# Diplomaterv

**Dolgozat címe**: Szájról olvasást segítő eszköz készítése siketek számára

Konzulensek neve: Takács György, Tihanyi Attila


Hallgató neve: Földi Balázs


**PÁZMÁNY PÉTER KATOLIKUS EGYETEM**

**INFORMÁCIÓS TECHNOLÓGIAI KAR**

# DIPLOMATERV-TÉMA BEJELENTÉS

Név: **Földi Balázs**

Tagozat: **nappali** Szak: **Műszaki Informatika**

Témavezető neve: **Dr. Takács György, Tihanyi Attila**

A dolgozat címe: **Szájról olvasást segítő eszköz készítése siketek számára**

A dolgozat témája:

Ismerje meg a siketek szájról olvasási képességét. Tanulmányozza a felhasználható taktilis információátviteli technikákat konferencia, folyóirat, cikkek és egyéb nemzetközi irodalom alapján. Válasszon ki egy megvalósítható hordozható, könnyen kezelhető rendszert, amelyet tervezzen és valósítson meg. A kialakított konstrukció legyen alkalmas legalább két taxel kezelésére. A taxeleket az éppen elhangzó beszédhang jellemzői vezéreljék, az egyiket a zöngésség a másikat a hang frikatív jellege befolyásolja. A kialakított konstrukció legyen alkalmas a teszteredmények és a további felhasználás során szerzett tapasztalatok alapján szükségessé váló módosítások rugalmas megvalósítására és a felhasználói igények szerinti egyedi kialakításra. A megvalósított rendszerrel végezzen használhatóságot igazoló teszteket.

A témavezetést vállalom:

.....
(a témavezető aláírása)

Kérem a diplomamunka témájának jóváhagyását.

Budapest, 2007. …….

.....
(a hallgató aláírása)

A diplomamunka-témát az Információs Technológiai Kar jóváhagyta.

Budapest, 2007. …….

.....
Nyékyné dr. Gaizler Judit
Dékán

A diplomatervet átvettem:

Budapest, 2007…….

.....
(a témavezető aláírása)

## Nyilatkozat a hallgató konzultáció rendszeres látogatásáról


# Pázmány Péter Katolikus Egyetem
## Információs Technológiai Kar
### Tanulmányi Osztály

1083 Budapest, Práter u. 50/A Tel/Fax: 886-4700 E-mail: tanulmanyio@itk.ppke.hu


Földi Balázs hallgató (neptunkódja: Z5W10Q) **témavezetője**
**bejelentem, hogy a diplomaterv**

**címe**: Szájról olvasást segítő eszköz készítése siketek számára

a diplomavédés időszakára (a beadási határidő: 2008. június 6.)
várhatóan

**elkészül** **nem készül el**

Budapest, 200

A konzulens
neve:……

_____

Konzulens aláírása

# Nyilatkozat az önálló munkáról

Alulírott Földi Balázs, a Pázmány Péter Katolikus Egyetem Információs Technológiai Karának hallgatója kijelentem, hogy ezt a diplomatervet meg nem engedett segítség nélkül, saját magam készítettem, és a diplomamunkában csak a megadott forrásokat használtam fel. Minden olyan részt, melyet szó szerint, vagy azonos értelemben, de átfogalmazva más forrásból átvettem, egyértelműen a forrás megadásával megjelöltem. Ezt a Diplomamunkát más szakon még nem nyújtottam be.

# Tartalomjegyzék

# Ábrajegyzék

# Összefoglaló

Alapvető emberi tulajdonság hogy képesek vagyunk önzetlenül segítséget nyújtani embertársainknak, különösen erős ez a jó szándék ha környezetünk női, gyermek, idős vagy betegségben szenvedő résztevőiről van szó. A siketséggel együtt élő betegek társadalmunk olyan jelenlévő szereplői, akik a természetes emberi beszéd folyamatából akár teljes mértékben is kimaradhatnak.

Ezen szakdolgozat témája egy olyan eszköz kifejlesztése, amely segítséget nyújt a hallássérültek számára a szájról való olvasásban. Ugyanis az emberi kommunikáció legfőbb eszközét a beszédet csak némileg lehet szájról olvasva megérteni. A halláskárosultak illetve siketek saját hibájukon kívüli hiányosság miatt, szélsőséges esetben akár teljesen is kieshetnek a kommunikáció eme alapvető és fontos formájának a megértéséből. Erre a problémára az új információs technológiai kutatások, fejlesztések hívatottak segítséget nyújtani a már meglévő technológiák alkalmazása mellett vagy akár helyettesítve is azokat.

A felhasznált információátviteli technikák közül a szájról olvasás és a jelbeszéd együttes alkalmazása a legmegfelelőbb eszköz a siketek számára az élőbeszéd közben fellépő kommunikációs hátrányok leküzdésére. A legtöbb megértési problémát az a gyakori eset okozza, amikor egy halláskárosult ember próbálja megérteni egy a jelbeszédet nem ismerő, ép hallású ember beszédét. Sajnálatos módon a siketek által használt speciális jelbeszédet az átlagember legfeljebb felismerni, de megérteni vagy alkalmazni képtelen. A szájról olvasásra támaszkodhat ilyenkor a halláskárosult, ami gyakran nehézségekbe ütközhet. Példának okáért a zöngés és zöngétlen hangok megkülönböztetése szájról olvasva nehézségekbe ütközhet. A hangszalagok rezgése beszéd közben kívülről nézve láthatatlan, így például a baba vagy a papa szó között szájról olvasva nincs különbség. Ezeket a fajta jellemzőket, amelyeket szájról olvasva képtelenség megkülönböztetni az élő beszéd hangjeleiből egyértelműen kinyerhetőek. Mi módon a felhasználó a segédeszköz használata közben a beszédpartnere száját figyeli ezért egy, a tekintet nem elvonó kijelzőt terveztem, ami a vizuális információ helyett mechanikus, tapintási információt közöl a használóval. Az ilyen fajta eszközöket, amelyek a tapintó szerveken keresztül képesek információt közölni a felhasználóval nevezik taktilis kijelzőnek.

Többféle taktilis kijelző létezik már, azonban siketek számára segédeszközként való felhasználása, valamint dinamikus, gyors beszédhang információk kijelzése teljesen új jellegű hasznosítás ezen a szakterületen. Az eszköz tervezése során figyelembe vettem a tapintásérzékelés jellemzőit, az emberi beszédképzés tulajdonságait, hogy az eszköz

információátadása jól érzékelhető legyen, valamint a siketek számára valóban hasznos információt közöljön a siket felhasználókkal.

Felépítését tekintve az általam tervezett rendszer prototípusa a következő részekből áll. Egy mikrofon veszi a hangjeleket, amit egy mikrokontroller processzora feldolgozza és továbbítja a taktilis kijelzőnek, hogy azt érzékelni lehessen. További fejlesztéseket végeztem az egyszerű és gyors tesztelhetőség érdekében, úgy hogy az eszközt PC-ről is vezérelni lehessen, valamint az automatikusan felismerni és kezelni legyen azt képes. Így már egy valóban sokoldalú és többféle képen hasznosítható eszköz prototípusára tettem szert.

# Bevezetés

A kitűzött célom, hogy egy taktilis segédeszközt készítsek el siketek számára, amelynek segítségével szájról leolvashatatlan vagy nehezen felismerhető beszédhang információkat lehetséges közölni a hallássérült felhasználóval, enyhítve ez által a kommunikációs nehézségeiket. A siketek szájról olvasási képességének tanulmányozása közben arra a felismerésre jutottam, hogy még gyakorlottabbak is nehézségekbe ütközhetnek, ugyanis szájról olvasva nem kapható teljes beszédinformáció. A feladatom, hogy erre a problémára adjak megoldást a korszerű információs technológiai vívmányok segítségével.

Az eddig létező taktilis kijelzők típusok elemzése után kiválasztottam azt a típust, amelyik egy gyors, dinamikus hangjel kijelzésére legalkalmasabbnak ítéltem. Ezeknek a feltételeknek a legjobban az elektromágneses stimulátorok felelnek meg a legjobban. A feladatkiírásban szereplő két taxel külön kezelésére, egy-egy elektromágneses stimulátort használok. A taxel a legkisebb érzékelhető tapintási egység.

A taktilis kijelző további tulajdonságainak meghatározásához a neurobiológia tapintásérzékelés részterületét vizsgáltam meg. Definiáltam a két kijelző távolságát olyan szempont alapján, hogy kézfejen, csuklón jól megkülönböztethetőek legyenek. Az elektromágneses stimulátor rezgési frekvenciáját 200 Hz-en határoztam meg, mert ezt az ingerlést lehet a legjobban érzékelni. A frekvencián kívül meg kellett állapítanom azt a legkisebb elmozdulást, amit a taktilis kijelzőn még érezni lehet. A szakirodalom ezt 0,17 milliméter értékben adja meg.

Tanulmányoztam az emberi beszédkeltés szakterületét. Ezek alapján meghatároztam azokat a hangjellemzőket, amit szájról olvasva lehetetlen, azonban hangjellemzők alapján elektronikus berndezéssel könnyedén meg lehet különböztetni.

Az eddig említett háttéranyagok alapján megalkottam a taktilis kijelzőt. Szükségem volt egy elektronikus vezérlő berendezésre, ami a taktilis kijelzőt meg tudja hajtani. Egy olyan fejlesztői eszköz, mikrokontroller használata mellett döntöttem, ami a kijelző vezérlésére mellett magába foglalja a takarékos működés feltételeit. A gazdaságos üzemmód ahhoz szükséges, hogy a segédeszközt a felhasználó önállóan, hordozható formában, telepesen működtetve azt használni tudja.

A személyi számítógépről való vezérlés érdekében kidolgoztam az eszközhöz egy USB protokollt. Így már egyszerű parancsokon keresztül lehetséges a segédeszközt számítógépről is irányítani, például a kijelző tüskéit ki-bekapcsolni.

Ezt követően azt valósítottam meg, hogy a már meglévő rendszert felruházzam egy olyan tulajdonsággal, aminek segítségével a személyi számítógép automatikusan felismeri és kezelni tudja azt. A konstrukció mivel USB kapcsolatra alkalmas, ezért annak lehetőségét vizsgáltam meg, hogy ezzel a fajta csatlakozóval PC-re kapcsolódva az operációs rendszer hogyan képes felismerni egy eszközt és vezérelni azt. A segédeszköz további fejlesztését, mint egy új USB eszköz fejlesztését folytattam tovább, ugyanis az USB eszközöket a mai modern operációs rendszerek automatikusan felismerni és kezelni képesek. A segédeszköz innentől fogva kettős szerepre alkalmas, egyrészt önálló telepes működésre, másrészt egy rugalmasan fejleszthető rendszer.

Az eddigi konstrukció alapján megterveztem, a fejlesztési mintát. A feleslegessé váló részek lehagyása és az elektromágneses stimulátor egy kisebb és lényegesen könnyebb változatát használva kétségtelen méret és súlycsökkentést értem el. Mindezek eredményéül egy karóraszerű viselésre alkalmas eszközt kaptam.

A fejlesztési minta segítségével teszteket végeztem. A kijelző által gerjesztett mechanikus jel tisztán érzékelhető, a két kijelző különböző jele jól elkülönül. Az emberi beszéd zöngés és frikatív hangjellemzőit az eszköz helyesen detektálja.

# 1. Siketek – szájról olvasás

Általában véve a „hallássérültek” megjelölés egy gyűjtőfogalmat takar, amely magában foglalja az enyhe, a közepes és a súlyos fokú nagyothallókat, valamint a siketeket. Siketeknek azokat a nagyothalló embereket nevezzük, akiknek a hétköznapi életben használható hallással nem rendelkeznek, audiológiai és gyógypedagógiai értelemben ez 100 dB-nél súlyosabb halláskárosodást jelent. Az egész hallássérült népesség mintegy 5%-át kitevő siketek, a nagyon mély hangokat és vibrációkat érzékeli, amin kívül esik az emberi beszéd. Éppen ezért kommunikációjukat nagyobbrészt a vizuális információk felvétele jellemzi, történetesen a szokványosokon kívül a jelnyelv és a szájról olvasás.

A szájról olvasás a beszéd megértését jeleni a beszélő beszédszerveinek mozgásának és az arckifejezésének vizuális megfigyelése alapján. Ezzel a tulajdonsággal minden ember természetes módon rendelkezik valamilyen fokon, a továbbiakban viszont a siketek által használt és tanult szájról való olvasási képességéről fogunk tárgyalni, ami sok gyakorlással egészen kifinomult szintig tud eljutni.

Ez a tanulható képesség egészen kifinomulttá tud válni, azonban a szájról olvasás során a nagyothalló rakja össze mozaikszerűen a képi hatásokat, ugyanis a kívülről látható beszédszervek a nyelv, az ajkak és az állkapocs csak az elhangzottak egy részének értelmezését teszik lehetővé. A hangképzés bizonyos folyamatai láthatatlanok maradnak ezen a módon, így egyes hangok alkotása nem ad vizuális információt, vagy az alkotott képek felcserélhetőek egymással, esetenként a megelőző vagy rákövetkező hangok módosíthatják a képi információt, ez nagymértékben akadályozhatja a megértést. Helen Keller amerikai írónő – aki halláskárosult volt és csaknem vak – fogalmazta meg a következő gondolatot: „Aki elveszíti a látását, az a tárgyakkal való kapcsolatot veszíti el, aki a hallását, az viszont az emberekkel való kapcsolatát veszíti el.” Vagyis a halláskárosultakat a kommunikációs képesség terén szenvedik el a legnagyobb veszteséget.

További akadálya lehet a szájról olvasásnak a rossz megvilágítási körülmények, ugyanis elengedhetetlen a megfelelő fényviszonyok hogy a beszélő arca jól látható legyen. A beszélő alkalmatlan elhelyezkedése is nehezíti a megértést, ajánlatos közel és szembe állni a szájról olvasóval. A gyors, elhadart beszédet az előbbiekben említett mozaikkirakás szerű jelleg miatt képtelenek a siketek követni. Még a normális beszédtempó mellett is koncentrált

figyelmet kíván a halláskárosult részéről a szájról olvasás, ami hosszú ideig igencsak fárasztó lehet.

A beszéd megértésén kívül, a beszéd képzése is problémát okozhat a siketek számára. A probléma forrása, hogy a hallássérült személy alig vagy nem is hallja saját beszédét. Egészében jellemző a lassú, rossz hangsúlyozású, furcsa ritmusú, különös hangszínű beszéd valamint hiányozhatnak bizonyos beszédhangok. A nagyobb mértékű hallásveszteség, nagyobb mértékű helytelen beszédképzést jelenthet, különösen az új kifejezések tanulása esetében jelentős ez, ami szókincsbeli hátrányokat okoz. A fiatal és gyermekkorú hallássérültek oktatása külön odafigyelést igényel az oktatók részéről.

Az itt felsorolt problémák döntő többségének leküzdésére, enyhítésére alkalmasak az információs technológiai fejlesztések. Ezek közül egy alkalmazható megoldás az automatikus fordítás jelnyelvre, amit már megoldottak cseh nyelvre ld.: Jakub Kanis és Ludek Müller: Automatikus cseh- jelbeszédfordítás című konferencia beszámolóját [16]. Azonban én egy másik megoldást választottam, ugyanis a beszédhang jeléből kinyerhetőek azok a jellemzők, amelyek a szájról leolvashatatlan információkat jellemzik. Az ilyen pillanatnyi információk birtokában jelentősen megnövekszik a siketek kommunikációs képessége. A következő feladat, hogy egy ilyen gyorsan változó információt, jelet hogyan adjunk át a hallássérült felhasználó számára érthető és hasznosítható módon.

Erre a feladatra kielégítő módszer például a felvillanó fény használata. Egy LED (fénykibocsátó dióda) képes ilyen gyors jelzésre, a gyakorlatban ez a beszéd megfelelő jellemzőire villogó fénykibocsátást jelent. A szájról olvasó felhasználó a beszélő arcát, száját figyeli, ezért alkalmatlan egy tekintetet elvonó alkalmazás. Az alapvető érzékelés szintjén a tapintás tűnik a legjobb megoldásnak. A gyors villogás helyett ebben az esetben egy gyorsan rezgő, tapintási információ továbbítása kell hogy biztosítja a szükséges többletinformációt. A mobiltelefonok rezgő hívásjel funkciója kiváló példa a rezgési információ hétköznapi felhasználására. A szerkezet használata közben a felhasználó nem használja a kezét, de hogy az esetleges jelbeszédet ne zavarja a berendezés ezért a segédeszköz karóraszerűen viselhető formában kell megvalósítani.

# 2. A tapintásérzékelés

Azt már megállapítottam, hogy egy tekintetet nem elvonó tapintási információt továbbító szerkezetet fogok létrehozni. Ilyenforma okokból kifolyólag szükséges a neurobiológia tapintásérzékelés részterületét megvizsgálni. Egy taktilis eszköz fejlesztési folyamata során nélkülözhetetlen már az első lépésekben az emberi tapintásérzékelés tanulmányozása. Ebben a fejezetben vizsgálom meg, hogy hogyan működik a tapintásérzékelés, mik a jellemzői és ehhez mérten milyen is a taktilis kijelzők optimális paraméterei.

## *2.1. Mi a tapintás?*

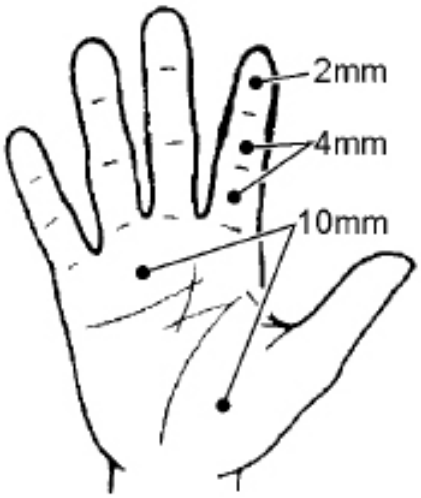
Az észlelés, érzékelés egyik módja a tapintás, amely a bőr mechanikai megzavarásával jön létre. A zavar milyensége függ a tárgy fizikai tulajdonságaitól. Ennek segítségével érzékeljük a tárgyak alakját, méretét, súlyát, textúráját és összenyomhatóságát. A tapintás érzékszerve a bőr, ami a kezünkön a legérzékenyebb. Bár nem mindig közvetlen érintkezéssel szerzünk információt a környezetből. Például egy bottal is megállapíthatjuk egy tárgy bizonyos jellemzőit. A tapintás a többi érzékszervhez képest a legmegbízhatóbb. Bizonytalan esetekben az észlelési döntőbíró. A tapintás legfontosabb szerepe a tárgyak azonosítása, emellett több szociális és pszichológiai okokból is nagyon jelentős a fejlődés és társas kapcsolatok terén. Fejlődéslélektan szempontjából az anya-gyermek kapcsolatban kitüntetett szerepe van az érintkezésnek. Társas kommunikáció terén például a kézfogás.

## *2.2. A kéz*

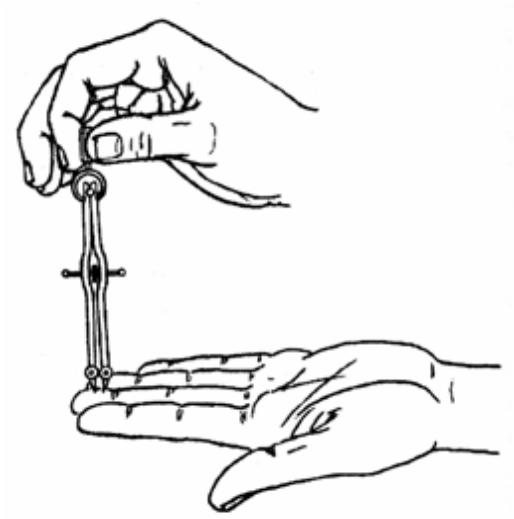
A tapintásban kitüntetett szerepe van a kéznek, hiszen az ujjak végén a legérzékenyebb a bőr. Itt a legnagyobb a mechanoreceptorok száma, és finommozgásokat végző izmok összetett csoportja. A mechanoreceptorok a finomabb érzékelést végzik. Míg az izmok a durvább érzékelést végzik, például súly megállapítása. A kettő együtt és a kéz mozgathatósága pontos képet ad egy konkrét tárgyról. Ezek alapján elmondhatjuk, hogy a kéz a tapintás a legfőbb szerve.

## 2.3. A tapintás érzékenysége, pontossága

Azt már megállapítottam, hogy az egész bőrfelületen, a kéz a legérzékenyebb és legfontosabb rész a tapintás szempontjából. Azonban szükség van arra, hogy ezt számszerűleg is mérni lehessen. Ilyen módszer többek között a kétpont-küszöb eljárás. Ennek segítségével lehet megállapítani, hogy egy adott bőrfelület milyen távolságú pontokat képes megkülönbözetni a kísérleti alany. Ezt úgy végzik, hogy két tűt közelítenek egymáshoz a bőrfelületen, egészen addig, míg csak egy pontnak érezhető a kettő. Azt a legkisebb távolság, amivel még érezni lehet a két tű hegyét, hívják kétpont-küszöb távolságnak. Ennek a módszernek a segítségével feltérképezték a bőrfelület számos területét. Az ujjbegy vizsgálata során kiderült hogy a 2 milliméteres távolság is érzékelhető, míg a háton a kétpont-küszöb távolság 70 milliméter. A taktilis kijelző szempontjából a kézfej a legfontosabb. Az ujjbegyen 2 milliméter, az alsó két ujjpercen 4 milliméter, a tenyéren 10 milliméter a kétpont-küszöb távolság.




**1. ábra: a kétpont-küszöb távolságok mérése**



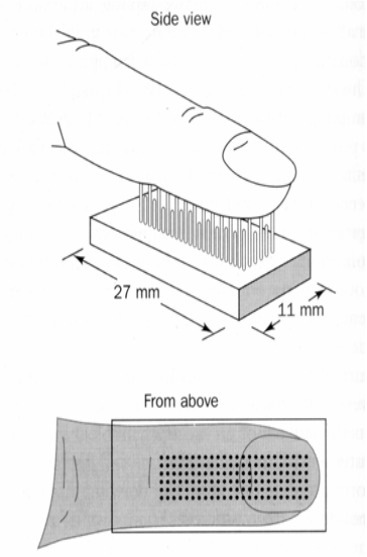
**2. ábra: a kétpont-küszöb távolság**

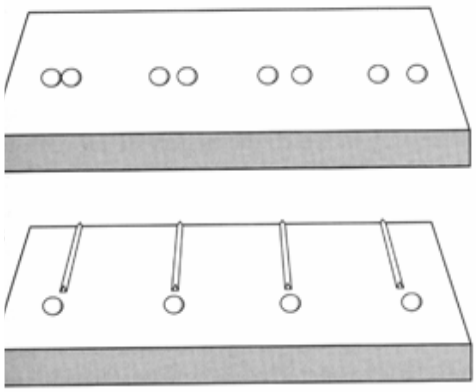
## 2.4. A relatív helyzet érzékelése

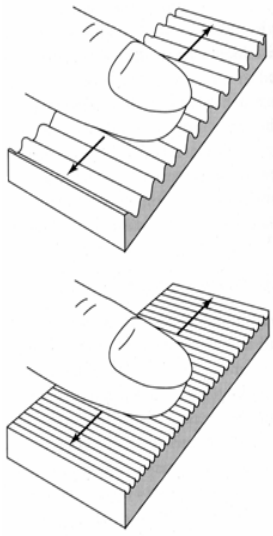
A relatív helyzet érzékelése sokkal pontosabb, mint két pont megkülönböztetése. A legkisebb elmozdulás, amit érzékelni lehet 0,17 milliméter. Ez töredéke a 2 milliméternek, ami annak köszönhető, hogy két pont megkülönböztetéséhez több mechanoreceptor szükséges, mint az elmozdulás megkülönböztetéséhez.

## 2.5. A tapintási inger

Egy merev pálcával rezgést okoznak a bőrön és figyelik, milyen frekvenciát hogyan érzékelünk. A legerőteljesebb a körülbelül 200 Hz-es ingerlés. A kis méretű pálcát jobban lehet érzékelni, mint a nagyobb pálcát kis frekvenciákon. A kísérletek szerint rezgésre a tenyér a legérzékenyebb és nem az ujjbegy.





**3. ábra: az OPTACON rendszer** **4. ábra: elmozdulás érzékenység mérése** **5. ábra: felszíni textúra**

## 2.6. A felszíni textúra érzékelése

A felszín eme fizikai jellemzőinek a felismerésére is képesek vagyunk. Egy mesterséges, szimmetrikusan változó, rácsmintázat finom ujjmozgás segítségével érzékelhető. A 0,5 milliméteres rácstávolság már érzékelhető, de persze ez függ a rács mélységétől és a

kézmozgatás sebességétől. Megfigyelték továbbá, hogy a bőr rugalmassága is számít. Így a mutató-, középső- és gyűrűs ujj érzékenysége is változik.

## 2.7. A beállítódás és tapintási érzékenység

Ha az inger forrása bizonytalan, akkor az érzékelés pontatlanabb. Ezt Craig egy az OPTACON-nál használt kijelzőhöz hasonló 6-szor 18-as tűsorozat segítségével bizonyította a mutató és középső ujjakra. Az OPTACON (OPtical to TActile CONverter) rendszert vakok számára 1970-es években kifejlesztett Braille írást érzékelhetővé tevő készülék (ld. Bővebben 4.1 fejezet). Ha előre megmondták, hogy melyik ujjra megy az inger könnyebb volt az érzékelés. Később Kapta megállapította, hogy csak az egyik ujjra kell figyelni, az tovább könnyíti a feladatot. Vagyis a kijelzőt úgy lenne érdemes megvalósítani, hogy csak egy ujjat kelljen használni sok tűhöz. Valamint ezek a feladatok gyakorlással javíthatók, így a gyakorlat azt mutatja, hogy a felhasználók meg tudják szokni egy bonyolultnak tűnő kijelző használatát is. Egy, az OPTACON-hoz hasonló berendezés megalkotása jelenti a megoldást a segédeszköz létrehozásában.

## 2.8. A tapintás anatómiája

Ezt a részt a kézből indulva fogom bemutatni, hiszen a tapintás legfőbb szerve a kéz. Két ideg található a kézben, amely a tapintással kapcsolatos. A medianus ideg a kar közepén kettéágazik és a hüvelyk-mutató és középső-gyűrűs ujj egyik felét, valamint a hozzá tartozó tenyérrészt idegzi be. Az ulnaris ideg, pedig a tenyér többi részéért, a kisujjért és a gyűrűsujj maradék feléért felelős. Receptív mezőnek nevezzük azt a bőrfelületet, amelynek az ingerlése befolyásolja egy rost aktiválását. Ezek a rostok lehetnek idői és téri jegyek. Idői jegyek lassan és gyorsan adaptálódó rostok. Az egész nyomás alatt tüzelnek a lassan adaptálódó rostok, az érintkezés kezdetén és befejésekor válaszolnak a gyorsan tüzelő rostok. Az utóbbinak a jelentősége új információkor van, az előbbi, ha egy inger már régóta fennáll. A téri rostok két fajtája a pontrostok és a diffúz rostok. A pontrostok kis, éles határvonalú, ovális alakú receptív mezők és 4-10 bőrbarázdát fog át. A diffúz rostok nagy receptív mezők, életlen határvonallal rendelkeznek és néha az egész ujj vagy a tenyér nagyobb részéről szállítanak információt.

## 2.9. A mechanoreceptorok

Elmondható, hogy minden afferens rost egy mechanoreceptorral áll összeköttetésben. Például a Meissner-testek, Merkel-korongok, Ruffini-végződések és Pacini-testek valamint a szabad idegvégződések. A Meissner-testek a bőr felső rétegében helyezkednek el, ez egy tokos receptorfajta. Egy kis szemölcsbe vannak beágyazva, ezek a kis szemölcsök alkotják a tenyér és ujjbegy barázdáit. A felszínre merőleges nyomást érzékelik, vagyis egy kis részről, pontról adnak információt. Öregedéssel számuk csökken. A Merkel-korongok nem tokos receptorok, a középső bőrrétegben helyezkednek el ötös-tízes csoportokban. Lassan adaptálódó pontrostok szállítják az általa felvett információt. Ezek a legérzékenyebb mechanoreceptorok. A Ruffini-végződések szintén nem tokos receptorok és szintén a középső rétegben helyezkednek el. Hosszúkás alakúak, a bőr felszínével párhuzamosak, vagyis nagyobb bőrterületről szállítanak információt. Akkor aktiválódnak, ha az ujjak mozognak és megfeszül a bőr. A Pacini-testek tokosak és a bőr alsó rétegében helyezkednek el. Méretük a legnagyobb, legkevesebb van belőlük. Nagy bőrfelszíni terület érzékeléséért felelősek és szerfelett érzékenyek az érintésre. A szabad idegvégződések hajszálvékony hálózatot alkotnak a bőr minden rétegében. Szőrtüszőnél is előfordul, ezáltal a szőrszál elhajlását is értékelhetővé teszi.

## 2.10. Az érzőkéreg

Az érzőkéreg, másnéven szomatoszenzoros kéreg. Két fő területe van az elsődleges és másodlagos érzőkéreg. A thalamusból kapják a bemenetet, a másodlagos az elsődlegesből is. Ellenoldali feldolgozás a jellemző. Térképszerű leképezés van jelen, az érzékenyebb részek nagyobb mértékben vannak jelen.

# 3. Emberi beszédkeltés és beszédérzékelés

Felmerül a kérdés, mit is jelöljenek az egyes tüskék a taktilis kijelzőn, mik azok a hangok hangcsoportok, amit szájról olvasva nehezen, vagy talán nem is lehet megkülönböztetni. Az előző fejezetben már láttuk, hogy az OPTACON rendszerhez hasonló felépítésű konstrukció a céljainknak megfelelő a segédeszköz, csupán kevesebb tüske van rajta, a két taxelnek megfelelően kettő darab. Ennek a kérdésnek a megválaszolása érdekében további háttéranyag elemzést végeztem, tanulmányoztam az emberi beszédhangok csoportjait és képzését.

## *3.1. A nyelv*

A nyelv az emberi kommunikáció és az emberi gondolkodás legfőbb eszköze. A kommunikáció a normális társadalmi élet és a munkamegosztás alapvető feltétele. Ettől függetlenül a nem nyelvi eszközök is része a kommunikációnak. A nyelv egy jelrendszer, amelynek elemeihez egy nyelvközösségen belül ugyanaz a jelenség tartozik. A beszéd a nyelv elsődleges megnyilvánulása az írás csak másodlagos. A gondolkozáshoz közvetlen közel áll.

## *3.2. A természetes beszédlánc*

A beszédlánc mindig különböző egyedek közt megy végbe. Áll egy beszélőből, valamilyen átvivő közegből, ez általában a levegőközeg és egy felfogóból valamint több szintből. Ezek a szintek az agyi szint, szervi szint és akusztikai szint. Az agyban megfogalmazott mondanivalót, a beszédszervek segítségével közli azt a beszélő a másik félnek. A beszélő három visszacsatoláson keresztül is kap visszajelzést a saját beszédjéről. Egyrészt érzi saját beszédszerveit beszélgetés közben, másrészt hallja is a saját hangját és a hallgató reakcióiból is nyer információt.

## *3.3. A beszédhang*

Beszédhangnak nevezzük a legkisebb olyan egységet, amelynek sorozatával egy nyelvet megvalósító beszéd akármilyen részlete az agy számára reprodukálható. A

beszédhangok a beszéd olyan szegmensei, részletei, amelyeket a nyelvet beszélő egymástól elkülöníteni és felismerni teljes biztonsággal képes. A beszédhangok különbözősége nem mindig jelentenek különböző jelentést. A beszédhangok a nyelvre jellemzőek, nyelvenként más és más. Egyes nyelvekben még a hangmagasság hajlítása is megkülönböztet beszédhangokat, ezeket nevezzük tonális nyelveknek. A nyelvcsoportok három szinten különböznek. A *szótári szint* nagyon változékony, a szavak gyakran kikopnak, vagy újak jönnek. A magyar nyelv mindössze már csak kevesebb, mint kétszáz finnugor szót tartalmaz. Az idegen szavak mindig a saját hangrendszer szerint kerül használatba, hogy jobban illeszkedjen. A magyar nyelvragozó nyelv, ezért *nyelvtani szint*en nagyon eltér az indogermán nyelvcsalád tagjaitól. A finnek hanghordozása hasonlít a magyarhoz. Jóllehet egyetlen mondatot sem tudunk azonos jelentéssel elmondani, mégis a *hangok és hangkapcsolódások szintje* megegyezik a finnekével. Az élőbeszéd olyan leírása, amely a beszéd hangzásának leírására törekszik – a fonetikai átírás. Ennek elterjedt rendszerei az **APhI** és a **SAMPA.**

A beszéd egymástól megkülönböztethető elemek szervezett időbeni egymásutánisága – soros szerkezet. Elem lehet egy összefüggő mondanivaló, egy hosszabb szünetekkel elhatárolt beszédrész, egy mondat, egy szó, egy beszédhang. Egy ötven beszédhangból álló nyelvben (leszámítva, hogy nem minden hang mondható egymás után) kb. egymillió különböző tíz hangból álló szó képezhető. A beszéd szerkezete felülről gyakorlatilag nyitott, alulról zárt.

Egy nyelv fonémakészlete elemek olyan minimális számosságú halmaza, amelyből minden szó jelentéshelyesen, de csak egyféleképpen állítható elő. A fonémakészlet elemei a fonémák. Az azonos fonémákat képviselő beszédhangok az allofonok. Például a szeg vagy szög esetén az „e” és „ö” azonos fonémák. Azonban az „e” és „ö” mégis különböző fonémák, mert létezik olyan szó, amelyikre már nem ad azonos jelentést cseréjük.

## 3.4. A beszédhangok csoportosítása

A beszédhangokat a következő képen lehet osztályozni:

1. magánhangzók: (pl.: i, é, ü, ö, e, á, a, o, u)
2. mássalhangzók
   - -zárhangok
     - nazálisok (pl.: m, n, ny, ng)
     - felpattanó zárhangok
       - zöngés (pl.: b, d, g, gy)
       - zöngétlen (pl.: p, t, k, ty)

-pergőhangok (pl.: r)

-oldalréshangok (pl.: l, j)

-réshangok (frikatív hangok)

zöngés (pl.: v, z, zs)

zöngétlen (pl.: f, s, sz, h)

-afrikáták (zárréshangok)

zöngés (pl.: dz, dzs)

zöngétlen (pl.: c, cs)

A zöngés és zöngétlen hangok közti különbséget a kvázi periodikus gerjesztés megléte adja. Zöngés hangoknál van kvázi periodikus gerjesztés. A zöngés hangok tipikus frekvenciatartománya a 200 – 300 Hz közötti érték. A zöngésség egy szájról nem leolvasható tulajdonság ezért ez egy olyan tulajdonság, amivel érdemes foglalkozni a kijelzőnél.

Az afrikáták, zárréshangok tulajdonsága az, hogy lökéshullám gerjesztés és turbulens gerjesztés is van. Képzésükkor zár képződik, ami réssé alakul át. Ezek a hangok a *dz, dzs, c* és *cs.*

A felpattanó zárhangok a *b, d, g, gy, p, t, k, ty*. A toldalékcsőben létrejövő zárképződéssel és annak felpattanásával jönnek létre. Lehetnek zöngések vagy zöngétlenek. Képzésükkor van lökéshullám gerjesztés, viszont nincs turbulens gerjesztés.

A réshangok más néven frikatív hangok a következők *v, z, zs, f, sz, s, h*. Kiejtésükkor rés képződik. Szintén lehetnek zöngések vagy zöngétlenek. Réshangok esetén van turbulens gerjesztés és nincs lökéshullám gerjesztés és 1000 és 3500 Hz közötti frekvenciatartományról beszélhetünk.

## 3.5. A hangképzés, hangképző szervek

A légzőrendszer három fő részből áll: a tüdő, a légutak rendszere és a rekesz-, bordaközi izmok. A tüdő a mellüregben található. Előröl a bordák, hátulról a gerincoszlop, alulról pedig a rekeszizom határolja a mellüreget. A légutak fő feladata, hogy a levegőt a tüdőbe be vagy ki vezesse. Ez tulajdonképpen egy elágazó csőrendszer, a levegő a következő módon jut el a tüdőbe: szájüreg és/vagy orrüreg, garat, gége, légcső, hörgők. A rekesz-, bordaközi izmok beszívják és kiszorítják a levegőt a tüdőből. A légzőrendszer által létrehozott levegőáram nélkülözhetetlen a beszédképzésben is.

A toldalékcső, más néven artikulációs csatorna, vagy hangképző üregrendszer a beszédszerveknek a hangréstől az ajkakig terjedő szakasza. A gégéből kiáramló levegő a

hangképző üregrendszerbe, egy aránylag hosszú, bonyolult formájú, de alapvetően cső jellegű térbe jut, amelyben az útja a garat és szájüregben és orrüregen keresztül vezet.

## 3.6. Mely beszédjellemzőket érdemes kijelezni?

Szájról olvasva nehezen, vagy egyáltalán nem is lehet megkülönböztetni az emberi hangképzés két jellemzőjét. Ezek a beszédjellemzők zöngésség és a frikativitás. Vagyis a beszédhang ennél a két típusnál vizuálisan nem mindig határozható meg egyértelműen, viszont elég jelentősek a megértés szempontjából, ezért ezt a két beszédjellemzőt kívánatos a taktilis kijelzővel megjeleníteni a siketek és halláskárosultak számára. Azonban ez a két információ nem magától értetődő a siketek számára, ezért értelmezésükhöz, a segédeszköz helyénvaló alkalmazásához szükséges a kijelző használatának előzetes gyakorlása.

## 3.7. A szűrők használata

A zöngés és frikatív hangok detekcióra számos módszer létezik. A szűrőket a real-time valamint a telepes használat érdekében alkalmazom, hogy ne legyen szükséges egy PC az eszköz működtetéséhez. A két jellemzőt a 0-5000 Hz közötti tartományba eső beszédjelből kell kiszűrni.

A zöngésség jellemző frekvenciatartománya a 200 – 300 Hz közötti érték. Ennek következtében 400 Hz-es alul-áteresztő szűrővel lehetséges a zöngés hangokat kiszűrni. A frikatív hangok esetén 1000 és 3500 Hz közötti frekvenciatartományról beszélhetünk, ebben az esetben egy 2,5 kHz-es felül-áteresztő szűrőt használok. Mivel a zöngésség és a frikativitás a beszédhang frekvenciatartomány jól egy-egy jól elkülöníthető tartományát foglalja el, ezért az előbbiekben említett szűrőkkel jól megvalósítható.

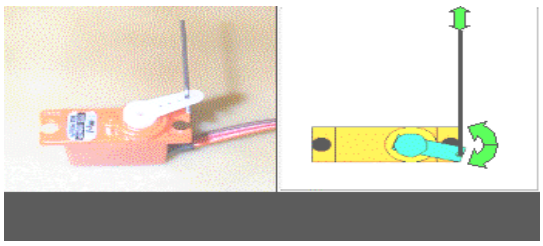
# 4. A taktilis segédeszköz

Sokféle taktilis kijelző létezik már napjainkban, csakhogy jellegzetesen siketek számára segédeszközként való felhasználására még nem került sor. Azon felül emberi beszédhang információk kijelzése teljesen új jellegű felhasználási terület az ilyen jellegű eszközök felhasználásával. Az általam tervezett eszköz egy gyors hangjelek jól érzékelhető kijelzésére szánt berendezés. A tervezés során figyelembe vettem a beszédhangjellemzőket, mint bemeneti jelek szükséges jellemzőit és a tapintásérzékelés jellemzőit is, mint a kimeneti jelek kívánatos jellemzőit. A ki-bemeneti jellemzők meghatározásának köszönhetően az eszköz információátadása érzékelhető és célravezető is a végfelhasználó számára.

## *4.1. A taktilis kijelzőkről általában*

A taktilis kijelző egy olyan eszköz, amelynek segítségével a tapintó szerveken keresztül képes információt közölni a felhasználóval. Ez az információ lehet hang vagy vizuális típusú, amit mechanikai ingerekké alakít át.

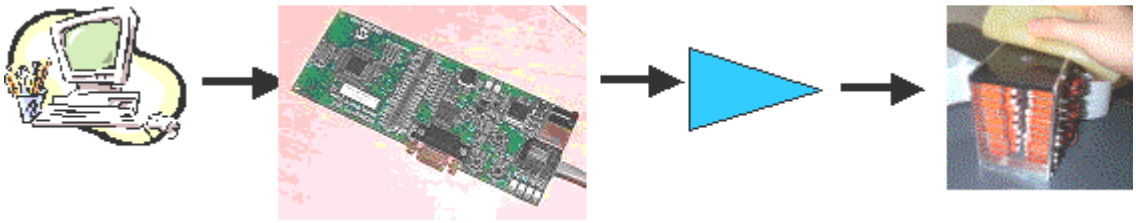
Tipikus alkalmazása vakok számára a Braille vagy a hagyományos írást valósítja meg és teszi érzékelhetővé. Ilyen például az OPTACON (OPtical to TActile CONverter), ezt az 1970-es évek elején fejlesztették ki. Ez egy hordozható olvasórendszer. A felhasználó egyik kezével egy kis kamera segítségével soronként olvassa be a síkírás egy-egy betűjét. A másik kezével, pedig magán a kijelzőn a 6-szor 24 vibráló tűvel érzékeli az információt. Ekkor a kijelző egy betű illetve annak egy részletének felismeréséhez elegendő. A kamera által rögzített kép real-time jelenik meg a kijelzőn. A gyakorlat azt mutatja, hogy a felhasználók meg tudják szokni, bár nagyon sok gyakorlással is csak elég lassan. A nyolc pontos Braille-írást megvalósító eszköznél elegendő kétszer négyes kijelző. Ezzel a szerkezettel gyorsabban tudják olvasni a kijelzett szövegét, hiszen a Braille-írás úgy lett kitalálva, hogy ez ne okozzon gondot a vakok számára.



**6. ábra: az OPTACON** **7. ábra: a szervomotor**

## 4.2. Egy taktilis kijelző egyszerű megvalósítása

A mechanikus ingert rezgő tűk hozzák létre. A tűk rezgését egy villanymotor forgó mozgása eredményezi, több motorból és az általa meghajtott tűkből lehet létrehozni egy kijelzőt. A kijelzőt egy mikrokontroller vezérli PC-ről felprogramozva. Statikus bemeneti jelek esetén ez elegendő, lásd Braille kijelző. Esetünkben azonban egy olyan gyorsan változó jel, mint a hang túl lassú a villanymotor. A dinamikus hangjel bemenethez egy elekromágneses stimulátor könnyebben tudja biztosítani a megfelelő kimeneti jelet.



**8. ábra: a kijelző megvalósítása**

## 4.3. Az új taktilis segédeszköz felépítése

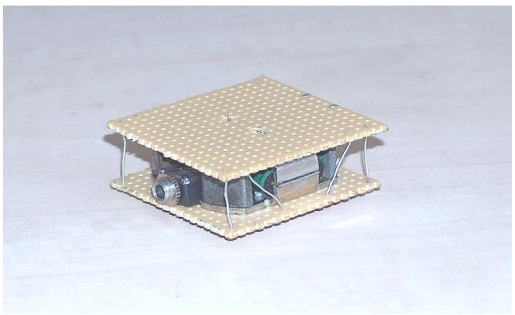
A segédeszköz önállóan működni kell hogy tudjon PC nélkül is. Vagyis fontos követelmény a mobilitás és ezen kívül az üzemtakarékosság is, amit a tervezés alatt figyelembe vettem. A felhasználás során karóraszerű viseletre szánom a terméket, így a segédeszköz a hétköznapi használat folyamán a természetes beszéd közben is segíti a szájról való olvasást, nem elegendő a jó előre rögzített adatok feldolgozása. Ezeket a szempontokat figyelembe véve egy real-time működő eszközt valósítottam meg.

**A segédeszköz főbb részei részei**: egy mikrokontroller, a vibrotaktilis kijelző a két tüskével és a tüskénkénti egy-egy szűrő valamint egy mikrofon. Az eszköz működése leegyszerűsítve a következő: az emberi beszéd jelet az eszköz a mikrofonon keresztül érzékeli, a mikrokontroller elvégzi az analóg-digitális feldolgozást, a szűrők elvégzik a

zöngedetekciót és a frikatív hangok detekcióját, majd a tüskék vibrotaktilis módon kijelzik ezeket.

A ***mikrokontroller*** a PICDES FS USB lap által használt mikrokontroller a PIC18F4550. Amely tartalmaz analóg-digitális átalakítót, timer modulokat, USB csatlakozót, vagyis a taktilis segédeszköz megvalósításra megfelel.

**9. ábra: a mikrokontroller**



**10. ábra: a taktilis kijelző**

A ***taktilis kijelző***t alkalmaztak eddig is ahogy már megjegyeztem, de csak statikus bemeneti jelek kijelzésére. Az én újításom a dinamikus jel kijelzése. Ezért a dinamikus hangjel bemenethez valóban egy elekromágneses stimulátort használtam a megfelelő kimeneti jel biztosításához. Az elktromágneses egység hátránya, hogy megnőtt a berendezés súlya.

A ***szűrők***et a real-time zöngedetekcióra és frikativitás detekció érdekében használjuk.

## *4.4. A kijelző felillesztése*

A kijelzőt felillesztettem a demonstrációs lapra. A két tüske a tervezésnek köszönhetően a demonstrációs lap két LED-jével azonos módon működik, vagyis egy LED bekapcsolására egy tüske lép működébe. A kijelző használata közben megfigyeltem, hogy a kis, alig 0,2 mm-es elmozdulás is tökéletesen érzékelhető, ahogy azt elvártam.

A következőkben a tesztelhetőség érdekében szükség van az önálló működési képességen kívül, hogy személyi számítógépről is vezérelhető legyen a berendezés.

# 5. A mikrokontroller

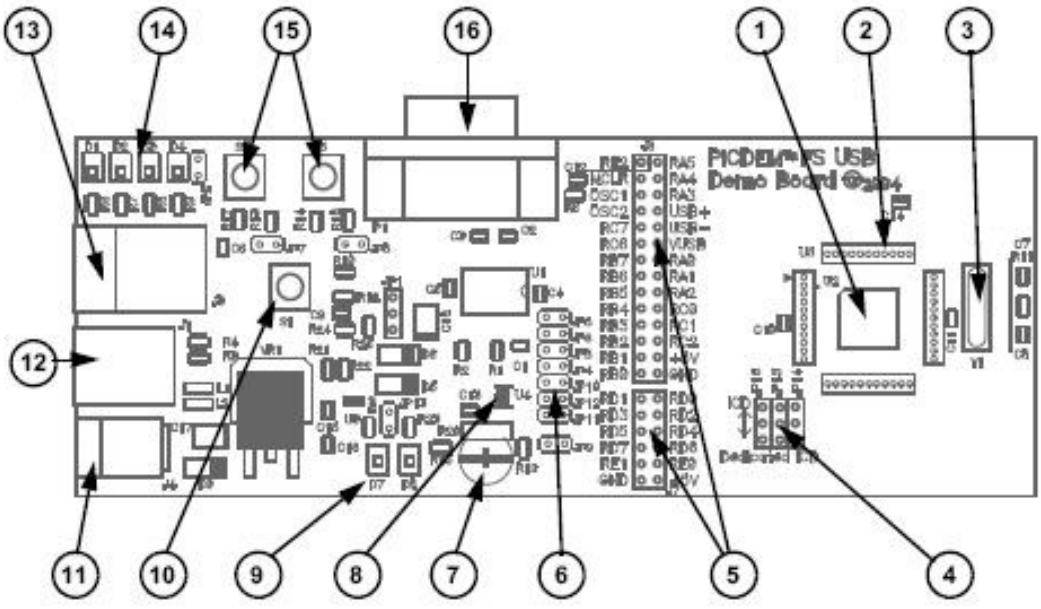
Felépítését tekintve a taktilis segédeszköz áll a kijelzőből és a hozzá csatlakoztatott vezérlő egység. Ebben a fejezetben a megvalósításhoz kiválasztott vezérlő egység, mikrokontrollert működését, felépítését vizsgálom meg.

## *5.1. A PICDEM FS USB lap jellemzői*

A PICDES FS USB lap által használt mikrokontroller a PIC18F4550. Ez a berendezés a PIC18F2455/2550/4455/4550 család tagja. Ezek közül mindegyik implementálja az energiatakarékos nanoWatt Technológiát, erősített Flash program memóriát és USB modult a következő képpen:

- USB 2.0 kompatibiitás
- Full-speed (12 Mbit/s) és low-speed (1,5 Mbit/s) művelet
- 1 Kbyte duális hozzáférésű RAM az USB részére
- On-chip megvalósítás, USB implementációhoz
    - USB adóvevő
    - USB feszültség szabályzó
    - USB pull-up ellenállások
- Interfész off-chip USB adóvevőhöz

A lap általános elrendezése a következő ábrán látható, a fő alkatrészek listájával.



**11. ábra: a demonstrációs lap elrendezése**

Az alkatrészek listája:

1. A mikrokontroller: A 44 pint tartalmazó, TQFP PIC18F4550 mikrokontroller a demonstrációs lap szíve, és ellát minden USB funkciót egy chip-en.

2. Az ICE Interfész Riser: A mikrokontroller egy 44-pin paddal van körülvéve, oldalanként 11 darab. Ez az elosztás lehetővé teszi a Microchip MPCLAB ICE 2000/4000 emulátor rendszer használatát.

3. Az oszcillátor: A demonstrációs lap egy 20 Mhz-es kristály oszcillátort használ a fő órajel biztosításához. A PIC18F4550 ezt az oszcillátort használja az USB soros interfész és a processzormag szükséges órajelének generálásához.

4. Az ICD konfigurációs jumperek: Ezek a három használaton kívüli jumper pozíció lehetővé teszi akármelyik ICSP és ICD portok kiválasztását.

5. A bővítő és PICtail headerek: Ezek a padok gondoskodnak a mikrokontroller I/O jel header és direkt hozzáférésről. Ennek köszönhetően használható a PICDEM FS USB lap, mint egy teszt platform és USB kommunikációs interfész.

6. A konfigurációs jumperek: Tizenhárom szabad jumper pozíció teszi lehetővé a lap konfigurálását. Alapesetben mindegyik jumper elérhető.

7. A potenciométer: A potenciométer szimulál egy analóg bemenetet a kontroller számára. Ezt a real-time értéket könnyen kezelhető és kényelmesen kijelezhető.

8. A hőmérsékletérzékelő szenzor: Egy Microchip TC77 digitális hőmérsékletérzékelő szenzor folyamatosan figyeli a lap körüli hőmérsékletét. Az adat továbbítódik egy 3 szalagos SPI interfészen keresztül a kontrollerhez, és szintén real-time kijelezhető, feldolgozható.

9. A power LED-ek (zöld): Ezek a lámpák mutatják azt az áramot, ami lapot ellátja. A LED D7 kijelzi, hogy a lap a buszról kap áramot, a D8 azt pedig, hogy más áramforrásról.

10. A reset gomb: Ez a kapcsoló a kontroller MCLR pin-re van csatlakoztatva a PIC18F4550 kontrolleren, a megnyomása egy hard device resetet okoz.

11. Az energia ellátás csatlakozója: Az energia biztosítható a lap számára egy külső energiaforrástól. Ez megtehető az USB csatlakozón keresztül is.

12. Az USB csatlakozó: Ez egy szabványos USB soros, „B” típusú foglalat. Az USB port az alapvető csatorna a kommunikációhoz és vezérléshez.

13. Az ICD csatlakozó: A 6 vezetékes RJ11 csatlakozó szolgáltatja a szabványos interfészt a Mikrochip fejlesztői és demonstrációs lap programozásához és debuggolásához a MPLAB ICD 2 használatával.

14. Az állapot LED-sor: A négy zöld LED-et a lap opcionális állapotának kijelzésére használják. Két LED-et (a D1 és a D2) az alkalmazás az USB csatlakozás kijelzésére használják. A másik két LED (a D3 és a D4) a felhasználó által definiálható.

15. A felhasználó által definiálható nyomó gombok: Ez a két kapcsoló (az S2 és az S3) biztosítja a digitális bemenet szimulációt. Bármelyik gomb megnyomása 0 olvasását okoz az adott porton.
16. Az RS-232 (DB9F) port: Egy szabványos csatlakozó, ami egy RS-232 soros kapcsolatról gondoskodik a demonstrációs lap számára.

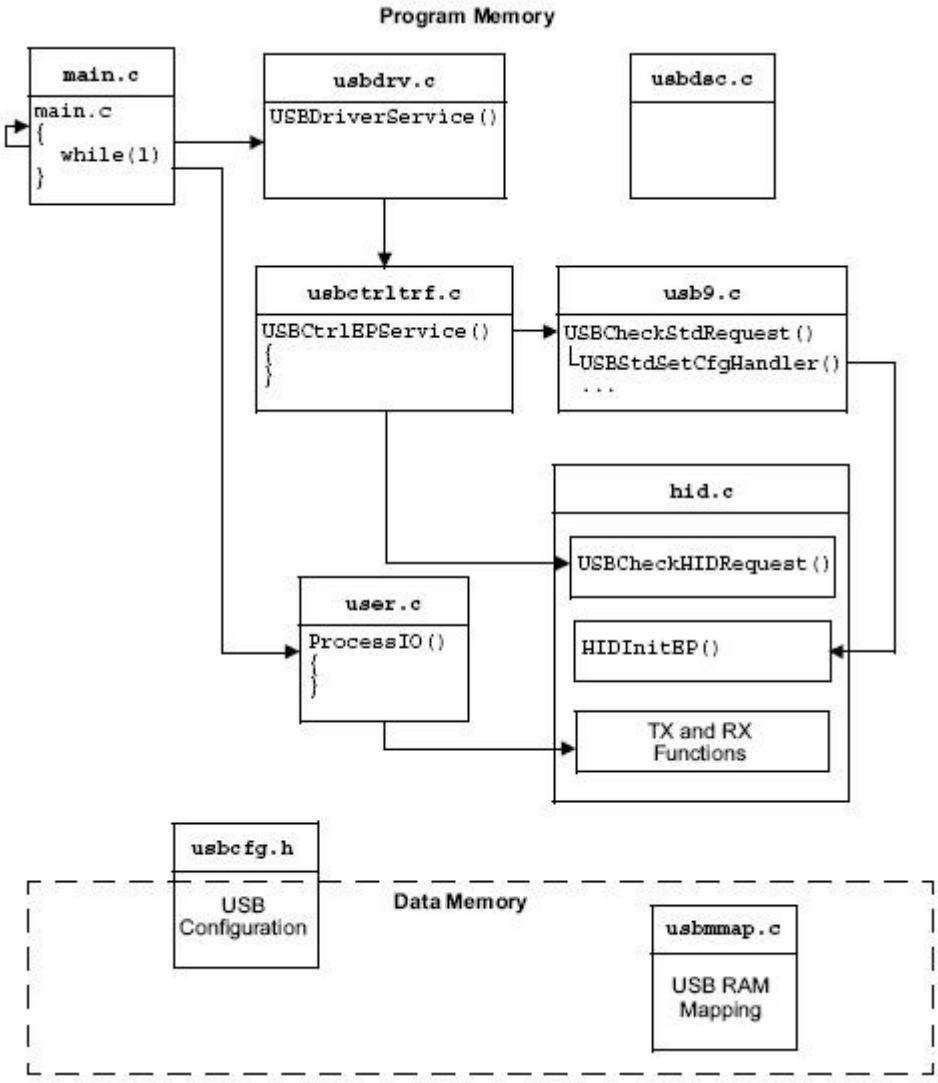
## 5.2. A Microchip USB Firmware Framework használata

A Microchip USB Firmware Framework egy fájlrendszer konstrukció, melynek segítségével új USB alkalmazásokat lehet készíteni. Ez lehetséges egy referencia projecten keresztül, tartalmazva a szükséges frimware kódot az USB műveletekhez, és a fejlesztő forrás kódjaival. Az egész projectet egy gyökér könyvtár tárolja, több alkönyvtárral.

### 5.2.1. A Framework struktúra

A fájlstruktúra tartalmazza az alkönyvtárakat és a fájlokat egy gyökér könyvtárban. Egy USB projectnek pontosan meg kell lennie a helyének és egy érvényes projekt névnek, ezeket a fejlesztés közben mindig karban kell tartani. Minden példa projekt a PICDEM FS USB demonstrációs laphoz, használja ezt a struktúrát. A következő alkönyvtárakat és fájlokat mindig prezentálni kell:
Az alkönyvtárak:
• _output: a kimeneti fájlok központi helye
• autofiles: Tartalmazza az USB globális konfigurációs fájlt és leírásokat
• system: Tartalmazza az USB firmwaret
• user: A felhasználó firmwarének a helye
A fájlok:
• CleanUp.bat: Töröl minden kimeneti fájlt ebben a könyvtárban és minden alkönyvtárban
• io_cfg.h: Bemeneti neveket köt össze bemeneti függvényekkel, ezt a fájlt csak úgy szabad megváltoztatni, hogy megfeleljen minden más projektbeli fájlnak.
• main.c: Ez a fájl tartalmazza a main() függvényt az alkalmazások
• MCHPUSB.mcp: MPLAB IDE a projekt fájl
• MCHPUSB.mcw: MPLAB IDE a munkafelület fájl

Az USB Firmware Framework szintén gondoskodik moduláris interfészekről, amelyek sok munkát kezelnek az USB kommunikáció implementálásához. A következő ábra egy tipikus USB program folyamatát szemlélteti.




**12. ábra: összefüggés a Microchip USB keretrendszer fájlai és a tipikus HID alkalmazások között**

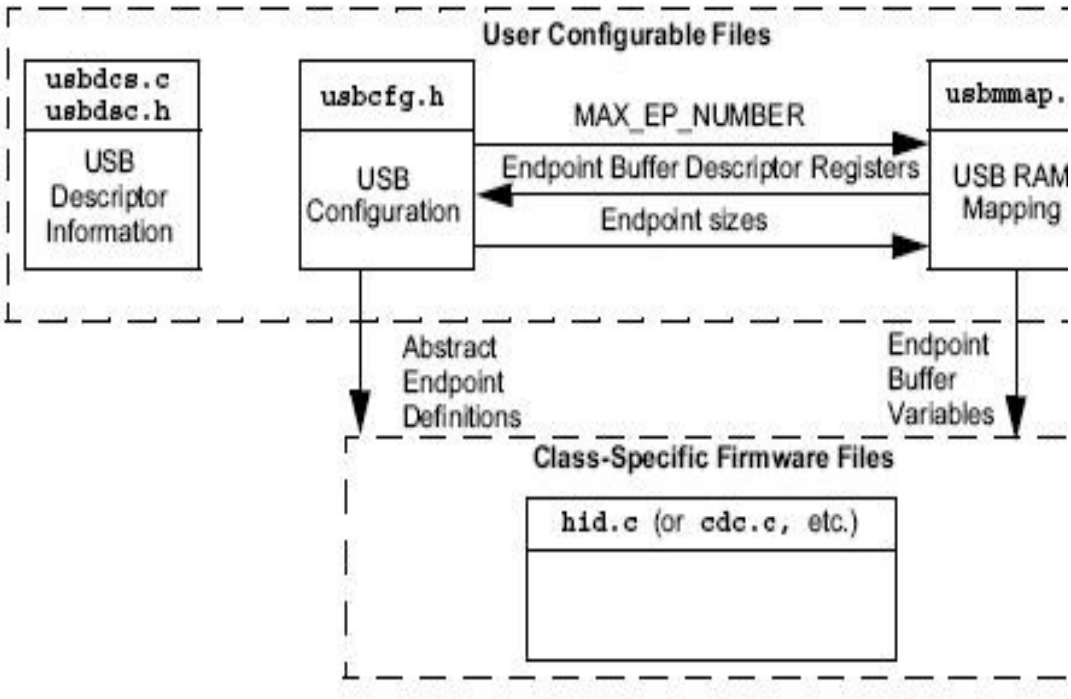
A main() függvény egy végtelen while ciklus, amely több feladatot hajt végre. Ez lehet akár USB vagy egyéb felhasználói feladat is. Az USB feladatokat az USBDriverService() függvény kezeli, amely valamennyi USB hardver interruptot tartalmazza. Ha egy vezérlő végpont tranzakció megy végbe, az USBCtrlEpService() függvény hívódik meg. Minden átvitelnek követnie kell a kontrol átviteli protokollt, ahogy az USB specifikációban le van írva.

A vezérlés átvitelét az usbctrltrf.c fájl által biztosított. A vezérlés átvitel első állomása kérésként érkezik meg. Egy USB kérés lehet szabványos vagy osztály specifikus. A szabványos kérést az USBCheckStdRequest() függvény szolgáltatja. Egy osztály specifikus kérést egy firmware fájlnak kell kezelnie az USB specifikációnak megfelelően. Például a Human Interface Device (HID) és Communication Device Class (CDC) osztályoknál az osztály függvényeit a specifikus osztály firmware fájlban kell tárolni például a hid.c és cdc.c fájlokban. A névsémák a következő formában néznek ki USBCheck<class name>Request(), ahol a <class name> az osztály specifikus neve kell, hogy legyen.

Az USB felsorolás folyamatot főleg az usb9.c fájl vezérli. Az egyik legfontosabb lépés a felsorolás folyamatban a SET_CONFIGURATION parancs kezelése, amelyet az USBStdSetCfgHandler() hajt végre. Ez a függvényt a felhasználó módosíthatja, hogy meghívja a megfelelő függvényeket az alkalmazás végpontok inicializálására. Minden specifikus osztály firmwarenek kell hogy legyen egy végpont inicializációs rutinja. Ennek a függvénynek a névkondíciója ebben az esetben <class name >InitEP(), ahol a <class name> bármely osztálynév lehet. Például ha a HIDInitEP(), ami mondjuk tartalmazza azt a kódot hogy hogyan inicializáljuk a végpontok egy csoportját amik egy bizonyos osztályban vannak használva. A felhasználóknak gondoskodniuk kell arról, hogy melyik konfigurációs indexet kéri a beállításhoz az USB host. Egy eszköznek lehet többféle konfigurációja és nem minden interfész ugyanolyan másféle konfigurációkon keresztül.

A felhasználó alkalmazás kódját szintén a fő program ciklusból hívják meg, ez alapból a ProcessIO() függvény alá tartozik. Ha az alkalmazásnak küldeni vagy fogadni kell egy USB tranzakciót, akkor meghívhat egy előre megírt függvényt, ami szolgáltatja a küldési vagy fogadási műveletet, ami deklarálva van az osztály specifikus firmware kódban.

Az USB eszköz konfigurációt egy moduláris módban is kezelik. A változók értékeinek megváltoztatásával néhány fájlban ezeket az információkat globálissá teszik, és így elérhetővé válnak fordítási időben. A fájlok teljesen függetlenek egymástól és információt továbbítanak egymás között a fordítási időben, hogy létrehozzák a teljes USB konfigurációt. Ezeket az összefüggéseket mutatja a következő ábra.




**13. ábra: USB konfigurációs fájlok**

Először egy meglévő bemutató programot írtam át. A main() függvénybe írtam egy LED villogtató eljárást. Később ezt a projectet folytattam.

## 5.3. Az interrupt eljárás

Létezik egy nagy prioritású interrupt és egy alacsony prioritású interruptlow service rutin is. Egy interrupt felfüggeszti egy futó alkalmazás végrehajtását, elmenti az aktuális környezeti információt és átadja az irányítást egy interrupt service rutinnak, így azt hajtják végre. Az interrupt service rutin befejeztével az előző környezeti információ visszatöltődik, és az alkalmazás normális végrehajtása folytatódik. Az interrupt számára, a minimális környezeti információ, ami elmentődik a WREG, a BSR és a STATUS. A magas prioritású interrupt az árnyékregisztereket használja a minimális környezeti információ tárolásához és visszatöltéséhez, amíg az alacsony prioritású interrupt a szoftver vermet használja elmentésére és visszatöltésére. E miatt egy magas prioritású interrupt egy gyors *visszatérési interrupt*ként hajtódik végre, amíg egy alacsony prioritású interrupt normál *visszatérési interrupt*ként hajtódik végre. Két MOVFF utasítás szükséges a környezeti információ minden bájtjához a szoftver vermen keresztül kivéve a WREG, aminek szüksége van egy

MOWF utasításra és egy MOVF utasításra, ezért a minimális környezeti információ megőrzéséhez az alacsony prioritású interrupt egy további 10 szavas fejrésszel is rendelkezik a magas prioritású interrupt követelményein kívül.

Az interrupt service rutinok használnak egy ideiglenes adatszekciót, eltérő módon egy rendes C függvénytől. Minden ideiglenes adat, ami szükséges a kifejezések kiértékelésekor az interrupt service rutinban meg le van foglalva ebben a területen, ami nem lapolódik át a függvények ideiglenes területével, beleszámítva más interrupt függvények területét is. Az interrupt eljárások hozzájárulnak, hogy az ideiglenes adatszekciókat el lehessen nevezni. Ha ez a szekció nincs elnevezve, akkor a fordító az ideiglenes változókat egy fname_tmp nevű szekcióba hozza létre. Például:

```
void foo(void);
...
#pragma interrupt foo
void foo(void)
{
/* az interrupt függvény itt szerepel */
}
```

A fordító az itt szereplő interrupt service rutin ideiglenes változóit a foo_temp szekcióban helyezi el.
Egy MPCLAB C18 interrupt service rutin olyan, mint egy C függvény, lehetnek lokális változói, és hozzáférhet globális változókhoz, ám egy interrupt service rutint nem lehet paraméterekkel és visszatérési értékekkel deklarálni, mert az ISR aszinkron hívják meg. A globális változók, amiket mind az ISR mind a szokásszerű függvények meghívnak illékony változóként kell deklarálni.

Egy ISR-t csak hardver interrupton keresztül lehet meghívni, más C függvényen keresztül nem. Az ISR a return form interruptot (RETFIE) használja inkább a függvényből való kilépéshez, a normál RETURN parancs helyett. A gyors RETFIE parancs használata könnyen elronthatja a WREG, a BSR és a STATUS értékeit.

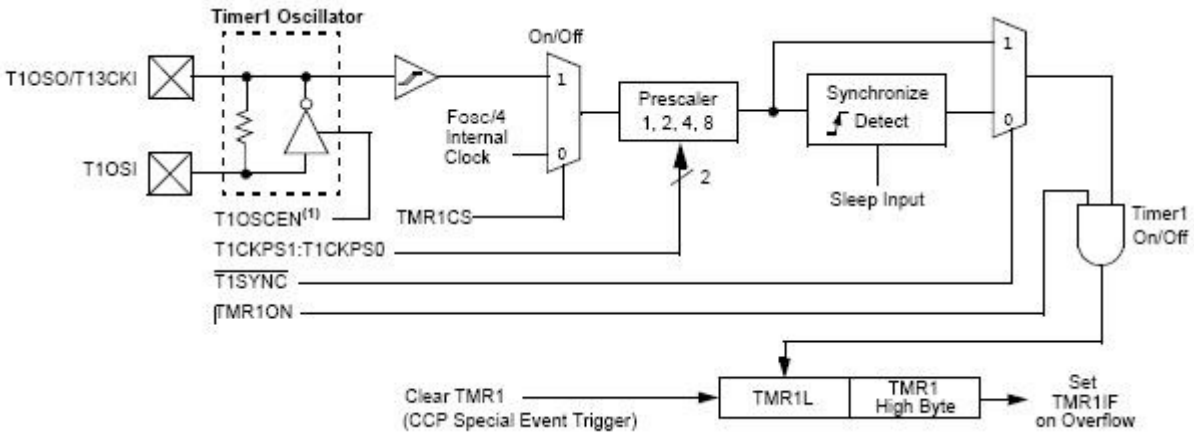
Az interrupt eljárást azért használtam az általam írt programban, mert a taktilis kijelző üzemtakarékos kell, hogy legyen. Ezért használaton kívül sleep üzemmódban lesz, és ha csak szükség lesz rá, fogja felébreszteni az interrupt a sleep üzemmódot. Így az egyszerű LED villogtató programot átírtam, hogy az interrupt függvényből működjön a LED villogtatás.

## 5.4. A TIMER modul

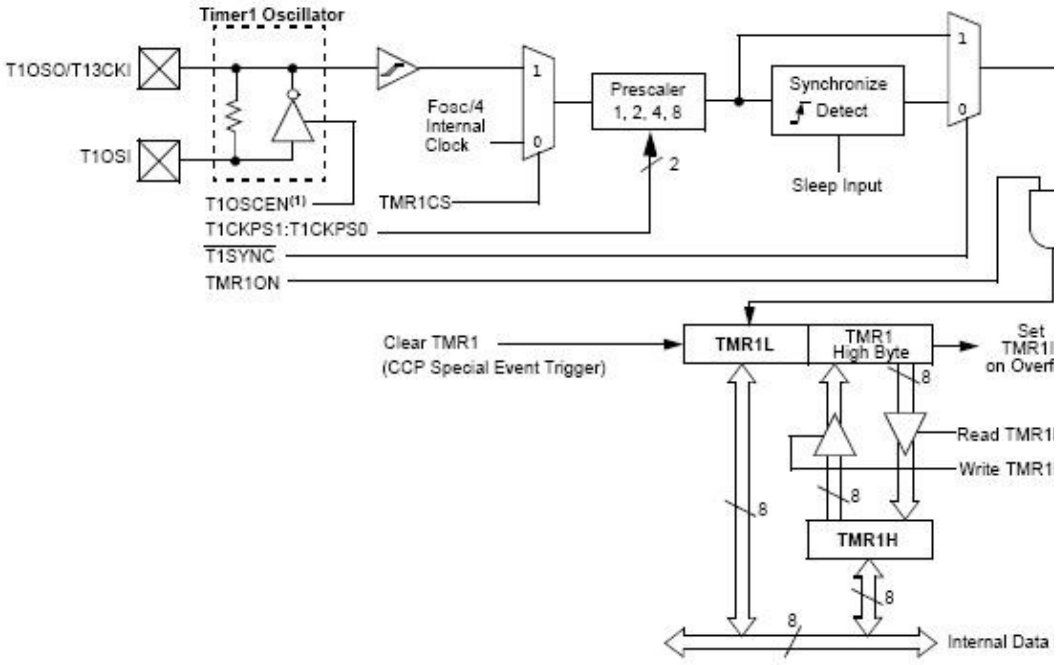
Az időzítő, számláló modul működését és használatát a TIMER1-en keresztül fogom bemutatni, mert és is ezt használtam először valamint a többi számláló hasonló módon kezelhető.

A TIMER1 időzítő vagy számláló magába foglalja a következő jellemzőket:

- Szoftver által kiválasztható művelet, mint 16 bites időzítő vagy számláló
- Olvasható és írható 8 bites regiszter (TMR1H és TMR1L)
- Választható órajel forrás (külső vagy belső), óraegység vagy Timer1 oszcillátor belső beállítások
- Interrupt-on-overflow
- Modul Reset
- Clock állapot flag (T1RUN)




**14. ábra: a TIMER1 modul blokk diagramja**



**15. ábra: a modul blokk diagramja írás-olvasás műveletkor**

A modul magába foglal egy saját alacsony energiájú oszcillátort, hogy gondoskodjon további clock opcióról. A TIMER1 oszcillátor használható úgy is, mint egy alacsony energiafogyasztású órajel forrás a mikrokontroller számára power-manage módban. Használható továbbá Real-Time Clock-nak (RTC), külső alkatrész minimális kiegészítőjének. A Timer1-et a T1CON vezérlő regiszteren keresztül vezérlik. Ez tartalmazza az oszcillátor engedélyezési bitjét (T1OSCEN). Engedélyezni vagy letiltani a TMR1ON (T1CON<0>) beállítás (setting) vagy törlés (clearing) bittel lehet.

| R/W-0 | R-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 |
|---|---|---|---|---|---|---|---|
| RD16 | T1RUN | T1CKPS1 | T1CKPS0 | T1OSCEN | $\overline{\text{T1SYNC}}$ | TMR1CS | TMR1ON |
| bit 7 | | | | | | | bit 0 |

**16. ábra: A T1CON: TIMER1 kontrol regiszter**

bit 7 **RD16:** 16 bites olvasás/írás módengedélyezés bit

1 = Elérhető a Timer1 regiszter 16 bites írás/olvasás művelet

0 = Elérhető a Timer1 regiszter 8 bites írás/olvasás művelet

bit 6 **T1RUN:** Timer1 Rendszer órajel állapot bit

1 = Egység órajel a Timer1 oszcillátortól származik

0 = Egység órajel más forrásból származik

bit 5-4 **T1CKPS1:T1CKPS0:** Timer1 Input Clock Prescale Select bits

11 = 1:8 Prescale érték

10 = 1:4 Prescale érték

01 = 1:2 Prescale érték

00 = 1:1 Prescale érték

bit 3 **T1OSCEN:** Timer1 Oscillator elérési bit

1 = a Timer1 oszcillátor elérhető

0 = a Timer1 oszcillátor elérhető

Az oszcillátor inverter és visszacsatolás ellenállás ki van kapcsolva.

bit 2 **T1SYNC:** Timer1 külső szinkron óra input kiválasztó bit

HA TMR1CS = 1:

1 = Nem szinkronizálja a külső óra bemenetét

0 = szinkronizálja a külső óra bemenetét

Ha TMR1CS = 0:

Ezeket a biteket ki kell hagyni. A Timer1 használhatja a külső órát, ha TMR1CS = 0.

bit 1 **TMR1CS:** Timer1 órajel forrás kiválasztó bit

1 = Külső órajel a RC0/T1OSO/T13CKI pinről

0 = Belső óra (FOSC/4)

bit 0 **TMR1ON:** Timer1 bit

1 = a Timer1 engedélyezése

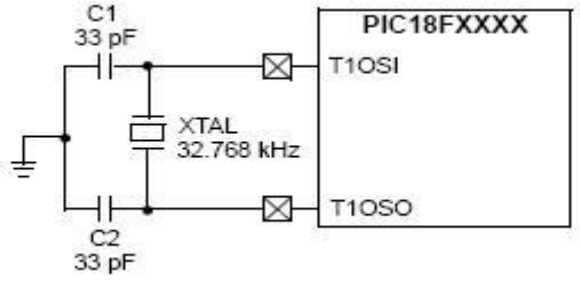
0 =a Timer1 megállítása

A Timer1 a következő módokon tud működni:

- Időzítő
- Szinkronszámláló
- Aszinkronszámláló

A működési módot meghatározza az órajel kiválasztási TMR1CS (T1CON<1>) bit. Ha a TIMER1 engedélyezve van, akkor a RC1, T1OSI, UOE és RCO, T1OSO, T13CKI pinek bemenetté válnak. Ez azt jelenti, hogy TIRSC<1:0> értéket kihagyják és a pinekről 0-t olvasnak.

Az írás, olvasási mód: A Timer1-et 16 bites olvasásra, írásra is lehet konfigurálni. Ha a RD16 kontrol bit (T1CON<7>) be van állítva, akkor a TMR1H címe át van irányítva a buffer regiszterbe. Egy olvasási művelet a TMR1L-ből a Timer1 high bájt tartalmából fog betöltődni a Timer1 high byte bufferébe. Ez lehetővé teszi, hogy pontosan ki lehessen olvasni a Timer1 mind a 16 bitjét, anélkül hogy meg kelljen határozni, hogy vajon a high bit olvasása követi-e a low bit olvasását és ellenőrizni hogy vajon ez által érvénytelenné vált-e a művelet. A Timer1 high bájtjába való íráshoz helyet kell foglalni a TMR1H buffer regiszterben. A Timer1 high bájt a TMR1H tartalmával frissül, ha egy írás művelet történik. Íráskor a high és low bájtokat egyben kell kezelni. A Timer1 high bájtja nem olvasható vagy írható direkt módon ilyenkor. Minden íráskor és olvasáskor helyet kell biztosítani a Timer1 High Bájt Regiszter segítségével. Az írás nem törli a Timer1 prescaler-t. A prescaler csak íráskor lehet törölni.



**17. ábra: egy tipikus oszcillátor áramköri rajza**

A timer1 oszcillátor: A chip-en elhelyezkedő kristály oszcillátor a T1OSI bemeneti és T1OSO kimeneti pinek között helyezkedik el. Ezt engedélyezni kell a Timer1 engedélyezési T1SCEN (T1CON<3>) bit segítségével. Az oszcillátor egy alacsony energiafogyasztású 32 kHz-es kristály. Ez képes működni energiatakarékos módozatokban is.

## *5.5. A 10 bites analóg-digitális átalakító modul*

Az analóg-digitális átalakító modulnak 10 bemenete van a 28 pines berendezések számára, és 13 bemenete van a 40/44 pines berendezések számára. Ez a modul lehetővé teszi egy analóg bemeneti jel konvertálását egy megfelelő 10 bites digitális számmá. A modulnak 5 regisztere van:

- A/D eredmény high regiszter (ADRESH)
- A/D eredmény low regiszter (ADRESL)
- A/D kontrol regiszter 0 (ADCON0)
- A/D kontrol regiszter 1 (ADCON1)
- A/D kontrol regiszter 2 (ADCON2)

Az ADCON0 regiszter az A/D modul műveleteit mutatja.

bit 7-6 **Nem implementált:** 0-t olvas

bit 5-2 **CHS3:CHS0:**Analóg csatorna kiválasztó bitek

0000 = csatorna 0 (AN0)

0001 = csatorna 1 (AN1)

0010 = csatorna 2 (AN2)

0011 = csatorna 3 (AN3)

0100 = csatorna 4 (AN4)

0101 = csatorna 5 (AN5)

0110 = csatorna 6 (AN6)

0111 = csatorna 7 (AN7)

1000 = csatorna 8 (AN8)

1001 = csatorna 9 (AN9)

1010 = csatorna 10 (AN10)

1011 = csatorna 11 (AN11)

1100 = csatorna 12 (AN12

1101 = nem implementált

1110 = nem implementált

1111 = nem implementált

bit 1 **GO/DONE:** A/D konverzió állapot bit

Ha ADON = 1:

1 = A/D konverzió folyamatban

0 = A/D készenlét

bit 0 **ADON:** A/D bekapcsolva bit

1 = A/D konverter modul elérhető

0 = A/D konverter modul nem elérhető

Az ADCON1 regiszter a port pinek konfigurációját mutatja.

bit 7-6 **Nem implementált:** 0-t olvas

bit 5 **VCFG1:** feszültség referencia konfigurációs bit (VREF- source)

1 = VREF- (AN2)

0 = VSS

bit 4 **VCFG0:** feszültség referencia konfigurációs bit (VREF+ source)

1 = VREF+ (AN3)

0 = VDD

| PCFG3: PCFG0 | AN12 | AN11 | AN10 | AN9 | AN8 | AN7(2) | AN6(2) | AN5(2) | AN4 | AN3 | AN2 | AN1 | AN0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0000(1) | A | A | A | A | A | A | A | A | A | A | A | A | A |
| 0001 | A | A | A | A | A | A | A | A | A | A | A | A | A |
| 0010 | A | A | A | A | A | A | A | A | A | A | A | A | A |
| 0011 | D | A | A | A | A | A | A | A | A | A | A | A | A |
| 0100 | D | D | A | A | A | A | A | A | A | A | A | A | A |
| 0101 | D | D | D | A | A | A | A | A | A | A | A | A | A |
| 0110 | D | D | D | D | A | A | A | A | A | A | A | A | A |
| 0111(1) | D | D | D | D | D | A | A | A | A | A | A | A | A |
| 1000 | D | D | D | D | D | D | A | A | A | A | A | A | A |
| 1001 | D | D | D | D | D | D | D | A | A | A | A | A | A |
| 1010 | D | D | D | D | D | D | D | D | A | A | A | A | A |
| 1011 | D | D | D | D | D | D | D | D | D | A | A | A | A |
| 1100 | D | D | D | D | D | D | D | D | D | D | A | A | A |
| 1101 | D | D | D | D | D | D | D | D | D | D | D | A | A |
| 1110 | D | D | D | D | D | D | D | D | D | D | D | D | A |
| 1111 | D | D | D | D | D | D | D | D | D | D | D | D | D |

**18. ábra: bit 3-0 PCFG3:PCFG0: A/D Port konfiguráció kontrol bitek**

Az ADCON2 regiszter konfigurálja az A/D órajel forrást.

bit 7 **ADFM:** A/D eredményforma kiválasztó bit

1 = jobb justified

0 = bal justified

bit 6 **Nem implementált:** 0-t olvas

bit 5-3 **ACQT2:ACQT0:** A/D beszerzési idő kiválasztó bitek

111 = 20 TAD

110 = 16 TAD

101 = 12 TAD

100 = 8 TAD

011 = 6 TAD

010 = 4 TAD

001 = 2 TAD

000 = 0 TAD**(1)**

bit 2-0 **ADCS2:ADCS0:** A/D konverzió órajel kiválasztó bitek

111 = FRC

110 = FOSC/64

101 = FOSC/16

100 = FOSC/4

011 = FRC

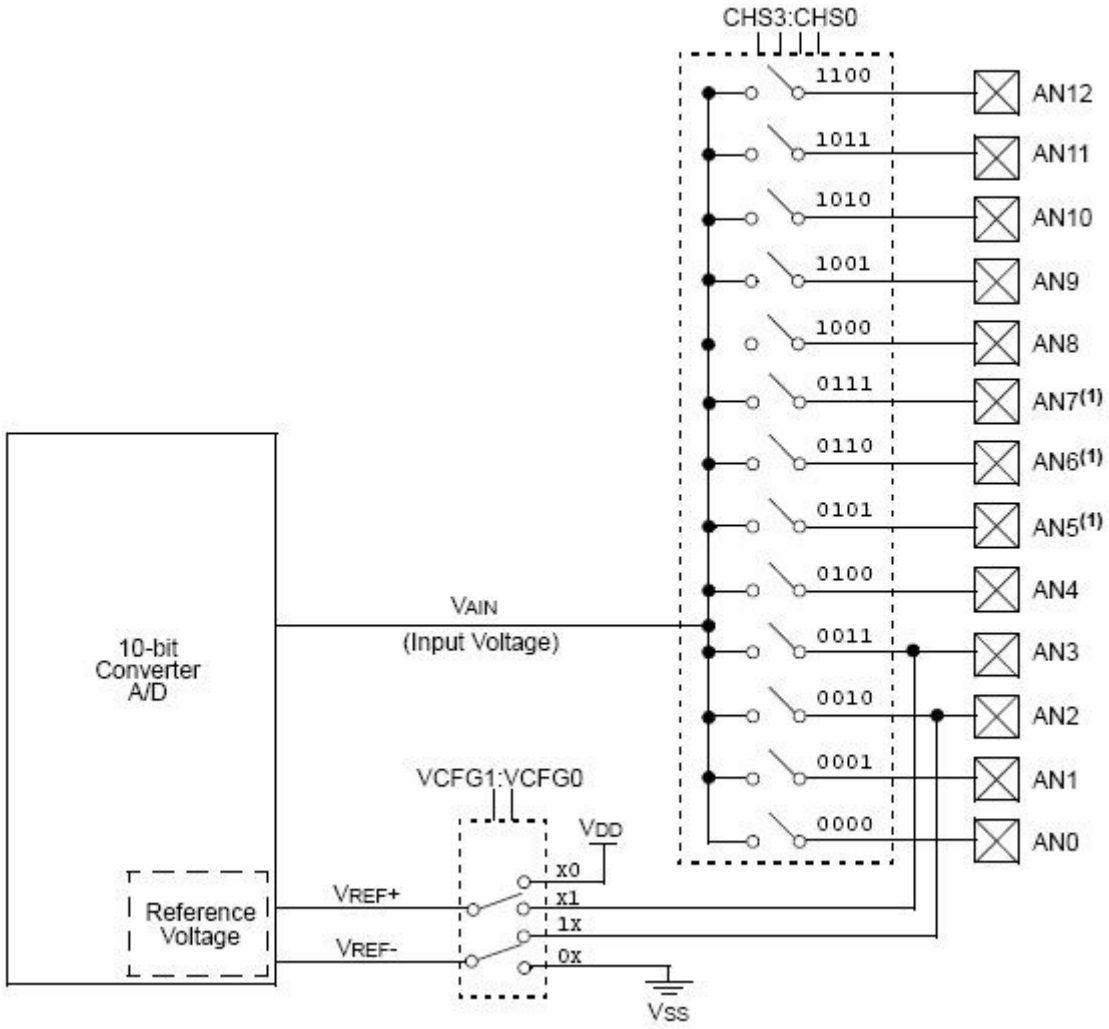
010 = FOSC/32

001 = FOSC/8

000 = FOSC/2

Az analóg referencia feszültség szoftveresen kiválasztható, akár pozitív vagy negatív feszültség vagy a feszültség szintje az RA3/AN3 és RA2/AN2 pineken. Az A/D konverternek vagy egy egyedi tulajdonsága, képes működni sleep üzemmód alatt is. Hogy működjön sleep üzemmódban az A/D konverziós órajelet a belső RC oszcillátorról kell, hogy kapja. Az eredményeket fokozatos megközelítéssel állítja elő.

Egy berendezés Reset parancsa minden regiszteren a Reset állapotára kényszerít. Ez a parancs az A/D modult kikapcsolja, és minden konverziós folyamatot megszakít. Az A/D átalakító analóg bemenetként vagy digitális ki-, be-menetként konfigurálható. Az ADRESH és ADRESL regiszterek tárolják az A/D átalakítás eredményét. Ha az A/D átalakítás kész van, az eredmény az ADRESH ADRESL regiszter párba töltődik, a GO/DONE bit törlődik és az A/D interrupt flag bit, ADIF beállítódik.




**19. ábra: Az A/D modul blokkdiagramja**

Az A/D átalakítás lépései a következők:

1. Az A/D modul konfigurálása
2. Az A/D interrupt konfigurálása
3. Várakozás a kérési időre
4. A konverzió elkezdése
5. Várakozás az A/D átalakítás elkészülésére
6. Az A/D regiszterek kiolvasása (ADRESH ADRESL)
7. A következő átalakításoz az 1. és 2. lépés végrehajtása

A taktilis kijelző megvalósításához szükség lesz A/D átalakításra. A mikrofonon beérkező analóg hangjelet kell majd átalakítani digitálissá, majd feldolgozni és a kijelzőn megjeleníteni a frikatív és zöngés hangokat. Az A/D átalakítás elsajátításához a programomban analóg bemenetnek a demonstrációs lapon lévő potenciométert használtam. Ezt a jelet átalakítva az egyik LED-en jelenítetten meg. A potenciométert tekergetve a LED lassabban illetve gyorsabban villog.

# 6. USB vezérlés PC-ről, az MPUSBAPI keretrendszer segítségével

A személyi számítógépről való irányításra a tesztelhetőség érdekében van szükség. A mikrokontroller és a számítógép közötti kétoldali kommunikáció eléréséhez szükséges hogy mind a PC oldali, mind a mikrokontroller oldali alkalmazás funkcióját hibátlanul betöltse, ezért párhuzamosan fejlesztettem mindkettőt. A számítógép felöli rész fejlesztését az MPUSBAPI keretrendszer használatával értem el.

## *6.1. Az MPUSBAPI keretrendszer*

Az MPUSBAPI alkalmazásprogramozó interfész segítségével lehetséges USB alkalmazásokat létrehozni. Ez tulajdonképpen egy DLL modul, amely wrapper funkciókat biztosít a Microchip által Windows alá tervezett, általános USB illesztő programjához, a mchpusb.sys fájlhoz valamint egyszerűsít a Win32 API funkcióinak komplexitásán. Az MPUSBAPI hét alapvető metódust tartalmaz, amelyek felhasználásával tudtam új alkalmazásokat produkálni.

A keretrendszer hét metódusa:

1) **DWORD *MPUSBGetDLLVersion(void)**

Az első metódus lekérdezi az MPUSBAPI.DLL verziószámát, amely egy 32 bites szám MMMMmmmm formátumban. Ez csak a szoftver kód verziószámát adja vissza, itt valójában az USB nem hajt még végre semmilyen műveletet.

2) **DWORD *MPUSBGetDeviceCount(PCHAR pVID_PID)**

A második metódus azon eszközök számát adja vissza, amelyeknek megfelel a VID és PID száma.

3) **HANDLE *MPUSBOpen(instance, pVID_PID,pEP,dwDir,dwReserved);**

Visszaadja az érvényes végponthoz tartozó kezelőt. Az útvonalak a FILE_FLAG_OVERLAPPED attribútummal nyílik meg. Ez lehetővé teszi az MPUSBRead, MPUSBWrite és MPUSBReadInt metódusok részére, hogy időtúllépési értékkel rendelkezzenek.

4) **DWORD *MPUSBRead(handle,pData,dwLen,pLength,dwMilliseconds);**

Ezt a metódust a kezelőről való olvasásra tervezték.

5) **DWORD *MPUSBWrite(handle,pData,dwLen,pLength,dwMilliseconds);**

Az előző párja, csak éppen ír a kezelőre. Ez a két függvény kulcsfontosságú a protokoll szempontjából.

6) **DWORD *MPUSBReadInt(handle,pData,dwLen,pLength,dwMilliseconds);**

Egész értékeket olvas be a kezelőről.

7) **BOOL *MPUSBClose(handle);**

Az utolsó metódus lezárja a megadott kezelőt.

A DLL linkelésnek két módja van, a fordítás idejű linkelés és a futtatás idejű linkelés. A fordítás idejű linkelés során egyszerűen hozzá kell adni az mpusbapi.lib fájlt a projekthez. A futtatás idejű linkelés során a funkciók linkelése valamivel bonyolultabb, mert nem használja a DLL fájlt. Ezért a futtatás idejű linkelést használtam.

## 6.2. Az új USB protokoll

Az USB-n keresztüli kommunikációt adatcsomagok segítségével valósítottam meg. Egy adatcsomagot a DATA_PACKET struktúra típus realizál. Ezen adattípus deklarációja a firmware programkódban, a user.h fejlécfájlban található. Az adatcsomag elemei:

- a byte típusú **_byte[USBGEN_EP_SIZE]** tömb, ez tárolja az elküldött és fogadott adatokat
- a hexadecimális alakú számokat felsoroló, és azokhoz string-et rendelő **CMD** elem, amely a PC oldali alkalmazás által a mikrokontrollernek küldött parancs, pl. **READ_VERSION, KULDES**
- és a byte típusú **len**, amely a küldött csomag hosszát jelenti.

A **user.c** fájl **ServiceRequests(void)** metódusában egy többszörös elágazás szerkezetben dolgozza fel a PC alkalmazástól küldött parancsokat, minden parancsra másképp reagál, pl. a READ_VERSION hatására elküldi a verziószámot, vagy a **LED3_ON** esetében felkapcsolja a demoboardon a harmadik LED-et. Nélkülözhetetlen minden case ágban a counter nevű változót a PC-re elküldött csomag méretére állítani. Ezután az **USBGenWrite((byte*)&dataPacket, counter)** függvény elküldi a csomagot. A PC oldali alkalmazás felé küldött adatok a **dataPacket._byte[]** megfelelő elemeibe kerülnek, pontosabban a harmadik elemtől kezdődően, mert az első kettő a firmware programhoz elküldött parancs, valamint a küldött csomag hosszának van fenntartva.

A PC oldali alkalmazást egy dev-c++ project-ben valósítottam meg. Fő része a **console.cpp** fájlban található, itt fogalmaztam meg parancsokat, amelyeket átad a PC a mikrokontrollernek. A konzolon bizonyos számokat bekérve és egy összetett elágazásban feldolgozva, a program a számoknak megfelelő parancsot fog küldeni, amelyre az USB eszköz különbözőképen viselkedik. A parancsokat függvényekbe foglaltam, amely a protokoll meghatározott lépésekeit hatja végre.

### 6.2.1. A protokoll lépései

**1. lépés:** Első ízben létrehozza az oda-vissza adatcsatornákat a számítógép a mikrokontroller között.

**2. lépés:** A mikrokontroller létrehozza a küldött és fogadott adatok számára a buffereket (ezek a BYTE send_buf[64] és receive_buf[64]).

**3. lépés:** Az ellenőrzésekhez meghatározza a fogadott csomag elvárt méretét (DWORD RecvLength).

**4. lépés:** Szükség van az elküldött parancsot és a küldött csomag hosszát definiálni, ezt a két adatot a további működéshez át kell adni a send_buf[] első két elemének.

**5. lépés:** A mikrokontroller létrehoz egy adatcsomagot, ami a DATAPACKET típusnak felel meg.

**6. lépés:** A DWORD SendReceivePacket(BYTE *SendData, DWORD SendLength, BYTE *ReceiveData, DWORD *ReceiveLength, UINT SendDelay, UINT ReceiveDelay) wrapper függvény ellenőrzi a csomagok méretét, majd elküldi az adatot a send_buf[] segítségével és fogadja a receive_buf[] használatával.

**7. lépés:** A mikrokontroller oldalon a void ServiceRequist(void) eljárásban az USBGenRead(*DATAPACKET dataPacket, int size) fogadja az adatcsomagot, melynek a CMD kompanensét egy case struktúrában feldolgozza, és esetenként adatot küld a PC-re

**8. lépés:** Végül lebontja az első lépésben létesített adatcsatornákat a két eszköz között.

Egy küldött, valamint fogadott csomag mérete legalább 2 hosszúságú kell hogy legyen, tekintettel arra hogy minden csomag első két eleme a parancs és a küldött csomag hossza.

### 6.2.2. A protokoll használata

Az új protokoll használatára különböző metódusokat hoztam létre. Ezek segítségével lehetséges verziószámot lekérni, LED-eket ki-bekapcsolni, stb. Eme metódusok sorra a következők:

- **void Leds(void);**

A Leds() függvénnyel a demonstrációs lapon lévő harmadik és negyedik LED-et lehet ki-bekapcsolni.

- **void Reset(void);**

Ezzel a paranccsal a demonstrációs lapot lehet reset-elni, hatását tekintve ekvivalens a lapon lévő reset gombbal.

- **void Csomag(void);**

A mikrokontrollerről lehet egész típusú számokat, karaktereket és sztringet fogadni.

- **void Kuldes(void);**

A konzolról bekér a számítógép két tetszőleges számot, ezt követően átküldi a mikrokontrollernek USB csomag formátumban feldolgozásra, az megnöveli az értéküket eggyel, végül visszajuttatja a számítógépnek és megjelennek a már megnövelt értékek a konzolon.

- **void ReadPot(void);**

Ezzel a függvénnyel le tudom olvasni a demonstrációs lapon lévő potenciométer aktuális értékeit.

- **void SetPot(void);**

A potenciométer aktuális két értéke két bufferbe olvassa be a demonstrációs lap, ezzel a paranccsal ennek a két buffernek az értékeit tudom tetszőlegesen a konzolról átállítani.

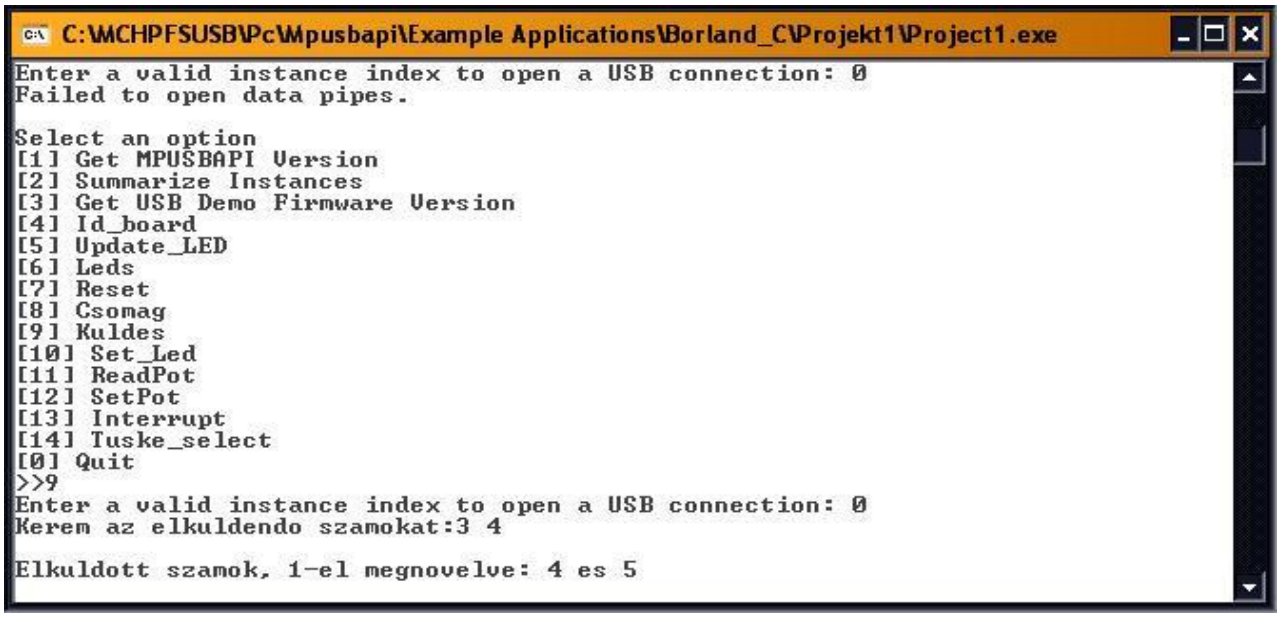
- **void Interrupt(void);**

Az interrupt eljárást tudom engedélyezni, illetve letiltani.

- **void Tuske_select(void);**

Hasonló a Leds() függvényhez, lényegében ugyan azt teszi, ugyanis ha a LED-eket ki-bekapcsolom, akkor a felillesztett taktilis kijelző tüskéit ugyan így ki-bekapcsolom. Az egyetlen különbség, a hármas LED fényerejét illetve első tüske rezgés intenzitását tudom szabályozni.

Ezen metódusok közül a protokoll pontosabb használatát a Kuldes() metódus segítségével mutatom be. A számítógép oldalon ez a metódus bekér két tetszőleges egész típusú számot a konzolról, elküldi a mikrokontrollernek USB csomag formátumban feldolgozásra, az megnöveli az értéküket eggyel, majd visszaküldi és a PC-n kiíratja a már megnövelt értékeket, ahogy ez az 12. ábrán látható.




**20. ábra: a Kuldes() metodus működése**

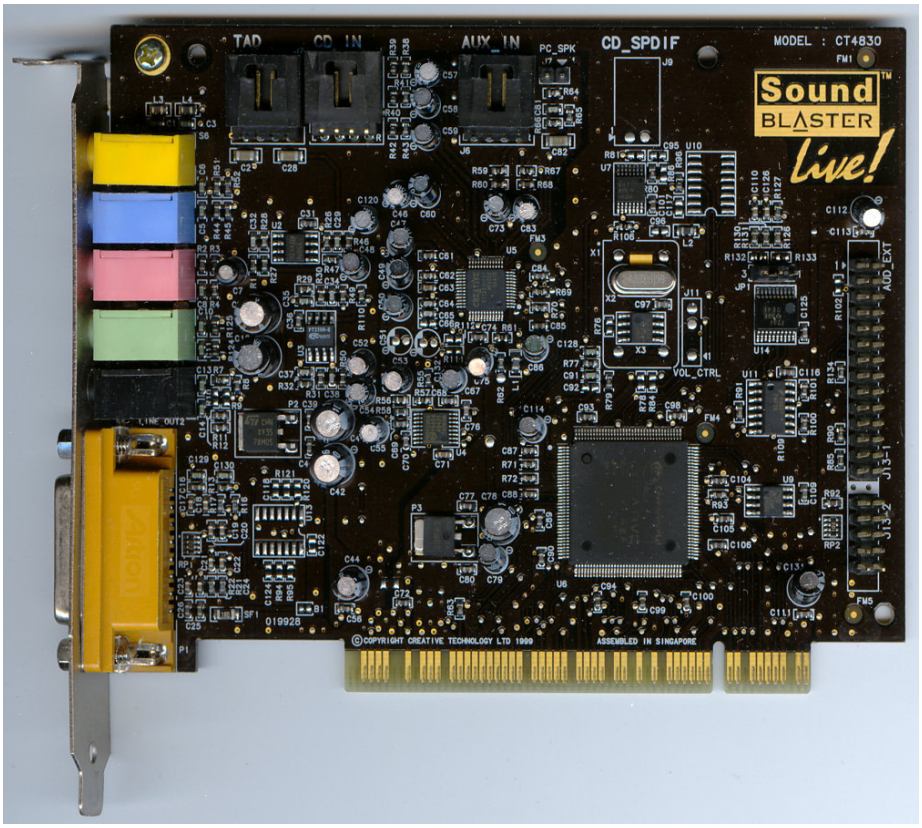
A mellékletben lehet megtalálni a Kuldes() metódos programrészletét. A mellékletben látható, hogy a bekért számokat a SendReceivePacket() wrapper függvény elküldi, majd fogadja mégpedig az MPUSBAPI hét alapfüggvénye közül az USBRead() és USBWrite() függvényekkel. A mellékletben látható még továbbiakban a mikrokontrolleren futó kód egy részlete. Itt látható hogy a kontroller feldolgozza a küldött adatokat, vagyis megnöveli az értéküket eggyel. Ebből már az is tisztán látható, hogy a kommunikáció során küldött adatokat fel is tudja dolgozni a mikrokontroller.

# 7. USB eszköz fejlesztése

A demonstációs lap USB csatlakozóval ellátott, így lehetőségem nyílt egy USB protokoll megteremtésére. Ennek a csatlakozónak a meglétével kilátásba kerül az is, hogy a számítógép automatikusan felismerje a fejlesztői eszközt. Ezt kihasználva megoldható, hogy a segédeszközt számítógéphez csatlakoztatva külső beavatkozás nélkül együttműködni legyen azzal képes.

## *7.1. A hangkártya*

A hangkártya a számítógép egy bővítőkártyája. Csatlakozóin keresztül hangokat fogad, és ad ki, illetve processzorával feldolgozza azokat. Felhasználói területei jellegzetesen a multimédiás alkalmazások és szórakozás például a zenehallgatás. Ez az eszköz általában a manapság vásárolható számítógépek alaplapjára van integrálva, régebben külön vásárolható és beszerelhető volt. Azonban a nagyobb teljesítményű hangkártyákat még mindig külön vásárolják a jobb minőségre vágyó professzionális felhasználók.




**21. ábra: egy hagyományos hangkártya**


**22. ábra: USB csatlakozóval ellátott hangkártya**

### *7.1.1. A hangkártya általános felépítése*

A hangkártya főbb részei a csatlakozók, ami lehet ki- vagy bemeneti is és a hangchip.

A hangchip magában foglal egy digitális-analóg konvertert, ami a számítógépen digitálisan tárolt fájlokat átalakítja és újra *hallgatható*, analóg formába hozza. A már felhasználó által is értelmezhető analóg jel egy jack-csatlakozóhoz megy, amihez legtöbbször hangszórót, fej- vagy fülhallgatót, vagy erősítőt csatlakoztatnak. Magasabb színvonalú hangkártyákon több hangchip is található, ezáltal a különböző feladatok megoszlanak és így a real-time hanghatások kiszámítására is lehetőség nyílik.

A hangkártya különböző csatlakozóit a megkülönböztethetőség érdekében eltérő színekkel jelölik. A Microsoft PC 99 szabványa szerint ezek a következők:

| Szín | | Funkció |
|---|---|---|
| | rózsaszín | Analóg mikrofon bemenet. |
| | világoskék | Analóg bemenet. |
| | világoszöld | Analóg kimenet a fő hangszóróknak vagy a fejhallgatónak. |
| | fekete | Analóg kimenet a hátsó hangszóróknak (4.0 vagy több esetén). |
| | ezüst | Analóg kimenet az oldalsó hangszóróknak (7.1 esetén). |
| | narancs | S/PDIF digitális kimenet (néha analóg kimenetként a mélynyomó és a középső hangszóró csatlakozik rá) |

**1. táblázat: PC99 szabvány szerinti színkódok**

Az első ábrán egy hagyományos hangkártya látható, melyen megfigyelhető a PC 99 szabvány szerinti csatlakozó színezések. A második ábrán egy USB csatlakozóval ellátott egyszerű hangkártyát lehet felismerni, amely jack-csatlakozóként mindössze egy darab mikrofonbemenetet és egy darab hangszóró kimenetet foglal magában.

### *7.1.2. Miért jó a hangkártya?*

A hangkártya a bemeneti csatlakozóján egy mikrofon jelét fogadja, feldolgozza, és kimenetelként hangszórón keresztül adja ki a hangjeleket. A segédeszköz esetén a hangszóró helyett most vibrotaktilis eszközt használok a célkitűzés érdekében. A szembetűnő hasonlóság miatt döntöttem a hangkártya vezérlés mellett. Ahogy a második kép mutatja kereskedelmi forgalomban is létezik már olyan hangkártya, amely a hagyományos hangkártyáktól eltérően azon túl hogy esetlegesen USB csatlakozóval ellátott kizárólag USB csatlakozóval képes csatlakozni a számítógéphez. Ezért feladatul tűztem ki, hogy az eddigi munkáim során használt demonstrációs lapot, mikrokontrollert felismertethessem a PC-vel mint hangkártyát.

## *7.2. USB történelem*

Az Universal Serial Bus (USB, univerzális soros busz) egy napjainkban szerfölött elterjedt számítógépes csatlakozófajta. Legjelentősebb előnye, hogy teljes körűen a Plug and Play módszert veszi igénybe. Ez gyakorlatilag azt jelenti, hogy szükségtelen minden egyes csatlakoztatáskor újra indítani a számítógépet. Valamennyi korszerű operációs rendszer 2001 óta támogatja, és azonos felépítésű, akárcsak PC úgymint Apple Macintosh számítógépen.

## *7.3. USB verzió, busz sebességek*

Az USB felépítését az USB Implementers Forum (USB-IF) szabványosította, amely egy ipari szabványokat kibocsátó testület. Jelenleg három fajta adatátviteli sebesség létezik az USB szabványokban: az 1,5 Mbit/s adatátviteli sebességre képes Low Speed, a 12 Mbit/s

sebességű Full Speed és a 480 Mbit/s átvitelétere is képes High Speed. Az 1996-ban kiadott USB 1.0 már képes volt a Low Speed mellett a Full Speed-re is, a 2.0-ás változat, pedig a High Speed-re is. 2007 szeptemberében már bemutatták az USB 3.0 prototípusát is, amely 4,8 Gbit/s sávszélességre tervezték és valószínűleg optikai kábeleket fog igényelni.

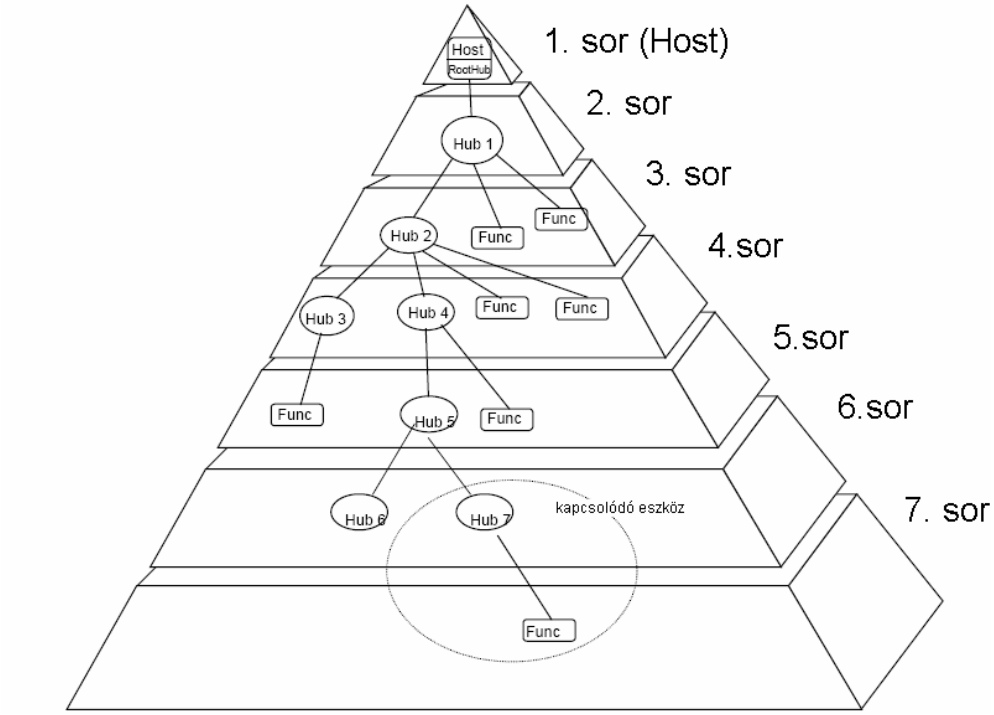
Az általam kialakított konstrukciót alkotó mikrokontroller képes a Full Speed adatátvitelre, valamint az USB 2.0 szabványnak felel meg. Ezek a jellemzők a feladatot tekintve kielégítők, sőt szükségesek.

## *7.4. USB terminológia*

Ebben az alfejezetben az USB 2.0 specifikációban használt lényeges elgondolásokat tekintem át, melyek betekintést nyújtanak például a Plug and Play működés hátterébe. Ezen alapfogalmak megértése és konzisztens használata elengedhetetlen egy USB eszköz fejlesztése közbeni dokumentáláshoz.

### 7.4.1. Busz topológia

Az USB eszközök az USB host segítségével kapcsolódnak a számítógéphez. A fizikai USB kapcsolat egy emeletes csillag topológia. A hub mindig a csillag középpontjában helyezkedik el. Minden egyes vezeték szegmentum egy pont-pont összeköttetés a host és hub között, vagy hub és hub között, vagy hub és funkció között. Maximálisan hét szintből, sorból állhat a topológia beleértve a host-ot is. A busz topológiát a 15. ábra mutatja.




**23. ábra: busz topológia**

### 7.4.2. Host

A host annak alapvető feltétele, hogy egy eszközre lehessen egy USB eszközt csatlakoztatni, magába foglalja a szükséges hardver platformot és az operációs rendszerrel való kommunikációhoz szükséges szoftvert is. Mindösszesen csak egy host van minden USB rendszerben. A gazdagép USB interfészét úgy nevezik hogy Host Controller. A Host Controller hardver, szoftver vagy firmware kombinációjaként van megvalósítva. A gyökér hub bele van építve a host rendszerébe.

### 7.4.3. Hub

A hub-ok (elosztófejek) kulcselemei az USB plug-and-play architektúrájának. A host egyszerű összekapcsolhatóság perspektíváját nyújtja a felhasználó számára, és viszonylagos robosztusságot, alacsony költséget és a használatban alacsony komplexitást biztosít.

Ezek az elosztófejek tartalmaznak egy darab felfelé irányuló csatlakozást és egy vagy több lefelé irányuló csatlakozási pontot. Az adatforgalmat mindkét irányba biztosítják.

#### 7.4.4. Funkció

A funkció egy eszköz, ami képes ellátni a működéshez szükséges tulajdonságokkal az USB eszközöket. Például egér, hangszóró, pendrive. Képes adatokat és vezérlő utasításokat küldeni vagy fogadni, konfigurációs információkat tartalmaz. Egy fizikai eszköz alkalmas lehet több funkcióra is. Egy kapcsolódó eszköznek tartalmaznia kell egy hub-ot és egy funkciót.

Tipikus funkciók:

-Human Interface Device (HID), pl.: egér, billentyűzet

-Communication Device Class (CDC), pl.: modemek

-Mass Storage Class, pl.: pendrive

### *7.5. VID és PID: Hogyan lép kapcsolatba a Windows egy USB eszközzel?*

Minden USB eszköznek van két kódja, aminek segítségével meg lehet különböztetni a többi USB eszköztől. Ezek a VID (Vendor ID) a gyártó céget azonosító egy tulajdonképpeni 16 bites kód, például a Microchip cég esetén ez 0x04d8 és a PID (Product ID) pedig a terméket azonosítja szintén 16 bites kóddal, a 0x000a jelzés a Microchip RS-232 csatlakozót szimuláló termékét jelenti. Ezeket az azonosító számokat a hardver belsejében kell tárolni, hogy megfeleljen az USB előírásoknak. A Vendor ID-k a gyártó cég tulajdonában vannak, és az USB-IF társaságtól kézvényezik, mert az tanúsítja, osztja ki ezeket az egyedi azonosító kódokat. A PID lehet bármi, amit a gyártó cég kíván, de tanácsos egyedi PID-et használni minden kereskedelmi célra szánt konstrukcióhoz. Továbbá az USB eszközök adott VID/PID kombinációk segítségével könnyen lehet ellátni azonosító számmal, ezáltal katalogizálni őket. Az operációs rendszerek megengedik, hogy különböző modellek azonos azonosító számokkal rendelkezzenek, ha a konfigurációs beállítások megfelelnek. Viszont az azonosítók használatát erősen javasolják.

### 7.6. Hól tárolják a VID/PID kódokat?

Egy USB eszköz rendszerint egynél több VID/PID párt tartalmaz. Van egyfajta, az említett azonosító párból alapértelmezés szerint a vezérlőegység bootkódjában. A bootkód egy lefordított program, amit egyéb programozó eszköz nélkül az eszközbe lehet írni. Az alapértelmezett VID mindig a vezérlőegység tényleges gyártót azonosítja. Az alapértelmezett PID legtöbbször a vezérlő eszköz sorszámának (device number) utolsó négy számjegyével egyezik meg. Ez az alapértelmezett mód abban az esetben érdekes, ha mondjuk, hogy egy nyomtatót gyártó cég által gyártott nyomtatók vezérlőegységét egy másik cég gyártja és szállítja be. Az alapértelmezett VID/PID kódok miatt szükségessé válhat minden termékben egy EEPROM (elektromos úton törölhető programozható csak olvasható memória) az egyedi VID/PID tárolása érdekében. Ez a második VID/PID, ami jelen van minden ilyen jellegű rendszerben. Ezek a rendszerek az eszköz belsejében képesek tárolni a VID/PID párt úgymint hub descriptor-ként az EEPROM fejlécben vagy firmware-en (memóriában tárolt szoftver) keresztül lehet beállítani az EEPROM-ban. Ha ezt a második módot használják akár hub descriptor-ként, akár firmware-ként akkor a bootkód közvetlenül ezeket használja, és az alapértelmezett vezérlőegység kódokat felülírja és jellemzik az USB eszközt. Ha az EEPROM alapú firware-t használják, akkor mindaddig hatástalanok, amíg a firmware végre nem hajtódik. A harmadik VID/PID pár akkor lehetséges, ha az AppLoader kezelő program tölt le firmware-t az USB vezérlőegysége számára, hogysem az EEPROM tárolja. Ebben az esetben a firmware felül kell hogy írja az addig érvényes értékeket az újakkal, hogy az eszköz egy új vezérlő programmal tudjon működni. Ha az eszköz letölti a firmware-t a PC-ről, akkor már egy beépített vezérlő programot használ. Így az eszköz már a végső vezérlő programmal áll helyesen kapcsolatban és használja azt, ezáltal biztosítva annak a célnak megfelelő működést.

### 7.7. Mi történik, ha az eszközt a buszra illesztik?

Ez elkövetkező pontokban sorba veszem az összes VID/PID-el összefüggő tevékenységet, ami előfordul csatlakoztatás után:

1. Az USB vezérlő bootkódja átadja az alapértelmezett VID/PID értékeket, és innentől ezek tekinthetőek érvényesnek. Ha a későbbiekben nem írja felül egy másik érték, akkor ezek az alapértelmezett értékek jelentkeznek az USB host-on.

2. Ha egy érvényes fejléc van az EEPROM-ban, és ha egy eszköz descriptor található, akkor a descriptor-ban található VID/PID írja felül az eddig érvényes értékeket a vezérlőben.

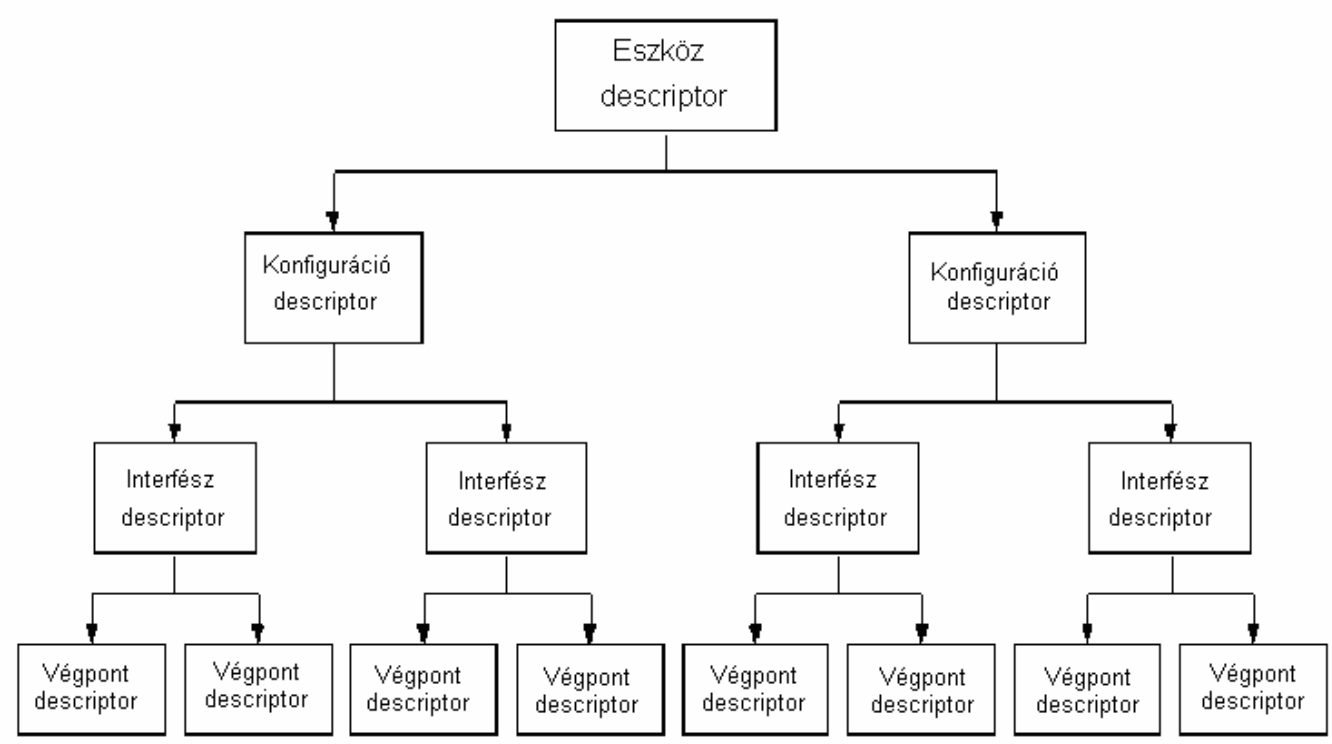
3. A firmware képes kicserélni az ez ideig érvényes VID/PID kódokat a vezérlőben és aktiválódik.

4. A host számba veszi az USB eszközt, és az érvényes VID/PID jelentkezik, majd összekapcsolódik egy vezérlő programmal.

5. Ha a vezérlő program valóban összekapcsolódott az eszközzel, akkor végrehajtja ezt a programot. Ha a vezérlő program az AppLoader, a firmware felül kell hogy írja az aktív PID azonosítót és egy szétkapcsolás – újracsatlakozást okoz, hogy a hub egy másik vezérlő programmal kapcsolódhasson össze.

### *7.8. USB descriptor-ok*

Minden USB eszköznek van egy descriptor hierarchiája, amely leírja a host számára hogy mi is ez az eszköz, ki gyártotta, melyik USB verziót támogatja, hányféle képen lehet konfigurálni, a végpontok száma mennyi és milyen a végpontok típusa, stb. A végpontok címezhető részei az USB eszközöknek, valamint a host és eszköz közötti kommunikáció információ forrásai és nyelői. Ezeket a rendelkezéseket az USB specifikáció írja elő. A descriptor-ok közül néhány külön-külön a fejlécben tárolható, a többit a firmware-ben lehet kezelni.




**24. ábra: descriptor hierarchia**

A leggyakoribb descriptor-ok a következők:

- eszköz descriptor
- konfiguráció descriptor
- interfész descriptor
- sztring descriptor
- végpont descriptor

Az USB eszközöknek csakis egy eszköz descriptor-a lehet. Az eszköz descriptor általános információkat ír le az USB eszközről, jelentősége abban nyilvánul meg, hogy olyan információkat tartalmaz, melyek elárulják hogy az eszköz melyik USB változatnak tesz eleget. Ezen kívül PID és VID azonosítókat használ a helyes vezérlő betöltéséhez, és az eszköz lehetséges konfigurációinak a számát tartalmazza. A konfigurációk száma jelzi a descriptor hierarchia ágainak számát. A descriptor-ok használata megengedi az egyéni konfigurációk jellemzőinek a tömör tárolását. Ugyanis egy adott konfiguráció talán újrahasználja a descriptor-okat vagy a descriptor-ok részeit más konfigurációkból, amiknek

ugyan azok a jellemzőik vannak. Ezen a módon a descriptor-ok hasonlítanak egyéni adat rekordra egy relációs adatbázisban. A descriptor-ok tartalmaznak hivatkozásokat sztringekre, amik az ember szára is olvasható formában vannak. A sztring descriptor-ok tárolása tetszés szerint elhagyható, a hivatkozás viszont mindenképp kötelező. Ha egy eszköz nem támogatja a sztring descriptor-t, akkor a sztring hivatkozást nullára kell visszaállítani, ami azt jelenti hogy a sztring descriptor elérhetetlen ebben az esetben.

A konfigurációs descriptor egy sajátságos eszköz konfiguráció információit írja le. Olyan értékeket ad meg, mint például az adott konfigurációnak szükséges áramellátás, a konfiguráció a buszról kapja-e a tápfeszültséget vagy máshonnan és a kapcsolódó interfészek számát. Ha egy eszköz csatlakozik, a host kiolvassa az eszköz descriptor-okat és ez alapján el tudja dönteni, hogy melyik konfigurációt engedélyezze. Egyszerre csakis egy konfiguráció lehet érvényes. Léteznek olyan konfigurációk, amelyek képesek különbözőfajta csatlakozók, különbözőfajta áramellátásával is helyesen működni. Egy adott konfigurációt lehet különféle áramellátással működtetni, ekkor viszont minden tevékenység leáll.

Az interfész descriptor egy sajátságos interfészt ír le a konfiguráción belül, tekinthetjük úgymint egy fejlécet, vagy úgymint a végpontok egy funkcionális csoportosítását, amik együttesen képzik egy adott jellemzőjét az eszköznek. Például egy több funkciós fax-scanner-nyomtató eszköz esetén egy interfész descriptor képes jellemezni a faxot, egy másik a scannert, egy harmadik pedig a nyomtatót. Egy időben több interfész descriptor is lehet érvényben, ellentétben a konfiguráció descriptor esetével, amikor csakis egy. Egy eszköz esetén lehetséges egy vagy több érvényes interfész descriptor is. Az interfész descriptor mindig egy konfigurációs descriptor részeként jelenik meg. Egy interfész tartalmazhat alternatív beállításokat is, ami követi a végpontok jellemzőit. Az alapértelmezett beállítás egy interfész számára mindig az alternatív beállítás vagy csupa nulla.

A sztring descriptor-ok opcionálisak. A sztring descriptor UNICODE kódolást használ, ezáltal több nyelvet is támogathat egy USB eszköz.

Minden végpont egy interfészhez kapcsolódik és descriptor-a van. Ez a descriptor tárolja a host számára a szükséges információkat a végpontról.

## 7.9. Eredmények, az USB eszközzel

Az USB eszköz fejlesztése folyamán elért eredményeket taglalom az elkövetkező pár oldalon.

### 7.9.1. A szükséges eszközök

Az általam felhasznált eszközök a Microchip cég által gyártott Demonstrációs lap, a PIC18F4550 mikrokontrollerrel, a Microchip MPLAB 7.62 fejlesztői környezet és a Microchip C18 Student Edition fordító. Ezeket a hardvereket és szoftvereket az Önálló laboratórium című tárgy keretében már használtam és ismertettem, ezért most ezek részletezésétől eltekintek.

Szükségem volt ezeken kívül a Microchip CDC (Communication Device Class) firmware-re. Ez a példaprogram egy RS-232 csatlakozót szimulál USB csatlakozóra. Az RS-232 egy szabványosított soros csatlakozó. Ilyen jellegű alkalmazásokkal ebben a félévben foglalkozom először, ezért röviden ismertetem ennek használatát.

### 7.9.2. RS-232 emuláció

Többek között a gyorsan terjedő USB csatlakozóknak köszönhetően a soros port egyre inkább kiszorul a használatból, jórészt a laptopok esetén hagyják el ezt a fajta csatlakozót. Abban az esetben, ha egy olyan szoftvert kell használni, ami kizárólag soros csatlakozóval képes működni lehet szükségünk egy szimulációs programra, ami képes mondjuk USB buszos emuláció segítségével elhitetni a számítógéppel, hogy az említett soros csatlakozóval tud dolgozni. Ehhez az emulációhoz szükségtelen új vezérlő program, hiszen a Windows 98SE operációs rendszer és az újabb verziók eleve tartalmazzák azt. Következésképpen minden olyan program, ami soros csatlakozóra van megírva módosítás nélkül képes működni. A használatban legfeljebb annyi különbség jelentkezik, hogy megnövekszik az adatátviteli sebesség.

Ezt a példaprogramot alakítottam át, hogy a PIC18F4550 mikrokontrollerrel működni tudjon, valamint a példaprogramban eredetileg használt High-Tech cég által kifejlesztett fordítóprogramról áttértem az eddig használt C18 fordító ingyenes Student Edition változatára. A program így már valóban működött és sikeresen szimulálta az RS-232 működését. A 6. ábrán látható a szimuláció eredménye. A Windows felismeri az USB eszközt, és ezt feltünteti egy felugró buborékban, amiben kiírja az eszköz típusát.

Innentől kezdve megoldottnak tekinthető a feladat, hiszen ennek a szimulációs programnak a használatával elvárásaimnak megfelelően a számítógép már képes felismerni a segédeszközt és kommunikálni vele. A célkitűzésnek ezt a részét már teljesítettem. A következő lépés, hogy a számítógép valóban hangkártyának ismerje fel a segédeszközt.

### 7.9.3. Áttérés hangkártyára

Ahhoz hogy az operációs rendszer valóban hangkártyának ismerje fel az eszközt descriptor-okat használtam. Szükséges volt a megfelelő konfiguráció és eszköz descriptor-ok beállítása.

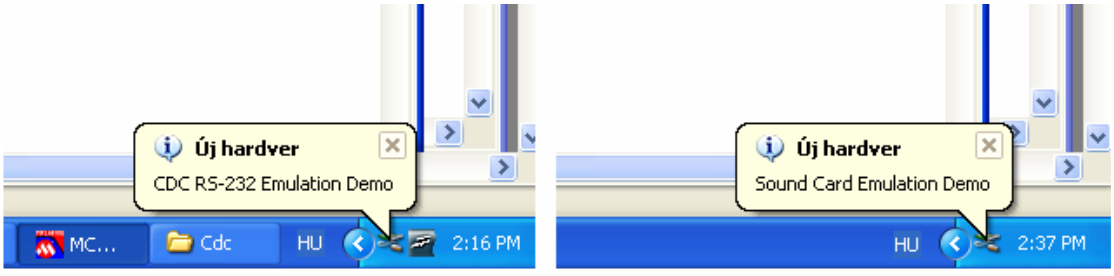
Megszabtam a konfigurációs descriptor segítségével az oszcillátor működést, az eszköz áramellátást és az USB átviteli sebességet Full Seed-re állítottam. Ezen kívül beállítottam minden olyan értéket, ami elengedhetetlen volt a működéshez. A konfigurációs beállítások egy programrészlete a függelékben látható. Működés közben az eszközből kiolvastam az MPLAB program segítségével a konfigurációs beállításokat és összevetettem az általam megadottakkal ellenőrzés képen. Ennek a beállításnak köszönhetően az eszköz azon felül, hogy hangkártyaként tünteti fel magát a Windows számára, a kommunikációja is annak megfelelő.

| Category | Setting |
|---|---|
| Full-Speed USB Clock Source Selection | Clock src from 96MHz PLL/2 |
| CPU System Clock Postscaler | [OSC1/OSC2 Src: /1][96MHz PLL Src: /2] |
| 96MHz PLL Prescaler | Divide by 5 (20MHz input) |
| Oscillator | HS: HS+PLL, USB-HS |
| Fail-Safe Clock Monitor Enable | Disabled |
| Internal External Switch Over Mode | Disabled |
| USB Voltage Regulator | Disabled |
| Power Up Timer | Enabled |
| Brown Out Detect | Disabled in hardware, SBOREN disabled |
| Brown Out Voltage | 2.0V |
| Watchdog Timer | Disabled-Controlled by SWDTEN bit |
| Watchdog Postscaler | 1:32768 |
| CCP2 Mux | RB3 |
| PortB A/D Enable | PORTB<4:0> configured as digital I/O on RESET |
| Low Power Timer1 Osc enable | Disabled |
| Master Clear Enable | MCLR Enabled,RE3 Disabled |
| Stack Overflow Reset | Enabled |

**25. ábra: konfigurációs beállítások**

Annak érdekében, hogy a számítógép csakugyan hangkártyának ismerje fel a demonstrációs lapot többek között a VID és PID kódokat használtam az eszköz descriptor-ban. Az eddig szereplő Microchip céget jelölő 0x04d8 VID kódot és a Microchip RS-232 csatlakozó szimulálót jelölő 0x000a értékeket átírtam egy hangkártyáról leolvasott megfelelő értékekre. A VID/PID kódok ilyen jellegű használata laboratóriumi körülmények között megengedett és az sem jelent problémát hogy a mikrokontroller gyártója és a hangkártya gyártója különbözik egymástól. Ebben az esetben azt a második típusú VID/PID párt használom, ami az alapértelmezett kódokat írja felül, amint ezt már a 3.6 fejezetben leírtam. Vagyis a tényleges gyártónak továbbra is a Microchip tekinthető.

Ember számára is olvasható formátumban feltüntettem a sztring descriptor használatával, hogy egy hangkártyát szimuláló programról van szó. Ennek a forráskódnak egy darabja a mellékletben látható. A sztring descriptor támogatja a UNICODE kódolást, ezért a magyar nyelvnek megfelelő ékezetes karaktereket használhattam.




**26. ábra: a demonstrációs lap, mint RS-232 szimulációs eszköz, és mint hangkártya**

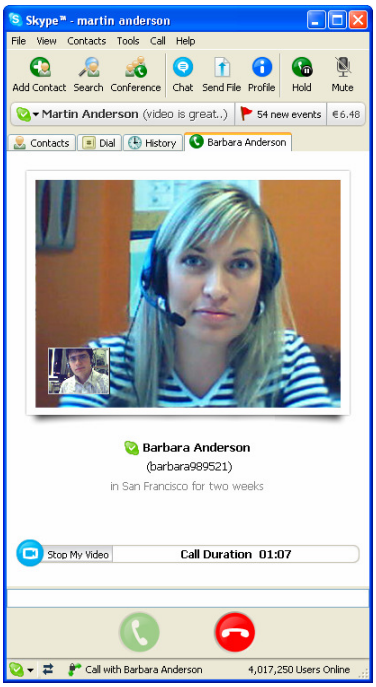
A hangkártyát szimuláló eszközt a Windows felismeri és felugró buborékban megjeleníti, ez látható a 18. ábra második felében.

### 7.9.4. Következtetések az USB eszközzel kapcsolatosan

Az eddigi mikrokontrollerre megírt programomat továbbfejlesztettem egy nagyon hasznos új tulajdonsággal. Ezen túl számítógéphez csatlakoztatva az operációs rendszer felismeri, és mint perifériát képes kezelni. Erre azért volt szükségem, hogy a segédeszköz tesztelhetőséghez nélkülözhetetlen hátteret tudjam biztosítani. Ekképpen a prototípus PC-re illesztve már könnyedén alkalmas a siketek által végzett teszteredmények révén szükségessé váló módosítások rugalmas megvalósítására. Vizsgálhatóvá vált a kijelzőt mennyire könnyű, vagy nehéz, mint mechanikai információt közlő eszközt megtanulni kezelni. Használatával felmérhető, hogy a siketek mennyire képesek alkalmazkodni a zöngésség fogalmához, amely számukra nyilvánvaló okok miatt nehezen értelmezhető. Valamint a különböző hangjel feldolgozások tesztelhetősége révén ki lehet választani a felhasználó számára könnyebben értelmezhető taktilis impulzus fajtákat.

### 7.9.5. További lehetséges felhasználási területek, PC-n, mobiltelefonon, PDA-n

Ezen kívül lehetőség nyílik arra is, hogy a fejlesztőn kívül a felhasználó is ki tudja használni az eszköz számítógéphez csatlakozási képességét. Ez azt jelenti, hogy az önálló működésre szánt segédeszközt PC-re csatlakoztatva is képes az igénybevevő halláskárosult hasznosítani. Például webkamera segítségével videóhívás közben a felhasználó a monitoron keresztül azon túl, hogy a szájról való olvasásra kell hogy hagyatkozzon, kihasználhatja az immáron már számítógépre csatlakoztatható segédeszköz előnyeit is.



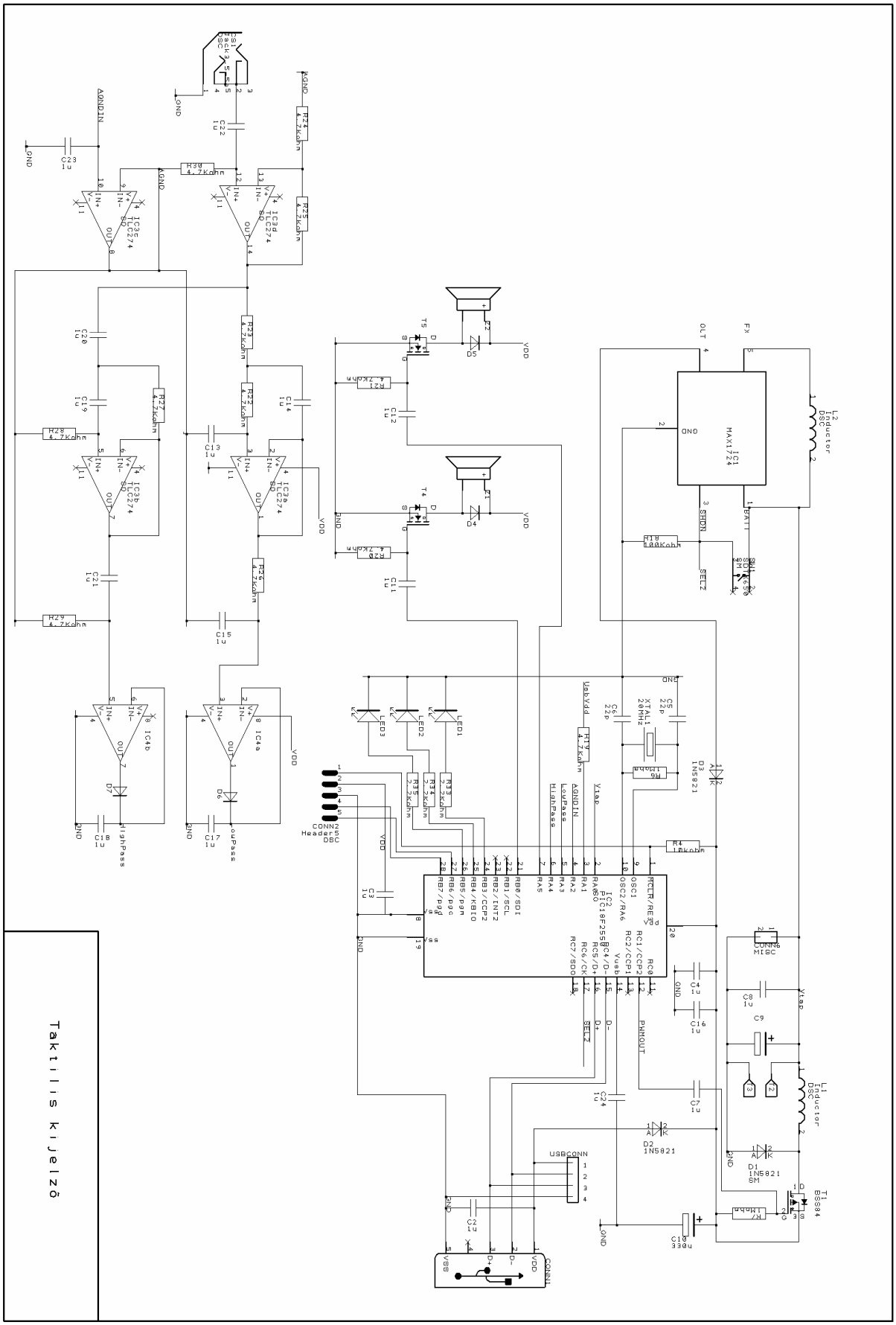

**27. ábra: videóhívás**

**28. ábra: mobiltelefon USB host-tal**

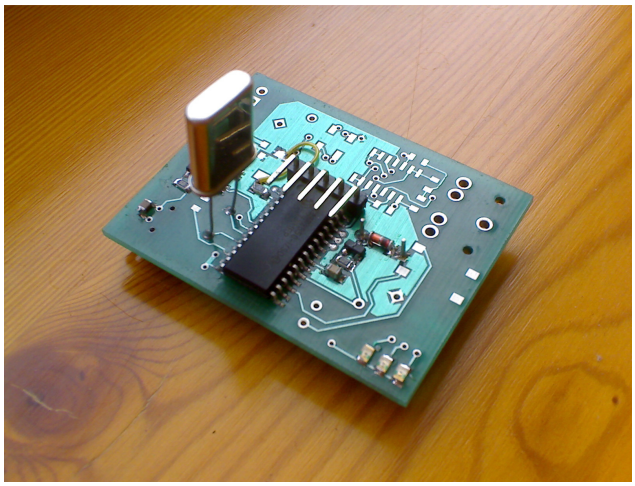
További lehetőség kínálkozik, hogy a segédeszközt össze lehessen kapcsolni mobiltelefonnal vagy PDA-val (Personal Digital Assistant, magyarul személyi digitális asszisztens). Minden USB kapcsolatban létezik egy host és egy slave egység. A host a felsőbbrendű eszköz, feladata hogy irányítsa a hozzá kapcsolódó slave-eket. Például egy számítógép és egy pendrive összekapcsolásakor a számítógép játssza a host szerepét, a pendrive pedig a slave szerepét. A PIC18F4550 és a legtöbb mobiltelefon egyedül slave funkciót képes betölteni, azaz lehetetlen más egyéb USB eszközt rácsatlakoztatni, csak a mikrokontrollert, mobiltelefont lehet egy host-ra legtöbbször számítógépre rákapcsolni. Azonban vannak olyan mobiltelefonok és még gyakrabban PDA-k, amelyek tartalmaznak host tulajdonságot is, főleg az újabb megjelenésű modellek. Ezekre az USB csatlakozókra a felhasználók kihangosítót, headset-et vagy PDA esetén akár pendrive-ot vagy egyéb háttértárolót szoktak csatlakoztatni. Ennek lehetőségét kihasználva, a taktilis kijelző számítógéphez csatlakoztatható képességén túlmenően, mobiltelefonhoz vagy akár PDA-hoz is tetszőlegesen kapcsolódhat.

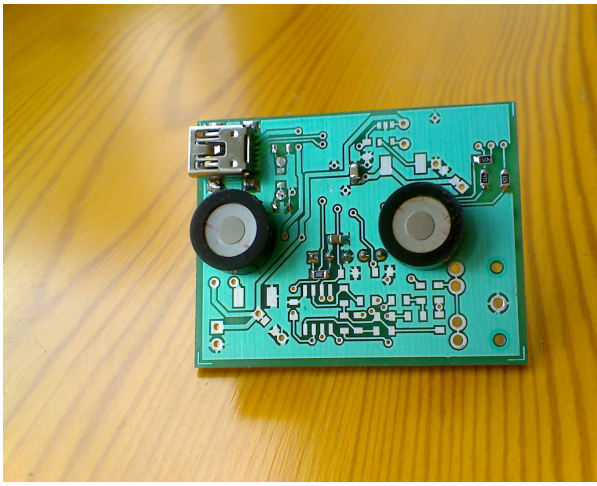
# 8. A fejlesztési minta

Az eddig használt hardver fő alkotó eleme a demonstrációs lap, amely egy széles körben elterjedt, sok fajta fejlesztésre alkalmas fejlesztői eszköz. Az általános felhasználás érdekében a legszükségesebbeken kívül számos egyéb, hasznos alkatrészt tartalmaz (ld.: 4. fejezet), ami a lap méretén is meglátszik. Egy egyéni célra fejlesztett eszköz esetében ezekből az alkotóelemekből szintén hasznosnak bizonyul néhány funkció, azonban nem szükséges minden egyes alkatrész. Esetünkben feleslegessé válik néhány LED, nyomógomb, csatlakozó és a hőmérő is. Ennek következtében a már meglévő konstrukció nyomán egy új felépítésű rendszert tervezése mellett döntöttem, ami nem tartalmazza az általános célokra létrehozott demonstrációs lap a taktilis segédeszköz funkció szempontjából szükségtelen részeit. Így a méretcsökkenés következtében egy kisebb, könnyebb és karóraszerűen hordozható struktúrát kaptam. A súlycsökkenés főleg a kisebb méretű stimulátornak köszönhető.




**29. ábra: a prototípus elrendezése**





**30. ábra: a fejlesztési minta**

A fejlesztés során áttértem egy másik mikrokontroller típusra, konkrétan a PIC18f4550 mikrokontrollerről a PIC18f2550 típusra váltottam, ami egy-egyszerűbb konstrukció, lényegében a számomra most mellőzhető részeket nem tartalmazza. További előny hogy a kétfajta típus ugyan abba a termékcsaládba tartozik, így a vezérlőprogram jelentősebb átalakítások nélkül ugyan olyan jól használható, megtartva az eddigi fejlesztéseket. A taktilis segédeszköz fejlesztési mintája elkészült, az eredmény a 30. ábrán látható. A célkitűzésnek megfelelő működés tesztelése még hátra van.

# 9. Értékelés, elemzés, továbbfejlesztési lehetőségek

Az előző fejezetben már említettem, hogy a siketek kommunikációját enyhítő taktilis segédeszköz fejlesztési mintája elkészült. A továbbiakban szükség van a korrekt működést alátámasztó tesztekre.

## *9.1. Teszteredmények*

A berendezés feladata hogy képes legyen két taxel kijelzésére. Ezt az elvárást egy-egy elektromágneses stimulátorral valósítottam meg. A kijelzők által gerjesztett jel jól érzékelhető, hiszen a legtisztábban érzékelhető 200 Hz-es ingerlést használok. A két kijelző távolsága megfelel annak az elvárásnak, hogy a két eltérő rezgésjel megkülönböztethető legyen egymástól. Elvárás a berendezéssel szemben hogy a két taxelt a közvetlenül elhangzó beszédjel zöngés és frikatív jellege határozza meg.

### 9.1.1. Zöngés hangok megkülönböztetése

A teszteléshez olyan szópárokat kerestem, amelyek a belső hangképző szervek rejtettsége miatt nem látható zöngés hangképzés szájról olvasva megkülönböztethetetlenek. Ilyen szópárok például a „baba-papa”, gép-kép, „létra-cékla”, „sziszeg-zizeg”, „állat-állad” és a „cica-lila”. A „baba-papa” szavaknál a „b” zöngés hang, viszont a „p” zöngétlen. A taktilis segédeszköz a pillanatnyi „b” beszédhangnál mechanikus jelet gerjeszt, a „p” hangnál viszont nem, vagyis az eszköz helyesen detektálta a zöngés hangot. A többi felsorolt szavaknál hasonló sikerrel jártam el. Megmutattam, hogy a taktilis segédeszköz a zöngés hangokat helyesen jelzi ki

### 9.1.2. Frikatív jelleg behatárolása

A frikatív hangok, vagyis réshangok a következők *v, z, zs, f, sz, s, h*. A jellegzetes szópárok a „hó-kő”, „varázs-darázs” és „sok-ok”. A „hó-kő” szavak példájában a „h” hangnál jelzett a berendezés, a „k” hangnál nem, ugyanis a „h” frikatív hang, a „k” nem. A többi példára is a célnak megfelelően működött a segédeszköz.

## 9.2. További lehetőségek

Az eszköz helyesen dönti el valós időben a beszédjel zöngés ill. frikatív jellegét. A berendezés által gerjesztett jel jól tapintható és a két kijelző jele kifogástalanul megkülönböztethető. Az eszköz használata azonban nem magától értetődő a felhasználók számára, hiszen hangjellemzők értelmezése siketek számára nem egyértelmű, használatát tanulni, gyakorolni kell. Kezdő felhasználók számára ebben a fejezetben használt szópárok használatával gyakorló feladatok és részletes használati utasítás készítését javaslom. Az egyszerre több tapintási információ értelmezése, a két különböző taktilis kijelző eltérő jeléből származóan gyakorlással szintén jól tanulható (ld. 2.7. fejezet).

További siketekkel végzett teszteredmények alapján a két kijelző más beszédjellemzők detektálására is könnyedén átalakítható, hiszen a berendezés egyéb átalakításokra alkalmas, esetleg egy harmadik kijelző beépítésére megfelelő.

# Összefoglalás

A feladatom volt hogy taktilis segédeszközt készítsek siketek számára, amely szájról leolvashatatlan beszédhang információkat közöl a felhasználóval, megkönnyítve ez által a siketek kommunikációját. Elemeztem a siketek szájról olvasási képességét. Ráébredtem, hogy még a gyakorlott szájról olvasók kommunikációja is nehézségekbe ütközhet, ugyanis szájról olvasva nem kapható meg a teljes beszédinformáció.

Az alapvető taktilis kijelző adottságok kijelöléséhez a neurobiológia tapintásérzékelés szakterületét tanulmányoztam. A két kijelző távolságát definiáltam olyan szemszög alapján, hogy kézfejen ill. csuklón jól megkülönböztethetőek legyenek a különböző jelek. A taktilis kijelző vibrálási frekvenciáját pontosan 200 Hz-en állapítottam meg, ugyanis ezt az ingerlési értéket lehet a legjobban észlelni. Szükséges volt azt a legkisebb elmozdulást meghatározni, amit az ember érzékelni képes. A tapintásérzékeléssel foglalkozó tudományág ismeretei alapján ezt az értéket 0,2 milliméter távolságban adtam meg. A szakirodalom megemlít vakok számára kifejlesztett taktilis kijelzővel ellátott rendszert, amely statikus jelek feldolgozására alkalmas.

Az eddig igénybe vett taktilis kijelző fajták taglalása után kiválasztottam azt a típust, amelyik egy gyors, dinamikus hangjel kijelzésére legalkalmasabbnak találtam. Az elvárásoknak az elektromágneses stimulátorok felelnek meg a legjobban.

Igazoltam a zöngés és frikatív hangok taktilis kijelzésének indokoltságát egy segédeszközön. Hiszen ezek azok a sajátosságok, amit szájról olvasva lehetetlen, azonban hangjellemzők alapján elektronikus berendezéssel könnyedén meg lehet különböztetni.

A kijelzők irányításához szükségem volt egy elektronikus vezérlő berendezésre, ami a taktilis kijelzőt meg tudja hajtani. A választásom a PIC18F4550 típusú mikrokontrollerre esett. A mikrokontroller vezérlőprogramját úgy terveztem meg, hogy magába foglalja a takarékos működés feltételeit. A gazdaságos üzemmód ahhoz szükséges, hogy a segédeszközt a felhasználó önállóan, hordozható formában, telepesen működtetve azt használni tudja.

A személyi számítógépről való vezérlés érdekében kidolgoztam az eszközhöz egy USB protokollt. Így már egyszerű parancsokon keresztül lehetséges a mikrokontroller vezérelte kijelzőket számítógépről is irányítani, például a kijelző tüskéit ki-bekapcsolni. Majd azt valósítottam meg, hogy a már meglévő rendszert felruházzam egy olyan tulajdonsággal, aminek segítségével a személyi számítógép automatikusan felismeri és kezelni tudja azt. A konstrukció mivel USB kapcsolatra alkalmas, ezért annak lehetőségét vizsgáltam meg, hogy ezzel a fajta csatlakozóval PC-re kapcsolódva az operációs rendszer

hogyan képes felismerni egy eszközt és vezérelni azt. Az USB 2.0 specifikáció tanulmányozása közben arra a felismerésre jutottam, hogy a mai korszerű operációs rendszerek, többek között a Windows az USB eszközöket támogatja, automatikusan azonosítani és vezérelni képes azokat. Következésképpen a segédeszköz további tökéletesítését, mint egy új USB eszköz fejlesztését folytattam tovább.

Taktilis kijelző, mint segédeszközt még nem egy mindennapos fogyasztási cikk, ezért egy hozzá hasonló viszont igencsak elterjedt eszközként, hangkártyaként ismertettem fel a számítógéppel. Így a PC már automatikusan képes azonosítani és kezelni a segédeszközt. Ezekkel a feltételekkel a konstrukció kettős szerepre alkalmas, egyrészt önálló telepes működésre, másrészt egy rugalmasan fejleszthető, számítógép segítségével könnyen tesztelhető rendszer. Az elvárásaimon felül a felhasználó az eszközt laptopjához, mobiltelefonjához vagy PDA-jához csatlakoztatva képes lesz, például videóhívások fogadása közben is kiaknázni a segédeszköz nyújtotta előnyöket.

A hordozhatóság érdekében egy kisebb eszközt kellett terveznem. A méretcsökkenést egy másik a PIC18F2550 típusú mikrokontroller alkalmazásával értem el, amiről lemaradnak a feladat céljából feleslegessé váló részek. Eredményéül egy karóraszerű viselésre alkalmas eszközt kaptam. Az eszköz tartalmaz egy 400 Hz-es alul-áteresztő szűrőt és egy 2,5 kHz-es felül-áteresztő szűrőt, ezek használatával detektálom a zöngés és frikatív hangokat.

Az eszköz segítségével használhatóságot igazoló teszteket végeztem. A berendezés gerjesztette mechanikai jelek jól érzékelhetőek, a két különböző kijelző jele jól különválik. Az emberi beszéd zöngés és frikatív hangjellemzőit az eszköz helyesen kijelzi.

# Köszönetnyilvánítás

# Irodalomjegyzék


[1.] Sekuler, R. – Blake, R.: Észlelés, Osiris, Budapest, 2000

[2.] Fonyó Attila: Az orvosi élettan tankönyve, Medicina, Budapest, 1999

[3.] Gordos - Takács: Digitális beszédfeldolgozás, Mûszaki könyvkiadó, Budapest 1983

[4.] „Hangkártya”, Wikipedia,
http://hu.wikipedia.org/wiki/Hangk%C3%A1rtya
[5.] „USB specifikáció”, USB Implementers Forum,
http://www.usb.org
[6.] „USB vezérlő források”, Microchip,
http://www.microchip.com/usb
[7.] „USB Complete”, Jan Axelson
http://www.Lvr.com
[8.] „Java USB API for Windows”,
http://www.steelbrothers.ch/jusb/
[9.] „Hardware & Drivers - USB ”, Apple Inc.,
http://developer.apple.com/devicedrivers/usb/
[10.] „Linux USB”,
http://www.linux-usb.org/
[11.] „American Foundation for the Blind”

http://www.afb.org/afbpress/pub.asp?DocID=aw040204

[12.] „Vakok és Gyengénlátók Borsod - Abaúj - Zemplén megyei Egyesülete”
http://web.t-online.hu/aurasoft/segedesz.htm

[13.] „Harvard BioRobotics Laboratory”
http://biorobotics.harvard.edu/research/tactile_display.html

[14.] „A Számítógép Hálózati Központ munkatársainak publikációi és konferenciaelőadásai az utóbbi években, A BRAILAB BESZÉLŐ SZÁMÍTÓGÉPCSALÁD”
http://www.kfki.hu/cnc/szhkpub/arato_kandidat4.rtf

[15.] „Wikipedia - Optacon”

http://en.wikipedia.org/wiki/Optacon

[16.] Jakub Kanis and Ludek Müller: Automatic Czech – Sign Speech Translation. In: Proceedings os 10th International Conference on TEXT, SPEECH and DIALOGUE TSD, Springer, Plzen, Czeh Republic, 2007

[17.] „Tihanyi Attila honlapja”

http://digitus.itk.ppke.hu/~tihanyia

# Mellékletek

### *A Kuldes() metódus*

```
      void Kuldes(void) {

          // First we need to open data pipes...
    DWORD selection;
    fflush(stdin);
    printf("Enter a valid instance index to open a USB connection:
");
    scanf("%d",&selection);

    myOutPipe = MPUSBOpen(selection,vid_pid,out_pipe,MP_WRITE,0);
    myInPipe = MPUSBOpen(selection,vid_pid,out_pipe,MP_READ,0);
    if(myOutPipe == INVALID_HANDLE_VALUE || myInPipe ==
INVALID_HANDLE_VALUE)
    {
        printf("Failed to open data pipes.\r\n");
        return;
    }//end if

    BYTE send_buf[64],receive_buf[64];
    DWORD RecvLength=4;

    DWORD szam1, szam2;
    szam1=0;
    printf("Kerem az elkuldendo szamokat:");
    scanf("%d %d",&szam1, &szam2);

    #define KULDES    0x43
    send_buf[0] = KULDES;      // Command
    send_buf[1] = 0x04;              // Expected length of the
result
    send_buf[2] = szam1;
    send_buf[3] = szam2;


if(SendReceivePacket(send_buf,4,receive_buf,&RecvLength,1000,1000)
== 1)
    {
        if(RecvLength == 4 && receive_buf[0] == KULDES &&
            receive_buf[1] == 0x04 /* && receive_buf[2] == 1 &&
receive_buf[3] == 2 */)
        {
            printf("\nElkuldott szamok, 1-el megnovelve: %d es
%d\r\n",receive_buf[2],
                    receive_buf[3]);

       //     printf("%s\n",&receive_buf[8]);
        }
    }
    else
    {    printf("USB Operation Failed\r\n");
        printf("RecvLenght: %lu\r\n",RecvLength); }
```

```
    // Let's close the data pipes since we have nothing left to do..
    MPUSBClose(myOutPipe);
    MPUSBClose(myInPipe);
    myOutPipe = myInPipe = INVALID_HANDLE_VALUE;

     }
```

***A mikrokontroller egy programkódrészlete***

```
void ServiceRequests(void)
{
    byte index;

    if(USBGenRead((byte*)&dataPacket,sizeof(dataPacket)))
    {
        counter = 0;
        switch(dataPacket.CMD)
        {
case KULDES:
                    dataPacket._byte[2]++;
              dataPacket._byte[3]++;
              counter=4;
              break;
}//end switch()
        if(counter != 0)
        {
            if(!mUSBGenTxIsBusy())
                USBGenWrite((byte*)&dataPacket,counter);
        }//end if
    }//end if
}//end ServiceRequests
```

***Konfigurációs beállítások***

```
…
#pragma config PLLDIV = 1
#pragma config FOSC = HSPLL_HS
#pragma config USBDIV = 2
#pragma config CPUDIV = OSC1_PLL2
#pragma config FCMEN = OFF
#pragma config IESO = OFF
#pragma config PWRT = ON
#pragma config BOR = OFF
#pragma config VREGEN = OFF
#pragma config WDT = OFF
#pragma config WDTPS = 32768
#pragma config MCLRE = ON
#pragma config LPT1OSC = OFF
…
```

### *Eszköz descriptor*

```
rom USB_DEV_DSC device_dsc=
{
    sizeof(USB_DEV_DSC),    // Size of this descriptor in
bytes
    DSC_DEV,                // DEVICE descriptor type
    0x0200,                 // USB Spec Release Number in BCD
format
    CDC_DEVICE,             // Class Code
    0x00,                   // Subclass code
    0x00,                   // Protocol code
    EP0_BUFF_SIZE,          // Max packet size for EP0, see
usbcfg.h
    0x1106,//0x04D8,///     // Vendor ID
    0x3059,//0x000A,//      // Product ID: CDC RS-232
Emulation Demo
    0x0000,                 // Device release number in BCD
format
    0x01,                   // Manufacturer string index
    0x02,                   // Product string index
    0x00,                   // Device serial number string
index
    0x01                    // Number of possible
configurations
};
```

### *Sztring descriptor*

```
rom struct{byte bLength;byte bDscType;word string[25];}sd001={
sizeof(sd001),DSC_STR,
'P', 'P', 'K', 'E', '-', 'I',' T', 'K', ':',
'M', 'é', 'r', 'n', 'ö', 'k', 'i', ' ', 'T', 'e', 'r', 'v',
'e', 'z', 'é', 's'};

rom struct{byte bLength;byte bDscType;word string[25];}sd002={
sizeof(sd002),DSC_STR,
'S', 'o', 'u', 'n', 'd', ' ', 'C', 'a', 'r', 'd',' ’,
'E', 'm', 'u', 'l', 'a', 't', 'i', 'o', 'n', ' ', 'D', 'e',
'm', 'o'};
```