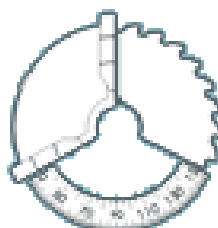


SZENT ISTVÁN EGYETEM GÉPÉSZMÉRNÖKI KAR

GÉPÉSZMÉRNÖKI
SZAK

MECHATRONIKA
TANSZÉK



VISSZHANG-LOKÁCIÓ ALAPÚ TÁJÉKOZÓDÁST SEGÍTŐ RENDSZER TERVEZÉSE, ÉPÍTÉSE ÉS VIZSGÁLATA

ECHOLOCATION-BASED ORIENTATION ASSISTANT SYSTEM DESIGN, CONSTRUCTION, AND TEST

Készítette:

Erdélyi Viktor Ferenc

Belső konzulens: Dr. Jánosi László egyetemi tanár, tanszékvezető

Külső konzulens: Dr. Soumelidis Alexandros MTA-SZTAKI tud. főmunkatárs

**GÖDÖLLŐ
2014**

Tartalomjegyzék

1. Bevezetés.....	1
1.1. Célkitűzés	1
1.2. Témaválasztás	1
2. „Echolocation”	2
2.1. Működési elv	2
2.2. FM-sweep.....	3
2.3. CF.....	4
2.4. Idegi mechanizmusok az agyban.....	4
3. Ultrahangos távolságmérő rendszerek áttekintése	5
3.1. Ultrahangos távolságmérő berendezések napjainkban.....	5
3.2. Ultrahangjeladók és -vevők bemutatása.....	6
3.3. Mikrokontroller, mint a beágyazott rendszer alapja	8
4. Készülék tervezése	10
4.1. Előzetes megfontolások	10
4.2. Tervezési irányvonal.....	10
4.3. Alkalmas eszközök kiválasztása.....	11
4.4. Tervezési szakasz.....	12
5. A kísérleti eszköz megépítése és vizsgálata.....	13
5.1. A tesztelő - megjelenítő eszköz tervezése	13
5.2. A távolságmérő áramkör megépítése	15
5.3. A működtető programok logikai felépítése	16
5.4. A kísérleti eszköz vizsgálata.....	20
5.5. Eredmények	24
6. A tesztsorozat értékelése.....	26
7. Továbbfejlesztés.....	27
7.1. A megfelelő hanggenerátor kiválasztása	27
7.2. A választott Hanggenerátor bemutatása, működése	30
7.3. Eredmények	32
8. Testreszabhatóság.....	34
8.1. Triggerelési módok.....	34
8.2. Sokoldalúság.....	35
9. Ergonómia, hordhatóság.....	36
9.1. Rapid prototyping	36
9.2. NYÁK.....	38

10. Az elkészült kísérleti eszköz tesztelése	39
10.1. Folytonos üzem:	39
10.2. Manuális üzem:	40
10.3. A tesztek kiértékelése, továbbfejlesztés aspektusai	42
11. Gazdasági számítás.....	43
12. Összefoglalás	45
13. Summary	46
14. Irodalomjegyzék.....	47
15. Köszönetnyilvánítás.....	49
16. Mellékletek	50
1. melléklet.....	50
2. melléklet.....	52
3. melléklet.....	54
4. melléklet.....	61
5. melléklet.....	66
6. melléklet.....	67
7. melléklet.....	68
8. melléklet.....	69
9. melléklet:.....	81

1. Bevezetés

1.1. Célkritizés

Olyan módszert és eszközt kifejlesztteni, mely hanghullámok visszaverődésén alapul, és virtuális kép előállítását teszi lehetővé.

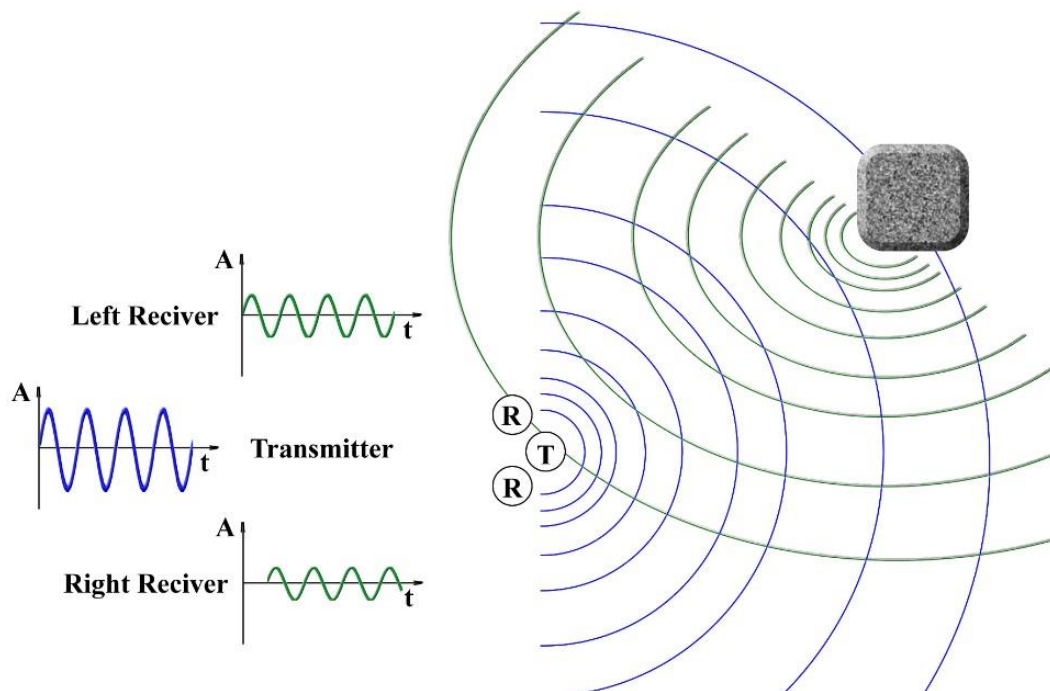
1.2. Témaválasztás

Az állatvilágban az emlősök egy része ultrahangos módszerrel tájékozódik, illetve léteznek olyan ősi „éjjellátó” technikák, melyeket jóval az elektronika megjelenése előtt is használtak bizonyos népek és azok harcosai. Ezek a technikák sok esetben a visszaverődött hangra épülnek. A denevér köztudott módon ultrahangot használ a zsákmányállatok felderítésére, illetve a tájékozódásra. L. Dijkgraaf (1949) és Matsuda et al. (2013). Ezt a jelenséget „echolocation”-nak nevezik, mely szabad fordításban annyit tesz, visszhangos helyzetmeghatározás. A denevérek ilyen módon való tájékozódását Donald Griffin és Robert Galambos állapította meg 1938-ban. L. Griffin (1958). Kézenfekvőnek tűnt tehát, ha az állatok (l. Jones. 2005), sőt az emberek is tudták használni ezt a technikát a sötétben való tájékozódásra, használjuk ugyanezt a módszert –az elektronika segítségével némiképp kiterjesztve–, virtuális képalkotásra. A készüléket kiegészítve egy hangszintetizátorral egy vakok és gyengén látók tájékozódását segítő eszközt kaphatunk. L. Mihajlik et al. (2004).

2. „Echolocation”

2.1. Működési elv

Ha a műszaki gyakorlathoz hasonlítanánk az echolocation technikáját, a legkézenfekvőbb analógia az aktív szonár lehetne. A SONAR angol mozaikszó, jelentése: SOund Navigation And Ranging. Működése a következő: Jones (2005) szerint az állat vagy –jelen esetben– a távolságmérő berendezés hangimpulzust bocsát ki, mely visszaverődik a felületekről, így visszajut a kibocsátóhoz, mely a jel intenzitásának csökkenéséből, illetve a kibocsátás és visszaérkezés közt eltelt időből számítja a távolságot, valamint a két fülbe vagy érzékelőbe jutó hangerősségből és időbeni eltérésből a vízszintes irányszöget. Az állatoktól elcsúszott tájékozódási módszert először a Titanic katasztrófája (1912) után egy angol meteorológus, Lewis Richardson szabadalmaztatta (1913). Hill (1962)



1. ábra: Az "Echolocation" működése

Suga et al. (1975) szerint a kibocsátott hangot különféle szempontokból vizsgálhatjuk.

- **Frekvencia:**

- **Frekvenciamoduláció (FM):** A frekvencia modulált keresőhang sweep, azaz söprő módon működik, vagyis FM-pingelésnél széles tartományban a magasabb frekvenciától lefelé söpri végig a frekvenciatartományt.
- **Állandó frekvencia (CF):** A keresőhang frekvenciája állandó. A CF-keresésnél a frekvenciasáv igen keskeny, így a keresőjel frekvenciája konstansnak tekinthető.

A denevérek jellemzően a két módszert vegyesen alkalmazzák. L. Stuart et al. (2000).

- **Intenzitás:** Az állatvilágban a keresőhang intenzitása jellemzően 60-140 decibel között mozog. A keresőhang amplitúdóját kis távolságok esetén csökkentik, mivel a visszaverődött hang hangnyomása károsíthatná a denevér fülét.
- **Impulzus időtartama:** A denevérek a keresőhang intervallumát 0,2-100 ms-ig terjedő skálán adják le, attól függően, hogy mekkora távolságot mérnek vele, illetve vadászat közben sűrítik a pingelést, így jobb felbontású képet képesek kapni a környezetükről. Rövidebb impulzushosszt kisebb távolságok esetén alkalmaznak.

2.2. FM-sweep

A széles frekvenciatartományon végigsöprő keresőhang alkalmazásának legnagyobb előnye, hogy igen nagy precizitással lehet vele távolságot mérni. J. A. Simmons (1975) kísérletekkel bizonyította, hogy a denevérek, ha ezt a módszert alkalmazták, képesek voltak fél milliméteres távolságkülönbséget is érzékelni két felület közt. Ez a képességük abból fakad, hogy a leadott hang egy frekvenciatartományon söpör végig, ami elősegíti a precízebb távolságmérést, mivel jobb felbontásban lehet érzékelni a leadott és visszaverődött hang közti

időbeni eltérést (ezáltal javítva a keresztkorrelációt a kettő közt). A pontosságot tovább javíthatja, ha a keresőhang harmonikusokat is tartalmaz. A hátránya az FM-metódusnak Suga et al. (1975), hogy csak kis távolságokban működik hatékonyan.

2.3. CF

A CF legnagyobb előnyének számít a nagyobb hatótávolság, illetve a Doppler effektusból származó sebességérzékelési képesség is. Mivel a denevér mozgó zsákmányállatokat keres, a Doppler effektusból származó vörös-, illetve kékeltolódás segítségével képes érzékelni

$$f = \left(\frac{c + v_r}{c + v_s} \right) f_0 \quad \text{annak mozgási irányát. Suga et al. (1975)}$$

Ahol:

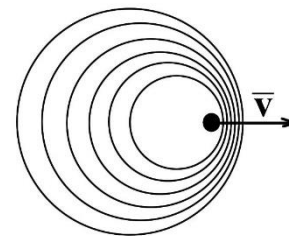
f : a visszaverődött hang frekvenciája

c : a hang sebessége

v_r : a denevér sebessége a közeghez képest

v_s : a visszaverő sebessége a közeghez képest

f_0 : a kibocsátott hang frekvenciája



2. ábra: A Doppler eltolódás

2.4. Idegi mechanizmusok az agyban

Mivel a denevérek a hallásuk segítségével tájékozódnak, így a fülük és az agyuk hallásért felelős része is speciálisan módosult. Az ember alapvetően látás alapján tájékozódik. A tapasztalat azt bizonyítja, hogy a megvakult emberek hallásért felelős agyszekciója, a halántéklebeny jelentősen jobb vérellátottságú a látó emberekénél, sőt idővel a látásért felelős nyakszirti lebeny nem elsorvad, hanem segít a hallott információk feldolgozásában. Burton (2003)

3. Ultrahangos távolságmérő rendszerek áttekintése

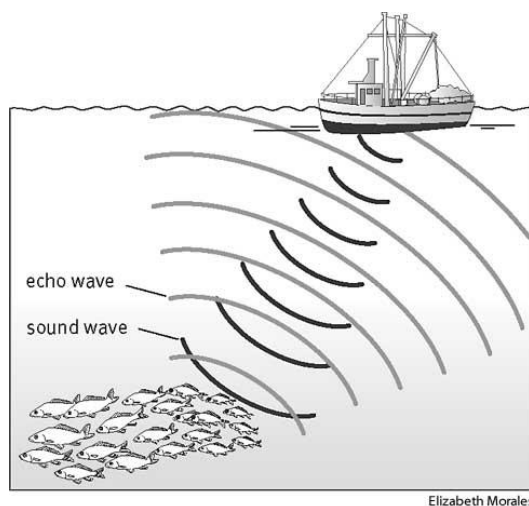
3.1. Ultrahangos távolságmérő berendezések napjainkban

A műszaki gyakorlatban előszeretettel alkalmazunk ultrahangos távolságmérést, mivel olcsó, és sok esetben elégséges pontossággal működik. Leggyakrabban az autóiparban és a robotikában használják ezt a típusú távolságmérést, de a halászatban, a harcászatban is előszeretettel alkalmazzák víz alatti tárgyak, élőlények felderítésére (SONAR). Mindennapi



3. ábra: Tolatóradar (Forrás: Nissan)

életünk során sokszor találkozhatunk a tolatóradarokkal, nagyobb gépjárművek esetén



Elizabeth Morales

4. ábra: Sonar működése (Forrás: YourDictionary)

parkolóradarokkal, melyek szintén ultrahangos távolságmérés alapú megoldások. Sok esetben bizonyos biztonságtechnikai berendezések, például riasztók is alkalmazzák ezt a típusú mérést nyílászárók nyitásának érzékelésére, üvegfelületek épségének vizsgálatára, és szintén az autóiparból kiragadott példa a gépjárművekben a mozgásérzékelő rendszer, mely az utastérben történő mozgást figyeli.

Ilyen megoldással lehet például egy lakásban a mozgást olyan módon regisztrálni, hogy nem

csak a mozgás meglétét, de a mozgást végző magasságát, valamint helyzetét is figyelemmel lehessen kísérni. L. Siddiquie (2009). Ultrahangos távolságmérést alkalmaznak továbbá a gyógyászatban is, tumorok, elváltozások felderítésére, de magzatvizsgálatra is. L. Arany et al. 2008. Nem csak ilyen területeken, de a gyártásban is használják az ultrahangot repedésvizsgálatra.

3.2. Ultrahangjeladók és -vevők bemutatása

Az ultrahangjeladók és -érzékelők igen hasonló felépítésűek. Ennek köszönhetően, sok esetben nem két külön eszközt alkalmaznak távolságmérésre, hanem ugyanazt a készüléket előbb hangimpulzus kibocsátására, majd a visszaverődött hang vételére használják (pl. tolatóradarok). Ezzel egy érzékelőt megspórolnak, a pontosság és a legkisebb mérhető távolság rovására. Ha viszonylag kis távolságokat mérünk, szükséges két eszközt: egy adót és egy vevőt használni. Egyik ezek közül csak hangforrásként üzemel, míg a másik csak vevőként működik. A hang sebessége ismeretében az adás és vétel között eltelt időből tudjuk számítani a hangot visszaverő objektum távolságát.

$$d = \frac{c}{2} \cdot t$$

ahol:

- d: a mért távolság [m]
- c: a közegben jellemző hangsebesség (levegő esetén $\approx 340 \frac{m}{s}$)
- t: a hangimpulzus adása és vétele közt eltelt idő [s]

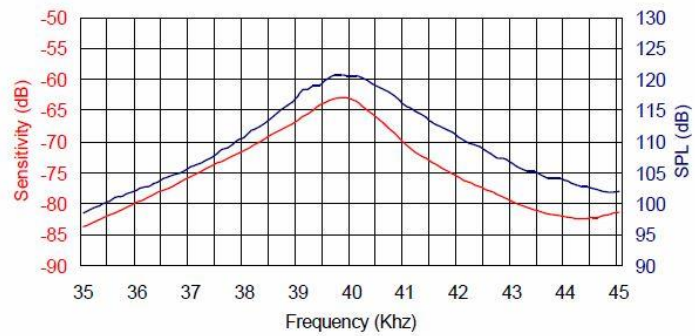
A kísérleti eszközben két darab intelligens szenzor került felhasználásra, melyek a trigger jel hatására 8 darab négyszögjelet küldenek 40 kHz-s frekvencián a transmitternek, minek hatására az 40 kHz-es 200 μ s hosszúságú ultrahang jelet bocsát ki megközelítőleg 120 dB hangnyomással. Ez a jel (Ultrasonic burst) az útjában lévő



5. ábra: Felhasznált intelligens távolságmérő szenzor

objektumról visszaverődik, így a vevőegységbe jut, ahol elektromos jellé alakul vissza. A küldés és visszaérkezés időbeni eltérését egy impulzushossz reprezentálja a készülék Echo kimenetén. A kísérleti eszköz eszköz az Echo időtartamát 100 μ s pontossággal méri, amely elégséges, hisz ez távolságban 17 mm-t jelent. Mind a transmitter, mind a receiver piezoelektromos készülék. Az alkatrészek karakterisztikái a 6. ábrán találhatóak.

Látható, hogy mindkét készülék egy bizonyos (éppen 40 kHz környéki) tartományban a leghatékonyabb, illetve a legérzékenyebb. Ez egyéb hangkeltőknél probléma, de jelen esetben igen hasznos, hisz így jó eséllyel más hangforrás nem zavarja a mérést. Ezen készülékkel elégséges pontossággal és megbízhatósággal 17 – 4000 mm-ig lehet mérni, ennél nagyobb távolságoknál nagyobb adóteljesítmény szükséges, és figyelembe kell venni, hogy a mért távolság növekedésével csökken a felbontás is, mivel az eszköz 15°-os kúpszögben mér.



6. ábra: A felhasznált szenzor transmitterének és receiverének átviteli görbéi (Forrás: 400ST/R120 adatlap)

A teszt sorozat eredményeként megállapítottam, hogy az ultrahangos szenzor csak kisebb távolságon belül, egyméteres tartományban képes viszonylag stabil és zajmentes adatszolgáltatásra. Ennek megfelelően módosítottam a mikrokontroller programját, mely így csak az egy méteren belül érkezett jeleket veszi figyelembe, viszont azt már 0,5 cm pontossággal. A további pontosságjavítás és hatótáv-kiterjesztés érdekében szükségessé válik egy szofisztikáltabb szenzor beszerzése.

3.3. Mikrokontroller, mint a beágyazott rendszer alapja

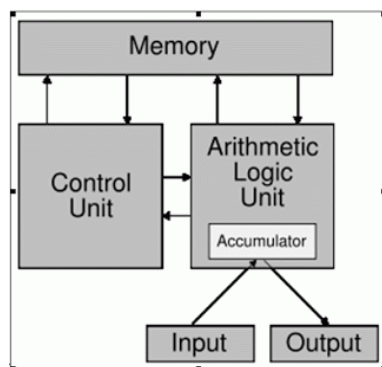
A mikrokontroller egyetlen lapkára integrált, általában vezérlési feladatokra optimalizált számítógép. L. Augarten et al. (1983). A szilíciumlapkán megtalálható a mikrokontroller processzora, a perifériák és a mikrovezérlő memóriája is. Régebbi típusoknál előfordul, hogy a memóriát nem integrálták a tokba, de a modern eszközökben ez szinte mindig megtalálható. A mikrokontrollerek beépített perifériái igen sokrétűek lehetnek, kezdve a hagyományos TTL vagy CMOS logikai szintű ki- és bemenetektől, az analóg-digitális, digitális-analóg konvertereken (ADC, DAC) és komparátorokon keresztül, egészen az analóg LCD kimenetekig és kommunikációs portokig (I2C, RS232, CAN, USB, Ethernet stb.). Allen et al. (2010)

Általában a mikrokontrollereket eltérő perifériákkal szerelik, így fontos a vásárlás előtt tájékozódni a kiszemelt mikrokontroller perifériáiról, hogy megfelelnek-e az adott irányítási feladat megoldásához (ezt leggyakrabban az adatlapok segítségével tehetjük meg). Fontos tulajdonsága még a mikrokontrollereknek, hogy hány bites felbontással képesek dolgozni, az ALU-juk tudja-e kezelni a lebegőpontos számításokat, valamint mekkora utasításkészlet áll rendelkezésre az adott mikrovezérlőn. Allen et al. (2002)

Harvard-architektúra, a mikrokontrollerek szokásos architektúrája

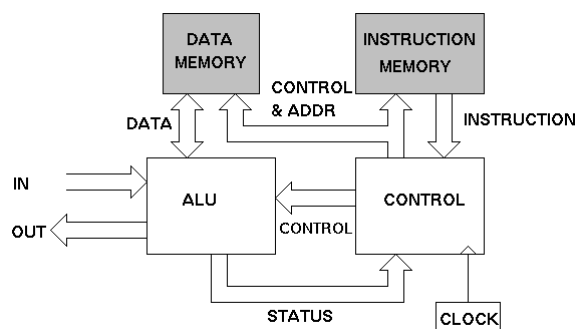
A modern számítógépek és mikrovezérlők architektúrája alapvetően különbözik egymástól, hisz míg az asztali számítógépünk Neumann-architektúrát használ, addig a mikrokontrollerek a Harvard-architektúrán alapszanak. A kettő közti lényeges különbséget először is a memóriánál találjuk meg, hisz míg a Neumann-féle megoldásnál nincs külön adat- és programmemória, addig ezt a Harvard-architektúrájú gépeknél külön találjuk.

A másik feltűnő különbség, hogy míg a Neumann-architektúrájú szerkezetekben egyetlen busz található (melyet ütemezetten vagy adattanszferre, vagy utasításátvitelre használhatunk), addig a Harvard-architektúrájú készülékekben az utasítás- és adatbuszok elkülönülnek, így egyidejűleg, azaz párhuzamosan folyhat a kommunikáció. Ez azt jelenti, hogy, bár a mikrokontrollerek felépítése bonyolultabb, mint egy általános célú processzoré, mégis gyorsabb műveletvégzést tesz lehetővé - köszönhetően a párhuzamos adat- és utasításkommunikációnak. Allen et al. (2002)



7. ábra: Neumann-architektúra

(Forrás: Embedded World)



8. ábra: Harvard-architektúra

(Forrás: Wikipedia)

4. Készülék tervezése

4.1. Előzetes megfontolások

A készülék tervezésének előkészítési fázisában figyelembe vettem az emberi agy adottságait, azt, hogy különbözik a speciálisan ilyen tájékozódásra szakosodott fajokétól. Tanulmányoztam az ősi, jól működő „éjjellátó” módszert, mely az ember által keltett hang visszaverődésén alapszik.

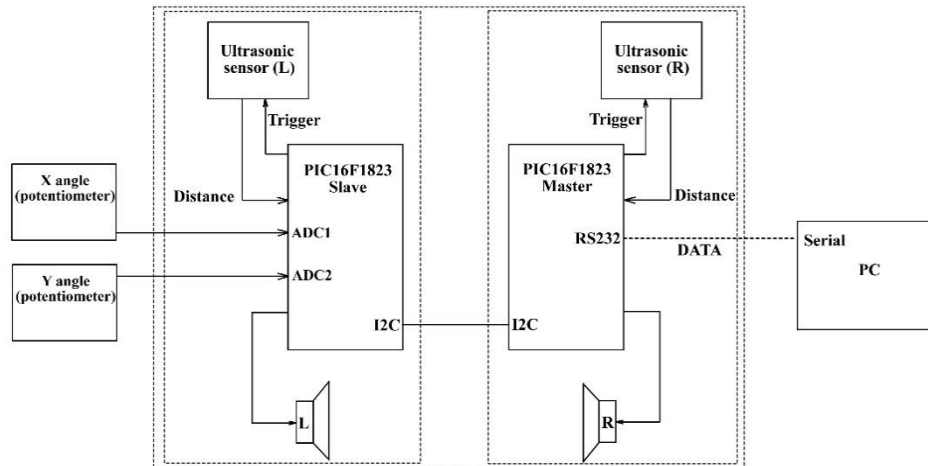
A koncepció felállításánál figyelembevett szempontok:

- hordozhatóság: Kis méret kis súly.
- akkumulátor-élettartam: Kisméretű áramforrás mellett hosszú működési idő.
- térhallás kihasználása: Az ember alapvető adottságainak felhasználása.

Ezen szempontok alapján egy állandó hullámhosszú távolságmérő eljárást választottam (CF).

4.2. Tervezési irányvonal

Két, egymástól 200 mm-re elhelyezkedő intelligens, ultrahangos távolságjeladó szenzort egy-egy mikrokontrollerrel a távolságmérésre utasítok (trigger), minek hatására a távolságmérő szenzor kimenetén egy –a távolsággal arányos hosszúságú– impulzus jelenik meg. Ennek hossza és a szenzor tulajdonságai ismeretében meghatározható a hangforrás és az objektum közti távolság.



9. ábra: A rendszer felépítésének vázlata

A két fülbe jutó hang erőssége alapján határozhatja meg az agy az előtte lévő akadály térbeni helyzetét, valamint annak távolságát, gyakorlatilag úgy, mintha a hangot maga az akadály adná ki.

A két mikrokontroller egyszerre adja ki a mérésre való utasítást, így a fülbe jutó hangnak nem csak az amplitúdója, de a frekvenciaeltolódása is értékelhető lesz.

Az intelligens távolságmérő szenzorok kimenetén megjelenő távolságértékeket mikrokontrollerrel feldolgozva hallható hangot ad a fülhallgatóba, mely alapján akár az emberi agy is felépítheti a háromdimenziós tér virtuális „képét”.

4.3. Alkalmas eszközök kiválasztása

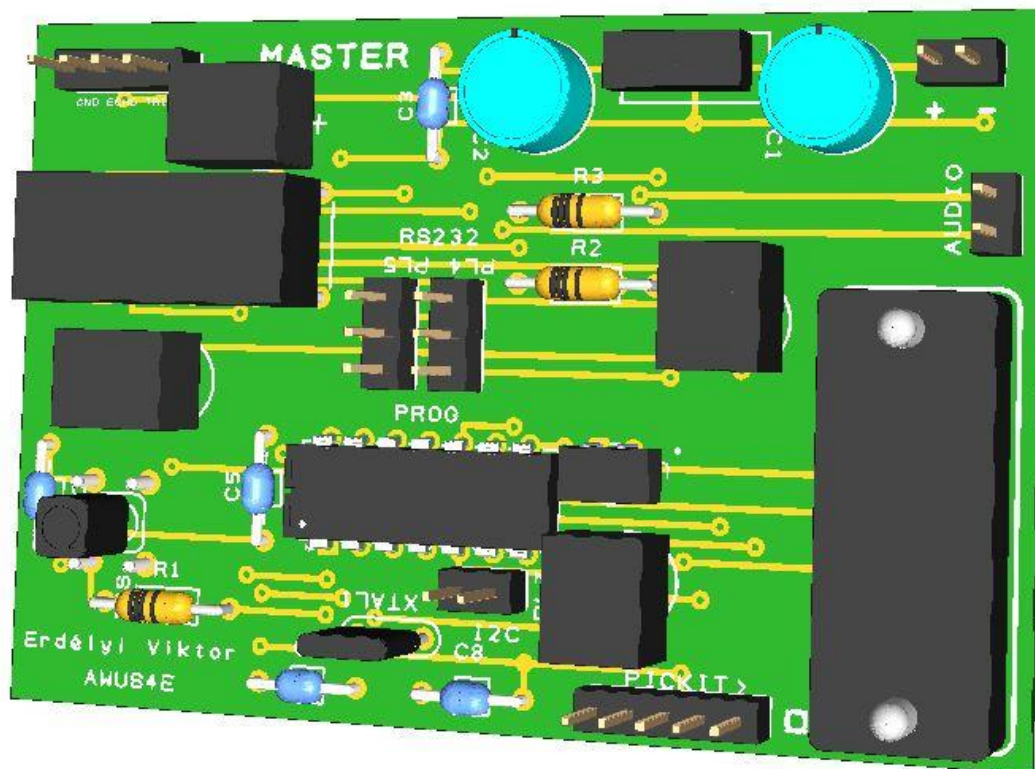
Távolságmérésre a kereskedelemben kapható intelligens távolságmérő szenzorokat választottam, melyek a mérést elvégezve a kimenetükön a távolsággal arányos hosszúságú jelet adnak. Ennek feldolgozására két darab PIC16F1823 típusú mikrokontrollert választottam, mely rendelkezik PWM modullal (a hangerő beállításához), I2C, SPI, kommunikációval (az egymással történő kommunikáció érdekében), USART porttal, mellyel képes számítógéppel kommunikálni esetleges mérési adatgyűjtés céljából, illetve elég gyors a μ s-nagyságrendű jelek feldolgozásához (max. 8 MIPS @ 32 MHz). A hallható tartományba eső hangot egy kereskedelemben kapható sztereo fejhallgató szolgáltatja, melyet egy normál tranzistoros végfokkal hajtok meg. A hangot szintén a mikrokontroller generálja a beérkezett jellel azonos időben, így a két hang frekvenciája bizonyos eltolásba kerül egymással.

4.4. Tervezési szakasz

Az áramkörök megtervezése:

Az áramkört a DesignSpark nevű programmal terveztem. A Kísérleti eszközt furatszerelt (DIP) alkatrészekből készülök összeállítani, ami nagyobb helyet foglal, viszont könnyebb a szerelhetősége. Ha a készülék beváltja a hozzá fűzött reményeimet, elkészítem ugyanezen kapcsolás SMD kivitelét is, így a készülék várhatóan kisebb helyen elfér majd. Ez fontos szempont a hordozhatóságot tekintve.

A mérések láthatóvá tételét elősegítendő, az áramkör kapott egy kommunikációs csatornát is, mely segítségével az általa mért adatokat egy számítógépen MATLAB segítségével meg lehet jeleníteni. Ez a kommunikáció egy RS232 soros adatátviteli megoldás segítségével valósul meg. A szükséges programokat C, illetve C# nyelveken írtam meg. (Ezek részletesen a mellékletekben találhatók.)

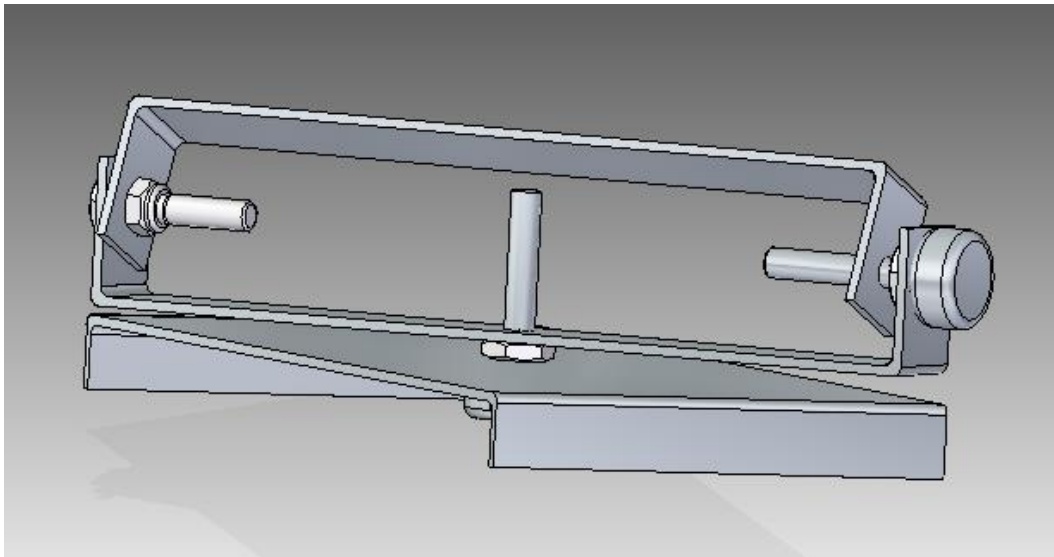


10. ábra: A prototípus-nyák 3D-s modellje

5. A kísérleti eszköz megépítése és vizsgálata

5.1. A tesztelő - megjelenítő eszköz tervezése

Ahhoz, hogy a rendszer működését a látó emberek számára is láthatóvá tegyük, illetve az értékelést elősegítsük, egy megjelenítő eszköz tervezése is szükségessé vált. Ennek segítségével a teret virtuálisan le lehet képezni. A rendszer továbbfejlesztéséhez ez hasznos információkkal szolgál.



11. ábra: Két szabadságfokú szkennertartó állvány szögmérővel

A háromirányú információkat megjelenítő eszköz működése:

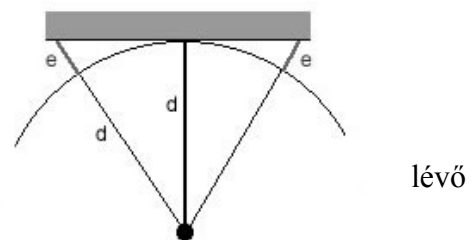
- függőleges és vízszintes irányszög mérése
- kétoldali távolságadat mérése
- adatok küldése PC-re (soros porton keresztül)
- adatok feldolgozása, megjelenítése MATLAB szoftver segítségével.

Az adatgyűjtést nem külön eszköz, hanem maga a fejlesztett berendezés végzi. A vízszintes és függőleges irányszögeket egy-egy potenciométer segítségével méri a slave mikrokontroller. Ezt a két szögállást a potenciométerekből kialakított feszültségosztó kimenetén megjelenő feszültség mérésevel végzi a rendszer. A feszültségmérés 10 bites analóg-digitális konverter segítségével valósul meg, melynek a 8 szignifikáns bitje egy-egy változóban kerül tárolásra.

A slave mikrokontroller ezen kívül méri és tárolja is a távolságot, mely szintén egy 8 bites változóban tárolódik. A master mikrokontroller kérésére a slave vezérlő I²C segítségével adatot szolgáltat, melyet a master mikrokontroller a benne tárolt egy távolságadattal együtt a számítógép felé továbbít. A számítógép fogadja a 4 bájt adatot, illetve tárolja is azokat. Így az eszközt kézzel mozgatva (szkennelve) az eltárolt adathalmazból MATLAB segítségével képet készítünk, még hozzá úgy, hogy a rögzített adatokból egy háromdimenziós felületet hozunk létre.

Gömbi torzulás kompenzálása szoftveres úton

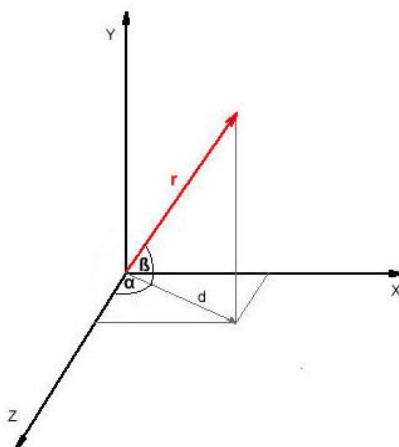
A szkennertartó állvány adottságai miatt a szkennelésnél szükséges figyelembe venni a gömbi torzulás jelenségét, mivel a mérőegységet tengelyek körül forgatjuk, így azonos távolságra pontokról (síkfelületről)



12. ábra: Gömbi torzulás oka

változó távolságadatokat kapnánk. Ha korrekció nélkül dolgoznánk fel a beérkezett adatokat egy tökéletes síkfelület, egy gömbsüveg képét adná vissza. Mivel a szkennerek gömbkoordináta-rendszerben működik, a torzulás elkerülése érdekében koordináta-transzformációt kell végrehajtani.

Az így kapott adatok már elhelyezhetőek Descartes-



13. ábra: Koordináta-transzformáció

$$d = \cos\beta \cdot r$$

$$x = \sin\alpha \cdot d$$

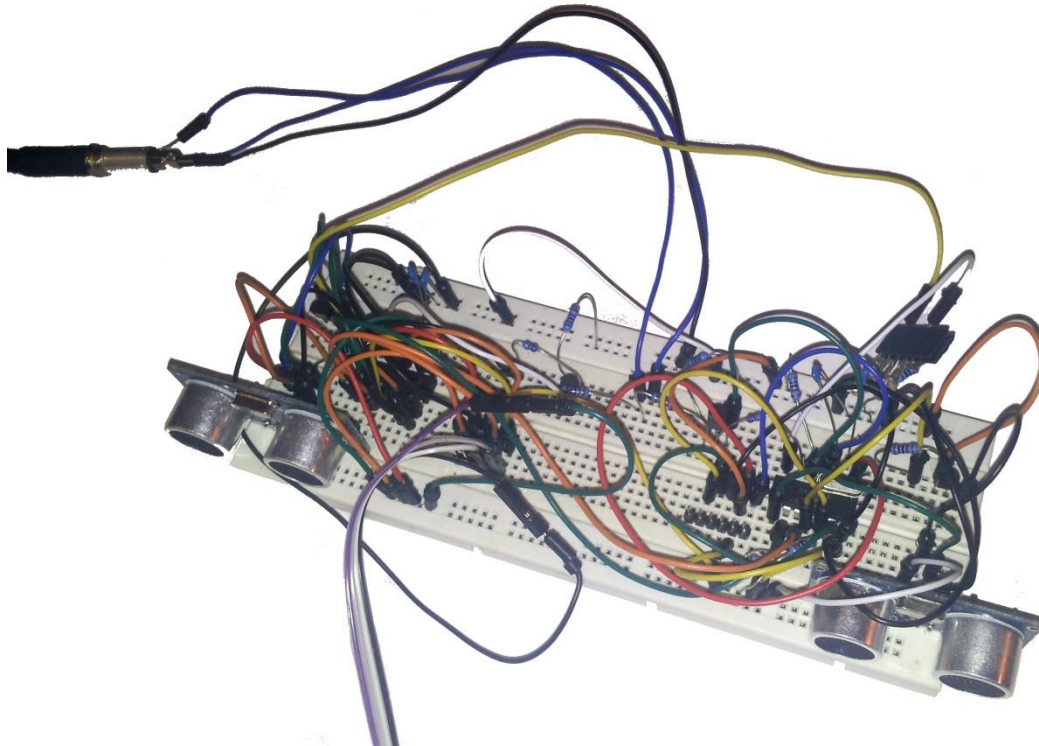
$$y = \sin\beta \cdot r$$

$$z = \cos\alpha \cdot d$$

féle koordinátarendszerben.

5.2. A távolságmérő áramkör megépítése

A kísérleti eszköz készüléket forrasztásmentes próbapanelen állítottam össze furatszerelt alkatrészekből. Az építés során az áramköri terveket néhol módosítottam. A tesztkészülék két nagyobb egységre bontható. Ezek a jobb és bal oldali szenzorok jeleinek feldolgozását, illetve a hangga alakítást végzik.



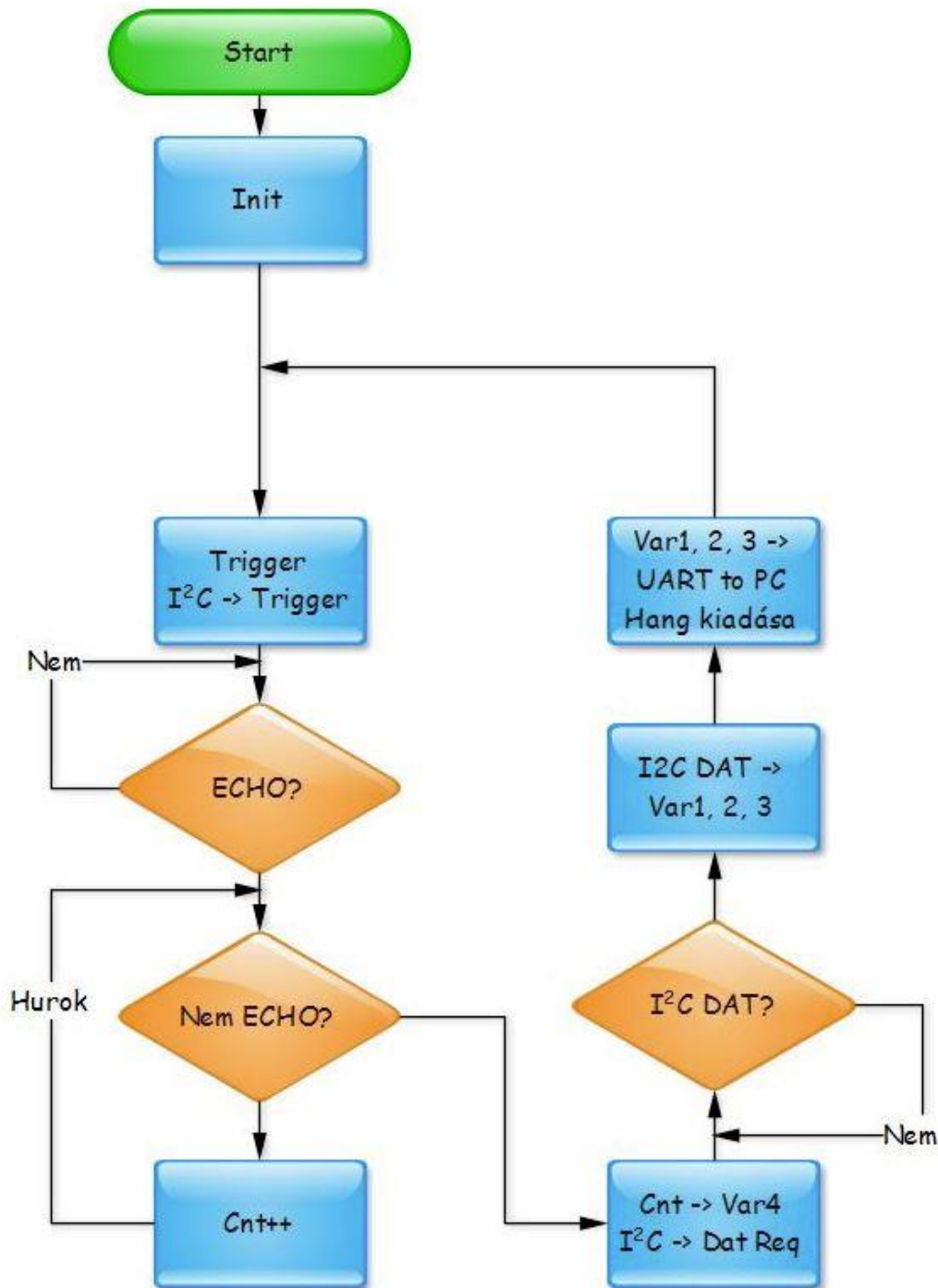
3. ábra: Kísérleti eszköz

A megépített eszköz lehetőséget nyújt arra, hogy az elképzelést (miszerint az ember is képes lehet hangok alapján tájékozódásra) vizsgáljuk.

5.3. A működtető programok logikai felépítése

A vezérlőprogramok az alábbi ábrán látható módon működnek.

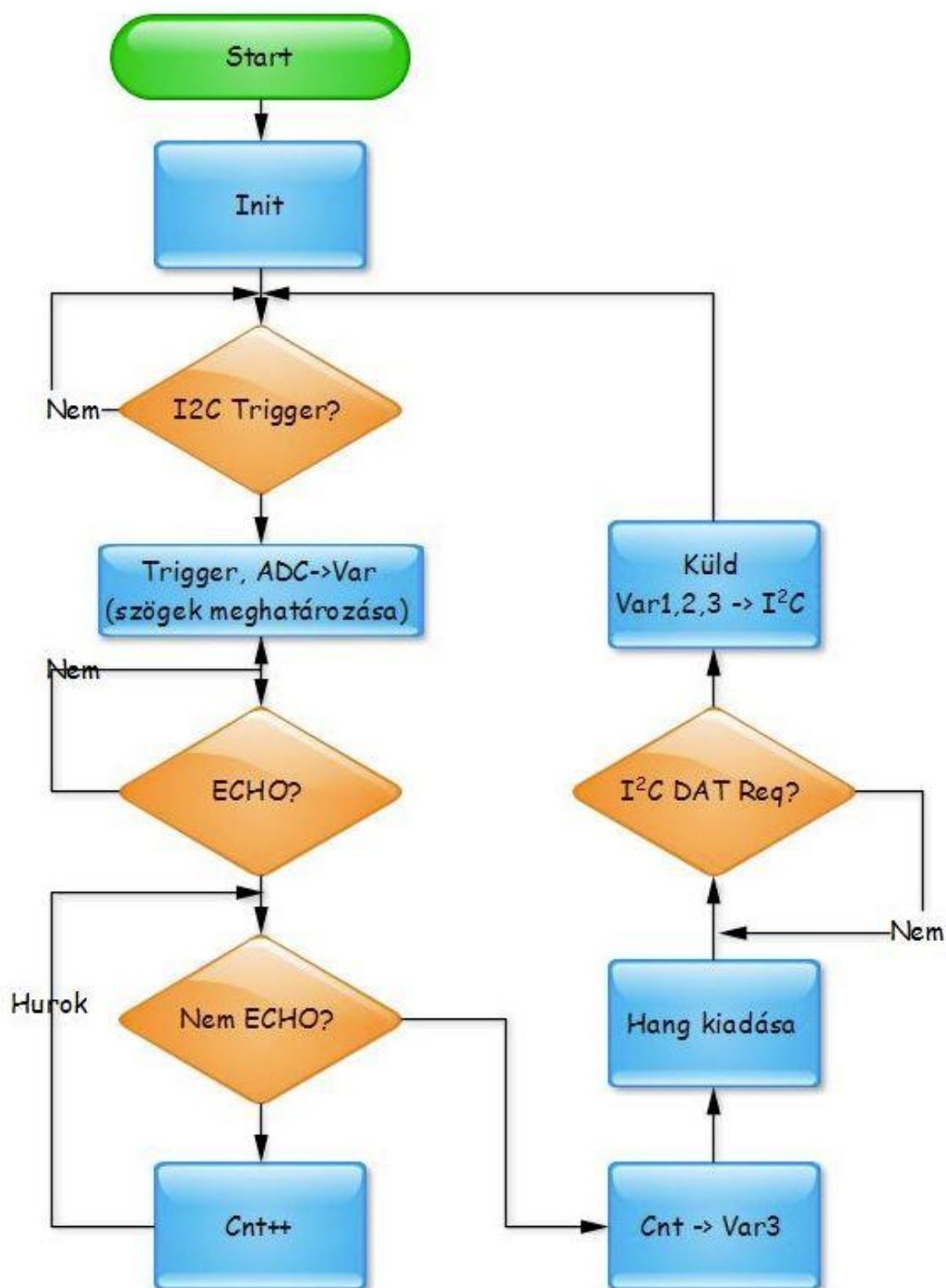
Program Master



4. ábra: Master vezérlő programjának logikai vázlat

A program elindulása után, egy hardver inicializáló ciklus fut le, mely beállítja a mikrokontroller portjait, konfigurálja a GPIO-kat, engedélyezi a megszakításokat illetve a megfelelő értékekkel feltölti a mikrokontroller option regiszterét. Ezután a mikrokontroller végtelen ciklusba kapcsol, így újra és újra elvégzi a szükséges feladatokat. A hurok első lépése, hogy 10 μ s hosszúságú trigger jelet ad az intelligens szenzor bemenetére, ezután I²C-n keresztül ugyanerre utasítja a slave mikrokontrollert is. A trigger jel elküldése után azonnal az echo jel megérkezését kezdi figyelni a mikrokontroller, egy alhurkot használva. Amint az echo jel megjelenik, a program egy számlálóciklusba ugrik, ahol 30 μ s –onként növeli a cnt számláló értékét, addig, amíg az echo jel fennáll. Ha az echo jel megszűnik a távolsággal arányos cnt értéként egy másik változóba (Var4) írja a program. Ezzel egyidőben a slave mikrokontroller felé I²C porton egy adatkérő üzenetet küld. Amint a szolga mikrokontroller adatot közöl, az adatot további három változóba írja, majd az ezekkel együtt 4 változót soros porton a számítógép felé továbbítja. Az információtovábbítás után a hangszintetizáló szubrutin fut le, melynek hatására megjelenik a távolsággal arányos erősségű jel a fejhallgatóban.

Program Slave



5. ábra: Slave vezérlő programjának logikai vázlata

A program elindulása után, egy hardver inicializáló ciklus fut le, mely beállítja a mikrokontroller portjait, konfigurálja a GPIO-kat, engedélyezi a megszakításokat illetve a megfelelő értékekkel feltölti a mikrokontroller option regiszterét. Ezután a mikrokontroller végtelen ciklusba kapcsol, így újra és újra elvégzi a szükséges feladatokat. A hurok első lépése az I²C-n érkező trigger utasítás figyelése. Ha ez megjelenik, 10 µs hosszúságú trigger jelet ad az intelligens szenzor bemenetére, illetve megméri a két szögadó potenciométer kimenetén lévő, a szögállással arányos feszültséget. Ezt a két szögadatot egy-egy változóban tárolja. Ezek után az echo jel megérkezését kezdi figyelni a mikrokontroller, egy alhurkot használva. Amint az echo jel megjelenik, a program egy számlálóciklusba ugrik, ahol 30 µs –onként növeli a cnt számláló értékét, addig, amíg az echo jel fennáll. Ha az echo jel megszűnik a távolsággal arányos cnt értéként egy másik változóba (Var3) írja a program. Ezután a hangszintetizáló szubrutin fut le, melynek hatására megjelenik a távolsággal arányos erősségű jel a fejhallgatóban. A következő lépésben a mikrokontroller figyeli, hogy érkezett-e adatkérés a master oldalról. Ha igen, a tárolt három változót (szögadatok, távolság) I²C-n elküldi a mester mikrokontrollernek.

5.4. A kísérleti eszköz vizsgálata

A kísérleti eszközt többféleképp vizsgáltam.

1. Hallás alapján tájékozódás

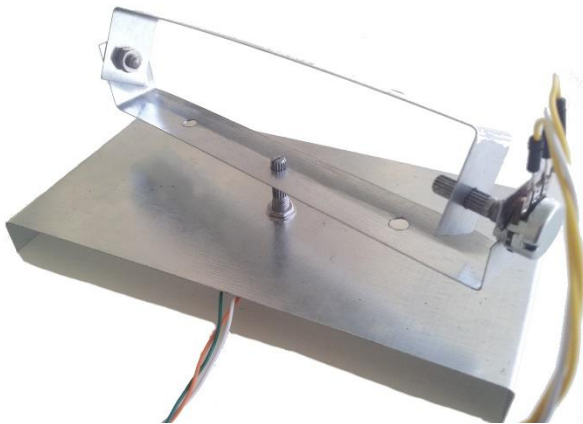
A készüléket egy fejhallgatóhoz erősítve először nyitott, majd csukott szemmel teszteltem. Idővel a látás nélkülözésével is kielégítő képet kaptam a körülöttem –különösen a közelemben– lévő tárgyak helyzetéről, illetve távolságáról. Ebből a tényből arra következtettem, hogy hosszabb tanulási procedúra nélkül is lehet az eszközt tájékozódásra használni.



67. ábra: Tájékozódás az eszköz segítségével

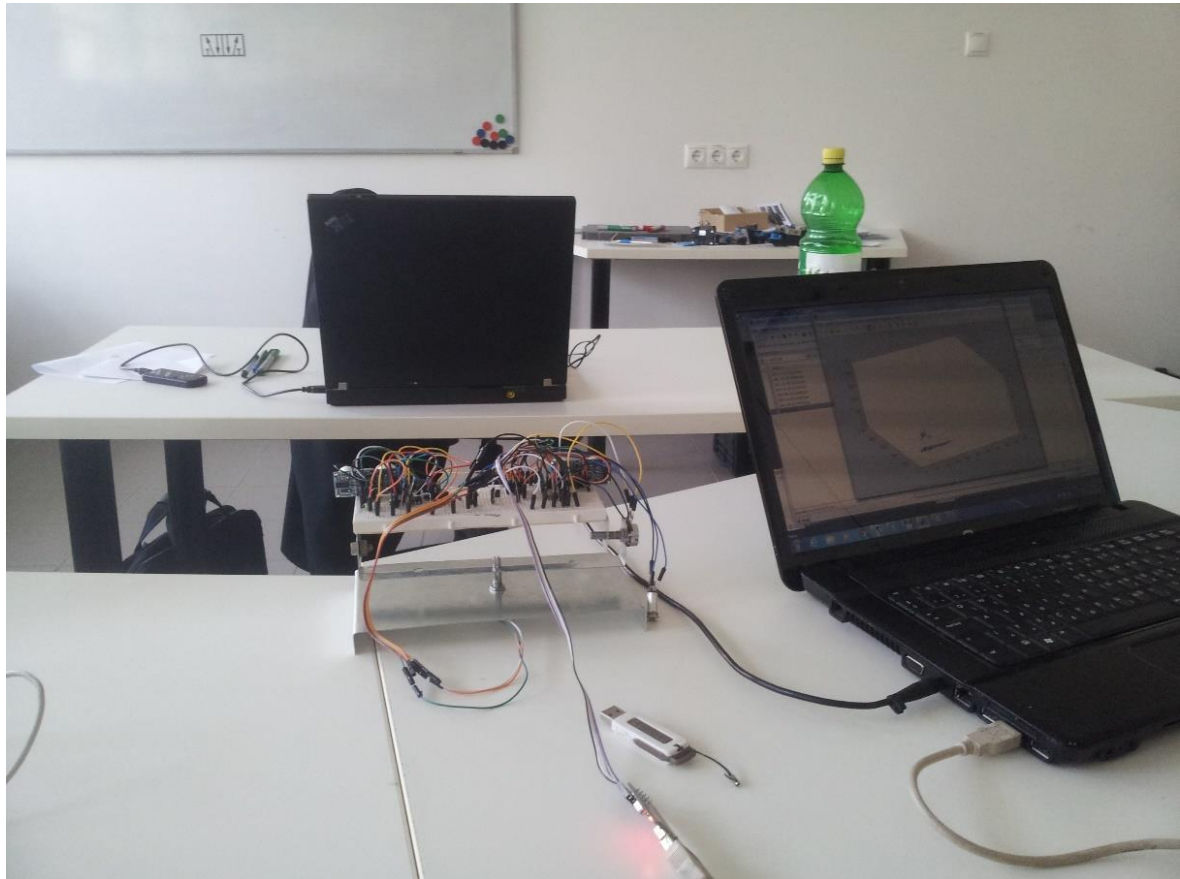
2. Tárgyak szkennelése, megjelenítése PC-n

Az eszközre kiegészítőleg felszereltem két potenciométert, melyekkel a vízszintes és függőleges irányszögeket méri a rendszer. Ezeket a szögeket, illetve a mért távolsági adatokat soros kommunikáció segítségével egy számítógépre küldi a készülék, ahol az adatok feldolgozásra kerülnek.



7. ábra: Két szabadságfokú mérőállvány szögadókkal

A feldolgozás eredményeként a MATLAB program egy képet jelenít meg, ahol az X és Y tengelyeken a koordinátatranszformáció következtében [mm] dimenziójú adatok jelennek meg, a Z tengely a mérőberendezés és az előtte lévő tárgy között lévő korrigált távolságadatokat reprezentálja. Ezáltal egy háromdimenziós pontthalmaz jelenik meg, így láthatóvá téve az elképzelés működését.



89. ábra: Kísérlet: Sík és hengeres felületek szkennelése

A további kutatások eredményeként várhatóan kifejlesztésre kerül egy OpenGL alapú környezet, mely képes lesz valós időben az adatok megjelenítésére, illetve a pontthalmazra való felületillesztésre is.

3. Teljesítményfelvétel

Mivel a rendszer tervezésénél kiemelkedően fontos szempont volt a hordozhatóság, vizsgálnunk kell az eszköz teljesítményfelvételét, hiszen ez nagyban befolyásolja az akkumulátoros üzem időtartamát. Távolságmérés közben végeztem az adatgyűjtést - fejhallgató és szögmérőrendszer csatlakoztatása mellett. A méréseket egy Maxwell MX-25 304 típusú multiméterrel hajtottam végre, a fejhallgató típusa: Panasonic RP-HT265 volt. Az eredmények az 1. táblázatban láthatóak.

1. táblázat: Teljesítményfelvétel távolság függvényében

#	Mért távolság [mm]	Áramfelvétel [mA]	Teljesítményfelvétel [mW]	Várható üzemidő 2000 mAh esetén [h]
1	50	27,5	137,5	72,73
2	100	27,6	138	72,46
3	150	27,5	137,5	72,73
4	200	27,5	137,5	72,73
5	250	27,4	137	72,99
6	300	27,5	137,5	72,73
7	350	26,4	132	75,76
8	400	27,3	136,5	73,26
9	450	27,4	137	72,99
10	500	27,4	137	72,99
11	550	27,3	136,5	73,26
12	600	27,3	136,5	73,26
13	650	27,3	136,5	73,26
14	700	27,2	136	73,53
15	750	27,1	135,5	73,80
16	800	27,2	136	73,53
17	850	27,1	135,5	73,80
18	900	27,1	135,5	73,80
19	950	27	135	74,07
20	1000	27	135	74,07
Átlag	525	27,255	136,275	73,39

A teljesítményfelvétel csökkenése a távolság függvényében azért következik be, mert a távolság növelésével a hangerőt csökkenti a rendszer, minek következtében a fejhallgató által felemésztett teljesítmény csökken.

4. A távolságmérés pontossága

A virtuális képalkotásnál elengedhetetlen fontosságú a mérés pontossága. Ezt megállapítandó a készüléket ismert távolságokra helyeztem el egy sík felülettől, majd számítógép segítségével a soros porton érkező jelet feldolgozva megállapítottam a készülék által mért távolságot. A mérési eredményeket a 2. táblázatban foglaltam össze.

2. táblázat: A távolságmérés pontossága, hibája

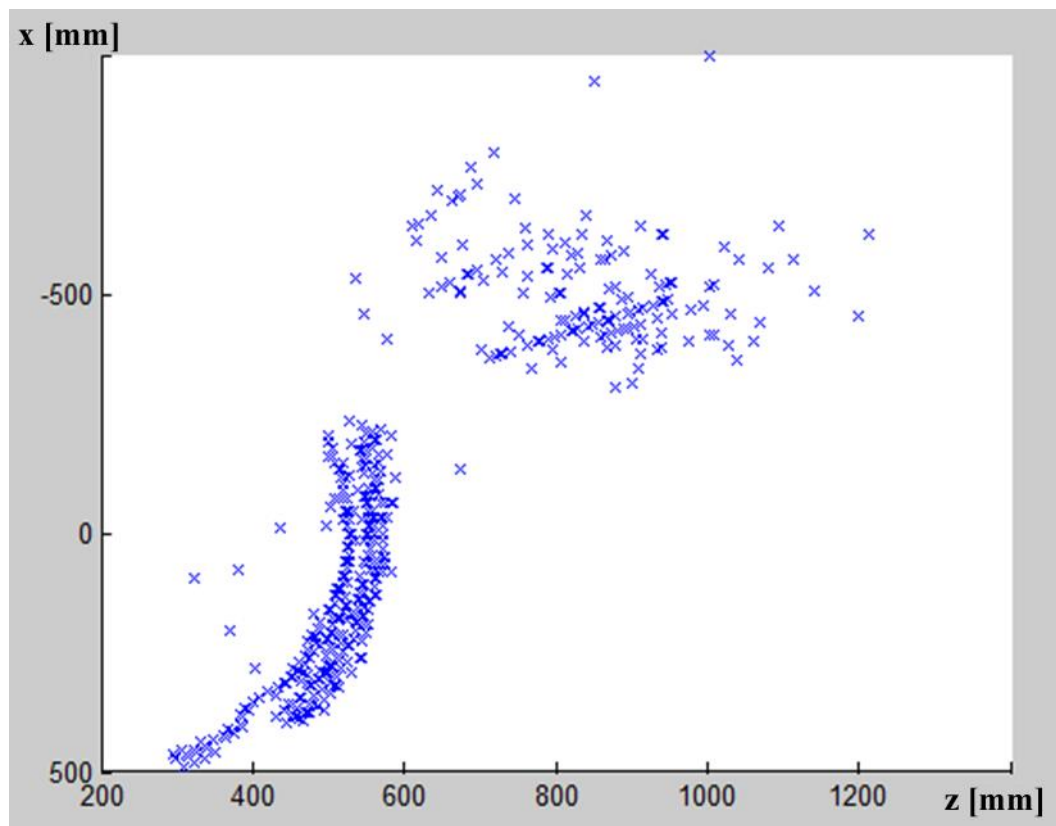
#	Ismert távolság [mm]	Mért távolság [mm]	Hiba [%]
1	50	55	10
2	100	100	0
3	150	155	3,333333
4	200	200	0
5	250	245	2
6	300	305	1,666667
7	350	345	1,428571
8	400	400	0
9	450	445	1,111111
10	500	490	2
11	550	545	0,909091
12	600	595	0,833333
13	650	640	1,538462
14	700	695	0,714286
15	750	740	1,333333
16	800	790	1,25
17	850	840	1,176471
18	900	890	1,111111
19	950	940	1,052632
20	1000	985	1,5
Átlag			1,64792

Az eredményekből látható, hogy az intelligens szenzor hibája a kisebb távolságoknál nagyobb, míg a mérési tartomány végén ez az érték kisebb, habár a mérési hiba nagyobb.

5.5. Eredmények

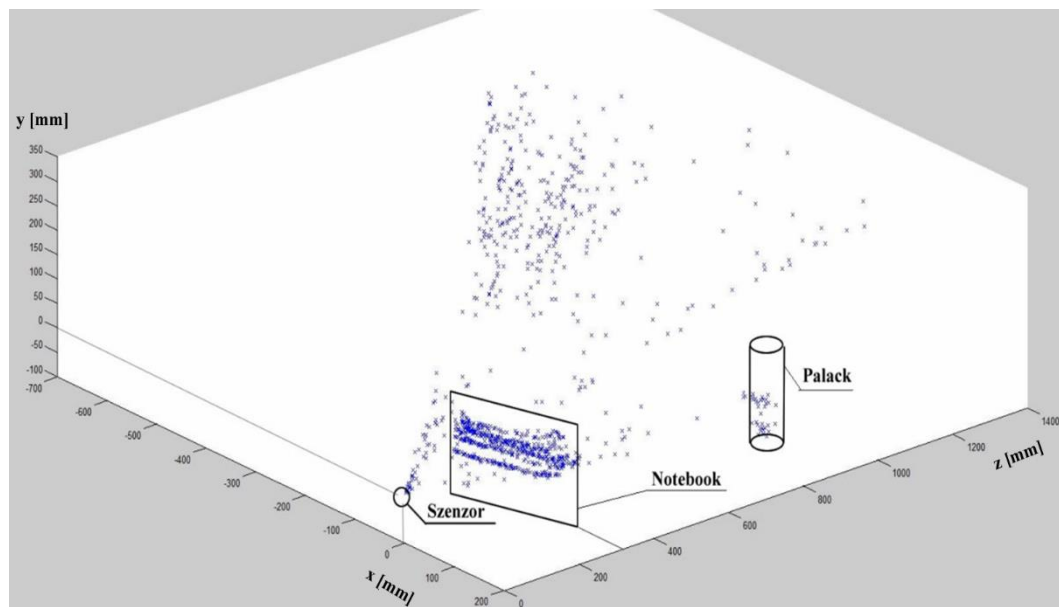
A kísérleti készülék tesztelése során arra a megállapításra jutottam, hogy az eszköz megfelel a vele szemben támasztott követelményeknek. Segítségével lehetőség nyílik virtuális képalkotásra, még hozzá költséghatékonyabban a jelenleg használatos rendszereknél (igaz kisebb pontossággal), illetve felhasználható a vakok és gyengén látók tájékozódásának segítésére.

A következőkben néhány, a készülékkel szkennelt kép látható:



20. ábra: A gömbi torzulás jelensége (sík mért felület, felülnézet)

A hibát megfelelően kompenzálva a notebook sík fedlapja és a hengeres palack képe:



21. ábra: Kézi szkennelés eredménye

A virtuális képeken felfedezhető szórt ponthalmazok, a szenzor gyenge minőségének következményei. Az 1m fölötti mért értékek igen zajosak.

6. A teszt sorozat értékelése

A kísérleti eszköz tesztelése fényt derített arra, hogy a kezdeti elképzelésem helyénvaló, az ember képes pusztán hangok alapján tájékozódni, illetve a rendszert lehetséges virtuális képalkotásra alkalmazni egy kiegészítő eszköz segítségével.

Mindkét téma magában is érdekes lehet, viszont jelen dolgozat további részében a tájékozódást segítő aspektusával foglalkozom.

A kísérleti eszköz fő hibája hangszintézis volt. A kísérleti eszköz egyszerű négyszögjel alkalmazott hangkeltésre, ráadásul ez a hang mindig egy bizonyos időtartamra (20 ms) került megjelenítésre. Ez egy hosszú távon kellemetlen hangot eredményezett. A hang kellemetlenségének fő okozója a négyszögjel használata mellett az volt, hogy a mikrokontroller nem képes párhuzamos feladatkezelésre, tehát nem képes egy időben távolságot mérni és hangot generálni. A megoldás első lépésként a megszakításos programozásban mutatkozott.

A megszakításos program olyan szempontból kielégítőnek tűnt, hogy a hang viszonylag folytonossá vált, viszont a négyszögjel kellemetlen hatása szűrőkondenzátor alkalmazása mellett is érezhető volt.

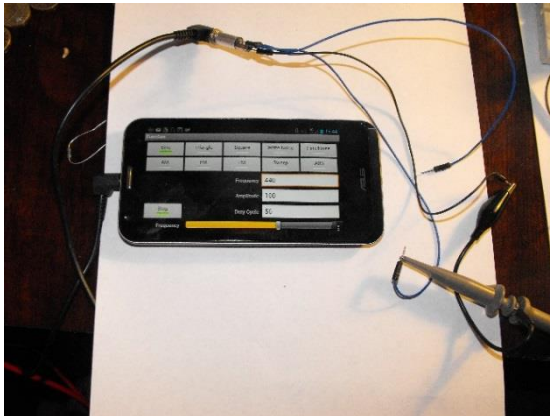
A megoldás így abban rejlett, hogy a távolság számítását, illetve a hangerő beállítását végző mikrokontroller mellé egy külön készüléket alkalmazzunk hanggenerálásra. Az elképzelés beváltotta a hozzá fűzött reményeket.

A következőkben a továbbfejlesztés részletei, illetve a továbbfejlesztett eszköz tesztelése és a vizsgálatok eredményei olvashatók.

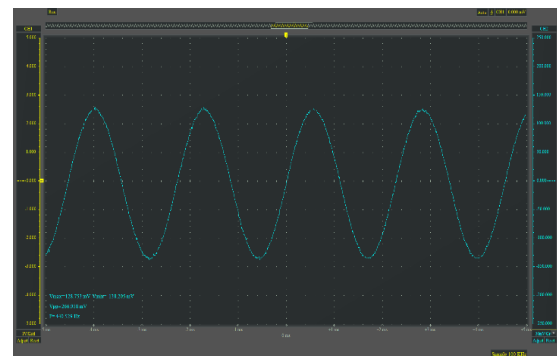
7. Továbbfejlesztés

7.1. A megfelelő hanggenerátor kiválasztása

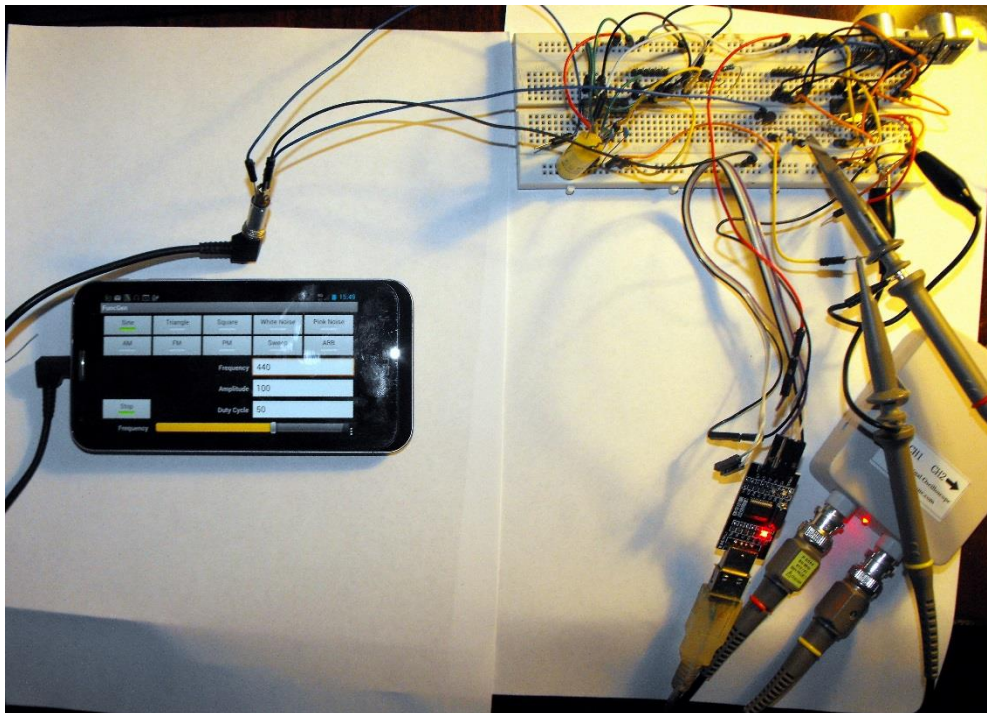
Az elképzelés helyességét, miszerint egy külső hanggenerátor jobb minőségű hangot, - így kényelmesebb használatot- biztosít viselője számára, úgy teszteltem, hogy okostelefonon futtatott funkciógenerátor program segítségével szinuszos jelet állítottam elő. -Ezen jel amplitúdójának változása reprezentálta a távolságot. Az amplitúdó beállítása ugyanúgy, mint az első készüléknél impulzus szélesség modulációval (PWM) történt.



22. ábra: Okostelefon, mint jelgenerátor



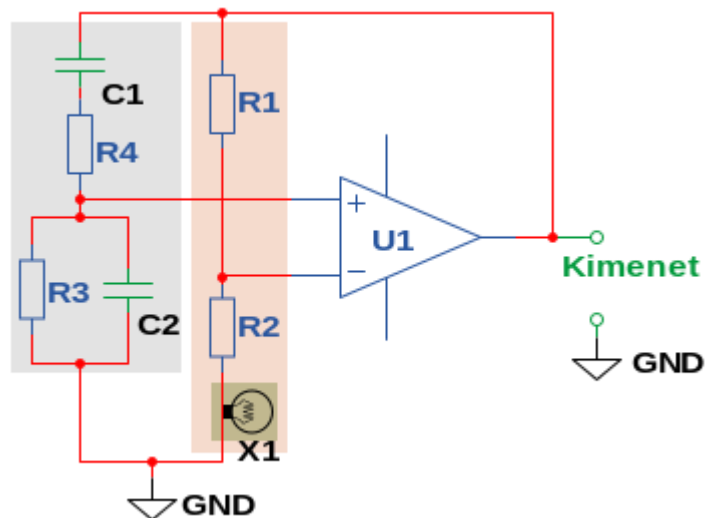
23. ábra: A mért jelalak



24. ábra: Az elképzelés tesztelése

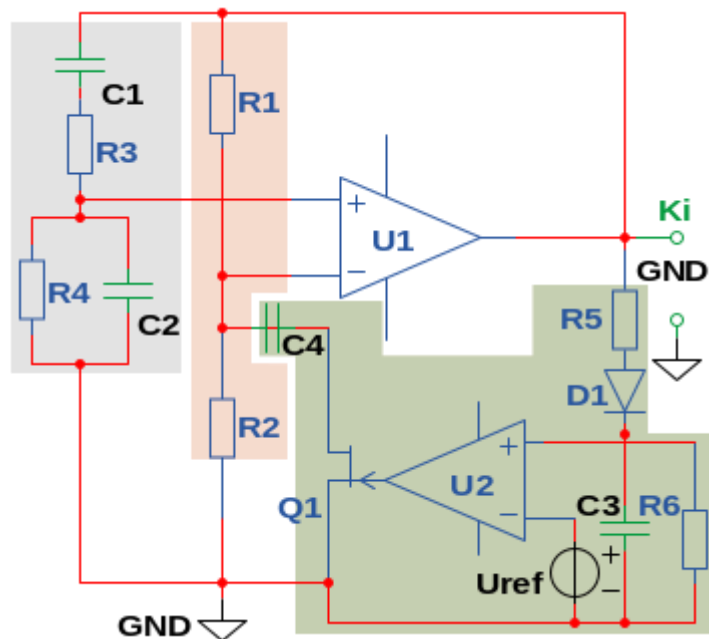
A teszt kielégítő eredménnyel járt. A szinuszos jel megfelelő hangminőségűnek bizonyult. Az impulzus szélesség modulációval megfelelően lehetett szabályozni a hangerőt, mely fordítottan arányos a mért távolsággal.

Az első és talán legkézenfekvőbb megoldás egy Wien generátor alkalmazása volt, viszont ezek ismertén instabil eszközök, legalábbis megfelelő visszacsatolás hiányában. A probléma abban áll, hogy a jel legszignifikánsabb tulajdonsága, az amplitúdó változhat drasztikusan a használat során. Amennyiben egyszerű „izzós” stabilizálást alkalmazunk, abban az esetben jelentősen csökkenne az eszköz hordozhatósága, illetve az izzó alkalmazása a megbízhatóságot is jelentősen befolyásolná.



25. ábra: Wien-hidas oszcillátor stabilizálása izzó segítségével
(forrás: Wikipedia)

Természetesen a Wien-híd stabilizálására létezik szofisztikáltabb megoldás, viszont ez főlegesen elbonyolítaná az áramkört.



26. ábra: Wien-hidas oszcillátor műveleti erősítős amplitúdó-stabilizálással (forrás: Wikipedia)

A fentieket mérlegelve elvettem a Wien-hidas oszcillátor megépítését, és kompaktabb megoldás felé fordultam.

Kereskedelmi forgalomban széles választékban lehet kifejezetten szignálgenerálásra alkalmazható IC-ket vásárolni. Ezek sok esetben képesek egy tokban többféle jel előállítására is. Ilyen a MAX038 típusú jelgenerátor. Ez a generátor kielégítené az alkalmazás által támasztott igényeket. Az integrált áramkör beszerzése viszont Magyarországon nehézkes, kizárólag külföldről lehet beszállíttatni. Éppen ezért ez a megoldás nem rentábilis, hisz a szállítási költséggel együtt a készülék ára 2500 Ft-ra rúgna.

Az ICL8038 típusú feszültségvezérelt oszcillátor szintén jó választás lehetne, viszont ennél a készüléknél igen sok külső diszkrét elemre van szükség a működéshez, ami a véges helykínálat miatt problémás, illetve a készülék ára sem elfogadható egy ilyen alkalmazás esetében, mivel darabja több, mint 1000 Ft-ba kerül.

Még egy lehetséges megoldás kínálkozott, mégpedig az AD5932. Ez egy digitálisan programozható hullámforma generátor, mely SPI-on keresztül kommunikálhat a mikrokontrollerrel. Ezen készüléket viszont minimum 100 db-os tételben kell rendelni, és így is jóval magasabb az ára, mint annak a megoldásnak, amit végül választottam.

A döntést segítő, egy táblázatot hoztam létre, melyben összevettem a kínálgzó lehetőségeket. A vizsgált eszközöket négy tulajdonságuk alapján hasonlítottam össze, ezekre egyenként max. 10-10 pontot kaphattak. A legmagasabb összpontszámot elért megoldást választottam.

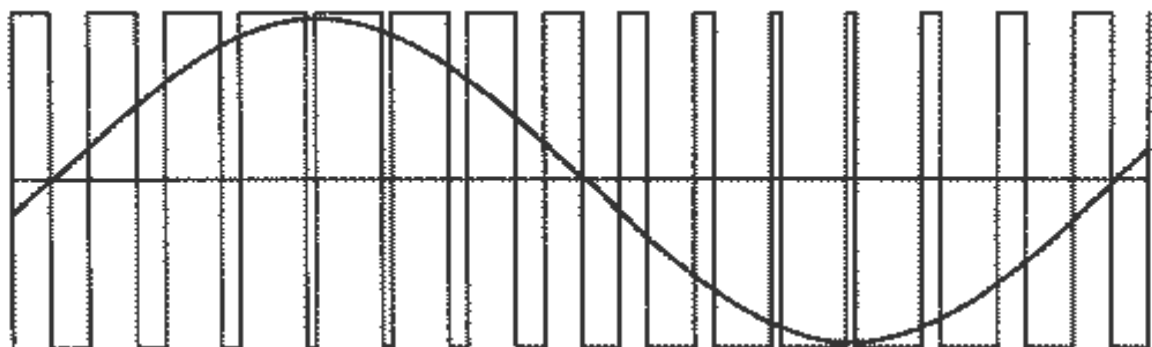
3. táblázat: Hanggenerátorok összehasonlítása: 0: legrosszabb, 10: legjobb

Eszköz	Megfelelőség	Elérhetőség	Ár	Egyszerűség	Össz
Wien-híd	0	10	10	9	29
Wien-híd izzó stab.	7	10	9	7	33
Wien-híd műveleti erősítős stab.	9	10	7	5	31
MAX038	9	5	2	10	26
ICL8038	9	8	4	10	31
AD5932	9	1	6	6	22
PIC16F1823	8	10	10	7	35

A legmegfelelőbb megoldásnak az mutatkozott, ha egy ugyanolyan mikrokontrollert alkalmazok a szinuszos jelforma előállításához, amelyet a távolságméréshez is. Meglepő módon, habár bonyolultabb eszközről beszélhetünk, mint az előbbiekben tárgyalt céleszközök, mégis jóval kevesebb anyagi ráfordítást jelent egy mikrokontrollert megfelelően felprogramozva alkalmazni. Ezzel az eszközzel a külső hanggenerálás gondja 280 Ft-ból megoldható lett.

7.2. A választott Hanggenerátor bemutatása, működése

A hanggenerálás módja a következő: a mikrokontroller a 400 Hz-es hangot PWM segítségével állítja elő (Keit H. 2005).



27. ábra: Szinus elállítás PWM segítségével (forrás: Power Electronics Design)

rad	A
0,5	0,479426
1,1	0,891207
1,7	0,991665
2,3	0,745705
2,9	0,239249
3,5	-0,35078
4,1	-0,81828
4,7	-0,99992
5,3	-0,83227
5,9	-0,37388



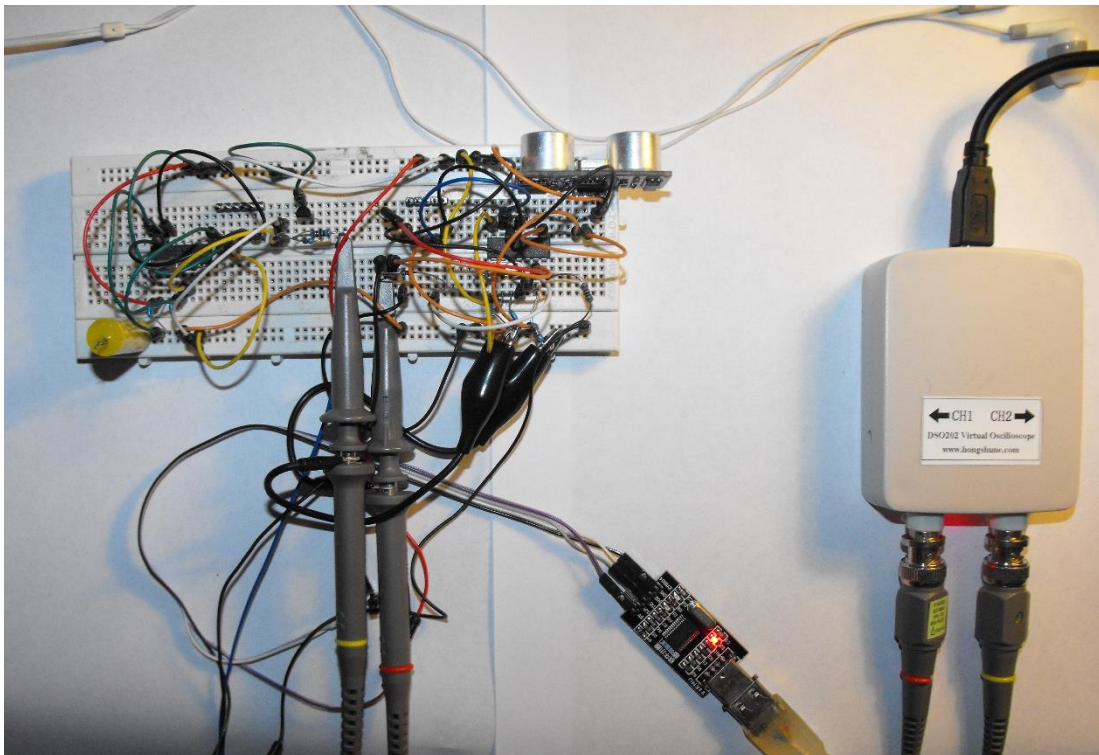
```
void szinusz(void){  
  //400 Hz quasi-sinusoidal  
  PWM1_Set_Duty(127);  
  delay_us(250);  
  PWM1_Set_Duty(188);  
  delay_us(250);  
  PWM1_Set_Duty(241);  
  delay_us(250);  
  PWM1_Set_Duty(254);  
  delay_us(250);  
  PWM1_Set_Duty(222);  
  delay_us(250);  
  PWM1_Set_Duty(158);  
  delay_us(250);  
  PWM1_Set_Duty(82);  
  delay_us(250);  
  PWM1_Set_Duty(23);  
  delay_us(250);  
  PWM1_Set_Duty(0);  
  delay_us(250);  
  PWM1_Set_Duty(21);  
  delay_us(250);  
  PWM1_Set_Duty(80);  
  delay_us(250);  
}
```

A képen látható, hogyan valósítható meg egy kvázi szinuszos jel generálása bonyolult matematikai függvények nélkül.

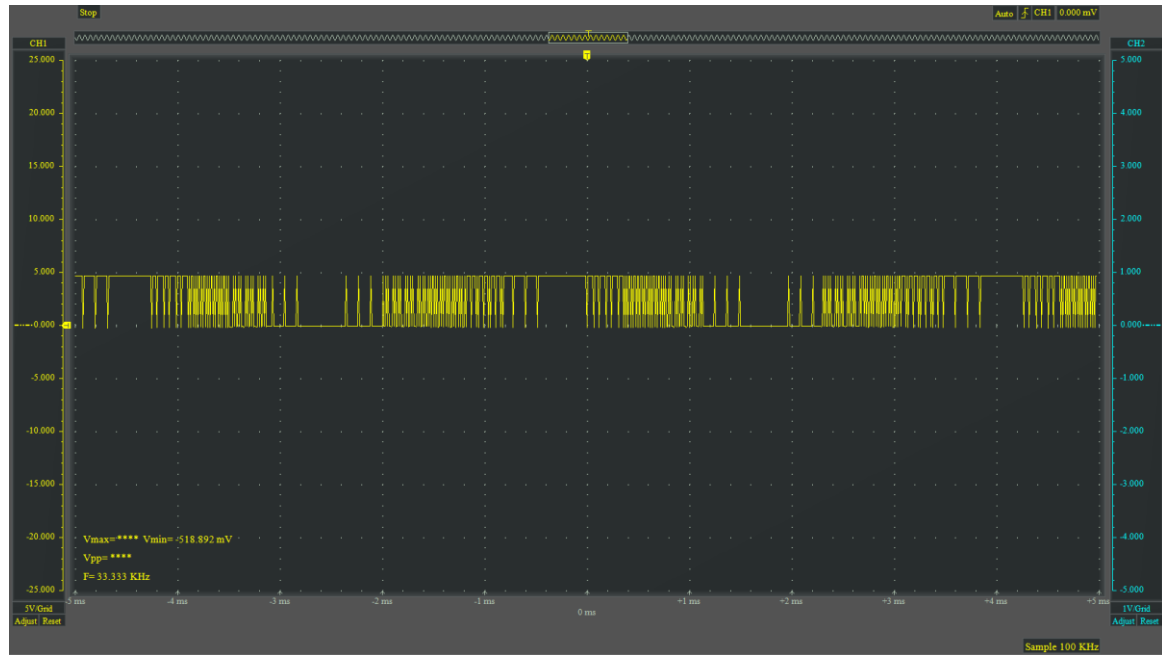
Ezzel a módszerrel elérhetővé vált, hogy egy nyolcbites mikrokontroller is képes legyen PWM segítségével szinuszos hang létrehozására.

30. ábra: A program releváns részlete

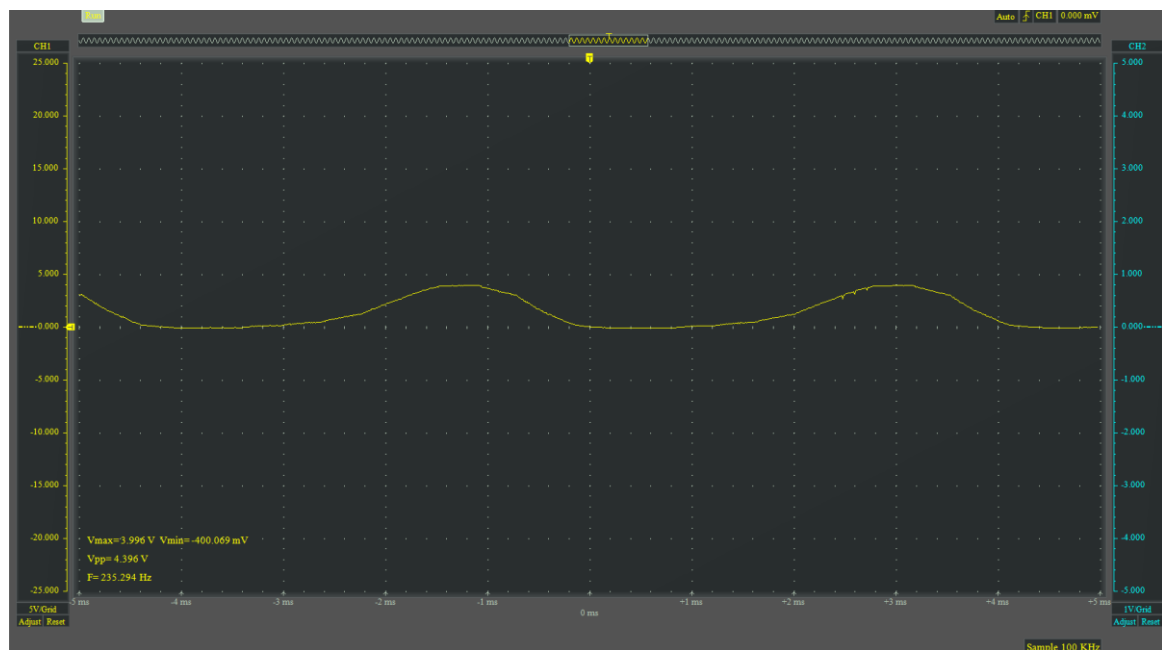
7.3. Eredmények



31. ábra: A tesztkonfiguráció



32. ábra: PWM segítségével generált szinuszos jel, szűrés nélkül



33. ábra: 4,7 uF-os fóliakondenzátor hatása a jelalakra

Mint látható, a jel szűrés után sokkal jobban hasonlít szinusz görbére. Az eltérést leginkább a gyenge minőségű szűrő magyarázza, viszont ez a jelalak tökéletesen kielégíti a követelményeket. –A generált hang megfelelő minőségű.

További fejlesztés történt az eszköz használhatóságának, illetve testreszabhatóságának érdekében. A két fő fejlesztést a következő fejezetek tárgyalják.

8. Testreszabhatóság

8.1. Triggerelési módok

A tesztkészülék eredetileg egyféle triggerelési módot alkalmazott: folytonosan mérte a távolságot és generálta a hangot. Ez az opció továbbra is elérhető marad, viszont egy külső eszköz, tulajdonképpen egy nyomógomb csatlakoztatásával elérhetővé válik az „aktív szonár” jellegű távolságmérés is.

Ez a módszer a következő módon működik:

A felhasználó egy kézi nyomógomb segítségével indíthatja a távolságmérést. Ekkor a keresőhang maximális amplitúdóval, rövid időre szólal meg a fejhallgatóban. A rendszer a távolságot lemérve, az időbeni eltérést emberi fül számára is hallhatóvá téve (magyarul a kiadott és visszaverődött hang közötti időt egy szorzóval felszorozva) jeleníti meg, méghozzá úgy, hogy a „visszavert” hang amplitúdóját is megváltoztatja a távolság függvényében.

A tájékozódás ezzel a módszerrel egyeseknek kényelmesebb lehet, hisz így nem szól folytonosan a keresőhang a fejhallgatóban, így például könnyebb a társalgás is. A felhasználó szükség szerint sűrítetheti is a „pingelést”, ahogyan azt a denevér is teszi vadászat közben, vagy zegzugosabb környezetben. Fenton (2012)

A két mód közötti váltás igen egyszerű. A készülék érzékeli, ha kiegészítőleg csatlakoztatjuk a nyomógombot, és automatikusan átvált a manuális trigger funkcióra.

8.2. Sokoldalúság

A készüléket elláttam egy audio csatlakozóval is, mely egy szabványos 3,5 mm-es 4 pólusú Jack aljat. Ez lehetővé teszi a készülék headset-ként történő alkalmazását is. Amennyiben a készülékhez okostelefont csatlakoztatunk, lehetőség nyílik az okostelefonok által kínált lehetőségek használatára is, mint amilyen például a szövegek felolvasása. A készülékbe így beépítésre került egy kondenzátormikrofon is, illetve egy hívásfogadó-megszakító gomb is.

További jellemzője ennek a módnak, hogy a felhasználó saját igénye szerinti keresőhangot használhat, hiszen így lehetőség nyílik külső forrásból származó keresőhangot alkalmazni, mely lehet valamilyen funkciógenerátor, de akár zene is, bár utóbbi nem javasolt a folytonos amplitúdó-változásai miatt.

9. Ergonómia, hordhatóság

Mivel mobil készülékről van szó, fontosnak tartottam a könnyű hordozhatóságot, illetve a kényelmes viselhetőséget is az akkumulátor üzemidő mellett. Sok elvetett ötlet után végül arra az elhatározásra jutottam, hogy a készüléket egy fejhallgatóba építve lenne a legkényelmesebb, s leghatékonyabb viselni.

Mivel olyan fejhallgató beszerzése nehézkes, melyben elegendő hely áll rendelkezésre az áramkörök elhelyezésére, így egy egyedi formát hoztam létre SolidEdge program segítségével. Hogy az így megalkotott fejhallgatót lehessen tesztelni is, egy gyors kísérleti eszközgyártó megoldást választottam, a kísérleti eszköz elkészítéséhez.



34. ábra: Az elképzelt fejhallgató

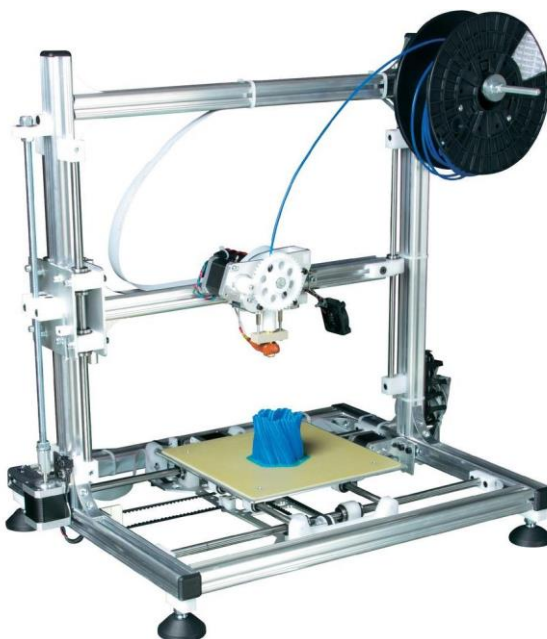
9.1. Rapid prototyping

Az alkatrészek legyártásához egy Wellmann K8200 típusú nyomtatót használtam, mely FDM (Fused Deposition Modelling) technológián alapul és termoplasztikus anyagot, ABS

műanyagot használ a háromdimenziós formák előállításához.

Az előállítandó alakzatokat SolidEdge program segítségével hoztam létre, melynek használata során végig szem előtt tartottam a nyomtató képességeit.

Ez azért fontos, mivel a FDM technológia nem alkalmas alátámasztást nélkülöző, „overhanging” részek nyomtatására.



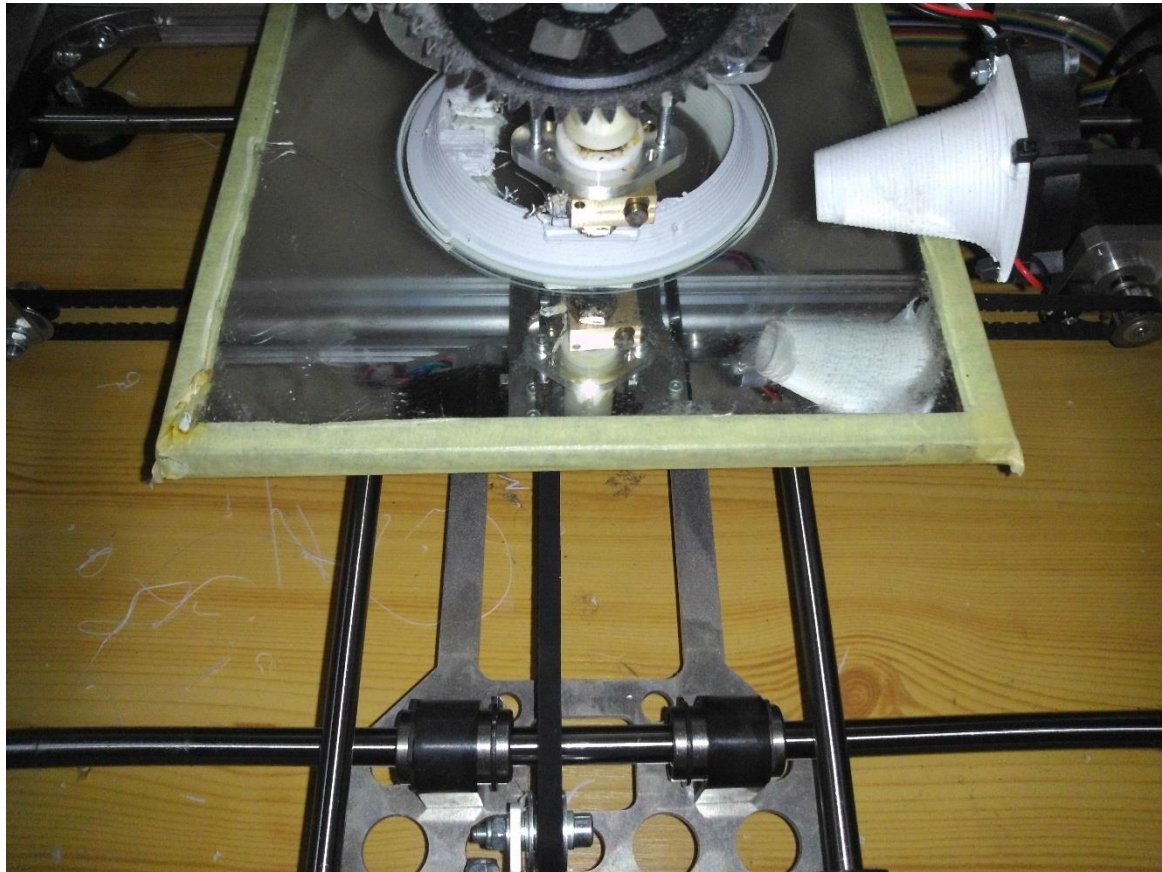
35. ábra: Wellman K8200 nyomtató

Miután az alkatrészeket megterveztem, és a program segítségével megvizsgáltam az összeszerelhetőséget, az alkatrészek rajzait STL formátumban elmentettem, mivel ez a szabványos formátuma minden gyors kísérleti eszközgyártó eljárásnak.

Az így elmentett fájlokat a nyílt forráskódú Slic3r nevű CAM szoftver segítségével daraboltam fel, mivel az FDM technológia vékony szeletek (ún. layerek) egymásra építésével alakítja ki a kívánt formát. A programot a nyomtatónak megfelelően paramétereztem.

A főbb paraméterek a következők:

- Rétegvastagság: 0,2 mm
- Első réteg vastagsága: 0,3 mm
- Kitöltés: Méhsejtszerűen
- Kitöltési tényező: 25 %
- Nyomtatási sebesség: 50 mm/s
- Gyorsulás: 200 mm/s²
- Technológiai szoknya: 5 mm
- Támaszanyag: nincs generálva
- Filament anyaga: ABS
- Filament átmérő: 2,93 mm
- Extrudálási szorzó: 1
- Extruder fúvóka átmérő: 0,5 mm



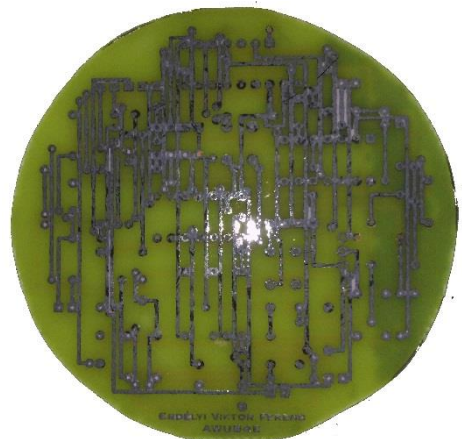
36. ábra: Gyors prototípusgyártás folyamatban

9.2. NYÁK

A prototípusgyártáshoz hozzá tartozott a nyomtatott áramkör megtervezése és legyártása is. Az áramkört DesignSpark program segítségével terveztem meg. A PCB létrehozására a vasalósos panelgyártási technológiát alkalmaztam, mivel olcsó, házilag könnyen elkészíthető, és minőségi NYÁK-ot lehet vele létrehozni.

A technika a következő:

A kívánt rajzolatot tükrözve lézernyomtatóval fényes papírra kinyomtatjuk, majd a rajzolatot mechanikailag és kémiaiilag is megtisztított nyers NYÁK-ra fektetjük. A helyére igazított rajzolatot ezután vasalóval átjuttatjuk a rézfelületre. A feleslegessé vált papírt leáztatjuk, majd a maratás következik. Az így készített NYÁK jó minőségű, de a technológia nem alkalmas sorozatgyártásra.



37. ábra: Az elkészült NYÁK

10. Az elkészült kísérleti eszköz tesztelése

Kétféle teszt sorozatot hajtottam végre a készülék vizsgálata során. Egyik típusú vizsgálat a folytonos üzem hatékonyságát, míg a másik a manuális működés használhatóságát vizsgálta. A tesztek kiértékelése során szükséges volt figyelembe venni a gyenge minőségű szenzorok által korlátozott hatótávolságot.

A tesztek kollégám, Tóth János segítségével hajtottam végre, melyek során véletlenszerűen elhelyezett akadályok mellett kellett elhaladnom a látásom teljes nélkülözésével, csak a készülékre hagyatkozva.

10.1. Folytonos üzem:

Folytonos üzemmódban használva a készüléket igen részletes képet sikerült kapnom a környezetemről, viszont hosszútávon ez kellemetlenek bizonyult, annak ellenére, hogy szignifikáns előrelépés mutatkozott az előd hangszintézisével szemben.



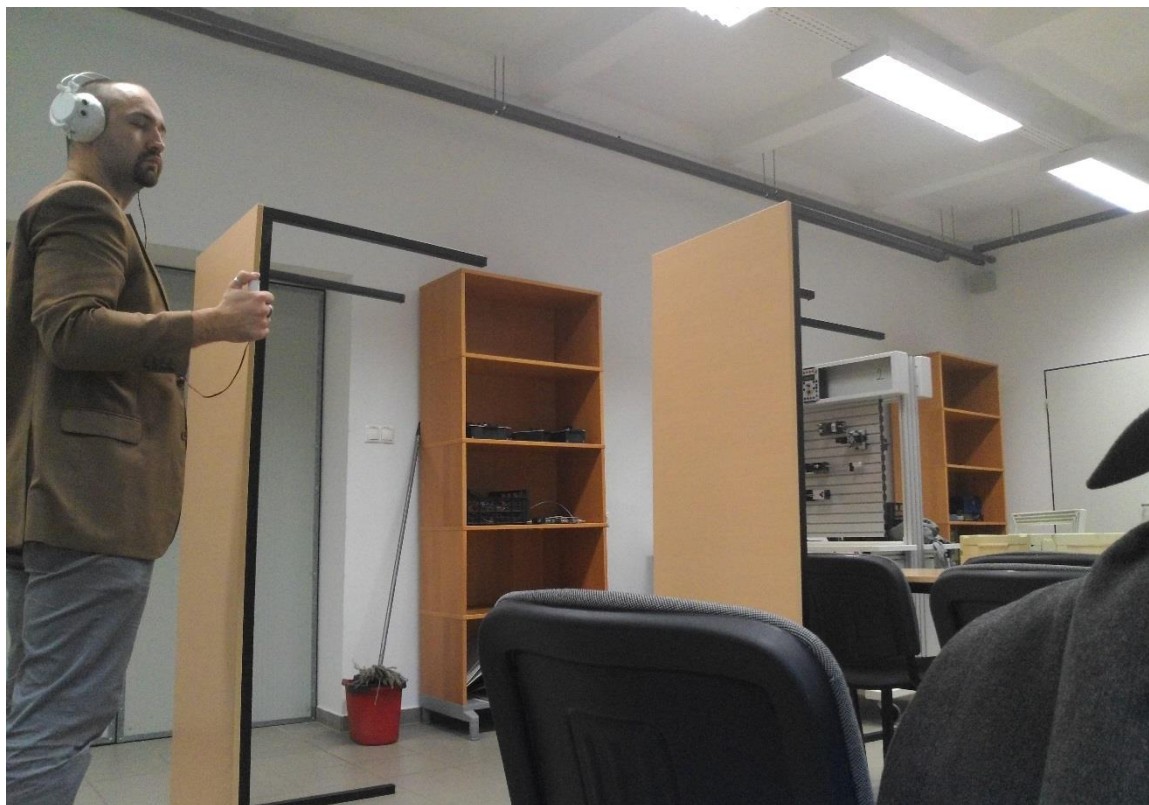
38. ábra: Folytonos üzem tesztelése, véletlenszerűen elhelyezett akadályokkal 1.



39. ábra: Folytonos üzem tesztelése, véletlenszerűen elhelyezett akadályokkal 2.

10.2. Manuális üzem:

Ezen üzemmód használata sokkal kellemesebb használatot biztosít, mivel nincs folytonos zajterhelés. A részletesség viszont függ a felhasználtól, hisz az ember kezdeményezi a távolságmérést, és így nem is olyan gyors reagálású a rendszer. A rendszer hiányossága ebben az esetben még, hogy a felhasználó egyik keze folyamatosan foglalt.



40. ábra: Manuális üzem tesztelése véletlenszerűen elhelyezett akadályokkal 1.



41. ábra: Manuális üzem tesztelése, véletlenszerűen elhelyezett akadályokkal 2.

10.3. A tesztek kiértékelése, továbbfejlesztés aspektusai

A készülék kielégítette a fejlesztési célokban meghatározott szempontokat. Kényelmesen használhatónak bizonyult még hosszú távon is, főleg a manuális üzemmódot alkalmazva. A kísérleti eszköz legfőbb hibája továbbra is a szenzorok mérési hibája. Sajnos ezekkel a szenzorokkal nem lehet értékelhető információhoz jutni 1 m távolság felett elhelyezkedő tárgyak távolságáról.

A tesztek figyelembe véve a főbb továbbfejlesztési irányvonalak a következők:

- A készüléket megfelelő minőségű szenzorokkal kell ellátni.
- A készülék méreteit csökkentendő, SMD alkatrészek felhasználásával szükséges az elektronikát felépíteni.
- Az üzemidő növeléséhez disszipatív tápegység helyett kapcsolóüzemű tápegység alkalmazása szükséges.
- A működtető 9 V-os telep leváltásához szabványos 3,7 V-os Li-Ion akkumulátor alkalmazása lehet a költséghatékony megoldás.
- Az áramforrás cseréjével megfontolandó az eddig alkalmazott TTL logikai feszültségszintű komponensek cseréje CMOS feszültségszintű elemekre.
- A készülék bluetooth headset-é alakítása lehetőséget nyújtana okostelefonnal történő összekapcsolásra, így a feldolgozó szoftver a telefonra kerülhetne, minek következtében a szoftverfrissítés nagyságrendekkel egyszerűsödne. Ennek következménye lenne az is, hogy a testreszabhatóság is jelentősen növekedne.

11. Gazdasági számítás

Mivel a készülék akár piaci forgalomba is kerülhet a további fejlesztések eredmény képpen, fontos gazdasági számításokat is végezni. –Így lesz meghatározható az az anyagi ráfordítás, melynek meg kell térülnie.

Nem elhanyagolható szempont a piac érzékenysége sem az árra, így fontos szem előtt tartani a költséghatékony megoldásokat is.

Mivel egészségügyi segédeszközzről beszélhetünk, érdemes megfontolni az esetleges állami támogatási lehetőségeket is.

Kísérleti eszköz előállítási költsége, fejlesztési költségek:

Kísérleti eszköz előállítási költsége

4. táblázat: Az elkészült kísérleti eszköz anyagköltsége

1	Szenzor	780	990,6	link	2	1560	1981,2
2	PIC16F1823	280	355,6	link	3	840	1066,8
3	Nyák	1000	1270	link	1	1000	1270
4	Kiegészítő elektronikai alkatrészek	4000	5080	link	-	4000	5080
5.	Műanyag alkatrészek	1270	1612,9	-		1270	1612,9
	Összesen:					8670	11010,9

5. táblázat: Szofisztikált szenzor alkalmazásával, és ipari minőségű NYÁKKAL

#	Megnevezés	Ár Nettó	Ár Bruttó	Link	db	Össz nettó	Össz Bruttó
1	Szenzor	7420	9423,4	link	2	14840	18846,8
2	PIC16F1823	280	355,6	link	3	840	1066,8
3	Nyák	6358	8074,66	link	1	6358	8074,66
4	Kiegészítő elektronikai alkatrészek	4000	5080	link	-	4000	5080
5.	Műanyag alkatrészek	1270	1612,9	-		1270	1612,9
	Összesen:					27308	34681,16

A számításoknál jelen pillanatban nem vettem figyelembe az esetleges sorozatgyártásnál jelentkező szignifikáns árcsökkenést alkatrészek terén. A fejlesztési költség meghatározására a készülék fejlesztésével töltött időt, illetve a fejlesztés közben felhasznált villamos energiát (számítógép, forrasztópáka, 3D nyomtató) vettem figyelembe.

6. táblázat: Egyéb költségek

Fejlesztéssel töltött idő:		Mérnöki órabér		Munkaórák pénzben kifejezett értéke			
120 h		1875 Ft (Nettó)	2850 Ft (Bruttó)	225000 Ft (Nettó)	342000 Ft (Bruttó)		
Elektromos energiahasználat		Átlagos energiafogyasztás	Villamos energia költsége	Elhasznált villamos energia	Elhasznált villamos energia értéke		
120 h		0,5 kWh	39,9 Ft/kWh	60 kWh	2394 Ft (Bruttó)		
Összesen:					344394 Ft (Bruttó)		

Összességében, ha a készülék egyedi darabnak készülne, **379075 Ft**-ba kerülne. Ez természetesen nem rentábilis, így legalább a kissorozat-gyártás lehetőségét is számba kell venni.

Ha a készülék kissorozat-gyártásba kerülne, és **Bruttó 40000 Ft**-ért lehetne értékesíteni, akkor a fejlesztés megtérülése **65 darab** eladása után következne be.

Mivel ez a szám beláthatóan alacsony, így érdemes a készülék forgalmazását megfontolni.

12. Összefoglalás

A dolgozatom kitűzött célja olyan tájékozódási stratégia, és egy ezt megvalósító eszköz kifejlesztése volt, mellyel virtuális képalkotásra nyílik lehetőség, illetve melynek segítségével vak és gyengén látó emberek tájékozódását lehet megkönnyíteni.

A módszer felépítéséhez áttekintettem a szakirodalmat, mely alapján egy ultrahangos távolságmérő rendszerre esett a választásom. Mivel a természetben ez működő módszer, megvizsgáltam az emlősök ultrahanggal történő tájékozódási technikájának irodalmát, melyet szakzsargonban „echolocation”-nek neveznek. Ez alapján kezdtem a műszaki megoldás kialakítását, melyhez mikroelektronikai és informatikai, illetve gépészeti eszközöket vettem igénybe. A feltételezésem bizonyítására (miszerint az ember is képes lehet a visszaverődött hangok alapján tájékozódni) egy tesztkészüléket építettem, melyet különféle módokon próbáltam ki, értékeltem.

A tesztsorozat eredményei megfelelőek voltak, maradéktalanul kielégítették a célkitűzésben foglaltakat. A készülékkel megvalósítható a virtuális képalkotás, segítséget nyújthat világtalan embereknek a térbeni tájékozódásban, és a továbbfejlesztés nyomán egy szignifikánsan alkalmazhatóbb eszköz valósult meg. Az eszköz továbbfejlesztési iránya a már létező, és jól működő okostelefonos alkalmazásokkal történő összekapcsolás, minek következtében a hangszintézis, illetve a jelfeldolgozás szerepét okostelefon venné át, mely nagyban megkönnyíti a további szoftverfrissítéseket.

13. Summary

The aim of my thesis was to develop a strategy for orientation, a device which give ability to create virtual images, a device which grants blind and visually impaired people to find their way.

To create the method, i reviewed the literature, and I chose the ultrasonic distancemeter system. This method works well in the nature, so I examined the literature of mammals' ultrasonic orientation technics (in the terminology: Echolocation). With this knowledge, I began to build a technical solution, using microelectronic, informatic and mechanical tools. To prove my expectation (that a person is able to navigate himself based on reflected sounds), I made a prototype and I tested it different ways.

The result of the testseries was appropriate and satisfying. With this device we can achieve to create virtual images, we may help blind people navigate in space, further development also planned. The further development is mainly intended to convert the device to a bluetooth headset. Therefore the device could be used with the existing, and well-functioning smartphone applications. As a consequence the smartphone could take the over the role of voice synthesis and the data processing. This could make the further software updates more easier.

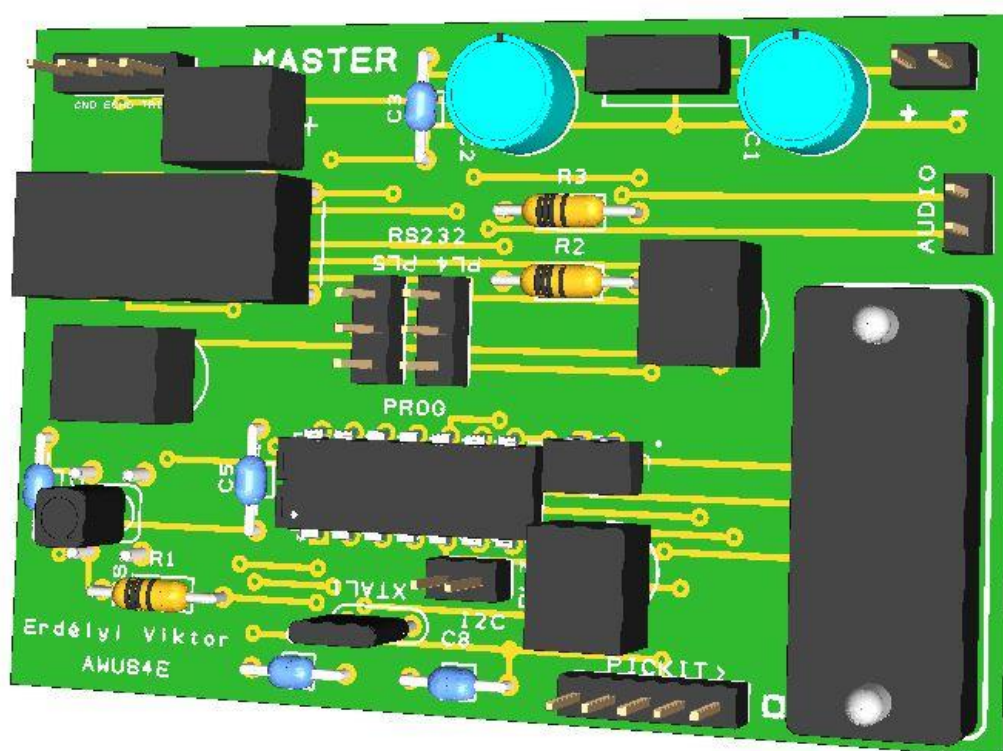
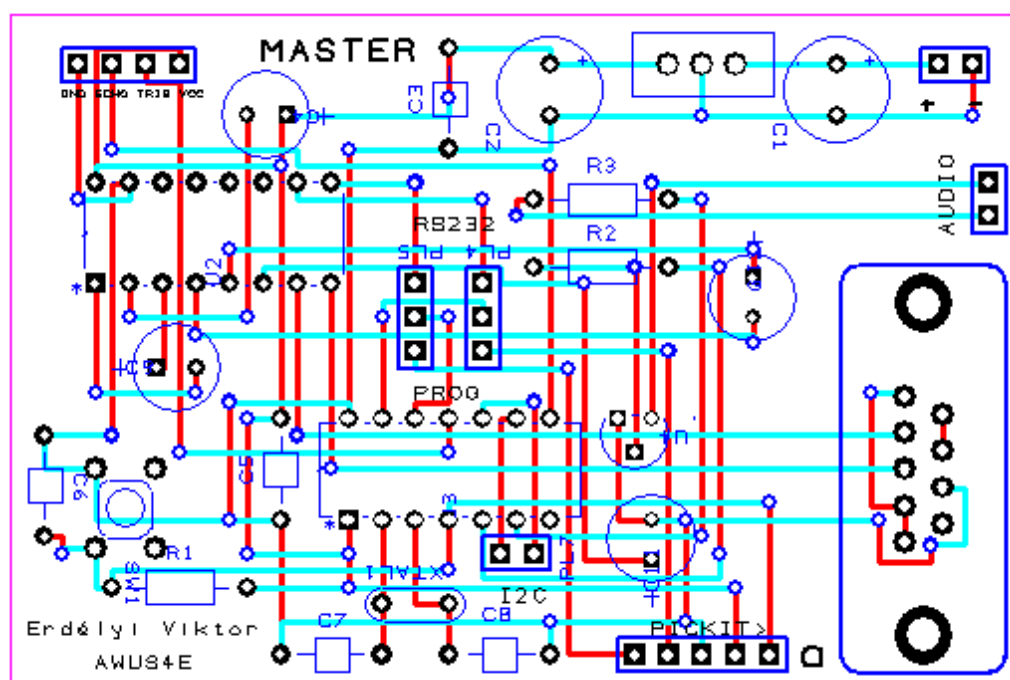
14. Irodalomjegyzék

1. Allan R. Hambley (2010) Electrical Engineering Principles and Applications fifth edition p. 408-415., 2011. ISBN-13:978-0-13-213006-6
2. Allan R. HambleyByte Craft Limited – First Steps with Embedded Systems p. 5., 2002
3. Augarten, Stan (1983). "The Most Widely Used Computer on a Chip: The TMS 1000". State of the Art: A Photographic History of the Integrated Circuit (New Haven and New York: Ticknor & Fields). ISBN0-89919-195-9. Retrieved 2009-12-23. p. 38-40.
4. D. R. Griffin (1958). Listening in the dark. Yale Univ. Press, New York.
5. Dr. Arany Antal, Dr. Vízer Miklós (2008): A három- és négydimenziós ultrahang-diagnosztika a szülészetben. IME 3. p. 36-39.
6. H. Burton (2003) The Journal of Neuroscience, 23(10): p. 4005-4011.
7. Hill, M. N. (1962). Physical Oceanography. Allan R. Robinson. Harvard University Press. p. 498.
8. Jones G. (2005). "Echolocation" Current Biology 15(13): p. 484-488.
9. Keith H. Sueker Power Electronics Design: A practitioner's Guide, © 2005,
10. M. Brock Fenton(2012) Bats átdolgozott kiadás. Fitzhenry & Whiteside 2012 ISBN 1550414828 Mihajlik Péter, Tatai Péter, Balog András (2004): Vakok tájékozódását segítő eszköz integrált sztereo ultrahangos és rádiós rendszerrel. IME III. évfolyam 2. szám p. 43-46.
11. Naohiro Matsuta, Shizuko Hiryu, Emyo Fujioka, Yasufumi Yamada, Hiroshi Riquimaroux and Yoshiaki Watanabe (2013) Adaptive beam-width control of echolocation sounds by CF–FM bats, *Rhinolophus ferrumequinum nippon*, during prey-capture flight The Journal of Experimental Biology 216. p. 1210-1218.
12. Rózsa Sándor (1977) Elektronikus amatőr mérőkészülékek. 2. javított kiadás. Műszaki Könyvkiadó. ISBN 9631018040
13. S. Dijkgraaf (1949). Spallanzani und die Fledermäuse. *Experientia* 5: p. 90-92.

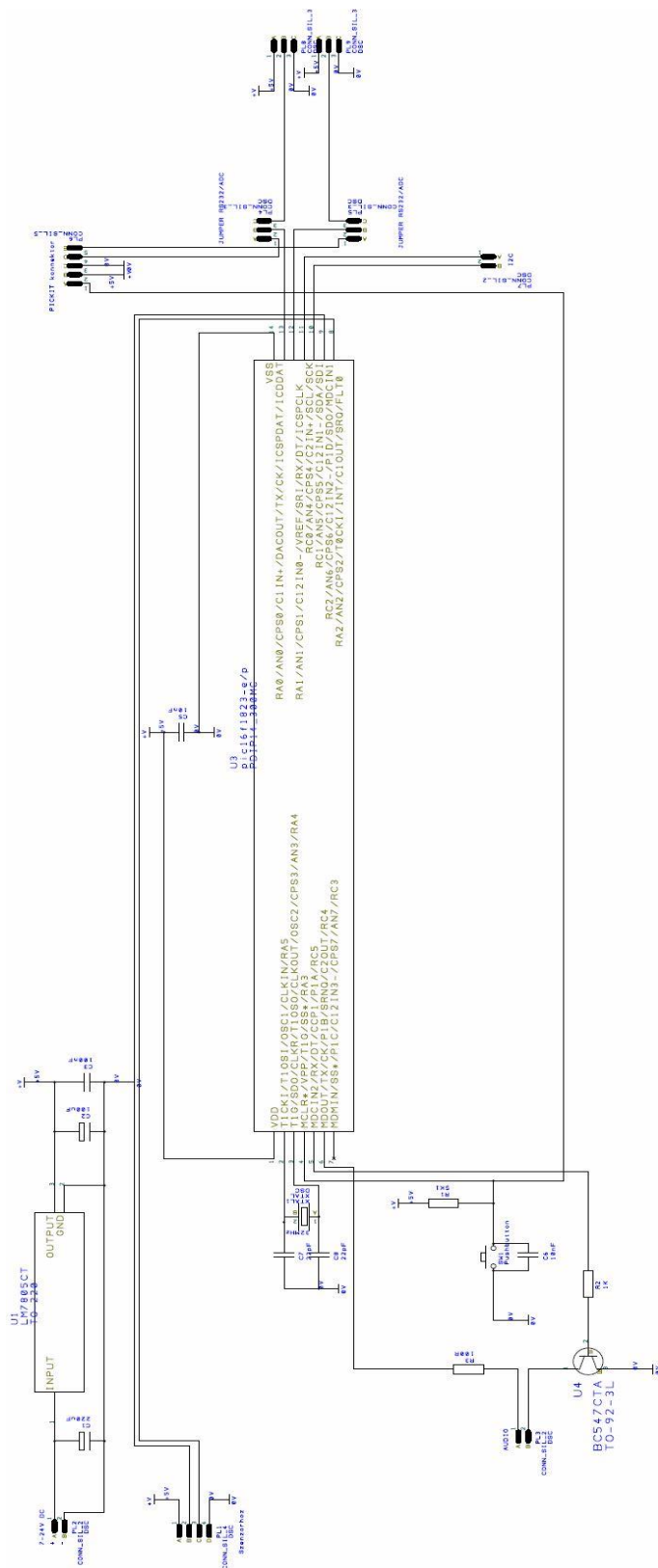
14. SciTechMansoor-Ul-Hassan Siddique (2009) Ultrasonic radar and its applications
Recent Advances in Applied Informatics and Communications ISBN: 978-960-474-
107-6. p. 225-231.
15. Stuart Parsons and Gareth Jones (2000) Acoustic identification of twelve species
of echolocating bat by discriminant function analysis and artificial neural networks
The Journal of Experimental Biology 203, p. 2641–2656.
16. Suga N, Simmons J.A. & Jen P.H. (1975) Peripheral specialization for fine analysis
of doppler-shifted echoes in the auditory system of the "CF-FM" bat *Pteronotus*
parnellii. Journal of Experimental Biology, 63. p.161-192.
17. U. Tietze Ch. Schenk (1990) Analóg és digitális áramkörök. 4. átdolgozott kiadás.
Műszaki Könyvkiadó. 1990. ISBN 9631082091

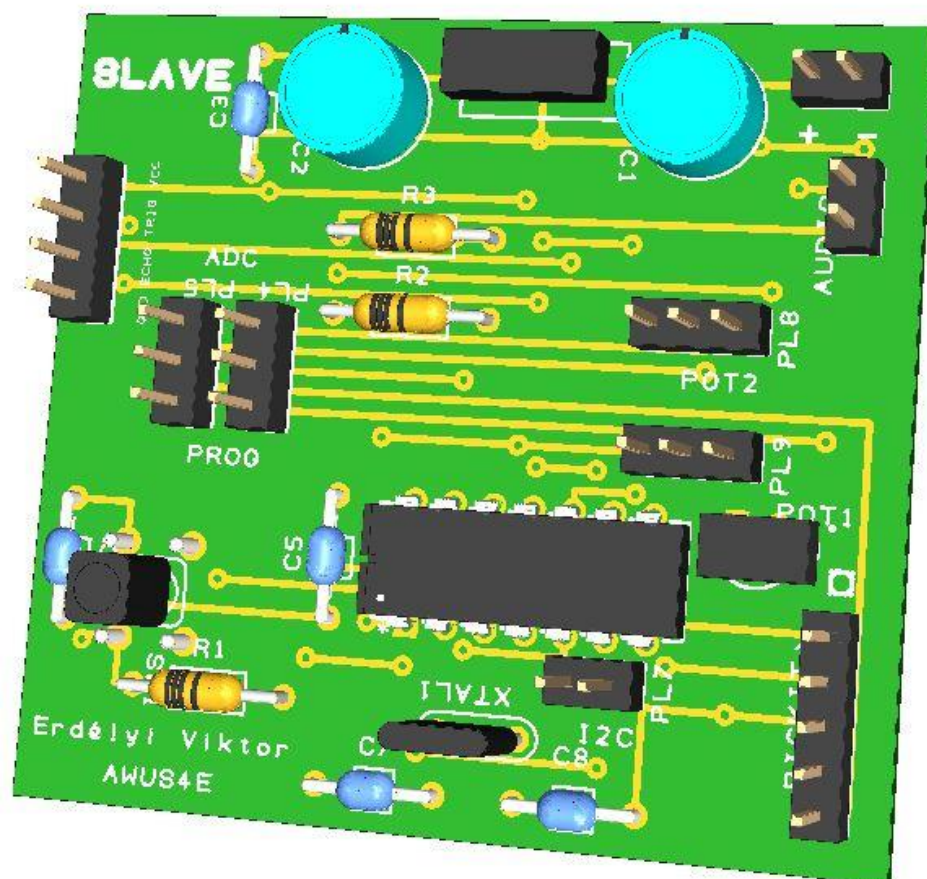
15. Köszönetnyilvánítás

Ezúton is szeretném megköszönni, és tiszteletemet kifejezni **Dr. Jánosi Lászlónak**, aki munkám során végig támogatott, hasznos tanácsokkal látott el, **Gergely Zoltánnak**, akire számíthattam, ha elektronikai, illetve hardver közeli programírási problémákba botlottam, **Tóth János** kollégámnak, akihez a felmerülő számítástechnikai gondjaimmal fordulhattam, és **Dr. Keresztes Róbertnek**, aki lehetővé tette, hogy a tanműhelyben elkészítsem a két szabadságfokú mérőállványt, mely fontos része a kísérleti készüléknek.



Slave áramkör kapcsolási rajza, nyákterve, 3D modellje:





3. melléklet

A Master mikrokontroller programkódja

```

/*****
*          "BatEye"          *
*TÁJÉKOZÓDÁSI STRATÉGIA*
*  ÉS IRÁNYÍTÓ ESZKÖZ    *
*      FEJLESZTÉSE        *
*LÁTÁSSÉRÜLTEK SZÁMÁRA *
*  PIC16f1823 @ 32MHz    *
*  2013 Erdélyi Viktor   *
*      MASTER            *
*****/
//I2C variables
#define TX_ELMNTS      1
#define RX_ELMNTS      3
// array for master to write data from slave
volatile unsigned char I2C_Array_Tx[TX_ELMNTS] = {0xFF};

// array for master to write to
volatile unsigned char I2C_Array_Rx[RX_ELMNTS] = {0x01, 0x00, 0xFD};

unsigned char I2C_slave_write_add = 0x30;    // slave address R/nW = 0
(write)
unsigned char I2C_slave_read_add = 0x31;     // slave address R/nW = 1
(read)
unsigned char BRG_val = 0xB3;                 // 0xB3 sets the SCL freq
to 100kHz

unsigned int index_i2c = 0;                   // index pointer
unsigned char junk = 0;                       // place to put unnecessary
data
unsigned char clear = 0x00;                   // used to clear array
location
unsigned char init_start = 1;                 // used to get start bit
set
unsigned char stop = 0;                       // used to know when to
stop
unsigned char write_to_slave = 1;             // used to set up master
transmission
unsigned char trans_wrt_add = 1;              // used to set up address
for write
unsigned char init_trans_data = 0;           // used to set up data
transmission
unsigned char init_mstr_rec = 0;              // used to set up master
reception
unsigned char rec_byte = 0;                   // used to know when byte
received
unsigned char set_RCEN = 0;                   // used to set RCEN bit
unsigned char trans_data = 0;                 // used to get transmission
going
unsigned char done = 0;                       // used to know when
finished
unsigned char read_from_slave = 0;            // used to set up reception
routine

```

```

unsigned char trans_rd_add = 1;           // used to set up address
for read
unsigned char read_rec_byte = 0;         // used to read data into
array
unsigned char set_ACKEN = 0;             // used to set ACKEN bit

// 8 bit variables
typedef unsigned char uint8;

// flags
bit oldstate;
bit echoold;

// Variables
uint8 Dist1;                             //Distance and azimuth
variables
uint8 Pot1;
uint8 Pot2;
uint8 Dist2;
uint8 Pitch;

// denominate I/O bits

sbit Echo at RC2_bit;
sbit Echo_d at TRISC2_bit;
sbit Trig at RA2_bit;
sbit Trig_d at TRISA2_bit;

void Tone1() {
    Sound_Play(500, 20);                 // Frequency = 659Hz, duration
= 1000ms
    //Sound_Play(Pitch, 20);             // Uncomment to modulate pitch
}

void hardinit(void) {
    CPSCON0.CPSON = 0;                   // Disable CPS module
    OPTION_REG.b7 = 0;                   // Disable weak pull up
    WPUC = 0;
    ANSELA = 0;                           // Configure AN pins as digital
    ANSELC = 0;
    C1ON_bit = 0;                         // Disable comparators
    C2ON_bit = 0;
    APFCON = 0b10000100;
    //OSCCON = 0b01111010;                // Internal OSC @ 16MHz
    OPTION_REG = 0b11010111;              // prescaler 1:65536
    //WDTCON = 0b00010111;                // period is 2 sec (RESET

value)
    PORTC = 0x00;                         // Clear PORTC
    LATC = 0x00;                           // Clear PORTC latches
    TRISC = 0b00000011;                   // Set RC0, RC1 as inputs for
I2C
    //I2C MASTER MODULE SET UP
    SSP1STAT = 0b10000000;                // Slew rate control disabled
for
    // standard speed mode (100
kHz, 1 MHz)

```

```

        SSP1CON1 = 0b00101000;           // Enable serial port, I2C
master mode,                             // clock=Fosc/(4*(SSPAD+1))
        SSP1CON3.SDAHT = 1;              // Minimum of 300 ns hold time
on SDA                                   // after the falling edge of
SCL
        SSP1ADD = BRG_val;                // This sets Baud rate
        PIR1.SSP1IF = 0;                  // Clear the serial port
interrupt flag
        PIR2.BCL1IF = 0;                  // Clear the bus coll interrupt
flag
        PIE2.BCL1IE = 1;                  // Enable bus collision
interrupts
        PIE1.SSP1IE = 1;                  // Enable serial port
interrupts
        INTCON.PEIE = 1;                  // Enable peripheral interrupts
        INTCON.GIE = 1;                  // Enable global interrupts
        Echo_d = 1;                       // Init I/O ports
        Trig_d = 0;
        Trig = 0;
        UART1_Init(9600);                 // Initialize UART module at
9600 bps
        Delay_ms(100);                     // Wait for UART module to
stabilize
        Sound_Init(&PORTC, 4);             // Init sound
        PWM1_Init(25000);                  // Initialize PWM1 module at
25KHz
        PWM1_Start();                      // Start PWM1
        oldstate = 0;                      // Init flags
        echoold = 0;
        Dist1 = 0;                         // Initialize variables
        Dist2 = 250;
        Pot1 = 0;
        Pot2 = 0;
        Pitch = 400;
}

```

```

void distM(void) {                         // Subroutine for distance
measuring
    Dist2 = 255;
    echoold = 0;
    oldstate = 0;
    Trig = 1;
    oldstate = 1;
    delay_us(11);
    Trig = 0;
    if (oldstate == 1){
        while (echoold == 0){
            if (Echo == 1){
                echoold = 1;
                while (Echo == 1) {
                    delay_us(100); // 17 mm precision
                    Dist2 --;

```

```

        //Pitch = Dist2*5+400;    // +17 mm => +5 Hz
(uncomment to modulate Pitch)
    }
    }
}

void trans(void){
    if(init_start)                // get ready to start
    {
        SSP1CON2.SEN = 1;        // Initiate START condition
        init_start = 0;
    }
    if(init_trans_data)          // get ready to transmit data
    {
        trans_data = 1;
        init_trans_data = 0;
    }
    if(done)                     // when all data is
transferred
    {                             // send stop bit
        stop = 1;
        done = 0;
        SSP1CON2.PEN = 1;        // stop bit set
    }

} // end transmission

void read(void){
    if((!write_to_slave) && (!read_from_slave)) // get ready to now read
    {
        read_from_slave = 1;      // get read routines going
        done = 0;
        index_i2c = 0;
        init_start = 1;           // get ready to start
    }

    if(read_from_slave)           // read from slave
    {
        if(init_start)            // set up start condition
        {
            SSP1CON2.SEN = 1;      // Initiate START condition
            init_start = 0;
        }
        if(init_mstr_rec)         // get ready to receive data
        {
            set_RCEN = 1;          // set master into receive
mode
            init_mstr_rec = 0;
        }
        if(rec_byte)             // check to see if data
arrived
        {
            read_rec_byte = 1;
        }
    }
}

```

```

        if(set_ACKEN)                                // set up ACKEN bit
        {
            set_ACKEN = 0;
            init_mstr_rec = 1;
            SSP1CON2.ACKEN = 1;
        }
        if(done)                                     // if array is full time to
stop
        {
            stop = 1;
            SSP1CON2.PEN = 1;                        // set stop bit
        }
    }
} //end read routine

void main() {
    hardinit();
    while(1) {

        trans();
        distM();                                    // Measuring distance
        PWM1_Set_Duty(dist2);                       // Set volume
        Tone1();
        read();
        UART1_Write(I2C_Array_Rx[0]);               // Data transmission to PC
        UART1_Write(I2C_Array_Rx[1]);
        UART1_Write(I2C_Array_Rx[2]);
        UART1_Write(Dist2);
        UART1_Write('\n');

    }
}

void interrupt()
{
    if(PIR1.SSP1IF)
// is it an SSP interrupt?
    {
        if(write_to_slave)
// write to slave
        {
            if(!done)
// if not done, continue
            {
                if(trans_wrt_add)
// get ready to transmit
                {
// address with write bit enabled
                    SSP1BUF = I2C_slave_write_add;
// load buffer with write address
                    trans_wrt_add = 0;
                    init_trans_data = 1;
                }
                if(trans_data)
// ready to transmit
                {
                    RC3_bit = 1;

```

```

        if(index_i2c < TX_ELMNTS)
// make sure valid data
        {
            SSP1BUF = I2C_Array_Tx[index_i2c];
// load buffer with data
            I2C_Array_Tx[index_i2c++] = clear;
// clear that location
        }
// and increment index
        else
        {
            trans_data = 0;
// done transmitting
            done = 1;
        }
    }
}
if(stop)
// done transmitting
{
    write_to_slave = 0;
// clear so no more data transmitted
    stop = 0;
}
}
if(read_from_slave)
// read from slave
{
    if(!done)
// if not done continue
    {
        if(trans_rd_add)
// set up read address
        {
            SSP1BUF = I2C_slave_read_add;
// address and read bit
            trans_rd_add = 0;
            init_mstr_rec = 1;
        }
        if(set_RCEN)
// set up module for read
        {
            set_RCEN = 0;
            SSP1CON2.RCEN = 1;
// set RCEN to enable read
            rec_byte = 1;
        }
        if(read_rec_byte)
// reading data
        {
            if(index_i2c < RX_ELMNTS)
// make sure valid location
            {
                read_rec_byte = 0;
                rec_byte = 0;
            }
        }
    }
}

```

```
        I2C_Array_Rx[index_i2c] = SSP1BUF;
// load array with data
        index_i2c++;
// increment counter
        set_ACKEN = 1;
// get ready to ACK
    }
    else
    {
// received all data
        init_mstr_rec = 0;
        rec_byte = 0;
        read_rec_byte = 0;
        done = 1;
    }
}
}
if(stop)
{
    read_from_slave = 0;
// clear so no more data is received
}
}

if(PIR2.BCL1IF)
// Did a bus collision occur?
{
    junk = SSP1BUF;
// dummy read to clear BF bit
    PIR2.BCL1IF = 0;
// clear BCLIF flag bit
}
PIR1.SSP1IF = 0;
// clear SSPIF flag bit
}
```

4. melléklet

A Slave mikrokontroller programkódja

```
/******  
 *      "BatEye"      *  
 *TÁJÉKOZÓDÁSI STRATÉGIA*  
 * ÉS IRÁNYÍTÓ ESZKÖZ *  
 *      FEJLESZTÉSE   *  
 *LÁTÁSSÉRÜLTEK SZÁMÁRA *  
 * PIC16f1823 @ 32MHz *  
 * 2013 Erdélyi Viktor *  
 *      SLAVE        *  
 *****/  
#define RX_ELMNTS      1  
#define TX_ELMNTS      3  
  
// 8 bit variables  
typedef unsigned char uint8;  
typedef unsigned int uint16;  
  
uint8 I2C_slave_address = 0x30;           // slave  
address  
volatile unsigned char I2C_Array[RX_ELMNTS] = {0}; // array for  
master to write to  
  
volatile unsigned char I2C_Array_TX[TX_ELMNTS] = // array for  
master to read from  
{0xFF, 0xEE, 0xDD};  
  
uint16 index_i2c = 0;           // used as an index pointer in array  
uint8 junk = 0;                 // used to place unnecessary data  
uint8 first = 1;                // used to determine whether data  
address                          // location or actual data  
  
// flags  
bit oldstate;  
bit echoold;  
  
// Variables  
uint8 Dist1;                    //Distance and azimuth  
variables  
uint8 Pot1;  
uint8 Pot2;  
uint8 Pitch;  
uint16 ADC0;  
uint16 ADC1;  
// denominate I/O bits  
  
sbit Echo at RC2_bit;  
sbit Echo_d at TRISC2_bit;  
sbit Trig at RA2_bit;  
sbit Trig_d at TRISA2_bit;  
sbit LED at RC3_bit;  
sbit LED_d at TRISC3_bit;
```

```

void Tone1() {
    Sound_Play(500, 20);           // Frequency = 500Hz, duration
    = 20ms                          // Uncomment to modulate pitch
    //Sound_Play(Pitch, 20);
}

void hardinit(void) {
    OPTION_REG = 0b11010111;      // WPU disabled, INT on rising
    edge, FOSC/4                   // Prescaler assigned to TMR0,
                                   // rate @ 1:256
    CPSCON0.CPSON = 0;             // Disable CPS module
    WPUC = 0;                      // Disable weak pull up
    ANSELA = 0b00000011;          // Configure AN pins as
    digital, expect AN0 and AN1
    ANSELC = 0;
    C1ON_bit = 0;                  // Disable comparators
    C2ON_bit = 0;
    PORTC = 0x00;                  // Clear PORTC
    LATC = 0x00;                   // Clear PORTC latches
    TRISC = 0b000011;              // Set RC0, RC1 as inputs for
    I2C                             //I2C SLAVE MODULE SET UP
    SSP1STAT = 0b10000000;         // Slew rate control disabled
    for standard                    // speed mode (100 kHz and 1
    MHz)                            // Enable serial port, I2C
    SSP1CON1 = 0b00110110;         // 7-bit address
    slave mode,                     // Clock stretching is enabled
    SSP1CON2.SEN = 1;               // SSPBUF is updated and NACK
    SSP1CON3.BOEN = 1;              // for a received address/data
    is generated                    // ignoring the state of the
    byte,                           // only if the BF bit = 0
    SSPOV bit                        // Minimum of 300 ns hold time
    SSP1CON3.SDAHT = 1;             // the falling edge of SCL
    on SDA after                    // Enable slave bus collision
    SSP1CON3.SBCDE = 1;
    detect interrupts                // Load the slave address
    SSP1ADD = I2C_slave_address;    // Clear the serial port
    PIR1.SSP1IF = 0;
    interrupt flag                  // Clear the bus collision
    PIR2.BCL1IF = 0;
    interrupt flag                  // Enable bus collision
    PIE2.BCL1IE = 1;
    interrupts                       // Enable serial port
    PIE1.SSP1IE = 1;
    interrupts                       // Enable peripheral interrupts
    INTCON.PEIE = 1;
    INTCON.GIE = 1;                // Enable global interrupts

    APFCON = 0b10000100;

```

```

        Echo_d = 1;                                // Init I/O ports
        Trig_d = 0;
        Trig = 0;
        Sound_Init(&PORTC, 4);                     // Init sound
        PWM1_Init(25000);                           // Initialize PWM1 module at
25KHz
        PWM1_Start();                               // Start PWM1
        ADC_Init();                                 // ADC init
        oldstate = 0;                               // Init flags
        echoold = 0;
        Dist1 = 0;                                  // Initialize variables
        Pot1 = 0;
        Pot2 = 0;
        Pitch = 400;
        LED_d = 0;
        LED = 0;
        I2C1_init;

    }

    void adcread(void){                             // Reading ADC values,
    converting into 8 bit format
        ADC0 = ADC_Read(0);                         // reading 10-bit value from ADC
        ADC1= ADC_Read(1);

        Pot1 = ADC0 >> 2;                           // writeing significant 8 bit to
variable
        Pot2 = ADC1 >> 2;

        I2C_Array_TX[0] = Pot1;
        I2C_Array_TX[1] = Pot2;
    }

    void distM(void) {                               // Subroitone for distance
    measuring
        Dist1 = 255;
        echoold = 0;
        oldstate = 0;
        Trig = 1;
        oldstate = 1;
        delay_us(11);
        Trig = 0;
        if (oldstate == 1){
            while (echoold == 0){
                if (Echo == 1){
                    echoold = 1;
                    while (Echo == 1) {
                        delay_us(100);    // 17 mm precision
                        Dist1 --;
                        //Pitch = Dist2*5+400;    // +17 mm => +5 Hz
                    (uncomment to modulate Pitch)
                    }
                }
            }
        }
        I2C_Array_TX[3] = Dist1;
    }

```

```

    }

void main() {
    hardinit();
    while(1) {

        if (I2C_Array[index_i2c++] == 0xFF) {
            adcread(); // Reading ADC
channels
            distM(); // Measuring
distance
            PWM1_Set_Duty(Dist1); // Set volume
            Tone1(); // Play Tone
            I2C_Array[index_i2c++] = 0x00;
        }
    }
}

void interrupt () {
    if (PIR1.SSP1IF) // check to see if
SSP interrupt
    {
        RC3_bit = 1;
        if (SSP1STAT.b2) // Master read (R_nW = 1)
        {
            //RC3_bit = 1;
            if (!SSP1STAT.b5) // Last byte was an address (D_nA
= 0)
            {
                SSP1BUF = I2C_Array[index_i2c++]; // load with value
from array
                SSP1CON1.CKP = 1; // Release CLK
            }
            if (SSP1STAT.b5) // Last byte was data (D_nA
= 1)
            {
                SSP1BUF = I2C_Array[index_i2c++]; // load with value
from array
                SSP1CON1.CKP = 1; // Release CLK
            }
        }
        if (!SSP1STAT.b2) // Master write (R_nW = 0)
        {
            if (!SSP1STAT.b5) // Last byte was an address (D_nA = 0)
            {
                first = 1; //last byte was address, next will be data
location
                junk = SSP1BUF; // read buffer to
clear BF
                SSP1CON1.CKP = 1; // Release CLK
            }
            if (SSP1STAT.b5) // Last byte was data (D_nA
= 1)
            {
                if (first)
                {

```

```

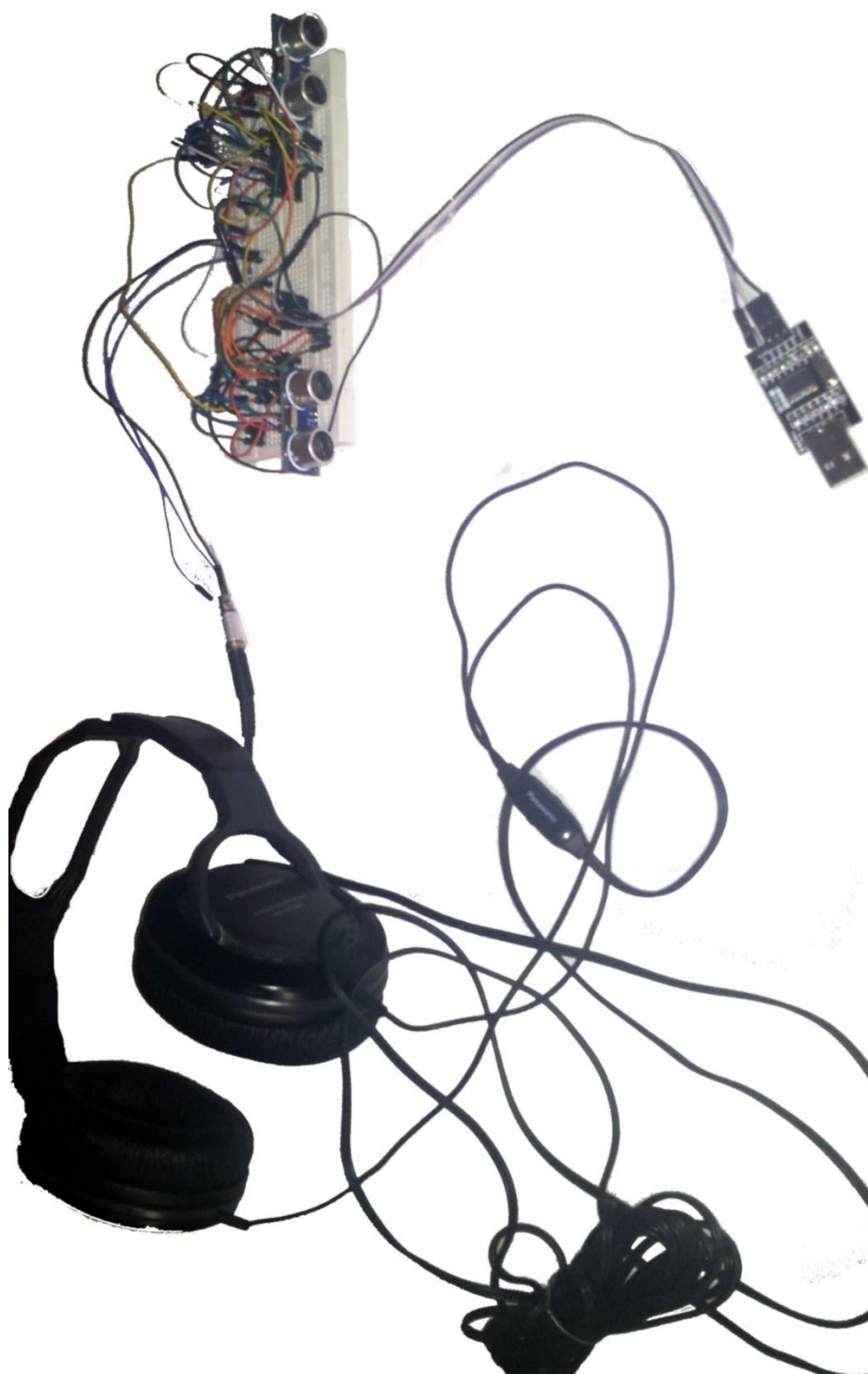
                                index_i2c = SSP1BUF;           // load index with array
location                                first = 0;           // now clear this since we
have                                }                           //location to read
from/write to                                //location to read

                                else
                                {
                                    if (index_i2c < RX_ELMNTS)           // make sure index
is not                                {                           //out of range of
array                                {                               //load array
with data                                I2C_Array[index_i2c++] = SSP1BUF;
                                }
                                else
                                {
                                    junk = SSP1BUF; //array location not valid,
discard data
                                }
                                }
                                if (SSP1CON1.WCOL)           // Did a write collision
occur?                                {
                                {
                                    SSP1CON1.WCOL = 0;           // clear WCOL
                                    junk = SSP1BUF;           // dummy read to clear
BF bit
                                }
                                SSP1CON1.CKP = 1;           // Release CLK
                                }
                                }
                                if (PIR2.BCL1IF)           // Did a bus
collision occur?
                                {
                                    junk = SSP1BUF;           // dummy read SSPBUF to
clear BF bit
                                    PIR2.BCL1IF = 0;           // clear bus
collision Int Flag bit
                                    SSP1CON1.CKP = 1;           // Release CLK
                                }
                                PIR1.SSP1IF = 0;           // clear SSPIF flag
bit
                                }

```

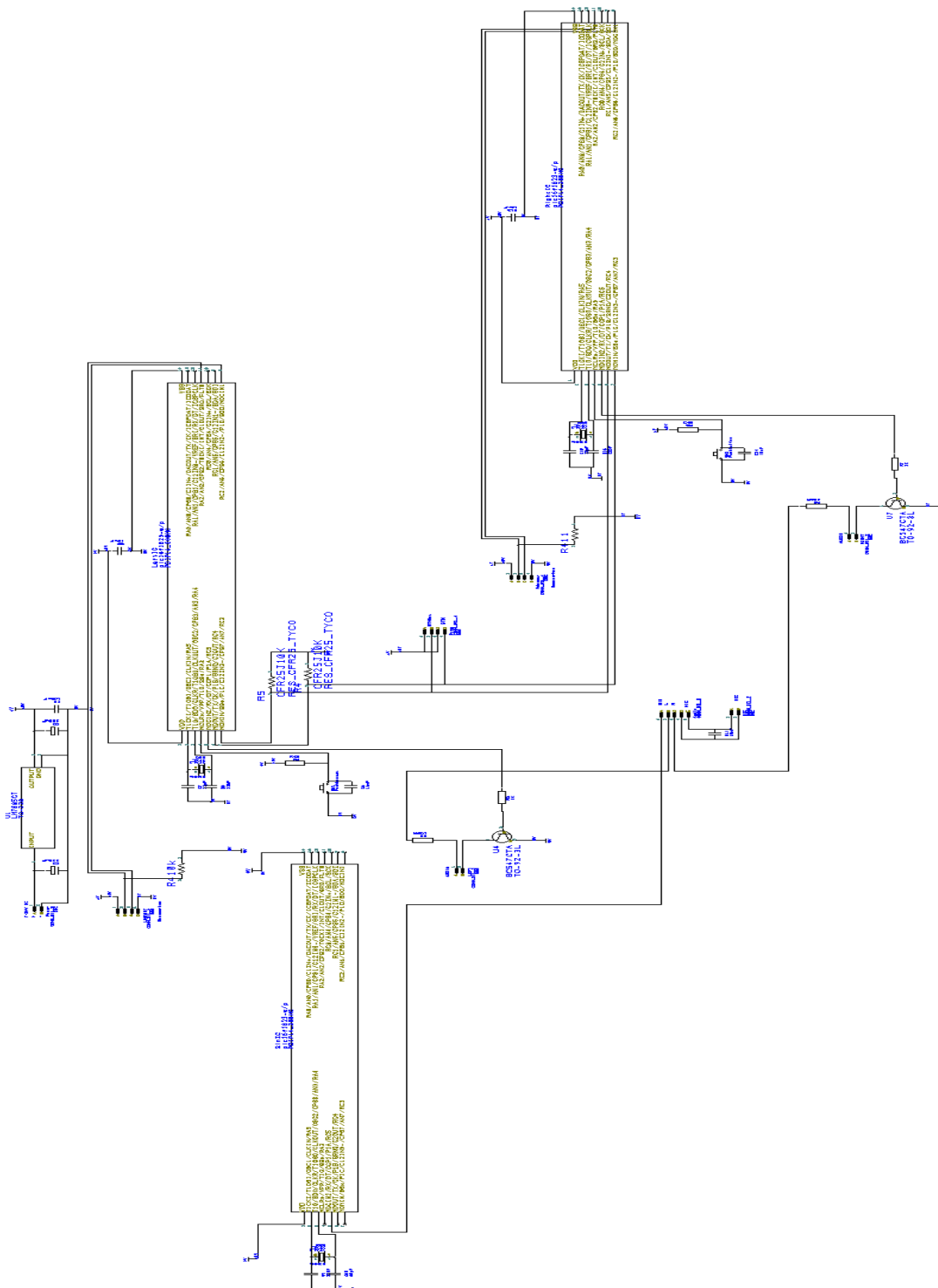
5. melléklet

A kísérleti eszköz fényképe



6. melléklet

A továbbfejlesztett eszköz kapcsolási rajza



7. melléklet

A kísérleti eszköz képe



8. melléklet

A továbbfejlesztett eszköz fő IC-jének programja

```
/*****
```

```
*      "BatEye"      *
```

```
*VAKOK ÉS GYENGÉNLTÓK *
```

```
* TÁJÉKOZÓDÁSÁT SEGÍTŐ *
```

```
* ESZKÖZ TERVEZÉSE, *
```

```
*      VIZSGÁLATA      *
```

```
* PIC16f1823 @ 32MHz *
```

```
* 8 MHz crystal, 4x PLL*
```

```
* 2014 Erdélyi Viktor *
```

```
*      MASTER      *
```

```
*****/
```

```
// 8 bit variables
```

```
typedef unsigned char uint8;
```

```
typedef unsigned int uint16;

// Flags

bit oldstate;

bit echoold;

bit BtnOldstate;

/* RC3 pin operating as function-flag*/

// Variables

uint8 Dist1; // Distance and azimuth variables

//uint8 PotX; // Necessary for 3D scanning

//uint8 PotY;

//uint8 Pitch; // Needed for Pitch modulation

uint8 Vol; // Volume

//uint16 ADCX; // Necessary for 3D scanning
```

```
//uint16 ADCY;

uint8 pingdel;                // Ping delay variable

uint8 i;                      // Variable for "for" cycle


// denominate I/O bits

sbit Echo at RC2_bit;

sbit Echo_d at TRISC2_bit;

sbit Trig at RA2_bit;

sbit Trig_d at TRISA2_bit;

sbit FFlag at RC3_bit;

sbit FFlag_d at TRISC3_bit;

sbit PngBtn at RC4_bit;

sbit PngBtn_d at TRISC4_bit;
```

```
/* Internal sound source subroutine*/

/*

void Tone1() {

    Sound_Play(400, 20);           // Frequency = 400Hz, duration = 20ms

    //Sound_Play(Pitch, 20);       // Uncomment to modulate pitch

}

*/

void hardinit(void) {

    //CPSCON0.CPSON = 0;           // Disable CPS module

    //OPTION_REG.b7 = 0;           // Disable weak pull up

    //WPUC = 0;

    ANSELA  = 0;                   // Configure AN pins as digital

    ANSEL0 = 0b00000011;          // Configure port C 0 and 1 bits as analogue

    CLON_bit = 0;                 // Disable comparators
```

```
C2ON_bit = 0;

APFCON = 0b10000100;           // Set APFCON register values

//OSCCON = 0b01111010;         // Internal OSC @ 16MHz ,if necessary

//OPTION_REG = 0b11010111;

//WDTCON = 0b00010111;         // prescaler 1:65536                      // per
iod is 2 sec (RESET value)

//PORTC = 0x00;                // Clear PORTC

//LATC = 0x00;                 // Clear PORTC latches

ADC_Init();

Echo_d = 1;                    // Init I/O ports

FFlag_d = 1;

Trig_d = 0;

Trig = 0;

PngBtn_d = 1;

// UART1_Init(9600);           // Initialize UART module at 9600 bps

Delay_ms(100);                 // Wait for UART module to stabilize
```

```
//Sound_Init(&PORTC, 4);           // Init sound

PWM1_Init(40000);                 // Initialize PWM1 module at 25KHz

PWM1_Start();                     // Start PWM1

oldstate = 0;                     // Init flags

echoold = 0;

//Dist1 = 0;                      // Initialize variables

//PotX = 0;

//PotY = 0;

//Pitch = 400;

}

/* Subroutine for 3D scanning (azimuth)*/

/*void adcread(void){              // Reading ADC values, converting into 8 bit format

    ADCX = ADC_Read(5);           // Reading 10-bit value from ADC

    ADCY = ADC_Read(4);
```

```
PotX = ADCX >> 2;           // Writeing significant 8 bit to variable

PotY = ADCY >> 2;

} */

/* Subroutine for distance measuring */
void distM(void) {

    Dist1 = 255;

    Vol = 255;

    pingdel = 0;

    echoold = 0;

    oldstate = 0;

    Trig = 1;

    oldstate = 1;
```

```
delay_us(11);

Trig = 0;

if (oldstate == 1){

    while (echoold == 0){

        if (Echo == 1){

            echoold = 1;

            while (Echo == 1) {

                delay_us(30);    // 5 mm precision    100 for 17 mm

                Dist1 --;

                Vol = Vol --;

                pingdel ++;

                //Pitch = Dist1*5+400;    // +17 mm => +5 Hz (uncomment to modulate Pitch)

                if (Vol <= 0) Vol = 0;

            }

        }

    }

}
```

```
        }  
    }  
}  
  
/* Sets ping delay in manual mode */  
void pingdelay(void) {  
    for(i=0; i<pingdel; i++) {  
        delay_ms(4);  
    }  
}  
  
void main() {  
    hardinit();
```

```
while(1) {

    /* System used without external ping button */

    while(RC3_bit) {                                     // When no external button connected

        //adcread();

        distM();                                         // Measuring distance

        PWM1_Set_Duty(Vol);                             // Set volume

        //Tone1();                                       // Uncomment to use internal sound sour

ce

        /* Data transmission to PC. -uncomment to use with PC*/

        //UART1_Write(PotX);

        //UART1_Write(PotY);

        //UART1_Write(Dist1);

        //UART1_Write('\n');

        //delay_ms(100);
```

```
    }

    /* System used with external ping button */

    while(!RC3_bit) {                                     // When external button connected

        if (Button(&PORTC, 4, 1, 1)) {                   // Detect logical one

            BtnOldstate = 1;                               // Update flag

        }

        if (oldstate && Button(&PORTC, 4, 1, 0)) {       // Detect one-to-zero transition

            BtnOldstate = 0;                               // Update flag

            PWM1_Set_Duty(255);                            // Set volume to simulate active ping

            delay_ms(100);

            PWM1_Set_Duty(0);

            distM();                                         // Measuring distance

            pingdelay();                                     // Simulating ping latency

            PWM1_Set_Duty(Vol);                             // Set Echo volume
```

```
        delay_ms(150);                                // Simulating echoed ping

        PWM1_Set_Duty(0);

    }

}

}
```

9. melléklet:

A „szinuszgenerátor” IC programja

```

/*****
*      "BatEye"      *
*TÁJÉKOZÓDÁSI STRATÉGIA*
*  ÉS IRÁNYÍTÓ ESZKÖZ  *
*    FEJLESZTÉSE      *
*LÁTÁSSÉRÜLTEK SZÁMÁRA *
*  PIC16f1823 @ 32MHz  *
*8 MHz crystal * 4x PLL*
* 2014 Erdélyi Viktor  *
*   Sinus generator    *
*****/

void hardinit(void) {
    CPSCON0.CPSON = 0;           // Disable CPS module
    OPTION_REG.b7 = 0;          // Disable weak pull up
    WPUC = 0;
    ANSELA = 0;                  // Configure AN pins as digital
    ANSEL = 0;
    C1ON_bit = 0;                // Disable comparators
    C2ON_bit = 0;
    APFCON = 0b10000100;
    PWM1_Init(41000);            // Initialize PWM1 module at 41KHz
    PWM1_Start();                // Start PWM1
}

void szinusz(void){
//400 Hz quasi-sinusoidal
    PWM1_Set_Duty(127);
    delay_us(250);
    PWM1_Set_Duty(188);
    delay_us(250);
    PWM1_Set_Duty(241);
    delay_us(250);
    PWM1_Set_Duty(254);
    delay_us(250);
    PWM1_Set_Duty(222);
    delay_us(250);
    PWM1_Set_Duty(158);
    delay_us(250);
}
```

```
        PWM1_Set_Duty(82);  
        delay_us(250);  
        PWM1_Set_Duty(23);  
        delay_us(250);  
        PWM1_Set_Duty(0);  
        delay_us(250);  
        PWM1_Set_Duty(21);  
        delay_us(250);  
        PWM1_Set_Duty(80);  
        delay_us(250);  
    }  
  
void main() {  
    hardinit();  
    while(1){  
        szinusz();  
    }  
}
```