SZAKDOLGOZAT

Katona-Zsombori Zsolt
2009

Iskolai robotok sokoldalú felhasználása a technika tanításában

Egy segédrobot Internetes irányításának megvalósítása

PÉCSI TUDOMÁNYEGYETEM
Természettudományi Kar
Matematikai és Informatikai Intézet
Pécs, 2009

Témavezető:

Dr. Hegyi Sándor

CSc, egyetemi docens
tanszékvezető

Készítette:

Katona-Zsombori Zsolt

Számítástechnika-Techika szak
nappali tagozat
katonazs@gmail.com

Tartalomjegyzék

Feladatkiírás

Szakdolgozat téma

HUNveyor-HUSAR 2 minimál gyakorló űrszonda modell fejlesztése, programozása.

Témavezető

Dr. Hegyi Sándor

Megoldandó feladat

HUNveyor-HUSAR 2 minimál gyakorló űrszonda rendszerhez tartozó C nyelven programozható ASURO, továbbá PIC processzorral vezérelt HUSAR mobil robotok (roverek) programozása és továbbépítése.

Információk az egyetemi gyakorló űrszonda modellről

http://hu.wikipedia.org/wiki/Hunveyor

Rendelkezésre álló eszközök

Terepasztalra helyezett gyakorló minimál űrszondák, robotok (roverek) és irányítástechnikai eszközök.

Előszó

Amikor még szakközépiskolába jártam csodálattal töltött el az a robot, amivel az akkori végzősök foglalkozhattak, programozhatták, gyakorlatban is kipróbálhatták az előző években tanultakat. Mire ötödikes lettem feledésbe merült, és már senki nem foglalkozott a KUKA nevű robottal. Mindig sajnáltam, hogy nem volt lehetőségem hozzáférni, felfedezni, kiismerni a működését. Ekkor már érdekeltek a számítógéppel vezérelhető, eszközök, de nem tudtam hozzáférni sehol még hasonló dolgokhoz sem.

A HUSAR (Hungarian University Surface Rover) program egy kísérleti gyakorló űrszonda modellje a HUNVEYOR projekten belül, amit a Spirit, és az Opportunity, Marson használatos roverek alapján építenek Magyarországon több egyetemen, és szakközépiskolában. A projekt célja az, hogy megismerkedjenek a hallgatók az űrkutatással, annak problémáit átlássák, és fantáziájukat elengedve új ötletekkel segítsék az űrkutatás további fejlesztésének menetét. Ez pedagógiai szempontból nézve sem elhanyagolható, hiszen a fejlesztés közben legtöbbször gyakorlatban, tapasztalati úton megszerzett rendszerszemlélet elsajátítása magától értetődően megy, részévé válik gondolkodásuknak.

Témaválasztáskor nagy mértékben szerepet játszottak a középiskolai előzmények, és az egyetemen folytatott űrkutató-fejlesztő munka, ami alatt az egyik rádió-távirányítású rover a *Husar-2b* felújítását értem (I.4 rész). A NASA teszten való részvétel után, a jármű szörnyű állapotba került, ez indokolta a teljes felújítást. Javítása közben felmerült bennem, hogy meg kellene oldani, hogy a rover irányítható legyen az Internetről. Persze teljesen másképpen képzeltem el, mint ahogy azt a NET-Husaron látni fogjuk a továbbiakban.

Dr. Hegyi Sándor a HUNVEYOR projekt PTE-n belüli vezetője által kiírt szakdolgozati témák között volt HUSAR robotok és távirányításának feladata is. Ezt megfelelő nehézségű kihívásnak éreztem, ezért belevágtam a megvalósításba.

Köszönetnyilvánítás

Külön köszönetet szeretnék mondani:

- **Gőcze Zoltánnak**, az elektronikai problémák leküzdésében,
- **Griechisch Erikának** és **Major Szabolcs Álmosnak** a lektorálásban
- **Imrek Gyulának** a NET-Husar mechanikus szerkezetének megalkotásában.

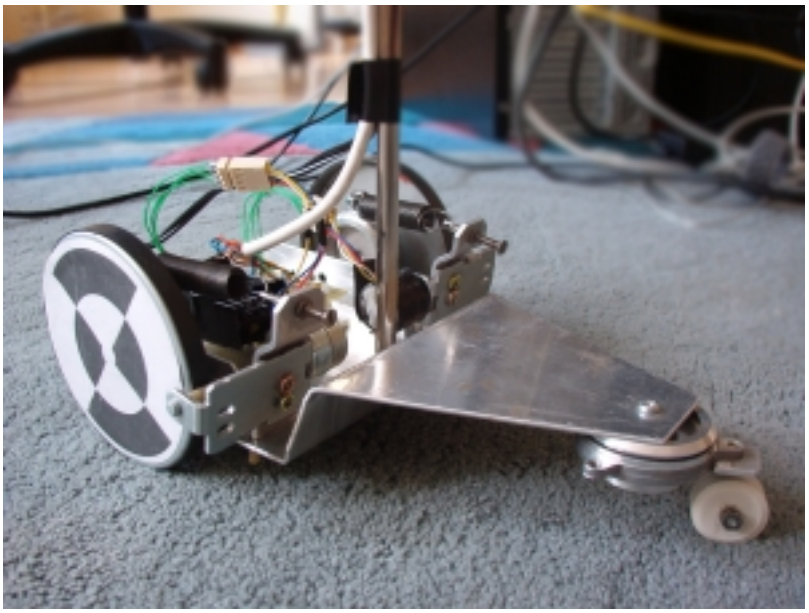
nyújtott segítségért.

I. fejezet

Bevezetés

I.1. Távirányítható Rover

A konkrét feladat egy segéd - rover Internetes irányításának megvalósítása volt, melyre a továbbiakban NET-Husarként (*I.1. ábra*) hivatkozom. Az űrtechnikának egy fontos kérdése a távolról irányíthatóság és az abból eredő problémák megoldása. Egy ilyen feladat modellezése már magában foglalja, például az adatátviteli kommunikációs közeg kérdéskörét, a meglévő technológiák összekapcsolásának problémáját is.



I.1. ábra. A NET-Husar összeszerelt állapotban

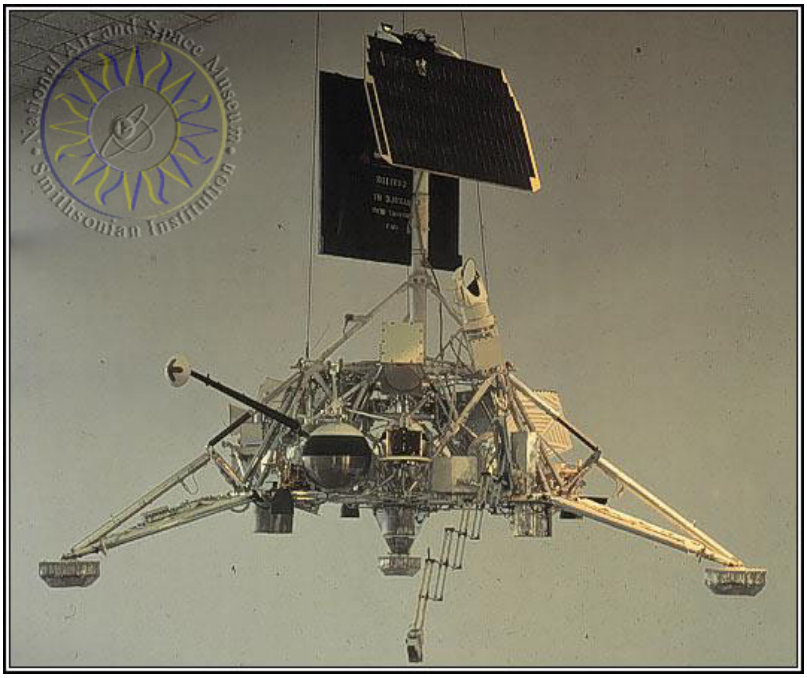
Pedagógus szemmel nézve az elvégzett feladatot egy káposzta leveleihez hasonló technológiai egymásra épülés vehető észre. Minden kis részfeladat egy nagy egészhez

járul hozzá, amely a végén egy komplex rendszert alkot a maga pozitívumaival és hibáival együtt.

A második fejezetben a NET-Husar fizikai felépítésével, meghajtás, vezérlés, elektronika elméletével ismerkedhetünk meg részletesen. A harmadik fejezetben azokról a szoftverekről, szoftverkomponensekről, lesz szó, amiket a fejlesztéshez felhasználtam, utána a rover *C* nyelven készült programját nézzük részletesen. Negyedik fejezet a tananyag telepítés nélkül kipróbálható fejlesztés alapját adó Live-DVD-ről, és használatáról, ötödik rész az Internetes irányításról a vezérlés közben lezajló kommunikációs folyamatról szól. A hatodik fejezet a NET-Husar oktatási szerepéről, abban való felhasználás módjáról, továbbfejlesztési célokról ad tájékoztatást. A hetedik rész a függelék, melyben forrásprogramok, kapcsolási rajzok és hibaelhárítási útmutató találhatók.

I.2. Hunveyor projekt

Az egyetem számítástechnika-technika tanári szakára kerültem 2004-ben. Negyedévesen ismerkedtem meg közelebbről a HUNVEYOR projekttel (A Husar-2b felújításakor). Az ELTE TTK Kozmikus Anyagokat Vizsgáló Űrkutató Csoportjánál (KAVÜCS) indult 1997-ben. Egy kísérleti és gyakorló jellegű űrszondamodell építéséhez fogtak, melyhez a NASA egy 1960-as években használt űrszondája, a Surveyor3 (*I.2. ábra*) felépítése szolgált mintául. Az űrszondamodell építésével a céljuk az, hogy egy összetett kísér-



I.2. ábra. Surveyor3 a Smithsonian Institution, National Air and Space Museum kiállításán

leti berendezés építésén keresztül olyan tapasztalatokra tegyenek szert, amelyek a természettudományos tárgyak oktatásában sokoldalúan használhatók. Az űrszonda modell a Hunveyor (Hungarian university Surveyor) nevet kapta.
A Surveyor3 sikerének titka az egyszerű felépítés, a jól megtervezett energetika, és elektronika volt. A Surveyor3 űrkísérletek elemzése, és tanulmányozása után a robotépítés először az ELTE TTK Általános Technika Tanszékén kezdődött meg, az Általános Fizika Tanszéken folytatódott, majd bekapcsolódott a Pécsi Tudományegyetem Informatika és Általános Technika Tanszéke és a Budapesti Műszaki Főiskola székesfehérvári Kandó Kálmán Karának munkacsoportja is, később csatlakozott néhány középiskola is.

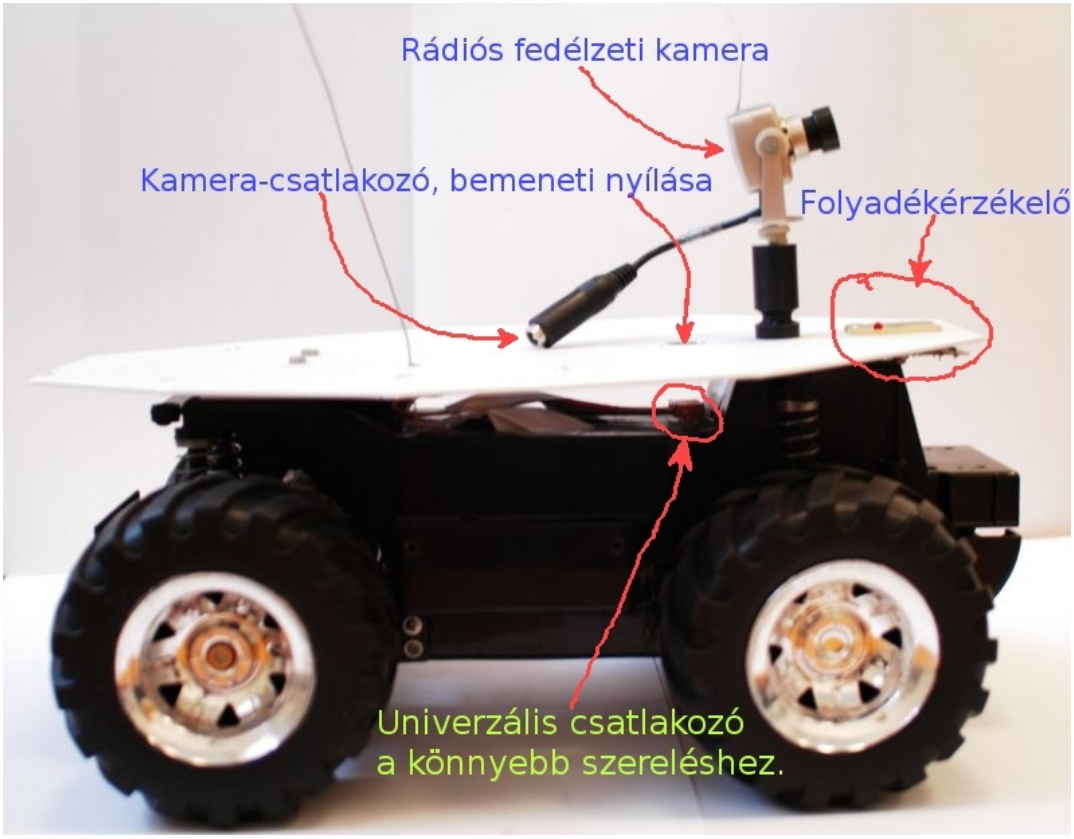
I.3. A Husar roverek

A Husar egy Hunveyor keretein belül működő önálló fejlesztési szál, mely egy mobil felfedező robot-szonda. A Husar - roverek ugyanazokat a mérési feladatokat végzik, mint az egy helyben álló Hunveyor, viszont mobilitásukból adódik, hogy sokkal jobban, és többmindenre használhatók. A mérési feladatok elvégzése után, az adatokat egy központi számítógépre küldik, melyet az kiértékel. A mobilitás kérdésével újabb problémák merülnek fel. Az általában elvárás, hogy a váratlan helyzetekkel megbírkózon, ha önállóan nem tud dönteni, vagy valami olyan probléma merül fel ami veszélyezteti a küldetés sikerét, akkor jeleznie kell a központi vezérlő egységnek, hogy az a kezelő segítségét kérje. Az önálló döntésekhez „gondolkodáshoz” valamilyen fedélzeti számítógép megléte szükséges. Erre a különböző Husar modelleknél különböző megoldások születtek.

I.4. Husar-2b felújítása

A bevezetőben említettem, hogy a Husar2-b felújítása vezetett a robotok világába. A *Mars Desert Station* programban való részvétel *(I.3.ábra)* után a munkát egyik csoporttársammal, barátommal Fülöp Balázzsal végeztük. Örültünk, hogy ilyen munkát kaptunk, hiszen darabjaira kellett szétszedni, majd az alkatrészek javítása után összerakni, amiből teljesen megismerhettük működését. Letakarítottuk a terepen összeszedett portól, megjavítottuk az eltörött napelemcellát, a meglazult webkamerának új konzolt készítettünk, majd fehér tapétával vontuk be a rá szerelt aluminium részeket. A folyadékérzékelő elektronika sem működött, azt át kellett forrasztani. Gondoltuk, újítunk is valamit, mert nehézkes a szerelése a fix elektronikai bekötések miatt, ezért egy csatlakozósort szereltünk rá. Így egy mozdulattal oldhatók az elektronikai kötések, a rá épített eszközök.

I.3. ábra. Husar-2b terepgyakorlaton, Utah (Mars Desert Station program)



I.4. ábra. Husar-2b feljavított változata

II. fejezet

NET-Husar

Az általam fejlesztett jármű, kevés, általában a Husarral szemben támasztott követelménynek felel meg, viszont jól modellezhető vele a komplex technikai rendszerek összetettsége, ami nem is a fejlesztések miatt, hanem az okatásban elfoglalandó szerepe miatt lehet fontos.

II.1. Fizikai felépítés

A NET-Husar felépítése adott volt, melyet egy előző fejlesztés alkalmával alakítottak ki, csak kisebb, a beállításokhoz és pontosabb működéshez szükséges átalakításokat kellett elvégezni rajta. Az alvázon (*II.1. ábra*) kívül, ami alumínium lemezből készült, az építőelemeket részben rossz, számunkra még használható számítógépelemekből nyertük. A kerekek meghajtáshoz, 5,25”-os floppy egységből az olvasófej mozgatását végző léptetőmotorokat használtunk fel.

Meg kell magyarázzam, miért éppen ezt a motortípust használtuk fel. A DC motorok jellemzője az, hogy elég nagy fordulaton működnek és csak áttétellel (fogaskerekekkel) lehet beállítani a kívánt fordulat/perc értéket, amit igen nehéz legyártani. Ha előregyártott áttételt választunk, azon utólag nem lehet egykönnyen módosítani. A léptetőmotorok ezzel szemben könnyen szabályozhatók, ami egy Internetes távirányítás megvalósításakor fontos szempont az idő-késések miatt. Ezért döntöttünk a léptetőmotor felhasználása mellett.

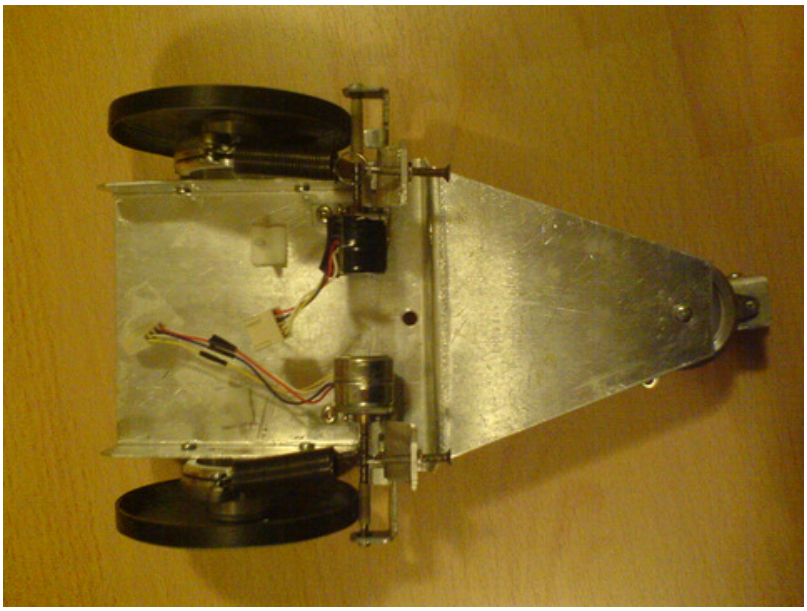
Lecserélt hardverelemek

A NET-Husaron lévő motorok meghajtását, az Országos Pedagógiai Intézet megbízásából, az ELTE Általános Technika Tanszék által (1980-as évek vége) fejlesztett Technomir egységek végezték. Ez a számítógéppel egy ISA csatolófelületű kártyán keresztül kommunikál. Ilyen csatlakozót a ’90-es évek közepe óta csak elvétve találni mű-

ködő alaplapokon, újakon egyáltalán nincs. Tehát az ISA csatlakozó és a Technomir egységek beszerezhetetlensége azonnal felvetette a vezérlés és a NET-Husarral való kommunikációs csatorna teljes átgondolását, és cseréjét.

Mire is cseréltük?

A döntés nem volt egyszerű. Mivel ma egyre nagyobb szerepet kap az USB[1], ezért elgondolkodtunk rajta, de viszonylagos bonyolultsága miatt, végül a nyomtatóport mellett döntöttünk. Előnye, hogy régi gépek majd 100%-án megtalálható. Az új alaplapokra is felkerül, de már nem mindegyikre. Az programozáshoz választott C nyelvben is megtalálható a port programozásához szükséges függvénytárak (header állományok).



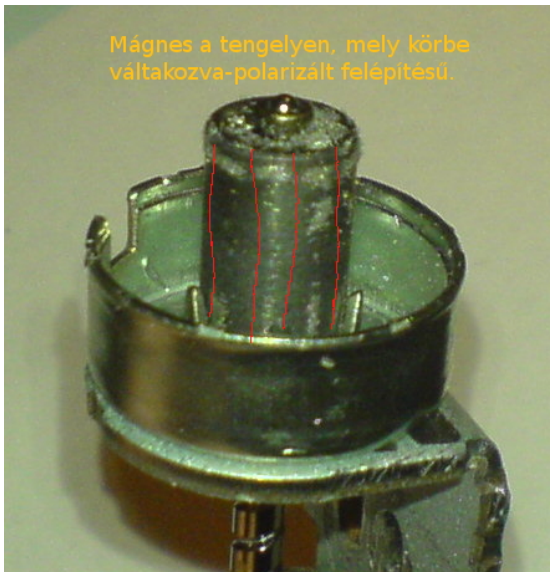
II.1. ábra. A NET-Husar összeszerelt állapotban

II.2. Meghajtás, léptetőmotorok működése

Ez a motortípus speciális feladatok ellátására készült, mint például a floppy, CD és a nyomtató-fej mozgatása, ahol az olvasó és írófejek pontos, precíz mozgására van szükség. Nagy teljesítményű léptetőmotorokat alkalmaznak az iparban is. Pl.: CNC-maró gépeknél. Nincs a motorban szénkefe, ezért az élettartamát leginkább a tengely csapágyainak minősége határozza meg. Digitális jelekkel vezérelhetők, melyeket egy integrált áramkör segítségével valósítanak meg legtöbbször. Felépítésük olyan, hogy minden impulzusra egyet lép vezérléstől függően egyik irányba, majd ott megáll és megtartja

[1] Universal Serial Bus - Egy szabványosított interfész, amelynek célja az egyszerű kommunikáció megoldása különböző eszközök és a számítógép között.

helyzetét. Ezt hívják nyílt hurkú szabályozásnak, mert nem kell semmilyen más eszköz ahhoz, hogy tudjuk, hogy a motor „hol áll”. A pontos működés feltétele az, hogy megfelelő feszültséget kapcsoljunk a megfelelő tekercsekre megfelelő sorrendben. A lépésköz attól függ, hogy hány tekercses a motor, és a forgó tengelyen levő mágnes hány része van „osztva” mágnesesen. A *II.2*. ábrán látható a mágnes kialakítása. A képen piros vonalakkal jelöltem a polaritás határait. Ebben az esetben egy körbefordulás alatt, a polaritás tízszer váltakozik. Ebből adódóan annyi impulzust kell adni a motornak egy fordulat megtételéhez. Ez megadja a Lépésszám / Fordulatot[2]. Negatívum az, hogy van egy maxi-



II.2. ábra. A polarizált tengely-mágnes

mális fordulatszáma, ami fölött nem képes teljesíteni, azért, mert már nem tudja követni a mágneses tér változásának gyorsaságát. Ebből adódik, hogy a magasabb fordulatszám nyomatékvesztéssel is jár. A motor tartónyomatéka[3] is fontos jellemző, hiszen ha áram alatt van a motor, akkor megtartja a hozzákapcsolt kerék, így a kocsi állapotát a lejtőn is.

Meg kell említenem, hogy a léptetőmotorok nem csak egész lépéseket tudnak megtenni, hanem fél lépéseket is. A programozás oldalát érinti jobban ha fél lépésekkel akarjuk megtenni a teljes fordulatot. A négy helyett hat bitet kellene meghatározott sorrendben megfelelő állapotba billenteni, ami bonyolultabbá tette volna a program írását, másrészt az elektronikát is ki kellene egészíteni, ha a későbbiekben szerelnénk a roverre bármilyen alkatrészt amit külön irányíthatóvá kell tenni.
A fejlesztés első fázisában megelégedtünk egy egyszerű, de stabilan működő elektronikai megoldással, amit természetesen tovább lehet fejleszteni a kívánalmaknak megfelelően.

[2] Lépésszám / Fordulat - A lépések száma a teljes fordulat alatt (360°)
[3] Tartónyomaték - álló helyzetből ekkora erőt kell kifejteni ahhoz, hogy kimozdítsuk a motortengelyt.

II.3. Vezérlés

II.3.1. Digitális jelek

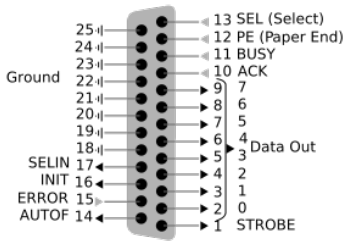
A számítógép logikai áramkörökből épül fel, amit TTL[4]-nek hívunk. Ezek az áramkörök és a hozzá kapcsolt eszközök elektromos jelek segítségével kommunikálnak egymás között, amelyek teljesítményt nem, de információt hordoznak. Ezek a digitális jelek. Tulajdonságuk, hogy két állapotuk van, a NULLA és az EGY az ami az információt hordozzák. Említettem az elektromos jeleket, amit össze kell kötnünk a feszültség-szintekkel, hogy érthető legyen. A standard feszültségszintek és a hozzá kapcsolódó logikai szintek a *II.1 táblázatban* láthatók. Ezekre a szintekre tudjuk majd a programunkkal állítani a számítógép LPT portjának kivezetéseit, amire majd rákötjük a motorokat vezérlő elektronikát.

Jelölés, logikai szint	Jelölés
L, 0, hamis	0 - 1,8V
H, 1, igaz	2,2 - 5V

II.1. táblázat. A logikai szintekhez tartozó feszültségszintek

II.3.2. A párhuzmos port (LPT)

A párhuzamos, másik nevén LPT-port[5] az a kommunikációs csatorna, amin folyamatosan elérjük majd a Roverünket. Ennek a portnak vannak ki és bemenetei egyaránt amelyekből fel fogunk használni párat. A *II.3. ábra* mutatja az LPT lábkiosztását. Az adat-kimeneti lábakat, 2-9, és a föld-lábakat 18-25 használjuk fel.



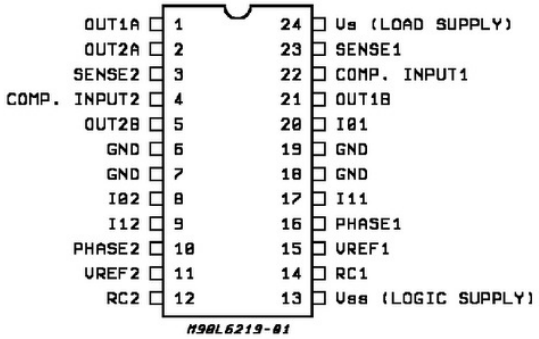
II.3. ábra. Az LPT port lábkiosztása

[4] *TTL* - Transistor-Transistor Logic, tranzisztorokból felépülő logikai áramkörök.

[5] *LPT-port* - Line Print Terminal. Magyarul párhuzamos port vagy nyomtatóport. A nyomtatókkal való párhuzamos kommunikációra fejlesztették ki.

II.3.3. A léptetőmotorokat meghajtó IC

Az általunk választott integrált áramkör STMicroelectronics ® gyártmányú, és az L6219 típusszámot viseli. Egy léptetőmotort képes meghajtani, a két tekercses fajtából, ezért majd kettőt használunk fel belőle. A *II.4. ábra* a 24 lábú IC rajza, amelyen láthatók a lábak funkciói. Meg lehet választani a motorra adott feszültség nagyságát is. Esetünk-



II.4. ábra. Az *L6219* lábkiosztása

ben az áramkör tápfeszültsége elegendő, ami 5V-t. Nagyobb teljesítményű motornál, vagy nagyobb nyomaték elérése érdekében több feszültséget is adhatunk, de vigyázzunk, az IC tápfeszültsége továbbra is 5V legyen.

II.3.4. A vezérléséhez szükséges tudnivalók

Logikai bemenetek - (I_0, I_1)

Az I_0 és I_1, a motorra kapcsolt feszültség szintjének szabályozására használható logikai bemenetek. A *II.2 táblázat* jelzi az I_0 és I_1 lábakon megjelenő jelszintekhez és azok kombinációjához tartozó kimeneti feszültség szintjét. Látható, ha mindkét lábat azonosan

I_0	I_1	**Motorra adott feszültség**
H	*H*	0 V_S
L	*H*	1/3 V_S
H	*L*	2/3 V_S
L	*L*	Maximális V_S

II.2. táblázat. Feszültség szabályozása I_0 és I_1 logikai szintjével

L szintre húzzuk, a motorokon eső feszültség maximális lesz, viszont ha mindkét lábra *„H”* jelszintet kapcsolunk, akkor a motor tekercsein nem folyik áram. Választhatunk $1/3V_S$, illetve $2/3V_S$ feszültséget is attól függően, hogy a V_S bemenetre mekkora feszültséget kapcsoltunk eredendően. Ezt azért fontos átgondolni, mert a léptetőmotorok nem viselik el ha huzamosan feszülség alatt tartjuk. Ha több mint 5V-tal szeretnénk meghajtani egy ilyen FLOPPY-motort, akkor gondoskodnunk kell programozáskor arról, hogy a

végrehajtott léptetések után a feszültséget csökkentsük a portra adott megfelelő bitmintával.

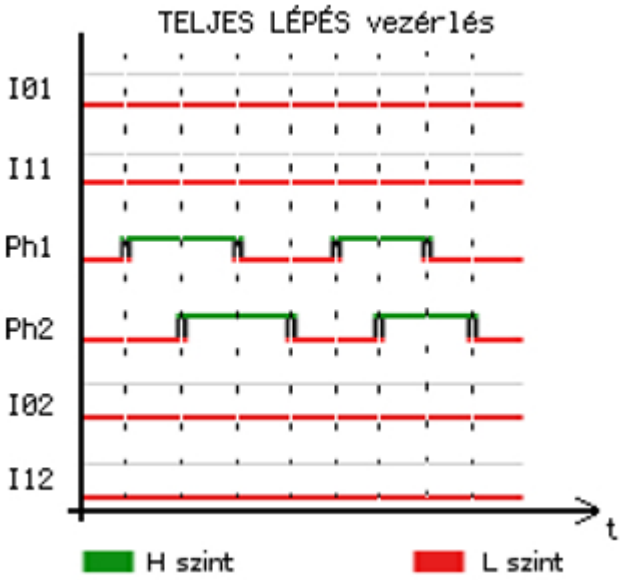
Például:
Tegyük fel, hogy $V_S = 5V$. Ahhoz, hogy a motorra $1/3\ V_S$, azaz $1{,}66V$ feszültség jusson, az

$$I_0 = L$$
$$I_1 = H$$

szinten kell legyen. A mi áramköri megvalósításunkban az I_0, I_1 össze van kötve, így egy vezérlőbittel kapcsolhatjuk *be* és *ki* a motort. Ezt a bitet mindig „L" szintre húzva, a motorra kapcsolt feszültség szintje állandóan maximális, tehát $V_S = 5V$. Ezt huzamosan melegedés nélkül is kibírja a motor, ráadásul így a lejtőn sem gurul vissza a járművünk.

Fázis lábak - (Ph_1, Ph_2)

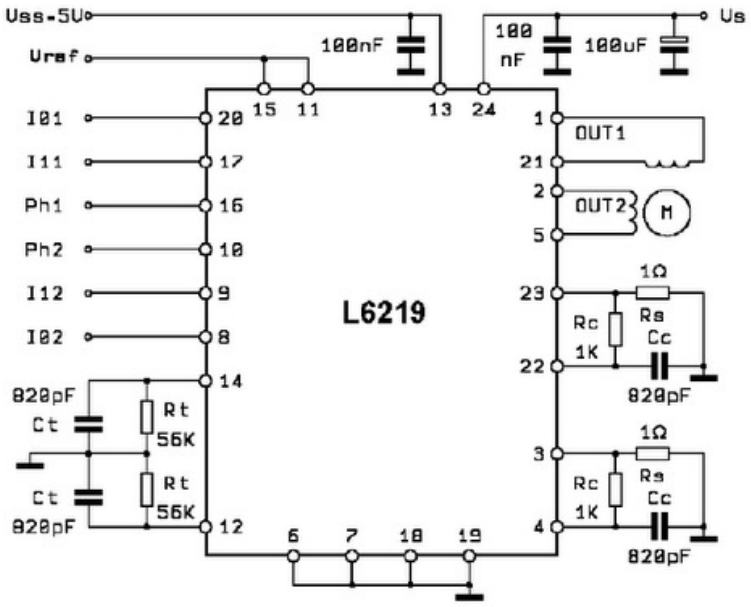
Ezek a bemenetek határozzák meg a motor tekercsein folyó áram irányát. A bemenetre adott bitsorozattal határozhatjuk meg a motor forgásának irányát.
A *II.5 ábrán* látható, hogy a *PHASE2*(IC-10.láb),és a *PHASE1*(IC-16.láb) lábakat milyen sorrendben és milyen állapotra kell hozni ahhoz, hogy a motor forogjon egyik irányba. A másik irányú forgatáshoz a portra adott jelsorrendet meg kell fordítani.



II.5. ábra. Egy teljes lépés megtételéhez az IC Ph_1 és Ph_2 lábaira kiadandó szintek és időbeli eltérésük egymáshoz képest

II.4. Áramkör, kapcsolási rajz

Az IC-hez letölthető prospektusban található az ajánlott kapcsolás. *II.6. ábra*, amitől el lehet térni, több esetben mi is ezt tettük. Az áramkört 5V feszültségről üzemeltetjük.



II.6. ábra. Az *L6219*-hez ajánlott kapcsolási rajz

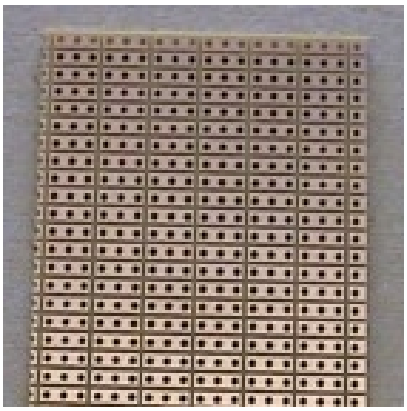
A transzformátort bármelyik elektronikai boltban megvehetjük. Mivel a transzformátor kimeneti feszültségét általában nem tudjuk beállítani 5V-ra (IC tápfeszültsége), ezért elő kell állítanunk. A feladatra megfelel egy *7805* típusjelű három lábú stabilizátor IC. Feladata a megfelelő feszültség előállításán kívül, hogy a hálózati feszültség ingadozásait kompenzálja. Egy viszonylag egyenletes feszültségszintet kapunk, ami fontos a digitális jelekkel működő áramkörök stabilitásához. Ha még azt is szeretnénk tudni, hogy be van-e kapcsolva az áramkör, akkor egy LED[6], és egy előtét ellenállás kell csak hozzá. Az ajánlott kapcsolást (*II.6. ábra*) kiegészítjük ezekkel az alkatrészekkel. A bővített kapcsolási rajz a Függelék *VIII.4. ábráján*, a felhasznált alkatrészek jegyzéke pedig a Függelék *VIII.1. táblázatában* látható.

II.4.1. Próba-áramkör építése

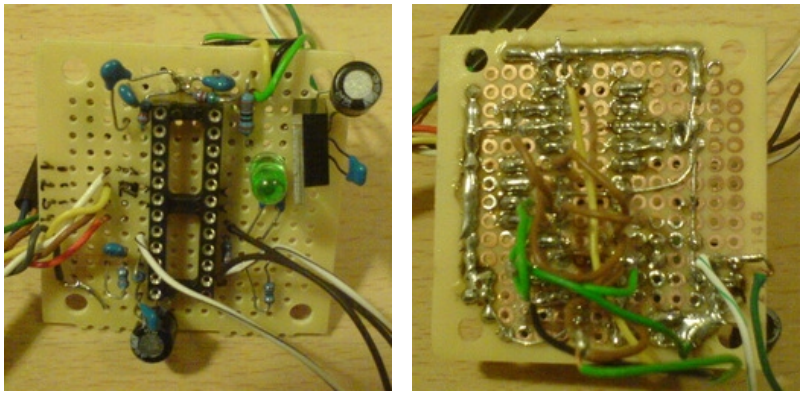
Minden fejlesztés előbb elméletben valósul meg, majd mikor úgy tűnik, hogy minden rendben, akkor kezdődhet a megvalósítás. Az első dolog, hogy megépítjük az elektronikát egy próbapanelre *(II.7. ábra)*, amely nagyon nem hasonlít a végleges változathoz, de működő áramkört kapunk. Ezzel meggyőződhetünk arról, hogy a tervünk jó, vagy kell-e rajta módosítani. Ha kell, akkor gyorsan tudjuk akármelyik alkatrészt cserélni, akár

[6] *LED* - Nevét a Light Emitting Diode angol elnevezés rövidítéséből kapta. Felfedezője Rubin Braunstein.(1955)

az áramkört is megváltoztathatjuk különösebb probléma nélkül. Ehhez szükséges néhány különleges szerszám, melyek nélkül csak bosszúság kíséri munkánkat. Ilyen az önelszívó, vagy önszippantó, amivel a folyékonnyá vált ónt felszívhatjuk a NYÁK-ról, vagy a beforrasztott alkatrész lábáról, így kivehető az alkatrész rángatás, tépés nélkül is. Összeszerelés után a próbapanel két oldala látható a *II.8. ábrán.*



II.7. ábra. Üres próbapanel



(a) Alkatrész oldal (b) NYÁK-oldal

II.8. ábra. Próbapanel beültetés után

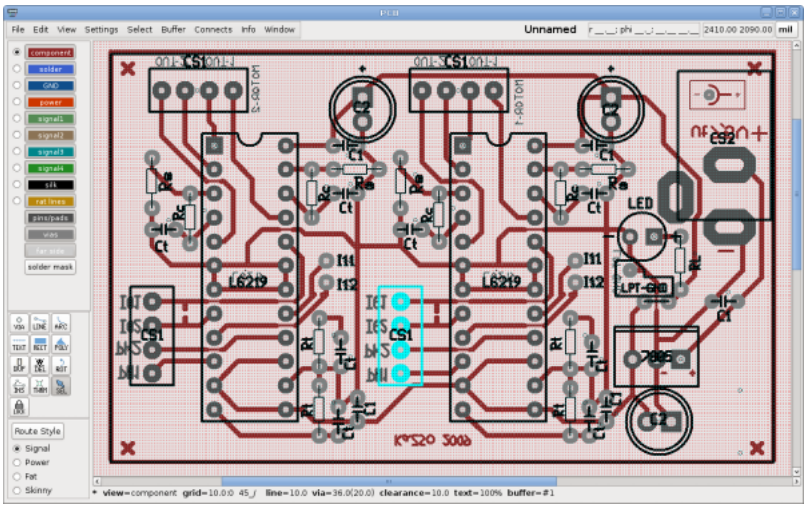
Miután a próbapanel működik, megtervezzük a PCB programmal. Azért, hogy nézzen ki valahogy, másrészt ha javítani kell, nem hiszem, hogy valaki képes lenne megmondani, hogy pl.: egy leszakadt vezeték hova volt beforrasztva.

II.5. Nyomtatott áramkör (NYÁK)

A jó és könnyen reprodukálható munkához hozzátartozik az elekronikus áramkörök számítógépes tervezése is. Ennek előnyeit nem kell hangsúlyoznom, hiszen aki már készített valamilyen tervet vagy dokumentumot, tudja mennyire könnyen lehet utólag módosítani.

II.5.1. Tervezés

Ebben a folyamatban az operációs rendszer, Debian GNU/Linux készítője által karbantartott, tehát „gyári” **PCB** programja segített. (Felhasznált szoftverek: *VIII.3 lista.*) Nem a legbarátságosabb nyomtatott áramkör tervező, de el lehet vele boldogulni ilyen nehézségű feladat esetén. Tervezéskor készült képernyőkép látható a (*II.9. ábrán*). Minden igyekvő meg tudja tanulni magától a program működését, sokat segíthet a haladásban az Interneten elérhető rengeteg leírás. Miután ezen túl vagyunk, következhet a NYÁK kivitelezése. Kétféle módszert ismertetek, mindkettő pénztárcakímélő, de az első, a sza-



II.9. ábra. Az *PCB* munka közben

badkézi megvalósítás csak azoknak ajánlott, akik nem először fognak bele ilyen munkába. Az alábbi bekezdésekben csak röviden pontokba szedem a fontosabb lépéseket, mert az elektronikus tananyagban részletesen kitérek mindenre.

II.5.2. Szabadkézi kivitelezés

RÉSZLETES TANANYAG A DVD-MELLÉKLETEN:

⇒ `Tananyag/Tananyag-Szabadkezi.pdf` ⇐

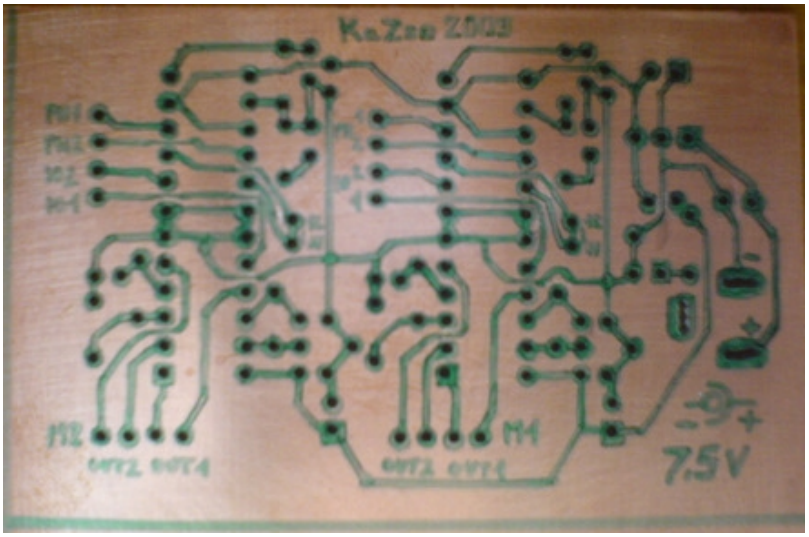
Lépések:

- A mellékelt Tananyag-DVD-ről nyissuk meg az alábbi dokumentumot, ami a tervezett NYÁK tükörképe. Ezt kell rárajzolni a réz felületre.

  `/NYAK/NET-Husar_szabadkezi.pdf`

- Nyomtassuk ki, mindegy milyen nyomtatóval.
- Vágjuk ki a rajzot a keret mentén, a kerettől kb. 1cm-re, majd ragasszuk fel átlátszó ragasztószalaggal a rézzel bevont oldalra úgy, hogy a rjaz felül legyen.
- Vegyünk egy pontozót és a furatok helyét vigyázva, pontozzuk meg.
- Ha biztosan minden pontot megpontoztunk, vegyük le a rajzot a NYÁK-ról.
- Neki is láthatunk egy lakkfilccel, a nyomtatott áramkör szabadkézi felrajzolásának.

Ez a régi, jól bevált módszer a legolcsóbb, de a leglassabb is. Ha készen van (*II.10. ábra*), kezdődhet a marási folyamat.



II.10. ábra. Kész állapotú szabadkézi NYÁK

II.5.3. Vasalós módszer

RÉSZLETES TANANYAG A DVD-MELLÉKLETEN:

⇒ Tananyag/Tananyag-Vasalos.pdf ⇐

Lépések:

- A mellékelt NET-Husar DVD-ről nyissuk meg a dokumentumot:

  NYAK/NET-Husar_vasalos.pdf

- Papírboltban kapható *„műnyomó”* papírra, nyomtassuk ki lézernyomtatóval. Ha nincs elérhető közelségben ilyen, akkor a bármilyen nyomtatóval kinyomott oldalt *fénymásoljuk* rá a műnyomó papírra.Hangsúlyozom, csak lézertechnológiás fényfásolóval lehet, multifunkciós tintasugaras nyomtatóval nem.

- Kivágás után ragasszuk a gondosan megtisztított NYÁK-ra úgy, hogy a nyomtatott felület legyen a rézzel bevont oldalon.

- Vasalóval égessük rá a papíron levő festéket a NYÁK-ra.

- Várjuk meg, míg kihűl.

- Tegyük bele vízbe és hagyjuk ázni, majd tíz perc után, mikor a papír teljesen átázott, és érezhetően megpuhult, óvatosan húzzuk le róla. Nem baj, ha maradnak rajta apró papírfoszlányok, viszont a nagyobb darabokat dörzsöljülk le a kezünnkel, mert a papír műanyagbevonata gátolja a megfelelő maratást.

Ha jól sikerült a festékanyag átvitele a NYÁK-ra, nem úgy mint ahogy a *II.11. ábrán*, akkor kezdődhet a maratás.

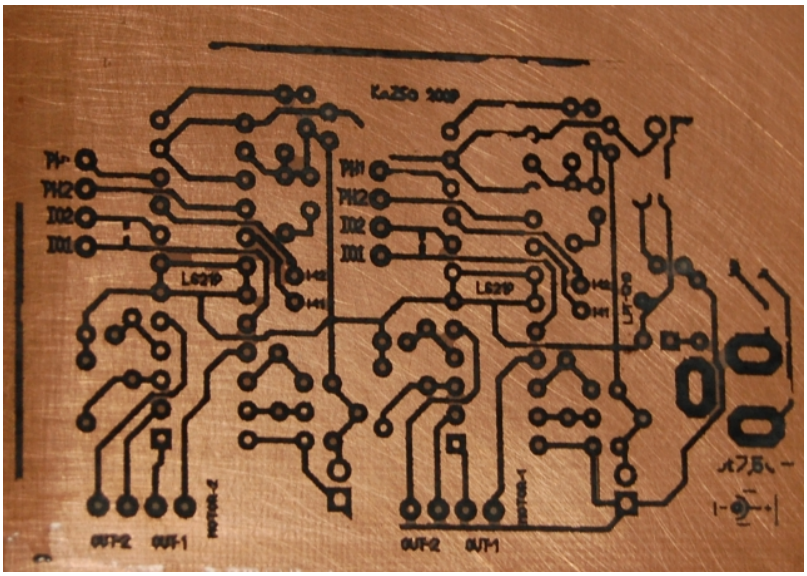
Összegzés

Mindkét módszernek a lényege az, hogy a rézfelületre vitt anyag ellenálló a maróanyag összetételével szemben. Első esetben a lakkfilc által hagyott lakkréteg, második esetben a lézernyomtató vagy fénymásoló festékanyaga. Mindkét eljárással egészen jó minőségű NYÁK-ot lehet készíteni, de nem szabad túl picire tervezni a rajzot, nehogy rajzoláskor a lakkfilc hegyének vastagsága legyen ami miatt újra kell tervezzük az egészet.

II.5.4. Maratás

RÉSZLETES TANANYAG A DVD-MELLÉKLETEN:

⇒ Tananyag/Maratas.pdf ⇐



II.11. ábra. Rosszul sikerült a festékátvitel a papírról. Jól kivehetőek a hibák.

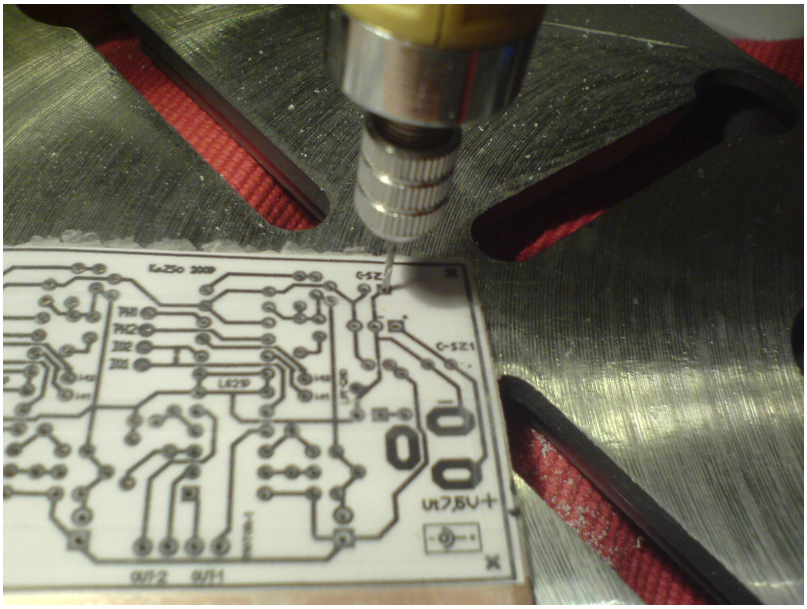
Miután a nyomtatott áramköri rajzot NYÁK-fóliára valamilyen módszerrel átvittük, és meggyőződtünk arról, hogy minden csík, pont a megfelelő helyre került, és nem érnek össze a megrajzolt vezetékek, nekiállhatunk a maratásnak. Lehet kapni a legtöbb mikroelektronikai alkatrészekkel foglalkozó boltban *TN 150 gyorsmaratót*[7] folyadékot. amely a fölösleges rézfelületet eltávolítja, de meghagyja a felrajzolt árakört (*II.12. ábra*). A művelethez öntsünk a szerből egy műanyag, vag üvegtálba annyit, hogy a NYÁK-ot ellepje. A marás gyorsasága a maratószer hőmérsékletétől is függ, ami optimális esetben 40°C kell legyen. Az ilyen meleg maratószer gyorsítja a folyamatot. Ha hideg folyadékkal maratunk akkor is gyorsíthatjuk a folyamatot, a tál lassú jobb, majd bal oldalának emelgetésével, a folyadék *„lötyögtetésével”*.

II.5.5. Fúrás

Az áramköri rajzunkban nincs hiba, eldöntöttük, hogy megfelel, akkor az alkatrészek beültetése előtt még a forrpontok közepébe lyukat kell fúrni. Ehhez a művelethez használjunk ø$0{,}5mm$ fúróhegyet. Az ilyen kis fúrószárat nem lehet bármilyen tokmányba befogni, ezért, ha nincs más választásunk, akkor egy kis akkumulátoros kézifúró is megteszi. Vigyázzunk, hogy a fúróhegy legyen éles, ellenkező esetben felgyűrheti a forrpont maradék rézbázisát, így nem lesz mihez forrasszuk az alkatrész lábát. A műveletet a rézoldalról hajtsuk végre. (Felfelé nézzen az áramkör.)

[7] *TN 150* - Vas-klorid alapú, gyorsított, és alámaródást csökkentő hatású maratószer, nyomtatott áramköri lapok és egyéb rézfelületek maratásához.

II.12. ábra. Maratás



II.13. ábra. Fúrási művelet

II.5.6. Beültetés

RÉSZLETES TANANYAG A DVD-MELLÉKLETEN:

⇒ Tananyag/Beultetes.pdf ⇐

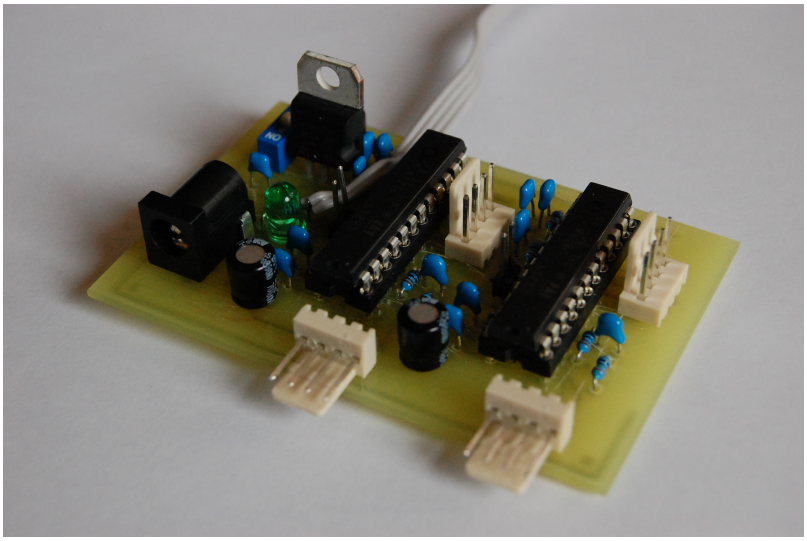
Ahhoz, hogy viszonyalag gyorsan és mérgelődés nélkül teljen az alkatrészek beültetésére szánt idő, bizonyos szabályokat be kell tartani. Az alábbi pontokba szedett tanácsokat fogadjuk meg, és hajtsuk végre szép sorban.

- Minden alkatrész lábát műszerészcsipesszel többször húzzuk végig, így eltávolítjuk az oxidációt, a forrasztás biztosabb, a rossz forrpontok miatt nem kell aggódnunk.

- Az alkatrészeket válogassuk szét, hogy könnyebben megtalálhassuk, mit, hova kell beültetni.
- Legyen előttünk nyomtatott, vagy elektronikus formában a beültetési rajz és alkatrészjegyzék. Javaslom a nyomtatott formát, mert tudunk rá jegyzetelni.

Ha ezekkel megvagyunk, akkor kezdődhet a beültetési eljárás. Célszerű figyelembe venni az alábbi tanácsokat a munka végrahajtásakor.

- Nézzük végig az alkatrészeket, és azzal kezdjük a beültetést, amelyik a legkevésbé áll ki a NYÁK felületéről. Általában a következő sorrendet tartjuk, ha nem SMD[8] alkatrészekkel dolgozunk:
  1. átkötések, ellenállások, diódák,
  2. IC-foglalatok, DIP-kapcsolók (mini kapcsolók), jumperek, kerámiakondenzátorok,
  3. elektrolit-kondenzátorok, LED-ek,
  4. csatlakozók, álló tokozású félvezetők (7805)
- Ha van folyasztószerünk, akkor kenjük be vele a NYÁK-ot, hogy a forrasztóón könnyen „befussa” a forrpontokat. Száradás után beültethetünk.



II.14. ábra. Beültetés végeztével

[8] *SMD* - Surface Mounted Device, ami azt jelenti, hogy az alkatrészeket a nyomtatott áramkör vezetékezett oldalára „ültetik’ rá, nem kell minden alkatrész lábánál megfúrni a panelt.

III. fejezet

Szoftverek

III.1. A NET-Husar egészének felülvizsgálata

A NET-Husar elődje amely ugyanezt a célt valósította meg, hogy irányítható legyen az Internetről, sok oldalon elavult. Egy rövid vizsgálat után úgy döntöttem, korszerűsítek, mivel azon formában szinte reprodukálhatatlan lett volna. A probléma mind hardveres, mind szoftveres oldalon fennállt.

A megvalósítás alapja egy Windows98® rendszer volt, amely elavultnak számít. A programozási feladatokat Borland Delphi®-ben készítették amihez egy saját *dll* függvénytárat is írtak, ami egyáltalán nem biztos, hogy kompatibilis lenne az újabb fejlesztésű Microsoft® rendszerek bármelyikével. Másrészt nem vagyok híve a kereskedelmi szoftvereknek, főleg nem akkor, ha operációs rendszerről van szó.

Mivel célom egy kis költségvetésű, könnyen reprodukálható rendszer megépítése volt, nem is gondolkodtam olyan szoftverekben, melyek nem szabadon felhasználhatók. Kellett keresni egy olyan rendszert amiben adottak a feltételek, ne kelljen külön forrásból összeszedni a feladathoz szükséges elemeket, majd azokat összegyúrni egy működő egésszé. A szerverek világában előszeretettel használt Linux terjesztések elérhetők bárhol a világon, ingyenesek, folyamatosan fejlesztik és a legtöbb terjesztés minden elemet tartalmaz, amire szükségünk van.

III.1.1. Ajánlott szoftverek, szoftverkomponensek

Természetesen a választott Linux disztribúciónak (terjesztésnek) az alábbi kritériumok minden alpontjának legalább egy komponensét tartalmaznia kell.

Követelmények részletesen. (Ajánlottak aláhúzva szerepelnek a listában.)

- **C-fordító**
  - gcc
- **Webszerver** CGI-szkript támogatással
  - Apache, Apache2
  - thttpd
- **Kameraszerver és képrögzítő**
  - motion
    (http://www.lavrsen.dk/foswiki/bin/view/Motion/WebHome)
- **Szövegszerkesztő** (Nem „WYSIWYG”[1] szövegszerkesztőre lesz szükség.)
  - vi, vim, gvim
  - nano
  - mcedit, az mc része.
  - Bluefish - Egy nagyon nagy tudású szövegszerkesztő, programozáshoz. (grafikus)
  - 'Gedit - Egy kicsi és kis tudású szövegszerkesztő a GNOME munkaasztalhoz. (grafikus)
- **Böngésző** java-script támogatással
  - Firefox, Iceweasel[2]
    (http://www.mozilla-europe.org/hu/firefox/)
  - Epiphany
  - Konqueror
  - Chimera
  - Conqueror

Nem kell sokáig válogatnunk, mert a legtöbb terjesztés eleget tesz a követelményeknek. Mivel a Debian GNU/Linux az általam leginkább kedvelt, az ezen alapuló Ubuntu nevű terjesztésre esett a választás, melyet biztonság, stabilitás, gyorsaság, széleskörű szoftver és hardware-támogatás jellemez. (A dolgozat készítéséhez használt programok a *Függelék, VIII.3. listájában láthatók.*)

[1] Szövegszerkesztésben a „What You See Is What You Get” = „azt kapja amit lát” elvet követő szerkesztő. Pl.: OpenOffice.org Witer programja.

[2] *Iceweasel v3.0.6* - a Debian GNU/Linux saját fejlesztésű Firefox alapú böngészője.

III.2. Programozás

Ha az elektronikai megvalósítási feladat végére értünk, akkor még semmi sem biztos, hiszen ki kellene próbálni, működik-e. Addig amíg nincs egy vezérlőprogramunk, otthoni körülmények között nem lehet próba alá vetni az áramkört. A Vezérléséhez szükséges tudnivalók c. *(II.3.4.) fejezetben* említettem, hogy bizonyos lábakra milyen jeleket kell adni, és milyen sorrendben ahhoz, hogy a motorunk lépjen valamilyen irányba. Itt a *C* nyelven írt `motor.c` program működésével foglalkozunk (*VIII.2. fejezet*). Ez a program, a kész elektronikához van illesztve. Amíg nem biztos, hogy működik rendesen az elektronikus rész, addig javaslom, ne nyúljunk bele a kódba, mert mindkettőtől függ, hogy a motorunk mozog-e. Így legalább a program biztosan működik. Lehet probléma a nyomtató portjának címe is, de nem tapasztaltam ilyet. Amelyik gépen próbáltam, működött.

A programot egyetlen paraméterrel lehet meghívni, ami a NET-Husar haladási irányának kezdőbetűje (**e**lőre,**h**átra,**j**obbra,**b**alra), Pl.:

`/usr/sbin/motor e` Jelentése: *NET-Husar, menj előre!*

III.2.1. A program részei és magyarázatuk működési sorrenben

Paraméterek számának vizsgálata

Meg kell vizsgálnunk, hogy kaptunk-e paramétert. *C* nyelv a paraméterek számát az *argc* változóban tárolja. Mivel nekünk csak egyetlen paraméter kell, ezért a változóban 2 kell legyen. Jogos a kérdés, hogy miért? Azért, mert az első paraméter mindig a futó program neve, a második amit paraméterben megadtunk, majd a harmadik az a megadott második paraméter és így tovább. Tehát az első paraméter a C szerint mindig a program neve. Ezt a paramétertömb *0*. eleméből tudjuk kiolvasni *(argv[0])*. Tehát, ha kevesebb a paraméterek száma mint 2, akkor nem kaptunk egyetlen külső paramétert sem. Így nem tudja a program, mit akar a hívó, ezért hibaüzenet után kilépünk.

```
if (argc!=2){
  printf(" (HIBA) - A programnak meg kell adni egy parametert, \n\n \\
      Hasznalat: motor [e | h | j | b] \n\n\t \\
          e - elore\n\t \\
          h - hatra\n\t \\
          j - jobbra\n\t \\
          b - balra\n\n");
  exit(1);
}
```

A port hozzáférésének beállítása

Az *ioperm* függvény paraméterezésétől függően beállítja az őt hívó rutin hozzáférési jogát

a megadott porthoz. Ha nem tudja beállítani, akkor hibát jelez, megoldási javaslatokat tesz, majd kilép.

```
if (ioperm(port,1,1))
    fprintf(stderr, "(HIBA) – Nem erem el a portot: %x\n \\
      1.) Lehet, nincs jogom hozza? MEGOLDAS, \\
          ha futtatod ezt a parancsot:\n\n\t\t \\
            'sudo chmod +s [eleresi ut]/motor'\n\n \\
      2.) Rosszul van megadva a programban a port cime. \\
          A program elején keress egy ilyen sort: \\
            #define port 0x378, \\
            majd írd át  a '378-at' '278'-ra.\n\n
      3.) Nincs LPT port a szamitogepben.\n\n", port), exit(10);
```

A megadott paraméter vizsgálata

Attól függően, hogy a paraméter a vizsgáltak közé tartozik, meghívja a hozzá tartozó függvényt, amely *előre*, *hátra*, *jobbra*, vagy *balra* irány lehet.

```
if (strncmp ("e", argv[1], 1) == 0)
    elore();
if (strncmp ("h", argv[1], 1) == 0)
    hatra();
if (strncmp ("j", argv[1], 1) == 0)
    jobbra();
if (strncmp ("b", argv[1], 1) == 0)
    balra();
```

Egy irányfüggvény magyarázata. (Irányfüggvények: *előre*, *hátra*, *jobbra*, *balra*)
Eddig arról volt szó a vezérlés tárgyalásakor, hogy a portra küldött bitmintáktól függ az, hogy miként viselkedik a vezérőportra kötött elektronika. Fontos, hogy ez csak akkor igaz, ha a port 2-9-ig tartó tüskéit vesszük, mert ezek az adattüskék, amiken a legkönnyebb kommunikálni. Azt, hogy melyik tüskére milyen jelet kell küldjünk, azt a bekötés, *III.1. ábra* határozza meg. A megértéséhez kell a *2*-es számrendszer ismerete. Mivel a kettes számrendszerben csak a *„0”* és *„1”* értelmezettek, a bitsorozat ebből a két számból állhat. Pl.:

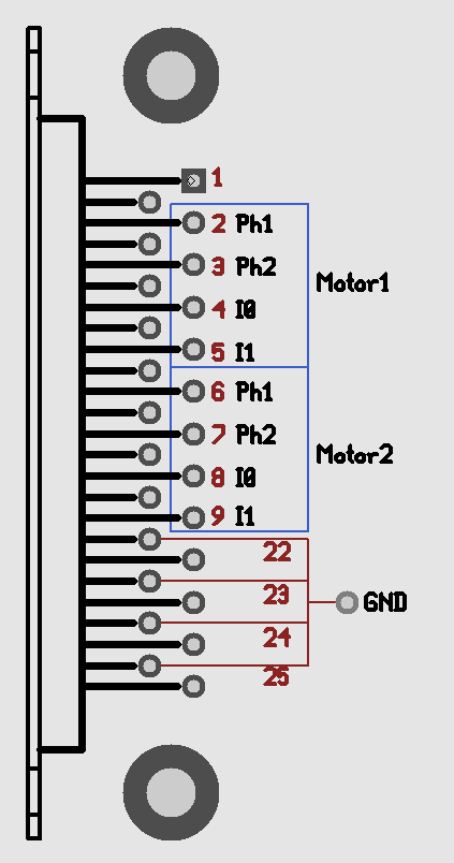
$$10100110_{(2)} = 166_{(10)}$$

A 0-ák és 1-esek bizonyos helyiértéken más-más értékkel rendelkeznek
Magyarázat:

$$2^7 + 0 + 2^5 + 0 + 0 + 2^2 + 2^1 + 0 = 166_{(10)}$$
$$128 + 0 + 32 + 0 + 0 + 4 + 2 + 0 = 166_{(10)}$$

Ha minden bit *1*, akkor az összes helyiértéken levő számot össze kell adni, így jön ki 255.

$$2^6 + 2^5 + 2^4 + 2^3 + 2^2 + 2^1 + 2^0 = 255_{(10)}$$
$$128 + 64 + 32 + 16 + 8 + 4 + 2 + 1 = 11111111_{(2)} = 255_{(10)}$$



III.1. ábra. Az áramkör kivezetéseinek bekötése a vezérlőportra

Amilyen 8 bites számot a portra írunk, annak megfelelő bináris kombinációba állnak be a kivezetések (*2-9*). Ha a kiírt számot binárissá alakítjuk, akkor azt tapasztaljuk, hogy az igaz, (TRUE) vagyis ’1’ értékű bitek közel +5V feszültségként jelentkeznek a csatlakozótüskéken, viszont ahol hamis, vagyis ’0’ érték van, ott a közel nulla voltos feszültséget mérhetünk.

Lépések istmétlése

Egy bitkombinációra az elektronika a motor tekercseit úgy gerjeszti, hogy egyet lépjen. Egy teljes körülforduláshoz *tíz* lépésre van szükség, amit egy ciklussal oldhatunk meg. Pl.: a hátramenet ciklusa 24. sortól kezdődik és a 40. sorig tart. A közte lévő sorok adják a ciklus magját, amit addig ismétel, míg a feltétel igaz.

Várakozás függvény

Ez a program legegyszerűbb, de egyben a legfontosabb része is, hiszen a portra írás sebessége olyan nagy, amit a motorok, nem tudnak követni. Ezért be kell iktatni egy várakozást ami a két jelsorozat portra küldése között a megadott időre megállítja a pro-

gramot, majd továbblép. A *var()* függvény 15-18. sorban található. A *usleep()* fügvény a megadott századmásodpercig várakozik.

Amint láttuk, nem egy bonyolult programról van szó. Akár az is megérti, a működését, akinek eddig kevés köze volt a programozáshoz. Ha jobban el szeretne valaki merülni a párhuzamos port programozásában, mert lát benne fantáziát, az Interneten sok építési és programozási leírást találhat magyarul is.

IV. fejezet

Live-CD/DVD

A mai elvárásoknak megfelelően egy ilyen tananyaghoz nem elég leírni, mit hogyan készítettem el. Ha megpróbálja valaki megépíteni és programozni a saját járművét, a megfelelő rendszer hiányában nem fogja tudni, vagy annyi időt tölt az összeállításával, hogy elmegy a kedve az egésztől.
Azért, hogy ezzel se kelljen foglalkozni készítettem egy telepítés nélkül kipróbálható, minden eszközzel felszerelt, úgynevezett *Live CD/DVD*-t. A számítógépet erről kell indítani (boot-olni), és minden automatikusan elindul (webszerver, motion). Nem kell beállítani semmit, kivéve ha a csatlakoztatott webkamerát nem látja a rendszer. Ennek orvoslására már nem tudunk figyelmet fordítani, de az Interneten nagyon sok jó leírás található, kamera típusonként eltérő megoldási javaslattal.

IV.1. Indítás

Arra kell figyelnünk, hogy mikor a bootolási folyamat elkezdődik, legyen minden csatlakozatva, gondolok itt a webkamerára. Ha csak később csatlakoztattuk, a legegyszerűbb ha újraindítjuk a gépet, mert ha a `motion` nem érzékel kamerát, azonnal leáll. Természetesen a gépkímélőbb megoldás, ha `motion`-t indítjuk újra. egy *Terminál*-t elindítva (*VIII.3. ábra*) a következő parancs kiadásával lehet újraindítani:

```
/etc/init.d/motion restart
```

Ha így sem működik, nézzünk utána, kameránk milyen felbontást támogat. A `motion` *352x288* felbontású képet kér a kamerától. Meglehet, az nem tud ilyen képeket készíteni, csak nagyobb felbontásút. Megoldási lehetőségek a *VIII.1 (Hibaelhárítás - Webkamera)* részben.

A rendszer, indulása után az *IV.1. ábrán* látható képpel fogad. Javaslom, hogy indítsunk egy böngészőt, és győződjünk meg arról, hogy a kamera működik.

IV.1. ábra. *Ubuntu Linux NET-Husar változata indulás után*

IV.2. Forráskód fordítása, futtatása

IV.2.1. Fordítás kézzel

Az egyik lehetőség, hogy mindent saját magunk végzünk lépésről lépésre. Ha nincs gyakorlatunk az ilyesmiben, akkor sokat vesződhetünk. Ha mégis ezt az utat választjuk, a következő lépéseket hajtsuk végre, miután a forrásprogramot módosítottuk. Nyissunk egy terminált (xterm, xterminál-emulátor), és adjuk ki a parancsokat:

- Váltsunk munkakönyvtárat: `cd /home/nethusar/NET-Husar/C_forras`
- Fordítsuk le a programot: `gcc $1 -o motor`
- Adjunk 's'-bitet[1] a fájlnak: `chmod +s motor`
- Másoljuk a telepítőcsomagba: `cp motor ../TELEPITO/usr/sbin`
- Másoljuk oda, ahonnan a CGI-script meghívja: `cp motor /usr/sbin`

Ha minden hibamentesen zajlott, próbáljuk ki böngészőből.

IV.2.2. Fordítás szkripttel

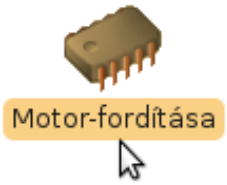
Azért, hogy könnyebb legyen a forrást módosítás után kipróbálni, használjunk egy egszerű *bash* szkriptet, ami nem csak fordítja, hanem beállítja a megfelelő jogokat, majd másolja a *TELEPITO* és az */usr/sbin* könyvtárba is a bináris állományt. Így rögtön kipróbálhatjuk böngészőből amit módosítottunk.

[1] *sticky-bit* Ha egy állomány futtatható és 'sticky-bit'-tel ruházzuk fel, futtatáskor nem a futtató nevében fut, hanem a tulajdonos nevében.

A fordító szkript elérhető az alábbi helyen és néven:

`/home/nethusar/NET-Husar/fordit.sh` (forráskód: *VIII.2.2* fejezet)

Találunk egy `Motor-forditasa` nevű linket (*IV.2*), mely a *home* könyvtárunkban van. Ha a fordítás hibaüzenet nélkül lezajlott, akkor ideje kipróbálni. Ha fordítószkriptet



IV.2. ábra. *Motor-forditasa.link*

használtunk, vagy minden kézi műveletet végrehajtottunk, lehetőségünk van a konzolos, és böngészős próbára is. Ha konzolon fordítottunk, akkor csak meg kell hívni a programot a megfelelő paraméterrel. Indítás a következő paranccsal:

```
./motor h
```

vagy

```
/usr/sbin/motor h
```

Ha nem kapunk hibaüzenetet, a motoroknak mozogni kell, működő elektronika esetén. Ha forognak a motorok, kipróbálhatjuk böngészőből is. Először csak helyi gépről próbáljuk, indítsunk egy Firefoxot, majd írjuk be a címmezőbe: *localhost*, majd üssünk ENTER-t. Ha bejön a weboldal (*V.2. ábra)*, akkor a webszerver már biztosan működik. Ha látjuk a kapmera képét, az kitűnő, majd kattintsunk valamelyik irányító nyílra. Ha konzolról működött, akkor így is működie kellene. Ha valami miatt nem megy, hajtsuk végre a konzolos utasításokat lépésről lépésre, majd próbáljuk újból.

IV.3. NET-Husar előre telepített rendszeren

Azokra is gondoltam, akik már rendelkeznek egy telepített Linux-szal. Nekik készült a Tananyagok DVD-n található TELEPITO csomag. Ez a program az alapvető programokat nem telepíti, csak a weboldalt, a motion beállítóállományát a motormozgató forrásprogramot, és fordított változatát. Mielőtt hozzáfognánk, telepítsük fel az `Apache2`, `motion`, `gcc` csomagokat. Debian GNU/Linux esetén az alábi parancs kiadásával lehet, rendszergazda terminálból:

```
apt-get install apache2 motion gcc
```

Ha ezek a csomagok települtek, át kell másolni a telepito könyvtárat a *home* könyvtárunkba.

Figyelem! Ha webszervert működtetünk, és az szolgáltat is a telepítés idején, akkor ne futtassuk a telepítőcsomagot, mert az nem figyeli, hogy van-e a /var/www könyvtárban valami, ezért rámásolja a működő weboldalra az általam készített forrást, ami nagy kavarodással, adatvesztéssel jár!

V. fejezet

Internetes irányítás

V.1. Vezérlési folyamat

Nézzük, miként kapcsolódnak egyáshoz összetett bonyolult technológiai rendszerek a kis gyakorlórobotunk irányításához. A *V.1. ábrán* végigkövethetők a rendszerek kapcsolódási pontjai és a kommunikáció folyamata kliens-gép, szerver-gép és a NET-Husar között, utána taglaljuk a minket közvetlenül érintő részeket

V.2. CGI

„CGI (azaz Common Gateway Interface) az NCSA (National Center for Supercomputing Applications) által kifejlesztett protokollszabvány, amely (például Perl, PHP vagy C nyelvű) alkalmazások információs szerverekhez - a gyakorlatban túlnyomóan webszerverekhez (például Apache vagy IIS) - való kapcsolódását teszi lehetővé. Ha a kliens kérése futtatható fájlra mutat, akkor a szerver futtatja azt és a kimenetet adja vissza a kliensnek”[6]

Ezt a techonológiát használjuk fel a NET-Husar irányítására. Ha megnyitjuk az oldalt, és rákattintunk az irányító gombokra akkor a böngésző meghívja a CGI-script-et és átad egy, a paramétert, ami minden gombnál más. A CGI-script lefut, megkapva a gombhoz rendelt paramétert, *elore, hatra, jobbra, balra lehet* megvizsgálja, majd meghívja a `/usr/sbin/motor` programot, ami a motorokat forgatja.

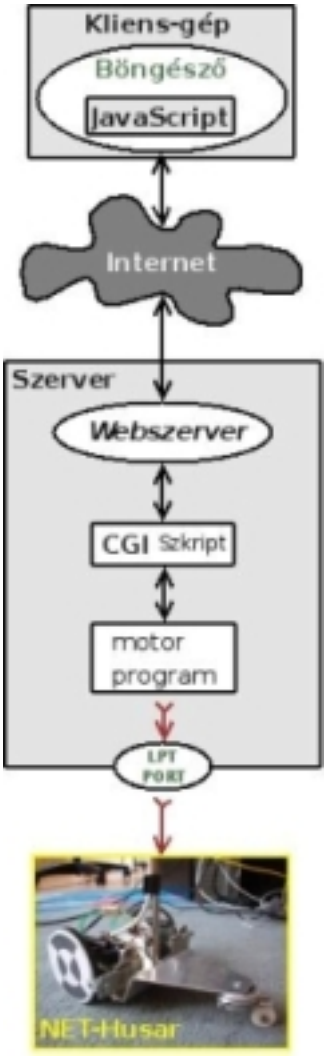
A szkript az `/usr/lib/cgi-bin/megy.cgi` útvonalon található a LIVE-DVD-n, ha a gépet arról inditottuk.

CGI-script

A program írásakor fejtörést okozott, miként tudom leszűrni a paramétert webszerver által visszaadott értelmetlennek látszó szövegből. Erre megoldást kínált egy weboldal, ahol megtaláltam, miként kell kezelni az így kapott paramétereket. [7] Már más dolgom nem volt, csak eldönteni, milyen paraméterrel hívjam meg a `motor` programot.

A szkript *(VIII.2.3. fejezet)* működése:

- HTML fejléc elküldése a kliens böngészőnek. (4-5. sor)
- A `PRG` változó feltöltése a futtatandó program nevével. (8.sor)
- A kliens böngészője által küldött paraméter leszűrése az `IRANY` változóba. (12.sor)



V.1. ábra. Kommunikáció a NET-Husar és a kliensoldali böngésző között

- A `KULSO_IP` változóba, a `/var/www/config/IP` állomány tartalmának beolvasása. (Azt a címet tartalmazza, amin elérhető a weboldal. (15.sor)
- A `vizsgal` függvény meghívása. (38.sor) A függvény (17-36.sor) az `IRANY` változót nézi meg egy *case* szerkezettel, majd meghívja a `motor` programot akkor ha az irany a `motor` program által értelmezett paraméter. Ha nem értelmezett paraméter, akkor nem végez műveletet.
- A HTML kód további részeinek elküldése kliensoldalra a böngészőnek, ami újratölti az oldal megfelelő részét a `KULSO_IP` változóban levő címről. (40-44.sor)

V.3. Weboldal

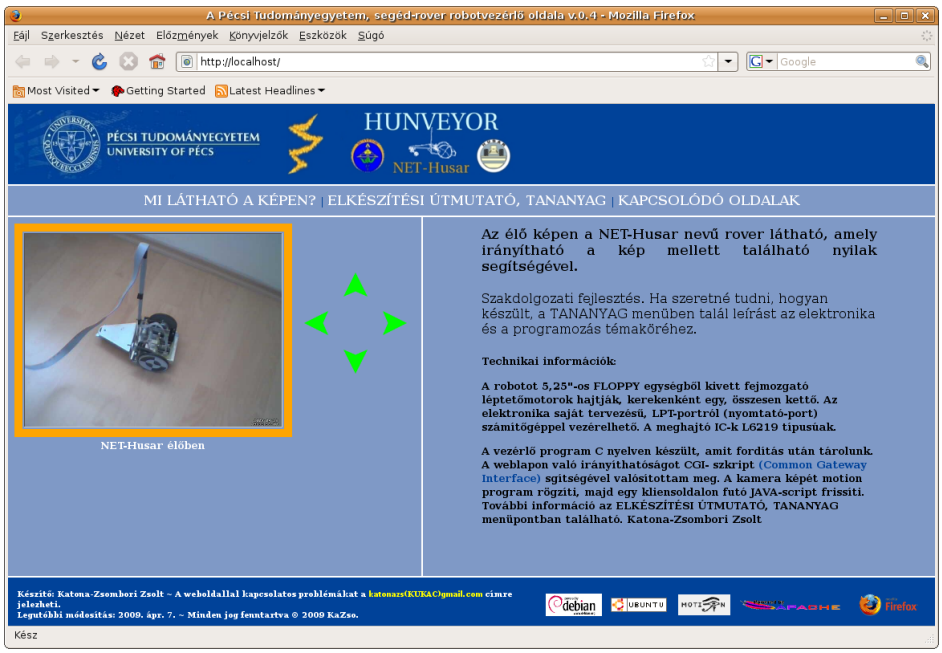
A Weboldal több célt szolgál. Tananyagok olvashatók, letölthetők, a rovert is ezen az oldalon keresztűl figyelhetjük, irányíthatjuk. Tehnikai részleteket tudhatunk meg ebben a fejezetben. Az oldal forráskódját a dolgozat nem tartalmazza, csak a mellékelt Live-DVD-n (`/var/www` könyvtárban) és a Tananyagok-DVD mellékleten a `TELEPITO` könyvtárban található meg.

Több *FRAME*-re osztottam fel az oldalt. Azért, hogy ne az egészet kelljen újratölteni, minden alkalommal, ha valamire kattintunk. Oldal betöltésekor a megfelelő *FRAME*-be a megfelelő HTML oldalrész kerül be. (*V.2. ábra*)

Felépítése

Az oldal a `/var/www` mappában található. Az oldal forrása a DVD-ről való indulás után a `/home/robuntu/NET-Husar/TELEPITO/www` könyvtárban található meg. Módosítás után a `TELEPITO` mappában az `install.sh` - scripttel telepíthetjük a megfelelő helyre.

- `index.html` - Az oldal *FRAME*-einek deklarációja.
- `cim.html` - Az oldal fejléc-képét rakja ki.
- `info.html` - NET-Husar projektről alapvető információk.
- `linkek.html` - Hunveyor projektel és az űrkutatással kapcsolatos oldalakra mutató linkeket tartalmaz
- `menu.html` - Menük megjelenítését végzi.
- `signo.html` - Készítő elérhetősége, és a felhasznált szoftverek logói linkekkel.
- `tananyag.html` - Elektronikus tananyagra mutató linkeket tartalmazza.



V.2. ábra. *Weboldal képernyőkép*

- `vezerles_index.html` - Vezérléshez tartozó két oldal *FRAME*-inek deklarációja. (`vezerles.html` és `js.html`)
- `vezerles.html` - Vezérlőgombokat rakja ki, kattintásra mehívja a `megy.cgi` CGI-szkriptet.
- `js.html` - A kamera képét lehívja a szerverről és egy keretbe rakva megjeleníti. Ezt JavaScript végzi.

Bállítása

A NET-Husar LiveDVD-n levő weboldal úgy lett beállítva, hogy a helyben-fejlesztést tartottam szem előtt. Ahhoz, hogy másik gépről elérjük a weboldalt, legyen élőkép, és az irányítás is működjön, át kell írnunk a HTML-kód bizonyos részeit. Az oldal a `/var/www` mappában található, ami a webszerver gyökere.
Két állományt kell szerkeszteni, a következőképpen:

- `/var/www/config/IP`
  - Ezt a sort: `localhost`
  - Erre cserélni: `http://domain.hu`,
- `/var/www/js.html`
  - Ezt a sort: `http://localhost/images/lastsnap.jpg`
  - Erre cserélni: `http://domain.hu/images/lastsnap.jpg`

VI. fejezet

A technika-oktatás szerepe

Az ember folyamatosan átalakítja környezetét, mesterséges környezetet teremt maga körül. A technika hasonló a művészetekhez mindig újat teremt. A művészet esztétikust, a technika hasznosat hoz létre. Fontos, hogy a diákok megismerjék és értékeljék azt a szellemi erőfeszítést, amely az életünket kényelmesebbé, jobbá tevő technikát szülte. Látniuk kell, hogy a technika nagy lehetőséget jelent : az ember szabadságát terjeszti ki, jobban tud gazdálkodni az erejével, egyre inkább ura a térnek és az időnek. A technikaoktatás célja, hogy lehetővé tegye a napról napra változó technika megértését, követését, használatát, fejlesztve a kreativitást. A gyorsan változó technikának vannak állandó részei, melyek az évezredek során felismert elvek, módszerek, bevált megoldások. A konkrét elméleti és gyakorlati ismereteken, jártasságokon és készségeken túl, ezekre alapozva ki kell alakítani a technikai műveltséget. Ehhez a gazdasági, környezetvédelmi, erkölcsi értékek elvárások együttes figyelembevételére van szükség. Így válik a technika szintetizáló területté.

VI.1. Űrtechnikai modellek helye a technika tanításában

Hunveyor projekt

Maga a Hunveyor-projekt oktatásfejlesztési céllal indult, melyet szerzői a *Fizikai Szemle* 2008/2. számában a következőképpen fogalmaztak meg.
„Egyfajta oktatási fölhasználási lehetőség volt az is, ha a Hunveyor-modellen folytatott építési munka elkészítési és megvalósítási folyamatát elemeztük. A Hunveyor építése összetett technológiai folyamat, ezért összefoglalható gyártási folyamatábrán. Ez a folyamatábra a műveletsorok térképe, melyen az idő függvényében láthatjuk a munka fázisait. Az oktatásban megjelenő szintézismódszerhez és a technológiák összehasonlító módszeréhez is közel áll a Hunveyor gyakorló űrszondán végzett építő és fejlesztő munka. A művelettérkép nemcsak sorba, hanem összképbe is rendezi a szakaszonként és külön-külön végzett építő műveleteket. A munka elemzésének végeredménye az is, hogy a diákok job-

ban átlátják az egyes munkafolyamatokat, a részfolyamatok egymáshoz való viszonyát. Képet alkothatnak a nagy munka egészéről és részeiről is, de a reájuk halmozott részletismeretek nélkül. Megismerhetik tehát a munkafolyamat ábrázolási hierarchiáját is. Ez pedig előnyösen formálja nézeteiket abban az irányban, hogy minden rendszert egy jól megragadható szinten érdemes először áttekinteni, fölösleges részletek elhagyásával. A szerkezeti hierarchia tehát a diákokban formálódó rendszerszemlélet része lesz. Ez a hierarchia ugyanúgy vonatkozik az anyagokra is, melyek tulajdonságait fölhasználják az építés során és a technológiákra, melyek segítségével az építést végzik."[4]

VI.2. Az oktatás hierarchiája

Fontos, hogy az oktatási rendszer felépítése olyan legyen, hogy a legkissebb elemtől a bonyolult, majd a rendszerbe szerveződő technológiákon át mutatssa be a természettudomány ágainak működését. Az óvodától az egyetemig fokozatosan ismerjük meg az egyszerű, majd egyre bonyolultabb rendszereket, ami hozzásegít a rendszerszemlélet kialakulásához. A technika oktatás három nagy fázisra osztható. Életkori sajátosságokat is figyelembevéve kövezkezőképpen néz ki.

Elemi szakasz

Ebben a szakaszban, 1-6.osztályban, a tanulók el kell sajátítsák az alapismereteket, ki kell alakuljanak a legfontosabb tanulási készségek, alapvető a „kéz pedagógiája", a kézügyesség fejlesztése. Ebben a folyamatban szerepet kell kapjon a felfedezés öröme, meg kell tanulni az anyagok felhasználásának lehetőségeit, a szerszámokkal végzett tevékenység folyamatát, alapanyagok modelleken, például papírmodellen keresztül való megismerését, felhasználási lehetőségeit.

Alsó középiskolai szakasz

Ez a 7-10. osztályosokra vonatkozó szakaszban a társadalmi, természettudományi, és technikai ismeretek elsajátítása, rendezett ismeretek megszerzése, a működés-működtetés folyamatának, gyártási folyamatoknak, a működési elvek, törvényszerűségk átlátása lenne célunk. Valamint fontos a kézügyesség és kreativitás fejlesztése. A munkadarab ötletének születésétől , tervezésén át az igényes, pontos kivitelezésig ezen elvek érvényesülnek, továbbá a környezettel való gazdálkodás fontossága is előtérbe kerül, pl.:Lufballonnal meghajtott légmotoros robot.

Felső középiskolai szakasz

Ebbe a szakaszba esnek a 10-12.osztályos tanulók akiknek lehetőségük van a Társadalomtudományi, Természettudományi és technikai ismeretek széles körű megismerésére. Ide értem az összetett informatikai rendszerek, hálózatok működését, pl.: NET-Husar vezérlésének folyamatát is, de ide kapcsolódik a többi Honveyor projekt keretein belül fejlesztett Husar modellek mindegyike.

VI.3. NET-Husar fejlesztés pedagógiai megközelítése

A Hunveyor-modellek, roverek építésének és használatának távlati célja, hogy új ismeretanyaggal és szemlélettel bővüljön a természettudományos tárgyak oktatása, és az ehhez hasonló komplex rendszerek átlátásának képessége beleivódjon a diákokba. A rendszerszemlélet kialakulása, nagyon fontos a hétköznapi életben is, nem beszélve arról mennyire fontos a fejlesztésbe való bekapcsolódáshoz. A részfeladatok egymáshoz való illesztése, részek kompatibilitásának megőrzése rendszerszemléletet kíván.
A tanárok a részletek tudásának birtokában le tudják szűrni a legfontosabb információkat, melyek átadásával nem belesulykolják, hanem felfedezésre ösztönzik a diákokat. A felfedezés öröme mindig nagyobb, mint a tudás elsajátításának gyötrelmes munkája. Ily módon kíván hozzájárulni a technika tanításának megújításához a Hunveyor projekt. Remélem ezt sokan belátják azok közül is, akik felelősek a technika és egyéb készségtárgyak szerepének csökkenéséért. Nagyon fontos lenne rövidtávon ezen fontos tárgyak helyének tisztázása a közoktatásban.

A mai világban fontos lenne, hogy a minket körülvevő technikai eszközarzenál működése ismert legyen a felhasználói, tehát az emberek számára. Ezt hivatott a technika, mint szintézistárgy nyújtani. Sajnos az általános iskolai képzés után nemigen találkozunk a technika tantárggyal középiskolában, vagy csak elvétve. A modern élet nagyméretű rendszerek, közösségi szervezetek működésének ismeretét, átlátásának képességét kívánja. Ez az ami nem alakul ki magától, és ha már nincs is ahol készségszinten elsajátítani, a felnővő generáció egy technikához hozzá nem értő „tudatlan” önállóan fejlődni nem képes, más társadalmaktól függő lesz.

A NET-Husar építését azon középiskoláknak ajánlom, melyek szeretnének a technika tanításán belül egy komplex rendszert megépíteni, azon mint modellen szemléltetni annak működését. Azért inkább középiskolák, egyetemek figyelmébe ajánlom, mert széleskörű ismereteket, rendszerben való gondolkodást követel a NET-Husar megvalósítása. Nem zárom ki, sőt javaslom általános iskolában is az ismertetését, hiszen a tanuló gyerek fantáziája sokkal jobban ki tud bontakozni, ha valami olyat lát, amit közvetlen közelről is megnézhet, netán részfeladatokat meg is valósíthat. Az tananyagot több részre bontottam, melyekre utalok a megfelelő fejezetekben. Egymásra épülő egységek azok, melyeket végrehajtva eljuthatunk a végkifejletig. Mivel önálló részekből tevődik össze,

de azok egymásra épülnek, főleg a gyakorlati és technikai részekre gondolok, fontos a sorrendiség. Tartalmaz alternatívát pl. a nyomtatott-áramkör készítési fejezetben, így a helyzetnek megfelelően lehet választani az alternatívák közül. Nem kizárt, hogy szakkör keretén belül valósuljon meg mindez, ahol a diákok már rendelkeznek a megfelelő technikai alapokkal.

Modellezés

A technika-tanítás fontos eleme a modellezés. A jó modell egy bonyolult rendszer egyszerű szemléltetését teszi lehetővé, ami csak azon elemeket tartalmazza amire szükség van. A gyakorlatban a méretarányos modellt, kicsinyített, vagy nagyított formában építik meg és vizsgálják. A Net-Husar is egy ilyen modell, ami nem egy nagyobb és nem egy kissebb rendszer másolata ugyan, de a rendszerek összetettségéből adódóan egy új modell, ami alapján el lehet jutni a tökéletesre.

Csoportmunka

A Net-Husar egy egyszerű mozgásra képes jármű, aminek megépítése nem bonyolult folyamat, de aránylag hosszú időt vesz igénybe. A feladatokat sok apró egymásra épülő részre lehet bontani, amit a csaoport különálló tagjai végezhetnek, majd a részfeladatok elvégzése után rendszerbe lehet foglalni, össze lehet rakni egy bonyolult egésszé. A feladatkiosztás a következőképpen lehetséges, amit különböző csoporttagok hajthatnak végre:

- Kell egy koordinátor, ez lehet a tanár, aki összefogja a feladatot, ütemezi a munkát.
- Az elektronika tervezése közös feladat, de lehet egyéni is.
- A forrasztási feladatot lehet több tanuló között felosztani az alkatrészek magasságától függően.
- Programozási feladat több részre osztása:
    - Weboldal szerkesztése külön embert igényel.(jó, ha egyvalaki csinálja végig)
    - A motor-meghajtóprogram fejlesztését végezheti valaki más, akár többe is.

A kis csoportokban fejlesztett egységeket össze kell illeszteni egy működő egésszé, tehát kellenek időszakosan olyan megbeszélések ami alatt a közös pontokat fixálják, megbeszélik azokat a dolgokat amik nehézséget okoztak. Mivel fontos, hogy ne mindenki csak a saját érdeklődési körének megfelelő területen dolgozzon, lehet coportok közötti cserét felajánlani a különböző munkaköröket végző tanulóknak, így mindenki beleláthat mindabba amibe eddig nem volt lehetősége. A tanár mint segítő vesz részt a folyamatban,

ha kell segít, koordinál, de nem főszereplőként van jelen. A diákok, mivel nem egyedül dolgoznak segítik egymást, nehézségeik leküzdésében. Mivel a csoportban összetartás szükséges a sikerek eléréséhez, kialalkul a csoportszellem ami hajtja a újabb és sikeresebb megoldások megtalálása felé a tagjait. Mindeközben kialakul a csoportok közötti természetes versenygés is. Ez hasonló a hétköznapi életben munkahelyen tapasztalható körülményekhez, ahol a nagyobb projekteken csapatban dolgoznak a probléma megoldásán.

Oktatás menete

Mivel egy olyan eszköz oktatásáról van szó, aminek anyagait elektronikus formában egy DVD-n külön mellékeltem, legtöbb hozzá kapcsolódó feladatot számítógépen kell végezni, nem gondolom, hogy fel lehet osztani szabályos órákra a megvalósítás menetét. Mivel csoportok külön is végezhetik az adott feladatokat, ezért a tananyag is ezen csoportokra lett felosztva. Minden alap munkálat elvégzése után konzultáció szükséges a további feladatok elosztása végett. A tanár a dolgozat alapján el tudja dönteni, milyen sorrendben hajtatja végre és melyik csoporttal a fennmaradó részfeladatokat. Nagyon sok segytséget kaphatnak a munkájukhoz, ha az Internetről is tájékozódnak mielőtt végrehajtanak bizonyos feladatokat. Természetesen a tananyag plussz információkkal is szolgál a dolgozathoz képest hiszen a vizuális megjelenítés alapján olyan plussz információkhoz juthatnak, amit szóban nem mindig lehet pontosan kifejezni.

Bemutatkozás az OTDK-n

Idén, is megrendezésre került a *29. Országos Tudományos Diákköri Konferencia*, melyen ismét indult BAKSA LÁSZLÓ, a *Robotfejlesztés és innováció a technika oktatásában* című előadásával, a *tantárgypedagógia és oktatástechnológia szekcióban*. Szép eredménnyel zárt, második helyezést ért el, első díjat nem osztottak ki. Az előadásában helyet kapott az egyéb feljesztési vonalak részben a *NET-Husar* is, ahol működésének rövid ismertetését, hallhatta a tisztelt publikum. Szó volt az oktatásban egyre szűkebb szerepet elfoglaló technika-tanítás fontosságáról és a folyamat visszafordításában szerepet játszó űrtechnikai kutatásokról is amik helyet kapnak a közoktatás legkülönbözőbb színterein a középiskolai szakkörön át az egyetemi képzésig. Előadás közben ki is lehetett próbálni a rover irányítását, akkor még tesztüzemi körülmények között.

VII. fejezet

Összefoglalás

A dolgozat elején eltökéltem, hogy készítek egy olyan tananyagot, amiből iskolák a saját meglévő erőforrásaikat felhasználva meg tudják ismételni azt amit én a fejlesztés ideje alatt létrehoztam.

Úgy gondolom, az elsők között lehetek abban a törekvésemben, hogy egy szabadon felhasználható ingyenes, továbbfejleszthető platformon valósítottam meg egy oktatási célzatú fejlesztést, ami, az egyébként is gyengén támogatott és kevés alternatívát nyújtó technika-tanítás eszközévé válhat. Ami még fontos mindemellett, hogy a diákok a technika olyan területére nyernek bepillantást, ahova csak álmaikban tudnának eljutni. Az űrtechnikába, annak fejlesztésébe való bekapcsolódás számos lehetőséget rejthet, szélesíti látókörüket, segítséget nyújthat jövőbeli szakmájuk megválasztásakor. Mivel rendszerszemléletet igényel, hamarabb kialakul a bonyolult rendszerek egyszerűsítésének igénye, ami azok átlátását igényli.

A mai kor igényeinek minden tekintetben megfelelő szabadon felhasználható operációs rendszerek régen kinőtték a *„nem felhasználóbarát”* jelzőt. Hihetetlen fejlődésük elért egy olyan pontot, amikor meg kell gondolnunk, megéri-e megvenni a kereskedelmi szoftvereket, vagy inkább az ingyenesen elérhető, rohamosan fejlődő szoftverek mellett döntünk.
Ez az ami arra sarkallt, hogy én is egy ilyen rendszeren fejleszzem a NET-Husart. Ennek szemléltetésére készült el a NET-Husar Live rendszer, ami a fejlesztéshez szükséges elemeket már tartalmazza. Telepítés nélkül kipróbláható, teljesértékü rendszert kapunk általa. Természetesen lehet telepíteni is, ami után nem kell minden alkalommal ugyanazokat a beállításokat megismételni kikapcsolás után.

Az elektronikai feljesztések alatt mind jobbat akartam kihozni a szerkezetből, de rájöttem, hogy közben bizonyos állomásokon meg kell állni, elemezni, majd újra átgondolva jobbat alkotni. Az épített saját tervezésű elektronika éppen elegendő volt céljaim eléréséhez, de elegendően összetett, hogy megfelelő elméleti és gyakorlati tudás elsajátítását tegye lehetővé ezen a szakterületen azoknak akik megvalósítását tűzik ki ennek a járműnek, vagy ez alapján bármilyen egyéb hasonló elven működőnek. Úgy gondolom

elértem azt a célt is, hogy ha tovább kellene lépnem a fejlesztésben, tudom pontosan, merre tennék lépéseket, mit javítanék, mit adnék hozzá.

Nem kis eredmény az, hogy érdemesnek találták a dolgozatom témáját megemlíteni egy Országos Tudományos Diákköri Konferencián elhangzó előadáson, ahol pedagógiai szempontok alapján értékeltek.

Jövőkép, további fejlesztési célok

Minden eszközt amit valaki kitalált, megvalósított, tovább lehet fejleszteni. Az ember igényei egyre nagyobbak, a technikai rendszerekkel szemben.Ha megfigyeljük a technika fejlődését, látjuk, hogy pár év alatt a legbonyolultabb, legjobb rendszerek is elavulttá válnak. A húzó iparágak között az űrtechnika nagy szerepet játszik, mely a folyamatos fejlesztési eredményeit lassan csöpögteti a hétköznapi életben használt eszközökbe.
Mivel a fejlesztés során nem az volt a fő szempont, hogy a legfejlettebb technikával szereljük fel a NET-Husart, ezért nem is tettünk semmilyen műszert a fedélzetre, viszont fennáll a továbbfejlesztés lehetősége.
Miközben nyomtatókat szedtem szét léptetőmotor után keresgélve, találtam egy szivattyúhoz hasonló szerkezetet, amit ugyancsak léptetőmotor hajt. Valószínüleg ugyanezzel az elektronikával lehetne működtetni és máris lenne egy folyadék-mintavevő szerkezet a roveren, ami egy kis erre a célra elkülönített tartályba tudna folyadékot felszívni,, majd a rover elszállítani további vizsgálatra.

A következő ötlet, az már a vezérlő számítógép kiváltását célozza meg. Mivel az általunk használt integrált áramkör meghatározott impulzusokra reagál amit a számítógépen futtatott programmal generálunk, a feladatot egy erre a célra fejlesztett integrált áramkör (IC) végezhetné el, amihez kevesebb vezérlőbit felhasználása lenne szükséges. Az így felszabadult vezérlőbiteket másra (pl.: szivattyú bekapcsolása, világítás ki/be kapcsolása) tudnánk felhasználni. Fel lehetne szereni rá egy kissebb mesterséges intelligenciát biztosító áramkört is, ami bizonyos helyzetekben önállóan dönthetne, például nem engedné, hogy szakadékba vezessük ezzel önmagát védené. Lehetne vezetéknélküli technológiákat felhasználni, lehetne egy másik vázat építeni más, erősebb motorokkal, ami újabb távlatokat nyitna a fejlesztés előtt.

Szoftveresen is lenne mit fejleszteni. Az egyik ötlet az, hogy ne engedjünk mindenkit egyszerre a rover irányításához hozzáférni. A weboldalon regisztráció után lehetne egy időintervallumot foglalni az irányítás megszerzéséhez. Gondolom mindenkinek lenne pár jó ötlete, ha elengedi a fantáziáját.

Ezen célok magvalósításához sok kitartás, és a közben felmerülő gondok megoldásához használható ötletek kellenek, amit egy gimnázium, vagy szakközépiskola lelkes csapata adhat. Ha nem is a teljes, de részleges magvalósítása a leírtaknak, a fejlesztő gondolkodást kialakulását nagyban elősegíti.

VIII. fejezet

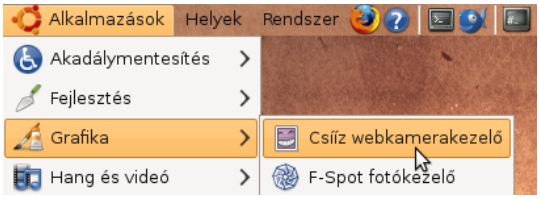
Függelék

VIII.1. Hibaelhárítás

Webkamera

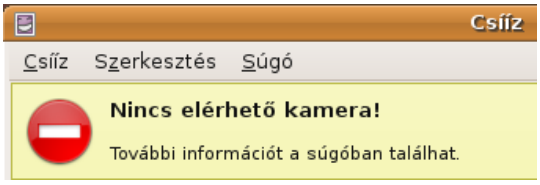
Ellenőrzés

Nézzük meg, hogy a *Csííz webka-merakezelő*-vel látjuk-e a kamera képét. Indítsuk el a programot. (*VIII.1. ábra*)



VIII.1. ábra. *Csííz webkamerakezelő helye a menüben*

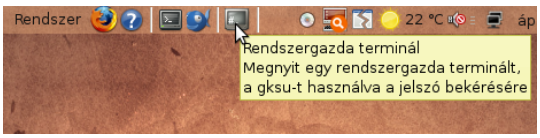
Ha rendszerünk nem ismerte fel a kamerát, a Csííz közli velünk (*VIII.2. ábra*). Ebben az esetben, Interneten kell megkeresni kameránkhoz a megfelelő beüzemelési útmutatót.



VIII.2. ábra. *A csííz nem látja a webkamerát*

A *motion* beállítása

Ha működik a kameránk, akkor csak a `motion` program beállításaival van probléma. Nyissunk egy *Rendszergazda terminál*-t. (*VIII.3*. ábra)



VIII.3. ábra. *Terminál megnyitása: kattintsunk az egérrel mutatott ikonra*

Adjuk ki ezt a parancsot:

```
nano /etc/motion/motion.conf
```

Ha megnyílt a konfigurációs állomány, keressük meg a

```
width=352
height=288
```

sorokat, és írjuk át arra a legkissebb értékre amit a kamera tud. Általában a dobozon fel vannak tüntetve a támogatott felbontások. Ha mégsem, próbáljuk ki az alábbi beállítást.

```
width=640
height=480
```

`Ctrl+o` billentyűkombinációval mentsük a változtatást, majd `Ctrl+x` -el lépjünk ki a `nano`-ból. Indítsuk újra a `motion`-t,

```
/etc/init.d/motion restart
```

majd ellenőrizzük a böngészővel. Nyissuk meg a kezdőoldalt *(localhost)*, ha van kép, akkor sikeresen elhárítottuk a hibát. Ha nincs kép, lehet, a motion valami miatt megint leállt. A továbbiakban a `/var/log/syslog` állomány utolsó pár sorát javaslom megnézni, mert a program ebbe írja a hibaüzeneteket, információkat.

VIII.2. Forráskódok

VIII.2.1. *„motor.c”*

```
1  /*
2    Ez egy printer-port bitjeit valtoztatni kepes program.
3    A parameterben megadott szamot kii rja a megadott port-ra (378)
4  */
5
6  #include <stdio.h>
7  #include <stdlib.h>
8  #include <unistd.h>
9  #include <sys/io.h>
10
11 #define port 0x378          /* a printer port cime */
12
13 /* ############## függvények ############## */
14
15 void var()      // Ez a függvény a megadott ezredmádsodpercig várakozik.
16 {
17   usleep(15000);
18 }
19
20 void hatra()   // Ez a függvény a NET-Husart hátra mozgatja
21 {
22     int i;   // helyi változó deklarálása ciklusváltozónak használjuk
23
24   for (i=0; i<10 ; i++)
25     {
26     /* az EGYIK motor egy lépése */
27          outb((unsigned char)16, port);      // A "port"-ra a "16"-t írunk
                .
28          var();                    // Várakozik
29          outb((unsigned char)48, port);
30          var();
31          outb((unsigned char)32, port);
32          var();
33     /* a MASIK motor egy lépése */
34          outb((unsigned char)1, port);
35          var();
36          outb((unsigned char)3, port);
37          var();
38          outb((unsigned char)2, port);
39          var();
40     } // CIKLUS vége
41 exit(0);
42 }
43
```

```
44 void elore()     // Ez a függvény a NET-Husart előre mozgatja
45 {
46     int i;
47     for (i=0; i<10 ; i++)
48     {
49     /* az EGYIK motor*/
50         outb((unsigned char)32, port);
51         var();
52         outb((unsigned char)48, port);
53         var();
54         outb((unsigned char)16, port);
55         var();
56     /* a MASIK motor */
57         outb((unsigned char)2, port);
58         var();
59         outb((unsigned char)3, port);
60         var();
61         outb((unsigned char)1, port);
62         var();
63     }
64 exit(0);
65 }
66
67 void jobbra()    // Ez a függvény a NET-Husart jobbra mozgatja
68 {
69     int i;
70     for (i=0; i<10 ; i++)
71     {
72     /* az EGYIK motor*/
73         outb((unsigned char)16, port);
74         var();
75         outb((unsigned char)48, port);
76         var();
77         outb((unsigned char)32, port);
78         var();
79     /* az EGYIK motor*/
80         outb((unsigned char)2, port);
81         var();
82         outb((unsigned char)3, port);
83         var();
84         outb((unsigned char)1, port);
85         var();
86     }
87 exit(0);
88 }
89
90 void balra()     // Ez a függvény a NET-Husart hátra mozgatja
91 {
```

```
92      int i;
93      for (i=0; i<10 ; i++)
94      {
95      /* az EGYIK motor*/
96          outb((unsigned char)32, port);
97          var();
98          outb((unsigned char)48, port);
99          var();
100         outb((unsigned char)16, port);
101         var();
102     /* az EGYIK motor*/
103         outb((unsigned char)1, port);
104         var();
105         outb((unsigned char)3, port);
106         var();
107         outb((unsigned char)2, port);
108         var();
109     }
110 exit(0);
111 }
112
113 /*############## Itt kezdodik a foprogram #############*/
114 main(int argc, char *argv[])
115 {
116
117 if (argc!=2){ // Ha a paraméterek száma nem 2, akkor kilépünk.
118     printf(" (HIBA) - A programnak meg kell adni egy parametert, \n\
        nHasznalat: motor [e | h | j | b] \n\n\te - elore\n\th - hatra\
        n\tj - jobbra\n\tb - balra\n\n");
119     exit(1);
120     }
121
122 if (ioperm(port,1,1))
123     fprintf(stderr, "(HIBA) - Nem erem el a portot: %x\n 1.) Lehet,
        nincs jogom hozza? MEGOLDAS, ha futtatod ezt a parancsot: \n\n\
        t\t 'sudo chmod +s [eleresi ut]/motor'\n\n2.) Rosszul van
        megadva a programban a port cime. A program elején keress egy
        ilyen sort :#define port 0x378 , majd írd át a '378-at' '278'-
        ra.\n\n3.) Nincs LPT (nyomtato) port a szamitogepben. :(\n\n",
        port), exit(10);
124
125     // printf("parameter:\n 0. %s\n\t1. %s \n\t2. %s\n", argv[0], argv
        [1], argv[2]);
126
127
128 if (strncmp ("e", argv[1], 1) == 0)    // Ha a paraméter "e"
129     elore();                  // Meghívjuk az elore függvényt
130
```

```
131 if (strncmp ("h", argv[1], 1) == 0)    // Ha a paraméter "h"
132     hatra();                  // Meghívjuk a hatra függvényt
133
134 if (strncmp ("j", argv[1], 1) == 0)    // Ha a paraméter "j"
135     jobbra();                 // Meghívjuk a jobbra függvényt
136
137 if (strncmp ("b", argv[1], 1) == 0)    // Ha a paraméter "b"
138     balra();                  // Meghívjuk a balra függvényt
139
140 exit(0); //Kilépés hiba nélkül
141 }
```

VIII.2.2. *„fordit.sh”* fordítószkript *motor.c*-hez

```
1  #!/bin/bash
2
3  cd /home/nethusar/NET-Husar/C_forras
4    # Paraméter vizsgálata
5  if ! test -z "$1" ; then
6    # gcc fordítóval lefordíttatjuk a "$1" paramáterben levő forrást
7    echo -en "Forditas... \t\t"
8    if gcc $1 -o motor ; then
9      echo -e "[OK]"
10
11     # jogot adunk mindenkinek, hogy a programot ROOT-ként futtathassa
12     echo -en  "Jogosultsag beallitasa... \t"
13     if chmod +s motor ; then
14       echo -e "[OK]"
15     else
16       echo -e "[HIBA]"
17     fi
18
19     # Bemasolja forditas utan a telelpitocsomagba
20     echo -en "Masolas: motor -> ../TELEPITO/usr/sbin \t"
21     if cp motor ../TELEPITO/usr/sbin ; then
22       echo -e "[OK]"
23     else
24       echo -e "[HIBA]"
25     fi
26
27     echo -en "Masolas: motor --> /usr/sbin \t"
28     if cp motor /usr/sbin ; then
29       echo -e "[OK]"
30     else
31       echo -e "[HIBA]"
32     fi
33   else
34     echo -en "[HIBA]"
35   fi
36 else  # Ha nincs paraméter... hibaüzenet
37   echo "Adj meg paraméterben egy C forrást! \n kimeneti allomany a '
       motor' lesz"
38 fi
39 echo "Varok 10 masodpercet mielott kilepek..." ; COUNTER=10
40  while [  $COUNTER -gt 0 ]; do
41    echo "$COUNTER"
42      let COUNTER=COUNTER-1
43    sleep 1
44  done
```

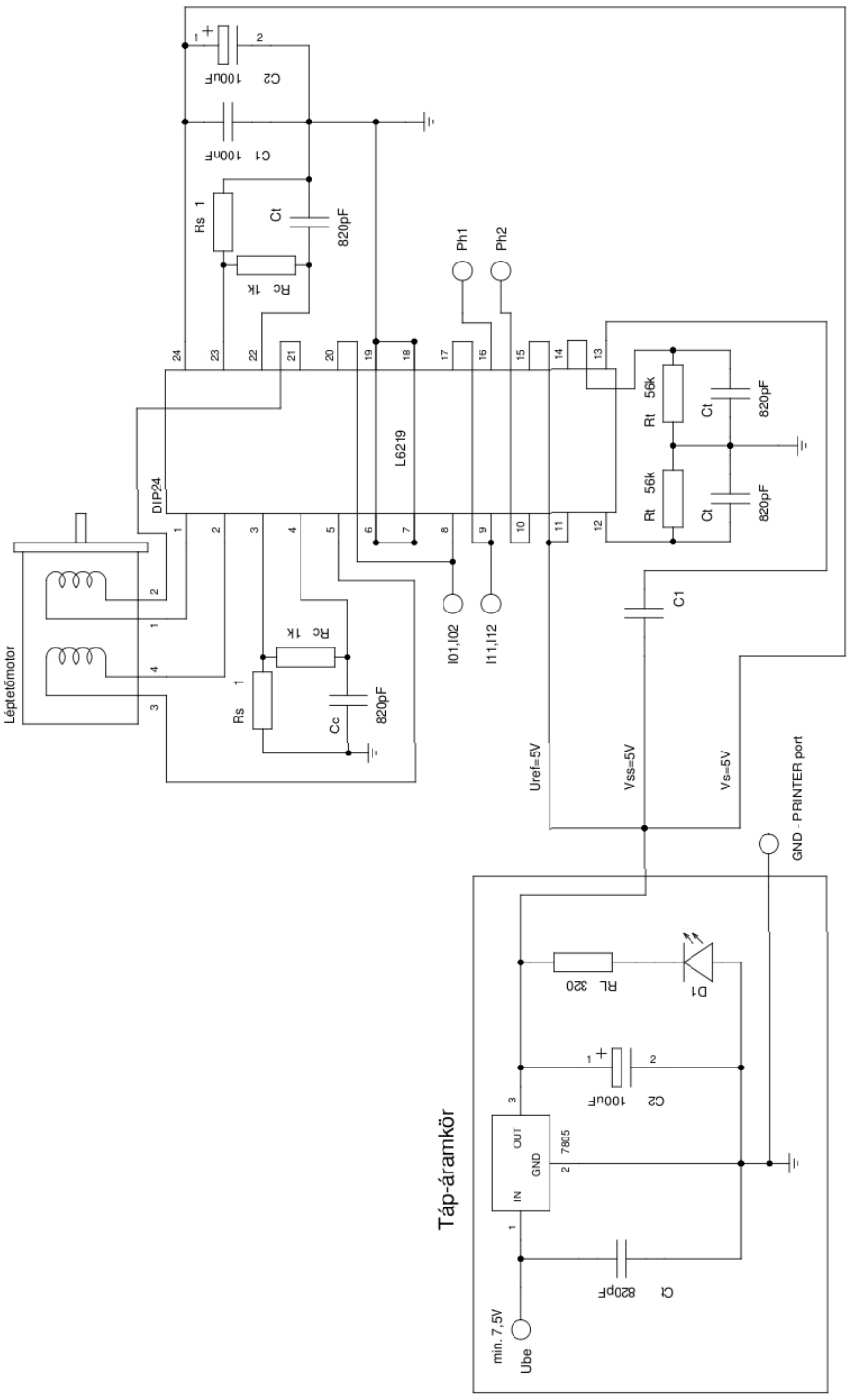
VIII.2.3. *„megy.cgi”*

```
1  #!/bin/bash
2
3  # HTTP fejléc
4  echo 'Content-type: text/html ; charset=ISO-8859'
5  echo
6
7  ## A változó a program teljes elérési útját és nevét fogja tertalmazni
8  PRG="/usr/sbin/motor"
9
10 ## Az irányt paraméterben kapjuk a kliens bongeszotol.
11 ## Amire szuksegunk van, azt leszurjuk. es belerakjuk az IRANY
       valtozoba
12 IRANY=`echo "$QUERY_STRING" | grep -oE "(^|[?&])IRANY=[^&]+" | sed "s
       /%20/ /g" | cut -f 2 -d "="`
13
14 ## Erre a külső IP-re fog átirányítani ez a generált HTML miután
       lefutott.
15 KULSO_IP=`cat /var/www/config/IP`
16
17 function vizsgal()
18 {
19 case "$IRANY" in
20  elore)
21   $PRG e
22  ;;
23  hatra)
24   $PRG h
25  ;;
26  jobbra)
27   $PRG j
28  ;;
29  balra)
30   $PRG b
31  ;;
32  stop)
33   $PRG s
34  ;;
35 esac
36 }
37
38 vizsgal
39
40 echo "<html><head>
41 <META HTTP-EQUIV=\"Refresh\"
42 CONTENT=\"0; URL=http://$KULSO_IP/vezerles.html\">
43 </head><body></body></html>"
```

VIII.3. Felhasznált szoftverek

- **Operációs rendszerek**
  - **Debian GNU/Linux 5.0 Lenny** - *Tervezésre, szerkesztésre, programozásra* (http://debian.org)
  - **Ubuntu 8.10 Hardy Heron** - *Oktatási eszköz, LIVE-CD. (webszerver, webkamera, motion)* (http://ubuntu.com)
- **Kapcsolási rajz készítéshez**
  - **gEDA** - *GPL Electronic Design Automation* gshem version 1.4.0.20080127 (http://www.gpleda.org/)
- **Nyomtatott áramkör tervezéshez**
  - **PCB** - *„PCB, az interaktív nyomtatott-áramkör tervező” (verzió: 20080202)* (http://pcb.sf.net)
- **Programozáshoz**
  - **Bluefish** szövegszerkesztő *(verzió: 1.0.7)* (http://bluefish.openoffice.nl/)
- **Ezen dolgozat írásához**
  - **LaTeX** tördelőrendszert. (http://www.latex-project.org/)
    - A „pdf” dokumentum *pdflatex*-el készült.
  - **Kate** szövegszerkesztő *(verzió: 0.96.1)*
- **Folyamatábra készítéshez**
  - **Dia** - *„Strukturált diagramok szerkesztésére szolgáló program.” (verzió: 0.96.1)* (http://projects.gnome.org/dia/)

Kapcsolási rajz



VIII.4. ábra. *Áramköri rajz*

Külön keretezve látható a tápegység rész. Az Integrált áramköri részt kétszer kellene feltüntetni, mivel így is nehezen vehető ki minden részlet, csak egyszer szerepel a rajzon.

Alkatrészjegyzék

Alkatrész	**jelölés**	**érték**	**Darab**	**Alkatrész**	**jelölés**	**Darab**
ellenállás	R_t	56kΩ	4	Feszültség stabilizátor IC	7805	1
ellenállás	R_s	1Ω	4	Léptetőmotor-hajtó IC	L6219	2
ellenállás	R_c	1kΩ	4	LED (ø5mm)	LED	1
ellenállás	R_L	320Ω	1	Kapcsoló	Dip_1	1
kondenzátor	C_t, C_c	820pF	8	Csatlakozópár (papa/mama)	CS_1	4
kondenzátor	C_1	100nF	3	Tápcsatlakozó	CS_2	1
kondenzátor	C_2	100μF	4			

VIII.1. táblázat. Alkatrészjegyzék

Hallgatói nyilatkozat

Alulírott diplomázó hallgató kijelentem, hogy jelen szakdolgozat saját munkám eredménye, a felhasznált szakirodalmat és eszközöket azonosíthatóan közöltem. Egyéb jelentős segítséget nem vettem igénybe.
Az elkészült szakdolgozatban található eredményeket a Pécsi Tudományegyetem, mint a feladatot kiíró intézmény, saját céljaira térítés nélkül felhasználhatja.

Pécs, 2009. május 7. .....

aláírás

Irodalomjegyzék

[1] *Kozmikus Anyagokat Vizsgáló Űrkutató Csoport (KAVÜCS) munkássága*
http://planetologia.elte.hu/

[2] *Wikipédia - Hunveyorhoz, űrkutatáshoz kapcsolódó cikkek*
http://hu.wikipedia.org/wiki/Hunveyor

[3] *Robotok és űrkutatás* Modern Iskola 2008/9.

[4] BÉRCZI SZANISZLÓ, HEGYI SÁNDOR, HUDOBA GYÖRGY: *A Hunveyor gyakorló űrszondamodell sokoldalú fölhasználása a fizika tanításában és a tantárgyi kapcsolatokban.* Fizikai Szemle 2008/2.

[5] *Művelődési és Közoktatási Minisztérium, Szakmai Irányítási Főosztálya 1992-1993, szerk:*BARANYI KÁROLY: *Ez lett volna a NEMZETI ALAPTANTERV 1993-ban*

[6] *Wikipédia - CGI(protokoll)*
http://hu.wikipedia.org/wiki/CGI_(protokoll)

[7] *CGI Scripting Tips for Bash or SH*
http://www.ffnn.nl/pages/articles/linux/cgi-scripting-tips-for-bash-or-sh.php

[8] STMICROELECTRONICS: *L6219 StepperMotorDriver manual*, 2005.
http://www.st.com