VISOKA TEHNIČKA ŠKOLA STRUKOVNIH STUDIJA - SUBOTICA
SZABADKAI MŰSZAKI SZAKFŐISKOLA
SUBOTICA TECH – COLLEGE OF APPLIED SCIENCES

TÖBB ÜZEMMÓDÚ ROBOTAUTÓ ARDUINO PLATFORM SEGÍTSÉGÉVEL

VIŠEFUNKCIJSKI ROBOT-AUTO OSTVAREN ARDUINO PLATFORMOM

MULTIFUNCTIONAL ROBOTCAR WITH ARDUINO PLATFORM

Hallgató

PÁSZTOR ZSOLT

Mentor

MR. PÓTH MIKLÓS

Szabadka, 2016

Absztrakt

A szakdolgozat témája egy több üzemmódú robotautó megtervezése, majd megvalósítása Arduinoval történő vezérlőlap segítségével, s ezen keresztül mikrovezérlő programozással. 4 különböző működési mód megvalósítása volt előlátva:

- fényforráskövetés
- akadályelkerülés
- fekete vonalkövetés
- szabadon történő irányítás megvalósítása

Ezeket a programokat mind egy szerkezetbe kellett implementálni és megoldani úgy a köztük lévő választási lehetőséget, IR távirányító segítségével, hogy ne kelljen két programváltás között új programot feltölteni a mikrovezérlőbe. A test megtervezésénél figyelembe kellett venni, hogy minél felhasználóbb barátiabbá igyekezzünk tenni azt, hogy akár egy kisebb gyerek is kezelni tudja, hisz ezt az autót egy játékként is lehet szabadalmaztatni a jövőben. Arra is figyelmet kellett fordítani, hogy ezek mindössze alapprogramok, s gondolni kellett a jövőbeli továbbfejlesztések lehetőségére is, hogy két funkciót keresztezni lehessen, vagy új funkciókat lehessen hozzáadni. Ahhoz, hogy ezt mind meg tudjam valósítani belemélyedtem magába az Arduino vezérlőlap irányításába, valamint ki kellett ismernem a hozzá beszerezhető bővítő lapok egy tetemesebb mennyiségét. Ezen kívül a robottestek világával is muszáj volt megismerkednem, hiszen a robotom, melyet megalkottam, egy ilyen testen helyezkedik el.

Tartalom

Jelmagyarázat

Jelzés/Rövidítés	Jelmagyarázat
PWM	Pulse Width Modulation (impulzus szélesség moduláció)
IR	infrared (infravörös)
DC	Direct Current (egyenáram)
Ω	ohm (ellenállás mértékegysége)

Köszönetnyilvánítás

Köszönetet szeretnék mondani mentoromnak, Mr. Póth Miklósnak, aki útmutatásával és ötleteivel segítette a projektem létrejöttét. Édesapámnak, Pásztor Lajosnak, aki biztosította számomra a szükséges szerszámokat, de ami fontosabb, hasznos tanácsokkal látott el, ha épp elakadtam valahol a munkám során. Bessenyei Szilárdnak, egykori demonstrátoromnak, a rengeteg információért, illetve építő kritikáért, amelyet nyújtott. Öcsémnek, Pásztor Norbertnek, aki nélkül sokkal nehezebb dolgom lett volna a gépészeti tervek megalkotásában, valamint barátnőmnek, Szőnyi Arabellának, aki készségesen segített az alkatrészek beszerzésében, ez által megkímélve engemet a felesleges utazástól.

A szakdolgozat témája

Belemélyedni a mikrovezérlők világába, konkrétabban az Arduino platformra koncentrálva, kiismerni a működését, s annak használatát. Megismerni a különbségeket, közös pontokat a digitális és analog kimenetek/bemenetek között, rajtuk keresztül történő vezérlés elsajátítása. Elmélyedni az Arduino fejlesztőkörnyezethez gyártott bővítő lapok világában, kipróbálni több elemet is közülük, megérteni a működésüket. Ezen folyamatok alatt akkora szaktudást elsajátítani, mely lehetővé teszi egy bonyolultam, több funkciójú robot megvalósítását is. A szerkezet helyváltoztatásra kell, hogy képes legyen automata és tetszés szerinti módozatban, mely által olyan különböző irányítási problémákkal és azok megoldásával történik találkozás, mely előkészítenek bonyolultabb mozgások megtervezésére is a jövőben. A test összeállítása saját kezűleg történjen, ezáltal gyakorlatiasabb, fizikálisabb tapasztalatok ismerete is végbemegy. Az mozgó testet olyan szenzorokkal és aktuátorokkal kell ellátni, melyek különböző specifikális működési lehetőségeket nyitnak meg és ezeket össze kell hangolni, úgy, hogy egymással megfelelően tudjanak kommunikálni. A kész többfunkciójú autón teszteléseket végezni, megismerkedni a különböző talajokon történő működéssel, s beállítani a zökkenőmentes haladást egy bizonyos súrlódású talajhoz viszonyítva.

1. Bevezető

Az emberiség életét, mióta az eszét tudja, folyamatosan a fejlődés, gyarapodás, alkotás kíséri végig. A történelem folyamán és jelenleg is mindig arra törekszik, hogy valami olyat hozzon létre, ami még nem volt, illetve hozzáadjon valamit ahhoz, ami már jól bevált és működik. A fejlődés során egyre inkább kezdi elfelejteni, hogy honnan is indult és az alap dolgoknak nem fordít már megfelelő figyelmet. Elkezdett földet művelni, a létfenntartás végett, majd rájött, hogy számos dolog van az életben, melyben ki lehet teljesedni, így felfedezte a szerszámokat, hogy könnyebb dolga legyen, és több szabadideje maradjon. Majd a szerszámokat elkezdte gépekhez csatolni, melyeket mindössze irányítani kellett, s ez által jelentősen tovább csökkentette az élelmezésre rászánni való időt. Manapság pedig már teljes automatizációt alkalmaz, az embernek egyre kevesebbet kell beleavatkozni a dolgok zajlásába, iparisította az élelmiszergyártást, az életének számos területével egyetemben. E szellemiséget követve akartam kicsit én is beletekinteni a robotizáció világába. Megérteni, hogyan is zajlik egy robot gyártása, alkotni valami újat, megismerni kicsit az alapokat, de építeni arra, amit már előttem lefektettek. Ehhez megfelelő belépési pontnak gondoltam az Arduino vezérlési forma megismerését, mellyel könnyen lehet alkotni bonyolultabb szerkezeteket is, elég számú lehetőséget biztosít egy alap robot felépítéséhez, illetve ezen a területen lévő szárnypróbálgatásaimhoz is megfelelő kezdést szolgáltat.

1.1. A probléma leírása

Ahhoz, hogy elkezdhessek Arduinoval foglalkozni szinte teljesen az alapoktól kellett kezdenem mindent. Noha volt már tapasztalatom mikrovezérlők programozása területén, azok csak alapszintű, kis kapcsolások voltak, pár elektronikai elemmel, melyek közül többnyire a 8 szegmenses kijelzőre működésre bírása számított a legösszetettebb feladatnak. Ezért próbáltam olyan témát választani, amely kellő kihívást biztosít számomra a fejlődéshez, ugyanakkor nem gördít elébem akkora akadályt, mellyel nem tudnék megbirkózni. Tekintve azt viszont, hogy számos utánajárással és témakereséssel töltött óra után sem tudtam eldönteni, hogy mit valósítsak meg pontosan, így úgy döntöttem, hogy vegyítem az egyes ötleteimet, és összeállítok egy személyre szabott, több üzemmódban működő, különböző szenzorokkal ellátott robotautót. Ahhoz, hogy ezt megtudjam tenni egy tehetősebb időmennyiséget kellett rászánnom a különböző elemek, bővítő lapok, magának az Arduino családnak és a szenzorok működésének a megismerésére. Azt tartottam szem előtt, hogy olyan ötlettel áljak elő, melyet később még tovább tudok fejleszteni, ha esetleg kellő motivációt érzek magamban, ne alkossak meg egy teljesen lezárt szerkezetet, hisz az életben minden folyamatosan fejlődik.

Arra a következtetésre jutottam végül, hogy az Arduino Mega 2560-as vezérlőlapot fogom majd használni, amely ATmega2560-as mikrovezérlővel rendelkezik. Ez a lap elegendő számítási kapacitás mellett biztosít számomra akkora mennyiségű digitális és analóg csatlakozót, mellyel akár a megvalósított funkciókat, meg is sokszorosíthatom majd a jövőben. A robotom működésének kezdetben 3 üzemmódot terveztem, mely a projektem végén kiegészült egy 4.-el is. A végső lista úgy nézett ki, hogy a robotautóm tudjon fényforrást követni, ha épp elébe kerül egy, akadályt kikerülni, amire később különböző feltérképező programokat lehet majd építeni, illetve képes legyen egy elébe húzott fekete vonalnak követni az útját, ezáltal biztosítva az okot, egy esetleges talaj feltérképezéshez szükséges szenzor felszerelését. Ezeken kívül bele szerettem volna még tenni egy szabadon irányítható módozatot is, ugyancsak továbbfejlesztési pillérként, ha valamilyen információt szeretnénk nyerni a környezetből, jó alapokat biztosíthat egy ilyen programrész. Ezen üzemmódok között pedig egy olyan váltási lehetőséget akartam megvalósítani, mely hanyagolja az internet és telefonok világában elterjedt módozatokat, mint amilyen a Bluetooth vagy WiFi kapcsolat, így

egy hagyományosabb, a hétköznapi életben már szinte lassan kihalni látszó módszert vettem elő, mégpedig az IR távirányítót. Ez az egyszerű és nagyszerű megoldás lehetőséget biztosít, hogy kis költségvetésből egy teljesen új szerkezetet hozzunk létre mely egy zárt rendszert alkot, és így nem kell vesződni a különböző kiegészítő elemek vásárlásával, vagy rendelkezésével, hogy teljes élvezeti faktort biztosítsunk a robotunknak. Hiszen egy távirányító nem kerül nagy költségekbe, még egy programozási nyelvet futtató telefon, vagy számítógép akár több száz eurót is felemészthet.

1.2. A szakdolgozat célja

A szakdolgozat célja bemutatni az ismerkedésemet az Arduino programozási felülettel, és ismertetni a robotfejlesztésem során lezajló folyamatokat. Első feladatul azt tűztem ki, hogy megismerkedjek magával az Arduino vezérlőlappal, melynek az Arduino Mega 2560-at választottam. Felfedezni az egyes lábakra való hivatkozást, alapkapcsolásokat végezni. Következő feladat, hogy a megvalósításhoz szükséges bővítő lapok működését is megismerjem. Ezek a 16x2-es LCD kijelző, az L298N motorvezérlő, az IR szenzor, a HC-SR04 ultrahangos szenzor, illetve a fényforrások érzékeléséhez szükséges fényérzékelő kapcsolások megismerése. A mikrovezérlővel és az egyes kiegészítő lapokkal történő ismerkedés egybe vehető, hiszen mindegyik elemet magán a vezérlőn keresztül tesztelek, próbálok ki. Az egyes részfolyamatok tesztelése után össze kell állítani a megfelelő testet, mely összetartja ezeket az elemeket és egy működő robotautót alkotnak. Ehhez meg kell tervezni az egyes elemek egymáshoz történő elrendezését, figyelembe véve, hogy a jövőben további elemek kerülhetnek rá a testre és kialakítani a megfelelő kapcsolatokat a megfelelő helyeken. Az összeállítást követően működésre kell bírni a szerkezetet, hogy pontosan akkor és azt végezze, melyet előláttam neki, össze kell hangolni az egyes elemek működését. Ez után teszteléseket kell végezni, hogy a legtöbb talajtípuson a legtökéletesebben működjön az autó, a fordulásokat beprogramozni, az időzítéseket beállítani. Végezetül levonni a következtetéseket, összefoglalni a tapasztalatokat.

2. Szakirodalmi áttekintés

2.1. Mikrovezérlők története

A fejlődés során az egyik fontos mérföldkő az elektronikus számítógép feltalálása volt. Az ENIAC (Electronic Numerical Integrator And Computer) a Manhattan-terv keretében készült. A gépet a Pennsylvania egyetemen építették, a munkát 1946-ban fejezték be. A munkában oroszlánrészt vállalt Neumann János. Ma is az általa kialakított elven működnek az asztali számítógépek. Olyan előrelátó és humánus volt, hogy elvét sosem engedte szabadalmaztatni. Azért, hogy ezt megakadályozza egy publikációban nyilvánosságra hozta a Neumann-elméletet, ami a szabadalmi védettséget lehetetlenné tette. Az ENIAC 18 ezer elektroncsövet tartalmazott, több mint 100 kW elektromos energiát fogyasztott és 450 m^2 helyet foglalt el (több mint 30 m hosszú termet építettek az elhelyezéséhez). A gép tömege 30 tonna volt, megépítése tízmillió dollárba került. Három nagyságrenddel gyorsabb volt, mint a relés számítógépek: az összeadást 0,2 ms, a szorzást 3 ms alatt végezte el. A programja azonban fixen be volt huzalozva a processzorba és csak a villamos csatlakozások átkötésével lehetett megváltoztatni. Az elektroncsövek megbízhatatlansága miatt a gép csak rövid ideig tudott folyamatosan működni. Az ENIAC-ot ballisztikai és szélcsatorna-számításokra használták. Egy útvonal kiszámítása a gépnek 15 másodpercig tartott, ugyanez egy szakképzett embernek asztali számológéppel 10 órás munka volt.

A fejlődés következő állomását a W. Schockley, W. H. Brattain és J. Bardeen által Bell laboratóriumában feltalált tranzisztor szolgáltatta. A tranzisztort 1947-ben fedezték fel és 1948-ban dobták a piacra. A találmány jelentőségét mutatja, hogy 1976-ban Nobel-díjat kaptak a feltalálói. A tranzisztor kis méretével és fogyasztásával, valamint hosszú élettartamával hamar kiszorította az elektroncsöveket. Megjelentek az első telepes, hordozható készülékek. A tranzisztor feltalálásával megjelentek a második generációs számítógépek. A kis méret lehetővé tette a nyomtatott áramkörök kialakítását. Az első nyomtatott áramkör 1958-ban jelent meg. Ez a megoldás kiküszöböli a vezetékes összekötésekből származó hibákat, nagymértékben növelve ezzel az áramkör megbízhatóságát.

Az integrált áramkörök megjelenése továbbvitte ezt a fejlődést. Az integrált áramkörben ugyanis egy hordozó szilícium lapkán egyetlen gyártási folyamatban alakítják ki az alkatrészeket, és az ezeket összekötő vezetőpályákat. Egyes szakemberek az integrált áramkörök feltalálásának jelentőségét a könyvnyomtatáséhoz mérik. A tetszés szerinti darabszámban, több nagyságrenddel olcsóbban és nagyságrendekkel megbízhatóbb minőségben előállítható elemek az élet minden területén nagy változásokat idéztek elő. Az első integrált áramkört 1959-ben készítették, de csak 1962-ben került kereskedelmi forgalomba. Az integrált áramkörök hatalmas fejlődést produkáltak (kezdve) a néhány alkatrészt tartalmazó SSI (Small Scale Integration) áramköröktől a több millió tranzisztort tartalmazó nagymértékben integrált VLSI (Very Large Scale Integration) áramkörökig. Gábor Dénes a nagy integráltsági fokú technológiát a XX. század legfontosabb találmányai között említi.

Ezek az integrált áramkörök vezettek a harmadik generációs számítógépek kialakulásához. A tranzisztorokat kiszorítják az integrált - egyelőre az alacsony és közepes integráltsági fokú - áramkörök. Megváltozik a

gépek struktúrája, átalakulnak funkcionális egységei, s kialakul a fejlett, egységes csatornarendszer, amely közvetlen kommunikációs kapcsolatot biztosít az egységek között.

Az 1971-es esztendő fordulatot hoz a számítógépek történetében, az Intel cég piacra dobja az első mikroprocesszort 4004-es néven. Ez a processzor 4 bit széles adatokkal és 8 bit széles utasításokkal dolgozott. Külön adat (1kB) és programmemóriával (4kB) rendelkezett. A 4004 46 utasítást ismert, 2300 tranzisztort tartalmazott és 16 lábú DIP tokba szerelték. 1972-ben kiadták a kibővített változatát 4040 néven. Az utasításkészletét és a memóriáját megnövelték az előzőhöz képest. Még ebben az évben megjelenik az immár 8 bites 8008-as mikroprocesszor.

A Texas Instruments az Intel 4004/4040 processzorát követve kiadja a 4 bites TMS 1000-t. Az 1974-ben megjelent TMS 1000 volt az első mikrovezérlő. Ez egyetlen tokba építve tartalmazott adatmemóriát (RAM), programmemóriát (ROM) és I/O egységet, lehetővé téve a működést külső kiegészítő áramkörök nélkül. A mikrovezérlő tartalmazott egy 4 bites akkumulátort, egy 4 bites Y és egy 2-3 bites X regisztert, amellyel a belső 64, illetve 128 félbájtos RAM-ot lehetett megcímezni, valamint egy 1 bites státuszregisztert. A 6 bites programszámláló kombinálva egy 4 bites lapozó regiszterrel és opcióként 1 bankváltó bittel 1, illetve 2kB-os memória (ROM) címzését tette lehetővé. A szubrutinok, elágazások kezelésére tartalmazott egy 6 bites vermet és egy 4 bites lapozó puffert. Érdekessége a programszámlálónak, hogy nem számláló, hanem egy visszacsatolt léptetőregiszter volt, így az utasítások a memóriában nem egymás után sorban következtek. Utasításkészlete úgy épült fel, hogy tartalmazott tizenkét 8 bites huzalozott (fix), valamint 31 darab 16 bites felhasználó által mikroprogramozott utasítást, amelyet egy PLA valósított meg. A hardveresen huzalozott utasítások végrehajtási ideje 1 gépi ciklus volt, megszakítási lehetőséggel nem rendelkezett.

Szintén 1974-ben jelenik meg a 8008 továbbfejlesztett változata a 8080-as Intel mikroprocesszor, amely 8 bites adatbuszt és 16 bites címbuszt tartalmazott. A belsejében hét 8 bites regisztert (A-E, H, L), tartalmazott, amelyekből a BC, DE, és HL összekapcsolhatók voltak 16 bitessé. A 16 bites veremmutatóval egy 8 mélységű verembe lehetett elmenteni a visszatérési címeket. A 8080-as processzor volt az első széles körben elterjedt személyi számítógép az Altair 8800-as agya. Az Intel később ezt tovább fejlesztette, és 1976-ban kiadta a 8085-ös processzort. Ebbe beépítettek két új utasítást, amellyel a három szintén új fejlesztésként megjelent megszakítási vonalat lehetett tiltani vagy engedélyezni, valamint kapott soros vonali kivezetéseket is. A hardver is egyszerűsödött, az új processzor már csak egyetlen +5 V - os tápfeszültséget igényelt.

A Zilog cég 1976-ban kiadja, a sokáig nagy népszerűségnek örvendő Z-80 - as processzort, ami a 8080 - as továbbfejlesztésének tekinthető (volt Intel mérnökök tervezték). A Z80 - as 8 bites adatbusszal és 16 bites címbusszal rendelkezett és tudta futtatni a 8080as összes utasítását, amelyet kibővítettek 80 új utasítással (1, 4, 8 és 16 bites műveletek, páros című blokkmozgató és I/O utasítások). A regisztertömböt két bankra osztva megduplázták, amely növelte a sebességet. A Z-80 - ba beépítettek két indexregisztert (IX, IY), valamint kétszintű megszakítási rendszert kapott. Az eredeti Z-80 órajel frekvenciája 2,5 MHz volt a Z80-H típusé 8MHz, a CMOS verzióé (Z80-C) pedig 10 MHz.

Nem sokkal a 8080 kiadása után 1975-ben a Motorola bemutatta 6800-as processzorát. Néhány Motorola tervező kilépve a cégtől a MOS Technologies - nél kezdett el dolgozni (később a Commodore megvásárolta),

és itt jött létre a 650x sorozat. Ebbe a sorozatba tartozott a 6501 (láb kompatibilis a 6800-as processzorral) és a 6502 (a korai Commodore, Apple és Atari gépekben használták). A 6800-as változatai közül a 6510-et a Commodore 64-ben, a 6507-et pedig az Atari 2600-ban alkalmazták. A 650x sorozat az úgynevezett little endian technikát használta (a cím alsó bájtját hozzá lehetett adni az indexregiszterhez, miközben a felső bájtot lehívtuk), utasításkészlete teljesen eltért a 6800-tól. Az ára mintegy negyede volt a 6800-hoz képest (kevesebb, mint 1000 dollár). Ez az alacsony ár tette lehetővé, hogy a korai személyi számítógépek processzorává váljon. 1977-ben a Motorola jelentősen továbbfejlesztette a 680x-es sorozatot, és kiadta a 6809-es processzort. A 6809-es két 8 bites akkumulátort (A és B) tartalmazott, amit lehetett egyetlen 16 bites regiszterként (D) használni. A processzor két indexregisztert (X, Y) és két veremmutatót (S, U) tartalmazott, amivel nagyon sokrétű címzési módot lehetett megvalósítani. A 6809 64kB memória kezelésére volt alkalmas. További jellegzetessége, hogy ez volt az első 16 bites aritmetika, ami tartalmazott szorzó utasítást.

Az Advanced Micro Devices (AMD) is színre lép az Am2901-es processzorával. Ez egy úgynevezett bitszelet mikroprocesszor, ami azt jelenti, hogy felépítése moduláris: tetszőleges szélességű (nx4bit általában) ALU építhető, tetszőleges utasításkészlet alakítható ki. Az Am2901 4-es bitszelet processzor volt 16 regiszterrel és 4 bites ALU - val. A vezérlése mikroprogramozott volt, a belső ROM - ból olvasódtak ki az egyes utasítások végrehajtásának lépései. Később az Am2903 már hardveres szorzót is tartalmazott. Az AMD alkotta meg elsőként a lebegőpontos műveletek elvégzésére alkalmas matematikai társprocesszort. Az AMD9511 1979-ben készült el. 32 bites lebegőpontos műveleteket tudott végezni. A 16 bites ALU képes volt összeadásra, kivonásra, szorzásra, osztásra, szinusz és koszinuszt számolására, műveletvégzési sebessége minden akkori processzornál nagyobb volt.

1977-ben az Intel is elkészíti a maga mikrovezérlőjét, a 8048-at, ami alacsony árával és kis méretével igen jó ötletnek számított. Az adatmemória a tokon belül volt kialakítva, a programmemóriát viszont kívülről kellett csatlakoztatni (Harvard-architektúra volt, bár a program és az adat ugyanazt a címvonalat használta). A 8048-at felváltotta a 8051 és a 8052, amely már beépített programmemóriát (ROM) tartalmazott. A 8051-es már rugalmas, két bájt széles utasításkészletet tartalmazott 8 bit széles regiszterekkel és akkumulátorral. Az adatmemória 128 bájtos volt, amit direkt vagy indirekt regiszteres címzéssel lehetett elérni. Ezen kívül felette tartalmazott még egy 128 bájtos részt, amit a 8052-nél csak indirekt lehetett elérni (veremként használták). Külső memóriát a 8048-hoz hasonlóan el lehet érni közvetlenül (256 bájtos lapokban az I/O portokon keresztül), vagy a 16 bites DPTR címregiszterrel. A közvetlenül elérhető adatterület feletti 32 hely bitenként címezhető. Az adat és a programmemória azonos címterületen osztozik (a címvezetékek is, ha külső programmemóriát alkalmazunk). Habár komplikált ez a memóriaszervezés, mégis rugalmas beágyazott tervezést enged meg. Népszerűségét bizonyítja, hogy 1988-ig több mint 1 billiót adtak el belőle.

A PIC mikrovezérlők gyökere a Harvard egyetemre (Harvard-architektúra) nyúlik vissza, a Védelmi Minisztérium projektjének keretében készült. A Harvard-architektúrát először a Signetics 8x300-ban használták, majd a General Instruments adaptálta és a perifériaillesztő vezérlőiben (peripheral interface controller, azaz PIC) alkalmazta. A gyártás később (1985) az arizoniai Microchip Technology-hoz került, s a PIC lett a cég fő terméke. A PIC mikrovezérlők típusválasztéka mára igen széles lett: PIC10x, PIC12x, PIC14x,

PIC16x, PIC18x, PIC24x. Ezekben a sorozatokban szinte minden alkalmazásra találhatunk megfelelő mikrovezérlőt. A 12-es sorozat például nagyon egyszerű mikrovezérlő, 6 I/O lábat, 25-128 regisztert, és egy beépített 8 bites számláló/időzítő modult tartalmazott. A 18-as sorozat pedig 16-70 I/O lábat, 256-3936 regisztert, több 8/16 bites számláló/időzítőt, számos perifériát: A/D átalakító, címezhető szinkron/aszinkron soros port, SPI és I2C busz, összehasonlító/kiolvasó/PWM modul, analóg komparátor, stb. tartalmaz. A Microchip nagyon jó kézikönyveket és alkalmazási leírásokat bocsát a felhasználók számára (ezek természetesen angol nyelvűek). A fejlesztéseket mindig úgy végzi, hogy előtte kikéri a felhasználók véleményeit, s ennek alapján készül el a következő generációs PIC. Ezen mikrovezérlők ára nagyon kedvező, 200-1000 din között mozog, a tanulók otthoni felhasználásra is megvásárolhatják nagyobb anyagi ráfordítás nélkül. A Microchip a programfejlesztéshez szükséges MPLAB programot ingyen biztosítja a felhasználóknak. [1]

2.2. A mikrovezérlők alkalmazási területei

Integráltságát tekintve a mikrovezérlő a mikroprocesszor és a PLC között áll. A mikroprocesszor – a fentiek értelmében – nem tartalmazza a működéséhez szükséges perifériális áramköröket, program- és adattárat, egyéb perifériákat, a PLC pedig már egy kész, tokozott eszköz, amely tartalmazza a működéséhez szükséges összes kiegészítő áramkört, mint a tápellátás áramköreit, a kimeneti meghajtó, a bemeneti illesztő áramköröket stb. (és egy mikrovezérlőt).

Más szóval, ha az elvégzendő munkát tekintjük, egy mikrovezérlős feladat megoldása – hasonlóan egy mikroprocesszoroshoz – elektronikai tervezéssel és kivitelezéssel kezdődik, míg egy PLC-s feladat esetében az egyes elemeket csak vezetékeznünk kell. Ezután jöhet a végrehajtandó program megírása. Ez a feladat mindhárom esetben megoldandó. A mikroprocesszorokat nagy számítási igényeknél alkalmazzuk, illetve olyan helyen, ahol a nagyfokú rugalmasság a cél. A mikroprocesszorral tervezett áramkörökben a perifériális áramköröket is külön - külön meg kell terveznünk, míg a mikrovezérlőkben a gyakran használt perifériális illesztő már a tokba van integrálva. Persze nem minden elem, például egy analóg bemeneti jelhez az A/D átalakító a tokban belül helyezkedik el, azonban az analóg jel méréséhez szükséges egyéb (pl. bemeneti illesztő) áramköröket már nekünk kell megtervezni. Így – kisebb célfeladatok megoldása esetében – a mikrovezérlő rendelkezik a mikroprocesszorok programozásának rugalmasságával, kiegészítve azzal a könnyítéssel, hogy a tokba integrált perifériális elemeket nem kell nekünk megterveznünk, hanem azokat csak használnunk kell.

Összefoglalva, mikor is használunk mikrovezérlőt:

- A feladat PLC-vel nem oldható meg
- Nagyobb darabszámoknál, amikor a termék kifejlesztésének költsége egy termékre levetítve már nem számottevő.
- Jól megfogalmazott feladatokat kell „csak” végrehajtani, mivel egy mikrovezérlő belső programtárat tartalmaz, a belső program korlátozottan változtatható.
- Lehető legkisebb helyigény

• Diszkrét áramköri elemekből a feladat már nehézkesen, sok elemből oldható meg. (Sokszor olcsóbb egy mikrovezérlő alkalmazása, mint a diszkrét logikai elemekből összeépített nyomtatott áramköri lap cm^2 ára.)

Mikor használunk PLC-t:

• Kis darabszámoknál (1-2) darab. Ekkor nem éri meg az áramköri fejlesztés költsége.

• Rövid határidőknél

Mikor mikroprocesszort (számítógépet):

• Nagy számítási igényeknél,

• Kis darabszámoknál

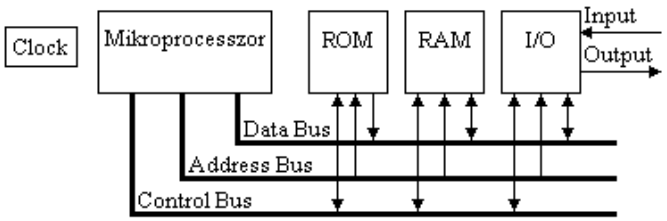
• Egyedi perifériális igényeknél,

• Mindig változó feladatok végrehajtásánál, kihasználva a külső és lényegesen nagyobb programtár adta lehetőségeket.

A mikrovezérlős rendszerek tervezésében sokszor egy plusz nehézséggel is szembe kell néznünk, nevezetesen, hogy mikrovezérlős alkalmazás tervezésekor a hardveres és a szoftveres szakember már nem különülhet le, a két feladatot szimultán kell kezelnünk. [2]

2.3. A mikroszámítógépek felépítése

A mikroszámítógépek alapja a mikroprocesszor. Elemei a mikroprocesszor, memória, és input/output eszközök. A komponenseket valamilyen buszrendszer köti össze, amelyen az egységek közötti adatcsere zajlik. A mikroszámítógép blokkvázlata az 1. ábrán látható. [3]



1. ábra: A mikroszámítógép felépítése

2.4. Az Arduino platform

Az Arduino platform története 2003-ig nyúlik vissza. Kolumbiában Hernando Barragán létrehozta a Wiring platformot diplomamunkájaként, majd ezt nyílt forráskód alatt tette közzé. A platform azóta is aktív, bár nem örvend akkora sikernek, mint az Arduino. Maga az Arduino platform 2005-ben született meg Massimo Banzi és Casey Reas munkájának gyümölcseként. A platform a nevét az Olaszországi Ivrea városának történelmi alakjáról Arduin of Ivrea-ról kapta. Az Arduino szó magyarul “bátor barát” - ot jelent.

Az évek alatt több, különböző modell jelent meg a platformból, ezek közös jellemzője (ami a hivatalosakat illeti) az, hogy Atmel mikrovezérlőket alkalmaznak, valamint egy egységes programozó környezetből használhatóak. Az összes modell kapcsolási rajza és a szoftverek forráskódja elérhetőek a hivatalos

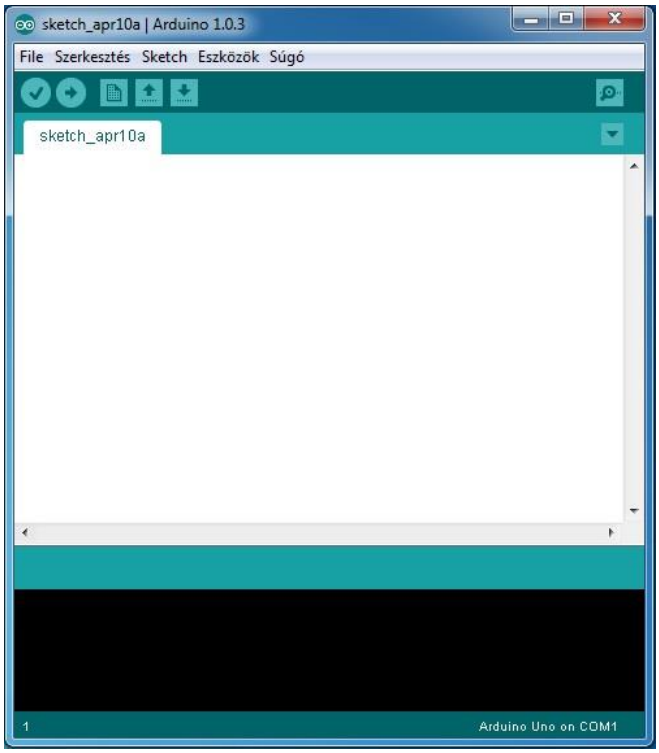
weboldalon: https://www.arduino.cc. A platform nyíltságából adódóan léteznek klón lapok is, amelyek tudásban, minőségben igencsak eltérhetnek az eredeti lapoktól.

Az Arduino hátránya az, hogy önmagában „csak” egy mikrovezérlő. Mindenképpen kell hozzá egy számítógép, amin az ingyenesen letölthető Arduino IDE szoftver segítségével végezhető el a programozás egy USB kábelen keresztül. Ez pillanatnyilag Windows, Linux és Mac OS X alá tölthető le. Ezen kívül a fejlesztéshez mindenképpen szükségesek más elektronikai komponensek is, amiket be kell szereznünk. A készítők szerencsére mára már több nyelven is piacra dobták az Arduino Starter KIT-et, ami egy Arduino UNO panelen és a 170 oldalas Project Book-on kívül számos alkatrészt is tartalmaz a kezdők számára.

Arduino típusok két nagy csoportra oszthatóak. A “Board” névvel a fő fejlesztő paneleket, vagy alaplapokat jelzik, míg és “Shield” (magyarul pajzs) a kiegészítő vagy bővítő lapok elnevezése. Ezek a kiegészítők plusz tudással vértezik fel az alaplapokat és a szabványosított csatlakozási felületnek köszönhetően megengedik a többszintes bővítést. A típusok listája viszonylag sűrűn változik, mivel egyes modellek népszerűbbek, mint mások, valamint a fejlesztések is folyamatosak. Az aktuális lista egyébként kapcsolási rajzokkal együtt a https://www.arduino.cc/en/Main/Products címen érhető el. [4]

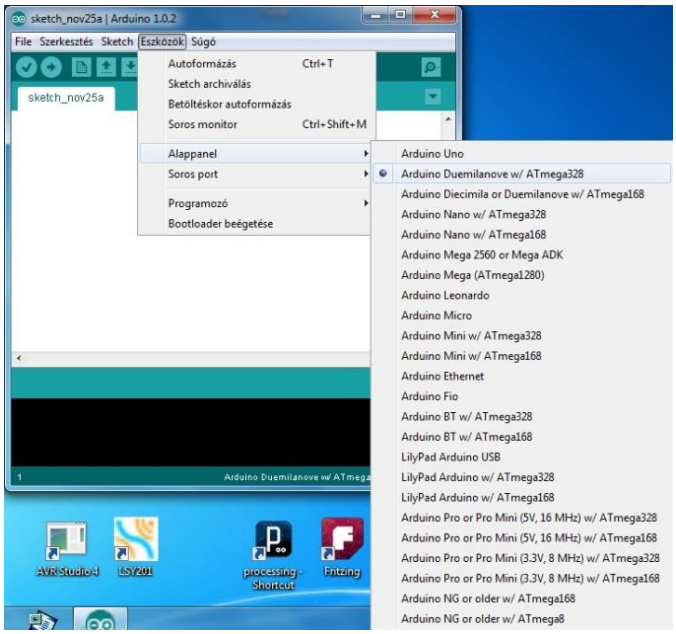
2.5. Arduino fejlesztőkörnyezet

Az Arduino fejlesztőkörnyezet legfrissebb verziója letölthető az alábbi linkről: http://arduino.cc/en/Main/Software.
A programot először elindítva az alábbi képernyő fogad:



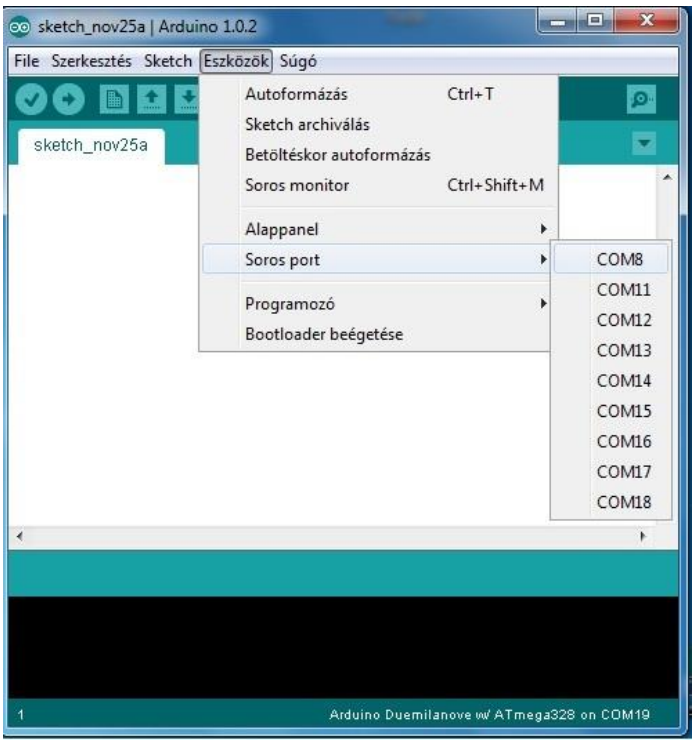
2. ábra: Az Arduino IDE nyitóképernyője

A File/Preferences/Editor Language menüpontban állíthatjuk át Magyar nyelvűre a programot. Mielőtt az Arduino panelt elkezdenénk használni, ahhoz hogy programozni tudjuk, az Eszközök/Alappanel menüpontban be kell állítani, hogy milyen típusú panelt használunk.



3. ábra: A programozni kívánt lapunk megadása

Az Eszközök/Soros port menüpontban pedig azt kell beállítani, hogy a panel melyik COM portra van csatlakoztatva. Fizikálisan viszont USB csatlakozón keresztül történik programfeltöltés, csak az IDE érzékeli úgy, hogy COM portot használunk.



4. ábra: A programozói csatlakozó megadása

A fentiek beállítása után a panelt már tudjuk programozni. Az eszközsorban az alábbi 7 gomb található:

5. ábra: Az Arduino IDE eszközsora

- Ellenőrzés: Mielőtt a programot az Arduino panelba töltenénk, le kell fordítanunk. Ezzel a gombbal fordítható le a kód és ellenőrizhető, hogy a programunk hibamentes-e.
- Feltöltés: A lefordított kód feltöltése az Arduino-ba.
- Új: Új projekt létrehozása.
- Megnyitás: Korábban létrehozott projekt megnyitása
- Mentés: A jelenlegi projekt elmentése.
- Soros Monitor: Az Arduino panel által küldött soros adatok megjelenítése egy terminálablakban.

A program felépítése:
Az Arduino programokat vázlatnak/skicc - nek (Sketch) hívják.
A program nem más, mint az Arduino által végrehajtandó utasítások sorozata.
A program három fő részből áll:

- Változók megadása
- Setup () – Általános Beállítások
- Loop () – Főprogram [5]

2.6. Főbb Arduino lapok (Boardok) és bővítő áramkörök (Shieldek)

2.6.1. Arduino Lapok

Az Arduino Board - ok többféle változatban készülnek, amelyek méretben, a belső memóriában, számítási kapacitásban, a be- és kimenetek számában különböznek. Vannak, amelyek rendelkeznek beépített Ethernet, vagy Bluetooth csatlakozóval.

Jelenleg is forgalomban lévő modellek:

- Arduino ADK
  A Mega androidra készített, eszközfejlesztéshez kitalált változata. A hardver azonos a Mega-val, csak ez támogatja USB perifériák fogadását is.
- Arduino DUE

  Az első ARM architektúrára épülő Arduino modell. 512kB kód memória, 96kB RAM, 84 MHz órajel, 2 csatornás analóg kimenet, 12 analóg bemenet, 54 digitális I/O. 3,3V-os logikai jelekkel dolgozik a többi modellel ellentétben.

- Arduino Fio

- Arduino Leonardo

Hasonló paraméterekkel rendelkezik, mint az Uno, azonban ez a modell valós USB támogatással rendelkezik, ami lehetővé teszi, hogy billentyűzetet vagy egeret emuláljon.

- Arduino LilyPad

  Az Uno képességeivel azonos tudású modell; kifejezetten viselhető elektronikai eszközök céljára, ruhákba épített elektronika készítéséhez lett kifejlesztve.

- Arduino Mega 2560

  256kB kód memória, 8kB RAM, 54 digitális I/O és 16 analóg bemenet.

- Arduino Nano

  Uno hardver tulajdonságaival megegyezik, a fizikai mérete viszont kisebb.

- Arduino Uno

  A legnépszerűbb modell, 32kB kód memória, 2kB RAM, 13 digitális I/O és 6 analóg bemenet.

- Arduino Yún

  Az első hibrid Arduino, Linux rendszert futtat és beépített WLAN támogatással rendelkezik.

- Arduino Esplora

  Leonardo alapú modell, kifejezetten játékvezérlők fejlesztésére.

Már nem gyártott modellek:

- Arduino Duemilanove
- Arduino Ethernet
- Arduino BT
- Arduino Mega

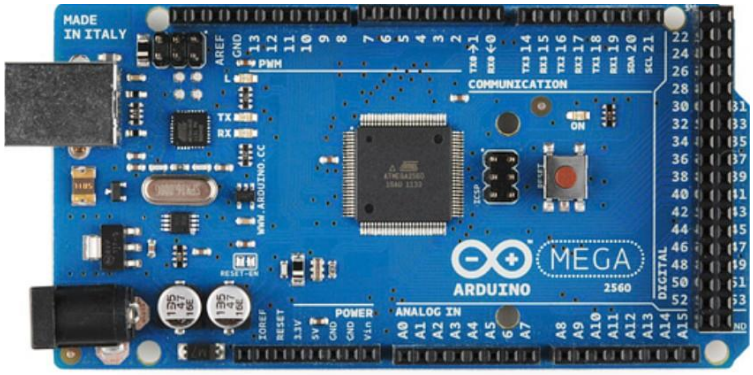
2.6.2. Arduino bővítő áramkörök

Az Arduino - k funkcionalitását könnyedén kiegészíthetjük az úgynevezett Shield - ekkel, amelyek a Board - okhoz egyszerűen illeszthető elektronikai áramkörök. Ezek segítségével az Arduino-t akár közvetlenül az internetre csatlakoztathatjuk, motorokat vezérelhetünk vele, vagy WiFi hálózatra kapcsolódhatunk. A jelenlegi hivatalos Arduino Shield - ek:

- Arduino Ethernet Shield
- Arduino WiFi Shield
- Arduino Proto Shield
- Arduino GSM Shield
- Arduino Motor Shield

Ezeken kívül több cég gyárt világszerte Arduino kompatibilis Shield-eket, különböző célokra. A különböző alkatrészeket, melyek külön Arduinora lettek legyártva, viszont máshol is felhasználhatóak, szintén, bizonyos szinten Shieldek – nek tekinthetjük. [6]

2.7. Arduino Mega 2560



6. ábra: Az Arduino Mega 2560 kinézete

Az Arudino Mega 2560, az Arduino Megának egy felújított változata, mellyel lecserélték az elődjét. Rendelkezik tápcsatlakozóval, mellyel táplálni lehet a berendezést és USB programozási vezetékkel látták el. Tulajdonságai a következőek [7] :

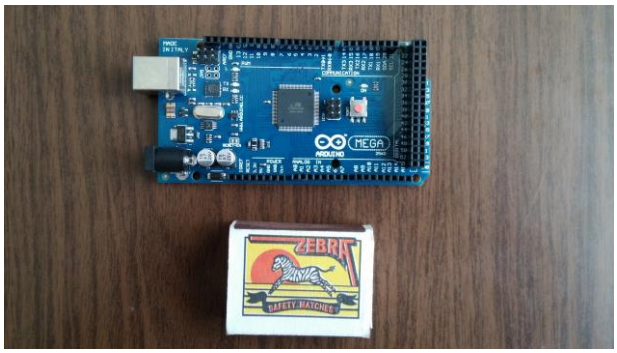
1. táblázat: Az Arduino Mega 2560 tulajdonságai

Mikrokontroller	ATmega2560
Működési feszültség	5V
Bemeneti feszültség (szükséges)	7-12V
Bemeneti feszültség (maximális)	6-20V
Digitális bemeneti/kimeneti lábak	54 (mely közül 15 biztosít PWM kimenetet)
Analóg bemeneti lábak	16
Egyenáram a bemeneti/kimeneti lábnak	20 mA
Egyenáram a 3.3V lábnak	50 mA
Belső memória	256 KB melyből 8 KB - ot használ a bootloader
SRAM	8 KB
EEPROM	4 KB
Számítási sebessége	16 MHz
Hossza	101.52 mm
Szélessége	53.3 mm
Tömege	37 g

3. Problémamegoldás

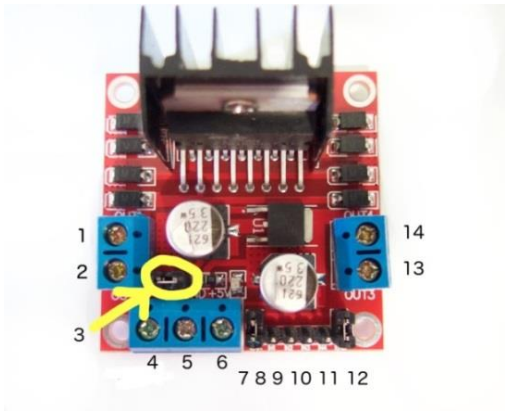
3.1. A probléma megoldásához felhasznált főbb alkatrészek rövid bemutatása

- Arduino Mega 2560 fejlesztői lap
  Azért választottam ezt a vezérlőt a számos Arduino közül, mivel ennek megfelelő számú bemenete van egy terebélyesebb projekt megvalósításához, s lehetőséget nyújt a munkám jövőben történő továbbfejlesztésére. Valamint azért is jó döntésnek gondoltam ezt, mivel megtörténhetett volna, hogy a népszerű UNO fejlesztői lap 32 kB – os belső memóriája nem lett volna elegendő a programom futtatásához. Ezzel szemben a Mega flash memóriája 256 kB – os, ami miatt nem szükséges a hely miatt aggódnom.



7. ábra: Arduino Mega 2560 mérete egy gyufásdoboz mellett

- L298N, két motor vezérlésére képes bővítő áramkör
  Ez az alkatrész azért szükséges, hogy a két DC motoromat, amelyek hajtsák az autómat, vezérelni tudjam a mikrovezérlőmön keresztül, mivel magában az fejlesztői lapnak nincs megfelelő tápfeszültsége a motorok meghajtásához. Ez nem egy túl nagy lap, egész jól fel van építve és 5 - 35V-os tápfeszültségről képes meghajtani a két kereket. A logikai műveletek elvégzését 5V - ról végzi, miközben maximum 25W - os teljesítményt képes leadni. Az egyes motorokat 2A - 2A-al tudja ellátni. Az áramköri lap alapja az L298N - es IC, s kinézete a következő:



8. ábra: Az L298N motorvezérlő felépítése

A képen látható csatlakozó kiosztás:

1. DC motor 1 "+" vagy step motor A+
2. DC motor 1 "-" vagy step motor A-
3. 12 V - os áthidalás – távolítsa el, ha a motorokat tápláló feszültség nagyobb, mint 12V egyenáram. Ezzel engedélyezi az 5V-os, lapra épített feszültségszabályzót.
4. A motorok tápfeszültségének helye, maximum 35V. Távolítsa el a 12V-os áthidalást, ha a tápfeszültség nagyobb, mint 12V.
5. GND – Földelés
6. 5V-os kimenet, ha a 12V-os áthidalás csatlakoztatva van, akkor ideális az Arduino lap táplálásának helyettesítésére (stb.)
7. A DC motor 1 engedélyező áthidalása. Hagyja rajta abban az esetben, ha step motort vezérel. Csatlakoztassa PWM kimenetre, ha a DC motor sebességét szeretné vezérelni.
8. IN1 – Motort vezérlő kimenet
9. IN2 – Motort vezérlő kimenet
10. IN3 – Motort vezérlő kimenet
11. IN4 – Motort vezérlő kimenet
12. A DC motor 2 engedélyező áthidalása. Hagyja rajta abban az esetben, ha step motort vezérel. Csatlakoztassa PWM kimenetre, ha a DC motor sebességét szeretné vezérelni.
13. DC motor 2 "+" vagy step motor A+
14. DC motor 1 "-" vagy step motor A- [8]

- 2 Egyenáramú (DC) motor kerekekkel
  Ezek az elemek képezik az autóm hajtását. Külön okos autóra vannak előlátva, Arduino, Raspberry Pi, stb. feladatokhoz. Azért döntöttem úgy, hogy ezeket a motorokat használom, mivel jártak hozzá kerekek, így megkönnyítve a dolgomat, hogy ne kelljen külön ezeknek a beszerzésével foglalkoznom. Valamint ezek egyforma, külön erre a célra legyártott meghajtó egységek, tehát a teljesítményük se tér el, ellenben ha külön kerestem volna meghajtást, akkor ügyelnem kellett volna erre is, hogy 2 darab egyforma erejű motort találjak.

9. ábra: A DC motor a kerékkel együtt

2. táblázat: A DC motor tulajdonságai

Feszültség	DC 3 V	DC 5 V	DC 6 V
Áram	100 mA	100 mA	120 mA
Csökkentés mértéke	48:1		
Fordulat percenként (Kerékkel)	100	190	240
Kerék átmérő	66 mm		
Autó sebesség (M/perc)	20	39	48
Motor tömege (g)	50		
Motor méretei	70 mm*22 mm*18 mm		
Zajosság	<65 dB		

Kerék paraméterei:

- Középső lyuk: 5.3 mm x 3.66 mm
- Kerék méret: 65x26 mm [9]

- HC-SR04 ultrahangos szenzor
  Tekintettel, hogy a robotautóm akadályelkerülő robotként is funkciónál, ezért szükség volt egy szenzorra, ami érzékeli az elébe kerülő akadályokat. Én az ultrahangos érzékelőt választottam erre a célra. Lehetett volna még megoldani ezt a problémát foto ellenállással és a visszaverődő fényt érzékelni akadály előkerülésekor, de én úgy gondoltam, hogy ez a megoldás profibb kivitelezést tesz lehetővé, illetve a foto ellenállásos módszert úgyis alkalmazni fogom a vonal követésénél, így változatosabb is lesz a projektem. Működési elve egyszerű, kibocsájt különböző hanghullámokat és megméri azok visszaérkezési idejét.

10. ábra: A HC-SR04 ultrahangos szenzor

A szenzor csatlakozásai a következőek:

- Vcc - 5 V tápfeszültség
- Trg - Távolságmérő impulzus bemenet (hanghullám érzékelés)
- Echo - Visszhang impulzus kimeneti (hanghullám kibocsájtás)
- GND – 0 V, föld [10]

3. táblázat: Az ultrahangos szenzor tulajdonságai

Működési feszültség	DC 5 V
Működési áram	15 mA
Működési frekvencia	40 Hz
Maximális távolság	4 m
Minimális távolság	2 cm
Mérési szög	15 fok
Érzékelő bemeneti jel	10uS TTL jel
Visszhang kimeneti jel	A bemeneti TTL emelőjel és a hatótávolság arányos
Méretek	45*20*15 mm

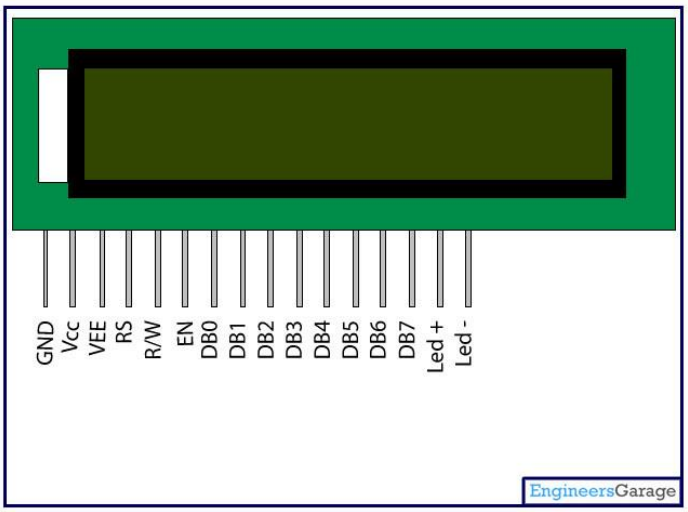
- IR szenzor, hozzá járó távirányítóval, Arduinohoz készítve
  Sok fontolgatás után, úgy döntöttem, hogy a távirányítás megvalósítását, ezzel a kész modullal fogom elérni, mivel távirányító és szenzor is jár hozzá, ráadásul az ár-érték aránya is egészen jónak mondható. Bár ez nem egy eredeti modul, Kínában gyártott, de nem is űrprojektet terveztem, így úgy gondoltam, hogy nem lesz probléma azzal, ha esetleg nem annyira nagy a pontosság.

11. ábra: Az IR távirányító és a hozzá tartozó érzékelő

A készlet adatai:

- A távirányító 17 gombos
- NEC protokolra épül
- A távirányító hatótávolsága legfeljebb 8 m
- Hullámhossza: 940 nm
- Frekvenciája: Kristály oszcillátoré: 455 kHz; IR hordozó frekvenciája: 38 kHz
- Működési szöge: 60 fok
- Statikus árama: 3-5 uA, Dinamikus árama: 3-5 mA
- A távirányító méretei: 8.5 x 4 x 0.65 cm (H x Sz x M)
- A távirányítóhoz szükséges elem: CR2025 - 3V [11] [12]

- 2x16 LCD kijelző 16 láb kivezetéssel, Arduinohoz készítve
  Ez a kijelző azt a célt szolgálja majd a munkámban, hogy különböző információkat juttasson a felhasználó számára, s így felhasználó barátiabbá tegye a robotautómat. Ezen fog megjelenni, hogy épp milyen funkciót lát el a robot, illetve ha sikerül továbbfejlesztenem, hogy berakjak egy meredekség érzékelőt, akkor itt fog az is megjelenni, hogy hány %-os dőlésszögű talajon áll a szerkezetem. [13]



12. ábra: Az LCD kijelző lábkiosztása

4. táblázat: Az LCD kijelző lábkiosztása

Láb szám	**Funkció**	**Név**
1	Föld (0 V)	Ground
2	Tápfeszültség; 5 V (4.7 V – 5.3 V)	Vcc
3	Kontraszt beállítás; egy változtatható ellenálláson keresztül	V_{EE}
4	Parancs regiszter kiválasztása, ha alacsony; és adat regiszter kiválasztása, ha magas	Register Select
5	Alacsony szinten regiszterbe írás; Magas szinten regiszterből olvasás	Read/Write
6	Adat küldés az adat lábra ha lemenőjel van adva	Enable
7	8-bit adat láb	DB0
8		DB1
9		DB2
10		DB3
11		DB4
12		DB5
13		DB6
14		DB7
15	Háttérvilágítás V_{CC} (5V)	Led+ (A)
16	Háttérvilágítás föld (0V)	Led- (K)

3.2. A problémamegoldáshoz szükséges alkatrészek és azok beszerzése

Miután meghatároztam, hogy mire lesz szükségem a probléma megoldásához, utána kellett járnom, honnan is fogom beszerezni az egyes elemeket. A szükséges alkatrészeket sajnos nem tudtam egy helyről megszerezni, s így csökkenteni a szállítási költségeket, hanem kisebb csoportokban voltam csak képes elvégezni a beszerzést. Magát az Arduinot, illetve a hozzá szükséges kiegészítő lapokat a http://www.kupujemprodajem.com internetes oldalról rendeltem meg (melyek nem eredetiek voltak, hanem klónok), még a hagyományosabb alkatrészeket elektronikai üzletből szereztem be. Ezen kívül volt egy-két elem, melyért nem kellett pénzt adnom, mivel már itthon a rendelkezésemre állt. A szükséges alkatrészeket és költségeiket (szállítási költség nélkül) lentebb ismertetem (5. táblázat):

5. táblázat: A problémamegoldáshoz felhasznált elemek

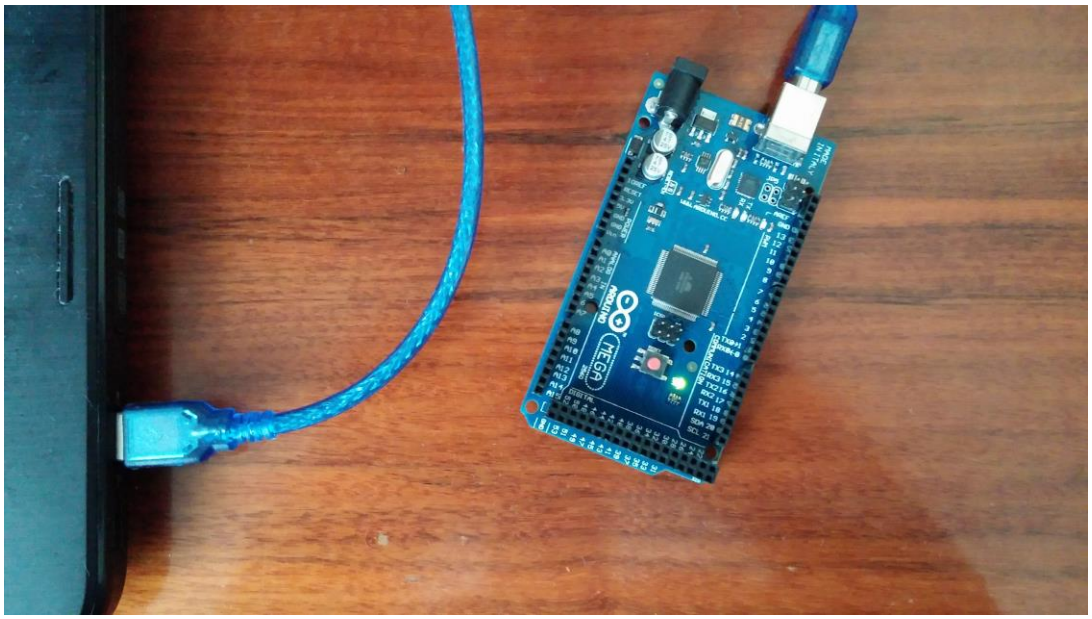
Elem neve	**Db.**	**Beszerzési hely**	**Költsége**
Arduino Mega 2560	1	KupujemProdajem (Milos) - Čačak	2.121,00 din
L298N motorvezérlő	1	KupujemProdajem (Djordje) - Újvidék	600 din
LCD 16x2	1	KupujemProdajem (Dragan) - Belgrád	420 din
HC-SR04 ultrahangos sz.	1	KupujemProdajem (Dragan) - Belgrád	250 din
IR szenzor távirányítóval	1	KupujemProdajem (Dragan) - Belgrád	375 din
DC motor kerékkel	2	KupujemProdajem (Dragan) - Belgrád	900 din
Vezetékek és csatlakozók	>40	KupujemProdajem több helyről	~300 din
Foto ellenállás	4	Elementa - Szabadka	~680 din
10 kΩ ellenállás	2	Itthon rendelkezésemre áll	Nem került semmibe
22 kΩ ellenállás	2	Elementa - Szabadka	~8 din
510 Ω ellenállás	2	Elementa - Szabadka	~8 din
Fehér LED	2	Elementa - Szabadka	~48 din
12 V tápforrás	1	Itthon rendelkezésemre áll	Nem került semmibe
Hátsó forgó kerék	1	Kínai üzlet	100 din
Iránytű, hátsó keréknek	1	Itthon rendelkezésemre áll	Nem került semmibe
Lezonit lap	1	Munkahelyemen szereztem be	Nem került semmibe
10 kΩ potenció méter	1	Itthon rendelkezésemre áll	Nem került semmibe
Próbalap	1	Itthon rendelkezésemre áll	Nem került semmibe
Kapcsoló	1	Itthon rendelkezésemre áll	Nem került semmibe
Tápforrás csatlakozó	1	Itthon rendelkezésemre áll	Nem került semmibe
Kábelkötegelő	2	Elektro Nagy – Ada	~12 din
Különböző csavarok	>25	BisProduct – Ada	~250 din
Műholdvevő távirányító	1	Itthon rendelkezésemre áll	Nem került semmibe
Összesített költségek:			~6072 din

3.3. Az alkatrészek és kapcsolások kipróbálása

Az alkatrészek kipróbálását, tesztelését egyszerű kapcsolásokkal végeztem. Arra voltam kíváncsi, hogy a frissen megérkezett alkatrészeim hogyan működnek, nem hibásak-e, nagyjából miket lehet velük csinálni. Végezetül jó ötletnek bizonyult ez a gondolatom, ugyanis egy-két elemnek már rögtön az első bekapcsolásnál kiderült a hátránya, ami abból adódott, hogy nem eredeti lapokkal akartam dolgozni, hanem klónokkal. Valamint később a végső robotom forráskódjában a tesztprogramjaim bizonyos részét, képes voltam felhasználni, s ezáltal csökkenteni a végső kódolással töltött időt.

3.3.1. Arduino Mega 2560

Ez volt az első dolog, amit teszteltem, hiszen, ha nem működik a programozói lapom, akkor a többi elemmel se tudok mit kezdeni. A mikrovezérlővel kapott USB kábel csatlakoztatása előtt telepítettem a hivatalos honlapról letölthető Arduino IDE programcsomagot, mellyel driveok is jártak. A csatlakoztatáskor, rögtön jelzett az operációs rendszer, hogy új eszközt talált, valamint a lap táplálása már el is kezdődött, melyet a felvillanó LED-ek jeleztek. Pár pillanat eltelte után telepítette is magát az eszközöm, a COM9-es portra. Ez annak tudható be, hogy a számítógép az Arduino programozását úgy kezeli, mintha soros portról történne a beállítás, viszont a soros csatlakozó háttérbeszorulása végett fizikailag USB-n történik a programozás. Ezután megnyitottam az Arduino IDE programozói felületet és elvégeztem az alapvető beállításokat. Ez azt takarja, hogy kiválasztottam a magyar nyelvű programban, hogy melyik porton tudja elérni az Arduinomat az applikáció, illetve, hogy pontosan melyik verziójú Arduino terméket is csatlakoztattam a gépemhez. A megfelelő beállítás után már csak a feltöltést s működést kellett letesztelnem. Ezt az alapvető, úgynevezett „Blink” program kipróbálásával végeztem. A programban vannak úgynevezett „Példák”, melyek már előre le vannak kódolva, nekünk csak fel kell töltenünk a lapunkra, s ez is egy ilyen sketch. A megnyitás után, gyorsan átolvastam, hogy mi is fog történni az Arduino lapomon. A program a Mega 13-as digitális lábának logikai szintjét változtatja 1 másodpercenként, ezáltal villogtatva egy LED-et a lapon, mely párhuzamosan van kötve az említett lábbal. A fizikai kapcsolás, illetve kód a következő képen néz ki:

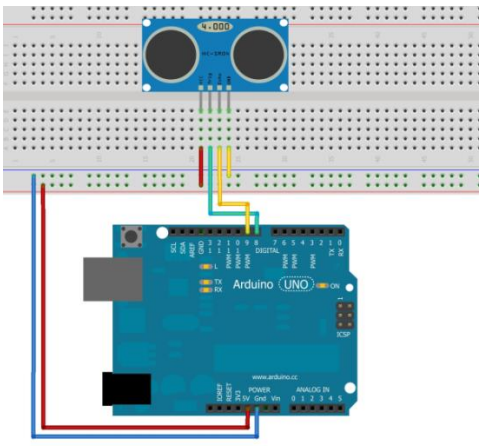


13. ábra: Az Arduino Mega 2560 - as összekötése a géppel

```
/*
  Blink
  Turns on an LED on for one second, then off for one second, repeatedly.

  Most Arduinos have an on-board LED you can control. On the Uno and
  Leonardo, it is attached to digital pin 13. If you're unsure what
  pin the on-board LED is connected to on your Arduino model check
  the documentation at http://www.arduino.cc

  This example code is in the public domain.


modified 8 May 2014
  by Scott Fitzgerald
 */


// the setup function runs once when you press reset or power the board
void setup() {
  // initialize digital pin 13 as an output.
  pinMode(13, OUTPUT);
}

// the loop function runs over and over again forever
void loop() {
  digitalWrite(13, HIGH);   // turn the LED on(HIGH is the voltage level)
  delay(1000);              // wait for a second
  digitalWrite(13, LOW);    // turn the LED off by making the voltage LOW
  delay(1000);              // wait for a second
}
```

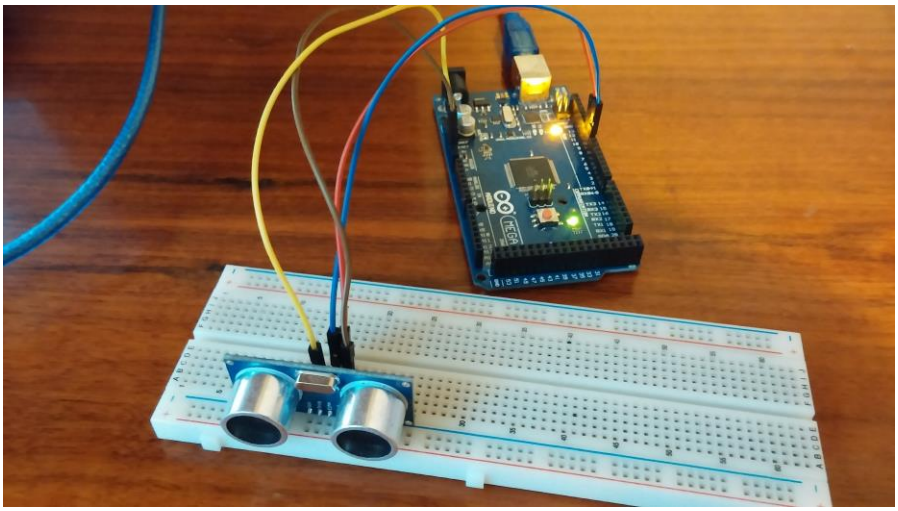
3.3.2. A HC-SR04 ultrahangos szenzor kipróbálása

Ez volt az első shield, amit kipróbáltam, miután elegendő mennyiségű alkatrészem érkezett meg, s már érdemi munkát tudtam végezni. A bekötése nem számított túl bonyolultnak, mindössze 4 dróttal kellett összekötnöm az Arduinot és a szenzort. Ehhez a művelethez egy próbapanelt hívtam segítségül, mivel így sokkal könnyebb volt megoldanom az egyes csatlakozásokat.

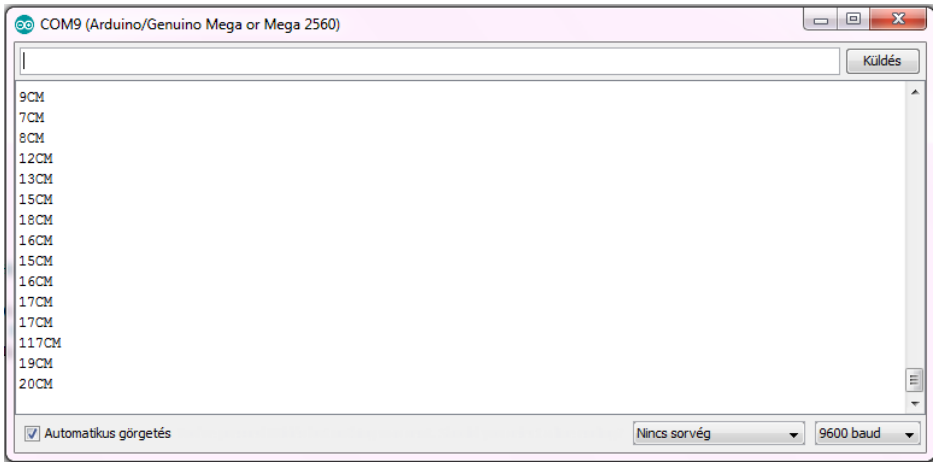


14. ábra: A HC-SR04 ultrahangos szenzor csatlakoztatása az Arduinohoz [14]

Miután az összekötést elvégeztem már csak a megfelelő kódot kellett begépelnem s feltöltenem. A teszteléshez a 9600-as soros portot vettem segítségül, mivel az Arduinom ekkor még nem rendelkezett kijelzővel, amire vissza tudtam volna adni a szenzor adatait. Persze tesztelhettem volna úgy is, hogy bekapcsolok egy LED-et, ha megfelelő közelségbe érzékel a szenzor, egy bizonyos tárgyat, de ezt az ötletemet hamar elvetettem, mivel kíváncsi voltam a szenzorérzékelési pontosságára, hatótávolságára is. Így a feladat, amit feltöltöttem a mikrovezérlőmre, azt csinálta, hogy visszaadta azt a távolságot, amennyit a szenzor érzékelt, tekintve ugye, hogy ez a bővítő lapka távolság mérésére készült. Mikor teszteltem, hamar kiderült, hogy a HC-SR04, amit az interneten rendeltem, mindössze 1-1.2 métert érzékel pontosan, utána már véletlenszerű adatokat ad vissza. Ezt annak tudtam be, hogy klón alkatrész révén, kisebb hatásfoka van, mint egy eredeti HC-SR04-nek, ami akár 4 méter távolságot is képes lefedni. Viszont mivel számomra annyira nem lényeges a lefedett távolság nagysága, mindössze az, hogy érzékelje a robotom az elébe került akadályokat, így ennél a probléma felvetülésekor nem estem pánikba. A fizikai kinézet, a visszaadott adatok, és a forráskód a lentebb láthatóak:



15. ábra: A szenzor tesztelésének fizikai kinézete

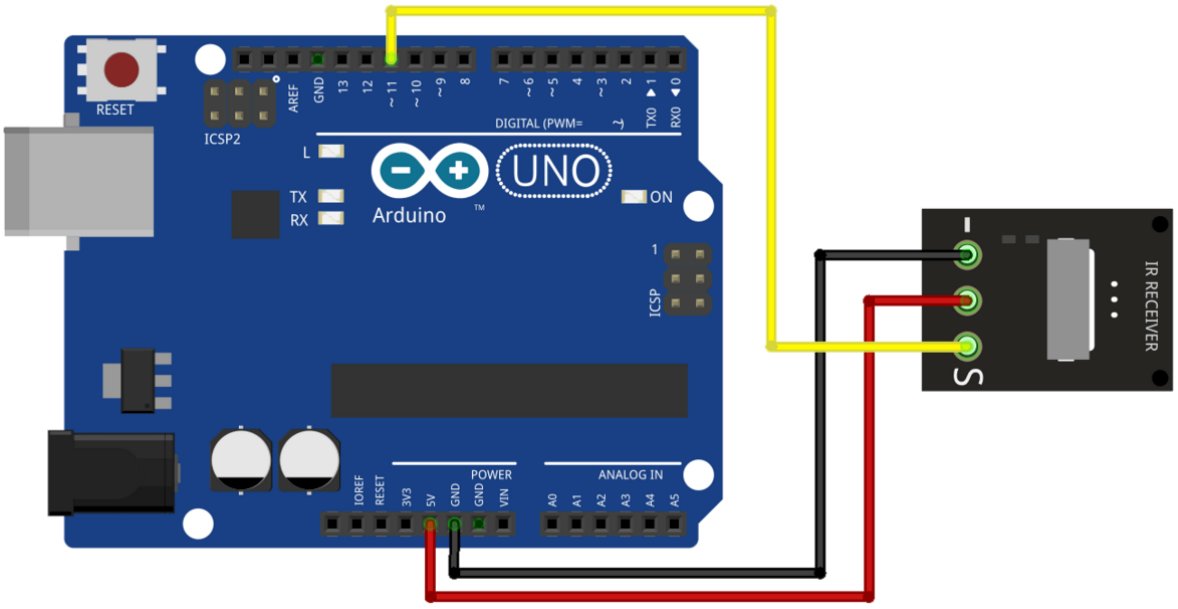


16. ábra: A szenzor által a soros portra küldött információk

```
#define echoPin 9 //definiálja az echoPin helyét a 9-es lábra
#define trigPin 8 //definiálja az trigerPin helyét a 8-as lábra


void setup() //alapvető beállítások
{
  Serial.begin(9600); //megnyissa a 9600-as serail portot, ahova kiírja majd az
                      //adatokat a program
  pinMode(trigPin, OUTPUT); //beállítsa a triger pint kimenetnek
  pinMode(echoPin, INPUT); //beállítsa az echo pint bemenetnek
}

void loop() //a végtelenségig futó főprogram ciklus
{
  int distance, duration; //definiáljuk a távolság és az időtartalom változókat
  digitalWrite(trigPin, HIGH);  //a trigerPin-t beállítsa magas logikai szintre
  delay (100); //0.1 s-et vár a program
  digitalWrite(trigPin, LOW);   //a trigerPin-t beállítsa alacsony logikai szintre
  duration = pulseIn(echoPin, HIGH); //időtartalom egyenlő az echoPin magas
                                     //logikai szint érzékelésével
  distance = (duration / 2) / 29.1; //távolságot kiszámító képlet
  Serial.print(distance); //kiírja a soros portra a distance változó értékét
  Serial.print("CM"); //kiírja a soros portra azt, hogy  "CM"
  Serial.println(""); //új sort kezd a soros port kiírásánál
  delay(100); //0.1 s-et vár a program
}
```

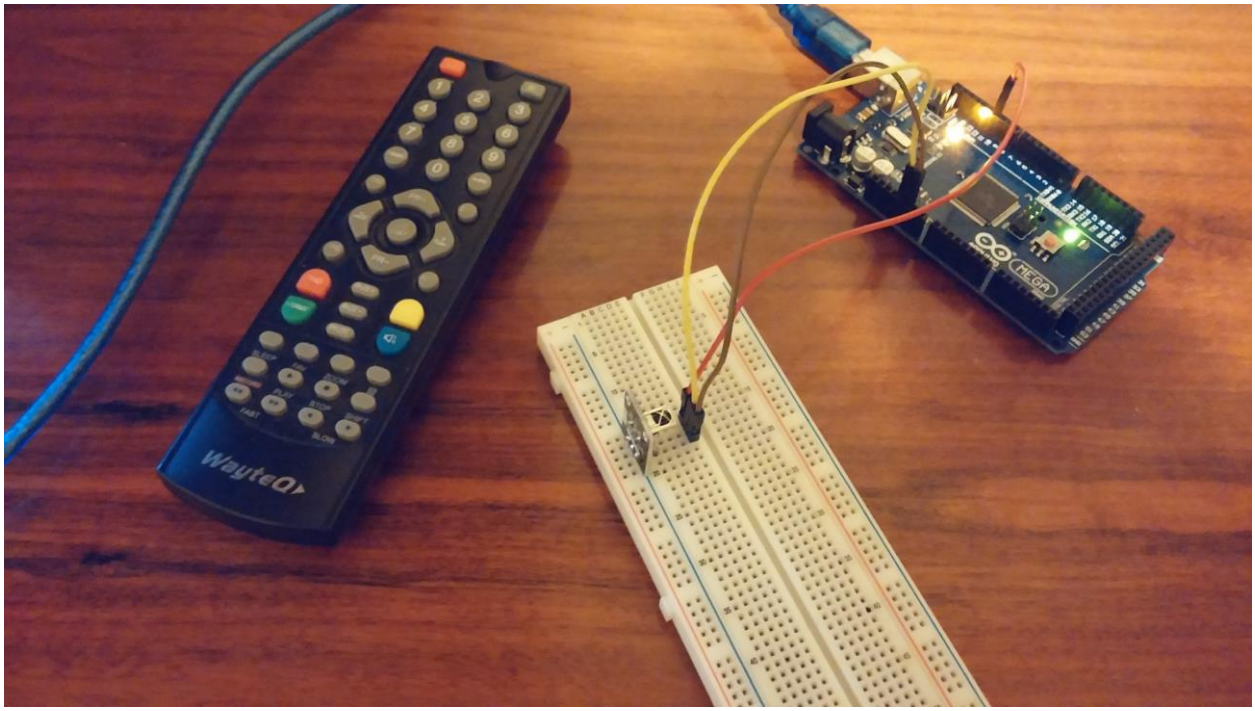
3.3.3. Az IR távirányító shield kipróbálása

A távirányító bővítő lapkájánál úgy gondoltam, hogy nem lesz különösebb gond a kipróbálásnál. Az érzékelő három lába közül mindössze egyetlen egy van, ami információt szolgáltat a mikrovezérlő számára, a másik kettő csak a működéshez szükséges 5 V - ot adja, illetve a földet.

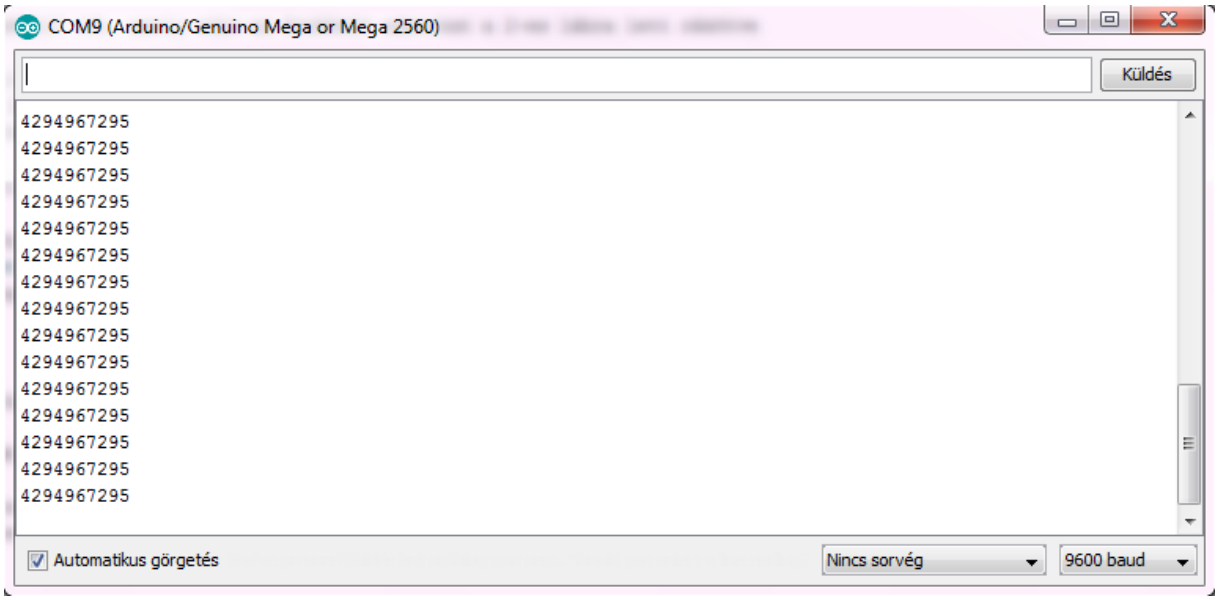


17. ábra: Az IR szenzor bekötése az Arduinoba [15]

Azonban rá kellett jönnöm, hogy a dolgok ritkán ilyen egyszerűek. Ahhoz, hogy működésre bírjam a szenzort, le kellett töltenem, egy úgynevezett „IRremote" library - t, amit be kellett másolnom az Arduinot programozó applikációm könyvtárai közé. Ennek a könyvtárnak több verzióját is megtaláltam, de én az összes közül csak az egyikkel tudtam érdemleges munkát végezni. Ez gondolom abból adódhatott, hogy más verziójú könyvtáraknál, más-más utasítások vannak, illetve másképpen kell rájuk hivatkozni. Miután bemásoltam a megfelelő fájlokat, egy újabb problémára bukkantam, mégpedig, hogy a programom két helyen is talált hasonló utasításokat tartalmazó könyvtárakat. Utánajárás követően kiderült, hogy van egy úgynevezett „RobotIRremote" library, ami hasonló adatokat tartalmaz, de azzal ez a verziójú távirányító lap nem kompatibilis, így azt a mappát ki kell venni a könyvtárak közül, hogy ne találja meg a program. Miután ezt mind elvégeztem, nekiálltam a tesztprogramomnak, mellyel ki tudtam próbálni, hogy nem kaptam-e hibás elemet a rendelésem során. A tesztelés megint soros portra küldött adatokkal történt, mint a szenzor esetében. Arra voltam kíváncsi, hogyha lenyomom a kapott távirányítón a gombot, akkor milyen adat érkezik decimálisan az Arduinora, hogy aztán erre tudjak majd „if" parancs segítségével különböző funkciókat írni a robotomnál. Ekkor felmerült egy elég komoly probléma, mégpedig az, hogy a távirányítóm különböző, véletlenszerű adatokat adott vissza a mikrovezérlőmnek. Több próbálgatás után rájöttem, hogy ez a távolságnak tudható be, ugyanis a távirányító hatótávolsága alig több 10-20 cm-nél, valamint a befogadási szöge a szenzornak se a legjobbnak mondható. Ekkor megfogtam egy egyszerű műholdvevő távirányítót, s elkezdtem ezzel tesztelni az IR szenzoromat. Ezzel az eszközzel már sokkal jobb adatokhoz jutottam, így úgy döntöttem, hogy a projektemhez nem a szenzorral kapott távirányítót fogom majd felhasználni, hanem keresek egy másikat, mellyel, sokkal jobb eredményeket tudok elérni. A fizikai kapcsolás, soros portra kapott adatok és a forráskód lentebb látható:



18. ábra: Az IR távirányítás tesztelésének fizikai látszata

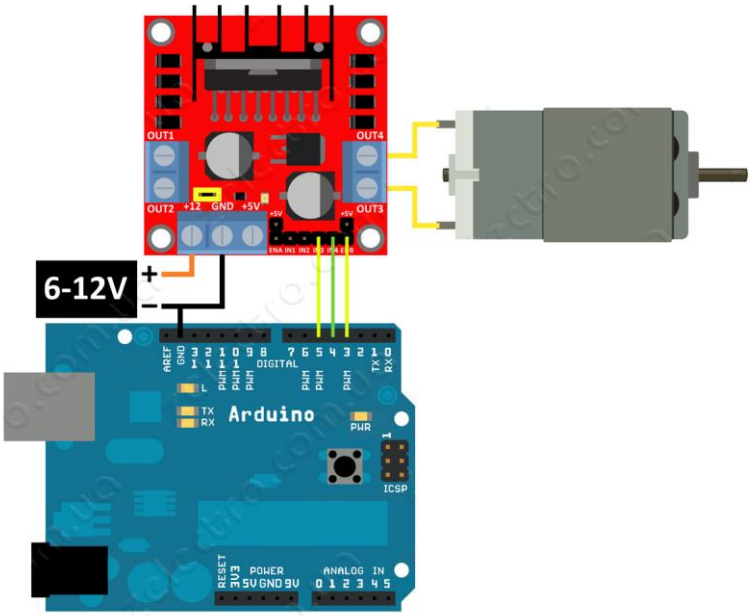


19. ábra: Az IR szenzor által érzékelt decimális kód az 5-s távirányítógomb lenyomásakor

```
#include "IRremote.h" //itt nyissuk meg a kapcsolatot az "IRremote" könytárral

int receiver = 11; // Az érzékelő láb, amely adja az érzékelt adatokat a 2-es lábra lett
                   // rákötve

/*-----( Objektumok deklarálása )-----*/
IRrecv irrecv(receiver); // létrehoz egy példányt az  'irrecv' - re
decode_results results; // létrehoz egy példányt a 'decode_results' - ra

void setup() //alapvető beállítások
{
  Serial.begin(9600); //megnyissa a 9600-as serail portot, ahova kiírja majd az adatokat
                      //a program
  Serial.println("IR Receiver Button Decode"); //kiírja a soros portra, hogy "IR Receiver
                                               //Button Decode"
  irrecv.enableIRIn(); // Engedélyezi az IR vételt
}

void loop() //a végtelenségig futó főprogram ciklus
{
  if (irrecv.decode(&results)) //kaptunk IR jelet?
  {
    translateIR(); //elindítsa a "translateIR()" nevű alprogramot
    irrecv.resume(); //a következő adat vétele, folytassa az adatvételt
  }
}

void translateIR() //itt történik az utasítás hozzárendelése a kapott bináris adathoz
{
  Serial.println(results.value); //kiírja a távirányító által kapott adatot a soros
                                 //portra
  delay(500); //ne folytatódjon rögtön az adatfelismerés, 0.5 s-et vár a program
}
```

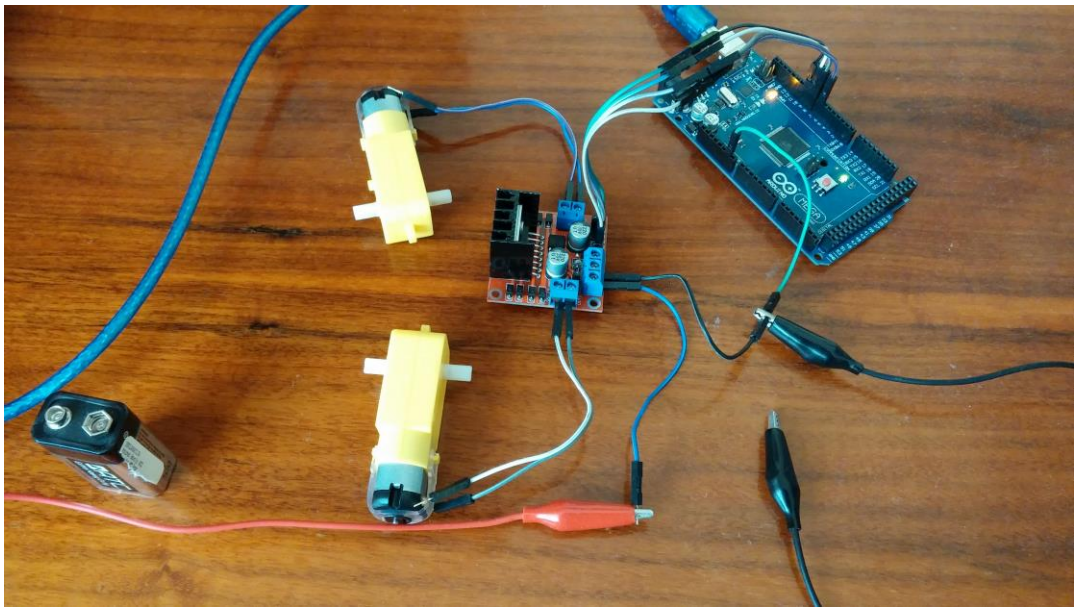
3.3.4. A motorok és a motorvezérlő (L298N) kipróbálása

A robotautómnak ez a legfontosabb része, hiszen, ha nem gurul a szerkezetem, akkor egyáltalán nem képes ellátni a többi funkcióját se. Ennek a kapcsolásnak a tesztelésénél nem alkalmaztam próbapanelt, mivel az L298N már alapból olyan tüskékkel, illetve foglalatokkal van ellátva, melyekre könnyű kapcsolódni a próbapanelokhoz járó áthidaló vezetékekkel. A csavaros foglalatokba való csatlakozás nem okozott gondot, viszont van 4 tüske, melyekkel a motorok forgási irányát lehet változtatni, s amelyekre csak női csatlakozókkal lehet kapcsolódni. Ezekre a kapcsolódást csak úgy tudtam megoldani, hogy párosítottam egy női-női és egy férfi-férfi csatlakozókkal ellátott vezetéket, ugyanis női-férfi vezetéket nem sikerült szereznem, mint ahogy szerettem volna.



20. ábra: Az L298N motorvezérlő csatlakoztatásának módja az Arduinohoz [16]

Energiai forrásnak egy 9V-os elemet használtam, amire csipeszcsatlakozókkal csatlakoztam. Ez két motor hajtásánál nem bizonyult elég erősnek, ha sokáig engedtem, hogy hajtsa a motorokat, akkor lecsökkent a sebesség, majd véglegesen leálltak a meghajtó elemeim. Viszont ha csak az egyiket szerettem volna meghajtani, akkor nem volt semmi gond a sebességgel, egységes, ingadozásmentes volt, kellő erővel rendelkező, mely szerintem elegendőnek kell, hogy legyen egy könnyebb szerkezet mozgatásához. Így úgy döntöttem, hogy a robotom végső alakjánál majd 12 V - os energiaforrást fogok használni, akár egy akkumulátort, akár egy 12 V - os elemet, attól függően még, hogy mit találok jobbnak, illetve melyiket tudom megoldani, hogy minél könnyebb szerkezetet tudjak majd megvalósítani. A PWM csatlakozókat nem használtam egyik motornál sem, így maximális sebességgel pörögtek a forgórészek. Ezzel egyelőre nem akartam foglalatoskodni, hiszen az alapvető terveim között nem szerepelt, hogy változtatni fogom az autóm haladási sebességét. A fizikai megvalósítás, illetve a teszteléshez használt forráskód lentebb látható:

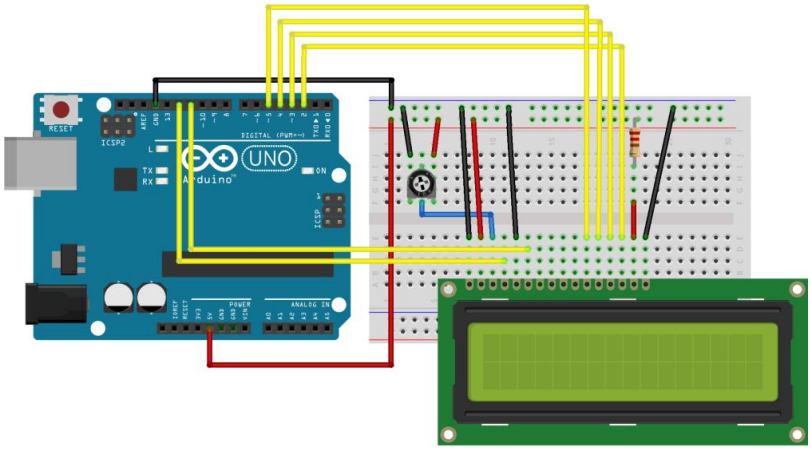


21. ábra: A motorjaim és az L298N vezérlő összekötése az Arduino Mega 2560 - al

```
const int in1 = 10; //definiáljuk, hogy az A motor forgását a 10-es lábbal vezéreljük
const int in2 = 9; //definiáljuk, hogy az A motor forgását a 9-es lábbal vezéreljük
const int in3 = 8; //definiáljuk, hogy a B motor forgását a 8-es lábbal vezéreljük
const int in4 = 7; //definiáljuk, hogy a B motor forgását a 7-es lábbal vezéreljük

void setup() //alapvető beállítások
{
  pinMode(in1, OUTPUT); //az "in1" vezérlőláb beállítása kimenetnek, innen kap
                        //információt az L298N
  pinMode(in2, OUTPUT); //az "in2" vezérlőláb beállítása kimenetnek, innen kap
                        //információt az L298N
  pinMode(in3, OUTPUT); //az "in3" vezérlőláb beállítása kimenetnek, innen kap
                        //információt az L298N
  pinMode(in4, OUTPUT); //az "in4" vezérlőláb beállítása kimenetnek, innen kap
                        //információt az L298N
}

void loop() //a végtelenségig futó főprogram ciklus
{
  digitalWrite(in1, LOW); //az "in1"-es lábat beállítjuk alacsony logikai szintre a
                          //vezérléshez
  digitalWrite(in2, HIGH); //az "in2"-es lábat beállítjuk magas logikai szintre a
                           //vezérléshez
  digitalWrite(in3, HIGH); //az "in3"-es lábat beállítjuk magas logikai szintre a
                           //vezérléshez
  digitalWrite(in4, LOW); //az "in4"-es lábat beállítjuk alacsony logikai szintre a
                          //vezérléshez
}
```

3.3.5. A 16x2-es LCD kijelző tesztelése

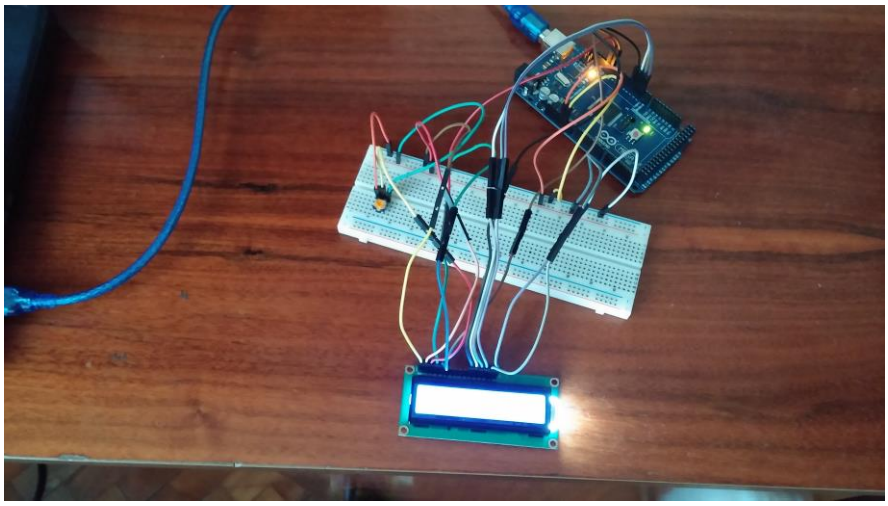
Ennek a modulnak a tesztelése, kipróbálása emésztette fel a legtöbb időt a felkészülési időszakból. A bekötése se volt egyszerűnek mondható, de ezen kívül egyéb probléma is fellépett vele. Mikor kézhez kaptam a kijelzőt, akkor ez az elem két külön részként került a kezeim közé. Az egyik egy lapka volt, amin a kijelző kapott helyett és rajta voltak a csatlakozási furatok, a másik része pedig egy tüskesor foglalt magába, melyet a lapka furataiba kellett beleforrasztani. Én a forrasztási részt hirtelenjében ki akartam hagyni, mivel gondoltam e nélkül is le tudom tesztelni a bővítő lapkát, azonban tévedtem.



22. ábra: Az LCD kijelző bekötési ábrája. Az ellenállás én a kapcsolásomban nem szerepelt [17]

Mikor összekötöttem a kapcsolásomat, s feltöltöttem a programomat, nem reagált semmire sem az LCD. Ekkor elkezdtem mozgatni a lapot, amibe a tüskesort mindössze beleillesztettem forrasztás nélkül, s elkezdtek megjelentek rajta különféle értelmetlen jelek. Itt tudatosodott bennem, hogy a rögzítés nélkül nem tudom letesztelni a kijelzőmet, mivel nem érintkeznek megfelelően a csatlakozók. A tüskesor forrasztását egy hőszabályzós forrasztópákával végeztem, nagyon óvatosan, mivel a furatok között minimális távolság volt. Miután ezt a javítást elvégeztem, már tökéletesen működött a kijelzőm, és elkezdhettem a tesztelést. A programírással nem volt problémám, a megfelelő könyvtárra való hivatkozás után, valamint az LCD kijelző definiálását követően könnyűszerrel tudtam használni az elemet. A kapcsolás fizikai megvalósítása, illetve a teszteléshez használt programom lentebb látható:

23. ábra: Az LCD kijelzőn megjelenő szöveg látszata



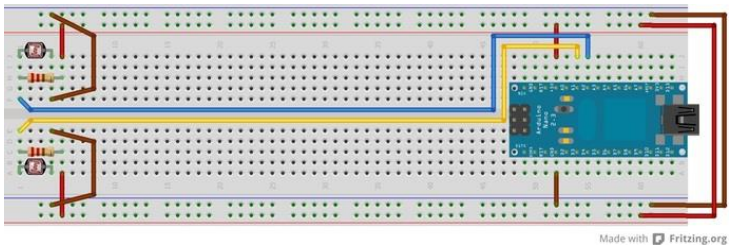
24. ábra: Az Arduino és az LCD kijelző összekapcsolása próbapanel segítségével

```
#include <LiquidCrystal.h> //itt nyissuk meg a kapcsolatot az "LiquidCrystal" könytárral

LiquidCrystal lcd(12, 11, 5, 4, 3, 2); //definiáljuk az lcd kijelzőt

void setup() //alapvető beállítások
{
  lcd.begin(16, 2); //az lcd kijelző hosszának definiálása
  lcd.clear(); //az lcd kijelző letörlése, ha lenne esetleg rajta valami
}

void loop() //a végtelenségig futó főprogram ciklus
{
  lcd.print("Printing text"); //az lcd kijelzőre kiírja a "Printing text" szöveget
  delay(3000); //3 s-et vár a program mielőtt továbblépne
  lcd.clear(); //letörli a kijelző tartalmát

  lcd.setCursor(0,1); //beállítsa a cursort az első oszlop 2. sorába
  lcd.print("Setting cursor"); //az lcd kijelzőre kiírja a "Setting cursor" szöveget
  delay(3000); //3 s-et vár a program mielőtt továbblépne
  lcd.clear(); //letörli a kijelző tartalmát

  lcd.print("Blink cursor"); //az lcd kijelzőre kiírja a "Blink cursor" szöveget
  lcd.blink(); //berak a kijelző adott helyére egy pislogó cursort
  delay(3000); //3 s-et vár a program mielőtt továbblépne
  lcd.clear(); //letörli a kijelző tartalmát
  lcd.noBlink(); //beállítja, hogy ne pislogjon tovább a cursor

  lcd.print("Unline cursor"); //az lcd kijelzőre kiírja az "Unline cursor" szöveget
  lcd.cursor(); //berak a kijelző adott helyére egy aláhuzó pislogásmentes cursort
  delay(3000); //3 s-et vár a program mielőtt továbblépne
  lcd.clear(); //letörli a kijelző tartalmát

  lcd.print("No cursor"); //az lcd kijelzőre kiírja a "No cursor" szöveget
  lcd.noCursor(); //eltünteti a cursort a kijelzőről
  delay(3000); //3 s-et vár a program mielőtt továbblépne
  lcd. clear(); //letörli a kijelző tartalmát
```

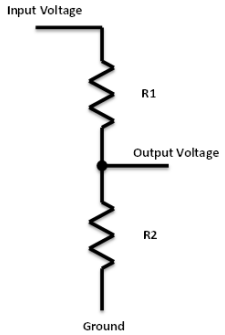
```
  lcd.print("No display"); //az lcd kijelzőre kiírja a "No display" szöveget
  delay(1000);//1 s-et vár a program mielőtt továbblépne
  lcd.noDisplay(); //kikapcsolja a kijelzőre kiírt karakterek láthatóságát
  lcd.clear(); //letörli a kijelző tartalmát

  lcd.print("Display on"); //az lcd kijelzőre kiírja a "display on" szöveget
  delay(3000); //3 s-et vár a program mielőtt továbblépne
  lcd.display(); //bekapcsolja a kijelzőre kiírt karakterek láthatóságát
  delay(3000); //3 s-et vár a program mielőtt továbblépne
  lcd.clear(); //letörli a kijelző tartalmát
}
```

3.3.6. Két foto ellenállással megvalósított fénykövetés letesztelése

Két fényérzékelő ellenállással a fényforrás helyének érzékelése, s annak alapján különböző LED-ek felkapcsolása. Ha a fényforrásra mind a két érzékelő rálát, akkor kigyullad egy LED, ha csak az egyik, illetve ha csak a másik, akkor 2 másik LED világit a helyzettől függően.
Erre a tesztelésre azért volt szükség, mert ez képezi a robotautóm fénykövetési funkciójának az alapját. A kapcsolás ugyan egyszerű, két feszültségosztót használ fel, viszont nem voltam biztos abban, hogy majd a forráskódommal feltöltve megfelelően fog működni a mikrovezérlőm.

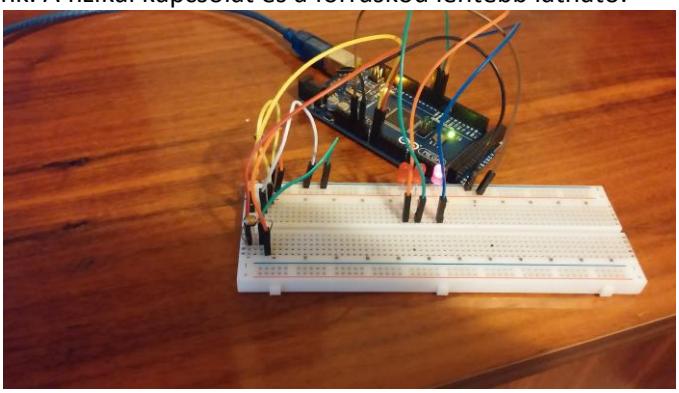


25. ábra: A fényérzékelési kapcsolás bekötése Arduino Nano esetén [18]



26. ábra: A fényérzékeléshez használt feszültségosztó kapcsolás [18]

A program lényege, melyet feltöltöttem az Arduinomra, hogy definiáltam a két foto ellenállás által érkezett analóg jel helyzetét, hogy melyik analóg portra is érkezik a jel, majd összehasonlítottam a két beérkező jelet és az eredménytől függően felkapcsoltam valamelyik LED – et a 3 közül. Ha fénykövető robotot szeretnénk, akkor ezeket az alprogramokat, melyek fényt kapcsolnak, át kell csak írni motorhajtásra, és már működik is a fényforrást követő autónk. A fizikai kapcsolat és a forráskód lentebb látható:

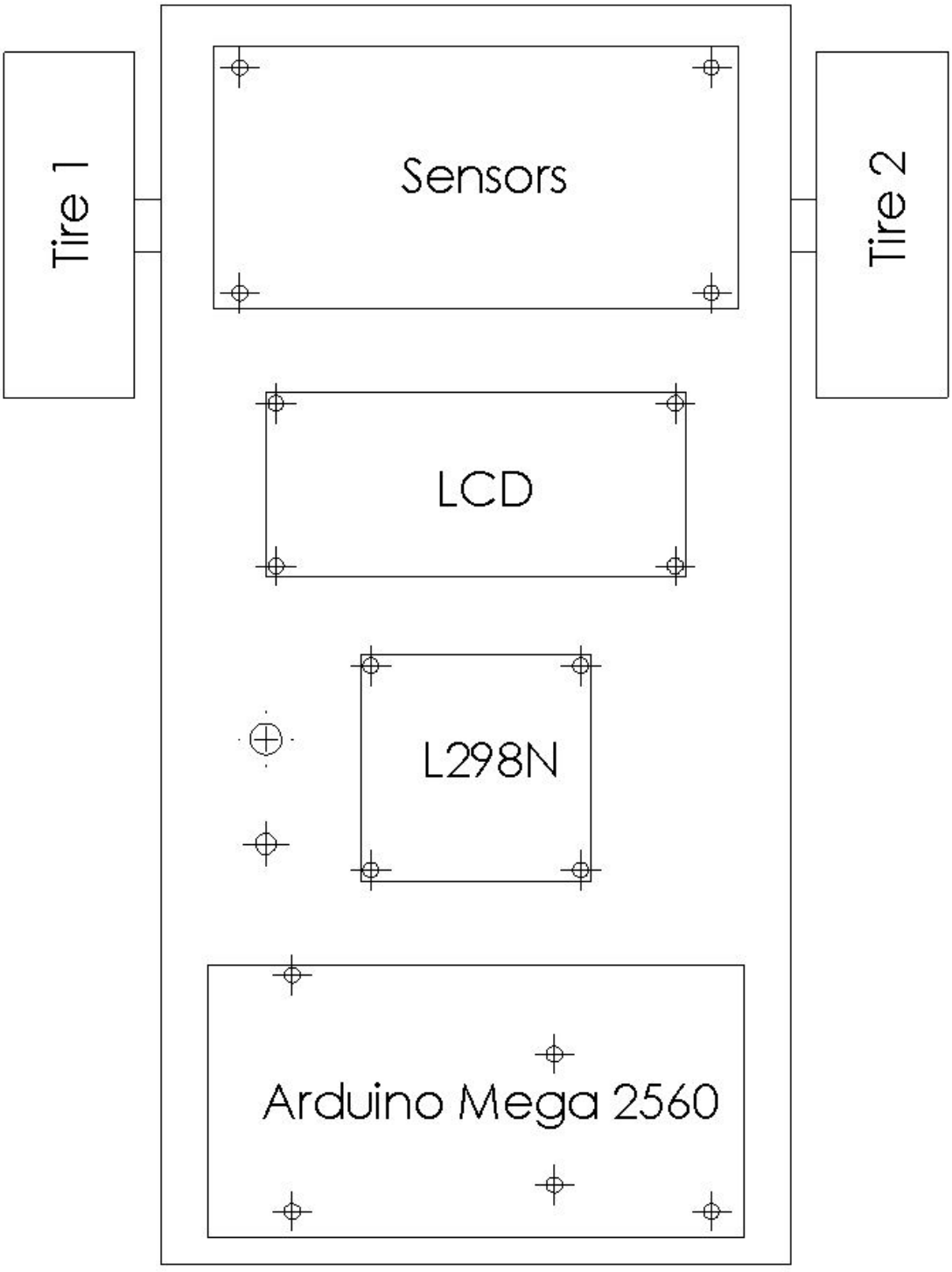


27. ábra: A fényérzékelő próbakapcsolás megvalósítása

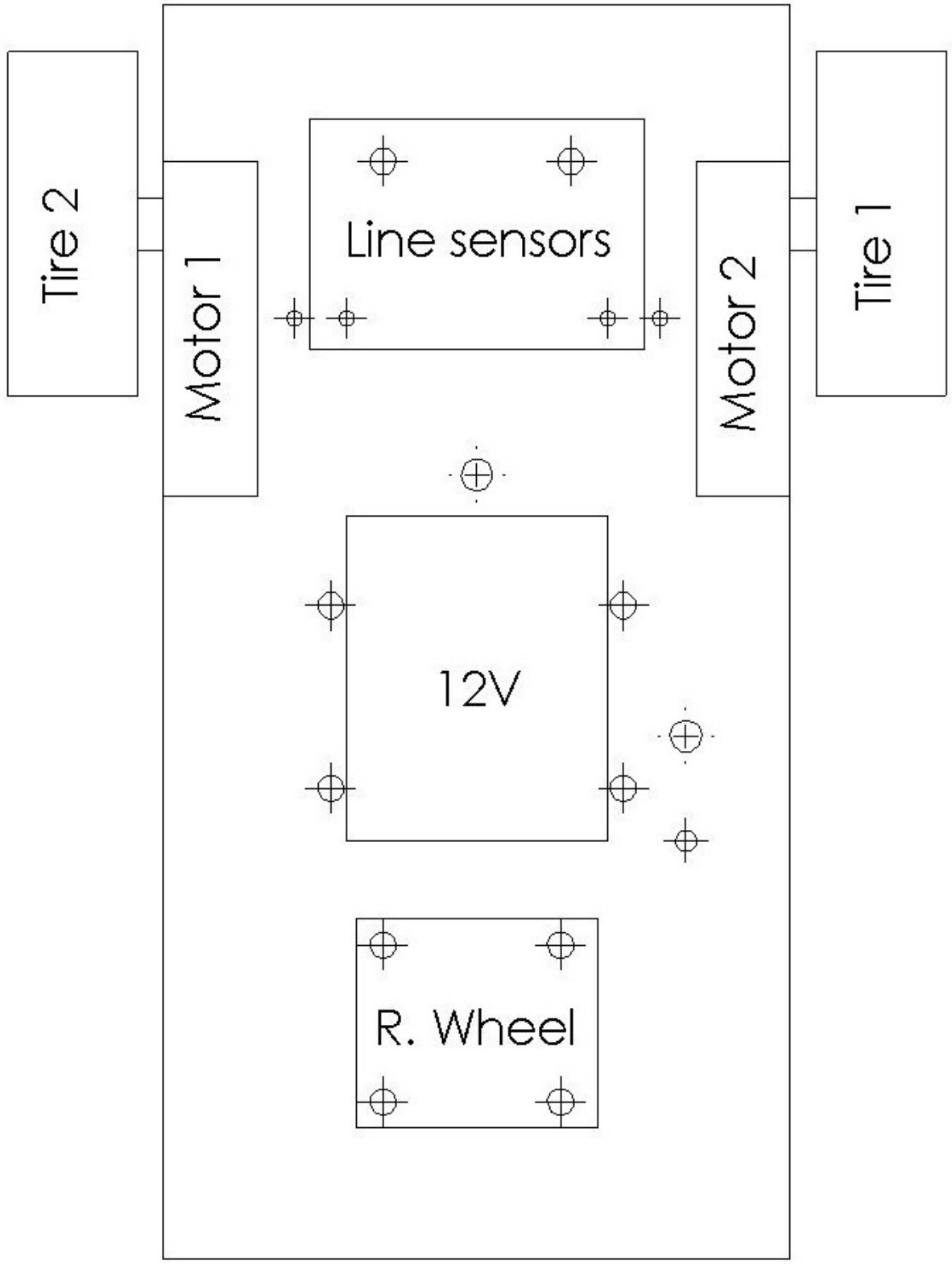
```
const int leftLDR = A0; //a bal fényérzékelő csatlakoztatva van az analog láb 0-ra
const int rightLDR = A1; //a jobb fényérzékelő csatlakoztatva van az analog láb 1-re
int leftValue;  //ez a változó tárolja azt az adatot, amit a bal fényérzékelő szolgáltat
int rightValue; //ez a változó tárolja azt az adatot, amit a jobb fényérzékelő szolgáltat
int difference; //ezt a változót használjuk, hogy összehasonlítjuk a két érzékelő álltal
                //szolgáltatott adatot

void setup() //alapvető beállítások
{
  pinMode(2, OUTPUT); //a 2-es digitális lábat kimenetre állítottuk
  pinMode(3, OUTPUT); //a 3-as digitális lábat kimenetre állítottuk
  pinMode(4, OUTPUT); //a 4-es digitális lábat kimenetre állítottuk
}

void loop() //a végtelenségig futó főprogram ciklus
{
  digitalWrite(2, LOW); //a 2-es digitális láb értékét beállítsuk alacsony szintre
  digitalWrite(3, LOW); //a 3-as digitális láb értékét beállítsuk alacsony szintre
  digitalWrite(4, LOW); //a 4-es digitális láb értékét beállítsuk alacsony szintre
  rightValue = analogRead(rightLDR); //tárolja a jobb fényérzékelő álltal szolgáltatott
                                     //analog információt
  leftValue = analogRead(leftLDR); //tárolja a bal fényérzékelő álltal szolgáltatott
                                   //analog információt
  difference = abs(rightValue-leftValue); //kiszámítsa a különbséget a két érték között
  if(difference < 50) //ha a különbség kissebb mint 50...
  {
    led1(); //bekapcsolja az 1-es LED-et
  }
  else //ha nem kissebb mint 50, akkor
  {
   if(rightValue > leftValue) //ha jobb oldalon sotétebb van
    {
     led2(); //bekapcsolja a 2-es LED-et
    }
    else //ha a bal oldalon van sötétebb
    {
     led3(); //bekapcsolja a 3-as LED-et
    }
  }
}

void led1() //1-es LED-et bekapcsoló alprogram
{
  digitalWrite(4, HIGH); //a 4-es digitális láb értékét beállítsuk magas szintre
}

void led2() //2-es LED-et bekapcsoló alprogram
{
  digitalWrite(2, HIGH); //a 2-es digitális láb értékét beállítsuk magas szintre
}

void led3() //3-as LED-et bekapcsoló alprogram
{
  digitalWrite(3, HIGH); //a 3-as digitális láb értékét beállítsuk magas szintre
}
```

3.4. A robotautóm összeállítási tervei és az elektronikai kapcsolása

A fizikai elrendezést a Solid Works 2013 nevezetű programban terveztem meg, testvérem segítségével, aki jelenleg gépészmérnöki tanulmányokat végez. Több terv is készült, én a végső változatokat mellékelem.

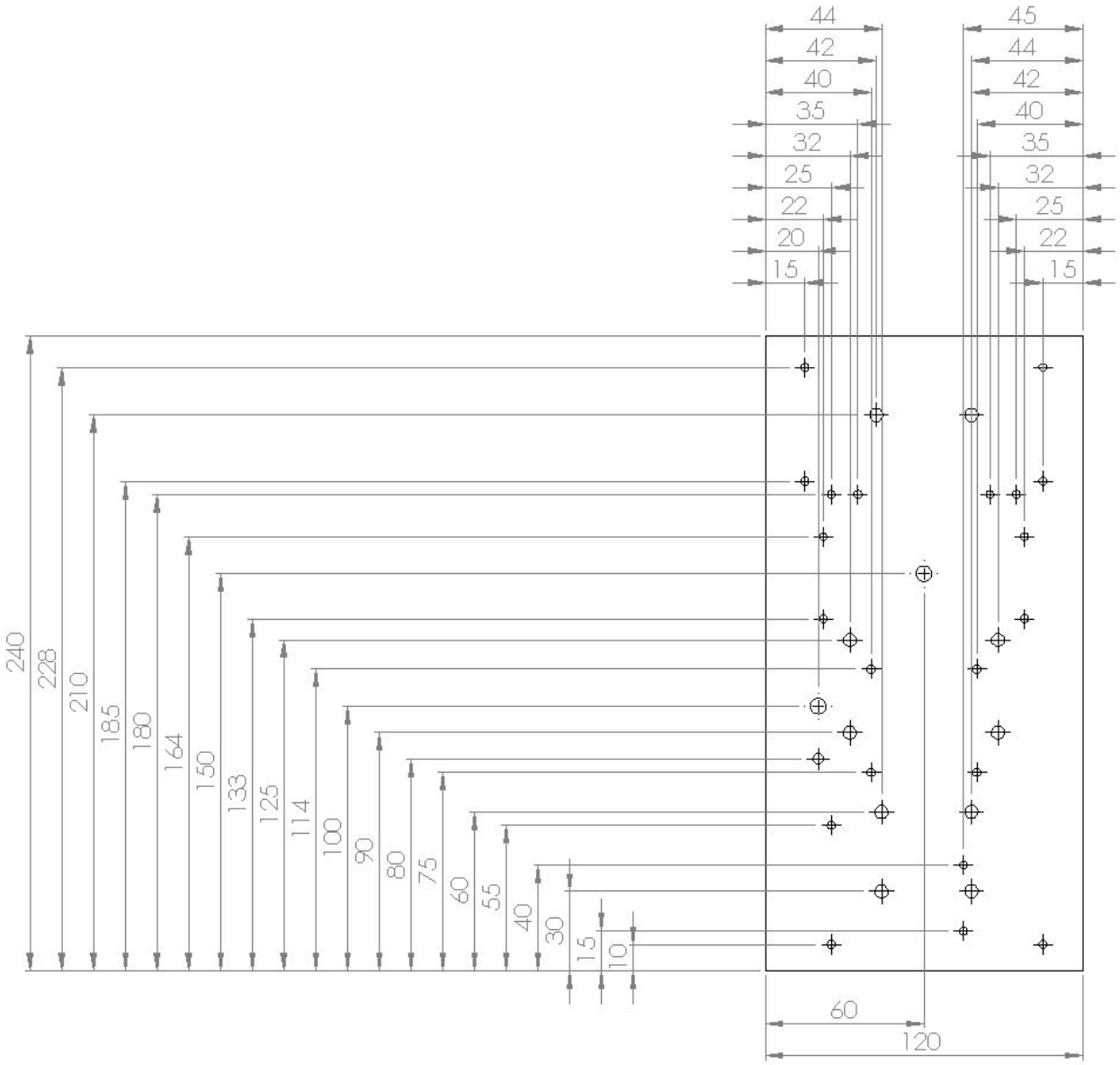


28. ábra: A robotautómon elhelyezkedő elemek felülnézetből

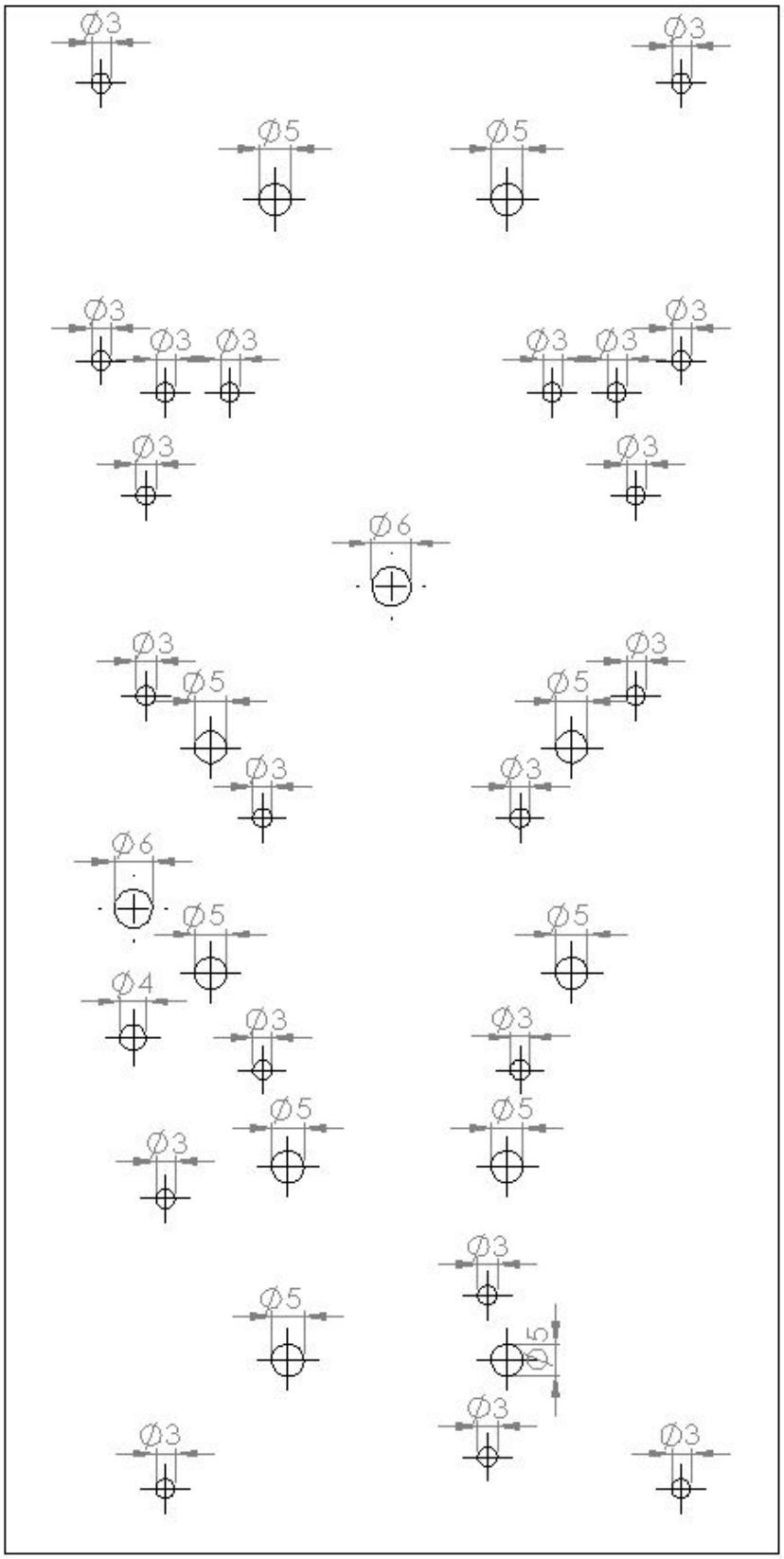


29. ábra: A robotautómon elhelyezkedő elemek alulnézetből

A furatok pontos méreteinek az ábrázolása, a műszaki rajzon, igen nehézkes feladatnak bizonyult, ugyanis rengeteg különböző elemmel dolgoztam, melyek különböző furattávolságokkal operálnak. A legtöbb alkatrész viszont legalább abban hasonlított egymásra, hogy mindegyik 3-as csavarral történő rögzítésre volt előlátva, így a legtöbb helyen ezt használtam. Kivételt képezve pár elemet, melyek 5-s csavarral vannak rögzítve a nagyobb stabilitás érdekében. Hogy ezeket a csavarkülönbségeket be tudjam mutatni, két külön ábrán szemléltetem a furatok térbeli elhelyezkedését, illetve azoknak a megfelelő átmérőjét.

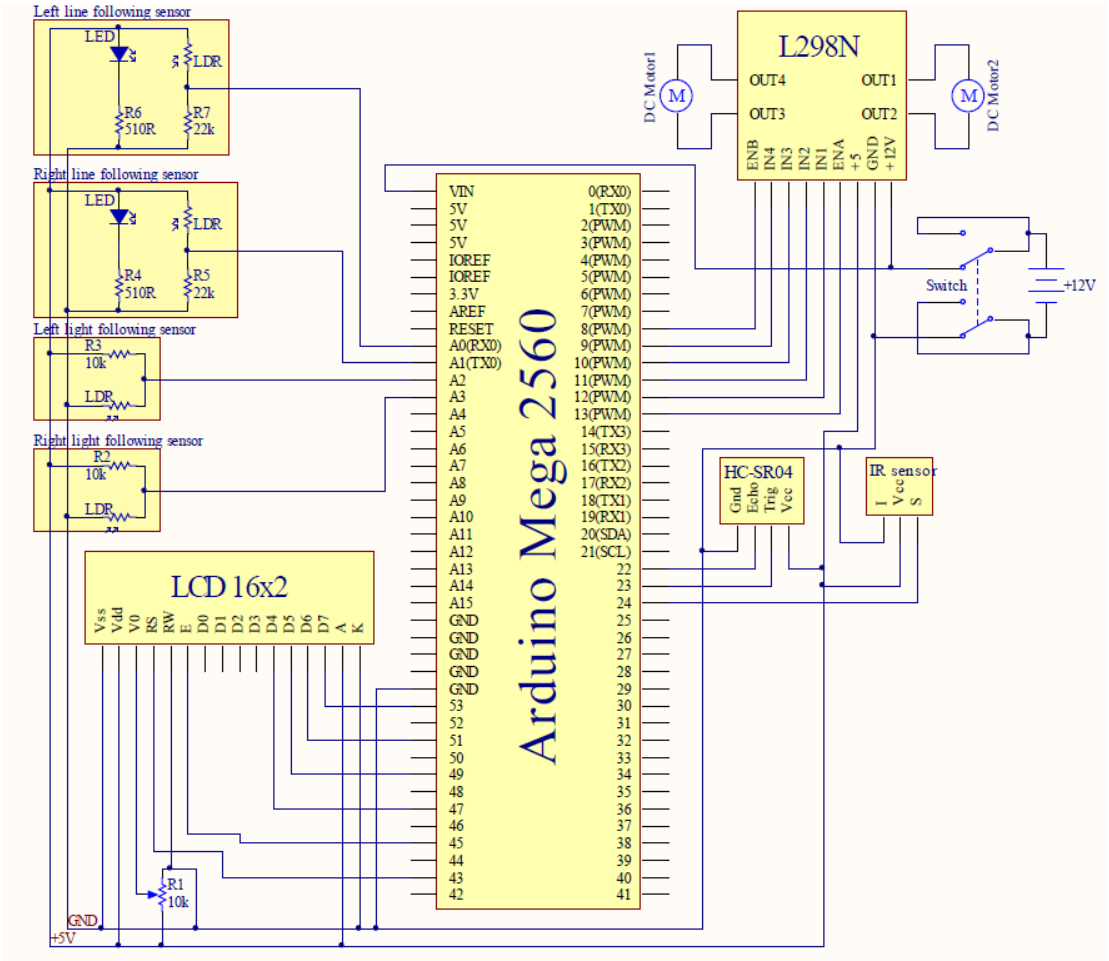


30. ábra: A alaptest furatainak pontos helyzete



31. ábra: A alaptest furatainak átmérője

A kapcsolási sémát az Altium Designer Winter 09 nevezetű programban készítettem el, melynek használatát a főiskolán történő tanulás közepette sajátítottuk el. Sajnos az elektronikai kapcsolatok bonyolultságára való tekintettel, nem tudtam teljes lapterjedelemben megjeleníteni a szakdolgozatomban a képet, úgy, hogy az esztétikus, átlátható elrendezést is biztosítsam. Ennek ellenére hiszem, hogy megfelelően látszódnak a kapcsolatok és egyértelműen tudtam ábrázolni a kapcsolási sémát.



32. ábra: Az autóm kapcsolási rajza

6. táblázat: Arduino Mega 2560 lábkiosztása

Láb	Csatlakoztatott elem	Láb	Csatlakoztatott elem	Láb	Csatlakoztatott elem
A0	bal vonalérzékelő szenzor	11	Motor 1 láb	45	LCD E
A1	jobb vonalérzékelő szenzor	12	Motor 1 láb	47	D4
A2	bal fénykövető szenzor	13	Motor 1 PEM	49	D5
A3	jobb fénykövető szenzor	22	HC-SR04 Echo	51	D6
8	Motor 2 PWM	23	HC-SR04 Trigger	53	D7
9	Motor 2 láb	24	IR szenzor	GND	GND vezeték
10	Motor 2 láb	43	LCD RS	VIN	12 V tápfeszültség

3.5. Az test elkészítése és összeállítása

Miután végeztem az alkatrészek elhelyezésének tervével, illetve nagyjából kigondoltam, hogy mely vezetékeket, merre s hogyan fogom elrendezni, következő dolgom volt az alaplap kivágása, mely a különböző elemek összetartását szolgálja. Alapanyagul egy 314 mm hosszúságú, 241 mm szélességű és 3 mm vastagságú lezonit lapot használtam, melyre ideiglenes munkahelyemen bukkantam rá, miközben pakolást végeztem. Megkérdezés után főnökeim készségesen rendelkezésemre bocsájtották az anyagot, ezáltal is csökkentve az előállítási költségeket.

33. ábra: Az alap lezonit, filctollal megjelölve a vágási vonalak mentén

Mivel az alaplap, mely rendelkezésemre állt, sokkal nagyobbnak bizonyult, mint az én elképzelésem, mely egy 240 mm x 120 mm nagyságú autótestet takart, így a felesleget kénytelen voltam eltávolítani. Ezt egy kis házi készítésű cirkulával végeztem, melyet eredetileg nyomtatott lapok (pertinax, vitroplaszt, stb.) vágására szereltek össze. Tekintve, hogy a vágólapot hajtó motor nem ilyen vastag faanyag vágására lett kitalálva, így csak egy kisebb trükk alkalmazásával voltam képes a művelet elvégzésére. A vágást, a felrajzolt vonalnál kezdtem, viszont elkezdtem egy párhuzamos vágást is a felrajzolt vonalam azon oldalán, melyre nem volt szükségem, s mely hulladékként végezte. Erre azért volt szükség, mivel a vágólap beszorult a vastag lezonit lapba, amikor már jobban belevágott az anyagba, tehát csökkentenem kellett a súrlódást, melyet úgy értem el, hogy a két vágott vonalam közötti részt kitörtem, ahogy haladtam egyre beljebb és beljebb az anyagba.

34. ábra: A vágásoknál felhasznált házi készítésű kis cirkula

Mikor megkaptam az alaptestemet, nekikezdhettem a megfelelő furatok kifúrásához, melyekkel rögzítettem az alkatrészeimet.

Ez előtt azonban még elkészítettem a motorjaim tartószerkezetét. Tekintettel, hogy a motorjaimhoz nem kaptam semmi rögzítési alkatrészt mikor megrendeltem, így nekem kellett kitalálnom, hogyan fogom rögzíteni az alaptesthez őket, hogy később, mozgás hatására se mozduljanak majd el. Ezt egy apai tanács hatására, kétoldalasan rézfóliázott üvegszálas nyomtatott lapok (vitroplaszt) segítségével oldottam meg. A megfelelő kimérést követően filctollal megrajzoltam a vágási vonalakat, illetve a furatok helyét, melyek szükségesek voltak, hogy egy motortartót elő tudjak állítani. Először a fúrást végeztem el, mivel kicsi elemekről beszélünk, amelyeket vágás után már körülményesebb lett volna kezelni. A pontozást követően a lyukakat 3-as fúróheggyel készítettem egy elektromos csavarozó segítségével, mivel nem volt szükségem akkora erőre, hogy rendes fúrógépet kelljen használnom. A vágást a már említett kis vágógép (30. ábra) segítségével oldottam meg, s lévén, hogy ilyen lapokra lett kitalálva a szerkezet, már nem kellett a párhuzamos vágási trükkhöz folyamodnom.

35. ábra: A készülő motortartók

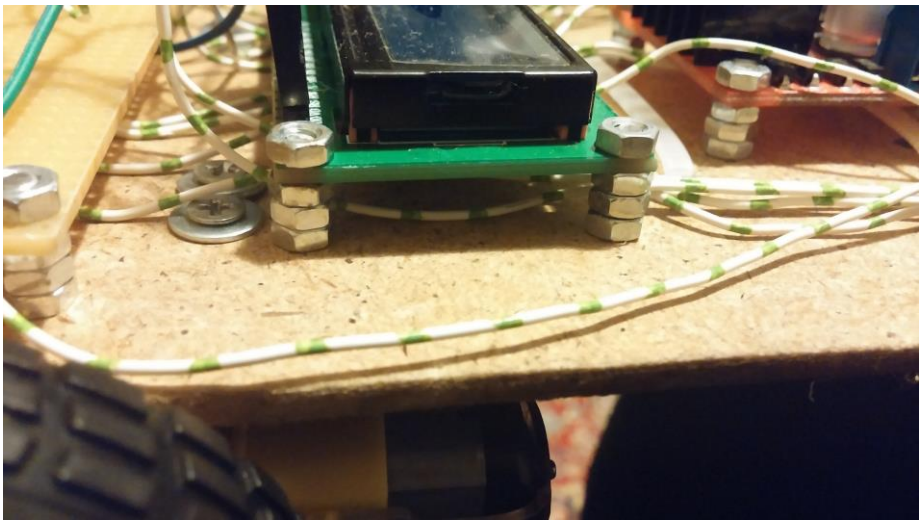
Ez után jött a kivágott s furatokkal ellátott elemek összeillesztése és itt mutatkozik meg, hogy miért is vitroplasztot használtam a tartók előállításához. Ugyanis az összeillesztésnél éleket kellett rögzíteni egymáshoz, s lévén, hogy ezeknek a lapoknak mind a két fele rézréteggel ellátott, így egyszerűen, forrasztópáka és forrasztóón segítségével össze tudtam őket forrasztani. A jobb forrasztás érdekében azonban forrasztózsírt is használtam, hogy biztosan megfelelően erős forrasztást kapjak eredményül.

36. ábra: Az elkészült motortartók

37. ábra: A forrasztási műveleteimnél használt forrasztópáka

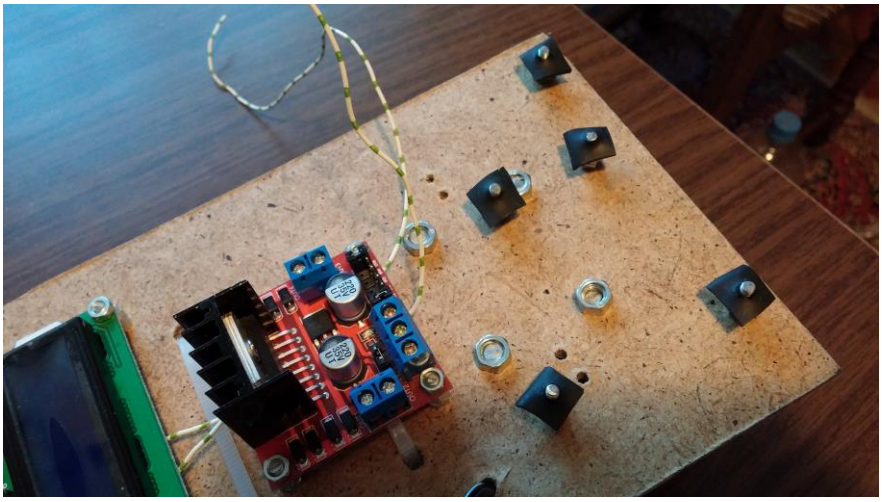
A motortartó elemek elkészültét követően, véglegesen rátérhettem a furatok kialakítására és az elemek felszerelésére. Ezt a megtervezett elhelyezési terveim alapján igyekeztem megvalósítani, de mivel nem bizonyultam túl jó fúrónak, így némely helyeken kompenzálnom kellett az alkatrészek elhelyezését, ami meglátszódik az LCD kijelző végleges helyén, illetve a motorjaim helyzetén is. Noha a motorjaimat azért voltam kénytelen kicsikét másképpen felerősíteni, mivel a kialakított tartóim kicsit deformáltak lettek, így a jobb és a bal motortartó elhelyezésén igyekeztem kompenzálni ezt a hibát, hogy a mozgásán ezt ne lehessen észrevenni. A legtöbb lyuknál 3-as süllyesztett csavarokat használtam, alátéttel, kivételt képezve ez alól pár elemet. A hátsó forgókeréknél és a tápforrás felszerelésénél, valamint a vonalérzékelő szenzornál, 5-s lyukakat használtam, míg a kapcsoló, illetve a középső, motor vezetékek csatlakozását segítő keresztülvezetést 6-s furatokkal oldottam meg. Ezen kívül sajnos került a testre még pár használaton kívüli lyuk is, mivel későig elnyúlt a test kialakítása, így fáradtan véletlenül felcseréltem az autóm elejét és hátulját, aminek az lett a következménye, hogy az első szenzorok alá akartam felerősíteni a hátsó kereket. A hibát már fúrás közben vettem észre, ezért helyeztem el úgy az alsó szenzorokat, ahogy állnak, hogy korrigáljam a vétett tévedésem egy részét, azonban 4 lyuk még így is használaton kívül maradt. Valamint a hátsó kereket tesztelés során le kellett végezetül cserélni, de erről bővebben majd a felmerült hibák fejezetben számolok be.

Miután elvégeztem a fúrómunkálatokat, következett az elemek felszerelése. A csavarokat, melyeket használtam, méretre kellett vágni, hogy ne legyenek túl hosszúak, mert azok rontották volna az esztétikai megjelenést, illetve ha hozzáérnek valamihez különböző problémákat okozhattak volna. Ezt egy csavarrövidítési lyukakkal ellátott fogó segítségével végeztem, noha, az 5-s csavarok rövidítésénél úgy tűnt, hogy nem fog megbirkózni a szerszám a feladattal, de nagyobb erőkifejtés hatására végül sikerrel jártam. Először a motorokat és a hátsó kereket szereltem fel, majd a tápforrást s ezután következetek a felső, kész elemek, melyeket nem kellett próbalapra forrasztani. Ugyanis voltak olyan alkatrészek, melyeket felszerelésük előtt kapcsolási lapra készítettem el, de ennek csak az alapelemek szerelése után álltam neki, viszont előre kifúrtam hozzájuk a megfelelő lyukakat. Az egyes elektronikai elemeket, melyek az autótestem felső felületére lettek elhelyezve, 3 anyacsavar segítségével megemeltem a felülettől, hogy alattuk még el tudjanak férni a csatlakozáshoz szükséges vezetékek.



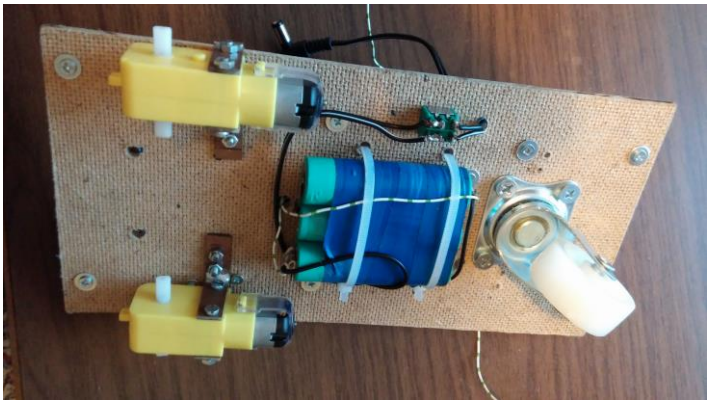
38. ábra: Az anyacsavarokkal megemelt elektronikai elemek

Azonban amikor a felszereléseket végeztem és az Arduino Megához értem, találkoztam egy problémával, mellyel eredetileg nem számoltam. Kiderült, hogy az Arduino nem úgy van kialakítva, hogy 3 - s csavarokkal lehessen rögzíteni, noha a kialakított lyukai első látásra ezt sugallták. Azonban mikor jobban megfigyeltem a mikrovezérlőt, kiderült, hogy némely helyeknél az anyacsavar olyan pozícióba kerülne, mely vezetést okozna egy-két láb között, amik valószínűleg különböző hibákat eredményeznének, vagy akár teljesen tönkre is tehetnék a vezérlőrendszeremet. Példának okáért a tápforrás két lába között is van rögzítő furat. Ebből arra engedtem következtetni, hogy plasztika távtartókkal való rögzítésre van kialakítva a Mega lap, viszont mivel én ezt nem tudtam beszerezni hirtelenjében, így alternatív megoldás után kellett néznem. Azt találtam ki, hogy az anyacsavarokra bicikligumiból kivágott kis négyzeteket helyezek el, melyek megakadályozzák a vezetést abban az esetben, ha valahol, valami átütne. Ez jó megoldásnak bizonyult, mivel mikor rögzítettem az Arduinomat és bekapcsoltam a programozása után, problémamentesen elvégezte a reá szabott feladatot.



39. ábra: Az Arduino Mega 2560 alatt használt szigetelés megvalósítása

S ha már az összeszerelés részletezésénél tartok, meg kell még említenem a tápforrás felszerelését, illetve annak a kiválasztását is. Az energiát, a kapcsolásomnak két külön helyen kell szolgáltatni, egyrészt 9 - 12 V szükséges az Arduino Mega 2560 működéséhez, másrészt 12 V- ra van szüksége az L298N motorvezérlőnek a motorok megfelelő meghajtására. Ezt a két táplálást én egy helyről oldottam meg, melyet a robotautóm alján helyeztem el. Ez a forrás egy 12 V – os, külön erre a célra, általam összerakott akkumulátorkapcsolást takar, melyet régi laptop tápegység cellákból állítottam össze soros kapcsolás segítségével. Három, egyenként 4V-os akkumulátorból áll az energiaforrás, melyek tökéletesen képesek hajtani a berendezésemet. Természetesen mivel ezek újrahasznosított elemek, így felléptek különböző problémák, mégpedig töltést követően pár cella tönkrement, melyeket cserélni kellett, de szerencsére rendelkezésemre álltak pótalkatrészek. Ezt háromszor végeztem el, mire kialakítottam egy olyan tápforrást, mellyel később a teszteléseket végeztem, s mely remélhetőleg ezentúl kitart a jövőbeli használat mellett is. Végezetül az összeszerelt akkumulátorcellák felszerelését pedig kábelszorítók segítségével oldottam meg, melyek tökéletes tartást biztosítanak, miközben mozog a szerkezetem.

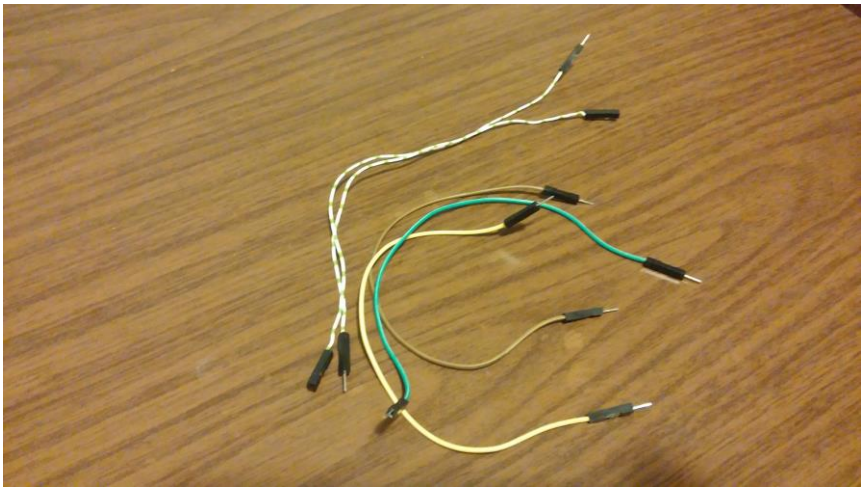


40. ábra: A felszerelt tápforrás helye, a projekt egy kezdeti szakaszában

A fenti ábrán látható egy kapcsoló, mely elhelyezése mellett a projektem készítése közben döntöttem. Úgy gondoltam, hogy a sok tápvezeték lecsatlakoztatása, illetve az L298N - es shieldnek a táplálás megszakítása se nem felhasználóbarát, se nem hibafaktortól mentes, így elhelyeztem egy háromállású kapcsolót. Azért háromállású kapcsolóra esett a választásom, mivel egy ilyennel már rendelkeztem, s így nem jelentett extra költségeket ennek beszerezése. Viszont nekem csak két állásra volt szükségem, így az egyik kapcsolási kört áthidaltam, ezáltal átalakítva kétállású kapcsolóvá az elektronikai alkatrészemet.

41. ábra: A felhasznált háromállású kapcsolóm, melyet kétállásúra alakítottam

A tápforrás, meghajtó rendszer, kapcsoló, és az egyéb már kész elemek felszerelése után elkezdtem összekötni a robotomat, hogy működésre bírjam. Ezekhez a motorok csatlakozásait leszámítva áthidalási kábeleket használtam, mivel maga az Arduino és a legtöbb bővítő lap, mint az LCD kijelző is, szabványos csatlakozókkal van ellátva, melyek biztosítják az áramkörök egyszerű összeszerelését. Viszont tekintettel arra, hogy én ilyen vezetékekből csak kevés darabszámmal rendelkeztem, illetve férfi-női csatlakozású vezeték nem is állt a rendelkezésemre, így kénytelen voltam a legtöbb helyen a saját igényeimhez szabni a csatlakozásokat. Ez alatt azt értem, hogy a már készen beszerzett drótok csatlakozásait eltávolítottam és vagy másfajta csatlakozót szereltem fel helyettük, hogy felemás csatlakozókat kapjak, vagy konkrétan teljesen új vezetékeket állítottam elő. Ez meglepően hosszú feladatnak bizonyult, ugyanis minden egyes csatlakozóról el kellett távolítanom a plasztikaborítást, úgy, hogy később visszarakható legyen. Ezt követően a fém csatlakozó egy részét le kellett csippentenem, mert össze volt nyomva a fém, a vezeték körül és nem lehetett pusztán forrasztással eltávolítani. Majd forrasztózsírt kellett ráhelyeznem a fém csatlakozókra, hogy új vezetéket tudjak forrasztani hozzá. Végezetül visszacsúsztattam a plasztika szigetelőréteket a csatlakozók köré és máris új csatlakozókábeleket kaptam, melyek igényeimnek megfelelő hosszúságúak voltak.

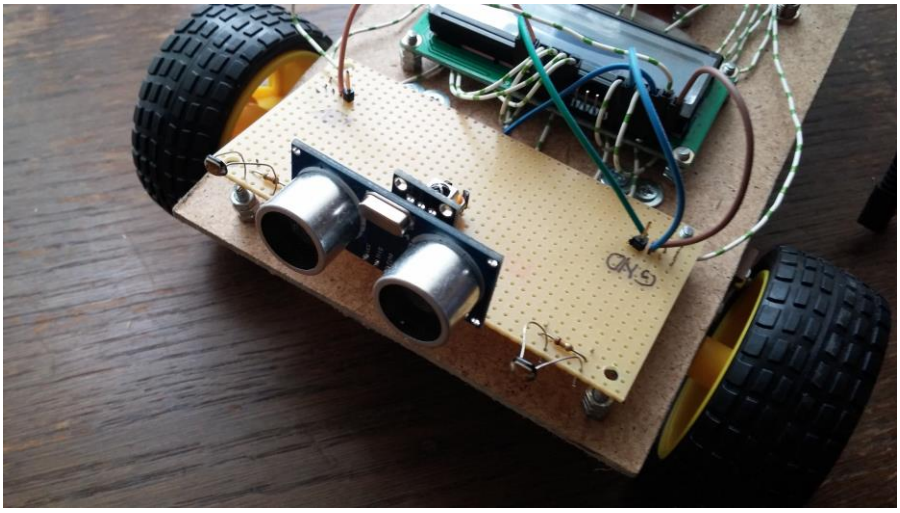


42. ábra: Eredeti és általam összeállított csatlakozó vezetékek

Ez után minden rendelkezésemre álló és már felszerelt elemet összekapcsoltam a kapcsolási rajz alapján és próbaprogramként beindítottam a motorokat, hogy megfelelően működnek-e. Ekkor még nem voltak az alaptestre rögzítve a szenzorok, se az akadályt érzékelők, se a vonalat, se a fényforrásra reagáló fényérzékelő ellenállások. Ezeket próbapanelra készítettem el, melyeket már előre kifúrtam és méretre vágtam.

3.6. A szenzorok összeállítása próbalapra

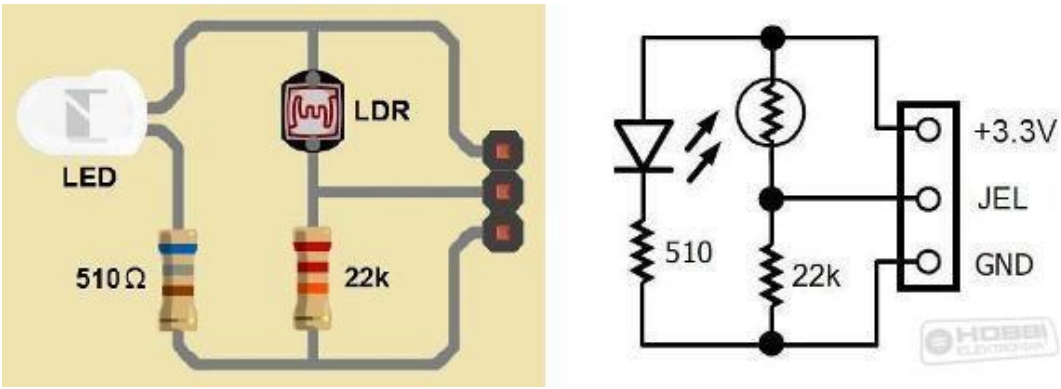
Két külön próbalapra készítettem el a szenzorok kapcsolását, az egyiket felülre, az autóm elejére szereltem, még a másikat szintén az elejére, csak alul helyeztem el, hogy a vonalkövetés feladatát el tudja látni. A felső próbalap 50 mm x 100 mm méretű, s azért választottam ekkora nagyságot neki, hogy a rendelkezésemre álló szenzorokon kívül a jövőbeli továbbfejlesztésre is hagyjak megfelelő mennyiségű helyet. Erre a felső lapra az ultrahangos szenzort, az IR szenzort, és a fénykövetéshez szükséges kapcsolást helyeztem el, valamint az LCD kijelző működéséhez szükséges 10 kΩ - os változtatható ellenállás is itt kapott helyet. Ezen elemek működésükhöz, mindnyájan 5 V - os feszültséget használnak, így létrehoztam a próbalapon egy 5 V – os csomópontot és egy GND csomópontot is, melyek az L298N - es motorvezérlőről kapják a táplálást. Úgy döntöttem, hogy az LCD kijelzőnek szükséges GND és 5 V kapcsolatokat is innen oldom meg, ezáltal is áttekinthetőbbé téve a munkámat. A sémák, melyek alapján dolgoztam, azok, melyeket a tesztelések során alkalmaztam, s amelyek a „3.3. Az alkatrészek és kapcsolások kipróbálása” című fejezetemben is megtalálhatóak, így ezeket nem részletezném még egyszer.



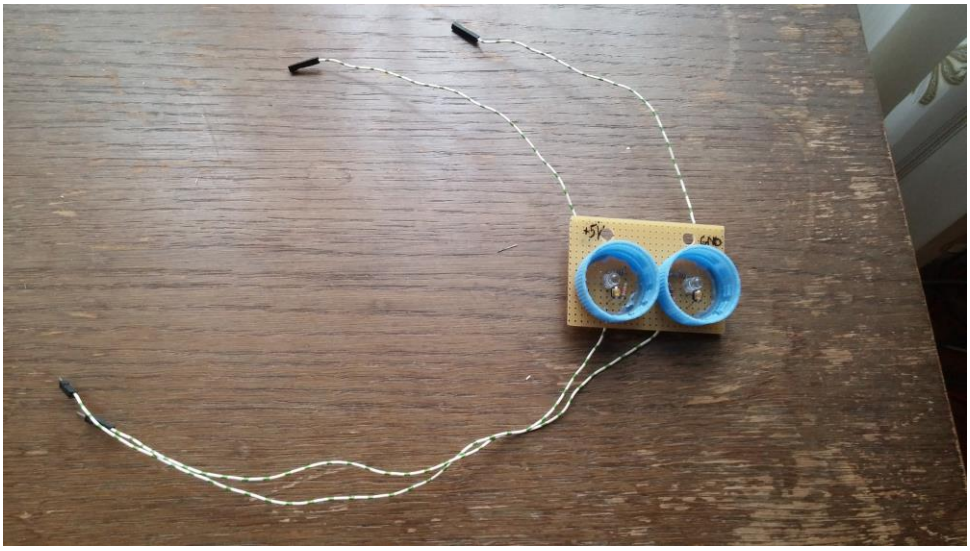
43. ábra: A felső szenzorlap elülső látszata

44. ábra: A felső szenzorlap forrasztásai és áthidalásai

A másik szenzorlap, mint már említettem alul helyezkedik el, és a vonal követés feladatához szükséges talaj feltérképezést végzi. A működési elve, ugyanaz, mint a fénykövetésnek, csak itt szükségünk van egy fényforrásra, mely megvilágítsa a talajt és a visszaverődő fénysugarak alapján a fényérzékelő ellenállás eldönti, hogy az autónk hol helyezkedik el a vonalhoz képest. Ezt a kapcsolást nem teszteltem le előre. A kapcsolást, melyet használtam, egy internetes oldalról szedtem, amely 3.3 V - ra ír elő tápforrásnak, de számolással és teszteléssel megbizonyosodtam arról, hogy 5V - on is ugyanúgy működik, csak bizonyos nagyságrenddel nagyobb eredményt ad vissza a fényérzékelő ellenállás. Végezetül az 5V – os feszültségről való működés mellett döntöttem, mivel ezt a tápfeszültséget könnyen el tudtam érni az autóm elején található szenzor panelről és nem kellett vezetéket húznom magából, az Arduino lapomból, ami biztosítani tudta volna a 3.3 V – ot. A szenzorokat, a megvalósítást követően műanyagflakon kupakokkal zártam el, hogy ne zavarják a fényérzékelő ellenállásaimat a környezeti fények. Ezen kívül azonban még szükség volt a LED - eket is szigetelőszalaggal ellátnom, hogy fényük ne világítsanak rá közvetlenül az ellenállásokra, kizárólag a visszaverődő fényeket érzékeljék az érzékelők.

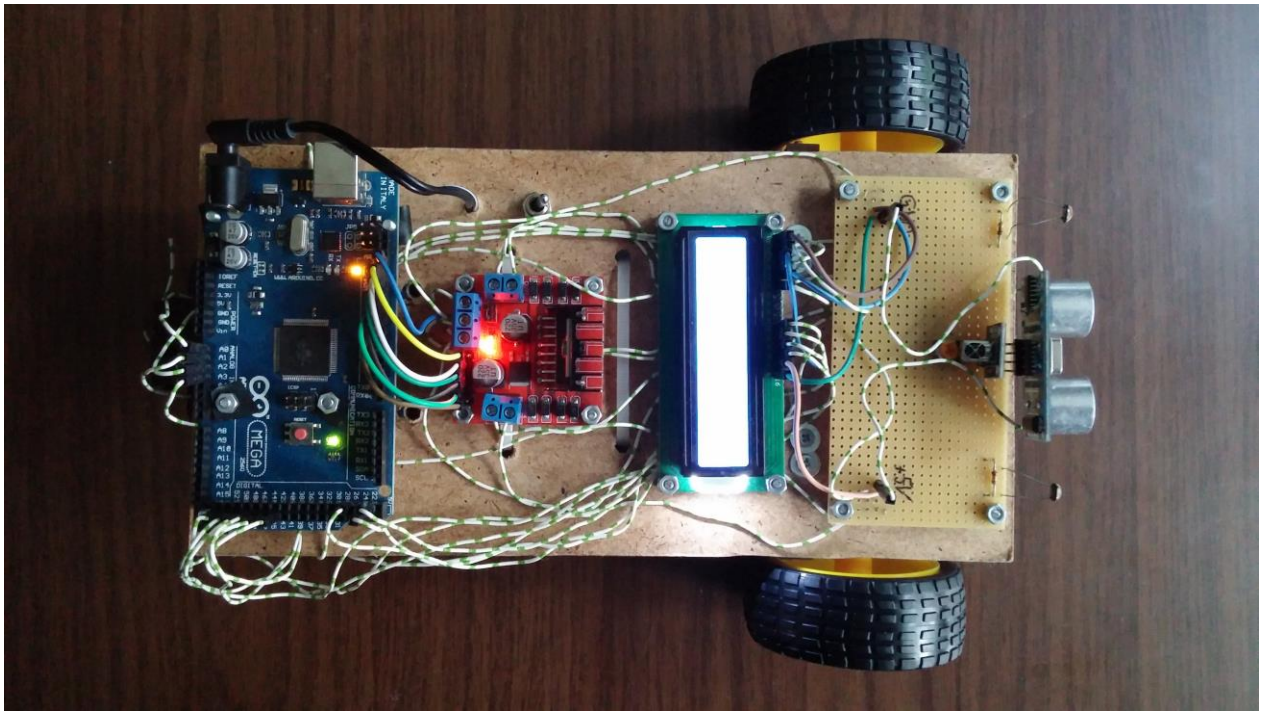


45. ábra: A vonalkövető kapcsolás sémája [19]

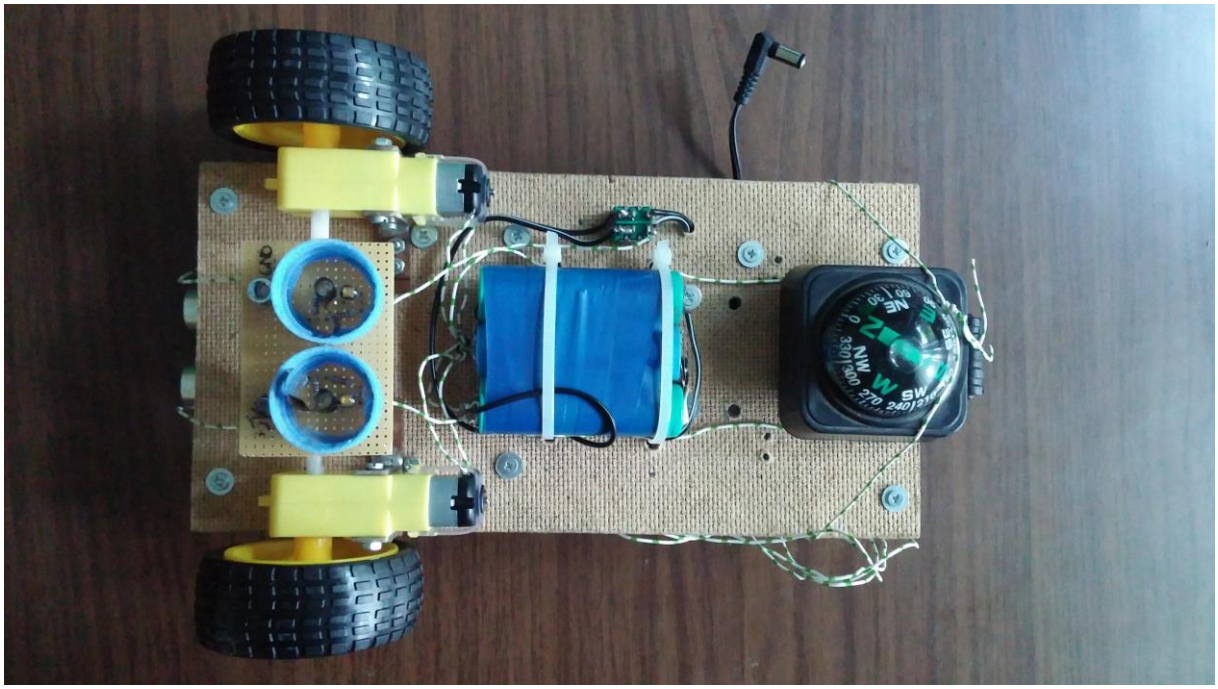


46. ábra: A vonalérzékelő szenzor látszata

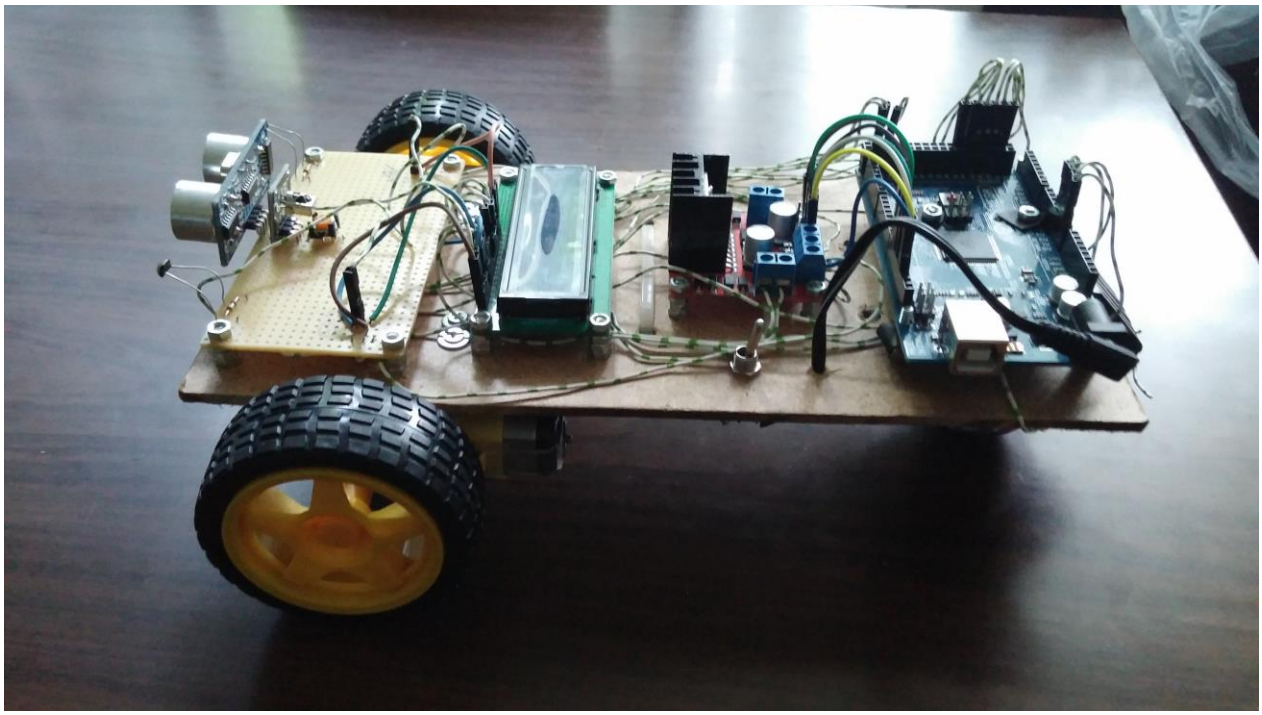
3.7. A megvalósított szerkezetem kinézete



47. ábra: A többfunkciójú autóm felülnézete



48. ábra: A többfunkciójú autóm alulnézetből

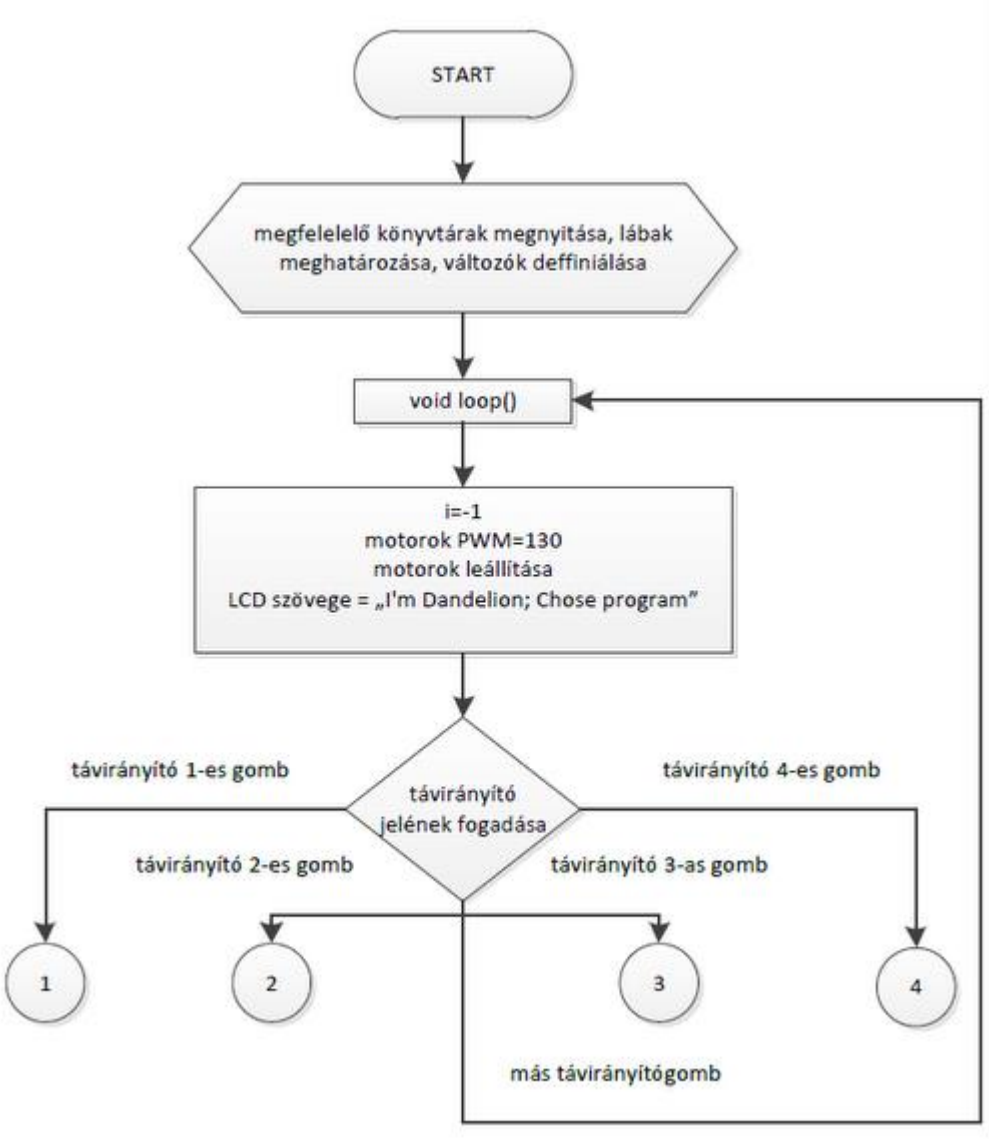


49. ábra: A többfunkciójú autóm oldalnézetből

50. ábra: A többfunkciójú autóm távirányítója

3.8. A robotom működése és folyamatábrái

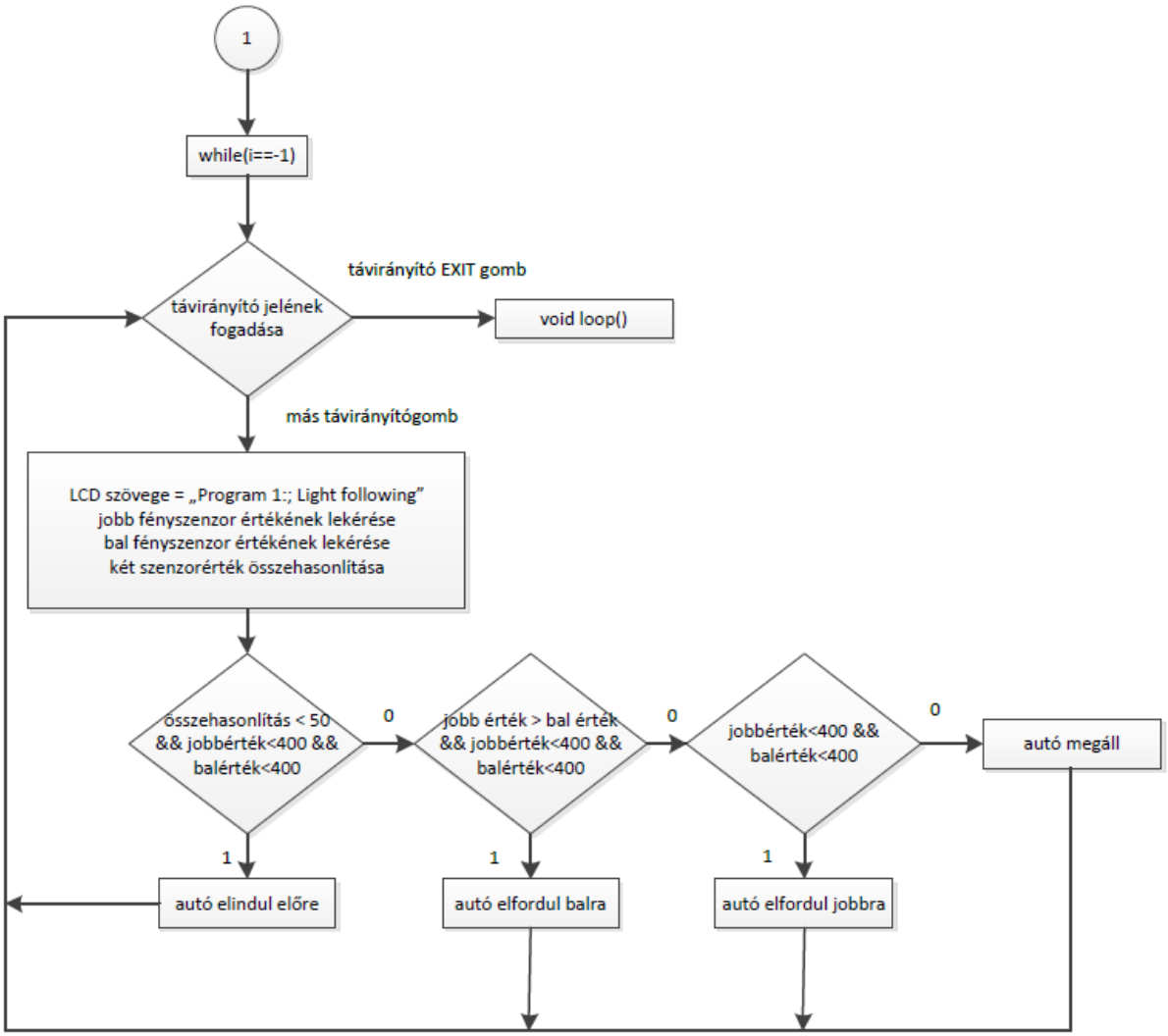
Az autóm, melyet összeállítottam 4 feladatot képes ellátni, melyek különböző alprogramokban helyezkednek el és melyet maga a főprogramként funkcionáló void loop() program vezérel. A vezérlés alatt azt értem, hogy folyamatosan figyeli a távirányítóról érkező jeleket és amennyiben egy olyan gombot nyomunk le rajta, mely valamilyen funkciót takar, akkor átirányítsa a mikrovezérlőt a megfelelő programrészhez. Abban az esetben, ha lenyomjuk az 1 - es gombot, a fénykövetés program indul el, ha a 2 - est, az akadályelkerülés, ha a 3 - ast, a vonalkövetés, még a 4 - es gombbal szabadon mozgathassuk a robotautót a térben. Tekintettel, hogy szerteágazó programról beszélünk, így mint folyamatábra szinten, mint elmagyarázás szinten külön-külön fogok beszélni róluk. A folyamatábrákat Microsoft Visio 2010 nevezetű programban készítettem.



51. ábra: Az alprogramok között váltó főprogram folyamatábrája

3.8.1. A fénykövetés algoritmusa

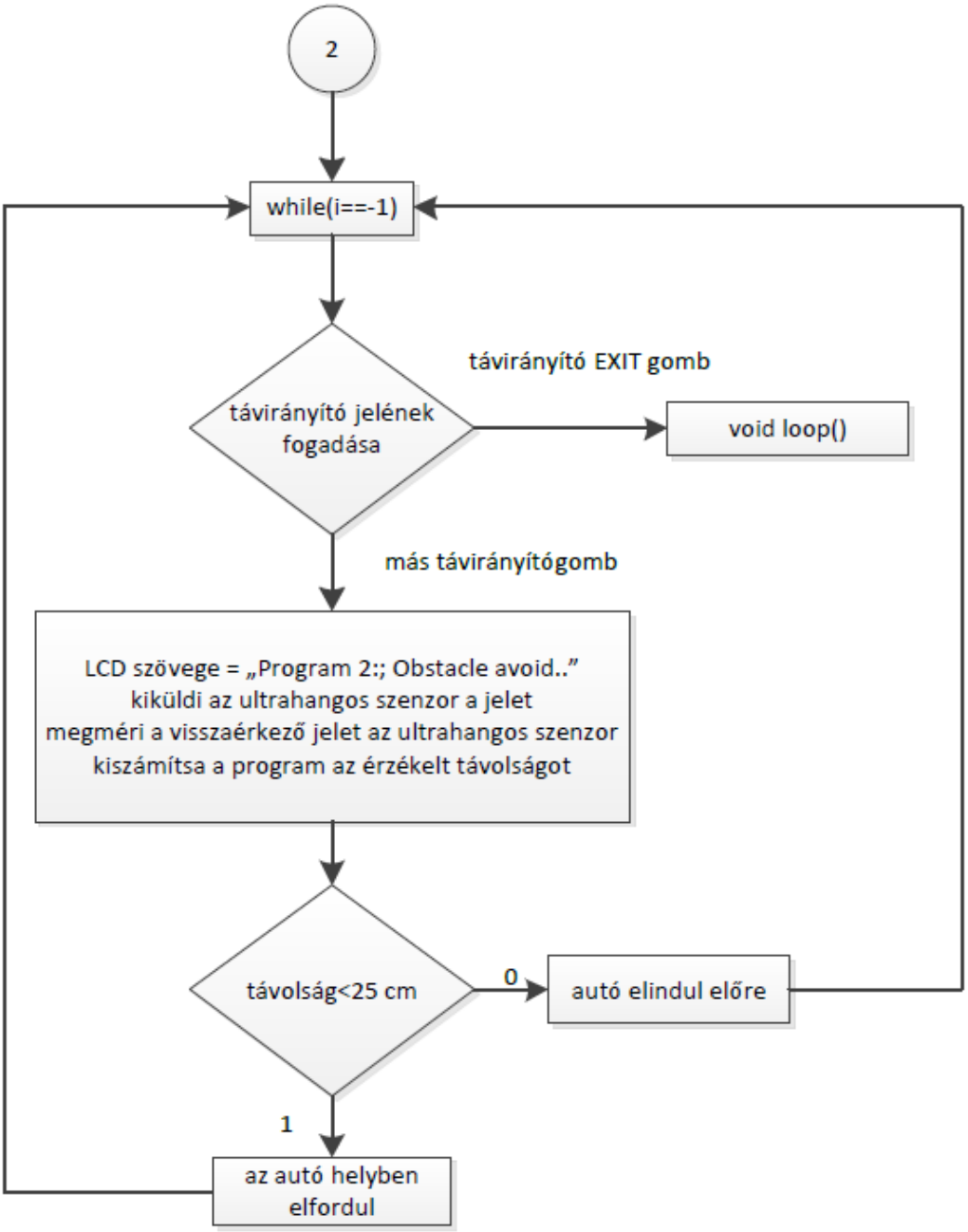
A fénykövetés programrész működéséhez szükség van az autó elején elhelyezkedő 2 fényérzékelő ellenállásra, melyek fény hatására csökkenő analóg jelet adnak vissza az Arduino A2 (bal szenzor) és A3 (jobb szenzor) bemenetére. Ezt a két értéket hasonlítsa össze a program és attól függően, hogy épp melyik ellenállásra esik a nagyobb fénysugár mennyiség, eldönti, hogy melyik irányba forduljon el/induljon el a robot. Ha a két szenzor értéke közötti különbség abszolút értékben kisebb, mint 30, akkor egyenest indul el az autó. Ha nem, s a bal érték a kisebb, akkor balra fordul a test, ha a jobb szenzor a kisebb, akkor jobbra kanyarodik a szerkezet. Valamint beiktattam egy határértéket is, mely alatt működnek a szenzorjaim, a célból, hogy a környezeti fények ne zavarjanak be a program működésébe, csak a vezérlő fényre reagáljon a robotom.



52. ábra: A fénykövetést elvégző programrész folyamatábrája

3.8.2. Akadálykikerülés algoritmusa

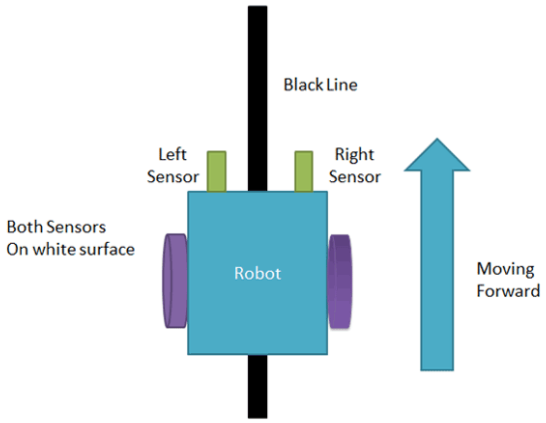
Az akadálykikerülés programrésznél a mikrovezérlő azt figyeli, hogy az autó elején elhelyezkedő HC-SR04 ultrahangos szenzor előtt van-e akadály. Ezt úgy valósítja meg, hogy kiküld a szenzor egy jelet, melynek a visszaérkezési idejét számolja. Ez alapján pedig egy megfelelő képlet alapján kiszámítható a tárgy pontos távolsága centiméterben. Ha 25 cm közelre érkezik egy tárgy, akkor elfordul az akadálytól és tovább folytassa az útját, ha nincs előtte akadály, akkor egyenesen megy és pásztázza az elébe kerülő tárgyakat.



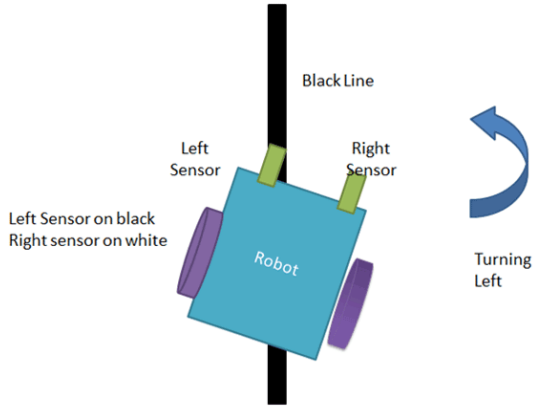
53. ábra: Az akadályelkerülést elvégző programrész folyamatábrája

3.8.3. Vonalkövetés algoritmusa

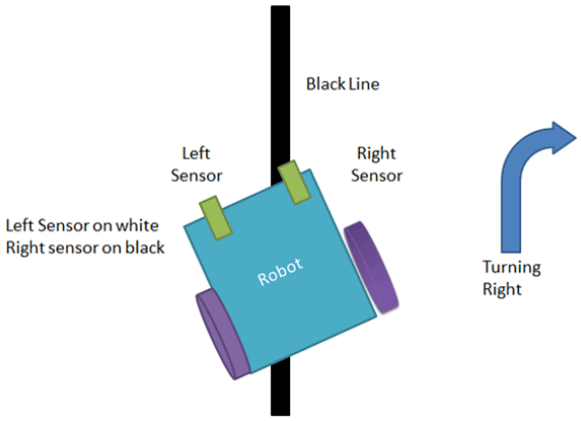
Ez a programrész nagyon hasonlóan működik, mint a fénykövetés, viszont itt nem az előtte lévő fényforrást figyeli a program, hanem az alatta lévő visszaverődésekre reagál. A test alatt el van helyezve két fényérzékelő ellenállás olyan kapcsolásban, melyek mellé egy-egy fehér LED is társul. A LED - ek fénye megvilágítsa a talajt és abban az esetben, ha sötét részt világítanak meg, akkor a fényérzékelő ellenállások által szolgáltatott analóg jelek értéke lecsökken. Tehát ha egy fekete vonalat húzunk a robotautó elé, illetve egy fekete szigetelőszalagot, akkor a szenzorok érzékelni tudják, hogy melyikük van már a vonal felett és merre kell elmozdulnia az autónak, hogy továbbra is a vonalat kövesse a test. Mivel sok irányváltást végeznek a motorok és gyorsan, néha nem volt elég erejük, hogy hirtelen meginduljanak és megakadt a robot mozgása. Ezt úgy korrigáltam, hogy abban az esetben, mikor a szenzorok hatására előrefelé indulna az autó, 4 ciklus lezajlási ideig nagyobb PWM – el mozog a test, majd ez lecsökken, abban az esetben, ha már átvészeltük az indulási nehézségeket. A PWM csökkentésére azért van szükség, mert ha túl gyorsan halad a szerkezet és kanyarhoz érkezik, akkor képes átcsusszanni felette és elvesztik a vonalat a szenzorok, mivel valamiért túl kicsi a mintavételezési idő, aminek a miértjére nem jöttem rá, hiába kerestem utána az interneten is.



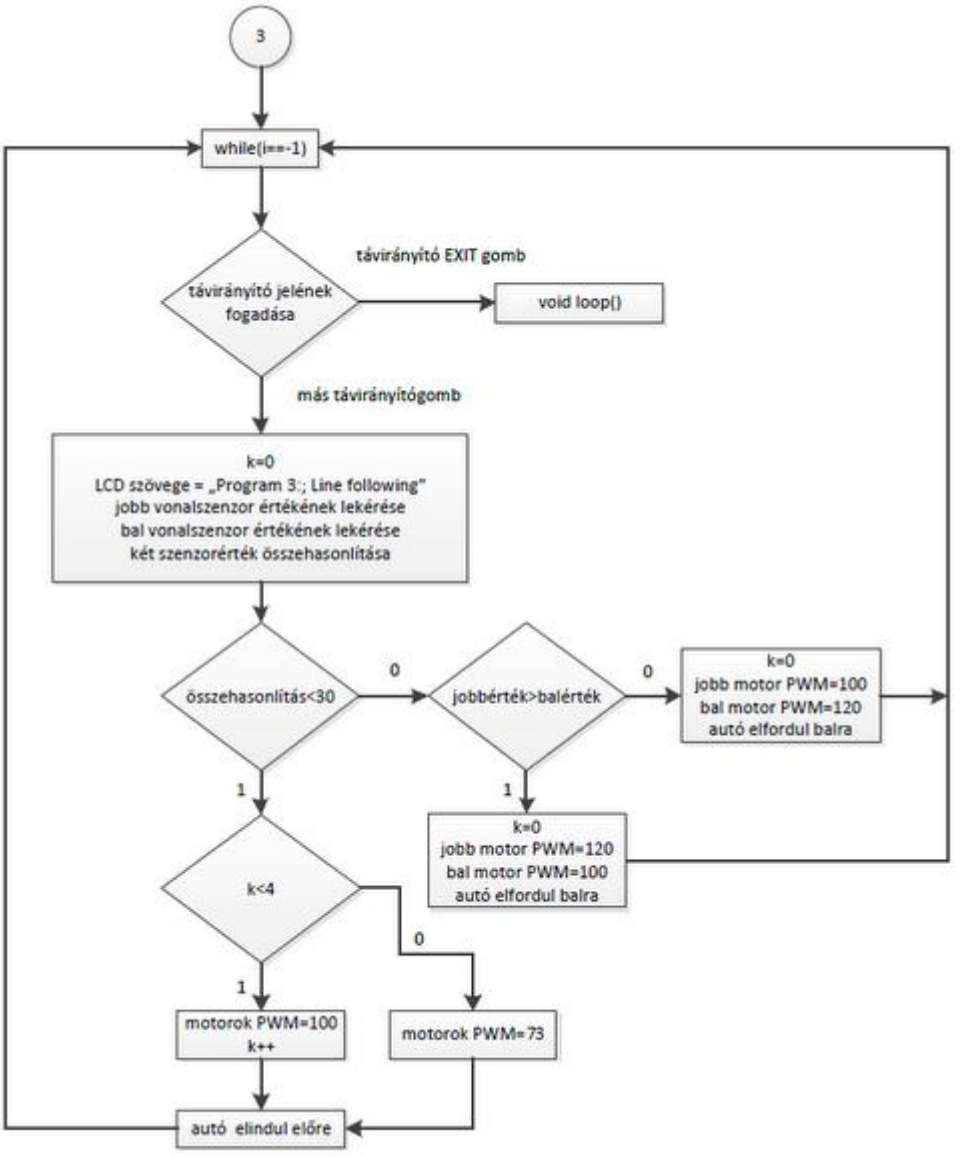
54. ábra: A robot egyenesen haladva követi a fekete vonalat [20]



55. ábra: A robot balra fordul, hogy továbbra is kövesse a vonalat [20]



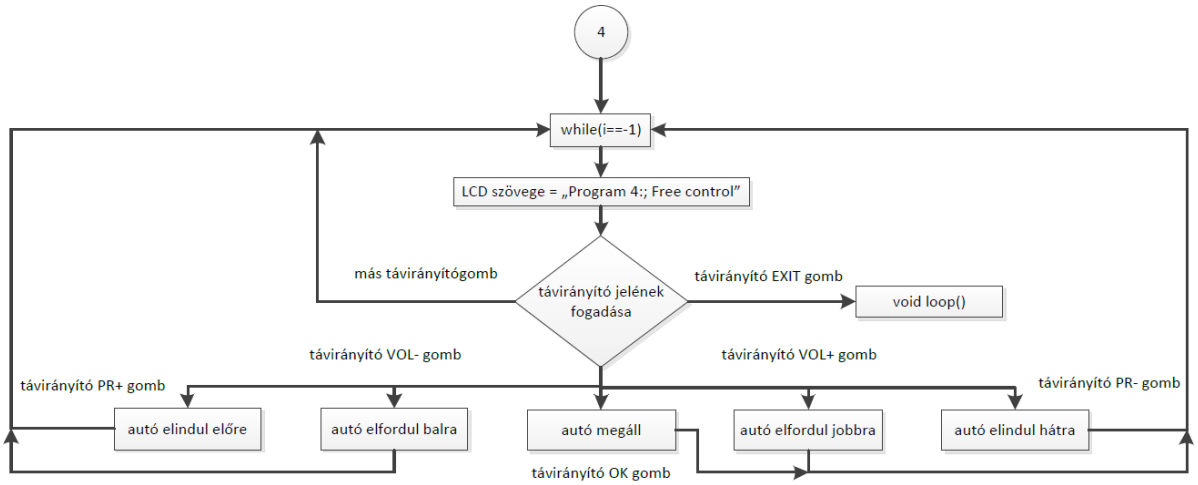
56. ábra: A robot jobbra fordul, hogy továbbra is kövesse a vonalat [20]



57. ábra: A vonalkövetést elvégző programrész folyamatábrája

3.8.4. A szabad vezérlés algoritmusa

Ez a programrész mindössze csak egy extra funkcióként került bele a feladatomba. Nem végez semmi bonyolult feladatot, mindössze megnyit egy alprogramot, mellyel a megfelelő gombok segítségével szabadon lehet vezérelni az autót. Ez a továbbfejlesztések folyamán lehet érdekes funkció, amennyiben feltérképezni szeretnénk egy bizonyos területet egy fajta adat megszerzése érdekében és azt be kell szabadon járni, kedvünk szerint. Ha ebben az üzemmódban PR+ gombot nyomunk a távirányítón, akkor előre kezd el haladni az autó, ha PR- gombot, akkor hátrafelé, ha VOL- - at, akkor elfordul balra, ha pedig VOL+ - ot, jobbra fordul a test. OK gomb megnyomására pedig megszakítsa az adott mozgást. Erre azért van szükség, hogy hirtelen le tudjuk állítani, s ne csapódjon neki semminek se a robot, illetve könnyebb irányítást is tesz lehetővé, ha az egyes irányváltásoknál mindig megállítsuk a szerkezet mozgását.



58. ábra: A szabad vezérlést elvégző programrész folyamatábrája

3.9. A megvalósítás során fellépő problémák és azok elhárítása

A munkám során számos nehézséggel találkoztam, melyek megoldása közben megannyi tapasztalattal gazdagodtam. Számosat már megemlítettem a szakdolgozatomon belül, ebben a fejezetben csak egy kisebb áttekintést szeretnék nyújtani ezekkel kapcsolatban.

- A legelső problémám az IR szenzorkészlet tesztelésénél jelentkezett, amikor is a működéséhez szükséges megfelelő könyvtár beszerzése után, a készlettel kapott távirányító különböző IR értékeket adott vissza. Elsőnek arra gyanakodtam, hogy hibás szenzort kaptam, s szerezhetek be egy újat. Azonban számos próbálgatás és különböző szögekből történő jelérzékelésből azt a következtetést vontam le, hogy a távirányító hatósugara nem egyezik meg a feltüntetett értékekkel. Ezért a munkám megvalósítását egy itthoni távirányító segítségével oldottam meg, mely eredetileg egy műholdvevő vezérlésére szolgált, és amely mesze nagyobb hatótávolságokkal rendelkezett és több szögből is a megfelelő IR jelet biztosította.

- A következő nagyobb problémám a test kivágásánál jelentkezett, amikor is lombfűrész nem állt rendelkezésemre, viszont méretre kellett vágnom a lezonit lapot. Ezt a már bemutatott kiscirkula (34. ábra) segítségével oldottam meg, amely viszont nem ekkora erőkifejtésre volt tervezve. Ezért párhuzamos vonalakat kellett vágnom a vágási vonalak azon oldalán, mely hulladékként végezte, hogy folyamatosan ki tudjak törni részleteket a lapomból s ez által csökkenteni tudjam a lapra nehezedő súrlódást, mely jelentősen megnövelte a motor terhelését. Ennek a művelet köszönhetően néhol kicsit eltértem a kitűzött méretektől 1-2 mm-el, s ez miatt az egyenetlen oldalszéleket vágás után smirglivel kellett kiegyengetnem.

- Problémába ütköztem a motorok felszerelését illetően is, hiszen nem állt rendelkezésemre a felerősítésüket lehetővé tevő tartóelem. Ezeket kétoldalasan rézfóliázott üvegszálas nyomtatott lapok segítségével állítottam végül össze, mivel itt nem volt szükséges, hogy az elem szigetelve legyen, s sokkal könnyebb volt az egyes részelemeket a széleknél forrasztással összeilleszteni, mintha valamilyen más módszerrel oldottam volna meg.

- Ezután összeszerelés követően lépett fel problémám, mikor is két elem rögzítésénél gondba ütköztem, az akkumulátorom rögzítésénél, illetve magának az Arduino Megámnak a felszerelésénél. A tápforrás testre erősítése azért okozott először nehézséget, mivel foglalat híján egyszerű csavarokkal nem tudtam megoldani a rögzítést. Ekkor két ötletem merült fel, hogy vagy drótokkal fogom rögzíteni a cellákat, vagy kábelek összefogásánál használt plasztikakötegelőket fogok alkalmazni. Végül a profibb és erősebb tartás végett a kábelkötegelők mellett döntöttem. A vezérlő lap testre szerelése pedig azért nem volt egyszerű, mivel mint kiderült az Arduino Mega 2560 nem 3-as csavarokkal történő rögzítésre van előlátva, még ha a rajta lévő furatok erre is engednek következtetni, hanem távtartókkal való felszerelésre lett kitalálva. Ez abban látszódik, hogy némely anyacsavar, a szükséges megemelések után hozzáért volna a lap hátuljához s ez által vezetést okozott volna. Ezt kiküszöbölve bicikligumiból kivágott kis négyzeteket használtam az anyacsavarok elszigeteléséhez a laptól (49. ábra).

- Ez után a végső tesztelések során lépett fel egy olyan problémám, amellyel először nem számoltam. A két motormeghajtású robotautóm hátsó kerekének egy kínai üzletekben kapható, olcsó kisebb forgókereket láttam elő, mely, mint kiderült nagyobb tömegekre volt előlátva. Ez rögtön meg is mutatkozott, mikor az autóm elfordulást követően egyenesen kezdett el haladni. A meghajtó elemek ugyan elindultak előre, ugyanazzal a sebességgel, viszont tekintettel, hogy a hátsó kerékre nem nehezedett tömeg, az nem fordult el a megfelelő irányba, és elkezdett körbe-körbe körözni a szerkezet. Ezt próbáltam úgy megoldani, hogy a csapágyas forgó részét megkentem, illetve a kerék súrlódását is megnöveltem gumibevonat által, azonban egyik sem bizonyult megfelelő megoldásnak, noha javítottam valamelyest az egyenesbe történő átálláson. Ekkor ötlött az eszembe az az ötlet, hogy hátsó kerék helyett csak egy gömb tartóelemet kellene a robotautóm hátuljára erősíteni, melynek elég kicsi a tapadási tényezője, hogy megfelelő erőkifejtés hatására mozogni tudjon az egész szerkezet, viszont nem túl kicsi, hogy ne tudja biztosítani a pontos mozgást. Ezután találtam rá itthon egy régi iránytűre, melynek a borítása plasztika, gömbölyű alakzat, s mely felerősítve a testemre megoldotta az problémát. Ezután nem kellett hihetetlenül nagy erőt alkalmaznom a robotom mozgatásához, viszont tökéletes stabilitást értem el. A gond csak a kanyarodásnál lépett fel, amikor is az elterjedt egy motoros forgás (egyik motor forog, másik megáll) nem volt képes elég erőkifejtést végezni, hogy elforduljon a szerkezetem. Ezt azzal korrigáltam, hogy átálltam a helyben fordulásra, amikor is, ha fordulásra van szükség, az egyik kerék elindul előre, még a másik motor, ellenkező irányba kezd el forogni, így megvalósítva a szükséges kanyarodást.

59. ábra: A régi és az új kerék elhelyezve egymás mellé

- Az utolsó nagy gondomat pedig a vonalkövetés okozta. Az egyik fényérzékelő ellenállásom rossznak bizonyult, a tesztelések során, s miután ki is cseréltem, még mindig nem követte megfelelően a fekete vonalat a szerkezetem. Arra a megállapításra jutottam, hogy ez annak tudható be, hogy túl gyorsan mozog a robotautóm, mikor egyenesen kellene haladni, és így elsiklik az elébe kerülő kanyarok felett. Ezt úgy tudtam megoldani, hogy levettem az autó sebességét, mikor nem érzékelte a fekete vonalat, viszont kanyarodásnál megnöveltem a PWM-et, tekintve azt a tényt, hogy sajnos fordulásnál nagyobb erőkifejtés szükséges ahhoz, hogy ne akadjon meg a szerkezetem. Ezeket a változtatásokat elvégezve, már tökéletesen működött a programrészletem fehér hammeron.

4. Összegzés

A projektem készítése során számos különböző ismereteket kellett használnom, vagy éppen szereznem, ha nem rendelkeztem abból a témakörből megfelelő szaktudással. Bele kellett kóstolnom a **gépészet** világába, amikor megterveztem a robotautóm fizikai kinézetét, tervezőprogramban, illetve a test előállításánál is ebben a témakörben tevékenykedtem. **Elektronikai** szinten szükségem volt az iskolában tanult kapcsoláskészítő szoftver ismeretére, forrasztások elkészítésére, kapcsolatok kialakítására. **Informatikai** ismeretek nélkül nem tudtam volna a mikrovezérlőbe kerülő programnak a megtervezését véghezvinni, ezen kívül pedig **irányítástechnikai** és **robotikai** problémákkal is sikerült találkoznom, mikor összehangoltam a motorok mozgását, valamint a különböző szabályozások kialakítását végeztem.

A kitűzött célokat sikerült megvalósítanom, bár az oda vezető út nem volt egyszerűnek mondható. Megismerkedtem magának az Arduino Mega 2560 – nak a felépítésével, működésével és közepes szintű használatával. Összebarátkoztam a megvásárolt bővítő lapoknak (L298N motorvezérlő, LCD kijelző, HC-SR04 ultrahangos szenzor, IR szenzor) a bekötésével, üzembe helyezésével és működésük összehangolásával. Rájöttem, hogy az LCD kijelző modulnak muszáj a változtatható ellenállás a kapcsolásában, mellyel a betűk élessége állítható, s ha ezt az elemet kihagyjuk nem kapunk éles képet. A tesztelések során találkoztam az egyes alkatrészek hiányosságaival, melyek valószínűleg abból adódtak, hogy klón elemekkel dolgoztam. Az IR szenzor érzékenysége nagyban eltért a megadott értékektől, s a HC-SR04 se azzal a hatótávolsággal rendelkezett, mint amennyi szerepelt a leírásában. A különféle elemekkel történő ismerkedés után elkezdtem tervezni a többüzemű robotautómat, a test megszerkesztésével nekikezdve. Első lépésként elhelyeztem az egyes elemek helyzetét az autótesten, melyet 120 mm x 240 mm – es lezonitból alkottam meg. Ehhez a gépészmérnökök által gyakran használt Solid Works programcsomagban dolgoztam. Ez után elkezdhettem a fizikai összeállítást, mely során számos különböző tapasztalatot szereztem s a kézügyességi képességekeimet gyarapítottam. Az elemek s alkatrészek elhelyezését követően kezdtem neki a programírásnak és a pontos működési módozatok megalkotásának, ám még előtte próbalapokra különböző szenzorköröket állítottam össze. Ezekre az esztétikus kinézet és egyszerű elhelyezésen kívül azért volt szükség, mivel minden itt elhelyezett elem 5 V – os tápfeszültségről működik, s könnyen létre tudtam hozni olyan csomópontokat, melyek ezt a feszültséget szolgáltatják. Valamint a szenzorok egy bizonyos részét el kellett különítenem egymástól, hisz máshova kerültek a vonalat érzékelő szenzorok és máshova a fénykövetésnél használt elemek. A fizikai test elkészülte után folyamatábrákat készítettem, Microsoft Visio szoftvercsomag segítségével, miközben körvonalazódott számomra, hogy pontosan mit s hogyan is fogok leprogramozni az ATmega2560 – as mikrovezérlőmbe. A folyamatábrát 5 részre bontottam, hiszen gyakorlatilag 5 programrész dolgozik a forráskódon belül. Ezek a főmenüként szolgáló üzemmód választás, a fénykövetést elvégző alprogram, az akadályelkerülés megvalósítása, a vonalkövetés és szabad vezérlés biztosítása. Ezen folyamatok közül a vonalkövetést végző funkcióval gyűlt meg a legjobban a bajom, noha ez egy alapproblémának tekinthető. A gondom ott kezdődött, hogy az én testem nagyobb, mint az alap robotautók, melyekkel általában találkozni lehet az interneten, s így a megvalósított vezérlés sokkal problémásabb. A kerék elhelyezését sajnos elrontottam, túlságosan hátra került a szerkezetemen, valamint csúszott is, és így nem tudtam biztosítani a megfelelően pontos irányítást. Ezt a problémát a kerék lecserélésével hárítottam el, valamint a szenzorokat is megfelelően össze kellett hangolnom, ugyanis különbségek voltak az általuk szolgáltatott adatokban. A kerékcsere által a többi üzemmódban is javult az irányítás, s noha furcsán néz ki a használt megoldásom, hogy a hátsó rész egy iránytűvel van felfogatva, tökéletesen ellátja a feladatát.

A projektemet a jövőben ki szeretném még bővíteni egy giroszenzor modullal, melynek segítségével az LCD kijelző rendeltetését még fontosabbá tudnám tenni, hiszen itt megjeleníthetnék olyan adatokat is, mint a talaj meredeksége, mert ez a modul méri a 3 dimenziós koordinátarendszerben történő elmozdulásokat. Ezen kívül szeretném még új funkciókkal bővíteni a robot működését, mint amilyen a labirintusból való kitalálás. Erre már a munkám elején is gondoltam, de időhiány miatt végül nem sikerült megvalósítanom. Illetve szeretném majd az ultrahangos szenzorok számát is növelni, hiszen egy ekkora szerkezetre, mint amekkorát én alkottam, nem elég lefedést biztosít csupán egy darab HC-SR04 – es elem. De mindezek előtt talán egy normális hátsó kerék keresése lenne az első dolgom, hiszen a mostani megoldás nem mondható épp professzionálisnak.

Irodalom és felhasznált források

[1] J. Róbert, *Mikrovezérlık oktatása - Tananyag.*

[2] H. János, „Mikroprocesszorok alkalmazása," Edutus Főiskola, 2012, 2012.

[3] D. M. István, „Mikrovezérlők jegyzet".

[4] R. G. P. elektronikák, „Málna PC," Málna PC, [Online]. Available: http://malnapc.hu/arduino/az-arduino-platform/.

[5] Fizikus, „HobbiRobot.hu - Robotika mindenkinek!," HobbiRobot.hu - Robotika mindenkinek!, [Online]. Available: http://hobbirobot.hu/content/arduino-kezdoknek.

[6] S. s. f. b. által, „Wikipedia," Wikipedia, [Online]. Available: https://hu.wikipedia.org/wiki/Arduino.

[7] A. cég, „Arduino," Arduino, [Online]. Available: https://www.arduino.cc/en/Main/ArduinoBoardMega2560.

[8] Tronixlabs, „Tronixlabs," Tronixlabs, [Online]. Available: http://tronixlabs.com.au/robotics/motor-controllers/l298n-dual-motor-controller-module-2a-australia/.

[9] survy2014, „ebay," [Online]. Available: http://www.ebay.com/itm/smart-Car-Robot-Plastic-Tire-Wheel-with-DC-3-6v-Gear-Motor-for-arduino-/400985432465.

[10] ElecFreaks, „micropiK," [Online]. Available: http://www.micropik.com/PDF/HCSR04.pdf.

[11] YourDuino, „YourDuino," YourDuino, [Online]. Available: http://yourduino.com/sunshop2/index.php?l=product_detail&p=369.

[12] kingelectronics15, „ebay," ebay, [Online]. Available: http://www.ebay.com/itm/1Pc-New-Infrared-IR-Wireless-Remote-Control-Module-Kits-for-Arduino-/221892279446?hash=item33a9cf7c96:g:FZAAAOSwuTxV~~1-.

[13] Kushagra, „EngineersGarage," EngineersGarage, [Online]. Available: http://www.engineersgarage.com/electronic-components/16x2-lcd-module-datasheet.

[14] oddwires, „oddwires," oddwires, [Online]. Available: http://www.oddwires.com/hc-sr04-ultrasonic-distance-sensor/.

[15] Brainy-Bits, „Brainy-Bits," Brainy-Bits, [Online]. Available: https://brainy-bits.com/tutorials/ir-remote-arduino/.

[16] MercadoLibre, „MercadoLibre," MercadoLibre, [Online]. Available:

http://articulo.mercadolibre.com.co/MCO-421490518-puente-h-l298n-driver-l298-motor-dc-pap-arduino-pic-robot-_JM.

[17] Arduino, „Arduino,” Arduino, [Online]. Available: https://www.arduino.cc/en/Tutorial/HelloWorld.

[18] Duino-Robotics, „Duino-Robotics,” Duino-Robotics, [Online]. Available: http://www.duino-robotics.com/ben.html.

[19] Fizikus, „Hobbielektronika.hu,” Hobbielektronika, [Online]. Available: http://www.hobbielektronika.hu/cikkek/5000ft-os_launchpad_robot.html?pg=4.

[20] Saddam, „Circuit Digest,” Circuit Digest, [Online]. Available: http://circuitdigest.com/microcontroller-projects/line-follower-robot-using-arduino.

5. Mellékletek

5.1. Elektronikus melléklet

- A végső, Arduino Mega 2560 – ba feltöltött C programkód megjegyzésekkel ellátva
- A tesztelések során felhasznált programkódok C nyelven megjegyzésekkel ellátva
- A szakdolgozatom szövege Word dokumentumban
- A szakdolgozatom szövege PDF dokumentumban
- A Solid Works 2013 programcsomagban készített fájlok
- Az Altium Designer Winter 09 programcsomagban készített fájlok
- A Microsoft Visio 2010 – ben készített fájlok
- Különböző képek, melyek az elkészítés során készültek
- A felhasznált irodalom egy része PDF dokumentumban