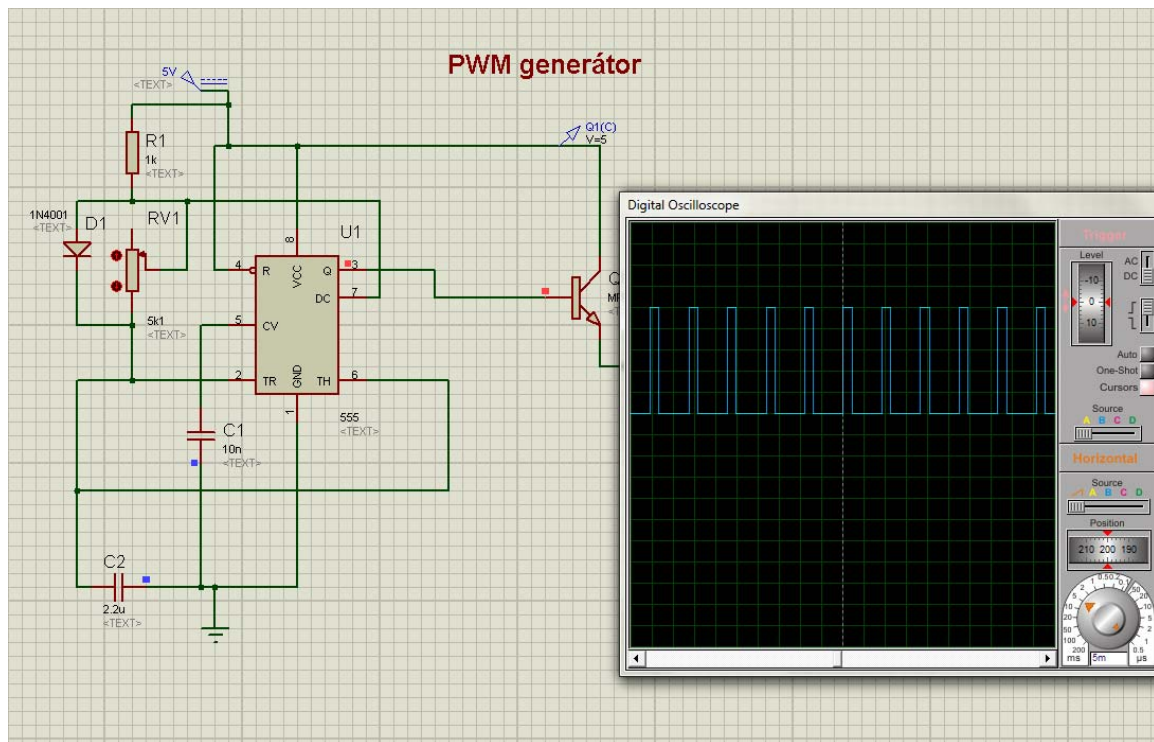
A PWM modul fontossága

Feladata:

A Teljesítménykör PWM modulátor FET-jét vezérelni annak érdekében, hogy a passzív szakaszban a motor tartóáramát 25%-os kitöltési tényezővel a hetedére csökkentse annak érdekében, hogy a motor ne melegedjen túl.

Főbb tulajdonságai:

IC: LM555



20. ábra: PWM modul kapcsolása és működése

A fentiekből kitűnik, hogy nem egy intelligens vezérlésről van szó, hiszen a tényleges feldolgozást a számítógép végzi, bár, a motorvezérlő kör tartalmazza a home pozíció felvételét, és a véghelyzetre futások kezelését.

„[...] léptetőmotor-vezérlő áramkör[...] csupán annyit tett, hogy a c/ock, a step és a dir (cw/-gcw) jelből a léptetőmotorhoz illesztett szintű és terhelhetőségű Gray-kódú jelsorozatot állított elő, vagyis a pozíció nyilvántartását és a lépésszámnak és - iránynak megfelelő impulzussorozatot előállítását egy magasabb szinttől (tipikusan egy PC-től, annak párhuzamos portján keresztül) várta. Ám, ahogy a PC egyre jobban a virtuális világ felé fordul és saját erőforrásait is egyre inkább elvirtualizálja a felhasználó előtt, úgy lesz egyre reménytelenebb feladat az átlagember számára egy vezérlés ennyire alacsony (időzítést és lehetőleg közvetlen portkezelést igénylő) szintjének PC-s megvalósítása. Ugyanakkor a piac tele van különböző képességű és árfekvésű mikrokontrollerekkel. Logikus tehát az az elképzelés, hogy a PC foglalkozzon azzal, amire újabban való, az alacsony szintű időzített vezérlési feladatokat pedig adja ki egy mikrokontrollernek. ”

(9. König - Pálincás, 2000.)

Pálincás úr meglátását hajlamos vagyok elfogadni, de jelen megoldásnál számítógép oldalon meg van oldva a feladat, mivel a motorok azonos ideig vannak teljes gerjesztésen, így nem a motor sebességek egzakt beállítását végeztetem a számítógéppel (ez igényelné leginkább a pontos időzítést), hanem párhuzamosan több motornak küldök vezérlőutasítást, melynek eredményeként a pályát mint koordináták sorozatát adom meg a vezérlőegységnek. Így lehetővé válik, hogy a léptetőmotor felbontásától függő simaságú pályagörbét írjon le az adott szerszám.

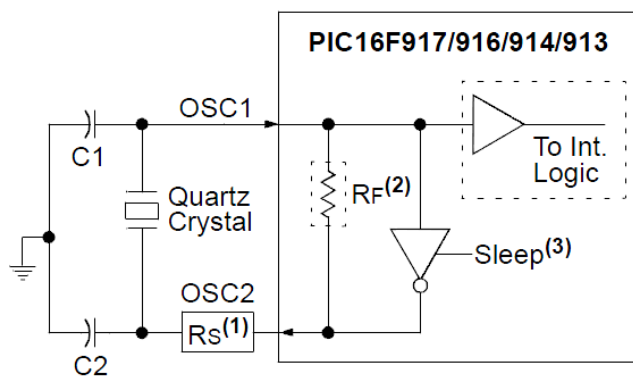
3.3. Felhasznált villamos elemek

Mikrokontrollerek

A PIC 8 bites palettájáról két mikrovezérlőt is használok, melyek szoros rokonságban állnak egymással, a különbség csak a GPIO számban látszik, és persze, hogy ennek megfelelően, hogy bizonyos portok hiányoznak a kisebbik készülékről. Mindkét fajta mikrokontroller a PIC16F9xx szériából való, az egyik a PIC16F913-as, és ennek a nagytestvére a PIC16F917-es. Ezek a készülékek alapvetően széles feszültségtartományban képesek üzemelni, flash-alapú memóriájuk van (erre utal az F betű a kódban), tartalmaznak LCD vezérlőt, és fel vannak szerelve a Microchip nanoWatt technológiájával.

Oscillátor

Mindegyik mikrokontrollert egységesen 8MHz-s kristályoszillátorral működtetem. A C1, és C2 értéke 22pF, Rs értéke 0Ω. A mikrokontroller saját, belső órajel-generátora 31 KHz-től 8MHz-ig képes órajelet generálni, így, elvileg megfelelne a célnak, de mivel a belső oszcillátor hőmérsékletfüggése jelentősen magasabb, mint a külső kristály-oszcillátornak, így ezt a megoldást választottam.



27. ábra: kristályrezonátor bekötési ábrája

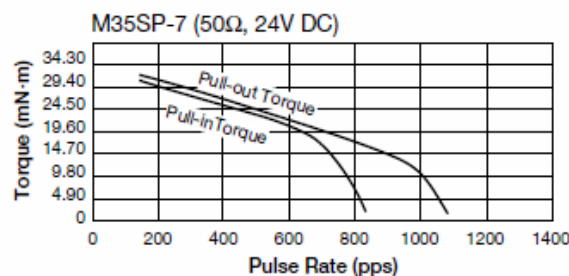
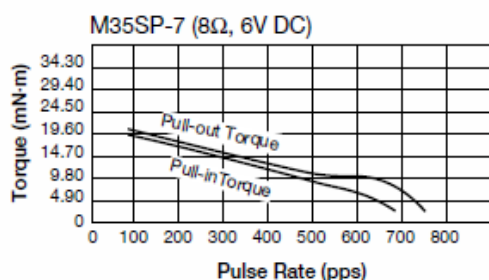
Motor

A prototípuskészülék motorja egy Mitsumi gyártmányú M35SP-7 típusú unipoláris 4 fázisú léptetőmotor. Ez a motor igen elterjedt különféle irodatechnikai gépekben, pl szkennerek, nyomtatók, mivel, a 35mm-es külső átmérőjű motorok közül ennek a legvékonyabb a motorteste, és viszonylag jó karakterisztikákkal, és viszonylag nagy teljesítménnyel rendelkezik.

Főbb tulajdonságai:

2. táblázat : M35SP-7 típusú léptetőmotor főbb tulajdonságai

M35SP-7		
Megnevezés	Érték	
Névlegesfeszültség	6V	24V
Névleges áramerősség	807mA max.	807mA max.
fázisok száma	4	4
lépés szög	7,5°/lépés	7,5°/lépés
tartónyomaték	29.4mN·m	29.4mN·m
maximális nyomaték	17.6mN·m/200pps	17.6mN·m/200pps
Maximális vezérlőfrekvencia	770pps	1050pps



28. ábra: M35SP-7 típusú unipoláris léptetőmotor karakterisztikái

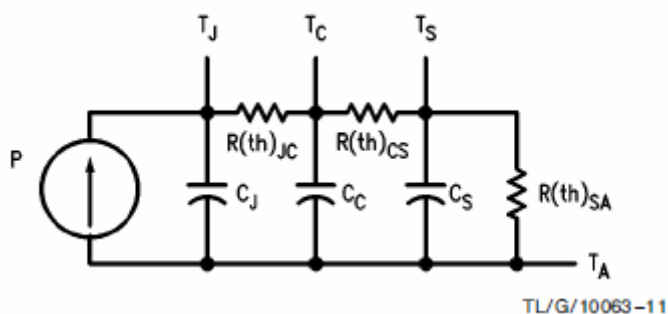
Kijelző

LCD modul: EW162G0GLY, mely egy traszflektív, szürke karakterszínű, 2x16 karakteres, sárga LED-es háttérvilágítással ellátott Alfánumerikus LCD, amit szabványosan 8 vagy 4 biten lehet vezérelni, és a hagyományos Hitachi vezérlőelektronikával rendelkezik.

Táp

Mivel az áramkör TTL logikájú alkatrészekből építkezik, ezért fontos egy stabil 5V-os áramforrás. Ezt egyszerűen egy L7805CV kódjelű pozitív feszültségregulátorral oldottam meg. Az IC-ben benne foglaltatik a túlmelegedés-védelem, a rövidzár-védelem, és az átmeneti túláram-védelem. Bemeneti oldalon 5V-35V feszültségű egyenáramot áteresztve, a kimeneti oldalon stabil 5V-ot kapunk a terheléstől függetlenül (bizonyos határokon belül). Mivel analóg feszültség szabályzóról van szó, és viszonylag nagy a feszültségesés rajta, így fontos foglalkozni a disszipációval is. Ha feltételezzük, hogy 5V-on az áramkörökön 500mA folyik, figyelembe vesszük, hogy az alkatrészen a feszültségesés 20V-os tápfeszültség mellett (notebooktöltő) 7V, akkor könnyen kiszámolható, hogy az IC disszipációja hozzávetőlegesen $7V \cdot 0,5A = 3,5W$. Ha figyelembe vesszük az alábbi képlettel kiszámított hőmérsékletet $R_{th} = dT/P$ (adatlap alapján: $R_{th} = 54 [^{\circ}C/W]$) $dT = R_{th} \cdot P = 54 \cdot 3 = 191^{\circ}C$ ami egy 20 fokos helyiségben $211^{\circ}C$ hőmérsékletet jelent, nyilvánvalóvá válik, hogy ezen hő elvezetéséhez fontos hűtőbordát alkalmazni. Ha hozzáteszünk

egy HS-135-25 típusú TO220 tokozáshoz való hűtőbordát, melynek termikus rezisztenciája adatlap szerint $12,6^{\circ}C/W$, akkor a hőmérséklet azonos



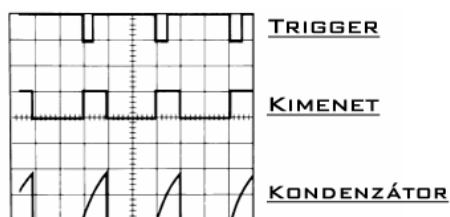
29. ábra: tranziens melegedési modell

körülmények között csak $56^{\circ}C$ lesz. Természetesen ez

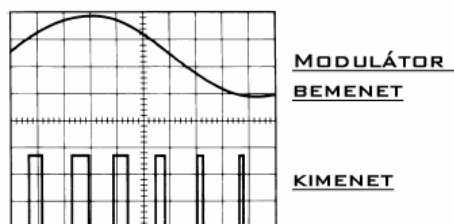
nem a külső tapintóval, hanem a készülék szilíciumlapján mérhető hőmérséklet. (10 Locher, 1988.)

Időzítő IC

Az időzítő IC-t két modulban is használok, nevezetesen a jelhosszabbító-, és a PWM modulban. Az IC egy LM555-ös IC mely egy teljesen átlagos, nagyon széles körben elterjedt analóg időzítő áramkör. Az 555-ös IC-t mindkét esetben monostabil kapcsolásban használok. (részletek lsd. Moduláris felépítés) A probléma az 555-ös megoldással, hogy szabadonfutó frekvenciával üzemel, azaz a PWM felhasználásnál nem lehetséges potméterrel csak a kitöltési tényezőt módosítani, a frekvencia is szükségszerűen el fog tolódni, és sajnos jelhosszabbításra sem alkalmas minden határon túl, hisz ha a trigger jel túl sokáig fennáll, a kondenzátor feltöltődik, így a trigger jel megszűnésekor a kimeneten azonnal alacsony szint jelenik meg. Szerencsére léptetőmotor vezérlésnél ez a probléma nem áll fenn, hisz a vezérlőjel kb 5ms hosszan tart, így az alkalmazott kapacitás, és ellenállás értékek mellett, kiválóan alkalmazható jelhosszabbításra. (11 NationalSemiconductor 2006.)

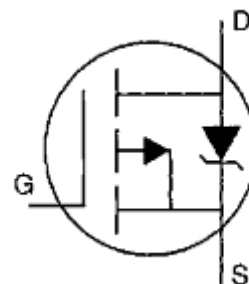


30. ábra: jelhosszabbító kapcsolás diagramjai



31. ábra: PWM kapcsolás diagramja

A P csatornás FETek működtetéséhez szükséges lehúzó tranzisztorokat alkalmazni, mivel a P csatornás FETek fordítva működnek, mint az N csatornás társaik. A lényegi különbség a működésükben, hogy míg az N csatornás készülékek a földhöz képesti pozitív feszültséggel kapcsolhatók, addig a P csatornás



34. ábra: N csatornás FET

megoldások a tápoldalhoz viszonyított negatív feszültséggel működtethetők. A P csatornás FETek alkalmazása nem túl

szerencsés, mivel a gyártástechnológiából adódóan a P csatornás FETek sajnos lassabbak, és kisebb teljesítményűek, mint N csatornás társaik, de ennél az alkalmazásnál szerencsére ez nem számottevő probléma.

Az alábbi közelítő számítással meghatározható a FETek ohmos disszipációja az alkalmazás körülményitől függő módon. (10 Locher, 1988.)

$$P_D = \frac{V_{DD}^2}{R} \left[\frac{t_{(on)} + t_{(off)}}{6} + R_{DS(on)} \cdot T \right] \cdot f_s$$

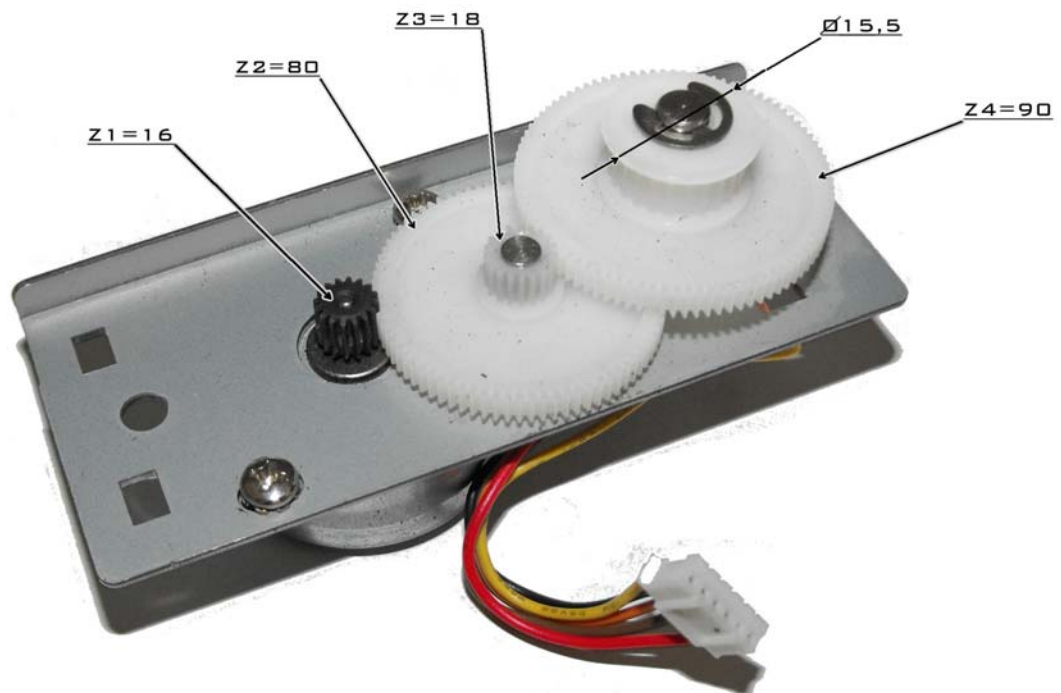
Ez az érték az adatlapon található, és az alkalmazásfüggő adatok alapján:

$P_D = 0,352013 \text{ W}$, ami jelen esetben elhanyagolható.

A biztonság kedvéért kiszámoltam az eredő hőmérsékleteket is, szintén közelítő számítással: $dT = R_{th} \cdot P = 62 \cdot 0,35 = 21,7^\circ\text{C}$, ami egy 30°C -os környezetben $51,7^\circ\text{C}$ -os abszolút hőmérsékletet jelent. Ez nem, sok, de a biztonság kedvéért a FET-eken is elhelyeztem, igaz kisebb hűtőbordákat, melyek alkalmazásával a hőmérséklet számításaim szerint 37°C -ra esik le.

Prototípus mechanizmus számításai

Bemutató készüléknél:



35. ábra: prototípus készülék motorja áttétellel

Áttételek számítása:

Motor paraméterek:

Angle/step=7,5°

Step/360°=48

Áttételek:

$i_1 = Z_2/Z_1 = 5$

$i_2 = Z_4/Z_3 = 5$

$\Sigma i = i_1 * i_2 = 25$

$K = 15,5 * \pi = 48,67 \text{ mm}$

Áttét után egy körülforduláshoz szükséges lépések száma: $48 * 25 = 1200$

Egy lépés hatására bekövetkező lineáris elmozdulás: $48,67/1200 = 0,04 \text{ mm}$

3.4. A működtető szoftver tervezése

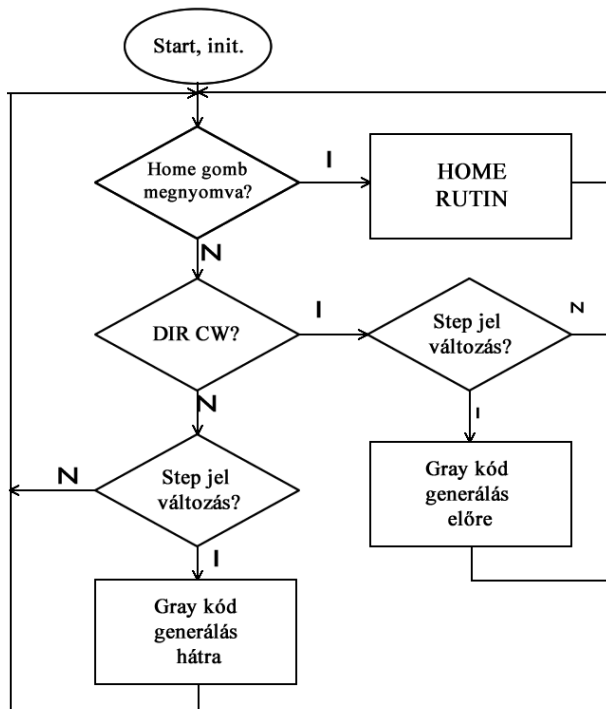
A teljes programkódok a mellékletekben megtalálhatók, ezen alfejezetben csak nagy vonalakban ismertetem a működésük.

Mikrokontrollerek programkódjai

A mikrokontrollerek programjait a Mikroelektronika által forgalmazott MicroC Pro for PIC nevű program ingyenes demoverziójával készítettem, a program nevéből is láthatóan C nyelven. A magas szintű programozási nyelv nagy előnye, hogy nem kell minden mikrokontroller saját assembly utasításait, regisztereit megtanulni, mivel a compiler, ha beállítjuk a céleszköz típusát, automatikusan annak az eszköznek érthető módon fordítja le a programot. Hátránya ellenben, hogy a compiler követhet el hibát, így nem használhatók az ilyen jellegű programok olyan készülékek programozására, melyen emberi életek múlhatnak, illetve súlyos sérüléseket lennének képesek okozni.

Motorvezérlő IC programkódja

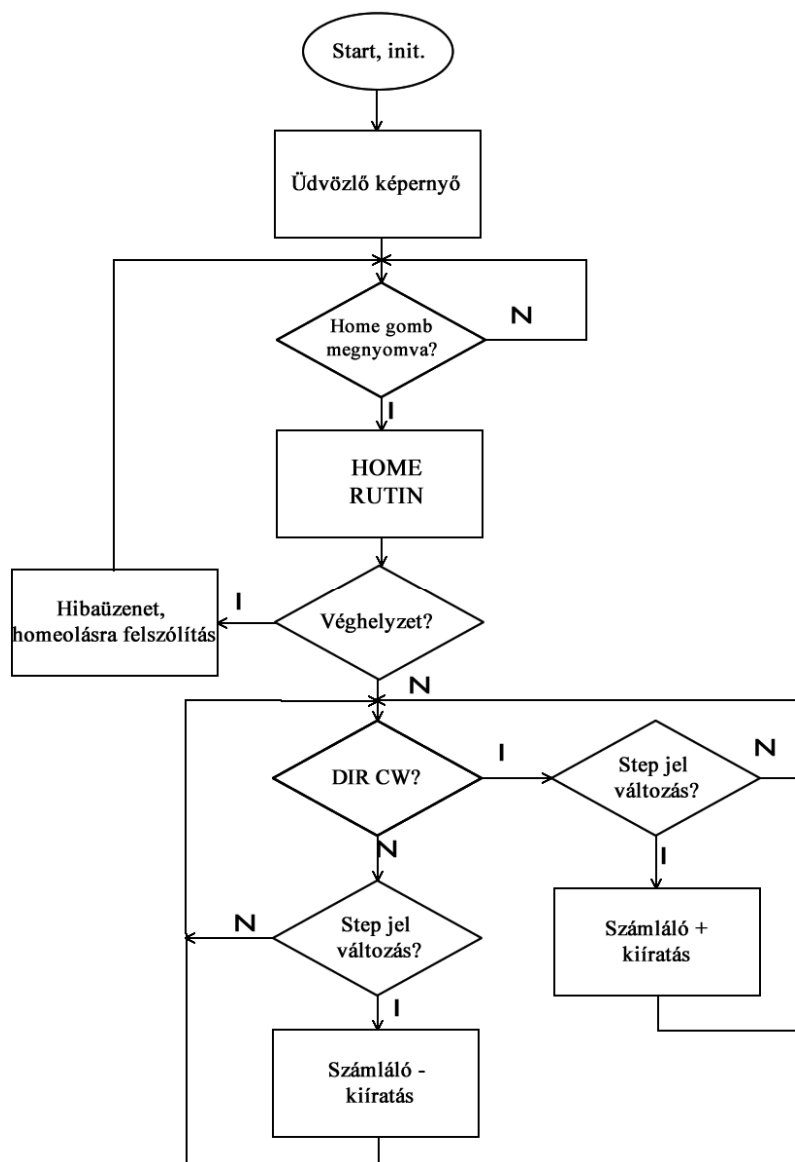
A Motorvezérlő mikrokontroller egy motort vezérel két jel (DIR, Step) alapján, tartalmazza a véghelyzetkezelést, és a home rutint.



14. ábra. A motorvezérlő program folyamatábrája

Kijelző modul programkódja

A kijelző programja meglehetősen hosszú, és sajnos a magja is elég bonyolult, így csak a 4. számú mellékletben közlöm. A MicroC saját prototípus függvényeit használva írtam meg ennek az egységnek a programját, viszont, mivel a mikrokontroller nem képes lebegőpontos számítások hardveres elvégzéséhez és a programozott lebegőpontos számítások rendre hibás működésre vezettek, így egy trükkel oldottam meg a kijelzésnél a törtek megjelenítését, miközben a mikrokontroller egészeket számol. A módszer lényege, hogy a számlálóban lévő int típusú számot egy karakterlánccá bontom fel, azt karakterenként tömbben tárolom, és gyakorlatilag az utolsó két számjegy elé kiíratáskor kirakatok egy tizedespontot a programmal, majd a fennmaradó karaktereket a pont előtt megjelenítem.



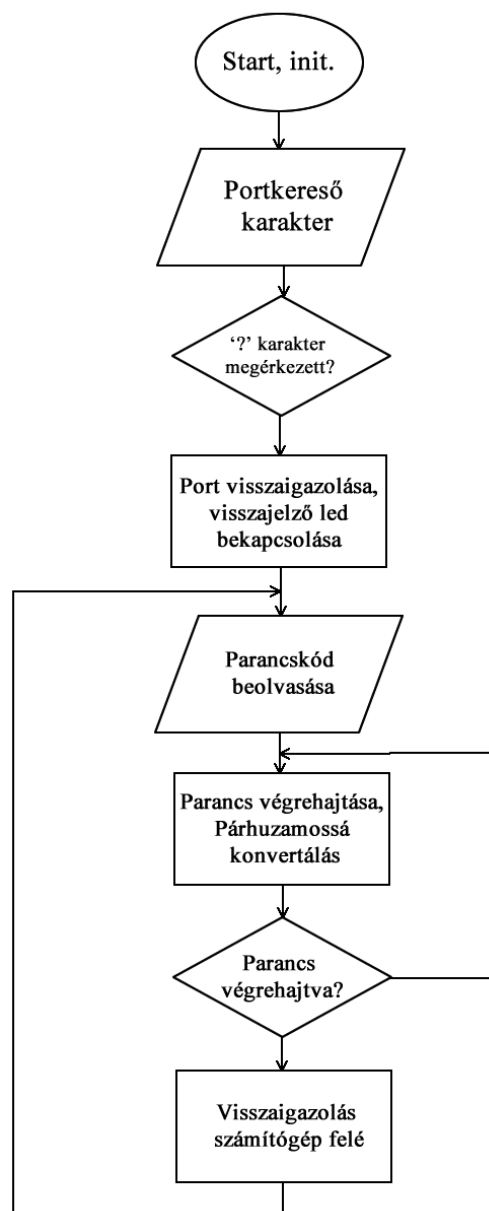
15. ábra. A kijelzőmodul programjának folyamatábrája

Illesztő modul programkódja

A kijelző programjához hasonlóan ezt is csak mellékletben közlöm (5. számú melléklet).

A működés:

RS232-porton érkező \$ karakter esetén a program elkezd a beérkező karaktereket (1 vagy 0) tömbben tárolni. A második \$ karakter beérkezésekor lezárja a tömböt, majd a tömb elemeit feldolgozva a motorvezérlő bitek állapotát frissíti, majd a számítógép felé elküld egy nyugtázó karaktert (k) mellyel a parancs végrehajtásának befejezését jelzi, így a számítógép küldheti a következő parancsot. A program tartalmazza továbbá az automatikus eszközkeresést, mely a következőképp működik: A számítógép az összes COM portra elküld egy kérdőjelet, (?) ezután várakozásba kezd. A mikrokontroller figyeli a saját portját, és ha „?” karakter érkezik rajta, visszaküld egy felkiáltójelet (!), ezzel egyidőben felkapcsol egy –a kommunikáció létrejöttét jelző- LED-et. A számítógép továbbra is figyeli a COM portokat, és amelyikről a válasz karakter „!” érkezik, azt a portot állítja be alapértelmezett kommunikációs csatornának. A készülék leválasztása hasonló egyszerűséggel történik, amennyiben a számítógépes programban elindítjuk a leválasztási procedúrát, a számítógép soros porton kiküld egy (e) karaktert, minek hatására a mikrokontroller kikapcsolja a kommunikációt jelző LED-et, és ismét várni kezdi a csatlakozásra felszólító „?” karaktert.

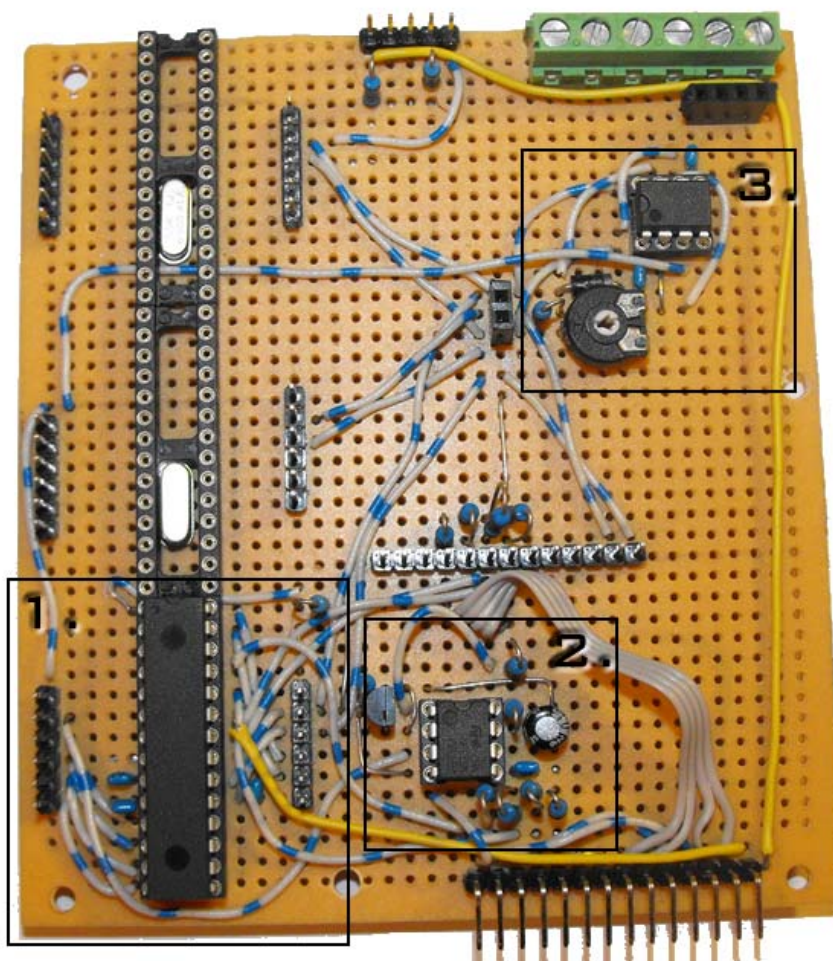


16. ábra. Az USB/RS232 illesztő modul programjának folyamatábrája

4. A kísérleti eszköz megépítése és vizsgálata

Az általam tervezett vezérlés prototípusát meg is építettem, ez a prototípus sok hasznos információval szolgált a további fejlesztési terveimhez. A prototípus gép alapját egy szkennert adja, melyből kizárólag a mechanikát hasznosítottam. A készülék bordásszíjas megoldást alkalmaz egy viszonylag nagy lépésközű léptetőmotorral, melyet azonban elláttak egy 1:25-ös lassító áttétellel. Tesztjeim eredményei természetesen a prototípuskészülékre vonatkoznak.

4.1. A kísérleti eszköz megépítése

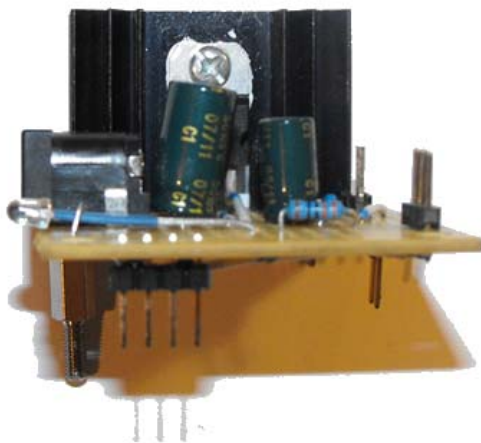


21. ábra. Az alaplap, rajta három modul

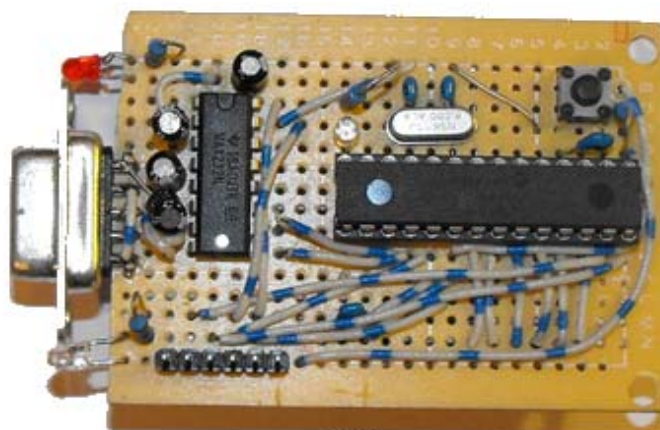
1.: Motorvezérlő modul

2.: Jelhosszabbító modul

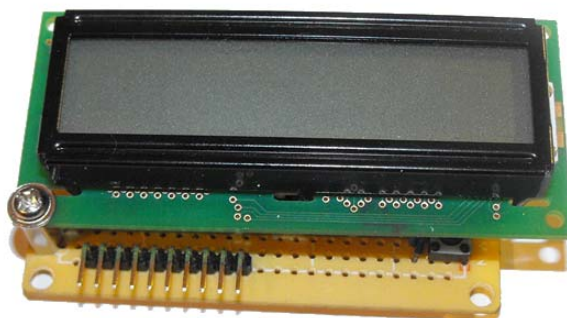
3.: PWM modul



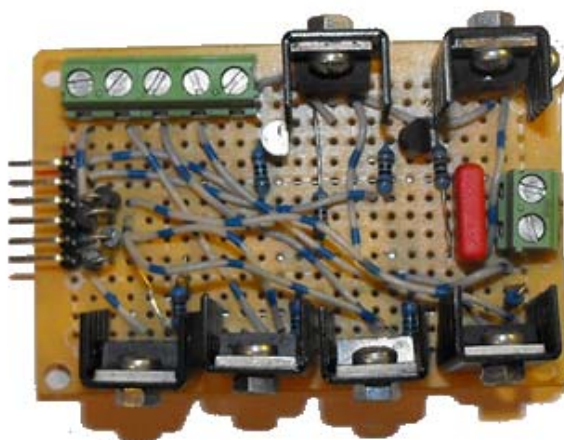
17. ábra: Tápmodul



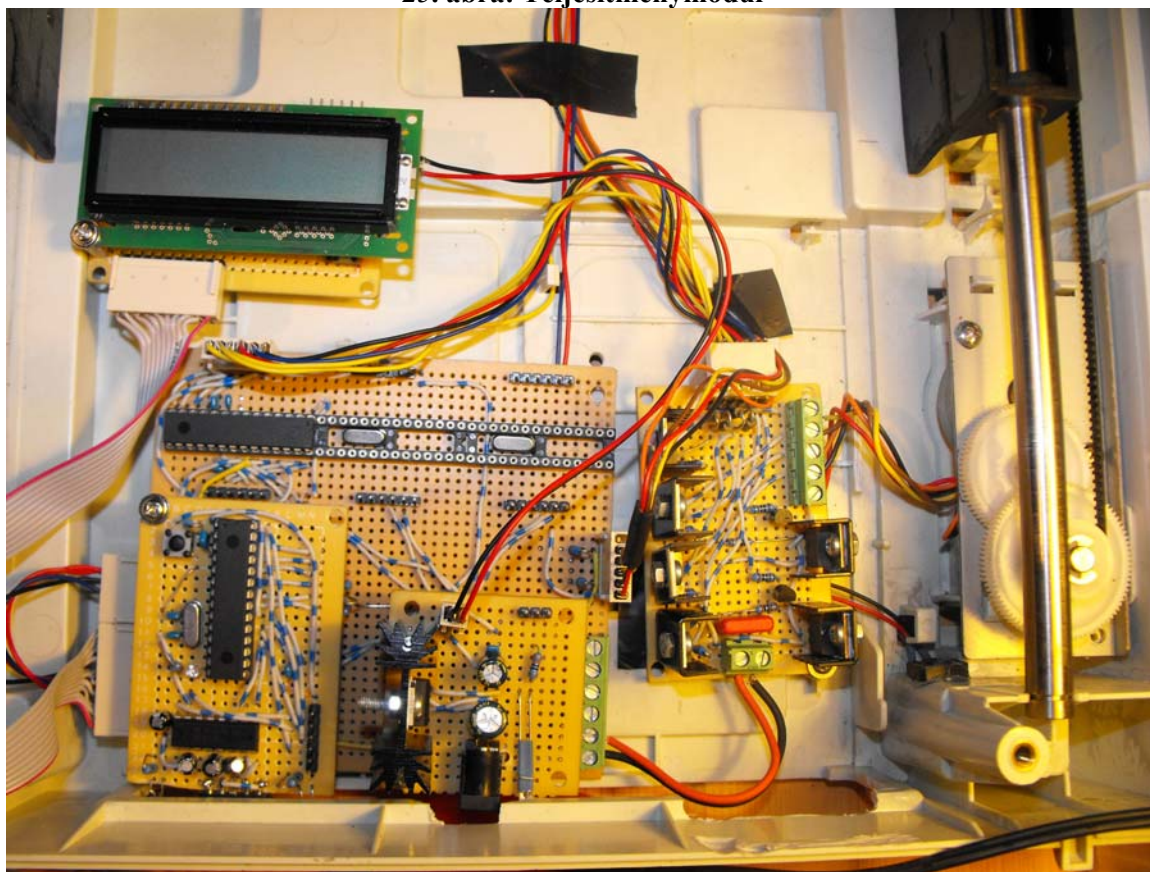
23. ábra: USB/RS232 illesztő modul



24. ábra: Kijelző modul



25. ábra: Teljesítménymodul



26. ábra: A kész modulok összeszerelve

4.2. A kísérleti eszköz vizsgálata

Pull In frekvencia, azaz a motor maximális vezérlőfrekvenciájának meghatározása

A teszt célja: Kideríteni azt a frekvenciát, mellyel a motort vezérelve az szinkronban képes maradni. Ezt a frekvenciát a műszaki gyakorlatban PullIn frekvenciának hívják, mértékegysége PPS.

A motort két üzemmódban üzemeltetem, a Home utasítás hatására nem teljes gerjesztéssel, annak csak a negyedével hajtom, míg végrehajtó üzemben a motor teljes gerjesztést kap a vezérlő impulzusok hatására.

Mivel két különböző működés lehetséges (HOME üzemmód, végrehajtó üzemmód) és ezt két külön gerjesztő feszültség mellett végezzük (HOME: PWM ~3V, végrehajtó üzem: 12V + PWM tartóáram) így mindkét esetet vizsgáltam.

Módszer: A frekvencia lépcsőzetes növelése, mindaddig, amíg a motor képes felvenni az ahhoz a frekvenciához tartozó fordulatszámot. A motor frekvenciáját ms-os nagyságrendben emeltem, azaz a két lépés közti várakozási időt ms-onként csökkentettem. A vizsgálatot 100ms-os lépésközzel kezdtem, majd folyamatosan feleztem a várakozási időt. Minden beállított érték után kipróbáltam, hogy a motor képes-e még felvenni az adott fordulatot.

Eszköz: A frekvenciát a MicroC Pro compiler segítségével, PICkit2 programozón keresztül égettem bele a motorvezérlő áramkörbe(HOME), illetve a soros átalakító modul(végrehajtó) mikrokontrollerébe.

Eredmények:

PWM ~3V Home üzemmódra jellemző érték

3. táblázat: maximális motorfrekvencia megállapítása Home üzemmódnál

Mérés sorszáma	lépésenkénti várakozás [ms]	Motorfrekvencia [Hz]	Sikeres teszt
1	100	10,0	igen
2	50	20,0	igen
3	25	40,0	igen
4	10	100,0	igen
5	5	200,0	nem
6	7	142,8	igen
7	6	166,6	igen

A fenti mérési eredmények alapján megállapítom, hogy ez az aktuátor jelen körülmények között ~166,6 Hz-es működést tesz lehetővé biztonsággal de valószínűsíthető, hogyha 1ms-nál kisebb lépésközt állítanék be, ez az eredmény magasabb is lehetne.

Végrehajtó üzem: 12V + PWM tartóáram

4. táblázat: maximális motorfrekvencia megállapítása végrehajtó üzemmódnál

Mérés sorszáma	lépésenkénti várakozás [ms]	Motorfrekvencia [Hz]	Sikeres teszt
1	100	10	igen
2	50	20	igen
3	25	40	igen
4	10	100	igen
5	5	200	igen
6	2	500	igen
7	1	1000	nem

A fenti mérési eredmények alapján megállapítom, hogy ez az aktuátor jelen körülmények között ~500 Hz-en képes működni, de valószínűsíthető, hogyha 1ms-nál kisebb lépésközt állítanék be, ez az eredmény magasabb is lehetne.

Jelhosszabbító modul

A motor teljes gerjesztésében tartásához alkalmazott jelhosszabbító modul számításainak ellenőrzése méréssel.

A teszt célja: Kideríteni, hogy a számítással mennyire egyezik a valóság, és hogy mennyi ideig marad teljes gerjesztésben a motor a vezérlőimpulzus megszűnése után.

Módszer: Oszilloszkóppal mértem a modul be és kimeneteit, majd kézzel egy impulzust adtam a jelhosszabbító bemenetére. A mért eredményeket összehasonlítottam a számítottakkal, majd még kétszer megismételtem a műveletet.

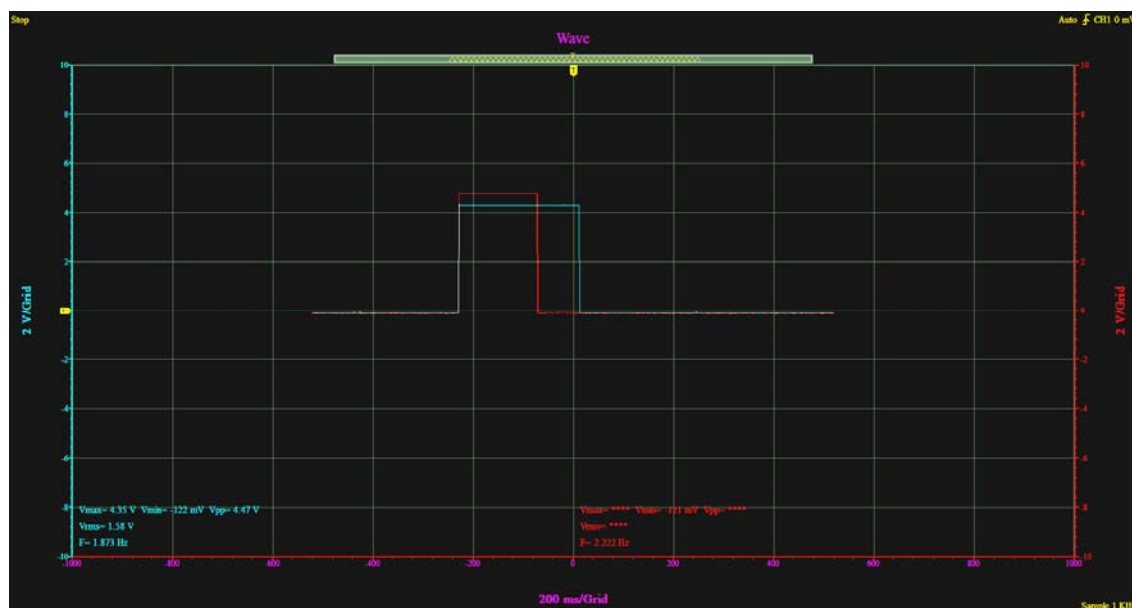
Eszköz: MDSO3.0 két csatornás 48MHz maximális mintavételi frekvenciájú kétsatornás PC Scope

Eredmények:

5.táblázat: jelhosszabbító modul mérési eredményei

Mérés sorszáma	Számított érték [ms]	Mért érték [ms]	különbség	elfogadható?
1	220	240	20	igen
2	220	239	19	igen
3	220	240	20	igen

A fenti eredmények alapján kijelenthetjük, hogy az elkészült szerkezet megfelel az elvárásoknak, habár a számítotthoz képest magas az eltérés.



36. ábra: jelhosszabbító szkópos vizsgálata

PWM

Az impulzus szélesség modulációt végző modul vizsgálata

Teszt célja: Megmérni a PWM modul jellemzőit, amiből a későbbi fejlesztésekhez fontos információkat nyerhetek.

Módszer: Oszilloszkóppal mértem a modul kimenetét, a mért jellemzőket feljegyeztem, ezt háromszor megismételtam (újraindítás), majd levontam a következtetéseket.

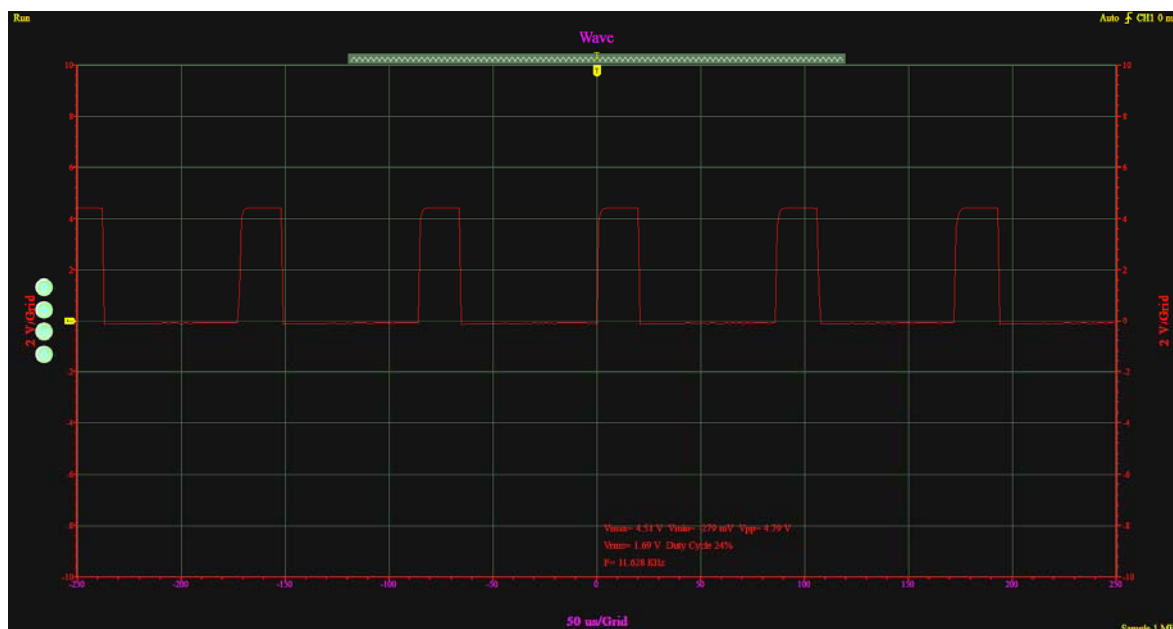
Eszköz: MDSO3.0 48MHz maximális mintavételi frekvenciájú kétsatornás PC Scope ,
Maxwell MX-25 304 multiméter

Eredmények:

6.táblázat: PWM modul mérési eredményei

Mérés sorszáma	Frekvencia [kHz]	RMS [V]	Kitöltési tényező [%]
1	11,63	1,20	24
2	11,61	1,15	23
3	11,62	1,20	24

A fenti eredmények alapján kijelenthetjük, hogy az elkészült szerkezet megfelel az elvárásoknak, bár, a mikrokontroller PWM moduljának jóval kisebb volna a hőmérsékletfüggése, s így nem is ingadozna a frekvencia, és a kitöltési tényező.



37. ábra: PWM jel

Hőmérséklet

Az üzembiztos működéshez feltétlenül tudnunk kell, nem melegedhet-e túl a készülék a használat során. A tervezés során közelítő számításokkal megbecsült teljesítménnyelvételek alapján számítottam ki a hőmérsékleti értékét a legjobban terhelt alkatrésznek, nevezetesen a L7805CV kódjelű táp IC-nek. Ezen eredményeket mindenképp szerettem volna a gyakorlatban is megvizsgálni.

Teszt célja: Megállapítani, hogy huzamos működtetés mellett is a túlhevülés veszélye nélkül lehet –e használni a készüléket

Módszer: A multiméter hőmérőszondáját a hűtőborda részébe helyezve elvégzem az alap mérést, amivel a kezdeti hőmérséklet értéket határozom meg. A készüléket áram alá helyezem, és szimulált feladatokat adok neki. (jellemzően mozgatom a motort) Percenként leolvasom a műszer által jelzett hőmérsékletet, majd az eredményeket összevetem az előzetesen számolt eredménnyel.

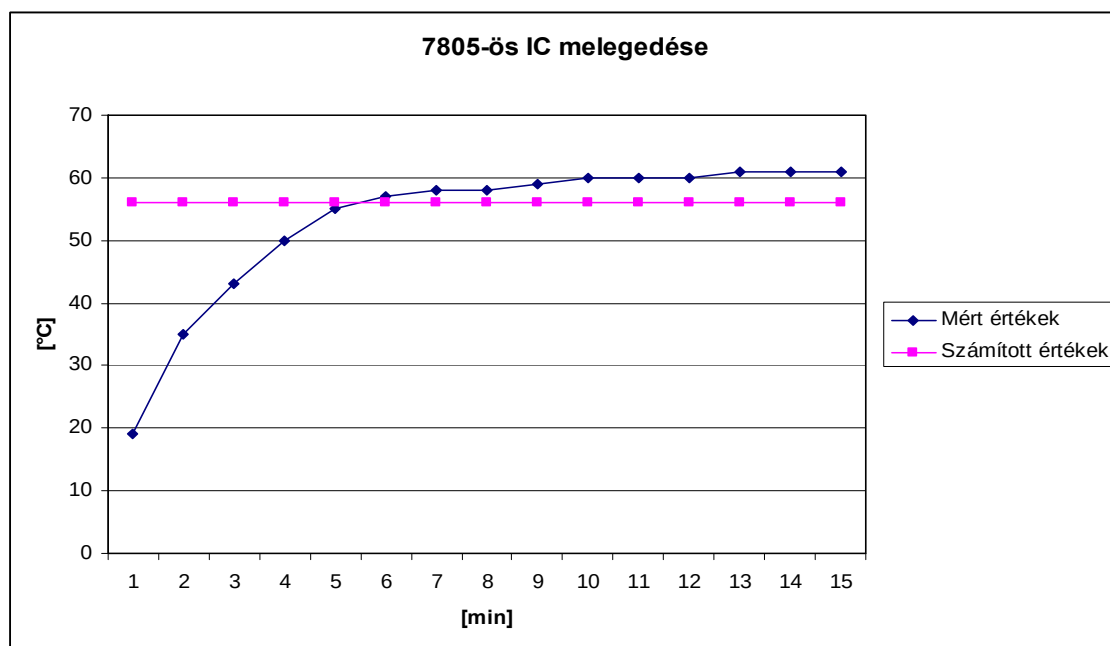
Eszköz: Maxwell MX-25 304 multiméter, K típusú hőmérő szondával.

Eredmények:

7.táblázat: L7805CV IC időbeni melegedése

Mérés sorszáma	eltelt idő [perc]	számított érték [°C] táll	mért érték [°C]	diff. [°C]	elfogadható?
1	0	56	19	37	igen
2	1	56	35	21	igen
3	2	56	43	13	igen
4	3	56	50	6	igen
5	4	56	55	1	igen
6	5	56	57	-1	igen
7	6	56	58	-2	igen
8	7	56	58	-2	igen
9	8	56	59	-3	igen
10	9	56	60	-4	igen
11	10	56	60	-4	igen
12	11	56	60	-4	igen
13	12	56	61	-5	igen
14	13	56	61	-5	igen
15	14	56	61	-5	igen

Habár a számított értékek erősen közelítő számítással lettek kiszámítva, a valósággal meglepően jó hasonlóságot mutatnak. Az összes érték elfogadható szinten van, így a berendezés 0 és 40°C között üzemeltethető.



38. ábra: L7805CV táp IC melegedése

Pontosság

Mivel a vezérlés diszkrét jelekkel dolgozik, belátható, hogy a készülék pontossága gyakorlatilag csak a szán hajtómechanizmusán és az aktuátor felbontásán múlik. A pontosságot szoftveresen is lehetséges növelni. Egyszerű programmódosítással félléptetési üzemi használatával a felbontást a duplájára lehetne növelni, vagy akár mikroléptetési vezérlést alkalmazva még tovább lehetne finomítani az egyes lépések közti távolságot. Ez utóbbi jócskán megnövelné a program bonyolultságát, és mindkét megoldás azzal járna együtt, hogy két lépés közt instabil pozícióba kerülne a motor, és egy esetleges fázisvesztés után kérdéses lenne a motor helyzete.

5. Gazdasági megfontolás

Az ipari életben nem mindegy, hogy egy készülék egy példányban, vagy esetleg sorozatban készül. Jelen esetben a prototípus elkészítésének költségeit tudom összevetni a piacon megtalálható megoldások áraival. Mivel kiskereskedésben alacsony darabszámban tudtam csak vásárolni az alkotóelemeket, így jóval drágább lett az előállítás, mintha nagy tételben cégesen tudtam volna ezeket megvenni. Ezt az értékelés részen sok más mellett szintén figyelembe veszem. A számítások során egy két tengely működtetésére képes tápegységgel, kijelzővel, és jelhosszabbító, PWM ill. teljesítménymodullal felszerelt, illesztőmodulon keresztül számítógéppel vezérelt készüléket veszek alapul.

Előállítás hozzávetőleges árának számítása

Mikrovezérlők: 3db PIC16F913 ~ 2500Ft
1db PIC16F917 ~ 1000Ft

Csatlakozók: Tüskesor x2 papa-mama ~ 800Ft
DC táp mama ~ 150Ft

RS232 ~ 150Ft

IC aljzat x7 ~ 400Ft

Sorkapocs x5 ~250Ft

Hűtőbordák: szimpla hajlított alu lemez x6 ~900Ft
Táphűtő, extrudált alu x1 ~ 200Ft

Teljesítmény elemek: N csat FET x4 ~600Ft
P csat FET x2 ~ 300Ft

Egyéb áramköri elemek: LCD ~ 1700Ft
tápIC ~ 100Ft

Kondenzátorok ca. 25 db ~600Ft

Ellenállások ca. 20 db ~ 200Ft

Időzítő IC x 3 ~ 200Ft

Raszter nyákok ~ 450Ft

Kristály x4 ~ 400Ft

Egyéb apróságok ~ 500Ft

Tápegység: 3-20V 12V 5000mA~ 5000Ft

Energiadíj: A forrasztásra szánt időből számolva ca. 1000Ft
Összesen:17400Ft.

Értékelés

A fentiek figyelembevételével véleményem szerint érdemes foglalkozni a dologgal, hisz ha nagy tételben állítanánk elő az általam tervezett rendszert tápegységgel, kijelzővel, programmal, és műszerdobozzal kerülne a három tengelyt vezérelni képes készülék 15000Ft- ba. Ez azt igazolja, hogy az általam létrehozott rendszer igen költséghatékony, bár a szervo rendszerekkel ellentétben nagyobb teljesítmények mellett költségesebbek a léptetőmotorok, mint a szervo megoldások.

6. Összefoglalás

A dolgozatom kitűzött célja egy árban és tudásban versenyképes léptetőmotor-vezérlés megtervezése volt, mellyel manuális szerszámgépek költséghatékony korszerűsítése valósítható meg és moduláris jellege miatt magas szinten testreszabható. A léptetőmotoros hajtás számos jó tulajdonsággal rendelkezik, főleg, ha költségkímélő, egyszerű megoldást keresünk. A felhasznált irodalom alapján nagy vonalakban elkészítettem a vezérlés tervét, majd mikor ez elkészült, finomítottam a terveken, moduláris felépítést adtam a rendszernek, magas szinten kompatibilissé tettem a mai világban használatos számítógépekhez. A villamos tervek elkészülte után elkészítettem az egyes modulok saját programját, majd a számítógép oldalán is elkészítettem egy egyszerű, tesztelésre alkalmas vezérlő programot. Mindezek után, azon villamos alkatrészeknél, amelyeknél fontos volt foglalkozni a túlmelegedés kérdésével, a műszaki gyakorlatban használt, közelítő beálló hőmérséklet-számítással meghatároztam az elemek állandósult szakaszban beálló hőmérsékletét. Biztonsági okokból mindkét helyen alkalmaztam addicionális hűtést. A kísérleti eszköz építése közben pár apróbb módosítást hajtottam végre mind az elektronikában, mind a programokban. Ezen eszköz építési költségei alapján hozzávetőleges számításokat végeztem gazdasági szempontból, és arra az eredményre jutottam, hogy az általam fejlesztett alkalmazás életképes lehet a piacon. Az értékelés folytán figyelembevettem azt is, hogy a visszacsatolást nélkülöző rendszerek megbízhatósága alacsonyabb, mint a zárt hurkú szabályzásoké. Arra jutottam azonban, hogy a fent vázolt szerkezet karbantartási igénye alacsonyabb, a bekerülési költségek szintén alacsonyabbak, a pontosabb, megbízhatóbb szervo megoldásokénál. Szükség esetén a léptetőmotorokat el lehet látni encoderekkel, így az általam tervezett megoldás átalakítható visszacsatolt rendszerré. Egy szerszámgép automatizálása előtt fontos kiszámítani a szükséges motorteljesítményt, ennek alapján a számított érték kétszeresével bíró aktuátort érdemes alkalmazni, elkerülendő a túlterhelésből adódó szinkronvesztési hibákat.

Ha a fentieket figyelembe vesszük, kijelenthetjük, hogy az általam tervezett vezérlés **kielégíti a vele szemben támasztott követelményeket.**

6.Summary

My dissertation's aim was to design a modular and highly customizable stepper-motor controll, wich makes it possible to modernize old machine tools in a cost effective way. A steppeing motor drive has many good traits, especially if we are looking for a low cost and simple solution. Based on references, I created the draft plan of the control, then refined and gave it a modular structure. The final layout is highly compatible with today's PC preferences. After finished with the plans of the electronical devices, I went on to write each module's program, and a simple control application for PC, wich can be used in later tests. As for the issue of overheating electronical components, I calculated approximately the stabilized temperature, and applied additional cooling where it was necessary. Some further modifications were made during the building of the test device. Calculations concerning the economic nature of my contraption, based on the test device's building cost, showed that it can be a viable compilation on the market. I also took in account that systems lacking feedback are less dependable, than those with feedback. Nevertheless the above discussed machinery has lower standards for maintenance and lower cost compared to precise and reliable, thus expensive servo drives. Should the need rise, the stepper-motors can be equipped with encoders, ensuring the feedback required. To avoid loss of synchronisation from overload, it is highly recommended to choose an actuator with twice the calculated output needed. The control I designed satisfies the applied requirements, that we can state considering the lines above.

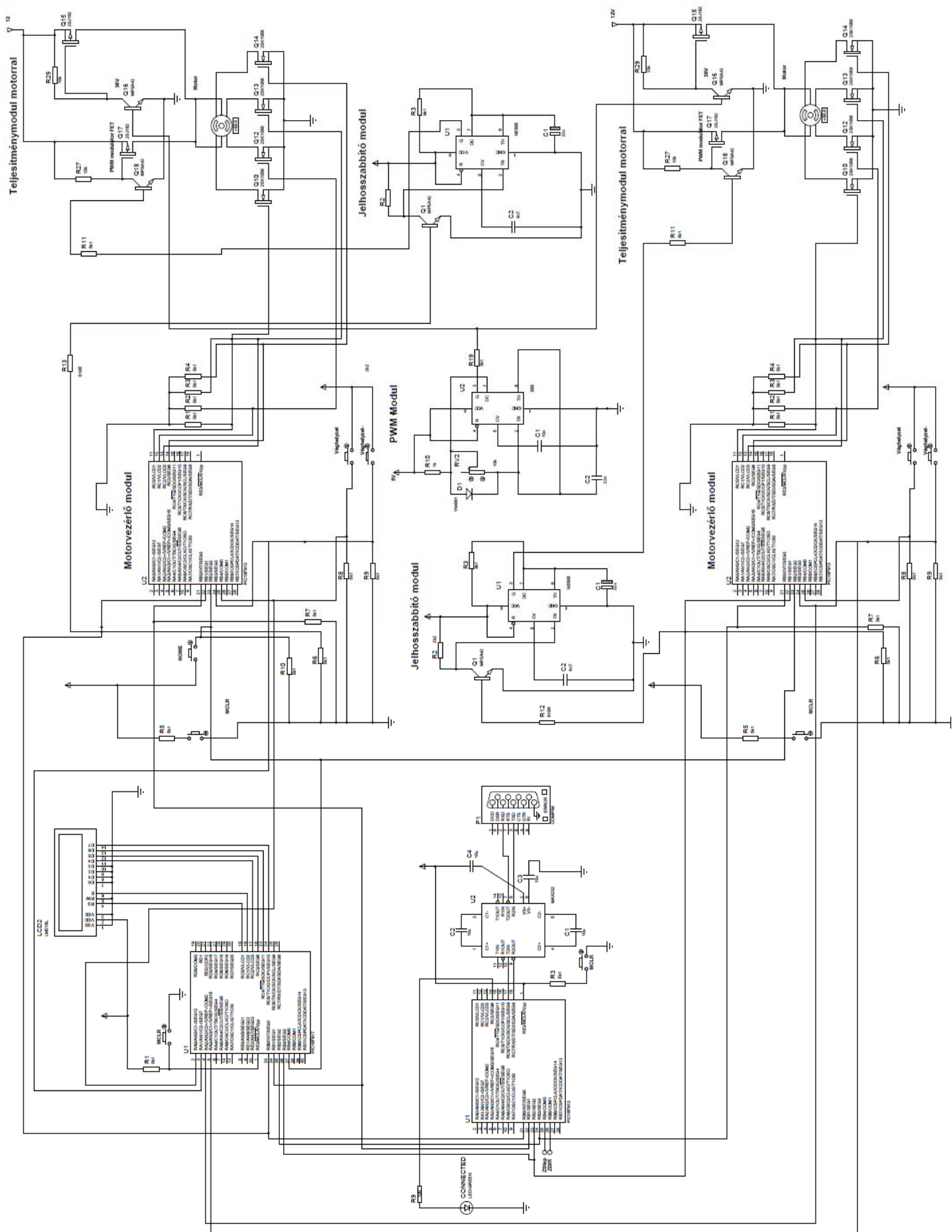
7.Irodalomjegyzék

- [1.] Wade Vonasek - Wood & Wood Products // CNC machinery goes mainstream
2009/2 p.49
- [2.] Csanády E., Kocsis Z. - Magyar Asztalos és Faipar // CNC technika elterjedése a
faiparban, 2007/10 p.92-96.
- [3.] Domenico Galeotalanza – CNC concepts, inc. // CNC Machines Versus Manual
Machines 2007. <http://www.cncci.com/resources/articles/cnc%20vs%20manual.htm>
- [4.] Gyukli Zoltán, Kocsis Imre CNC Praktikum 2011.
- [5.] Zsuffa Attila - Magyar Elektronika // Villamos hajtások és mozgásvezérlők -1 rész,
2006/6 p.44.
- [6.] Antal László, http://qtp.hu/elektro/leptetomotor_mukodese.php, 2008.
- [7.] Kébir Boukas ; Fouad M. AL-Sunni - Mechatronic Systems p.27, 2011-
ISBN 978-3-642-22323-5
- [8.] Byte Craft Limited – First Steps with Embedded Systems p.5. ,2002
- [9.] Kőnig Imre ; Pálinkás Tibor - Hobby Elektronika // Intelligens, kéttengelyes
léptetőmotor-
vezérlő (1.) p.169, 2000/5
- [10.] Ralph Locher- National Semiconductor Application Note 558 p.6-9, 1988.
- [11.] National Semiconductor – LM555 Timer p.7-9, 2006

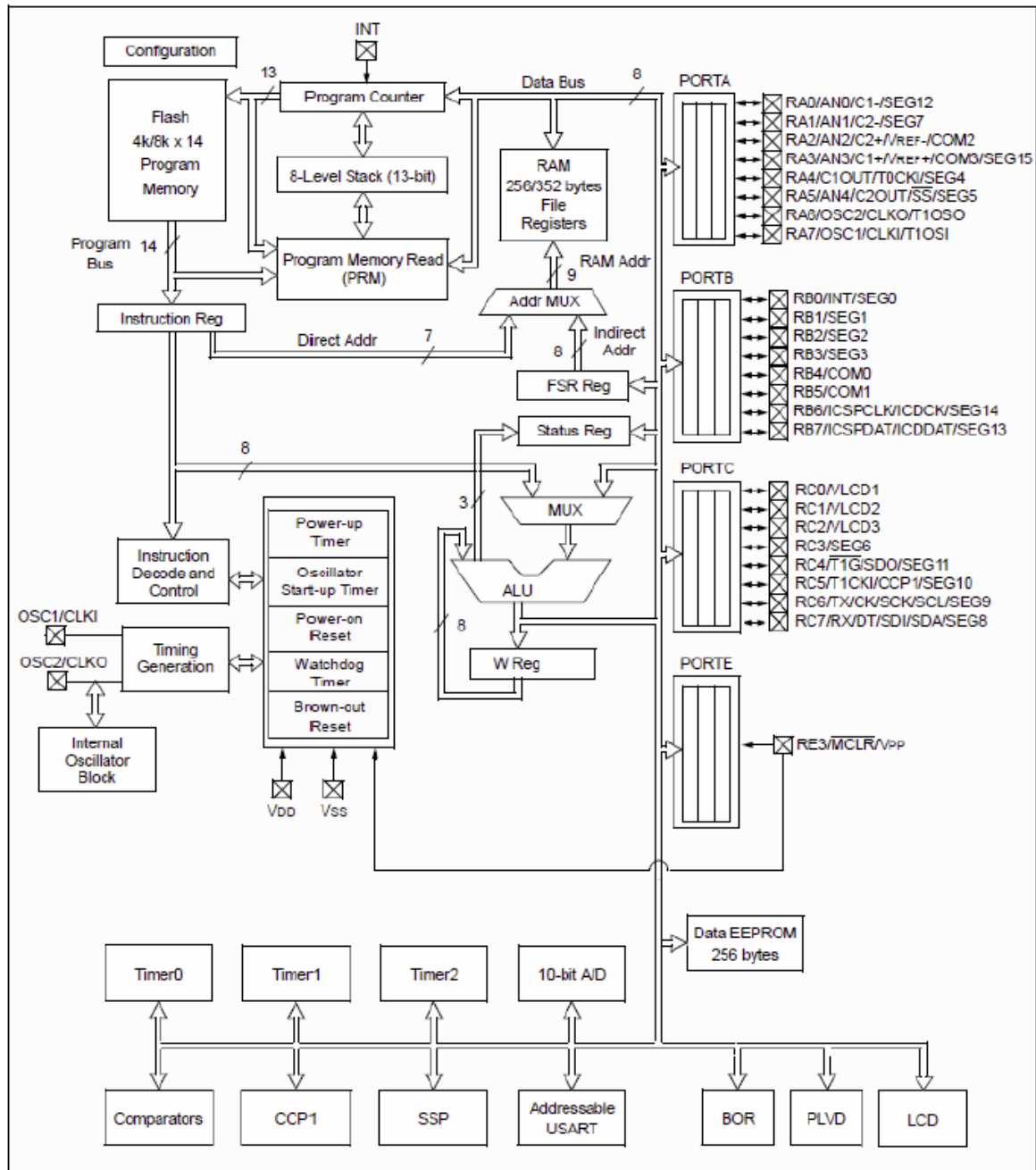
8. Köszönetnyilvánítás

Ezúton is szeretném megköszönni, és tiszteletemet kifejezni **Dr. Jánosi Lászlónak**, aki munkám során végig támogatott, hasznos tanácsokkal látott el, **Dr. Soumelidis Alexandros-nak** aki külső konzulensemként szintén sok fontos kérdésben tudott segíteni nekem, **Jánosi Andrásnak**, aki éjszakába nyúlóan segített leküzdeni egy makacs problémát **Gergely Zoltánnak**, akihez mindig bizalommal fordulhattam elektronikai kérdésekben, **Kellner Ferencnek**, aki problémáim hallatán remek tippekkel halmozott el és **Tóth János** szaktársamnak, akihez a felmerülő számítástechnikai problémákkal fordulhattam.

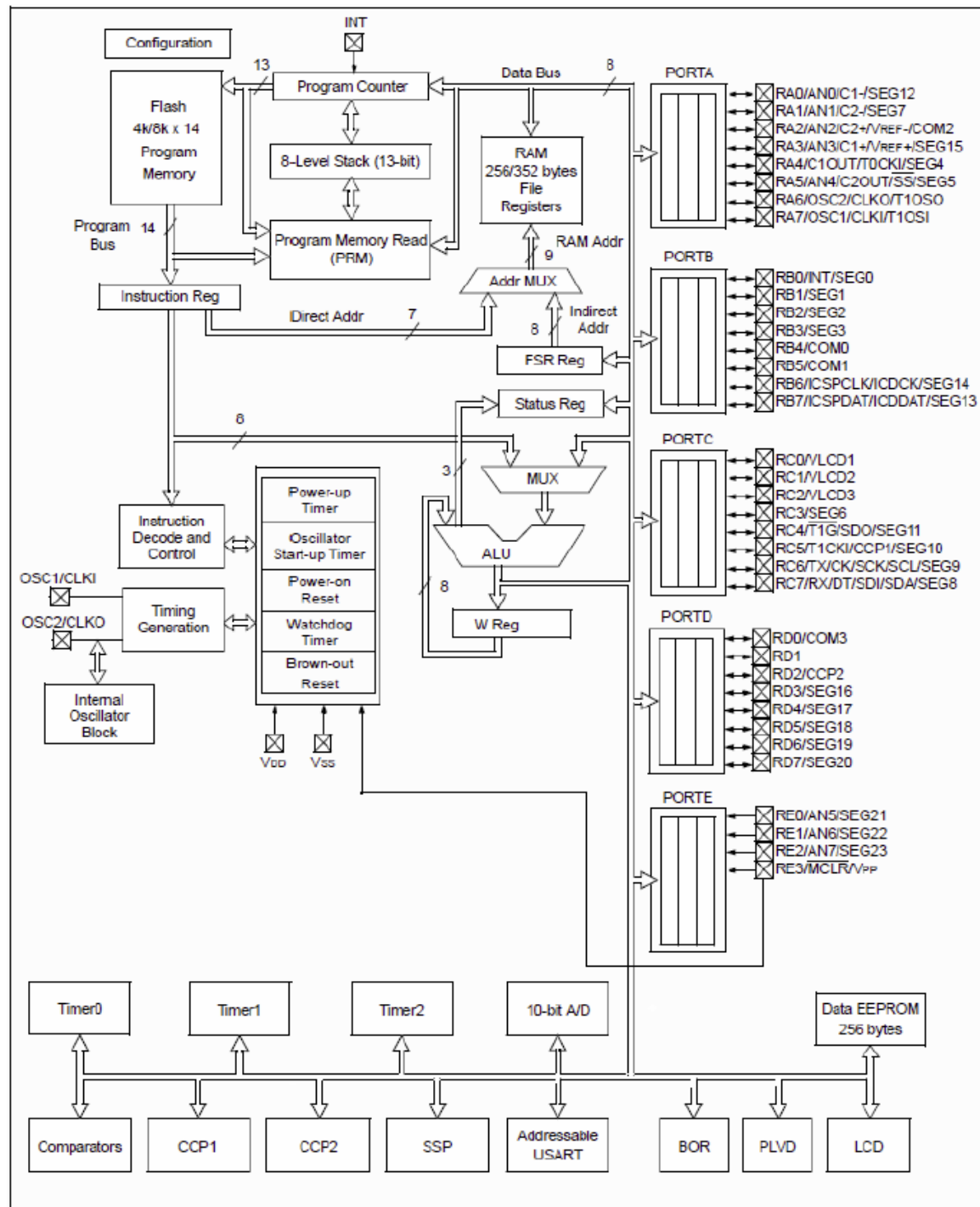
9.Mellékletek



2. Melléklet: PIC16F913 blokkdiagramja



3. Melléklet: PIC16F917 blokkdiagramja



4. Melléklet: A motorvezérlő programja

```
/*
*Erdélyi Viktor Ferenc (AWUS4E)
*   Unipoláris léptetőmotor vezérlés
*       2.2-es verzió
*
* Verziótörténet:
* PIC16F917 @ 8MHz
* 0.1 : Egyirányú forgatás
* 0.2 : Kétirányú forgatás
* 0.5 : Motortartóáram korlátozás szoftveresen, kétirányú forgatás
* 1.0 : Motortartóáram korlátozás szoftveresen, kétirányú forgatás, pár hiba javítva-
* működőképes verzió
* 1.2 : Motortartóáram korlátozás hardveresen, kétirányú forgatás, - működőképes verzió
* 1.5 : Motortartóáram korlátozás hardveresen, kétirányú forgatás, véghelyzetérzékelés,
* véghelyzet esetén fagyás.
*   - félig működőképes verzió
* 1.8 : Motortartóáram korlátozás hardveresen, kétirányú forgatás, véghelyzetérzékelés,
* véghelyzet esetén hibaüzenet.
*   - működőképes verzió
* 2.0 : Motortartóáram korlátozás hardveresen, kétirányú forgatás, véghelyzetérzékelés,
* véghelyzet esetén hibaüzenet,
*   Homeolás funkció(véghelyzetre áll). - működőképes verzió
* 2.1 : Motortartóáram korlátozás hardveresen, kétirányú forgatás, véghelyzetérzékelés,
* véghelyzet esetén hibaüzenet,
*   Homeolás funkció(véghelyzetre áll, majd lejön róla 4 lépéssel.). - működőképes
* verzió
* 2.2 : Motortartóáram korlátozás hardveresen, kétirányú forgatás, véghelyzetérzékelés,
* véghelyzet esetén hibaüzenet,
*   lépés 0-1 átmenet esetén következik be, Homeolás funkció(véghelyzetre áll, majd
* lejön róla 4 lépéssel.).
*   - működőképes verzió
* 3.0 : Átportolva PIC16F913.as mikrokontrollerre helytakarékoság, és a 917-es IC
* GPIO-inak kihasználatlansága miatt
*
* PIC16F913 @ 8MHz
* Budapest 2012.09.17
*/
```

```
/*HOME button pinout, and direction settings*/
sbit H at RB2_bit;
sbit H_d at TRISB2_bit;
```

```
/*axis limit pinout*/
sbit Xn at RB4_bit;
sbit Xp at RB3_bit;
```

```

/*axis limit direction*/
sbit Xn_d at TRISB4_bit;
sbit XP_d at TRISB3_bit;

/* X Motor pinout settings */
sbit MX_1 at RC0_bit;
sbit MX_2 at RC1_bit;
sbit MX_3 at RC2_bit;
sbit MX_4 at RC3_bit;

/* X Motor pin direction */
sbit MX_1_d at TRISC0_bit;
sbit MX_2_d at TRISC1_bit;
sbit MX_3_d at TRISC2_bit;
sbit MX_4_d at TRISC3_bit;

/*Motorcontrol pinout settings */
sbit Xstep at RB1_bit;
sbit Xdir at RB0_bit;

/*Motorcontrol pin direction*/
sbit Xstep_d at TRISB1_bit;
sbit Xdir_d at TRISB0_bit;

//variables
char j;
char i;
//flag
bit oldstate;
bit HOM;
bit kész;
void main() {
  ANSEL = 0;
  LCDCON = 0;           // Configure LCD pins as digital
  wpub = 0;
  T1CON = 0;
  SSPCON=0;
  CCP1CON=0;
  RCSTA=0;
  IOCB =0;
  intcon =0;

  Xdir_d = 1;
  Xstep_d = 1;          // Motorcontrol bit set as input

  Xn_d=1;                // set Axis Limit bits as input
  Xp_d=1;
  H_d=1;                 //set HOME button as input

  MX_1_d = 0;

```

```

MX_2_d = 0;           //X Motor direction set as output
MX_3_d = 0;
MX_4_d = 0;

MX_1 = 0;
MX_2 = 0;
MX_3 = 0;             // Set Motor value 0
MX_4 = 0;

oldstate=0;
HOM=0;
kesz=0;               //initializes oldstate flag
j=0;                  //set counter 0
i=0;
while(1){              //Start Loop
    if (H==1){
        HOM =1;
        while (Xn==0){ //do it 'till hardware limit
            if (j == 0){
                MX_2 = 0;
                MX_3 = 0;
                MX_4 = 0;
                MX_1 = 1;
                j++;
                delay_ms(2);
            }

            if(j==1){
                MX_1 = 0;
                MX_3 = 0;
                MX_4 = 0;
                MX_2 = 1;
                j++;
                delay_ms(2);
            }

            if (j == 2){
                MX_2 = 0;
                MX_1 = 0;
                MX_4 = 0;
                MX_3 = 1;
                j++;
                delay_ms(2);
            }

            if(j==3){
                MX_2 = 0;
                MX_3 = 0;

```

```

        MX_1 = 0;
        MX_4 = 1;
        j=0;
        delay_ms(2);
    }
    } //end subloop
}

if (HOM==1 && Xn==1){
    delay_ms(100);
    while(xn==1){
        if (j == 0) {
            j=4;
            MX_1 = 0;
            MX_2 = 0;
            MX_3 = 0;
            MX_4 = 1;
            j--;
            i++;
            delay_ms(100);
        }

        if (j == 3) {
            MX_1 = 0;
            MX_2 = 0;
            MX_3 = 1;
            MX_4 = 0;
            j--;
            i++;
            delay_ms(100);
        }

        if (j == 2) {
            MX_1 = 0;
            MX_2 = 1;
            MX_3 = 0;
            MX_4 = 0;
            j--;
            i++;
            delay_ms(100);
        }

        if (j == 1) {
            MX_1 = 1;
            MX_2 = 0;
            MX_3 = 0;
            MX_4 = 0;
            j--;
            i++;
        }
    }
}

```

```

        delay_ms(100);
    }
}

}

if (XDIR ==0 && Xn==0){           //LEFT
    if (j == 0){
        if (Xstep==1 && oldstate==0){
            oldstate=1;
            MX_2 = 0;
            MX_3 = 0;
            MX_4 = 0;
            MX_1 = 1;
        }

        if(Xstep==0 && oldstate==1){
            oldstate=0;
            j++;
        }
    }
}

if (XDIR ==1 && Xp==0){           //RIGHT
    if (j == 0) {
        if (Xstep==1 && oldstate==0) {
            j=4;
            oldstate=1;
            MX_1 = 0;
            MX_2 = 0;
            MX_3 = 0;
            MX_4 = 1;
        }

        if(Xstep==0 && oldstate==1 && j==4){
            oldstate=0;
            j--;
        }
    }
}

if (XDIR ==0 && Xn==0){           //LEFT
    if(j==1){
        if(Xstep==1 && oldstate==0){
            oldstate=1;
            MX_1 = 0;
            MX_3 = 0;
            MX_4 = 0;

```

```

        MX_2 = 1;
    }

    if(Xstep==0 && oldstate==1){
        oldstate=0;
        j++;
    }
}

if (XDIR ==1 && Xp==0){           //RIGHT
    if (j == 3){
        if (Xstep==1 && oldstate==0){
            oldstate=1;
            MX_2 = 0;
            MX_1 = 0;
            MX_4 = 0;
            MX_3 = 1;
        }

        if(Xstep==0 && oldstate==1){
            oldstate=0;
            j--;
        }
    }
}

if (XDIR ==0 && Xn==0){           //LEFT
    if (j == 2){
        if (Xstep==1 && oldstate==0){
            oldstate=1;
            MX_2 = 0;
            MX_1 = 0;
            MX_4 = 0;
            MX_3 = 1;
        }

        if(Xstep==0 && oldstate==1){
            oldstate=0;
            j++;
        }
    }
}

if (XDIR ==1 && Xp==0){           //RIGHT
    if (j == 2){
        if (Xstep==1 && oldstate==0){
            oldstate=1;
            MX_1 = 0;
            MX_3 = 0;

```

```

        MX_4 = 0;
        MX_2 = 1;
    }

    if(Xstep==0 && oldstate==1){
        oldstate=0;
        j--;
    }
}

if (XDIR ==0 && Xn==0){           //LEFT
    if(j==3){
        if(Xstep==1 && oldstate==0){
            oldstate=1;
            MX_2 = 0;
            MX_3 = 0;
            MX_1 = 0;
            MX_4 = 1;
        }

        if(Xstep==0 && oldstate==1){
            oldstate=0;
            j=0;
        }
    }
}

if (XDIR ==1 && Xp==0){           //RIGHT
    if (j == 1){
        if (Xstep==1 && oldstate==0){
            oldstate=1;
            MX_2 = 0;
            MX_3 = 0;
            MX_4 = 0;
            MX_1 = 1;
        }

        if(Xstep==0 && oldstate==1){
            oldstate=0;
            j=0;
        }
    }
}

//End Loop
}

```

5. Melléklet: Az LCD vezérlő programja

```
/*átportolva PIC16F917-re a 913-as IC GPIO-inak elégtelen száma miatt.  
lépésenkénti elmozdulás beállítva 0,04mm/step-re  
*/
```

```
/* LCD pinout settings */  
sbit LCD_RS at RC0_bit;  
sbit LCD_EN at RC1_bit;  
sbit LCD_D7 at RC2_bit;  
sbit LCD_D6 at RC3_bit;  
sbit LCD_D5 at RD0_bit;  
sbit LCD_D4 at RD1_bit;
```

```
/* LCD pin direction */  
sbit LCD_RS_Direction at TRISC0_bit;  
sbit LCD_EN_Direction at TRISC1_bit;  
sbit LCD_D7_Direction at TRISC2_bit;  
sbit LCD_D6_Direction at TRISC3_bit;  
sbit LCD_D5_Direction at TRISD0_bit;  
sbit LCD_D4_Direction at TRISD1_bit;
```

```
//counter input bits  
sbit Xdir at RB0_bit;  
sbit Xstp at RB1_bit;  
sbit Ydir at RB2_bit;  
sbit Ystp at RB3_bit;
```

```
//counter input pin directions  
sbit Xdir_d at TRISD5_bit;  
sbit Xstp_d at TRISD6_bit;  
sbit Ydir_d at TRISC6_bit;  
sbit Ystp_d at TRISC5_bit;
```

```
//limit  
sbit Xnv at RD4_bit;  
sbit Xpv at RC7_bit;  
sbit Ynv at RC4_bit;  
sbit Ypv at RD3_bit;
```

```
//limit dir  
sbit Xnv_d at TRISD4_bit;  
sbit Xpv_d at TRISC7_bit;  
sbit Ynv_d at TRISC4_bit;  
sbit Ypv_d at TRISD3_bit;
```

```
//HOME  
sbit HOME at RD7_bit;
```

```

sbit HOME_d at TRISD7_bit;
bit HOM;
unsigned char t;
int o;
char i;
int j;
int x;
int y;
bit oldstate;
bit oldstate1;
bit xp;
bit xn;
bit yp;
bit yn;
int k;
char Xkoor[] = {"\0","\0","\0","\0","\0","\0","\0","\0"};
char Ykoor[] = {"\0","\0","\0","\0","\0","\0","\0","\0"};

void num2str(char ch[]);

void main() {

    ANSEL = 0;    // Configure AN pins as digital
    LCDCON = 0;   // Configure C ports as I/O
    wpub=0;       // weak pull up disabled
    IOCB=0;       // interrupt disabled
    Xnv_d=1;
    Xpv_d=1;
    Ynv_d=1;
    Ypv_d=1;
    Xdir_d = 1;
    HOM=0;
    Xstp_d = 1;
    Ydir_d = 1;
    Ystp_d = 1;
    oldstate=0;
    oldstate1=0;
    HOME_d=1;
    x=0;
    t = 0;
    y=0;
    k=0;
    j=0;
    o=4;          // változtatás: 0.04 -> 4
    xp =0;
    xn=0;
    yp=0;
    yn=0;
    LCD_RS_Direction = 0; //LCD RS,EN direction Output

```

```

LCD_EN_Direction = 0;

Lcd_Init();
//Initialising LCD
Delay_ms(100);
Lcd_Cmd(_LCD_CLEAR);
Delay_ms(100);
Lcd_Cmd(_LCD_CURSOR_OFF);
Lcd_Out(1, 2, "Erdelyi Viktor");
Delay_ms(100);
Lcd_Out(2, 6, "AWUS4E");
Delay_ms(2000); //wait, till Operator read

for(i=0; i<16; i++)
{
    Lcd_Cmd(_LCD_SHIFT_right); //TEXT move from rigt to left
    Delay_ms(250);
}
Lcd_Cmd(_LCD_CLEAR); // CLEAR Screen
Delay_ms(250);
Lcd_Cmd(_LCD_CLEAR);
Lcd_Out(1, 4, "Vegyen fel");
Lcd_Out(2, 4, "nullpontot!");

while(1){ //start of loop
    if (HOME==1){
        HOM=1;
        Lcd_Cmd(_LCD_CLEAR);
        Lcd_Out(1, 5, "HOMEOLAS");
        Lcd_Out(2, 3, "FOLYAMATBAN");
    }
    if (HOM==1 && Xnv==1){
        Lcd_Cmd(_LCD_CLEAR);
        delay_ms(100);
        Lcd_Out(1, 2, "Motor vezerles");
        Lcd_Out(2, 3, "keszenletben.");
        HOM=0;
        x=0;
        y=0;
        k=0;
        j=0;
        oldstate=0;
        oldstate1=0;
        xp =0;
        xn=0;
        yp=0;
        yn=0;
        delay_ms(100);
    }
}

```

```

if(Xdir==1){
    if (Xstp==1) {          // Detect logical one
        xp = 1;
        oldstate=1;          // Update flag
    }
    if (xp && Xstp==0) { // Detect one-to-zero transition
        x=x+1;
        xp=0;
    }
}

if(Xdir==0){
    if (Xstp==1) {          // Detect logical one
        xn = 1;
        oldstate=1;          // Update flag
    }
    if (xn && Xstp==0) { // Detect one-to-zero transition
        x=x-1;
        xn=0;
    }
}

if(Ydir==1){
    if (Ystp==1) {          // Detect logical one
        yp = 1;
        oldstate=1;          // Update flag
    }
    if (yp && Ystp==0) { // Detect one-to-zero transition
        y=y+1;
        yp=0;
    }
}

if(Ydir==0){
    if (Ystp==1) {          // Detect logical one
        yn = 1;
        oldstate=1;          // Update flag
    }
    if (yn && Ystp==0) { // Detect one-to-zero transition
        y=y-1;
        yn=0;
    }
}

if(oldstate1==0){
    if(oldstate==1){
        Lcd_Cmd(_LCD_CLEAR);          // CLEAR Screen
        //Delay_ms(250);
        Lcd_Out(1, 1, "X:");
    }
}

```

```

    Lcd_Out(1, 11, "mm");
    Lcd_Out(2, 1, "Y:");
    Lcd_Out(2, 11, "mm");
    j=0;
    delay_ms(1);
    oldstate1=1;
    }
}

if((oldstate1==1) && (oldstate==1)){
    t++;
    if (t==10)
    {
        k=x*o;
        IntToStr(k, Xkoor);
        num2str(Xkoor);
        Lcd_Out(1, 4, Xkoor);          // print coordinates

        j=y*o;
        IntToStr(j, Ykoor);
        num2str(Ykoor);
        Lcd_Out(2, 4, Ykoor);          // print coordinates
        t=0;
    }
    }                                     //end of loop
}

void num2str(char ch[])                // argumentuma: string (char[])
{
    volatile int i;                    // optimizált változó a 'for' ciklushoz
    volatile int ctr = 8;

    char temp1 = '\0';
    char temp2 = '\0';

    for(i = 7; i>=1;i--)                // tömb mérete 14, addig megy a ciklus
    {
        if(ch[i] == '\0') ctr--;        // amég nem tizedespont (az elejétől), növeljük a számlálót
        else break;                    // amint tizedespont kilépünk a 'for' ciklusból
    }

    if(ctr == 1){
        temp1 = ch[ctr-1];
        //temp2 = ch[ctr];
        ch[0] = '0';
        ch[1] = '.';
        ch[2] = '0';
        ch[3] = temp1;
    }else if (ctr == 2){

```

```

    if(ch[0] == '-') {
        temp1 = ch[ctr-1];
        ch[1] = '0';
        ch[2] = '.';
        ch[3] = '0';
        ch[4] = temp1;
    } else if(ch[0] != '-') {
        temp1 = ch[ctr-2];
        temp2 = ch[ctr-1];
        ch[0] = '0';
        ch[1] = '.';
        ch[2] = temp1;
        ch[3] = temp2;
    }
} else if (ctr == 3) {
    if(ch[0] == '-') {
        temp1 = ch[ctr-2];
        temp2 = ch[ctr-1];
        ch[1] = '0';
        ch[2] = '.';
        ch[3] = temp1;
        ch[4] = temp2;
    } else if(ch[0] != '-') {
        temp1 = ch[ctr-2];
        temp2 = ch[ctr-1];
        ch[ctr-2] = '.';
        ch[ctr-1] = temp1;
        ch[ctr] = temp2;
    }
} else {
    temp1 = ch[ctr-2];
    temp2 = ch[ctr-1];
    ch[ctr-2] = '.';
    ch[ctr-1] = temp1;
    ch[ctr] = temp2;
}
}
}

```

6. Melléklet: Az USB/RS232 illesztő modul programja

```

/* UART_control.c
 * Soros porton történő vezérlés.
 * Érkező adat szintaxisa: $xxxxxxx:yyyyyy$
 *   - '$' = üzenet eleje és vége
 *   - x = vezérlő bitek (max 8)
 *   - ':' = for-ciklust vezérlő érték jele
 *   - y = for-ciklus maximuma (max 8 karakter (számjegy))
 *
 * 2012, Erdélyi Viktor
 */

// apcsolatot jelző led
sbit LED_CONN at RC3_bit;
sbit LED_CONN_D at TRISC3_bit;

// lobális változók
char uart_ch; // beolvasott karakter
int ctr = 0; // '$' számláló
int index = 0; // aktuális parancstömb indexe
int is_for = 0; // for-ciklus jelző
int ready = 0; // parancs végrehajtás jelző
int dec = 0; // tizes helyiérték jelző
int for_end = 0; // for-ciklus maximuma (lépések)
char command[8]; // parancs tömb (karakter)
int bin_cmd[8]; // parancs tömb (bináris , 0-1)
char steps[8]; // lépéseket tároló tömb
const int WAIT = 1; // lépések közötti várakozás [ms]
int j;

// egyszerű hatványozó fv.
int _pow(int x, int y){
    int i;
    int ret = 1;

    for(i = 0; i < y; i++){
        ret *= x;
    }
    return ret;
}

// steps tömböt számokká alakító fv.
int charr2num(char ch[],int dec){
    int ret = 0; // inicializált visszatérési érték
    int i = 0; // for-ciklus számlálója

    // visszatérési érték számítása:
    // pl: 256 = 2*10^2 + 5*10^1 + 6*10^0
    // char 2 = int 50 (ASCII)

```

```

// 50 - 48 = 2
// dec = 3
// i = 0 -> 2
// kitevő : 2 -> 0
for(i = 0; i < dec; i++){
    ret += ((int)ch[i]-48)*_pow(10,dec - 1 -i);
}

return ret; // visszatérési érték beállítása
}
// kiindulási helyzetbe állítás
void stop(void){
    ctr = 0;
    index = 0;
    is_for = 0;
    ready = 0;
    dec = 0;
    for_end = 0;

    for(j = 0; j < 8; j++)
    {
        command[j] = '0'; // parancs tömb (karakter)
        steps[j] = '0'; // lépéseket tároló tömb (karakter)
        bin_cmd[j] = 0; // parancs tömb (bináris , 0-1)
    }
}

// fő fv.
void main() {
    TRISB = 0; // portok iránya
    PORTB = 0; // portok értéke
    LCDCON=0; //LCD portok kikapcsolva
    wpub=0; //weak pull up kikapcsolva
    IOCB=0; //interruptok kikapcsolva

    LED_CONN_D = 0;
    LED_CONN = 0;

    UART1_Init(9600); // UART beállítása
    Delay_ms(100); // várakozás a kapcsolódásra, inicializálásra

    // mag
    while(1){
        if(UART1_Data_Ready()){ // ha lehet adatot olvasni

            uart_ch = UART1_Read(); // adat karakterenként történő beolvasása

            if(uart_ch == '?'){ // eszköz keresése
                UART1_Write("!"); // visszaigazolás
            }
        }
    }
}

```

```

    LED_CONN = 1;      // kapcsolódást jelző led bekapcsolása
} else if(uart_ch == 'e'){ // kapcsolat lezárása
    LED_CONN = 0;      // kapcsolódást jelző led kikapcsolása
    stop();
} else if(uart_ch == '$'){ // adat eleje v vége
    ctr++;              // '$' számláló növelése (max 2)
} else if(uart_ch == ':'){ // for-ciklust vezérlő érték jele
    is_for = 1;        // for-ciklus beállítása
    index = 0;         // index nullázása
} else if(ctr == 1 && is_for == 0 ){ // adatok olvasása ,DE még nem for-
ciklushoz
    command[index] = uart_ch; // parancs tömb feltöltése
    index++;                // index léptetése
} else if(ctr == 1 && is_for == 1 ){ // adatok olvasása for-ciklushoz
    steps[index] = uart_ch;   // lépések tömb feltöltése
    index++;                  // index növelése
    dec++;                    // helyiérték számláló növelése
}

if(ctr == 2){               // vége az üzenetnek
    index = 0;              // index nullázása
    ctr = 0;                // '$' számláló nullázása
    ready = 1;              // kész jelzés;
}

if(ready == 1){
    for_end = charr2num(steps,dec); // for-ciklus maximumának konvertálása
    dec = 0;                    // tizes helyiérték számláló nullázása
    is_for = 0;                 // for-ciklus számláló nullázása

    // char: 0; ASCII: 48 => 48-48 = int 0
    // char: 1; ASCII: 49 => 49-48 = int 1
    for(j = 0; j < 8; j++){
        bin_cmd[j] = (int)(command[j]) - 48;
    }
}

// parancs végrehajtása
if(ready == 1){
    int i; // lokális for-ciklus számláló
    for(i = 0; i < for_end; i++){
        // PINek értékadása
        portb.f0 = bin_cmd[0];
        portb.f1 = bin_cmd[1];
        portb.f2 = bin_cmd[2];
        portb.f3 = bin_cmd[3];
        portb.f4 = bin_cmd[4];
        portb.f5 = bin_cmd[5];
        portb.f6 = bin_cmd[6];
        portb.f7 = bin_cmd[7];
    }
}

```

```
        // állapot megtartása
        Delay_ms(WAIT);

        // nullázás
        portb.f0 = 0;
        portb.f1 = 0;
        portb.f2 = 0;
        portb.f3 = 0;
        portb.f4 = 0;
        portb.f5 = 0;
        portb.f6 = 0;
        portb.f7 = 0;

        // állapot megtartása
        Delay_ms(WAIT);
    }

    // parancs befejezésének jelzése
    ready = 0;

    // parancs vége karakter visszajelzése
    UART1_Write('k');
}
}
}
```

7. Melléklet: A kész kísérleti eszköz fényképe

